HCLSoftware

HCL Z Data Tools
Customization Guide
Version 17.1.0



Note

Before using this document, read the general information under Notices on page dlxiv.

Edition Notice

This edition (published October 2026) applies to Version 17 Release 1 Modification Level 0 of HCL Z Data Tools (program number HCL19OP1220), and to all subsequent releases and modifications until otherwise indicated in new editions.

Contents

Note....ii
Edition Notice....iii
Summary of changes....xv
About this document....xvii
Overview of Z Data Tools....xvii
Who might use this document....xix
Syntax notation....xx
Part I. Customizing Z Data Tools....22
Chapter 1. Preparing to customize Z Data Tools....23
Checklist for installing and customizing Z Data Tools....23
Library names after you finish installing....24
Alternatives for making Z Data Tools available....25
Concatenating libraries to the LINKLIST....26
Modifying the TSO logon procedure....27
Planning for running Z Data Tools with or without APF-authorization....27
Alternatives for controlling Z Data Tools Base auditing....28
Chapter 2. Customizing the operating environment for Z Data Tools....30
License and enable Z Data Tools....30
Modifying the ISPF environment....30
Preparing Z Data Tools to run with LIBDEFs....31
Invocation of Z Data Tools using LIBDEFs....32
Adding Z Data Tools to the ISPF menu....36
Defining Z Data Tools in an ISPF command table....37
Avoiding conflicts between ZDT commands and other applications....38
Making Z Data Tools the default VSAM editor....38
Invoking Z Data Tools from ISPF 3.4 or a data set list....38
Implementing Z Data Tools edit models in ISPF Edit....39
Enabling Z Data Tools to work with certain products....39
Z Data Tools supported software environments....39
Enabling WebSphere® MQ support....41
Customizing for processing COBOL copybooks....42
Customizing for processing PL/I include books....46
Customizing for processing HLASM copybooks....49
Customizing to use DFSORT to improve Z Data Tools performance....49
Binding Db2® to use Z Data Tools object access method (OAM) functions....52
Disabling Websphere MQ feature by system name....52
Running Z Data Tools with APF-authorization....53
Authorizing the PL/I Compiler Library for batch processing....53
Troubleshooting problems with APF-authorization....53
Language Environment® considerations....54
Chapter 3. Customizing Z Data Tools....55
Changing the default options....55
Setting the default national language....55
Setting up Z Data Tools to use a template repository....55
Changing the JCL skeleton for batch mode....57
Customizing Z Data Tools batch return codes....60
Changing the print and display translation tables....62
Changing the translation tables in English....63
Changing the ASCII translation tables....63
Chapter 4. Customizing the Z Data Tools security environment....65
Setting up the security environment by using RACF® or an equivalent security product....65
Controlling access to Z Data Tools functions with SAF....68
Controlling fullpack access to DASD volumes....70
Controlling Bypass Label Processing (BLP)....73
Controlling the use of the COBOL compiler....76
Unprotected functions and profile names for protected functions....76
Setting up the security environment for IBM® MQ....81
MQ security examples....82
Chapter 5. Customizing Z Data Tools to write audit records to SMF....86
Using System Management Facilities (SMF) for audit logging....86
Chapter 6. Customizing the Z Data Tools audit facility for the Z Data Tools Base component....88
HFM0POPT-controlled auditing....89
Audit data set configuration....89
SAF-controlled auditing for Z Data Tools Base component....91
SAF-controlled auditing using SYS1.PARMLIB....91
SAF-controlled auditing without SYS1.PARMLIB....93
When a security product other than RACF® is in use....94
Implementing SAF-rule controlled auditing....94
Understanding how Z Data Tools uses SAF rules to control auditing....95

How Z Data Tools determines whether audit log records should be written....96
SAF rule examples....98
Controlling where Z Data Tools writes audit log records....99
Controlling auditing of Z Data Tools functions....100
Z Data Tools auditing FACILITY and XFACILIT class resource names....101
Chapter 7. Customizing Z Data Tools for national languages....104
Changing the print and display translation tables for languages other than English....104
Translating the message text....106
Providing a multicultural version of HFM0MENU....106
Translating the ISPF messages text....107
Translating the panel text....107
Using the translated messages and panels....108
Customizing for the Japanese national language....109
Modifying the Japanese translation tables....109
Changing the Japanese message text....109
Chapter 8. Customizing Z Data Tools Service Provider for z/OS® Connect....111
Installing Z Data Tools Service Provider for z/OS® Connect....111
Installing Z Data Tools Service Provider on z/OS®....112
Installing the service archive files....115
Installing the Build Toolkit plug-in....115
TLS for the Build Toolkit plug-in....116
Maintaining Z Data Tools Service Provider....116
Uninstalling Z Data Tools Service Provider....118
Chapter 9. Customizing Z Data Tools to use library management system libraries....120
Accessing source code in CA-Panvalet libraries....120
Other library management systems....121
Accessing source code in an LMS with SUBSYS interface....121
Chapter 10. Customizing Z Data Tools to use the Optim™ Data Privacy Provider API....122
Chapter 11. Customizing Z Data Tools to use an I/O exit....123
Exit protocol....124
Writing your exit....124
Exit control block data names....126
Using the I/O exit control block....133
Function codes and the flow of I/O exit processing....133
Information that should always be returned from the I/O Exit to Z Data Tools....134
Initialization call discussion....134
Termination call discussion....135
Open call discussion....135
Close call discussion....135
Read and Write call discussion....136
Installing your exit....136
Chapter 12. Customizing Z Data Tools to use a scrambling exit....138
Exit protocol....138
Writing your exit....138
Exit control block description....139
Using the scrambling exit control block....145
Function codes....145
Return codes....145
Return output field values....146
Installing your exit....146
Chapter 13. Verifying the customization of Z Data Tools....148
Verifying the Japanese component....150
Part II. Customizing Z Data Tools Db2® Component....152
Chapter 14. Preparing to customize ZDT/Db2....153
Checklist for installing and customizing ZDT/Db2....153
Alternatives for making ZDT/Db2 available....155
Concatenating libraries to the LINKLIST....155
Modifying the TSO logon procedure....155
Using LIBDEFs to allocate the ZDT/Db2 Libraries....155
Ensuring the Db2® libraries are available to ZDT/Db2....155
Planning for running ZDT/Db2 with APF-authorization....156
Improving ZDT/Db2 performance....156
Alternatives for controlling ZDT/Db2 auditing....157
Chapter 15. Customizing the operating environment for ZDT/Db2....159
Modifying the ISPF environment....159
Adding ZDT/Db2 to the ISPF menu....159
Defining ZDT/Db2 in an ISPF command table....159
Customizing the ZDT/Db2 Primary Option Menu....160
Granting access to the Db2® catalog (required)....161
Sample jobs to grant SELECT access on the Db2® catalog tables....163
Adding ZDT/Db2 to the Db2® Administration Launchpad....165
Adding commands to start ZDT/Db2 functions from the Db2® Administration Tool....166
Starting ZDT/Db2 functions from an external application....167
Chapter 16. Customizing ZDT/Db2....172
Binding Db2® (required)....172
Running multiple versions of ZDT/Db2....173
Defining all Db2® systems that ZDT/Db2 will access in HFM2POPT (required)....174

Examples of HFM2SSDM macros....175
Customizing the ZDT/Db2 options....178
Examples of HFM2POPI macros....178
Usage tips....180
Changing the default options....180
Setting the default national language....181
Changing the JCL skeletons for batch mode....181
Customizing to protect update functions in ZDT/Db2....183
Customizing ZDT/Db2 for use in production environments....184
Chapter 17. Customizing the audit facility for ZDT/Db2....189
HFM2POPT-controlled auditing....190
Audit data set configuration....191
SAF-controlled auditing for Z Data Tools Db2® component....192
SAF-controlled auditing using SYS1.PARMLIB....193
SAF-controlled auditing without SYS1.PARMLIB....195
Implementing SAF-rule controlled auditing....195
Understanding how ZDT/Db2 uses SAF rules to control auditing....196
How ZDT/Db2 determines whether audit log records should be written....197
Controlling where ZDT/Db2 writes audit log records....202
SAF rule examples....203
Controlling where ZDT/Db2 writes audit log records....203
Controlling auditing of update access to Db2® objects....204
Controlling auditing of read access to Db2® objects....205
Testing SAF-rule controlled auditing....206
ZDT/Db2 auditing FACILITY and XFACILIT class resource names....207
Chapter 18. Customizing ZDT/Db2 for national languages....211
Changing the print and display translation tables for languages other than English....211
Translating the message text....212
Providing a multicultural version of HFM2MENU....212
Translating the ISPF messages text....213
Translating the panel text....214
Using the translated messages and panels....214
Providing a multicultural version of HFM2DENU....215
Customizing for the Japanese national language....217
Changing the Japanese message text....217
Chapter 19. Verifying the customization of ZDT/Db2....218
Step 1. Define Db2® objects to be used during verification....218
Step 2. Run the HFM2CHCK sample job....222
Step 3. Take a copy of the ZDT/Db2 IVP tables....222
Step 4. Use the ZDT/Db2 editor (normal and related edit)....224
Step 5. Use the ZDT/Db2 Basic SELECT Prototyper....227
Step 6. Use the ZDT/Db2 Copy function....231
Step 7. Use the ZDT/Db2 Export function....233
Step 8. Use the ZDT/Db2 Import function....237
Step 9. Use the ZDT/Db2 Object List utility....239
Chapter 20. Re-installing ZDT/Db2 after migration of a Db2® system....242
Part III. Customizing Z Data Tools IMS™ Component....243
Chapter 21. Preparing to customize ZDT/IMS....244
Checklist for installing and customizing ZDT/IMS....244
Supported databases....245
Region types....246
DLI mode....246
BMP mode....246
PSB types....247
Dynamic PSBs....247
Static PSBs....247
IMS™ region controller parameters....248
Security....248
Controlling access to databases by ZDT/IMS functions....248
Controlling access to IMS™ subsystems and ZDT/IMS functions....249
Resource access security and AGN security considerations....249
RACF® PADS security considerations....250
IMS™ subsystem access....250
Avoiding resource contention....251
IMS™ management of ACBs....253
Templates....254
How static and dynamic templates are used....255
View and criteria set usage rules....256
Alternatives for making ZDT/IMS available....257
Concatenating libraries to the LINKLIST....258
Modifying the TSO logon procedure....258
Alternatives for controlling ZDT/IMS auditing....258
Chapter 22. Customizing the operating environment for ZDT/IMS....261
Modifying the ISPF environment....261
Adding ZDT/IMS to your ISPF menu....261
Defining ZDT/IMS in an ISPF command table....261
Customizing IMS™ to support the use of dynamic PSBs....262

Declaring the dynamic PSBs..... 262
Providing a DOPT ACBLIB data set.....263
Chapter 23. Customizing ZDT/IMS..... 264
Customizing the ZDT/IMS installation options module..... 264
Migration considerations..... 264
ZDT/IMS macro statements..... 265
Specifying the IMS™ subsystems..... 266
Specifying the IMS Group..... 266
Specifying the parameters passed to the IMS™ region controller..... 267
Controlling access to the subsystems..... 268
Specifying selected processing options..... 269
Specifying IMS™ and ZDT/IMS data sets..... 270
Specifying miscellaneous IMS™ subsystem details..... 271
Specifying AGNS..... 272
Setting the default national language..... 273
Examples of HFM1POPD, HFM1POPI and HFM1AGNT macros..... 273
Tailoring the job control skeletons..... 279
Customizing for DEDB randomizing modules..... 280
Chapter 24. Customizing the ZDT/IMS security environment..... 282
The Database Access Control facility..... 282
IMS™ subsystems and ZDT/IMS functions access control facility..... 284
Controlling access to the update or read-only functions..... 285
Controlling access to individual IMS™ subsystems by the update or read-only functions..... 286
Controlling access to individual functions..... 286
Controlling access to individual IMS™ subsystems by individual functions..... 287
What governs whether access is granted or denied..... 287
Important information for users of non-IBM security products..... 289
Customizing the ZDT/IMS security exit..... 289
Types of security exits..... 289
Invoking the security exit..... 290
Common Exit Parameters..... 290
Security Exit Parameters..... 291
Sample programs for a security exit..... 295
Chapter 25. Customizing the Z Data Tools audit facility for IMS™ component..... 297
HFM1POPT-controlled audit logging..... 298
Audit data set configuration..... 299
SAF-controlled auditing for Z Data Tools IMS™ component..... 300
SAF-controlled auditing using SYS1.PARMLIB..... 300
SAF-controlled auditing without SYS1.PARMLIB..... 303
Implementing SAF-rule controlled auditing..... 303
Understanding how SAF controls ZDT/IMS audit logging..... 304
Controlling where ZDT/IMS writes audit log records..... 305
Controlling whether an audit trail is created 307
Controlling if users can request an audit trail when one is not required (Edit function only)..... 308
Chapter 26. Customizing ZDT/IMS for national languages..... 310
Changing the print and display translation tables for languages other than English..... 310
Translating the message text..... 311
Providing a multicultural version of HFM1MENU..... 311
Translating the ISPF messages text..... 312
Translating the panel text..... 313
Using the translated messages and panels..... 313
Customizing for the Japanese national language..... 314
Changing the Japanese message text..... 314
Chapter 27. Verifying the customization of ZDT/IMS..... 315
Step 1. Build the sample IMS™ databases to be used during verification..... 315
Step 2. Start ZDT/IMS..... 315
Step 3. Display settings menu..... 316
Step 4. Display DLI mode settings menu..... 316
Step 5. Verify subsystem details..... 317
Step 6. Verify DLI mode parameters..... 318
Step 7. Verify DLI mode data sets..... 318
Step 8. Verify DLI mode options..... 319
Step 9. Verify DBD library..... 320
Step 10. Start the browse dialog..... 320
Step 11. Verify database data sets..... 321
Step 12. Database positioning..... 322
Step 13. Browse the database..... 323
Part IV. Customizing Z Data Tools CICS® Component... 324
Chapter 28. Preparing to access CICS resources from Z Data Tools 325
Checklist for installing and customizing Z Data Tools to access CICS resources..... 326
Chapter 29. Customizing the operating environment for ZDT/CICS..... 330
CICS® resource definitions for ZDT/CICS..... 330
Updating the CICS® startup procedures..... 330
Customizing for CICS® TCP/IP..... 330
IPv6 support..... 331
Providing OMVS segments for ZDT/CICS users..... 332
Accessing other Z Data Tools functions from the primary option menu..... 332

Providing CICS® resource definitions for ZDT/CICS on interconnected regions....332
Limiting the regions to which ZDT/CICS can connect....333
Setting up the External CICS interface (EXCI) access....333
Chapter 30. Customizing ZDT/CICS....336
Customizing the batch procedure....336
Modifying and submitting HFMCINST....337
Modifying and submitting HFM3INST and HFM3PRFD....337
Modifying and submitting HFM3PRDU....340
Changing the default options....341
HFM3POPI....342
Chapter 31. Customizing the ZDT/CICS security environment....345
Customizing to protect update functions in ZDT/CICS....345
Customizing to invoke other Z Data Tools components from ZDT/CICS....345
Chapter 32. CICS® security and ZDT/CICS....347
Introduction....347
CICS® security considerations for ZDT/CICS....347
Logon and user security....347
Resource security....348
Command security....348
Transaction security....348
Intercommunication security....348
Controlling ZDT/CICS processing....348
Chapter 33. Customizing the Z Data Tools audit facility for CICS® component....352
Alternatives for controlling Z Data Tools CICS® auditing....352
Audit logging under ZDT/CICS and CICS® logging...353
HFM3POPT-controlled auditing....355
Audit data set configuration....355
SAF-controlled auditing for Z Data Tools CICS® component....357
SAF-controlled auditing using SYS1.PARMLIB....357
SAF-controlled auditing without SYS1.PARMLIB....359
Implementing SAF-rule controlled auditing....360
Understanding how ZDT/CICS uses SAF rules to control auditing....360
Understanding SAF rule access levels....360
How ZDT/CICS determines whether audit log records should be written....361
Controlling where ZDT/CICS writes audit log records....364
SAF rule examples....364
Controlling where ZDT/CICS writes audit log records....364
Controlling auditing of ZDT/CICS functions....366
ZDT/CICS auditing FACILITY and XFACILIT class resource names....366
Chapter 34. Customizing ZDT/CICS for national languages....368
Changing the print and display translation tables for languages other than English....368
Setting the LANGUAGE option....368
Verifying that the CICS® terminal is DBCS-capable....369
Translating the ZDT/CICS logon messages....369
Providing a multicultural version of HFM3MENU....370
Translating the panel text....371
Using the translated messages and panels....371
Customizing for the Japanese national language....372
Changing the Japanese logon message text....372
Chapter 35. Verifying the customization of ZDT/CICS....373
Part V. Z Data Tools on VS Code....377
Chapter 36. Installation requirements....378
Part VI. Preparing for Z Data Tools Remote Services....379
Part VII. Customizing Z Common Components server...380
Chapter 37. Customizing the ZCC server....381
Z Common Components server configuration....381
Additional security considerations....381
Address space timeout....382
Appendix A. Z Data Tools options....383
Customizing miscellaneous options in HFM4POPT....383
ABENDCC....384
ALTCOB....384
ALTPLI....385
ASCII....385
AUDDATAC....386
AUDITHLQ....386
AUDITLOG....388
AUDMGMTC....389
AUDPQTY....389
AUDSQTY....389
AUDSTORC....389
AUDSUNIT....390
AUDUNIT....390
AUXDATAC....390
AUXDSN....390
AUXHLQ....391
AUXMDSN....392
AUXMGMTC....392
AUXSTORC....392
BDY....392

CCSID....393
COBDBCS....393
COBDPC....393
COBEXTND....393
COBMAXRTN....394
COBMCASE....394
COMMADEC....394
COMPLANG....395
CREPLACE*n*....395
CSYSLIB*nn*....396
CYLHD....396
DATAHDR....396
DSINFO....397
DSPINC....397
DSPMAX....397
DSPMIN....398
DSPNUM....398
DUMP....398
EDMAXVIRT....398
EDITCAPS....398
EOD....399
EXCITRAN....400
FMEDITOR....400
HEADERPG....400
HLDBCS....401
HLMAXRTN....401
HLNOALIGN....401
HSYSLIB*nn*....402
ISPFGLBL....402
ISPFPACK....402
JCL....403
LANGUAGE....403
LMS....404
LMSUBSYS....404
LOADLIB....405
MQREPHLQ....405
MSGUPPER....406
NOTRUNC....406
OPSCRAM....406
PAD....407
PAGESIZE....407
PAGESKIP....407
PDATAC....408
PLI31DIGIT....408
PLI63BIT....408
PLIGRAPHIC....408
PLIMAXRTN....409
PLIUNALIGN....409
PMGMTC....409
PRINTDSN....409
PRINTLEN....410
PRINTOUT....410
PRTCLASS....411
PRTDATAC....411
PRTDISP....411
PRTMGMTC....411
PRTPQTY....412
PRTSQTY....412
PRTSTORC....412
PRTSUNIT....412
PRTTRANS....413
PRTUNIT....413
PSTORC....413
PSYSLIB*nn*....413
PUNIT....414
RECLIMIT....414
RLS....414
SEC....415
SHOWCOPY....415
SMFNO....416
TAPELBL....416
TDATAC....417
TEMPHLQ....417
TERMTYPE....418
TMGMTC....418
TRACECLS....418
TRACEDSN....419
TRACELIM....419
TRACEOUT....419
TRCDATAC....419
TRCMGMTC....420
TRCPQTY....420
TRCSQTY....420
TRCSTORC....420
TRCSUNIT....420
TRCUNIT....421
TSTORC....421
TUNIT....421
USEIOX....422
VSAUTO....422
VIEWAUX....422
VSCHGAUTO....423
VSCHGFRQ....423
VSSAVE....423

WBLKSIZE ... 424
WIDEPRT ... 424
WLRECL ... 424
Appendix B. ZDT/Db2 options ... 426
HFM2SSDM ... 426
ATTACH ... 426
AUDIT ... 427
AUDITBROWSE ... 428
AUTH_ACCESS ... 428
AUTO_COMMIT ... 429
CCSIDWARNIGNORE ... 429
CPYCPYN ... 430
DB2ALIB ... 430
DB2CLIB ... 430
DB2ELIB ... 430
DB2LLIB ... 431
DB2MLIB ... 431
DB2PLIB ... 432
DB2PROC ... 432
DB2RLIB ... 432
DB2SLIB ... 433
DB2TLIB ... 433
DESC ... 433
DISPLAY ... 433
EDIT_MAX_ROWS ... 434
EDITOR_TIMEOUT ... 435
FORCE_WITH_UR ... 436
LDFDDLN ... 437
LOCATION ... 437
LOCATION_NICKNAME ... 437
LODINDN ... 438
OPTEVT1 ... 438
OPTEVT2 ... 438
OPTEVT3 ... 438
OPTEVT4 ... 439
PLAN ... 439
PLAN2 ... 439
PROD_EDIT ... 439
RBXWRKN ... 440
ROGUNLN ... 441
SLDJCL1 ... 441
SLDJCL2 ... 441
SLDJCL3 ... 441
SLDJCL4 ... 441
SSID ... 442
STMJCL1 ... 442
STMJCL2 ... 442
STMJCL3 ... 443
STMJCL4 ... 443
TABLE_LOCKING ... 443
TMPDDLN ... 444
TYPE ... 444
UNLPUNN ... 444
UNLUNLN ... 444
USER_SELECT_EDIT ... 445
HFM2POPI ... 446
CATOWNERCDRM ... 446
CATOWNER ... 448
CCSIDWARN ... 448
CONNECT ... 449
EDITCAPS ... 450
OP34MOD ... 451
LIST ... 452
SHOWDATAC ... 454
SHOWMGMTC ... 454
SHOWPQTY ... 454
SHOWSQTY ... 454
SHOWSTORC ... 454
SHOWSUNIT ... 455
SHOWUNIT ... 455
SSIDCMD1 ... 455
SSIDCMD2 ... 456
Appendix C. ZDT/IMS options ... 457
HFM1POPD and HFM1POPI macros ... 457
ACBLIB ... 458
ACBMGMT ... 458
ACBSHR ... 459
AUTOSAVE ... 459
BSDSHLQ ... 460
CATALIAS ... 460
CHGAFREQ ... 460
CHKPINTVL ... 461
COMPAT ... 461
DBDLIB*n* ... 462
DBRC ... 463
DESC ... 463
DFSDF ... 464
DFSRRC00 ... 464
DFSVSAMP ... 464
DYNACB ... 465
DYNALLOC ... 465
DYNPRFN ... 466
DYNPRFX ... 467
DYNPSB ... 467

DYNTPLT **468**
EDITFREQ **468**
GSGNAME **469**
IEBFREQ **469**
IMSAUDLG **470**
IMSBKO **471**
IMSGROUP=identifier **471**
IMSNBA **472**
IMSOBA **472**
IMSPLEX **473**
IRLM **473**
IRLMNAME **474**
LIST **474**
LKEYDATAC **475**
LKEYMGMTC **475**
LKEYPQTY **475**
LKEYSQTY **475**
LKEYSTORC **475**
LKEYSUNIT **476**
LKEYVOL*n* **476**
LOADFREQ **476**
LOCKMAX **477**
LOGDATAC **477**
LOGDSN **478**
LOGMGMTC **479**
LOGPQTY **479**
LOGSQTY **479**
LOGSTORC **480**
LOGSUNIT **480**
LOGUNIT **480**
LOGUSAGE **480**
MACLIB **481**
MAXGN **481**
PADS **482**
PARDLI **482**
PROCLIB **483**
PROCOPTB **483**
PROCOPTP **484**
PROCOPTX **484**
PROCOPTY **485**
PSBLIB*n* **485**
PSBTYPE **486**
PSBTYPES **486**
READONLY **487**
REGCATLG **487**
REGTYPES **488**
RESLIB*n* **488**
RKEYDATAC **489**
RKEYMGMTC **489**
RKEYPQTY **489**
RKEYSQTY **489**
RKEYSTORC **489**
RKEYSUNIT **489**
RKEYVOL*n* **490**
SDSNLOAD*n* **490**
SKELLIB **491**
SSID **491**
TIMEOUTI **491**
TMINAME **491**
TPLLIB*n* **492**
UACBLIB **492**
UAGNS **493**
UAUTOSAV **493**
UBUF **494**
UDBDLIB **494**
UDBRC **495**
UDFSVSMP **495**
UIEBFREQ **496**
UIEFRDER **496**
UIMSBKO **497**
UIMSNBA **497**
UIRLM **498**
ULOADFRQ **498**
ULOCKMAX **499**
ULOGDSN **500**
ULOGUSAG **500**
UMACLIB **501**
UPARDLI **501**
UPROCOPB **502**
UPROCOPP **502**
UPROCOPX **503**
UPROCOPY **504**
UPSBLIB **504**
UPSBTYPE **505**
URECON **505**
URESLIB **506**
URSR **506**
USEDDL **507**
UTPLLIB **508**
VCURUDT **508**
VCURULE **509**
VSMPMEM **510**
XDOPTLB **510**
XKEYDATAC **511**

XKEYMGMTC..... 512
XKEYPQTY..... 512
XKEYSQTY..... 512
XKEYSTORC..... 512
XKEYSUNIT..... 512
XKEYUNIT..... 513
HFM1AGNT macro..... 513
AGN..... 513
DESC..... 513
SSID..... 514
Appendix D. Z Data Tools options specified in PARMLIB members..... 515
Z Data Tools options specified in HFM0PARM..... 515
FMAUDIT..... 515
FMOPTMOD..... 516
Facilities for customizing the HFM0PARM definitions..... 516
Tags..... 517
Rules for specifying tags..... 517
Example of tag usage..... 517
Included members..... 519
Comments..... 520
Continuing specifications across multiple lines..... 520
ZDT/IMS options specified in HFM1PARM..... 521
FMAUDIT..... 521
FMOPTMOD..... 522
Facilities for customizing the HFM1PARM definitions..... 522
Tags..... 523
Rules for specifying tags..... 523
Example of tag usage..... 523
Included members..... 525
Comments..... 526
Continuing specifications across multiple lines..... 526
ZDT/Db2 options specified in HFM2PARM..... 526
FMAUDIT..... 527
FMOPTMOD..... 527
Facilities for customizing the HFM2PARM definitions..... 528
Tags..... 528
Rules for specifying tags..... 529
Example of tag usage..... 529
Included members..... 531
Comments..... 532
Continuing specifications across multiple lines..... 532
ZDT/CICS options specified in HFM3PARM..... 532
FMAUDIT..... 533
FMOPTMOD..... 533
Facilities for customizing the HFM3PARM definitions..... 534
Tags..... 534
Rules for specifying tags..... 535
Example of tag usage..... 535
Included members..... 537
Comments..... 538
Continuing specifications across multiple lines..... 538
Appendix E. The library management system exit..... 539
Coding the exit: the basics..... 539
The main program and first argument: the operation code..... 539
The return code..... 540
Initializing the exit..... 541
Tracing..... 542
Terminating the exit..... 542
Coding the exit: required services..... 543
Common argument and the name of the library 543
Validate the library..... 544
Get member records..... 544
Get member information (metadata)..... 546
Get display information..... 547
Supporting multiple library management systems..... 548
Writing the exit in HLASM..... 548
Appendix F. Z Data Tools audit records..... 549
Z Data Tools (Base) audit records..... 553
Z Data Tools (Base) audit data items..... 553
Z Data Tools (Base) audit record types..... 554
Data items specific to Z Data Tools (Base) audit records..... 555
ZDT/Db2 audit records..... 555
ZDT/Db2 audit data items..... 555
ZDT/Db2 audit record types..... 556
Data items specific to ZDT/Db2 audit records..... 557
ZDT/IMS audit records..... 558
ZDT/IMS audit data items..... 558
ZDT/IMS audit record types..... 560
Data items specific to ZDT/IMS audit records..... 560
ZDT/CICS audit records..... 561
ZDT/CICS audit data items..... 561
ZDT/CICS audit record types..... 562
Data items specific to ZDT/CICS audit records..... 562
Notices..... dlxiv

Programming interface information.....dlxvi
Index..... 567

Customization Guide

This publication provides information needed to plan for, customize, maintain, and diagnose problems with the Z Data Tools Base component, Db2® component (ZDT/DB2), IMS™ component (ZDT/IMS), and CICS® component (ZDT/CICS).

Summary of changes

These changes, and smaller corrections and additions, are indicated by a "|" change bar in the left margin of the page.

October 2026, changes for V17R1M0

This edition of the document provides information applicable to Z Data Tools Version 16 Release 1 Modification Level 4. Here are the major changes to this document from the previous release:

Installation requirements topic for the Eclipse plug-in and VS Code Extension has been updated.

- Added support for specifying an IMSGROUP to identify the active IMS subsystem in BMP mode. This allows ZDT/IMS to connect to an active IMS subsystem associated with a specified IMSGROUP. See Specifying the IMS Group on page 266
- Added the IMSGROUP parameter to the Table 47: Miscellaneous IMS subsystem details on page 271 to document the support identifying an IMS subsystem by its associated IMSGROUP.
- Added new topic IMSGROUP=identifier on page 471 parameter to document the support on the HFM1POPI macro to identify IMS subsystems that belong to an IMSGROUP in BMP mode.
- Added the `LIST` parameter to the ZDT/IMS installation options to control which IMS subsystems are included in the ZDT/IMS subsystem selection list. See topic LIST on page 474
- Updated the IMS and ZDT/IMS datasets table to document the `SDSNLOAD` dataset parameter and indicate that it cannot be overridden. See table Table 46: IMS and ZDT/IMS data sets on page 270.
- Added support for DFSESL DD in FM IMS, both for ISPF & Batch edit in BMP Mode. See topic SDSNLOADn on page 490.
- Added a new topic to document the use of compiler via security profiles. See topic Controlling the Use of the Compiler via Security Profiles.
- Updated `SHOWPLI` command section to describe the output messages. See topic Customizing for processing PL/I include books on page 46.
- Added documentation for the DB2ALIB on page 430 option, including its syntax, purpose, and usage requirements for supporting the GEN command externally with Db2 Administration Tool® load libraries.
- Updated the PLAN2 on page 439 option documentation to clarify its relationship with **DB2ALIB**, including the requirement to specify the plan name when **DB2ALIB** is used.
- Updated the restriction in the **IMS management of ACBs** section. Clarified that ZDT/IMS requires **IMS APAR PH40873 (PTF UO06108)** to access logical databases in an IMS-managed ACB environment. See IMS management of ACBs on page 253.
- Clarified the preference of the Z Data Tools internal PL/I compilers over the licensed site compiler and documented how to select the internal compiler version using the `ALTPLI` option. Added a table that documents the order of precedence for internal and external PL/I compilers. See topic Customizing for processing PL/I include books on page 46.
- Enhanced the `SHOWPLI` command documentation to explain that the command identifies whether Z Data Tools is using the internal or external PL/I compiler. Added the batch and ISPF output messages that indicate which PL/I compiler is in use, including the message displayed when a licensed PL/I compiler is used. See SHOWPLI command on page 47.

- Added new topics under to document how to use and configure a licensed site PL/I compiler, including LINKLIST and batch JCL skeleton configuration, APF authorization requirements, and suppression of PL/I compiler warning messages.

  See topics: Using a licensed site PL/I compiler on page 47, Adding the PL/I compiler library to the LINKLIST on page 48, Adding the PL/I compiler library to the batch JCL skeletons on page 48, Authorizing the PL/I compiler library for batch processing on page 48, and Suppressing PL/I compiler warning messages on page 48.

- Added information about the `FILEM.PLI.EXTERNAL` FACILITY class profile to control use of the licensed site PL/I compiler, including RACF access requirements and handling of `RACROUTE STATUS=ACCESS` requests. See topic Controlling the use of the PL/I compiler on page 70.
- Updated Changing the JCL skeleton for batch mode on page 57 to document how to configure the STEPLIB DD statement for a licensed PL/I compiler and how to use the `FMNPLI DD DUMMY` statement to force the Z Data Tools internal PL/I compiler.

About this document

These topics provide information needed to plan for, customize, maintain, and diagnose problems with HCL Z Data Tools.

Z Data Tools comprises four components:

- Z Data Tools Base function
- HCL Z Data Tools Db2® component (ZDT/Db2)
- HCL Z Data Tools IMS™ component (ZDT/IMS)
- HCL Z Data Tools CICS® component (ZDT/CICS)

To install and use ZDT/Db2, ZDT/IMS, or ZDT/CICS, you **must** first install the same version of Z Data Tools Base function. You cannot use ZDT/Db2, ZDT/IMS, or ZDT/CICS with any other version of Z Data Tools. You can install any of ZDT/Db2, ZDT/IMS, and ZDT/CICS in any order, at any time after you have installed Z Data Tools Base function.

This book is divided into five sections:

- Customizing Z Data Tools on page 22, which provides information about the Z Data Tools Base function.
- Customizing Z Data Tools Db2 Component on page 152.
- Customizing Z Data Tools IMS Component on page 243.
- Customizing Z Data Tools CICS Component on page 324.
- The appendixes provide information about the Z Data Tools options, the library management system exit, and how to maintain Z Data Tools.

You will need to read Part 1. Read Parts 2 to 4 if you intend to install that component.

If you intend to use ZDT/CICS with the ZCC server, you will need to read Customizing the ZCC server on page 381.

Parts 1 to 4 are further divided into chapters describing the preparation for customizing Z Data Tools and its components, and for customizing Z Data Tools, its components and the operating environment. The last chapter in each section describes how to verify your installation and customization.

This book also provides information needed to customize the Japanese language versions of Z Data Tools and the ZDT/Db2, ZDT/IMS, and ZDT/CICS components.

Unless otherwise stated, the information provided in this book refers to the English versions of Z Data Tools, ZDT/Db2, ZDT/IMS, and ZDT/CICS.

Before you read this book, read the Z Data Tools Program Directory.

Overview of Z Data Tools

Z Data Tools helps you to work with various storage media and maintain data.

Z Data Tools provides production and development logical file manipulation for WebSphere MQ queue data, z/OS® Unix files, CICS® files, TS and TD queues, and QSAM and VSAM data sets. VSAM data sets include ESDS, KSDS, RRDS, and VRRDS.

Z Data Tools provides applications programmers and systems programmers with the capability to access, view, edit, find, modify, copy, maintain, repair, migrate, print, and manage data and files. The Template Workbench allows users (particularly applications programmers) to format data stored in any of the above files based upon the definitions in COBOL, PL/I, or High Level Assembler copybooks, or to define data formats dynamically. Adding to this capability is an interactive interface that allows programmers to view data in both table and single-record formats. In addition, selection of data can be performed using simple REXX-like conditions that can easily be tailored by the programmer. All of the features available online to manipulate data can also be applied in batch jobs.

You can access Z Data Tools functions in several ways:

- In full-screen mode as an ISPF or CICS® application.
- Through a Microsoft Visual Studio Code Extension.
- For routine tasks, you can use Z Data Tools in batch jobs using control statements.

Z Data Tools uses 31-bit addressing. It is enabled for multicultural support and can be translated if required. Z Data Tools provides Service Oriented Architecture (SOA) support, allowing the generation of XML data from files.

ZDT/Db2

The Z Data Tools Db2® component extends this functionality to the Db2® environment in the following ways:

- Extends the Z Data Tools editor facilities to apply to Db2® data
- Extends the ISPF object list utility to apply to Db2® data
- Extends the Z Data Tools data create facility to Db2® data
- Extends the Z Data Tools data copy facility to include Db2® data as both source and target
- Extends the Z Data Tools template processing to the Db2® environment
- Provides an SQL analysis feature for both existing plans and embedded source code
- Provides an easy interface to SQL statement generation for those who do not know SQL
- Provides an interface to generate JCL and Db2® control statements for selected Db2® utilities, without the user needing to know the syntax for these utilities

ZDT/IMS

The Z Data Tools IMS™ component further extends this functionality to provide support for IMS™ databases. ZDT/IMS provides database edit and browse functions such as:

- Ability to view and edit data in IMS™ databases with full-screen ISPF panels
- Display either multiple segments or a single segment on the one screen
- Formatted segment display (in context of COBOL or PL/I data structure)
- Formatted tabular display of data for a single segment type
- Selectable segment display (choose which segments to display based on segment keys or segment contents, or both)
- Selectable field display (choose which fields to display, and in which order)
- Unformatted segment display

- Ability to insert, delete, and repeat segments and their children
- Ability to find and change data across an entire database hierarchy
- Optional audit trail of all database updates

ZDT/IMS allows the user to define templates and views which associate COBOL and PL/I data structures with each segment type in a DBD; and define views which use the COBOL or PL/I data structures to specify the selection criteria for segment types.

ZDT/IMS provides support for either static (existing) or dynamic PSBs, the option to access databases via either DLI or BMP processing, and support for HALDB, HDAM, HIDAM, HISAM, HSAM, DEDB, MSDB, and logical databases, including databases with secondary indexes.

ZDT/CICS

The Z Data Tools CICS® component provides a powerful set of utility functions for editing, browsing, printing, and altering the status of CICS® resources. The CICS® resources supported are files, temporary storage queues, and transient data queues. With appropriate authority the user can also modify the status of the CICS® resources. ZDT/CICS incorporates much of the functionality of Z Data Tools into the CICS® environment.

ZDT/CICS also incorporates access to CICS® resources from batch and ISPF.

ZDT/CICS uses Interactive Panel Viewer, a feature of Z Common Components, to display panels on CICS® with a similar capability to ISPF panels. ZDT/CICS panels allow you to select options and to specify parameters, commands and program function (PF) keys to simplify requests for common functions. ZDT/CICS panels provide full screen format for information display and editing.

Who might use this document

This book is for system programmers and system administrators who plan for, customize, and maintain Z Data Tools.

It is also relevant to users who carry out diagnostic tasks on this product.

To use this book, you need to be familiar with the z/OS® operating system, the publications that describe your system, and job control language (JCL) or exec processing.

Experienced installers

If you are experienced in installing products with SMP/E, see the following fast path sections in the Z Data Tools Program Directory:

1. “Installation Requirements and Considerations”: see sub-section “DASD Storage Requirements”.
2. “Installation Instructions”: follow each instruction as described in the *Program Directory*.

Other documentation that you might need

For the installation of Z Data Tools, and the Db2®, IMS™, and CICS® components, or for the installation of the Z Data Tools Japanese component, you will need to refer to the Z Data Tools Program Directory.

You may also need to refer to the *Z Data Tools User's Guide and Reference*, the *Z Data Tools User's Guide and Reference for DB2 Data*, the *Z Data Tools User's Guide and Reference for IMS Data*, or the Z Data Tools User's Guide and Reference for CICS.

Syntax notation

Throughout this book, syntax descriptions use the structure defined below.

- Read the syntax diagrams from left to right, from top to bottom, following the path of the line.

  The >>—— symbol indicates the beginning of a statement.

  The ——–> symbol indicates that the statement syntax is continued on the next line.

  The >—— symbol indicates that a statement is continued from the previous line.

  The —–>< indicates the end of a statement.

  Diagrams of syntactical units other than complete statements start with the >—— symbol and end with the ——–> symbol.

- **Keywords** appear in uppercase letters (for example, ASPACE) or upper and lower case (for example, PATHFile). They must be spelled exactly as shown. Lower case letters are optional (for example, you could enter the PATHFile keyword as PATHF, PATHFI, PATHFIL or PATHFILE).

  Variables appear in all lowercase letters in a special typeface (for example, *integer*). They represent user-supplied names or values.

- If punctuation marks, parentheses, or such symbols are shown, they must be entered as part of the syntax.
- Required items appear on the horizontal line (the main path).

```
INSTRUCTION required item
```

- Optional items appear below the main path. If the item is optional and is the default, the item appears above the main path.

```
INSTRUCTION [ { default item  | optional item } ]
```

- When you can choose from two or more items, they appear vertically in a stack.

  If you **must** choose one of the items, one item of the stack appears on the main path.

```
INSTRUCTION { required choice1  | required choice2 }
```

  If choosing one of the items is optional, the whole stack appears below the main path.

```
INSTRUCTION [ { optional choice1  | optional choice2 } ]
```

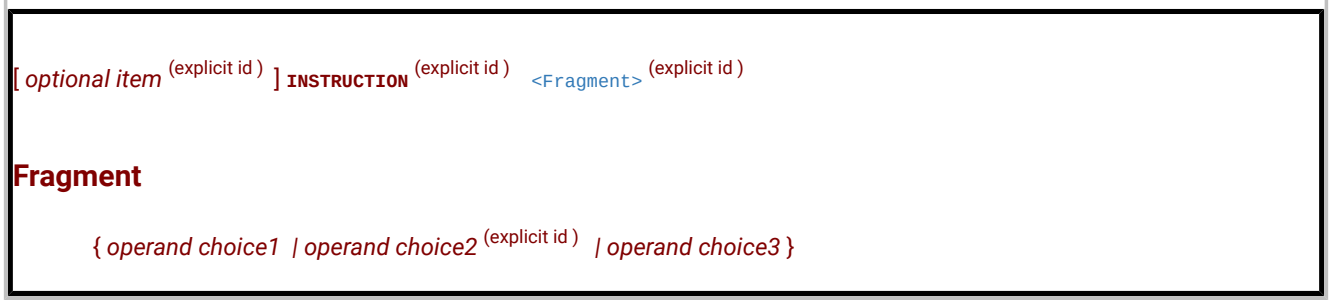
- An arrow returning to the left above the main line indicates an item that can be repeated. When the repeat arrow contains a separator character, such as a comma, you must separate items with the separator character.



A repeat arrow above a stack indicates that you can make more than one choice from the stacked items, or repeat a single choice.

The following example shows how the syntax is used.

Syntax diagram



Part I. Customizing Z Data Tools

Chapter 1. Preparing to customize Z Data Tools

Before you can use Z Data Tools, you will need to customize Z Data Tools and the operating environment. Table 1: Summary of steps for customizing Z Data Tools and the operating environment on page 23 lists the customization tasks you can perform for Z Data Tools. Read the referenced sections to see if you need to perform the customization described.

Checklist for installing and customizing Z Data Tools

Table 1. Summary of steps for customizing Z Data Tools and the operating environment

	Description
__ 1	Concatenate Z Data Tools libraries to the LINKLIST. See Concatenating libraries to the LINKLIST on page 26
__ 2	Modify the TSO logon procedure. See Modifying the TSO logon procedure on page 27.
__ 3	Enable/Register Z Data Tools See License and enable Z Data Tools on page 30.
__ 4	Add Z Data Tools to the ISPF menu. See Adding Z Data Tools to the ISPF menu on page 36.
__ 5	Define Z Data Tools in an ISPF command table. See Defining Z Data Tools in an ISPF command table on page 37.
__ 6	Make Z Data Tools the default VSAM editor. See Making Z Data Tools the default VSAM editor on page 38.
__ 7	Enable WebSphere® MQ support. See Enabling WebSphere MQ support on page 41.
__ 8	Customize to use COBOL copybooks. See Customizing for processing COBOL copybooks on page 42.
__ 9	Customize to use PL/I include books. See Customizing for processing PL/I include books on page 46.
__ 10	Customize to use HLASM copybooks. See Customizing for processing HLASM copybooks on page 49.
__ 11	Customize for DFSORT™ to improve performance. See Customizing to use DFSORT to improve Z Data Tools performance on page 49.
__ 12	Bind Db2® for OAM functions. See Binding Db2 to use Z Data Tools object access method (OAM) functions on page 52.
__ 13	Authorize Z Data Tools. See Planning for running Z Data Tools with or without APF-authorization on page 27 and Running Z Data Tools with APF-authorization on page 53.

Table 1. Summary of steps for customizing Z Data Tools and the operating environment (continued)

	Description
__ 14	Change the default options. See Changing the default options on page 55.
__ 15	Change the batch JCL skeleton. See Changing the JCL skeleton for batch mode on page 57.
__ 16	Customize the batch return codes. See Customizing Z Data Tools batch return codes on page 60.
__ 17	Provide a user I/O exit. See Customizing Z Data Tools to use an I/O exit on page 123.
__ 18	Change the print and display translation tables. See Changing the print and display translation tables on page 62.
__ 19	Change the ASCII translation tables. See Changing the ASCII translation tables on page 63.
__ 20	Customize the Z Data Tools security environment. See Customizing the Z Data Tools security environment on page 65.
__ 21	Decide how to customize the Z Data Tools audit facility. See Alternatives for controlling Z Data Tools Base auditing on page 28.
__ 22	Customize Z Data Tools for national languages. See Customizing Z Data Tools for national languages on page 104.
__ 23	Customize Z Data Tools to use library management system libraries. See Customizing Z Data Tools to use library management system libraries on page 120.

Library names after you finish installing

Throughout this book it is assumed that you have installed Z Data Tools into the default libraries. The default high level qualifier is HFM. Therefore after you have installed Z Data Tools the names of the target and distribution libraries will be those listed in Table 2: Z Data Tools target and distribution libraries on page 24.

If you have installed a Japanese component, the Japanese target and distribution libraries will be those listed in Table 3: Z Data Tools Japanese target and distribution libraries on page 25.

Table 2. Z Data Tools target and distribution libraries

Target library	Distribution library	Usage
HFM.SHFMMOD1	HFM.AHFMMOD1	Load modules
HFM.SHFMMOD2	HFM.AHFMMOD2	ZDT/CICS-specific load modules

Table 2. Z Data Tools target and distribution libraries (continued)

Target library	Distribution library	Usage
HFM.SHFMMODA	HFM.AHFMMODA	ZDT/CICS-specific load modules that must be APF-authorized
HFM.SHFMSAM1	HFM.AHFMSAM1	Sample jobs and source code
HFM.SHFMPENU	HFM.AHFMPENU	English panels
HFM.SHFMMENU	HFM.AHFMMENU	English messages
HFM.SHFMSLIB	HFM.AHFMSLIB	Skeletons
HFM.SHFMTENU	HFM.AHFMTENU	English tables
HFM.SHFMMAC1	HFM.AHFMMAC1	Macros
HFM.SHFMEXEC	HFM.AHFMEXEC	Execs
HFM.SHFMCLIB	HFM.AHFMCLIB	CLISTs
HFM.SHFMDBRM	HFM.AHFMDBRM	DBRMs

Table 3. Z Data Tools Japanese target and distribution libraries

Target library name	Distribution library name	Usage
HFM.SHFMMODJ	HFM.AHFMMODJ	Japanese BMS maps
HFM.SHFMPJPN	HFM.AHFMPJPN	Japanese panels
HFM.SHFMMJPN	HFM.AHFMMJPN	Japanese messages
HFM.SHFMTJPN	HFM.AHFMTJPN	Japanese tables

Alternatives for making Z Data Tools available

You can make Z Data Tools available to your users in one of four ways:

- By concatenating HFM.SHFMMOD1 to your LINKLIST.
- By adding HFM.SHFMMOD1 to the STEPLIB DD statement in your TSO logon procedure.
- By activating the HFM.SHFMMOD1 data set using the TSO command, TSOLIB.
- By adding HFM.SHFMMOD1 to the ISPLLIB DD concatenation.

Important: If you implement LIBDEFs to allocate the Z Data Tools load libraries for a TSO user, and that user also has access to Z Data Tools load libraries without having to issue LIBDEFs, the Z Data Tools load libraries must be the same (identical version and maintenance levels).

Example

The Z Data Tools libraries are allocated in a user's TSO logon procedure. The ISPF main menu is customized to allow ZDT/Db2 as an option. Selecting this option runs an exec based on HFMINIT to allocate Z Data Tools libraries using LIBDEFs.

The Z Data Tools libraries allocated in the TSO logon procedure are different to the Z Data Tools libraries allocated in the exec using LIBDEFs.

In this situation problems will arise if the user has ZDT/Db2 running on one ISPF logical session, and Z Data Tools Base component or ZDT/IMS running on a second ISPF logical session.

Explanation

Most Z Data Tools modules are reentrant. In an ISPF split screen environment each Z Data Tools module will be loaded only once by the first ISPF session that requires it. If the user starts another ISPF session and runs Z Data Tools, any modules needed by Z Data Tools will only be loaded if the module in not already loaded. This can result in unpredictable results - in terms of the module loaded - depending on whether the Z Data Tools Base component or the Z Data Tools IMS™ component is started before a Z Data Tools Db2® session, or vice versa.

To make Z Data Tools readily available from ISPF, configure your ISPF environment as described in Customizing the operating environment for Z Data Tools on page 30.

Concatenating libraries to the LINKLIST

To make Z Data Tools commonly available, add the HFM.SHFMMOD1 library to your concatenated LINKLIST.

If you plan to use functions using PL/I include books, add the Language Environment® runtime library, SCEERUN, to your concatenated LINKLIST, in front of any other PL/I runtime library.

If you plan to use functions using High Level Assembler (HLASM) copybooks, ensure that the HLASM load library, ASM.SASMMOD1, is available to Z Data Tools. You can do this by adding this library to your concatenated LINKLIST.

If you plan to use Z Data Tools to access COBOL copybooks or PL/I include books stored in library management system libraries, add the Language Environment® runtime library, SCEERUN, to your concatenated LINKLIST. For information about using library management system libraries, see Customizing Z Data Tools to use library management system libraries on page 120.

To concatenate these libraries to the LINKLIST, add them to either your LNKLST*xx* or PROG*xx* member in SYS1.PARMLIB.

Modifying the TSO logon procedure

If you did not add HFM.SHFMMOD1 to your LINKLIST, as described above, you can add this library to the STEPLIB DD statement or the ISPLLIB DD statement in your TSO logon procedure.

You also need to add the following Z Data Tools libraries to your TSO logon procedure.

```
DDNAME ISPMLIB:  add library HFM.SHFMMENU
DDNAME ISPPLIB:  add library HFM.SHFMPENU
DDNAME ISPSLIB:  add library HFM.SHFMSLIB
DDNAME ISPTLIB:  add library HFM.SHFMTENU
DDNAME SYSEXEC:  add library HFM.SHFMEXEC
DDNAME SYSPROC:  add library HFM.SHFMCLIB
```

If you are installing the Japanese component, add the following Z Data Tools libraries to your TSO logon procedure.

```
DDNAME ISPMLIB:  add library HFM.SHFMMJPN in front of HFM.SHFMMENU
DDNAME ISPPLIB:  add library HFM.SHFMPJPN in front of HFM.SHFMPENU
DDNAME ISPTLIB:  add library HFM.SHFMTJPN in front of HFM.SHFMTENU
```

If you are using the ISPF alternate libraries, make the following change into your concatenation.

```
DDNAME: ISPSALT: add library HFM.SHFMSLIB
```

Planning for running Z Data Tools with or without APF-authorization

Z Data Tools can be run in batch mode with or without APF-authorization. Z Data Tools cannot run APF-authorized under ISPF.

There are a number of reasons why you might want to run Z Data Tools APF-authorized in batch mode:

- Users can be permitted to use disk fullpack processing. For more information, see Controlling fullpack access to DASD volumes on page 70.
- Users can be permitted to use bypass label processing (BLP), even if the system does not support BLP. For more information, see Controlling Bypass Label Processing (BLP) on page 73.
- Users can perform catalog actions that require APF-authorization, as described in *z/OS DFSMS Access Method Services Commands*.

If you plan to use SMF to record audit trail information for the Z Data Tools Base function, or for the Db2® or IMS™ components, you must make Z Data Tools (or at least the module HFMSMF) APF-authorized. See Using System Management Facilities (SMF) for audit logging on page 86 for more information.

If you plan to run Z Data Tools APF-authorized in batch mode, and you want to run Z Data Tools batch jobs using COBOL copybook templates, then you **MUST** also authorize the COBOL compiler library.

If you are accessing data in an LMS without a SUBSYS interface, or you are using HFMCRAEX to access data in an LMS, then you cannot run Z Data Tools APF-authorized.

Alternatives for controlling Z Data Tools Base auditing

Z Data Tools auditing is an optional facility. There is no requirement to implement it and Z Data Tools works if Z Data Tools auditing is not implemented. You should consider:

- Whether user access to data sets and other resources using Z Data Tools requires auditing.
- The information that Z Data Tools audit log records can provide.
- The information that Z Data Tools audit log records cannot provide, and possible alternatives to obtaining that information.
- If you do decide to use Z Data Tools auditing, how you will handle any issues associated with large audit log data sets, or additional SMF records.
- How you will use the information provided by Z Data Tools audit log records.

If your site requires a record of a user's read access to data sets, an external security product such as RACF® can be configured to log access by some or all users, and may be a better alternative.

Z Data Tools audit of read access to data sets does not write audit log records for every record processed, rather the name of the data set and how many records were processed are written to the audit log.

Z Data Tools audit of changes to data sets typically writes two log records, a before and after image of the record that was changed. If you intend to log update changes to data sets that are subject to heavy update activity you need to consider the performance impact of writing many audit log records, also the size of any audit log data sets that may be produced.

You have two choices with respect to auditing of Z Data Tools audit activities:

HFM0POPT controlled auditing

The facilities available with HFM0POPT controlled auditing are that you can specify auditing to the user's audit log data set, to the user's audit log data set with automatic (mandatory) printing of the audit log at the completion of the session, or to SMF. This auditing only applies to changes made by means of the Z Data Tools editor.

SAF-rule controlled auditing

This relies on various SAF FACILITY and XFACILIT resource rules which you define with an external security product, such as RACF® (or equivalent product).

These points summarize the facilities available with SAF-rule controlled auditing:

- Auditing can be (optionally) specified for all Z Data Tools functions.
- Different auditing requirements can be specified for different TSO user IDs.
- Different auditing requirements can be specified for access to different resources.
- You can provide Z Data Tools users with a "Create audit trail" option for the Z Data Tools edit functions. This is also SAF-rule controlled. The presence of the "Create audit trail" option does not guarantee that the user can switch off auditing, since this depends on the level of access the user has to the appropriate SAF resource names. When a user

has access to the "Create audit trail" option, they can always turn on auditing, even if the relevant SAF resource rules do not require auditing.
- You can specify auditing to the user's audit log data set, to the user's audit log data set with automatic (mandatory) printing of the audit log at the completion of the session, or to SMF. Dual logging (to the user's audit log data set and to SMF) can also be specified.

Some other points to consider are:

- Auditing to the user's audit log data set can result in large numbers of audit log data sets. This may have disk space implications. You may need to consider implementing automatic purging or archiving of audit log data sets.
- Auditing to SMF (only) requires additional set-up, but provides a more reliable and secure environment for capturing audit information than audit logging to the user's audit log data set.
- If you implement SAF-rule controlled auditing you need to decide how Z Data Tools auditing will be enabled. This is described in more detail in Customizing the Z Data Tools audit facility for the Z Data Tools Base component on page 88. There are two alternatives: one requires an enabling SAF rule and the presence of a member in SYS1.PARMLIB, the other requires an enabling SAF rule but has no requirement for a member in SYS1.PARMLIB. The use of a member in SYS1.PARMLIB provides additional facilities compared with the alternative that does not require the use of SYS1.PARMLIB. The additional facilities are documented in Z Data Tools options specified in PARMLIB members on page 515.

When you have determined the appropriate type of auditing for your installation, follow the instructions in Customizing the Z Data Tools audit facility for the Z Data Tools Base component on page 88.

Chapter 2. Customizing the operating environment for Z Data Tools

This chapter describes how to customize the operating environment for Z Data Tools. You do this after you have installed Z Data Tools.

License and enable Z Data Tools

Product enablement in IFAPRDxx

You have the option to include an entry for Z Data Tools in the IFAPRDxx parmlib member as follows:

```
PRODUCT OWNER('HCL')
NAME('HCL Z DATA TOOLS')
ID(19OP1220)
VERSION(*) RELEASE(*) MOD(*)
FEATURENAME('HCL-ZDT')
STATE(ENABLED)
```

Sample registration parmlib entry HFMWIFA, which contains the above statements, is provided in your *hlq*.SHFMSAM1 data set.

After the IFAPRDxx parmlib member is updated, it can be activated dynamically (until the next IPL) using the following console command:

```
SET PROD=xx
```

Additional Z Data Tools IFAPRDxx processing

If a product above is not defined in IFAPRDxx, when Z Data Tools is first invoked it will register during initialization as shown in the HFMWIFA sample.

If a product above is defined with STATE(DISABLED) or STATE(NOTDEFINED), the product will not be selected for registration.

To prevent Z Data Tools from running, use the following IFAPRDxx entry:

```
PRODUCT OWNER('HCL')
NAME('HCL Z DATA TOOLS')
ID(19OP1220)
VERSION(*) RELEASE(*) MOD(*)
FEATURENAME('HCL-ZDT')
STATE(DISABLED)
```

HCL advises against defining IFAPRDxx entries that have NAME(*) or ID(*) fields, as this could result in unintended product registrations.

Modifying the ISPF environment

To make it easy to start Z Data Tools under ISPF, configure your ISPF environment as described in the following sections.

TSO region size

When using Z Data Tools from ISPF, it is recommended you use at least a 4 MB logon region size (in conjunction with the 32 MB of "extended" storage that TSO generally provides) to provide enough 24-bit storage for various functions to operate effectively. A suggested minimum TSO region size is 50 MB.

This starting value might need to be increased when performing intensive virtual storage functions, such as:

- The use of templates with many fields or layouts.
- In-memory editing of large files.

Preparing Z Data Tools to run with LIBDEFs

Copy the member HFMELIBD from *hlq*.SHFMSAM1 to a library that is available in the SYSPROC DD statement for your TSO/ISPF users. Modify the exec as follows.

hlq = 'HFM'

Specifying this will result in names in the form *hlq*.SHFM*xxxx* if `sfx = ''`

optl = ''

Specify a value if there is a separate installation options library.

sfx = ''

Specifying this will result in names in the form *hlq*.SHFM*xxxx*.*sfx*

hlq2 = ''

LIBDEF fixtest library first in the form *hlq2*.SHFM*xxxx*

sfx2 = ''

LIBDEF fixtest library first in the form *hlq2*.SHFM*xxxx*.*sfx2*

CICS parameters

HFMCICS= ''

Specify the data set that contains the CICSVTAM applids and descriptions required to support generic queries. If this data set is already allocated to a TSO procedure or as part of a logon command, then it need not be specified here.

SDFHEXCI= ''

If the EXCI load library is not available to ISPF applications, then specify the *cicshlq*.SDFHEXCI data set name here.

WebSphere® MQ load libraries

If the SCSQANLE, SCSQAUTH, and SCSQLOAD are not available to the ISPF session then specify the corresponding data set names using the following parameters:

- SCSQAUTH = ' '
- SCSQANLE = ' '
- SCSQLOAD = ' '

Notes:

1. For a standard installation you will only need to set `hlq =` to the installation value. Specify `optl = data set name` if you have customized versions of the installation options module in a different library to *hlq*.SHFMMOD1.
2. *hlq2* and *sfx2* are there to support BETA library testing.

Invocation of Z Data Tools using LIBDEFs

HFMELIBD is a multi-purpose REXX exec to provide LIBDEF invocation of Z Data Tools features and functions. This eliminates the need for various user-coded EXECs that perform LIBDEFs before invoking Z Data Tools features or functions. The exec runs as follows.

Value

Description

ZDT

Invoke the Z Data Tools primary options menu.

ZI

Invoke the ZDT/IMS primary options menu.

ZD

Invoke the ZDT/Db2 primary options menu.

FND

Invoke Z Data Tools enhanced search.

LST

Invoke Z Data Tools enhanced reference lists.

FUN

Invoke supported Z Data Tools functions with data set names to populate the entry panels. See FUN parameter syntax on page 33.

DEF

Perform LIBDEFs for Z Data Tools libraries from a REXX that invokes Z Data Tools functions in keyword mode.

DROP

Clear LIBDEFs setup from HFMELIBD DEF command.

INV

To perform the equivalent function as HFMINV as specified in the ISPF Configuration tables.

- HFMELIBD INV DSE / is the LIBDEF equivalent of HFMINV DSE /
- HFMELIBD INV DSB / is the LIBDEF equivalent of HFMINV DSB /
- HFMELIBD INV DSV / is the LIBDEF equivalent of HFMINV DSV /

See Making Z Data Tools the default VSAM editor on page 38 for more details.

MENU

To provide a LIBDEF invocation of a cut-down menu — equivalent to the HFMR exec.

See Invoking Z Data Tools from ISPF 3.4 or a data set list on page 38 for more details.

Note: For ease of use, create an intermediate exec with a short name as follows:

```
Example. Create member LFM in a data set in the SYSPROC or SYSEXEC concatenations.
 /* REXX - LIBDEF invocation of Z Data Tools from ISPF 3.4 */
 ARG DSN
 CALL HFMELIBD 'MENU 'dsn
 EXIT
```

LFM, in this case, is the command to invoke the Z Data Tools cut-down menu from an ISPF 3.4 or a data set list.

FUN parameter syntax

If you specify FUN then the following syntax applies.

```
HFMELIBD FUN function 'dsn1(mbr1)' 'dsn2(mbr2)' 'dsn3(mbr3)' 'dsn4(mbr4)'
```

function must be one of the following supported functions.

Value

Description

CLM

Compare load module

DSB

Data set browse

DSC

Data set copy

DSE

Data set edit

DSG

Data set create

DSM

Data set compare

DSV

Data set view

PBK

Print and view copybook or template

SCS

Catalog services

TPIMP

Template import

TPEXP

Template export

TPED

Template edit

TPUP

Template update

VLM

View load module

'dsn1(mbr1)' is the input data set or path name.

'dsn2(mbr2)' is the input template (specify '-' to bypass).

'dsn3(mbr3)' is the output data set or path name.

'dsn4(mbr4)' is the output template.

Note: You can use the HFMELIBD REXX exec to invoke functions DSC, DSE, and DSV to copy, edit, and view data sets, CICS resources, and MQ queues using the appropriate format. For example, the format to invoke HFMELIBD for function DSE is the following:

Data set

```
call HFMELIBD FUN DSE 'data.set.name(memname)'
```

CICS resource

```
call HFMELIBD FUN DSE 'rt:applid:rname'
```

MQ queue

```
call HFMELIBD FUN DSE 'MQ:managerid:queuename'
```

See the sections about specifying a CICS resource and specifying an MQ manager or queue in the *Z Data Tools User's Guide and Reference* for more information about specifying these resources.

Example

Example 1. Invoking Z Data Tools, ZDT/IMS, and ZDT/Db2 primary options from a selection panel

```
⋮)BODY CMD(ZCMD) ...
9       IBM                     Products IBM program development products
10      SCLM                    SW Configuration Library Manager
11      Workplace               ISPF Object/Action Workplace
ZDT     Z Data Tools            Z Data Tools                          NEW
ZI      ZDT/IMS                 Z Data Tools IMS component            NEW
ZD      ZDT/Db2                 Z Data Tools Db2 component            NEW
⋮
)PROC
⋮
&ZSEL = TRANS( TRUNC (&ZCMD,'.')
⋮
9,'PANEL(ISRDIIS) ADDPOP'
10,'PGM(ISRSCLM) SCRNAME(SCLM) NOCHECK'
11,'PGM(ISRUDA) PARM(ISRWORK) SCRNAME(WORK)'
ZDT,'CMD(HFMELIBD ZDT)' /* Z Data Tools */                           ⓃⒺⓌ
ZI,'CMD(HFMELIBD ZI)'   /* ZDT/IMS      */                           ⓃⒺⓌ
ZD,'CMD(HFMELIBD ZD)'   /* ZDT/Db2      */                           ⓃⒺⓌ
```

Example

Example 2. ISPF command table with LIBDEF invocations

```
     Verb      T  Action
____ ZDT       2  SELECT CMD(%HFMELIBD ZDT &ZPARM)      <= Z Data Tools Base
____ ZI        2  SELECT CMD(%HFMELIBD ZI  &ZPARM)      <= ZDT/IMS
____ ZD        2  SELECT CMD(%HFMELIBD ZD  &ZPARM)      <= ZDT/Db2
____ ELIST     2  SELECT CMD(%HFMELIBD LST &ZPARM)      <= Enhanced reference list
____ EFIND     2  SELECT CMD(%HFMELIBD FND &ZPARM)      <= Enhanced search
```

Example

Example 3. Using REXX to invoke the VLM function (view load module)

This example will run Z Data Tools option 3.10.1 and populate the input data set field with 'HFM.LOAD'

```
/* REXX */
call HFMELIBD FUN VLM 'HFM.LOAD'
```

Example

Example 4. Using REXX to invoke the DSE function (data set edit)

This example will run Z Data Tools option 2 and populate the input data set field and template fields.

```
/* REXX */
call HFMELIBD FUN DSE 'HFM.FMDATA'  'HFM.TEMPLATE(SAMPLE)'
```

Example

Example 5. Using REXX to invoke the DSE function (MQ queue edit)

This example will run Z Data Tools option 2 and populate the input data set field with the MQ queue.

```
{{/* REXX */
call HFMELIBD FUN DSE 'MQ:MQ91:HFM.QUEUE.TEST'}}
```

Example

Example 6. REXX load module report using the DEF and DROP parameters

```
/* REXX - VIEW LOAD MODULE - XML REPORT    */
/*                                         */
CALL HFMELIBD DEF
ADDRESS TSO
  "ALLOC FI(SYSPRINT) DUMMY REUSE"
  'ALLOC FI(HFMXMLO) NEW SP(1,5) TRACKS,
        LRECL(1024) BLKSIZE(32720) RECFM( V B )'
ADDRESS ISPEXEC
  "SELECT CMD(HFMMAIN $VLM DSNIN='HFM.LOAD'",
      "MEMBER=TURBO1,FUNCTION=PRINT,DATEFORM=YYMMDD,XML=YES)"
ADDRESS TSO
  'EXECIO * DISKR HFMXMLO (STEM XMLO. FINIS'
  IF RC = 0 THEN DO
    DO I = 1 TO XMLO.0
      SAY XMLO.I
    END
  END
  "FREE FI(HFMXMLO)"
  "FREE FI(SYSPRINT)"
CALL HFMELIBD DROP
EXIT 0;
```

Adding Z Data Tools to the ISPF menu

To add Z Data Tools to your ISPF Primary Option Menu panel (ISR@PRIM), insert the additional lines (◄ **New**) as shown in this figure:

Figure 7. Adding Z Data Tools to the ISPF Primary Option Menu panel

```
⋮
)BODY  CMD(ZCMD)
⋮
 9  IBM Products  IBM program development products
 10 SCLM          SW Configuration Library Manager
 11 Workplace     ISPF Object/Action Workplace
 Z  Z Data Tools  Z Data Tools            ◄   New
⋮
)PROC
⋮
 &ZSEL = TRANS( TRUNC (&ZCMD,'.')
⋮
  9,'PANEL(ISRDIIS) ADDPOP'
 10,'PGM(ISRSCLM) SCRNAME(SCLM) NOCHECK'
 11,'PGM(ISRUDA) PARM(ISRWORK) SCRNAME(WORK)'
  Z,'PANEL(HFMSTASK) SCRNAME(ZDTOOLS) NEWAPPL(HFM)' /* Z Data Tools */ ◄ New
⋮
```

To invoke Z Data Tools with LIBDEFs, see Example 1. Invoking Z Data Tools, ZDT/IMS, and ZDT/Db2 primary options from a selection panel on page 35.

For information about configuring your ISPF Primary Option Menu panel, see z/OS ISPF Planning and Customizing.

Defining Z Data Tools in an ISPF command table

ISPF supports four different command tables where you can define an ISPF command to invoke Z Data Tools:

- Application command table
- User command table
- Site command table
- System command table

You can use the ISPF Command Table Utility (ISPF option 3.9) to create or change a command table that is not currently in use (the system command table, ISPCMDS, is always in use). When you add a command for Z Data Tools to one of these command tables, you can invoke Z Data Tools from any ISPF panel without prefixing the command with TSO.

For example, add the following entry to a command table to enable Z Data Tools to be run from any ISPF panel by entering HFM on the command line.

Verb	HFM
Action	SELECT PANEL(HFMSTASK) OPT(&ZPARM) SCRNAME(ZDTOOLS) SUSPEND NEWAPPL(HFM)
Description	Z Data Tools

To invoke Z Data Tools with LIBDEFs, see Example 2. ISPF command table with LIBDEF invocations on page 35.

For information about ISPF command tables, see *z/OS ISPF Planning and Customizing* and *z/OS ISPF User's Guide Vol II*.

Avoiding conflicts between ZDT commands and other applications

If you encounter conflicts between Z Data Tools commands and the same commands defined in ISPF command tables for other products, one solution is create an application command table that is specific to Z Data Tools. Add an entry for the command that is used in the other application to the Z Data Tools application command table and specify the PASSTHRU option.

Making Z Data Tools the default VSAM editor

You can configure ISPF to specify Z Data Tools as the VSAM editor that is to be invoked automatically when a VSAM data set is specified in ISPF Options 1, 2, 3, or 11. You do this by updating the ISPF Configuration Table Keyword File to provide values for the VSAM-related keywords listed in Table 4: VSAM-related keywords and values on page 38:

Table 4. VSAM-related keywords and values

Keyword	Suggested value
VSAM_EDIT_ENABLED	YES
VSAM_EDIT_COMMAND	HFMINV DSE /
VSAM_EDIT_LIMITED	NO
VSAM_BROWSE_ENABLED	YES
VSAM_BROWSE_COMMAND	HFMINV DSB /
VSAM_BROWSE_LIMITED	NO
VSAM_VIEW_ENABLED	YES
VSAM_VIEW_COMMAND	HFMINV DSV /
VSAM_VIEW_LIMITED	NO
VSAM_RESTRICTED_BROWSE_DATASET	NONE
VSAM_RESTRICTED_EDIT_DATASET	NONE
VSAM_RESTRICTED_VIEW_DATASET	NONE

You then use this keyword file to build an ISPF configuration load module. For information about how to do this, see *z/OS ISPF Planning and Customizing*.

Invoking Z Data Tools from ISPF 3.4 or a data set list

You can invoke a cut-down Z Data Tools primary options menu against any data set on an ISPF 3.4 or data set list by invoking the supplied exec HFMR as a line command. Any subsequent entry panel navigated to via the cut-down menu has the primary data set name populated with the corresponding data set name from the ISPF 3.4 or data set list.

Note: This assumes the hlq.SHFMEXEC library has been added to SYSEXEC for the ISPF session.

Implementing Z Data Tools edit models in ISPF Edit

Configure your ISPF environment to enable access to Z Data Tools edit models from an ISPF Edit session.

To implement the Z Data Tools ISPF edit models, perform the following steps:

- Ensure that HFM.SHFMPENU is concatenated to ISPPLIB.
- Ensure that HFM.SHFMSLIB is concatenated to ISPSLIB.
- Modify the ISPF panel ISREMCLS to include a new model class named JCL.

  A sample of the updates required can be found in HFM.SHFMSAM1(HFMEMCLS).
- Modify the ISPF panels ISREMEXC and ISREMRXC to include an option for processing Z Data Tools external REXX functions from the EXEC and REXX model panels.

  A sample of the updates required for ISPEMEXC can be found in HFM.SHFMSAM1(HFMEMEXC) and the updates required for ISPEMRXC can be found in HFM.SHFMSAM1(HFMEMRXC).

Enabling Z Data Tools to work with certain products

You can enable Z Data Tools to work with IBM® languages and products. These products include COBOL, PL/I, DFSORT, Db2®, and WebSphere® MQ for z/OS® (WebSphere® MQ). The following sections describe how you do this.

Z Data Tools supported software environments

Supported software products that can be used with various Z Data Tools functions in a z/OS® environment:

- IBM Enterprise COBOL
- IBM Enterprise PL/I (included internally with Z Data Tools and usable only for compiling copybooks)
- IBM CICS® Transaction Server for z/OS
- IBM Information Management System (IMS™)
- IBM Db2®
- IBM MQ for z/OS

Install the latest Z Data Tools maintenance to resolve issues that might not be directly associated with the basic support for a software product.

Z Data Tools supports software products with either:

- A general availability (GA) release with no maintenance applied.
- A GA release with the specified PTF (TFA*nnnnn*) applied.

The following table shows the minimum maintenance level to apply to Z Data Tools V17 for each supported software product.

Note: The minimum software product level is the level listed in the table or the current minimum level supported by IBM, whichever is most recent.

Note: To get the most up-to-date information on supported operating systems, detailed system requirements, and hardware requirements, you can generate a dynamic software product compatibility report.

1. Go to the Software Product Compatibility Reports page.
2. Type the product name in the Search by product name field and click the product name in the list.
3. In the search results, select a release.

Table 5. Supported software environments

Supported environment	Supported environment version	Z Data Tools V17 minimum maintenance level
Enterprise COBOL	V6.5	V17.1.0 (UO0667)
	V6.4	V16.1.2 (UO05337)
	V6.3	GA
	V4.2	GA
Enterprise PL/I	V6.1	GA
	V5.3	V17.1.0 (UO05337)
CICS Transaction Server	V6.3	GA
	V6.2 (750)	GA
	V6.1 (740)	GA
	V5.6	GA
	V5.5	GA
IMS	V15.6.*x*	GA
	V15.5.*x*	GA
	V15.4.*x*	GA
	V15.3.*x*	GA
	V15.2.*x*	GA
Db2 for z/OS	V13.1 (Function level 508)	V16.1.1
	V13.1	GA

Table 5. Supported software environments (continued)

Supported environment	Supported environment version	Z Data Tools V17 minimum maintenance level
	V12.1 (Function level 500 or higher is required)	GA
z/OS	V3.2	GA
	V3.1	GA
IBM MQ for z/OS	V9.4	GA
	V9.3	GA
	V9.2	GA
	V9.1	GA

Enabling WebSphere® MQ support

To access WebSphere® MQ queue managers and queue contents on the local z/OS® system where Z Data Tools is running, the WebSphere® MQ libraries must be made available to Z Data Tools and users must be granted read and update access.

For the Z Data Tools Base function, the WebSphere® MQ load libraries SCSQANLE, SCSQAUTH, and SCSQLOAD must be made available to the TSO or batch job user, either in the linklist, or as part of the STEPLIB in the TSO procedure (or batch JCL). Z Data Tools MQ functions make use of sending messages to the SYSTEM.COMMAND.QUEUE and creating a dynamic queue using SYSTEM.COMMAND.REPLY.MODEL queue to retrieve the replies. The high level qualifier for this reply queue is customizable in the HFM0POPT table, as keyword MQREPHLQ. The sample options module contains a value of HFMTMQL. for the high level qualifier. When the dynamic queue name is passed to the MQ APIs for construction, it will use the HLQ as specified in the options module, appended with the current user ID and a ".*" string. The MQ manager being used will then return the full dynamic name to be used. You can change this high level qualifier to suit your installation's MQ security rules.

Note: For an HFMELIBD invocation of Z Data Tools, the WebSphere® MQ load libraries can be specified when customizing theHFMELIBD exec. See Preparing Z Data Tools to run with LIBDEFs on page 31 for more details.

Z Data Tools users who must access IBM® MQ resources require the following access:

- Update access to the dynamic queues *MQREPHLQ.userid*.*
- Update access to the SYSTEM.COMMAND.INPUT queue
- Read access to the SYSTEM.COMMAND.REPLY.MODEL queue

Requesting users also require relevant permissions to access specific IBM® MQ resources. These resources might include these permissions:

- To connect to queue managers
- To read or alter queue manager or queue attributes

- To create and delete resources
- To browse or destructively get queue messages

To determine what permissions are required, see *"Setting up security on z/OS"* in the IBM® MQ security documentation in IBM Knowledge Center.

For ZDT/CICS using the ZCC server, they must be added to the configuration data for the HFMLIB concatenation in the Z Common Components CONFIG DD concatenation. For more information about the ZCC server and configuration data, see Customizing the ZCC server on page 381.

Customizing for processing COBOL copybooks

To use the view, edit, copy, print, or data create functions with a COBOL copybook, you need to make a COBOL compiler available to Z Data Tools.

This functionality requires ADATA, therefore the COBOL compiler must have the ADATA capability. COBOL compilers such as VS COBOL II, and OS/COBOL, which do not support ADATA, cannot be used. All currently supported versions of IBM® Enterprise COBOL for z/OS® and OS/390® provide this support, and can therefore be used.

Z Data Tools provides two internal versions of the COBOL compiler for use in preference to a licensed site COBOL compiler. These compilers are installed into HFM.SHFMMOD1, if you installed Z Data Tools into the default libraries, with load modules named HFM8*xxxx* or HFM9*xxxx*. (The Z Data Tools COBOL compilers are reduced-function compilers and cannot be used to generate executable COBOL programs.)

The Z Data Tools internal COBOL compiler is used in preference to the licensed site compiler. The version of the internal compiler is selected with the ALTCOB option. See the table Table 71: Available COBOL compilers on page 385 for details.

For information about using a licensed site COBOL compiler, see Using a licensed site COBOL compiler on page 45. For more information about using the Z Data Tools internal COBOL compiler, see Using the Z Data Tools COBOL compiler on page 44.

The following table indicates the checking sequence to determine which COBOL compiler will be used.

Table 6. COBOL compiler order of precedence

	Option	Version	COBOL Module	Notes
HFMCOB	DD DUMMY	Internal	HFMxRCTL	Can be allocated in job step, TSO procedure, or via ALLOC.
HFMCOB	DD DSN=compiler library	External	IGYCRCTL	Can be allocated in job step, TSO procedure, or via ALLOC.
RACROUTE	STATUS=ACCESS FACILITY Class Profile FILEM.COBOL.EXTER	External	IGYCRCTL	Forces Z Data Tools to use an external COBOL compiler.

Table 6. COBOL compiler order of precedence (continued)

	Option	Version	COBOL Module	Notes
	NAL CONTROL access or higher			
If none of the above, Z Data Tools will LOAD module HFMxRCTL		Internal	HFMxRCTL	This is the default compiler. See the table, Table 71: Available COBOL compilers on page 385, under the ALTCOB option for COBOL versions provided.

Note:

- External refers to licensed site COBOL compiler (IGYCRCTL).
- Internal refers to the cutdown COBOL compiler shipped with Z Data Tools (HFM8RCTL or HFM9RCTL).

Use the SHOWCOB command to check whether Z Data Tools is using its internal COBOL compiler or another compiler.

- In batch, submit a Z Data Tools batch job with this control statement:

  ```
  $$FILEM SHOWCOB
  ```

  If a licensed, supported COBOL compiler is available to Z Data Tools, the job output will show:

  ```
  HFM0024I The Customer Licensed COBOL Compiler will be used.
  ```

  If Z Data Tools is using its own COBOL compiler, the job output will show:

  ```
  HFM0023I The Z Data Tools supplied COBOL Compiler will be used
  ```

- Under ISPF, enter SHOWCOB on any Z Data Tools command line. If a licensed site supported COBOL compiler is available to Z Data Tools, this message is displayed:

  ```
  Using Licensed COBOL
  ```

  If Z Data Tools is using its own COBOL compiler, this message is displayed:

  ```
  The Z Data Tools supplied COBOL Compiler will be used
  ```

Note: Additional messages will be produced detailing why the Z Data Tools COBOL compiler is being used.

Note that your COBOL compiler must be both licensed and supported. If you are using a licensed, non-supported compiler (for example, a licensed version of VS COBOL II), the message will still be: `Using Licensed COBOL`, but any attempt to compile copybooks will fail.

Using the Z Data Tools COBOL compiler

Z Data Tools uses its internal COBOL compiler by default.

Note: The Z Data Tools compiler requires a region size of at least 50 MB and possibly more, depending on the complexity of the copybook being compiled.

If Z Data Tools is to use its internal COBOL compiler, review the following sections for customization tasks you might need to perform. Not all the functions available using a licensed site COBOL compiler are available using the Z Data Tools compiler, and for functions that are available, the customization might differ.

Authorizing the COBOL compiler library for batch processing

If you are running Z Data Tools APF-authorized in batch mode, there is no further action you need to take to use the Z Data Tools COBOL compiler authorized. If you do not want to run your batch jobs authorized when using this compiler, then you must run all your Z Data Tools batch jobs un-authorized.

Suppressing COBOL compiler warning messages

If you have set TERM as a COBOL compiler option, add an allocation for HFMTERM DD to your TSO logon procedure:

```
//HFMTERM DD DUMMY
```

This will suppress unnecessary compiler warning messages.

Using COBOL compiler options with Z Data Tools

Z Data Tools overrides certain COBOL compiler options when it invokes the COBOL compiler. These options are:

```
ADATA          INEXIT
ADEXIT         LIB
ARITH          LINECOUNT
COMPILE        OBJECT
DBCS           OPTIMIZE
DECK           PRTEXIT
FLAG           SQL
```

If you are using the Z Data Tools COBOL compiler, these options are not marked as *"fixed"* in the provided COBOL default options module, HFM0DOPT, and you cannot change this. There is no facility to manipulate the options provided in the Z Data Tools COBOL compiler.

Using a licensed site COBOL compiler

By default, the Z Data Tools internal compiler will be used.

If you wish to use the licensed site compiler, there are two methods:

- Before invoking Z Data Tools, allocate ddname HFMCOB to the desired compiler library, usually hlq.SIGYCOMP, or
- Create a FACILTY Class Profile FILEM.COBOL.EXTERNAL with CONTROL access or higher. See Controlling the use of the COBOL compiler on page 76.

Note: The COBOL compiler requires a region size of at least 200 MB, and possibly more depending on the complexity of the copybook being compiled.

Adding the COBOL compiler library to the LINKLIST

If you added HFM.SHFMMOD1 to your LINKLIST, also add the COBOL compiler library to the LINKLIST.

If you did not add HFM.SHFMMOD1 to your LINKLIST, but instead added this library to the STEPLIB DD statement in your TSO logon procedure or your batch job, also add the supported COBOL compiler library to the STEPLIB DD statement.

As an alternative to adding the COBOL compiler library to the LINKLIST or the STEPLIB DD, you can provide access to the COBOL compiler library by allocating an HFMCOB DD for your COBOL compiler library:

```
//HFMCOB DD DSN=your.complib,DISP=SHR
```

You can allocate this DD statement dynamically, in your TSO logon procedure, or in a batch job.

Adding the COBOL compiler library to the batch JCL skeletons

You might also want to add your COBOL compiler library to the STEPLIB DD statements in the Z Data Tools batch JCL skeleton, HFMFTEXC, and the ZDT/IMS batch JCL skeleton, HFM1FTEX. For more information about these JCL skeletons, see Changing the JCL skeleton for batch mode on page 57 and Tailoring the job control skeletons on page 279.

Authorizing the COBOL compiler library for batch processing

If you are running Z Data Tools APF-authorized in batch mode, and you want to run Z Data Tools batch jobs using COBOL copybook templates, then you **MUST** also authorize the COBOL compiler library. Alternatively, run your Z Data Tools batch jobs un-authorized. (If you have not authorized your COBOL compiler library and it is in a JOBLIB or STEPLIB concatenation with your Z Data Tools load library, then Z Data Tools will not be APF-authorized in batch.)

Suppressing COBOL compiler warning messages

If you set TERM=YES (TERM) as a default COBOL compiler option, add an allocation for HFMTERM DD to your TSO logon procedure,

```
//HFMTERM DD DUMMY
```

This will suppress unnecessary compiler warning messages.

Using COBOL compiler options with Z Data Tools

Z Data Tools overrides certain COBOL compiler options when it invokes the COBOL compiler. Therefore these options cannot be marked as 'fixed' in the COBOL default options module IGYCDOPT. These options are:

```
ADATA          LIB
ADEXIT         LINECOUNT
ARITH          MSGEXIT
COMPILE        OBJECT
DBCS           OPTIMIZE
DECK           PRTEXIT
FLAG           SQL
INEXIT         STGOPT
```

If you have any of these options fixed at your site, you will need to build a Z Data Tools-specific COBOL options module that does not have these options fixed, and make this module available to the COBOL compiler invoked by Z Data Tools. For information on changing the COBOL default options and building an options module for a specific purpose, see the appropriate COBOL Customization book.

If you build a Z Data Tools-specific COBOL options module, but do not want to make it generally available other than to Z Data Tools, make a copy of your COBOL compiler library containing this options module. You can then make this copy of the COBOL compiler library available to Z Data Tools only, by means of the HFMCOB DD statement, allocated dynamically, in your TSO logon procedure, or in a batch job.

Customizing for processing PL/I include books

Z Data Tools provides a choice of two internal PL/I compilers that are used in any addition to external PL/I compiler previously installed.

The Z Data Tools internal PL/I compiler to be used is selected with the ALTPLI option. See ALTPLI on page 385.

The SHOWPLI command can be used to display the PL/I compiler being used.

Z Data Tools uses its internal PL/I compilers instead of the licensed site compiler. Use the ALTPLI option to select the version of the internal compiler. For more information, see the Table 71: Available COBOL compilers on page 385 for details.

For information about using a licensed site PL/I compiler, see Using a licensed site COBOL compiler on page 45. For information about using the Z Data Tools internal PL/I compiler, see Using the Z Data Tools COBOL compiler on page 44.

If you want to use Z Data Tools view, edit, copy, print, or data create functions with a PL/I include book, you must use the Language Environment® runtime library.

To ensure that you use the correct Language Environment® runtime, either add the Language Environment® runtime library, SCEERUN, to your LINKLIST, or add it to the STEPLIB DD statement in your TSO logon procedure.

Table 7. PL/I compiler order of precedence

	Option	Version	PL/I Module	Notes
HFMPL/I	DD DUMMY	Internal	HFM provided	Allocated in JCL or TSO command
HFMPL/I	DSN=library	External	HCLPL/I	Allocated in JCL or TSO command
RACROUTE	ATUS=ACCESS FACILITY Class PROFILE FILEM.PLI.EXTERN AL CONTROL access or higher	External	HCLPL/I	Force Z Data Tools to use an external PL/I compiler
If none of the above		Internal		Default. See ALTPLI on page 385.

SHOWPLI command

The SHOWPLI command displays the level of the internal PL/I compiler or if the usage of external PL/I compiler is in effect.

In batch, submit a Z Data Tools batch job with the following control statement:

```
$$FILEM SHOWPLI
```

If a licensed PL/I compiler is available, the output includes the following message: HFMNB920I The Customer Licensed PL/I Compiler will be used.

If the Z Data Tools supplied PL/I compiler is used, the output includes the following message: HFMNB925I The File Manager supplied PL/I Compiler will be used.

Under ISPF, enter SHOWPLI on any Z Data Tools command line alternatively the following messages, according to the PL/I complier in use.

```
Using ZDT PL/I V61
The Z Data Tools supplied Vxx PL/I Compiler will be used
```

Or

```
HFMPLI is allocated
The licensed PL/I Compiler in library dataset name allocated via HFMPLI will be used.
```

Using a licensed site PL/I compiler

By default, Z Data Tools uses the internal PL/I compiler.

To use the licensed site compiler, use one of the following methods:

- Before you invoke Z Data Tools, allocate the `FMNPLI` DD to the required compiler library, usually `hlq.SIBMZCMP`.
- Create a FACILITY class profile named `FILEM.PLI.EXTERNAL` with `CONTROL` access or higher. For more information, see **Controlling the use of the PL/I compiler**.

Note: The PL/I compiler requires a region size of at least 50 MB. Complex copybooks might require a larger region size.

Adding the PL/I compiler library to the LINKLIST

If you add `FMN.SFMNMOD1` to the LINKLIST, also add the supported PL/I compiler library to the LINKLIST.

If you do not add `FMN.SFMNMOD1` to the LINKLIST and instead add it to the `STEPLIB` DD statement in your TSO logon procedure or batch job, also add the supported PL/I compiler library to the `STEPLIB` DD statement.

As an alternative to adding the PL/I compiler library to the LINKLIST or the `STEPLIB` DD, you can provide access to the PL/I compiler library by allocating an HFM DD for your PL/I compiler library:

```
//HFMPLI DD DSN=your.complib,DISP=SHR
```

You can allocate this DD statement dynamically, in your TSO logon procedure, or in a batch job.

Adding the PL/I compiler library to the batch JCL skeletons

You can also add the PL/I compiler library to the `STEPLIB` DD statements in the Z Data Tools batch JCL skeleton, `FMNFTEXC`, and the ZDT/IMS batch JCL skeleton, `FMN1FTEX`.

For more information about these JCL skeletons, see Changing the JCL skeletons for batch mode on page 181 and Tailoring the job control skeletons on page 279.

Authorizing the PL/I compiler library for batch processing

If Z Data Tools runs APF-authorized in batch mode and you use PL/I copybook templates, you must also authorize the PL/I compiler library.

Alternatively, run Z Data Tools batch jobs without APF authorization.

If the PL/I compiler library is not authorized and is included in the `JOBLIB` or `STEPLIB` concatenation with the Z Data Tools load library, Z Data Tools does not run APF-authorized in batch mode.

Suppressing PL/I compiler warning messages

If `TERM=YES` (`TERM`) is set as a default PL/I compiler option, add the following allocation to your TSO logon procedure:

```
//FMNTERM DD DUMMY
```

This allocation suppresses unnecessary compiler warning messages.

Customizing for processing HLASM copybooks

If you want to use Z Data Tools view, edit, copy, print, or data create functions with a HLASM copybook, ensure that the HLASM load library, ASM.SASMMOD1, is available to Z Data Tools. You can do this either by adding it to your linklist, or to the STEPLIB DD statement in your batch job.

To use HLASM in a batch job, specify LANG=HLASM on the Z Data Tools function statement. For example, to print a member of a data set, using an HLASM copybook, you would specify a Z Data Tools function statement similar to:

```
$$FILEM DSP FORMAT=TABL,TCIN='HFM.SHFMSAM1(HFMACPY)',LANG=HLASM
```

Customizing to use DFSORT to improve Z Data Tools performance

Z Data Tools can use DFSORT to improve the performance of the Data Set Copy and Data Set Print functions under ISPF and in batch.

If DFSORT is enabled, its COPY function will be used to copy data to Auxiliary Storage (VSAM RRDS) when editing large data sets under ISPF. Z Data Tools accesses DFSORT using the aliases ICEDFSRT and ICEDFSRB.

Z Data Tools will use DFSORT provided it can find DFSORT and verify that it is at the correct maintenance level for Z Data Tools use. DFSORT is an optional, priced feature of z/OS®; you must have a DFSORT license to use DFSORT outside Z Data Tools. However, the DFSORT code is always shipped with z/OS®; as long as you did not delete the DFSORT libraries when you installed z/OS®, you can enable Z Data Tools to use the DFSORT code.

The following topics describe how you can customize Z Data Tools to use DFSORT or a sort product from an independent software vendor.

Customizing when DFSORT is the primary sort product

If you use DFSORT as your primary product, you will have installed it in one of the following ways:

- **Resident:** The DFSORT library, SORTLPA, is in LPALST, and the DFSORT library, SICELINK, is in LINKLIST.
- **Non-Resident:** The DFSORT libraries SICELINK and SORTLPA are private libraries.

1. If you have installed DFSORT resident, you need take no further action to make DFSORT available to Z Data Tools.
2. If DFSORT is installed non-resident, you will need to take different actions to enable Z Data Tools to use DFSORT under ISPF and in batch.

   Under ISPF (interactively)

   To enable Z Data Tools to use DFSORT under ISPF, add the DFSORT libraries, SICELINK and SORTLPA, in that order, to the STEPLIB DD statement in your TSO logon procedure. (If you do not want to add these libraries to your TSO logon procedure, you can add them via the TSOLIB command, before invoking ISPF.)

In batch

To enable Z Data Tools to use DFSORT in batch, add the DFSORT libraries, SICELINK and SORTLPA, in that order, to the JOBLIB or STEPLIB DD statement for the job or step that uses Z Data Tools. You might want to create a Z Data Tools JCL procedure for use at your site.

You could also add the DFSORT libraries to the STEPLIB DD statement in the batch JCL skeleton. If you do this, you must add them in the order SICELINK, SORTLPA. For more information about the batch JCL skeleton, see Changing the JCL skeleton for batch mode on page 57.

Customizing when DFSORT is not the primary sort product

If you use a non-IBM sort product as your primary sort product, and you want to enable Z Data Tools to use DFSORT, you **must** place the DFSORT libraries **after** your primary sort product libraries in the system search order.

Note: DFSORT and non-IBM sort products typically have some entry points with the same names, for example SORT and ICEMAN. Therefore if the DFSORT libraries are placed before your primary sort products libraries, DFSORT will become the primary sort product. In this situation, if you do not have a license for DFSORT all sort jobs that use your primary sort product will fail.

If you use a non-IBM sort product as your primary sort product, you will have installed it in one of the following ways:

- **Resident:** Your primary sort products libraries are in LPALST or the LINKLIST or both.
- **Non-Resident:** Your primary sort products libraries are private libraries.

1. If you have installed your sort product resident, place DFSORT's SORTLPA and SICELINK libraries in the LINKLIST, (**after** your primary sort product libraries in the LINKLIST, if any), to enable Z Data Tools to use DFSORT.
2. If you have installed your sort product non-resident, you will need to take different actions to enable Z Data Tools to use DFSORT under ISPF and in batch.

   Under ISPF (interactively)

   To enable Z Data Tools to use DFSORT under ISPF, add the DFSORT libraries, SICELINK and SORTLPA, in that order, **after** your primary sort product libraries, to the STEPLIB DD statement in your TSO logon procedure. (If you do not want to add these libraries to your TSO logon procedure, you can add them via the TSOLIB command, before invoking ISPF.)

   In batch

   To enable Z Data Tools to use DFSORT in batch, add the DFSORT libraries, SICELINK and SORTLPA, in that order, to the JOBLIB or STEPLIB DD statement for the job or step that uses Z Data Tools, **after** your primary sort products libraries. You might want to create a Z Data Tools JCL procedure for use at your site.

You could also add the DFSORT libraries to the STEPLIB DD statement in the batch JCL skeleton. If you do this, you must also add your primary sort products libraries. You must add them in this order, your sort product libraries, followed by SICELINK, then SORTLPA. For more information about the batch JCL skeleton, see Changing the JCL skeleton for batch mode on page 57.

DFSORT uses an SVC in the SORTLPA library. By default this is SVC 109. This SVC is used by DFSORT to record SMF type-16 records or to use IBM® cached DASD devices (such as the 3990 model 3). If your non-IBM sort product uses its own version of SVC 109, you will need to make an alternative SVC number the default SVC for DFSORT. For information on installing DFSORT's SVC, see DFSORT Installation and Customization for your release of DFSORT.

Z Data Tools does not need the DFSORT SVC for its own operations. Therefore, if DFSORT if not your primary sort product, it is not necessary to install the DFSORT SVC.

Determining if DFSORT is being used

Use the SHOWSORT command to check if Z Data Tools is using DFSORT.

If you expect Z Data Tools to be using DFSORT to improve performance, and it does not appear to be doing so, you can check this both in batch and under ISPF.

In batch, submit a Z Data Tools batch job with this control statement:

```
$$FILEM SHOWSORT
```

If DFSORT is available to Z Data Tools, the job output will show:

```
SORT debugging is on
```

If DFSORT is not available, the job output will show:

```
DFSORT not available
```

Under ISPF, enter SHOWSORT on any Z Data Tools command line. If DFSORT is available to Z Data Tools, this message is displayed: `SORT debugging is on`. If DFSORT is not available, this message is displayed: `DFSORT not available`.

Summary

Once you have determined the correct way to implement DFSORT for use by Z Data Tools at your site, follow the instructions in DFSORT Installation and Customization. You do not need to customize the DFSORT COPY function because Z Data Tools does this automatically. Z Data Tools does not use the DFSORT SORT or MERGE functions.

Note: When Z Data Tools uses DFSORT it presumes that DFSORT will produce message output and will open data set HFMSRTP for DFSORT messages. If the DFSORT installation option MSGPRT is set to NONE (MSGPRT=NONE) then

an OPEN abend may occur when the Z Data Tools DFSORT interface module (HFMDFSRT) tries to open message data set HFMSRTP. To avoid this set the DFSORT installation option MSGPRT to ALL (MSGPRT=ALL).

Binding Db2® to use Z Data Tools object access method (OAM) functions

If you intend to use Z Data Tools OAM functions, you must bind the Z Data Tools plan, HFMODIRS, into the Db2® system where the OAM objects are defined, to allow the proper working of OAM functions. The following sample jobs are provided in HFM.SHFMSAM1 and can be used to bind the HFMBDIRS plan/collection into a Db2® subsystem.

- **HFMBDI2K**: Db2® V12 JOB to bind the collection associated with plan HFMBDIRS.
- **HFMBDI2P**: Db2® V12 JOB to bind the plan HFMBDIRS.
- **HFMBDI3K**: Db2® V13 JOB to bind the collection associated with plan HFMBDIRS.
- **HFMBDI3P**: Db2® V13 JOB to bind the plan HFMBDIRS.

Customize the sample jobs according to your Db2® installation requirements. See the instructions in the sample jobs for more information about changes you need to make.

The DBRM associated with the plan HFMBDIRS is distributed in HFM.SHFMDBRM.

The following Db2® authorities must also be granted to each user of OAM functions:

- Directory processing requires READ authority for the following tables:
  - GROUP*nn*.OSM_OBJ_DIR (all group directories) *nn* = 00 … 99 inclusive
  - SYSIBM.SYSTABLES
  - *owner*.VOLUME
  - OAMADMIN.CBR_COLLECTION_TBL
  - OAMADMIN.CBR_MGT_CLASS_TBL
  - OAMADMIN.CBR_STO_CLASS_TBL
- Restoring objects requires UPDATE authority for the following tables:
  - GROUP*nn*.OSM_OBJ_DIR (all group directories) *nn* = 00 … 99 inclusive
  - OAMADMIN.CBR_COLLECTION_TBL

Disabling Websphere MQ feature by system name

You can disable the usage of Z Data Tools Websphere MQ support by using the following FACILITY class profile if you have Security Server RACF® 1.9 (or later) or an equivalent security product.

HFMMQ.DISABLE.*system_name*

Where:

system_name

The value specified on the SYSNAME parameter in the IEASYMxx parmlib member. The value for any given system is displayed in the System ID field on the Z Data Tools primary option menu or alternatively the ISPF primary option panel ISR@PRIM.

If the profile has been defined and a user has READ or higher access to the profile name then all Websphere MQ functionality within Z Data Tools will be disabled for that user.

Example 1. Define a profile to disable all users on the system that is named HFMPROD except *myuser*.

```
RDEF FACILITY HFMMQ.DISABLE.HFMPROD AUDIT(NONE) +
   UACC(READ) OWNER(ownerid)
PE HFMMQ.DISABLE.HFMPROD +
   CLASS(FACILITY) ID(myuser) ACC(NONE)
```

Example 2. Define a profile to enable all users on the system that is named HFMPROD except *myuser*.

```
RDEF FACILITY HFMMQ.DISABLE.HFMPROD AUDIT(NONE) +
   UACC(NONE) OWNER(ownerid)
PE HFMMQ.DISABLE.HFMPROD +
   CLASS(FACILITY) ID(myuser) ACC(READ)
```

Running Z Data Tools with APF-authorization

In batch mode, Z Data Tools can run APF-authorized or non APF-authorized. Z Data Tools cannot run APF-authorized under ISPF. You make Z Data Tools authorized by adding HFM.SHFMMOD1 to your site-specific IEAAPF*xx* or PROG*xx* member in SYS1.PARMLIB.

Note:

1.

   If you are running Z Data Tools APF-authorized in batch mode, and you want to run Z Data Tools batch jobs using COBOL copybook templates, then you **MUST** also authorize the COBOL compiler library. If you do not do this, you will receive an ABENDS306 from the batch job.
2. If you plan to access source code in a library management system that does not provide a SUBSYS interface, then you cannot run Z Data Tools APF-authorized.

To activate any changes you have made to SYS1.PARMLIB members, either restart your system, or use the appropriate commands for your site to dynamically activate the changes.

Authorizing the PL/I Compiler Library for batch processing

If Z Data Tools is running APF-authorized in batch mode and batch jobs use PL/I include book templates, the external PL/I compiler library must also be APF-authorized. Otherwise, the batch job ends with an `ABENDS306`.

Troubleshooting problems with APF-authorization

If you expect Z Data Tools to be running APF-authorized in batch mode, and it does not appear to be doing so, you can check this by submitting a Z Data Tools batch job with the control statement:

```
$$FILEM VER
```

If the VER function indicates that Z Data Tools is not APF-authorized, it means that HFM.SHFMMOD1 is not APF-authorized. Check that IEAAPF*xx* is set up correctly and selected in IEASYS*xx*. You will need to re-IPL to activate IEAAPF*xx*. Alternatively, you can use PROG*xx* (if it is available on your system).

If a JOBLIB or STEPLIB statement is used to specify the Z Data Tools load library, ensure that the Z Data Tools load library is not concatenated with a non APF-authorized library. If a library concatenation includes a non-authorized library, (for example, you have not authorized your COBOL compiler library), then no library in that concatenation will be authorized.

If you start Z Data Tools from an ISPF selection panel, the VER command will always show that Z Data Tools is not APF-authorized, as Z Data Tools cannot run APF-authorized under ISPF. Z Data Tools functions that require APF-authorization are not supported under ISPF.

If you plan to use SMF for audit logging you must ensure that the module HFMSMF is APF-authorized. You also need to provide an SMF number, either in the appropriate option's module or parmlib member. See Customizing Z Data Tools to write audit records to SMF on page 86.

The current Z Data Tools function is terminated when auditing to SMF fails.

Message *"Unable to link to HFMSMF. TSOLNK RC=20 (Dec) REASON CODE=56 (Dec)"* is diagnostic of an APF authorization failure. The most common causes are:

- The library containing HFMSMF is not APF authorized.
- The DDNAME to which the library containing HFMSMF is allocated includes other libraries and at least one of those libraries is not APF-authorized.

Language Environment® considerations

During the Z Data Tools Initialization process, Language Environment® (LE) preinitialization facilities (CEEPIPI) are used to create and initialize a common run-time environment.

This means that if LE is active when Z Data Tools is invoked, there are restrictions on Z Data Tools functionality because the CEEPIPI functions are not available.

Note that these restrictions **only** apply if LE is active when Z Data Tools is invoked. (LE active means that Z Data Tools has been invoked directly via an LE application (for example, from Enterprise COBOL).)

The restrictions are:

- TRAP(ON,NOSPIE) should be set to allow Z Data Tools abend handler to function correctly.
- An LE-enabled I/O exit cannot be used.
- MOD_DATE REXX Built-in function cannot be used.
- Scramble Exit (if used) must be in Assembler (non-LE).

Chapter 3. Customizing Z Data Tools

This chapter describes how to customize Z Data Tools. You do this after you have installed Z Data Tools.

Changing the default options

Default processing options are supplied with Z Data Tools in the module HFM0POPT. You can change these options to suit your installation requirements by using the usermod HFMUMODP. For a description of the options and the values you can specify, see Z Data Tools options on page 383.

The default macro statements are in HFM.SHFMSAM1(HFM0POPT).

You change the options as follows:

1. Copy the member HFM0POPT from HFM.SHFMSAM1 into your own source library.
2. Change the default options in the HFM0POPT member in your library as required.
3. Modify the HFMUMODP member in HFM.SHFMSAM1 to meet your site's requirements. Refer to the usermod for information about changes you might need to make.
4. Install SMP/E usermod HFMUMODP.

Note: If you do not want to use SMP/E you can use the sample job HFM0POPH to assemble HFM0POPT .

Setting the default national language

If you installed the Japanese component (or you have provided locally translated messages and panels), you can change the default national language used in the Z Data Tools ISPF and batch interfaces.

The language used by Z Data Tools under ISPF depends on the language setting for your ISPF session.

The language used by Z Data Tools for batch processing depends on the LANGUAGE option in HFM0POPT. See Table 18: Keyword values for the LANGUAGE option on page 108 for the value to specify for LANGUAGE. See LANGUAGE on page 403 for more information about the LANGUAGE option.

ZDT/CICS also uses the LANGUAGE option in HFM0POPT to determine the language for ISPF panels and messages. If you have installed the ZDT/CICS Japanese component, change the setting to that language. If you have provided ZDT/CICS messages and panels in another language, change the setting to that language.

Setting up Z Data Tools to use a template repository

A template repository is a VSAM file that contains cross-references of data sets, CICS® files (including CICS® TS and CICS® TD queues), path names, and WebSphere MQ queue names to their respective templates.

There are three ways to identify the template repository to a Z Data Tools session. The methods are evaluated in the following order and the first specification is used for the rest of the session.

- Specifying the repository name in the respective HFMxPARM PARMLIB member.
- Allocating the TPREPOS ddname to the data set name
- Customizing HFM4POPT. See Customizing miscellaneous options in HFM4POPT on page 383.

Specifying the repository data sets using PARMLIB options

You can specify one or more repository data sets in the PARMLIB member. Use this method to connect users to a repository data set name and to mandate templates or views for specified users.

- Use the <U *userid1*, *userid2*, … *useridn*> tag to connect specified users to a repository data set name.
- Use the <X *xfacilit.profile*> tag to apply the rule to all users who have read access or more to a specified XFACILIT profile.

Enter the repository definitions in order from the most specific to the most general. The first definition that matches the current user will be used.

Syntax

REPOS [**DSN** = *dsname*] [{ **MANDATORY** = **NO** | **MANDATORY** = **YES** }] [{ **RESTRICT** = **NO** | **RESTRICT** = **YES** }]

Where:

DSN

The name of the repository data set.

MANDATORY

Specify YES if the repository identifies resources for the user that have templates or views that are to be used to anonymize or scramble the data.

Note: To enable dynamic scrambling through the TESTMASK command the resource entry must be marked as mandatory.

RESTRICT

Specify YES to restrict usage to the resources specified in the repository.

Example

Example 1

Enter this PARMLIB definition to set up a mandatory repository for three users and default repository for the rest.

```
<U FRED, BOBBYB, BIILBO>
REPOS DSN=MASK.CREDIT.CARDS.REPOS,MANDATORY=YES
</U>
REPOS DSN=DEFAULT.REPOS
```

Example

Example 2

Enter this PARMLIB definition to set up a mandatory repository for any user having read access or more to the MY.MANDATED.USERS XFACILIT class profile.

```
<X MY.MANDATED.USERS>
REPOS DSN=MASK.CREDIT.CARDS.REPOS,MANDATORY=YES
</X>
REPOS DSN=DEFAULT.REPOS
```

Allocating TPREPOS

The ddname TPREPOS can be allocated to a TSO/ISPF session or specified as a DD card to a batch job.

JCL:

```
//DD TPREPOS DISP=SHR,DSN=hlq.TEMPLATE.REPOSTRY
```

TSO command:

```
'ALLOC FI(TPREPOS) DA('hlq.TEMPLATE.REPOSTRY') SHR REUSE'
```

You can also add a DD specification to the CONFIG=ZDT for Z Data Tools tasks that require the ZCC server:

TPREPOS=HFM.TEMPLATE.REPOSTRY

Changing the JCL skeleton for batch mode

Several functions in Z Data Tools are available in batch mode and the TSO batch environment. For these functions to run successfully, appropriate job control language statements must be provided. This is done by means of the set processing options, and a job control skeleton. The job control skeleton for Z Data Tools is the member HFMFTEXC in HFM.SHFMSLIB.

HFMFTEXC

There are some DD statements in HFMFTEXC that you may need to change. These statements are commented out in the distributed version of the skeleton. They are documented below.

```
//STEPLIB  DD DSN=&HFMSMOD1,DISP=SHR
//* If using WMQ queues via batch, uncomment and update MQHLQ to suit
//* your sites WMQ high level qualifier
//*        DD DSN=MQHLQ.SCSQLOAD,DISP=SHR
//*        DD DSN=MQHLQ.SCSQANLE,DISP=SHR
//*        DD DSN=MQHLQ.SCSQAUTH,DISP=SHR
//*        DD DSN=IGY.SIGYCOMP,DISP=SHR
//*HFMCOB  DD DUMMY     Uncomment to force use of ZDT COBOL Compiler
//*HFMCLERR DD SYSOUT=* Uncomment to force output of Compiler listing
```

HFMFTEXC assigns a STEPLIB DD statement to the Z Data Tools load library. HFMFTEXC assumes that you have installed Z Data Tools into the default target libraries, and that the load library is HFM.SHFMMOD1. If you have installed Z Data Tools into a different library than HFM.SHFMMOD1, you may either change the LOADLIB parameter in the HFM0POPT module or change the //STEPLIB DD statement in the HFMFTEXC skeleton.

If necessary, change &HFMSMOD1 in this statement to the name of your Z Data Tools load library.

```
//STEPLIB  DD DSN=&HFMSMOD1,DISP=SHR
```

Z Data Tools can process WebSphere® MQ queues in both online and batch scenarios for the common utilities; create, copy, compare and print. To interface with WebSphere®, the libraries SCSQLOAD, SCSQANLE, and SCSQAUTH must be made available to the JOBLIB, STEPLIB or linklist concatenation. You can do this by uncommenting these lines:

```
//*        DD DSN=MQHLQ.SCSQLOAD,DISP=SHR
//*        DD DSN=MQHLQ.SCSQANLE,DISP=SHR
//*        DD DSN=MQHLQ.SCSQAUTH,DISP=SHR
```

The STEPLIB DD statement in this skeleton also concatenates the following statement, which appears as a comment in HFMFTEXC.

```
//*        DD DSN=IGY.SIGYCOMP,DISP=SHR
```

In this statement, IGY.SIGYCOMP is the supported, licensed COBOL compiler library. Some batch functions which make use of COBOL require this library. If you did not add your supported COBOL compiler library to your LINKLIST, remove the * to uncomment the line, and change the DSN to the name of your supported COBOL compiler library. All currently supported versions of IBM® Enterprise COBOL for z/OS® and OS/390® are supported by Z Data Tools.

If you have created a COBOL compiler library for Z Data Tools with a special version of IGYCDOPT, you can use this DD statement to make it available to Z Data Tools for all batch jobs using COBOL templates. See Using COBOL compiler options with Z Data Tools on page 46 for information about a special version of IGYCDOPT for Z Data Tools.

If the licensed PL/I compiler is not in the LINKLIST, apply the same logic. Edit the STEPLIB DD statement to reference the licensed PL/I compiler library.

If you do not have a supported COBOL compiler available to Z Data Tools (for example, if you are using Z Data Tools in a production environment, where the COBOL compiler is not used), Z Data Tools can use its own internal COBOL compiler. To enable this feature, remove the * to uncomment the third statement:

```
//*HFMCOB  DD DUMMY     Uncomment to force use of ZDT COBOL Compiler
```

If Z Data Tools encounters a statement, //HFMCOB DD DUMMY, the Z Data Tools COBOL compiler will be used. For information about the Z Data Tools COBOL compiler, see Using the Z Data Tools COBOL compiler on page 44.

If a supported PL/I compiler is not available to Z Data Tools, for example, when Z Data Tools is used in a production environment where the PL/I compiler is not used, Z Data Tools can use its internal PL/I compiler. To enable the internal PL/I compiler, remove the asterisk (*) to uncomment the following statement:

```
//*FMNPLI DD DUMMY      Uncomment to force use of FM PL/I Compiler
```

If Z Data Tools encounters the `//FMNPLI DD DUMMY` statement, it uses the Z Data Tools PL/I compiler. For information about the Z Data Tools PL/I compiler, see Customizing for processing PL/I include books on page 46.

If you want to produce a compile listing of copybooks processed by Z Data Tools functions, in case of compile errors, remove the * to uncomment the fourth statement:

```
//*HFMCLERR DD SYSOUT=* Uncomment to force output of Compiler listing
```

You can also change the specification of SYSOUT on this statement, if necessary, to suit your site's requirements.

If you plan to enable Z Data Tools to use the DFSORT COPY function to improve Z Data Tools performance, you might want to add the DFSORT libraries to this STEPLIB DD statement. If you do this, you must add them in the order SICELINK, followed by SORTLPA. If you choose to do this and DFSORT is not your primary sort product, you must also add your sort products libraries in front of the DFSORT libraries. For more information about using DFSORT to improve Z Data Tools performance, see Customizing to use DFSORT to improve Z Data Tools performance on page 49.

If you plan to enable Z Data Tools to access COBOL copybooks, PL/I include books, or HLASM copybooks in library management system (LMS) libraries, and you plan to link edit your LMS exit, HFMCRAEX, into your own load library, you might want to add this load library to the STEPLIB DD statement in the JCL skeleton. For more information about enabling Z Data Tools to access LMS libraries, see Customizing Z Data Tools to use library management system libraries on page 120.

You use the usermod HFMUMODB to modify the job control skeleton. HFMUMODB is distributed in HFM.SHFMSAM1. To do this:

1. Copy the HFMFTEXC member from HFM.SHFMSLIB to your own source library.
2. Modify the HFMFTEXC member in your own library. Change the name of the Z Data Tools load library on the first line of the STEPLIB DD statement to the name of your load library, if necessary. Uncomment the second line of the STEPLIB DD statement if you want to add your COBOL compiler library, and change the DSN to the name of your COBOL compiler library, if necessary. Uncomment the WMQ lines, if necessary. Add your sort and LMS exit libraries if necessary.
3. Modify the HFMUMODB member in HFM.SHFMSAM1 to meet your site's requirements. Refer to the usermod for information about changes you might need to make.
4. Install SMP/E usermod HFMUMODB.

Note: Z Data Tools does not provide support for the automatic generation of job routing control statements in the JCL generated by HFMFTEXC.

Customizing Z Data Tools batch return codes

Z Data Tools provides the facility for you to control the value of return codes issued by the Z Data Tools batch utilities. You can set return codes to values more suitable for your site, for example, to suit your job control environment. Return codes cannot be changed on an ad hoc basis. They are established for the whole installation.

You can customize selected return codes issued by the following batch utilities:

- Data set copy (DSC)
- Data set generate (DSG)
- Data set compare (DSM)
- Data set print (DSP)
- Data set update (DSU, and including DSEB)
- Display VTOC (DVT)
- Find/change (FCH)

You customize the batch utility return codes using the macro statements sample member, HFM0RETC, and the usermod HFMUMODR, to create a load module, HFM0RETC. HFM0RETC in HFM.SHFMSAM1 shows the conditions that can be customized for each utility, with the default return codes for those conditions.

During initialization, Z Data Tools will attempt to load HFM0RETC, and if the module is found its contents will be used to build a table of customized condition return codes. If HFM0RETC is not found, all the customizable conditions will issue their default return codes.

To customize the return codes:

1. Copy the HFM0RETC member from HFM.SHFMSAM1 to your own source library.
2. Modify the HFM0RETC member in your own library, as required. Change the number against the required conditions in HFM0RETC to the return code values you want those conditions to give. Note that you cannot change any condition's value to 16. The value 16 is reserved for use by Z Data Tools. Figure 9: Default HFM0RETC macro statements supplied with Z Data Tools on page 61 shows the default set of HFM0RETC macro statements.
3. Modify the HFMUMODR member in HFM.SHFMSAM1 to meet your site's requirements. Refer to the usermod for information about changes you might need to make.
4. Install SMP/E usermod HFMUMODR.

Figure 9. Default HFM0RETC macro statements supplied with Z Data Tools

```
HFM0RETC TITLE 'Z Data Tools : Return Codes (Batch Utilities)'
HFM0RETC CSECT
HFM0RETC AMODE 31
HFM0RETC RMODE ANY
         HFM0RETI FUNC=DSC,      DSC conditions                        X
               NORECSSOME=1,      No records copied for some members   X
               NORECSANY=2,       No records copied for any members    X
               NONESEL=4,         No records selected to copy          X
               NOMEMBERS=4,       No members to process                X
               EMPTY=4,           Empty input data set                 X
               SKIPPED=4,         Input packed, so skipped             X
               NOREPLACE=4,       NOREPLACE prevented copy             X
               NOCPYDUPE=2,       No copy - duplicate                  X
               NOCPYREXX=2,       No copy - REXX                       X
               RECTRC=0,          Records truncated                    X
               FLDTRC=0,          Fields  truncated                    X
               RECSHORT=8,        DSC external form output truncated   X
               REXXCONFLICT=3     REXX return message conflict
         HFM0RETI FUNC=DSG,      DSG conditions                        X
               NORECSOUT=4        Zero records specified
         HFM0RETI FUNC=DSM,      DSM conditions                        X
               NOTMATCH=1,        Compare sets not empty & don't match X
               ONEEMPTY=2,        One compare set empty                X
               SKIPPED=4,         Input packed, so skipped             X
               BOTHEMPTY=4,       Both compare sets empty              X
               INVMAP=4,          Invalid mapping                      X
               INVDATA=8          Invalid data                         X
               SYNCERR=8          Key Synchronization error
         HFM0RETI FUNC=DSP,      DSP conditions                        X
               MBRERR=1,          Some members not printed             X
               PRTERR=2,          Print error encountered              X
               NONESEL=4,         No records selected to print         X
               NOMEMBERS=4,       No members to process                X
               SKIPPED=4,         Input packed, so skipped             X
               EMPTY=4,           Empty input data set                 X
               NOPRTREXX=2,       No print - REXX                      X
               REXXCONFLICT=3     REXX return message conflict
         HFM0RETI FUNC=DSU,      DSU conditions                        X
               MBRNOTUPDT=1,      Some members not updated             X
               NOCHANGE=2,        Change failed                        X
               NONEUPDATED=4,     No records updated                   X
               NOMEMBERS=4,       No members to process                X
               SKIPPED=4,         Input packed, so skipped             X
               EMPTY=4            Empty input data set
         HFM0RETI FUNC=DVT,      DVT conditions                        X
               NOENTRY=4          No matching entries
         HFM0RETI FUNC=FCH,      FCH conditions                        X
               FSOME=1,           Some OK, some not OK                 X
               CFAIL=2,           Change failed                        X
               NOHIT=4,           No strings found to change           X
               NOMEMBERS=4,       No members to process                X
               SKIPPED=4,         Input packed, so skipped             X
               LOWSTOR=12,        Ran low on storage during FCH        X
               EMPTY=4            Empty input data set
         HFM0RETI END
         END   HFM0RETC
```

The return code from a batch utility will be the highest return code, (customized or otherwise), for all the conditions that arose during the execution of the utility. For example, suppose you have changed the return code for "No members to process" in the DSC utility from 4 to 9. If a condition arises that in isolation would result in return code 8 (normally an error), and the "No members to process" condition also occurs, the return code for the DSC utility will be 9.

The return code from a batch job executing more than one utility will continue to be the maximum of the codes, (customized or otherwise), returned by each of the utilities. The exception to this is that if any of the utilities encounters a terminating condition then the job step will terminate with a return code 16.

Note:

1. Several conditions can be present in a function concurrently. For example, using DSP with a template on an empty file may result in both the "empty input data set" condition, and the "no records selected to print" condition. If you customize a return code, be aware that a higher return code for another condition may override it or be over-ridden by it. If you customize one return code for a given function, review that function's other return codes for compatibility.
2. In batch, you might want Z Data Tools to abend rather than end with the original or customized non-zero return code, to prevent the execution of successive steps or jobs. You do this by setting the ABENDCC installation option to a value less than or equal to the return code. This will then be transformed to an abend (Abend 999, Reason Code=888 (hex: 378)). For more information, see ABENDCC on page 384, and also the ABENDCC option of the SET command, described in section *"SET (Set Processing Options)"* in *Z Data Tools User's Guide and Reference* .

Changing the print and display translation tables

By default, Z Data Tools translates unprintable characters in printed output to blanks (option PRTTRANS=ON), and non-displayable characters in displayed output to periods.

Z Data Tools provides default translation tables for English (HFMTRTBS), German (HFMTRDEU), and , and Japanese (FMNTRJPN).

For other languages you will need to provide and maintain your own print translation table and display translation table. See Customizing Z Data Tools for national languages on page 104 for information on customizing Z Data Tools for your language.

Sometimes you might want to display special characters on a terminal during a Z Data Tools session, or print Z Data Tools output in lowercase alphanumeric characters. To do this, you change the English translate table source member, HFM.SHFMSAM1(HFMTRTBS).

The translate tables that are used are:

TRPRT

used to translate unprintable characters to blanks in printed output. Change this table to include all the characters that you want to print.

TRBRW

used to translate non-displayable characters to periods in displayed output. Change this table to include all the characters that you want to display on your terminal.

TRUPC

used to translate lower case characters to uppercase characters for Z Data Tools command verbs and command keywords only. This table generally does not need to be customized in an SBCS environment.

TRUP

used to translate lower case characters to uppercase characters.

TRINV

used as a second-step translate table to translate characters that are not supported by your particular display terminal. Display data is first translated using the TRBRW table, and the result of that translate is translated using this table. You should not need to change this table.

Changing the translation tables in English

If you are using English, you use the usermod HFMUMODT and HFMTRTBS to change the print and display tables.

1. Check that the terminal on which you want to display Z Data Tools panels supports the display of special characters, or that the universal character buffer (UCB) of your printer has the characters you want to use.
2. Change the Z Data Tools options to specify PRTTRANS=ON. For information on how to do this, see Changing the default options on page 55.
3. Modify the Z Data Tools translation table as follows:
    a. Copy the member HFMTRTBS from HFM.SHFMSAM1 into your own source library.
    b. Change the translation table definition statements in HFMTRTBS in your source library, according to your requirements, as described above.
    c. Modify the HFMUMODT member in HFM.SHFMSAM1 to meet your site's requirements. Refer to the usermod for information about changes you might need to make.
    d. Install SMP/E usermod HFMUMODT.

To change the tables in languages other than English, see Customizing Z Data Tools for national languages on page 104.

Changing the ASCII translation tables

You can use Z Data Tools to translate tape data as follows:

- Translate tape input from ASCII format to EBCDIC format.
- Translate tape output from EBCDIC format to ASCII format.
- Translate tape input from ASCII format to EBCDIC format, and translate tape output from EBCDIC format to ASCII format.

If you want to use an ASCII or EBCDIC character set other than the character sets supplied by HCL, you can change the translation table definition statements.

You use the usermod HFMUMODA and the sample member HFMASCII to change the ASCII translation tables.

1. Copy the member HFMASCII from HFM.SHFMSAM1 to your own source library.
2. Change the translation table definition statements in HFMASCII in your source library, as required.
3. Modify the HFMUMODA member in HFM.SHFMSAM1 to meet your site's requirements. Refer to the usermod for information about changes you might need to make.
4. Install SMP/E usermod HFMUMODA.

Chapter 4. Customizing the Z Data Tools security environment

Z Data Tools provides security for system-oriented functions through RACF® (or an equivalent security product).

If RACF® or an equivalent security product is active, the System Authorization Facility (SAF) with the Z Data Tools enhanced security facility is used for access control and authorization verification. Authorization is controlled by Z Data Tools-specific profiles in the FACILITY class. See Setting up the security environment by using RACF or an equivalent security product on page 65 for information on defining profiles. If you use another security product than RACF®, consult the documentation for your product to determine how to define the FILEM facility to your product.

As a minimum, you should:

- Define the following profiles:

```
RDEFINE FACILITY FILEM.DISK.*          UACC(READ)   or NONE
RDEFINE FACILITY FILEM.DISK.FULLPACK   UACC(NONE)
RDEFINE FACILITY FILEM.LOADMOD.UPDATE UACC(READ)   or NONE
RDEFINE FACILITY FILEM.TAPE.*          UACC(READ)   or NONE
RDEFINE FACILITY FILEM.TAPE.BLP        UACC(NONE)
RDEFINE FACILITY FILEM.VSAM.*          UACC(READ)   or NONE
RDEFINE FACILITY FILEM.OAM.*           UACC(READ)   or NONE
RDEFINE FACILITY FILEM.OTHER.ALL       UACC(READ)   or NONE
```

- Define data set profiles for data sets allocated to HFMAUTH DD with UACC(NONE). For more details, see Preparing for Z Data Tools Remote Services on page 379.
- Set up your storage management routines so that data sets allocated to HFMAUTH DD are encrypted. For more details, see Preparing for Z Data Tools Remote Services on page 379.
- Transition from the use of ***PASSWORD=REMEMBER** to ***PASSWORD=PASSTICKET** in ZDT/CICS. ***PASSWORD=REMEMBER** will be deprecated in a future release. See sections Checklist for installing and customizing Z Data Tools to access CICS resources on page 326 and Modifying and submitting HFM3INST and HFM3PRFD on page 337 for further details.

Setting up the security environment by using RACF® or an equivalent security product

You perform the following steps to define profiles for RACF® or your equivalent security product. These steps assume that your security administrator has already controlled access to DASD volumes (DASDVOL) and facilities (FACILITY).

The following sections contain examples of setting up facility classes for Z Data Tools using RACF®. For more information about RACF® resource profiles, see *z/OS Security Server RACF Command Language Reference*.

Note: If you are using an equivalent security product, refer to that product's documentation for information on how to define and use facility classes.

You can give or deny some, or all, users access to any of the following groups of Z Data Tools functions:

FILEM.DISK.INPUT

Disk input functions

FILEM.DISK.UPDATE

Disk update functions

FILEM.TAPE.INPUT

Tape input functions

FILEM.TAPE.OUTPUT

Tape output functions

FILEM.TAPE.DUPLICATE

Tape copy functions

FILEM.TAPE.UPDATE

Tape update functions

FILEM.VSAM.UPDATE

VSAM update functions

FILEM.OAM.OUTPUT

OAM output functions

FILEM.OAM.UPDATE

OAM update functions

FILEM.LOADMOD.UPDATE

Load module update functions

FILEM.OTHER.ALL

All other functions

FILEM.TAPE.BLP

See Controlling Bypass Label Processing (BLP) on page 73

FILEM.DISK.FULLPACK

See Controlling fullpack access to DASD volumes on page 70

For more information about these groups, see Table 10: Z Data Tools function to profile name cross-reference on page 79.

Controlling access

Three facility groups are provided to allow you to control access to Z Data Tools Base function, and to ZDT/IMS and ZDT/Db2, from the ZDT/CICS primary option menu. These groups are:

FILEM.CICS.BASE

Access to Z Data Tools Base function

FILEM.CICS.IMS

Access to ZDT/IMS

FILEM.CICS.DB2

Access to ZDT/Db2

If a user ID running ZDT/CICS has read access to any of these groups, then the associated function (HFM, ZDT/IMS or ZDT/Db2) will appear on the ZDT/CICS primary option menu and the user can invoke these functions, if they are installed.

To achieve this Z Data Tools makes RACROUTE calls, with STATUS=ACCESS, to the CICS® SAF FACILITY profiles. When RACF® is used, the STATUS=ACCESS request works as documented, and no security-related logging or abends are generated, even if you do not have access to the profile.

However, when other security products such as ACF2 are used, an S047 abend may be issued in response to the above RACROUTE request. In this case you should consult the documentation for your security product and make changes accordingly.

If you have installed and customized the ZDT/CICS component, you should review your requirement for this access.

For more information about ZDT/CICS, see Customizing Z Data Tools CICS Component on page 324, and also the *Z Data Tools User's Guide and Reference for CICS*.

Protecting update functions

Three facility groups are also provided to enable you to protect update functions in Z Data Tools Base function, ZDT/Db2, and ZDT/CICS. They are:

FILEM.BASE.UPDATE

Protect update functions in the Z Data Tools Base component

FILEM.DB2.UPDATE

Protect update functions in ZDT/Db2

FILEM.CICS.UPDATE

Protect update functions in ZDT/CICS

This aspect of security is handled differently for ZDT/IMS. See IMS subsystems and ZDT/IMS functions access control facility on page 284.

These facility classes also require the option SEC=YES to be specified in HFM0POPT (for Z Data Tools base), HFM2POPT (for ZDT/Db2), and HFM3POPT (for ZDT/CICS). For information about the SEC option, see SEC on page 415. For more information about the protected functions, see Unprotected functions and profile names for protected functions on page 76. For a list of functions that are protected by this method, see Table 8: Z Data Tools unprotected functions on page 77, Customizing to protect update functions in ZDT/Db2 on page 183, and Customizing to protect update functions in ZDT/CICS on page 345.

If you do not specify SEC=YES in your options modules, then no checking of these facility classes is done.

Examples of giving or denying access

You can also give or deny some, or all, users access to an individual Z Data Tools function. The following examples illustrate this.

- To give universal access of NONE to a group of functions (for example, disk input functions), enter a RACF® command similar to this:

  ```
  RDEFINE FACILITY FILEM.DISK.INPUT     UACC(NONE)
  ```

  This means that no users can use any functions in the group unless otherwise specified.

- To give all users access to a group of functions (for example, tape input functions), enter a RACF® command similar to this:

  ```
  RDEFINE FACILITY FILEM.TAPE.INPUT     UACC(READ)
  ```

- To give a user (with user ID *userid*) access to a group of functions (for example, tape output functions), enter a RACF® command similar to this:

  ```
  PERMIT FILEM.TAPE.OUTPUT    CLASS(FACILITY) ID(userid) ACCESS(READ)
  ```

  Similarly, to deny a user access to tape output functions, enter a RACF® command similar to this:

  ```
  PERMIT FILEM.TAPE.OUTPUT    CLASS(FACILITY) ID(userid) ACCESS(NONE)
  ```

  The PERMIT statement for FILEM.TAPE.OUTPUT overrides the universal access that you specified for FILEM.TAPE.OUTPUT.

- To give a user access to a specific function (for example, the VSAM to Tape function), enter a RACF® command similar to this:

  ```
  PERMIT FILEM.FUNCTION.VT    CLASS(FACILITY) ID(userid) ACCESS(READ)
  ```

  Similarly, to deny a user access to the VT function, enter a RACF® command similar to this:

  ```
  PERMIT FILEM.FUNCTION.VT    CLASS(FACILITY) ID(userid) ACCESS(NONE)
  ```

  The PERMIT statement for FILEM.FUNCTION.VT overrides any access that you specified for FILEM.TAPE.OUTPUT.

- To give a user (with user ID *userid*) permission to update a load module, enter a RACF® command similar to this:

  ```
  PERMIT FILEM.FUNCTION.LMU   CLASS(FACILITY) ID(userid) ACCESS(READ)
  ```

  The PERMIT statement for FILEM.FUNCTION.LMU overrides any universal access that you specified for FILEM.LOADMOD.UPDATE.

- If the FACILITY class is not already active on your system, enter the following RACF® commands to activate it:

  ```
  SETROPTS CLASSACT(FACILITY)
  SETROPTS GENERIC(FACILITY)
  SETROPTS GENCMD(FACILITY)
  ```

Controlling access to Z Data Tools functions with SAF

SAF controls access to Z Data Tools functions as follows:

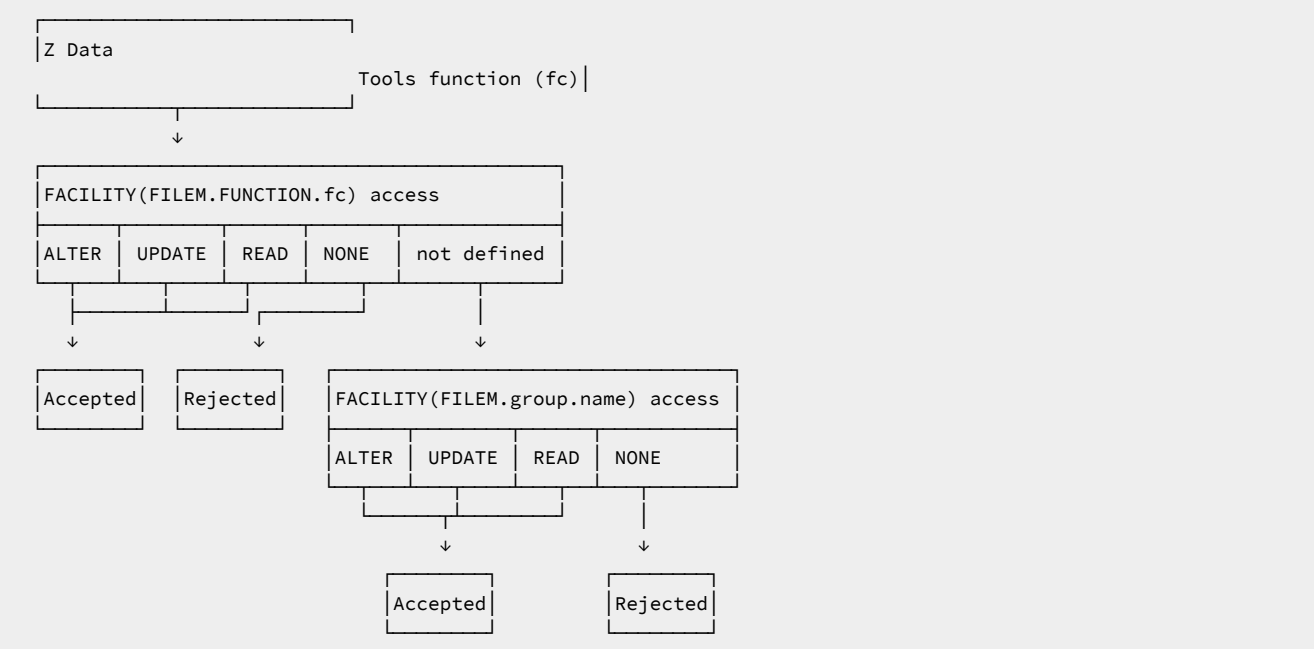
- If access to the profile FACILITY(FILEM.FUNCTION.*fc*) in the FACILITY class is defined (where *fc* is the function code), this controls access to the function.
- If access to the profile FACILITY(FILEM.FUNCTION.*fc*) in the FACILITY class is not defined, the profile name shown in Table 10: Z Data Tools function to profile name cross-reference on page 79 (in the form FILEM.*group.name*) is used.
- If no profile name as shown in Table 10: Z Data Tools function to profile name cross-reference on page 79 is defined, then FILEM.OTHER.ALL is used. If this does not permit access then access is denied.

Some Z Data Tools functions are protected, by default, by the FILEM.OTHER.ALL profile. These functions are listed in Table 9: Z Data Tools functions protected by FILEM.OTHER.ALL on page 78.

ALTER, UPDATE or READ access means that the user can use the function. Access NONE means that the user cannot use the function.

This is illustrated in Figure 10: Access to Z Data Tools functions on page 69.

Figure 10. Access to Z Data Tools functions



For example, the TP function is part of the FILEM.TAPE.INPUT group. You can control access to the TP function in any of the following ways:

- To give a user access to the TP function, regardless of the user's access to FILEM.TAPE.INPUT, give the user ALTER, UPDATE, or READ access to FACILITY(FILEM.FUNCTION.TP).
- To prevent a user from using the TP function, regardless of the user's access to FILEM.TAPE.INPUT, give the user NONE access to FACILITY(FILEM.FUNCTION.TP).

- To give a user access to any tape input function, unless overridden by a FILEM.FUNCTION.*fc* entry, give the user ALTER, UPDATE, or READ access to FACILITY(FILEM.TAPE.INPUT).
- To prevent a user from using any tape input function, unless overridden by a FILEM.FUNCTION.*fc* entry, give the user NONE access to FACILITY(FILEM.TAPE.INPUT).

Controlling the use of the PL/I compiler

A FACILITY CLASS profile is provided to allow you to control access to a licensed site PL/I compiler, IBMZPLI:

```
FILEM.PLI.EXTERNAL
```

If a user ID running Z Data Tools has `CONTROL` access or higher to this profile, Z Data Tools always uses the licensed site PL/I compiler, if available, to compile templates from PL/I copybooks.

If the profile does not exist, or if the user ID has less than `CONTROL` access to the profile, Z Data Tools uses the Z Data Tools PL/I compiler.

For more information, see Using a licensed site PL/I compiler on page 47 and Customizing for processing PL/I include books on page 46.

To determine whether the user ID has access to the profile, Z Data Tools issues `RACROUTE` requests with `STATUS=ACCESS` to the FACILITY profiles.

When RACF® is used, the `STATUS=ACCESS` request works as documented. No security-related logging or abends occur, even if the user ID does not have access to the profile.

When a non-RACF security product, such as ACF2, is used, an `ABEND S047` might occur in response to the `RACROUTE` request. If this occurs, consult the security product documentation and make the required changes.

Important information for users of non-IBM security products

Z Data Tools issues a RACROUTE TYPE=AUTH for the required profile, and if the required profile is not defined, RACF® returns RC=4. If RC=4 is returned, Z Data Tools issues a RACROUTE for FILEM.OTHER.ALL. If a higher return code is returned, no RACROUTE is issued for FILEM.OTHER.ALL and the access request fails immediately.

You should be aware that not all non-IBM security products issue RC=4 when a RACROUTE TYPE=AUTH for a profile fails; in this case no RACROUTE is issued for FILEM.OTHER.ALL and the access request fails immediately. If this applies to your security product, you will have to provide individual profiles for all Z Data Tools functions listed in Table 9: Z Data Tools functions protected by FILEM.OTHER.ALL on page 78.

Controlling fullpack access to DASD volumes

Z Data Tools provides the ability to work with the entire disk volume (disk fullpack). To do this specify a Z Data Tools disk function without specifying a data set name.

Access to disk fullpack processing is controlled as follows:

- If Z Data Tools is not running APF-authorized, disk fullpack processing is unavailable.
- If Z Data Tools is running APF-authorized, the user's access to the profile FILEM.DISK.FULLPACK in the FACILITY class is checked. The following access levels are possible:

  ALTER

  Read and update access to all volumes

  UPDATE

  Read access to all volumes, update access to specific volumes

  READ

  Read and update access to specific volumes

  NONE

  No fullpack access.

  Note: Since Z Data Tools cannot run APF-authorized under ISPF, and since disk fullpack processing is not available if Z Data Tools is not running APF-authorized, it follows that disk fullpack processing is not available under ISPF.

  Access to specific volumes is controlled with the DASDVOL class. The user needs READ access for disk read functions and ALTER access for disk update functions.

Figure 11: Fullpack processing for disk read functions on page 72 and Figure 12: Fullpack processing for disk update functions on page 73 show how this works for disk read and disk update functions.

Figure 11. Fullpack processing for disk read functions

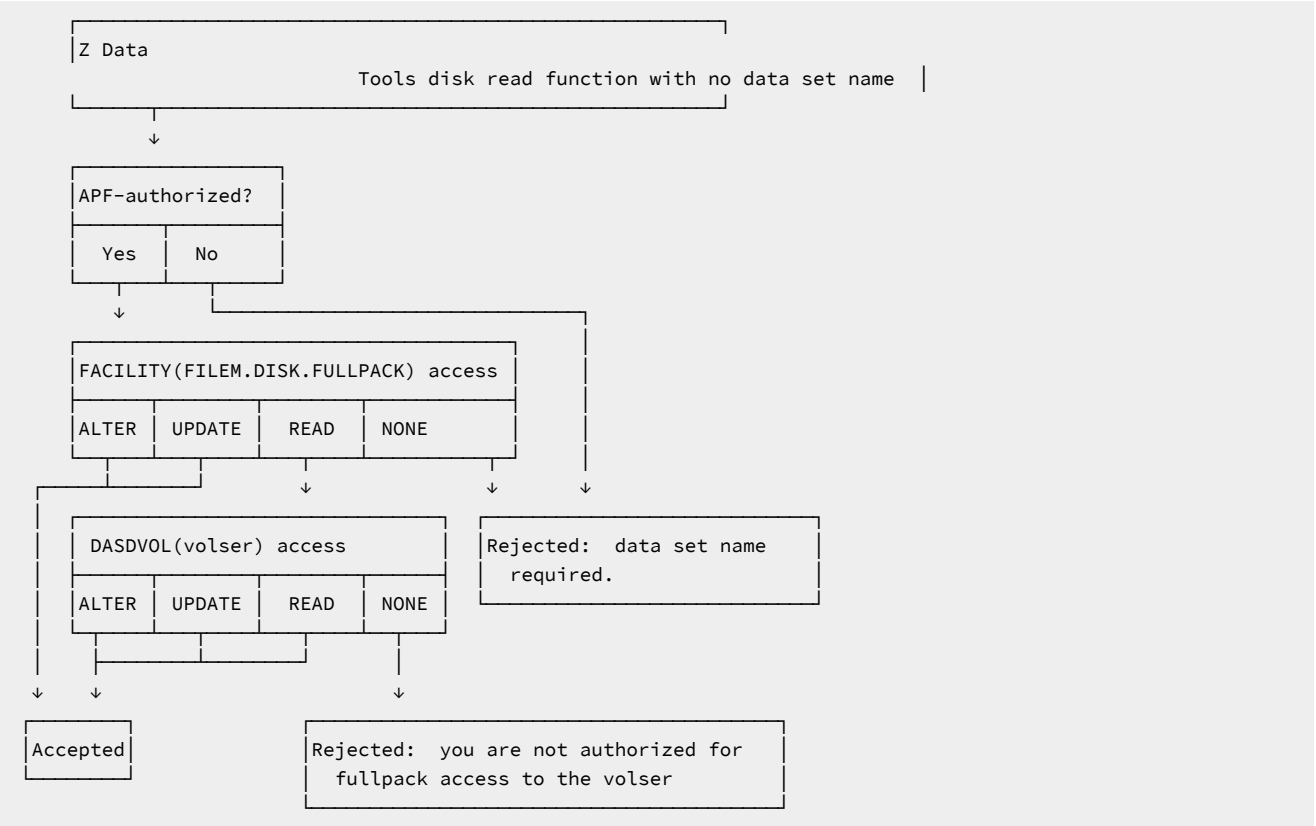
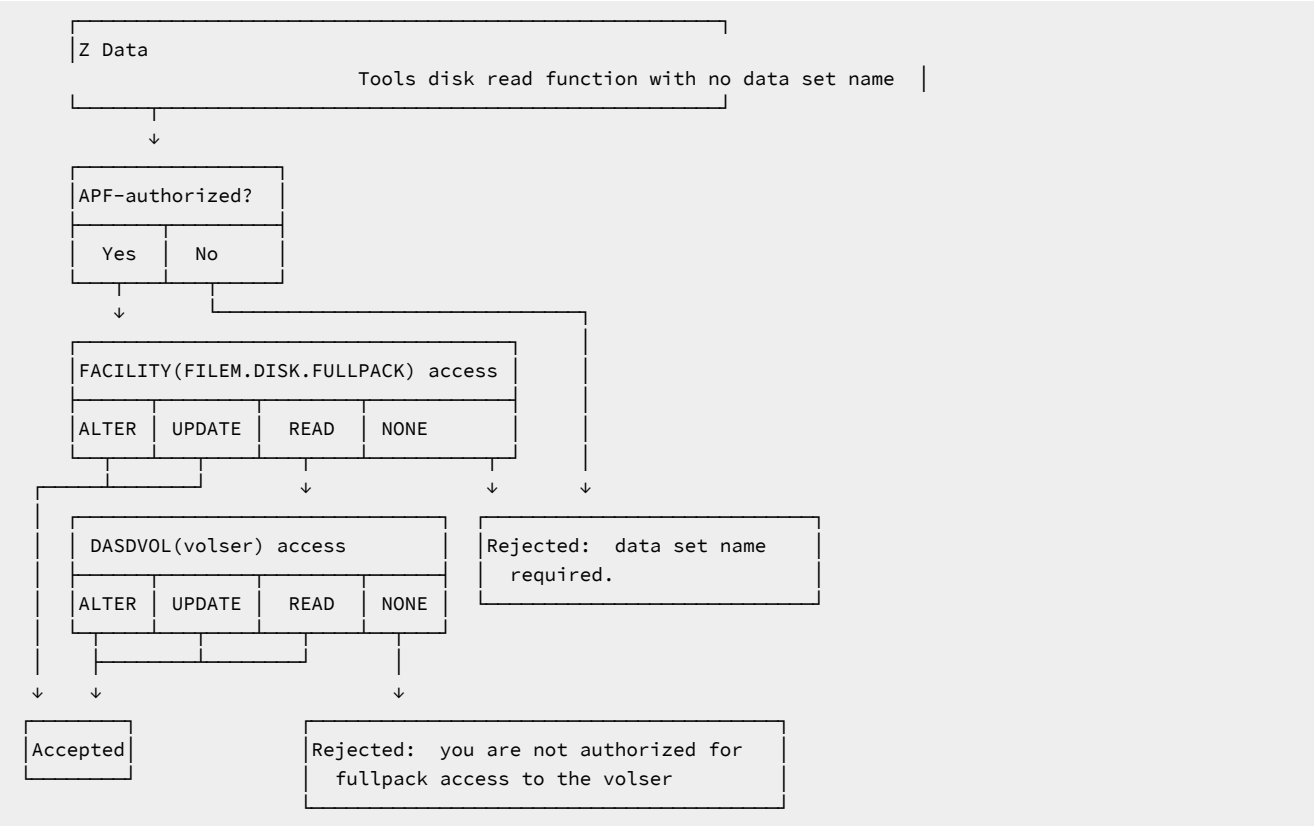


Figure 12. Fullpack processing for disk update functions



Controlling Bypass Label Processing (BLP)

You can use Z Data Tools to control tape label processing by your users even if your system installation parameters (JES parameters) would not allow BLP.

If your installation allows BLP usage (as specified in your JES parameters), normal open processing checks if the user has access to ICHBLP. Any user with READ access or greater can use BLP. If ICHBLP is not defined to your security product, all users can use BLP.

If your installation does not allow BLP usage (as specified in your JES parameters), then if LABEL=(,BLP) is coded on the JCL control statement, BLP is converted to NL. However, Z Data Tools users can still use BLP subject to the following conditions:

- Z Data Tools must be running APF-authorized.
- For any function other than TLB, the user must have access to FILEM.TAPE.BLP.
- If ICHBLP is defined to your security product, the user must have READ access or greater to it.

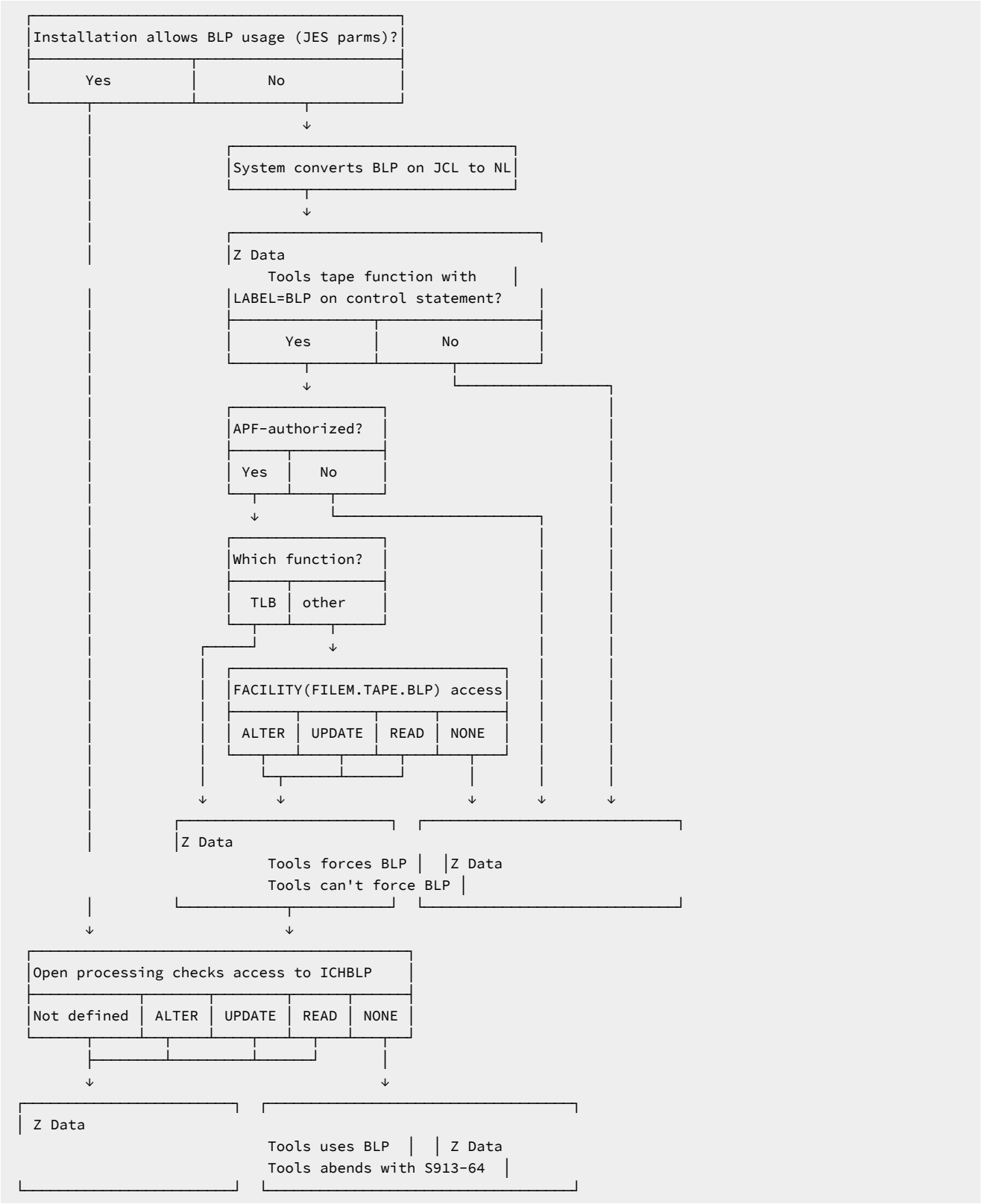
In this case, if either of the following is coded on the JCL control statement:

- LABEL=(,BLP)
- LABEL=(,NL)

the user can specify LABEL=BLP on the first Z Data Tools function that uses a tape. This means that the user wants to use bypass label processing (BLP) for the tape.

This is illustrated in Figure 13: BLP processing by Z Data Tools on page 75.

Figure 13. BLP processing by Z Data Tools



Controlling the use of the COBOL compiler

A FACILITY CLASS profile is provided to allow you to control access to a licensed site COBOL compiler, IGYCRCTL:

```
FILEM.COBOL.EXTERNAL
```

If a user ID running Z Data Tools has CONTROL access (or above) to this profile, then the licensed site compiler, if available, will always be used for compiling templates from COBOL copybooks. If this profile does not exist, or if the user ID has less than ACCESS CONTROL to this profile, the Z Data Tools COBOL compiler will be used. See Using a licensed site COBOL compiler on page 45 and Using the Z Data Tools COBOL compiler on page 44.

To achieve this, Z Data Tools makes RACROUTE calls, with STATUS=ACCESS, to the FACILITY profiles.

When RACF® is used, the STATUS=ACCESS request works as documented, and no security-related logging or abends are generated, even if you do not have access to the profile.

When a non-RACF security product such as ACF2 is used, an abend S047 might be issued in response to the above RACROUTE request. In this case, consult the product documentation and make changes accordingly.

Unprotected functions and profile names for protected functions

Table 8: Z Data Tools unprotected functions on page 77, Table 9: Z Data Tools functions protected by FILEM.OTHER.ALL on page 78, and Table 10: Z Data Tools function to profile name cross-reference on page 79, show Z Data Tools functions with their descriptions and (where applicable) the equivalent Z Data Tools panel you can use under ISPF.

Table 8: Z Data Tools unprotected functions on page 77 lists the functions that cannot be protected by RACF® or an equivalent security product.

However, there are exceptions to this. Some functions listed in Table 8: Z Data Tools unprotected functions on page 77 can be protected in Z Data Tools Base function by the facility class FILEM.BASE.UPDATE when the option SEC=YES is specified in HFM0POPT. These functions are indicated in Table 8: Z Data Tools unprotected functions on page 77 by ✓ in the fourth column.

Similarly these functions (and others) can be protected in ZDT/CICS by the facility class FILEM.CICS.UPDATE, when SEC=YES is specified in HFM3POPT. For more information about protecting the CICS® functions, see Customizing to protect update functions in ZDT/CICS on page 345.

The normal security checking of FILEM.FUNCTION.*function_code* applies to these functions when SEC=YES has been specified in HFM0POPT, HFM2POPT, or HFM3POPT.

Functions not listed in Table 8: Z Data Tools unprotected functions on page 77 are protected, and Table 10: Z Data Tools function to profile name cross-reference on page 79 shows the profile name used in the FACILITY class to protect a specific function.

All protected functions not listed in Table 10: Z Data Tools function to profile name cross-reference on page 79 are protected by the FILEM.OTHER.ALL profile.

In particular, the Z Data Tools functions listed in Table 9: Z Data Tools functions protected by FILEM.OTHER.ALL on page 78 are protected by default when you receive Z Data Tools. They are protected by the profile "FILEM.OTHER.ALL", and if you want to use these functions, you must first give the users access to these functions, by defining the profile "FILEM.OTHER.ALL" to RACF® or your equivalent security product.

For information on Z Data Tools profiles, and defining them to your security product, see Setting up the security environment by using RACF or an equivalent security product on page 65. In particular, if you use a non-IBM security product, see the warning in Controlling access to Z Data Tools functions with SAF on page 68.



Note: In Table 10: Z Data Tools function to profile name cross-reference on page 79, where more than one function code is listed for the same function, and the first function code is in **bold** letters, you should use this function code when defining the relevant function profile.

Table 8. Z Data Tools unprotected functions

Function	Equivalent panel	Description	Protect with SEC=YES
AF		Template Workbench	
APB	3.7	AFP Print Browse	
AUD		Print Audit Trail	
BTB		Batch template build	
BTU		Batch template update	
CLM	3.11.2	Load module compare	
DSC, COPY	3.3	Copy Utility	✓
DSE	2	Data Set Edit	✓
DSEB		Data Set Edit (batch)	✓
DSG	3.1	Data Create Utility	✓
DSI	3.4 I	Data Set Information	
DSM, DSCMP	3.11.1	Data Set Compare	
DSP	3.2	Print Utility	
DSU		Data Set Update (batch)	✓
DSV	1	Data Set View	
DSX	5.6	Data Set Extents	
DVT	3.5	Display VTOC	
DX		Decimal to hex conversion	

Table 8. Z Data Tools unprotected functions (continued)

Function	Equivalent panel	Description	Protect with SEC=YES
FCH	3.6	Find/Change Utility	✓
FMT	3.0	Set DBCS Format	
LVL		Show Z Data Tools Release and PTF Level	
MB, TST	3.8	Memory Browse	
NEW, NEWS		Show Z Data Tools Release News	
NODF, NOSORT		Do not use DFSORT	
PB	3.9	Print Browse	
PBK		Print/view copybook	
SCS	3.4	Catalog Services	
SET		Set Z Data Tools Processing Options	
VER		Show Z Data Tools Release and PTF Level	
VLM		View Load Module	
XD		Hex to decimal conversion	
(none)		Template Functions	

Table 9. Z Data Tools functions protected by FILEM.OTHER.ALL

Function	Equivalent panel	Description	Profile
BSF	4.15.1	Backward space file (tape)	FILEM.OTHER.ALL
BSR	4.15.3	Backward space record (tape)	FILEM.OTHER.ALL
DVB	5.8	Data in virtual browse	FILEM.OTHER.ALL
FSF	4.15.2	Forward space file (tape)	FILEM.OTHER.ALL
FSR	4.15.4	Forward space record (tape)	FILEM.OTHER.ALL
OB	6.2	Object browse	FILEM.OTHER.ALL
ODL	6.1	Object directory list	FILEM.OTHER.ALL
OP	6.3	Object print	FILEM.OTHER.ALL
OQ, OS	6.6.2	Object to sequential data	FILEM.OTHER.ALL
OV	6.6.1	Object to VSAM	FILEM.OTHER.ALL

Table 9. Z Data Tools functions protected by FILEM.OTHER.ALL (continued)

Function	Equivalent panel	Description	Profile
REW	4.15.5	Rewind (tape)	FILEM.OTHER.ALL
RUN	4.15.6	Rewind and unload (tape)	FILEM.OTHER.ALL
VX		VSAM to REXX stem	FILEM.OTHER.ALL
XV		REXX stem to VSAM	FILEM.OTHER.ALL

Table 10. Z Data Tools function to profile name cross-reference

Function	Equivalent panel	Description	Profile
(none)		Using BLP for tape processing	FILEM.TAPE.BLP
(none)		Handling full disk packs	FILEM.DISK.FULLPACK
BT	4.7	Create Tape File	FILEM.TAPE.OUTPUT
DB	5.1	Disk Browse	FILEM.DISK.INPUT
DCN		Disk to console	FILEM.DISK.INPUT
DP	5.3	Disk Print	FILEM.DISK.INPUT
DRS	5.4	Disk Record Scan	FILEM.DISK.INPUT
DTE	5.2	Disk Track Edit	FILEM.DISK.UPDATE
EOF	5.5	Write EOF Record	FILEM.DISK.UPDATE
ERT	4.13	Erase Tape	FILEM.TAPE.UPDATE
EVC	4.2.8	Exported Stacked Volume Copy	FILEM.TAPE.DUPLICATE
EVL	4.14	Exported Stacked Volume List	FILEM.TAPE.INPUT
HT		z/FS to tape	FILEM.TAPE.OUTPUT
INT	4.12	Initialize Tape	FILEM.TAPE.UPDATE
LMU	2	Load module edit/update	FILEM.LOADMOD.UPDATE
OE	6.5	Object Erase	FILEM.OAM.UPDATE
OO	6.6.5	Object to Object	FILEM.OAM.OUTPUT
OU	6.4	Object Update	FILEM.OAM.UPDATE
QO, SO, TLO	6.6.4	Sequential Data to Object	FILEM.OAM.OUTPUT
QT, ST, STP	4.2.7	Sequential Data to Tape	FILEM.TAPE.OUTPUT

Table 10. Z Data Tools function to profile name cross-reference (continued)

Function	Equivalent panel	Description	Profile
TB	4.1	Tape Browse	FILEM.TAPE.INPUT
TC		Tape to card	FILEM.TAPE.INPUT
TCN		Tape to console	FILEM.TAPE.INPUT
TDL		Tape data set list	FILEM.TAPE.INPUT
TH		Tape to z/FS	FILEM.TAPE.INPUT
TLB	4.8	Tape Label Display	FILEM.TAPE.INPUT
TLT	4.2.3	Tape to Labeled Tape	FILEM.TAPE.DUPLICATE
TMP	4.6	Tape Map	FILEM.TAPE.INPUT
TP	4.5	Tape Print	FILEM.TAPE.INPUT
TQ, TS, TSQ	4.2.5	Tape to Sequential Data	FILEM.TAPE.INPUT
TRL	4.4	Tape Record Load	FILEM.TAPE.UPDATE
TRS	4.10	Tape Record Scan	FILEM.TAPE.INPUT
TT	4.2.1	Tape to Tape (copy)	FILEM.TAPE.DUPLICATE
TTC	4.9	Tape to Tape Compare	FILEM.TAPE.INPUT
TTR	4.2.2	Tape to Tape Reblocked	FILEM.TAPE.DUPLICATE
TU	4.3	Tape Update	FILEM.TAPE.UPDATE
TV, TVS	4.2.4	Tape to VSAM	FILEM.TAPE.INPUT
TX		Tape to REXX Variable	FILEM.TAPE.INPUT
VO	6.6.3	VSAM to Object	FILEM.OAM.OUTPUT
VRU		Update VSAM record	FILEM.VSAM.UPDATE
VT, VTP	4.2.6	VSAM to Tape	FILEM.TAPE.OUTPUT
VU	5.7	VSAM Update	FILEM.VSAM.UPDATE
WTM	4.11	Write Tape Mark	FILEM.TAPE.UPDATE
XT		REXX Variable to Tape	FILEM.TAPE.OUTPUT
Others		Protects all functions not otherwise listed in Table 8: Z Data Tools unprotected functions on page 77 or this table.	FILEM.OTHER.ALL

Setting up the security environment for IBM® MQ

Z Data Tools provides security features to secure access to MQ resources when using Z Data Tools functions.

These features work in conjunction with IBM® MQ security, not as a replacement. If access is not restricted by Z Data Tools, it may still be restricted by IBM® MQ security. Similarly, if access is not restricted by IBM® MQ security, it may still be restricted by Z Data Tools security. Z Data Tools security features for MQ are only applicable when accessing MQ resources using Z Data Tools.

Z Data Tools security for MQ is applicable to the user attempting to use a Z Data Tools function that accesses an MQ resource. By default, Z Data Tools does not secure access to MQ resources.

Activating security for a queue manager

To secure a queue manager, you must define a security resource indicating that security is required for a nominated queue manager on a given sysplex.

The security resource takes the form HFMMQ.SECURITY.*sysplex*.*qmgr* and must be defined to the FACILITY class. For example:

```
RDEFINE FACILITY HFMMQ.SECURITY.SYSPLEXD.CSQ1 UACC(READ)
```

Granting READ access to a user indicates that Z Data Tools security is applicable to that user for the nominated queue manager on the nominated sysplex. If no access is granted, security is not active. When security is active, users must be granted further access to resources to access the queue manager's attributes and queues. If security is not active for a queue manager, Z Data Tools permissions related to the queue manager do not apply.

Securing queue manager resources

When security is active for a queue manager, a user cannot access any of a queue manager's resources unless the user has at least READ access to a security resource of the form HFMMQ.*sysplex*.*qmgr* defined to the XFACILIT class.

For example:

```
RDEFINE XFACILIT HFMMQ.SYSPLEXD.CSQ1 UACC(NONE)
PERMIT HFMMQ.SYSPLEXD.CSQ1 CLASS(XFACILIT) ID(JOHND) ACCESS(READ)
```

Granting READ access to a user allows the user to list the queue manager's attributes, its queues, and its queue's attributes. To alter a queue manager's attributes, a user must have ALTER authority. For example, the following authority also allows a user to modify the queue manager's queue attributes, delete existing queues, and create new queues.

```
PERMIT HFMMQ.SYSPLEXD.CSQ1 CLASS(XFACILIT) ID(JOHND) ACCESS(ALTER)
```

Securing queue messages

When security is active for a queue manager, a user cannot access a queue's messages unless the user has at least READ access to a security resource of the form HFMMQ.*sysplex*.*qmgr*.*queue* defined to the XFACILIT class.

For example:

```
RDEFINE XFACILIT HFMMQ.SYSPLEXD.CSQ1.* UACC(NONE)
PERMIT HFMMQ.SYSPLEXD.CSQ1.* CLASS(XFACILIT) ID(JOHND) ACCESS(READ)
```

Granting READ access to a user allows the user to browse messages on the queue. To edit, insert, delete, or destructively get messages on a queue, a user must have at least UPDATE authority. For example, the following authority also allows a user to reset or clear a queue's messages:

```
PERMIT HFMMQ.SYSPLEXD.CSQ1.* CLASS(XFACILIT) ID(JOHND) ACCESS(UPDATE)
```

UPDATE authority also allows a user to reset or clear a queue's messages.

Securing Z Data Tools commands

When security is active for a queue manager, a user must have appropriate access to target MQ resources pertinent to the Z Data Tools command being executed.

There is a range of commands that can affect IBM® MQ resources. In each case the following security resource definitions and permissions are required:

- When a command reads a queue manager's attributes or its queue's attributes, the requesting user must have READ authority to resource HFMMQ.*sysplex*.*qmgr* in the XFACILIT class for the queue manager being read.
- When a command alters a queue manager's attributes or defines a queue, the requesting user must have ALTER authority to resource HFMMQ.*sysplex*.*qmgr* in the XFACILIT class for the queue manager being modified.
- When a command reads message data, the requesting user must have READ authority to resource HFMMQ.*sysplex*.*qmgr*.*queue* in the XFACILIT class for the queue being read.
- When a command updates message data, the requesting user must have UPDATE authority to resource HFMMQ.*sysplex*.*qmgr*.*queue* in the XFACILIT class for the queue being updated.

Securing message context

Z Data Tools uses MQ security controls to edit messages and message context.

When a Z Data Tools session is started, Z Data Tools checks if the user has CONTROL access to the *qmgr*.CONTEXT.*queue* resource in the MQADMIN class. For some External Security Managers, this check might require certain permissions. When a message is updated, the MQ message context is preserved if the user has CONTROL access to the *qmgr*.CONTEXT.*queue* resource in the MQADMIN class. If not, the message context is replaced with a default message context in accordance with the MQ normal operation.

MQ security examples

These examples illustrate techniques for controlling access to MQ resources in Z Data Tools.

Activate security for all queue managers on a sysplex

To activate security for all users and for all queue managers and their resources on a sysplex, define the following resource:

```
RDEFINE FACILITY HFMMQ.SECURITY.sysplex.* UACC(READ)
SETROPTS RACLIST(FACILITY) REFRESH
```

Where *sysplex* is the name of the sysplex hosting the queue managers. Without further permissions, this will disable all Z Data Tools MQ function for all queue managers on the sysplex.

Activate security for a specific queue manager on a sysplex

To activate security for all users of a specific queue manager and its resources, define the following resource:

```
RDEFINE FACILITY HFMMQ.SECURITY.sysplex.qmgr UACC(READ)
SETROPTS RACLIST(FACILITY) REFRESH
```

Where *sysplex* is the name of the sysplex hosting the queue manager, and *qmgr* is the name of the relevant queue manager on the sysplex. Without further permissions, this will disable all Z Data Tools MQ function for the nominated queue manager.

Permit all users read access to a queue manager's resources

When security is active, to permit all users read access to all resources of a specific queue manager, define the following resources:

```
RDEFINE XFACILIT HFMMQ.sysplex.qmgr UACC(READ)
RDEFINE XFACILIT HFMMQ.sysplex.qmgr.* UACC(READ)
SETROPTS RACLIST(XFACILIT) REFRESH
```

Where *sysplex* is the name of the sysplex hosting the queue manager, and *qmgr* is the name of the relevant queue manager on the sysplex. This will allow all users to list the queue manager’s attributes, its queues, and its queue’s attributes, as well as read messages from its queues.

Permit all but a single user read access to a queue manager’s resources

To permit all but a single user read access to all resources of a specific queue manager, define the following resources and permissions:

```
RDEFINE XFACILIT HFMMQ.sysplex.qmgr UACC(READ)
RDEFINE XFACILIT HFMMQ.sysplex.qmgr.* UACC(READ)
PERMIT HFMMQ.sysplex.qmgr CLASS(XFACILIT) id(userid) ACCESS(NONE)
PERMIT HFMMQ.sysplex.qmgr.* CLASS(XFACILIT) id(userid) ACCESS(NONE)
SETROPTS RACLIST(XFACILIT) REFRESH
```

Where *sysplex* is the name of the sysplex hosting the queue manager, *qmgr* is the name of the relevant queue manager on the sysplex, and *userid* represents the restricted user. This will allow all but the nominated user to list the queue manager's attributes, its queues, and its queue’s attributes, as well as read messages from its queues.

Permit a single user read access to a queue manager's resources

To permit only a single user read access to all resources of a specific queue manager, define the following resources and permissions:

```
RDEFINE XFACILIT HFMMQ.sysplex.qmgr UACC(NONE)
RDEFINE XFACILIT HFMMQ.sysplex.qmgr.* UACC(NONE)
PERMIT HFMMQ.sysplex.qmgr CLASS(XFACILIT) id(userid) ACCESS(READ)
PERMIT HFMMQ.sysplex.qmgr.* CLASS(XFACILIT) id(userid) ACCESS(READ)
SETROPTS RACLIST(XFACILIT) REFRESH
```

Where *sysplex* is the name of the sysplex hosting the queue manager, *qmgr* is the name of the relevant queue manager on the sysplex, and *userid* represents the permitted user. This will allow only the nominated user to list the queue manager's attributes, its queues, and its queue's attributes, as well as read messages from its queues.

Permit all users update access to all message data

When security is active, to permit all users update access to all queue's message data of a specific queue manager, define the following resources:

```
RDEFINE XFACILIT HFMMQ.sysplex.qmgr UACC(READ)
RDEFINE XFACILIT HFMMQ.sysplex.qmgr.* UACC(UPDATE)
SETROPTS RACLIST(XFACILIT) REFRESH
```

Where *sysplex* is the name of the sysplex hosting the queue manager, and *qmgr* is the name of the relevant queue manager on the sysplex. This will allow all users to list the queue manager's attributes, its queues, and its queue's attributes, as well as create, read, update, and delete messages from its queues.

Permit all but a single user update access to all message data

To permit all but a single user update access to all message data of a specific queue manager, define the following resources and permissions:

```
RDEFINE XFACILIT HFMMQ.sysplex.qmgr UACC(READ)
RDEFINE XFACILIT HFMMQ.sysplex.qmgr.* UACC(UPDATE)
PERMIT HFMMQ.sysplex.qmgr CLASS(XFACILIT) id(userid) ACCESS(NONE|READ)
PERMIT HFMMQ.sysplex.qmgr.* CLASS(XFACILIT) id(userid) ACCESS(NONE|READ)
SETROPTS RACLIST(XFACILIT) REFRESH
```

Where *sysplex* is the name of the sysplex hosting the queue manager, *qmgr* is the name of the relevant queue manager on the sysplex, and *userid* represents the restricted user. This will allow all but the nominated user to list the queue manager's attributes, its queues, and its queue's attributes, as well as create, read, update, and delete messages from its queues.

Permit a single user update access to all message data

To permit only a single user update access to all message data of a specific queue manager, define the following resources and permissions:

```
RDEFINE XFACILIT HFMMQ.sysplex.qmgr UACC(NONE)
RDEFINE XFACILIT HFMMQ.sysplex.qmgr.* UACC(NONE)
PERMIT HFMMQ.sysplex.qmgr CLASS(XFACILIT) id(userid) ACCESS(READ)
PERMIT HFMMQ.sysplex.qmgr.* CLASS(XFACILIT) id(userid) ACCESS(UPDATE)
SETROPTS RACLIST(XFACILIT) REFRESH
```

Where *sysplex* is the name of the sysplex hosting the queue manager, *qmgr* is the name of the relevant queue manager on the sysplex, and *userid* represents the permitted user. This will allow only the nominated user to list the queue manager's attributes, its queues, and its queue's attributes, as well as create, read, update, and delete messages from its queues.

Permit all users alter access to a queue manager's resources

When security is active, to permit all users alter access to all resources of a specific queue manager, define the following resources:

```
RDEFINE XFACILIT HFMMQ.sysplex.qmgr UACC(ALTER)
RDEFINE XFACILIT HFMMQ.sysplex.qmgr.* UACC(UPDATE)
SETROPTS RACLIST(XFACILIT) REFRESH
```

Where *sysplex* is the name of the sysplex hosting the queue manager, and *qmgr* is the name of the relevant queue manager on the sysplex. This will allow all users to list and update the queue manager's attributes, its queues, and its queue's attributes, as well as create, read, update, and delete messages from its queues.

Permit all but a single user alter access to a queue manager's resources

To permit all but a single user alter access to all resources of a specific queue manager, define the following resources and permissions:

```
RDEFINE XFACILIT HFMMQ.sysplex.qmgr UACC(ALTER)
RDEFINE XFACILIT HFMMQ.sysplex.qmgr.* UACC(UPDATE)
PERMIT HFMMQ.sysplex.qmgr CLASS(XFACILIT) id(userid) ACCESS(NONE|READ)
PERMIT HFMMQ.sysplex.qmgr.* CLASS(XFACILIT) id(userid) ACCESS(NONE|READ)
SETROPTS RACLIST(XFACILIT) REFRESH
```

Where *sysplex* is the name of the sysplex hosting the queue manager, *qmgr* is the name of the relevant queue manager on the sysplex, and *userid* represents the restricted user. This will allow all but the nominated user to list and update the queue manager's attributes, its queues, and its queue's attributes, as well as create, read, update, and delete messages from its queues.

Permit a single user alter access to a queue manager's resources

To permit only a single user alter access to all resources of a specific queue manager, define the following resources and permissions:

```
RDEFINE XFACILIT HFMMQ.sysplex.qmgr UACC(NONE)
RDEFINE XFACILIT HFMMQ.sysplex.qmgr.* UACC(NONE)
PERMIT HFMMQ.sysplex.qmgr CLASS(XFACILIT) id(userid) ACCESS(ALTER)
PERMIT HFMMQ.sysplex.qmgr.* CLASS(XFACILIT) id(userid) ACCESS(UPDATE)
SETROPTS RACLIST(XFACILIT) REFRESH
```

Where *sysplex* is the name of the sysplex hosting the queue manager, *qmgr* is the name of the relevant queue manager on the sysplex, and *userid* represents the permitted user. This will allow only the nominated user to list and update the queue manager's attributes, its queues, and its queue's attributes, as well as create, read, update, and delete messages from its queues.

Permit a single user access to a subset of queues

To permit a single user access to a subset of resources of a queue manager, define the following resources and permissions:

```
RDEFINE XFACILIT HFMMQ.sysplex.qmgr.DEV.* UACC(NONE)
PERMIT HFMMQ.sysplex.qmgr.DEV.* CLASS(XFACILIT) id(userid) ACCESS(READ|UPDATE)
SETROPTS RACLIST(XFACILIT) REFRESH
```

Where *sysplex* is the name of the sysplex hosting the queue manager, *qmgr* is the name of the relevant queue manager on the sysplex, and *userid* represents the permitted user. This will allow the nominated user relevant access to queues prefixed "DEV.".

Chapter 5. Customizing Z Data Tools to write audit records to SMF

You can use the audit logging facilities of Z Data Tools to create an audit trail of editing and other activities against data sets, Db2® objects, IMS™ databases and CICS® resources. For more information, refer to:

- Customizing the Z Data Tools audit facility for the Z Data Tools Base component on page 88
- Customizing the audit facility for ZDT/Db2 on page 189
- Customizing the Z Data Tools audit facility for IMS component on page 297
- Customizing the Z Data Tools audit facility for CICS component on page 352

Audit records can be written to an audit log data set, or to SMF, or both (dual logging). See the topics referred to above for information about restrictions that apply to dual logging.

If you intend to write audit log records to SMF for any component of Z Data Tools, complete the steps below.

Using System Management Facilities (SMF) for audit logging

If you intend to use SMF for audit logging, you must do the following:

- Select an SMF record number between 128 and 255 for the audit log records, and include it in your SMF parmlib member SMFPRMxx.
- Specify this SMF record number in one of these locations:
  - The HFM0POPI macro for the appropriate HFMxPOPT module. (See Note 1).
  - The HFMxPARM member in SYS1.PARMLIB, or other library in the logical PARMLIB concatenation. (See Note 2).
- Ensure that the load module HFMSMF is APF-authorized. You can make HFMSMF APF-authorized either by authorizing the load library, HFM.SHFMMOD1, or by copying HFMSMF to another authorized library. For more information about authorizing HFM.SHFMMOD1, see Running Z Data Tools with APF-authorization on page 53.
- Add the load module HFMSMF to the AUTHTSF list in member IKJTSOxx in SYS1.PARMLIB. If you do not do this, even if you have selected to record to SMF and you have specified an SMF record number, no recording is done.

Note:

1. Each Z Data Tools component has a customization module:

   HFM0POPT

   For Z Data Tools Base component

   HFM1POPT

   For ZDT/IMS

   HFM2POPT

   For ZDT/Db2



HFM3POPT

For ZDT/CICS

All the customization modules include an HFM0POPI macro specification, which is described in Z Data Tools options on page 383. The SMF record number is specified using the SMFNO parameter of the HFM0POPI macro. See SMFNO on page 416. You should specify the SMF record number in the HFMxPOPT member when you are using HFMxPOPT controlled auditing, or SAF-controlled auditing without the use of a member in SYS1.PARMLIB.

2. Auditing for each Z Data Tools component can be controlled using a member in SYS1.PARMLIB, or other library in the logical PARMLIB concatenation. The member names for each component are:

```
HFM0PARM  For Z Data Tools Base
                        component
HFM1PARM  For ZDT/IMS
HFM2PARM  For ZDT/Db2
HFM3PARM  For ZDT/CICS
```

   Specify the SMF record number in the HFMxPARM member when you are using SAF-controlled auditing and a member in SYS1.PARMLIB.

To activate any changes you have made to SYS1.PARMLIB members, either restart your system, or use the appropriate commands for your site to dynamically activate the changes.

For more information about SMF, see *z/OS MVS System Management Facilities (SMF)*.

To report on the audit trail information collected by SMF, you must extract this information from SMF to your own data set. The information in this data set can then be printed by the Z Data Tools Print Audit Trail utility. To do this select the *"Audit trail"* option from the Utilities menu.

A sample job, HFMSMFX, is provided in HFM.SHFMSAM1 to help you extract the SMF data to your own data set. See the comments in the job for information about changes you need to make to the job. The sample job can be used to extract audit log records for all Z Data Tools components (Base, ZDT/Db2, ZDT/IMS, and ZDT/CICS). The logon ID used to run the sample job must have read access to the SYS1.MANx data sets to run successfully.

Chapter 6. Customizing the Z Data Tools audit facility for the Z Data Tools Base component

Z Data Tools can write audit log records to either SMF or an audit log data set.

If auditing is not required:

- Set AUDITLOG=NO in the HFM0POPI macro specification of the HFM0POPT module. See AUDITLOG on page 388 for more information.
- Set SMFNO=0 in the HFM0POPI macro specification of the HFM0POPT module. See SMFNO on page 416 for more information.
- Skip the customization described in the rest of this chapter.

Z Data Tools provides two different methods (HFM0POPT-controlled and SAF-controlled auditing) for controlling whether audit records are written for Z Data Tools Base component. These are described in detail in Alternatives for controlling Z Data Tools Base auditing on page 28.

You should determine which of the two methods is appropriate for your site's requirements.

Use the checklist to determine the customization required for the Z Data Tools audit facility.

Table 11. Checklist for audit customization, Z Data Tools Base component. This table lists choices and decisions.

Audit customization choice	Decision (Yes|No|Not applicable)
1. Control auditing using the HFM0POPT options module	
2. Control auditing using SAF rules and a member in SYS1.PARMLIB	
3. Control auditing using SAF rules, without any changes to SYS1.PARMLIB	
4. Audit records are to be written to a data set	
5. Audit records are to be written to SMF	

For choices 1 to 3, you should answer YES for one choice only. Mark the other two choices as 'Not applicable'.

If you answer YES for choice 1, you can answer YES for one of choices 4 and 5. Mark the other choice as 'Not applicable'.

If you answer YES for choices 2 or 3, you can answer YES for one or both of choices 4 and 5. If you answer YES for both choices you are implementing dual-logging, which is only available with SAF-controlled auditing.

Table 12. Customization steps for audit customization choices. This table lists choices and related actions.

Customization choice	Sections to complete
1. Control auditing using the HFM0POPT options module	• HFM0POPT-controlled auditing on page 89
2. Control auditing using SAF rules and a member in SYS1.PARMLIB	• SAF-controlled auditing for Z Data Tools Base component on page 91 • Implementing SAF-rule controlled auditing on page 94
3. Control auditing using SAF rules, without any changes to SYS1.PARMLIB	• SAF-controlled auditing for Z Data Tools Base component on page 91 • Implementing SAF-rule controlled auditing on page 94
4. Audit records are to be written to a data set	• Audit data set configuration on page 89
5. Audit records are to be written to SMF	• Customizing Z Data Tools to write audit records to SMF on page 86

HFM0POPT-controlled auditing

To implement HFM0POPT-controlled auditing you need to set the AUDITLOG option in the HFM0POPI macro of the HFM0POPT module.

For information about changing the options in HFM0POPT see Changing the default options on page 55.

For information about the options see Z Data Tools options on page 383.

- If you want to produce an audit trail, specify AUDITLOG=YES in HFM0POPI macro.
- If you want to produce an audit trail and report on the changes made at the conclusion of an edit function then specify AUDITLOG=DEMAND. The job submitted will be determined by skeleton member HFM0FTAD found in HFM.SHFMSLIB. You should customize the job card and JCL to specify the reporting options you require. For changing the audit report options in the skeleton see "AUD (Print Audit Trail Report)" in the *Z Data Tools User's Guide and Reference*.

Audit data set configuration

The format of the audit log data set name is determined by the setting of the AUDITHLQ parameter in the HFM0POPI definition in HFM0POPT. See AUDITHLQ on page 386 for more information about the AUDITHLQ option.

The following data set name formats may be generated:

- `userid.HFMLOG.Dyymmdd.Thhmmss` (when AUDITHLQ= (blank))
- *`audithlq`*`.userid.HFMLOG.Dyymmdd.Thhmmss` (when AUDITHLQ=*audithlq*)
- `qual1.<qual2.><qual3.>Dyymmdd.Thhmmss` (when AUDITHLQ=*qual1.<qual2.><qual3>*)

where:

audithlq

Any 1-8 character constant that is valid in the context of a data set name.

userid

The user ID creating the data set.

Dyymmdd

The date of the activity.

Thhmmss

The time of the activity.

When AUDITHLQ contains one or more periods, the AUDITHLQ value is treated as a data set prefix, with one, two or three levels. Each level of the prefix can be:

XXX

Any 1-8 character constant that is valid in the context of a data set name.

&&PREFIX

Indicates that the user's TSO prefix should be used. This will be null if TSO NOPREFIX is in effect and, after substitution, the appropriate level of the audit log data set name prefix will also be null.

&&USER

Indicates that the user's logonid (ISPF system variable ZUSER, stored in the shared pool) should be used.

&&UID

Indicates that the user's TSO prefix should be used, when the value is non-blank. When TSO NOPREFIX is in effect, the user's TSO logonid (ISPF system variable ZUSER, stored in the shared pool) should be used.

&&FUNCOD

Indicates that the Z Data Tools internal function code should be used. Specifying this parameter allows the Z Data Tools function that generated the audit log data set to be included in the audit log data set name.

Set the AUDITHLQ parameter in the HFM0POPI macro for the HFM0POPT to the required value, based on the above information and your site's requirements.

You can print the information in a Z Data Tools Base component audit data set using the Z Data Tools Print Audit Trail utility. To do this select option 3.12 from the Z Data Tools Primary Option Menu.

SAF-controlled auditing for Z Data Tools Base component

There are two methods for implementing SAF-controlled auditing for Z Data Tools Base component. These are:

1. Control auditing for Z Data Tools Base component using an enabling SAF Facility class rule and a member in SYS1.PARMLIB.

   To use this method complete the customization described in SAF-controlled auditing using SYS1.PARMLIB on page 91.

2. Control auditing for Z Data Tools Base component using an enabling SAF Facility class rule, without any changes to SYS1.PARMLIB.

   To use this method complete the customization described in SAF-controlled auditing without SYS1.PARMLIB on page 93.

SAF-controlled auditing using SYS1.PARMLIB

You need to define an enabling SAF facility profile as described below:

Define SAF facility profile

```
FILEM.PARMLIB.BASE
```

and ensure all Z Data Tools Base function users to be audited have at least read access to that facility. See the example below:

Example

User PROD1 to have SAF-rule controlled auditing using SYS1.PARMLIB.

Write this RACF® rule:

```
RDEF FACILITY FILEM.PARMLIB.BASE AUDIT(NONE) UACC(NONE) OWNER(ownerid)
PE FILEM.PARMLIB.BASE ACC(READ) ID(PROD1) CLASS(FACILITY)
```

Add member HFM0PARM to SYS1.PARMLIB (or any other library in the logical parmlib concatenation). See Defining the HFM0PARM member on page 92.

Once the above SAF rule is defined and activated, auditing for Z Data Tools Base component users is controlled by the FMAUDIT parameter in the HFM0PARM member. See Z Data Tools options specified in HFM0PARM on page 515 for more information. If audit log records are to be written to SMF, the SMF record number is specified as an FMAUDIT parameter option. See FMAUDIT on page 515, and SMF_NO on page 516.

Note: Z Data Tools does not start if a user has read access to the above facility and the HFM0PARM member does not exist in the logical parmlib concatenation.

If SAF processing is not active, or the rule is not defined, or the rule is defined and the user has no access, then no parmlib processing is performed.

Defining the HFM0PARM member

If auditing is to be controlled from parmlib, then member HFM0PARM must be defined in SYS1.PARMLIB (or any other library in the logical parmlib concatenation).

Default parmlib member HFM0PARM is provided in the SHFMSAM1 library. Copy this member to the appropriate system parmlib library. See below for details of methods that can be used to make this change.

Note: The sample HFM0PARM member supplied in SHFMSAM1 also includes a FMSECRTY statement. This option is not used at present, and can be either omitted, or commented out. It has no effect.

There are two methods that can be used to include the HFM0PARM member in a library in the logical parmlib concatenation. The choice of method depends on whether the installation's security software is configured to allow Z Data Tools users READ access to the data set SYS1.PARMLIB. Method 1 can only be used when Z Data Tools users have read access to SYS1.PARMLIB. Method 2 can be used regardless of whether Z Data Tools users have READ access to SYS1.PARMLIB or not, and must be used when Z Data Tools users do not have READ access to SYS1.PARMLIB.

Method 1

Place the HFM0PARM member in any library in the current logical parmlib concatenation. No IPL or other action is required to activate the new member (unless a new library was added to the logical parmlib concatenation).

Note:

1. Method 1 cannot be used in any situation where Z Data Tools users do not have READ access to SYS1.PARMLIB. For example, when Z Data Tools users have READ access to another library in the logical parmlib concatenation, and the HFM0PARM member is placed in the latter

library. This will not work. The key issue is whether the Z Data Tools user has READ access to SYS1.PARMLIB.

2. Using this method results in message IEE252I being written to the system log whenever a Z Data Tools user accesses SYS1.PARMLIB. These messages cannot be suppressed. To avoid these messages use Method 2.

Method 2

This method must be used when Z Data Tools users do not have READ access to SYS1.PARMLIB, or when suppression of the IEE252I messages is required.

1. Create a new library with dataset attributes similar to SYS1.PARMLIB.

   The library name for this data set must include the string "HFMPARM" in one of the qualifiers. You can choose any data set name that meets this requirement. Examples of suitable data set names are:

   SYS1.PARMLIB.HFMPARM
   SYS8.HFMPARM.PARMLIB
   HFMPARM.SYS8.PARMLIB
   SYS2.HFMPARMS.LIB
   SYS8.XHFMPARM.PARMLIB
2. Add member HFM0PARM to the new library, specifying the appropriate FMAUDIT parameter.
3. Add the new library to the logical parmlib concatenation. This can be done dynamically, or by means of a system IPL.

Note: When Method 2 is used, the HFM0PARM member must be located in the library created in step 1 on page 93. If the HFM0PARM member specifies any include statements (see Facilities for customizing the HFM0PARM definitions on page 516), all of the included members must also reside in the same library.

You use the HFM0PARM member to define:

- Whether Z Data Tools uses SAF to control Z Data Tools audit logging.
- The SAF resource name prefix to be used by Z Data Tools when determining access to various resources.
- Whether Z Data Tools loads the HFM0POPT module from a specific library.

For more information, see Z Data Tools options specified in HFM0PARM on page 515.

SAF-controlled auditing without SYS1.PARMLIB

You need to define an enabling SAF facility profile as described below:

Define SAF facility profile

```
FILEM.SAFAUDIT.BASE
```

and ensure that all Z Data Tools Base function users to be audited have at least read access to that facility. See the example below.

Example

User PROD2 to have SAF-rule controlled auditing without using SYS1.PARMLIB.

Write this RACF® rule:

```
RDEF FACILITY FILEM.SAFAUDIT.BASE AUDIT(NONE) UACC(NONE) OWNER(ownerid)
PE FILEM.SAFAUDIT.BASE ACC(READ) ID(PROD2) CLASS(FACILITY)
```

If you use this method and intend to write audit records to SMF, the required SMF number is specified in the HFM0POPT module. See Customizing Z Data Tools to write audit records to SMF on page 86 for more information.

When a security product other than RACF® is in use

Z Data Tools makes RACROUTE calls, with STATUS=ACCESS, to SAF FACILITY profiles:

FILEM.PARMLIB.**
FILEM.SAFAUDIT.** (only if the first call fails)

STATUS=ACCESS is specified because the RACROUTE call should not audit the call. The documentation in the *z/OS® Security Server RACROUTE Macro Reference*, in the description of RACROUTE REQUEST=AUTH,STATUS=ACCESS, states:

ACCESS

The request is simply to return the user's highest current access to the resource specified. Upon successful completion, the user's access is returned in the RACF® reason code. **No auditing is done for this request.**

When RACF® is used, the STATUS=ACCESS request works as documented, and no security-related logging or abends are generated, even if you do not have access to the profile.

However, when non-RACF security products such as ACF2 are used, an S047 abend may be issued in response to the above RACROUTE request. These users should consult the product documentation and make changes accordingly.

Implementing SAF-rule controlled auditing

Use the following checklist to implement SAF-rule controlled auditing:

1. Determine the SAF FACILITY and XFACILIT rules that will be required. See Understanding how Z Data Tools uses SAF rules to control auditing on page 95 for more information.
2. Write the relevant SAF rules. See the examples in SAF rule examples on page 98.
3. Activate SAF auditing for a specific logon using the chosen method of activating SAF-controlled auditing. See SAF-controlled auditing for Z Data Tools Base component on page 91.
4. Using the selected logon, test the configuration to ensure that auditing is occurring as required.
5. When testing is complete, activate SAF-controlled auditing for all Z Data Tools users.

Understanding how Z Data Tools uses SAF rules to control auditing

SAF (System Authorization Facility) allows applications, such as Z Data Tools, to define "resources" that might need to be protected. The "resource" to be protected need not be something specific, such as a data set; it can be essentially any type of resource or facility that the application considers to be important. For Z Data Tools and auditing, the "resource" is the ability to write audit log records. The resource names reflect either the type of auditing that is to occur (eg to SMF), or the Z Data Tools function and data set.

Z Data Tools uses two types of SAF resource names to control auditing. The SAF resource rules used by Z Data Tools to control auditing are shown in Table 15: Z Data Tools auditing FACILITY class resource names on page 101 and Table 16: Z Data Tools auditing XFACILIT class resource names on page 101).

Understanding SAF rule access levels

SAF provides for five levels of access to any FACILITY or XFACILIT resource. The levels of access form a hierarchy, so that a user with the highest level of access to a resource also has access to all the lower levels. The levels of access are specified in RACF® rules using the following mnemonics:

NONE

No access

READ

Level 1 access

UPDATE

Level 2 access

CONTROL

Level 3 access

ALTER

Level 4 access.

It is important to understand that the mnemonics used (READ, UPDATE and so on) can and do mean different things, depending on the context in which the SAF resource name is used. This can be confusing since READ and UPDATE have obvious meanings when it comes to, for example, accessing a data set. For SAF rules used to control Z Data Tools audit, it may aid understanding to think of the mnemonics as indicating level 1 access and level 2 access.

For the SAF resource rules used by Z Data Tools, the meanings of the various levels of access are:

NONE

The user does not have access to the resource; this typically means the user cannot write audit log records.

READ

The user has level 1 access to the resource; this typically means that the user can write audit log records.

UPDATE

The user has level 2 access to the resource. This level of access only has meaning for FACILITY rule 2 (see Table 15: Z Data Tools auditing FACILITY class resource names on page 101). A user with level 2 access can write audit log records to the user's audit log data set, and the audit log data set will be printed at the end of the user's session (online execution only). This is equivalent to the DEMAND audit option in the non-SAF case.

CONTROL

The user has level 3 access to the resource. This level of access is not used by Z Data Tools.

ALTER

The user has level 4 access to the resource. This level of access is not used by Z Data Tools.

How Z Data Tools determines whether audit log records should be written

The determination of whether audit records are to be written for a particular Z Data Tools function and a given TSO logonid follows this three step process:

1. Step 1.
   - If auditing is being controlled by means of parmlib, the FMAUDIT specification of the HFM0PARM member is used as follows.

     The FMAUDIT specification setting in the HFM0PARM member (in SYS1.PARMLIB or any other library in the logical parmlib concatenation) is the "main" switch for SAF-rule controlled auditing. Note that there are facilities available to specify different settings in the HFM0PARM member for different TSO logonids, see Z Data Tools options specified in HFM0PARM on page 515 for more information. For any given TSO logonid, there are two possibilities:

     SAF_CTRL=NO

     SAF-rule controlled auditing is not in effect. Auditing is determined by the settings in the HFM0POPT module, see Customizing the Z Data Tools audit facility for the Z Data Tools Base component on page 88.

     SAF_CTRL=YES

     SAF-rule controlled auditing is in effect. Processing continues to Step 2 on page 96.
   - If auditing is being controlled using the method which does not access the parmlib concatenation, the TSO logonid has READ access to the SAF FACILITY rule FILEM.SAFAUDIT.BASE for processing to continue to Step 2 on page 96.
2. Step 2.

   Does the user have access to write audit records?

   This is determined by the user's access to rules 1 and 2 in Table 15: Z Data Tools auditing FACILITY class resource names on page 101; the various outcomes are summarized in Table 13: Determination of a user's ability to write audit log records on page 97.

Table 13. Determination of a user's ability to write audit log records

TODSN access[1]	TOSMF access[2]	OPTION access[3]	Can write audit records?	Demand logging?	"Create audit trail" option[4]
NONE	NONE	ANY	No	No	Not visible
READ	NONE	NONE	Yes, data set only	No	Not visible
READ	NONE	READ	Yes, data set only	No	Visible
UPDATE	NONE	NONE	Yes, data set only	Yes	Not visible
UPDATE	NONE	READ	Yes, data set only	Yes	Visible
NONE	READ	NONE	Yes, SMF only	No	Not visible
NONE	READ	READ	Yes, SMF only	No	Visible
READ	READ	NONE	Yes, to data set and SMF	No	Not visible
READ	READ	READ	Yes, to data set and SMF	No	Visible
UPDATE	READ	NONE	Yes, to data set and SMF	Yes	Not visible
UPDATE	READ	READ	Yes, to data set and SMF	Yes	Visible

If the user does not have the ability to write audit log records, then no check of SAF resource names in Step 3 occurs.

A user's access to write audit log records at Step 2 only indicates that auditing *might* occur. The final decision depends on the user's level of access to the XFACILIT resource name (or names) that apply to the particular Z Data Tools function.

3. Step 3.

   Does the user have access to write audit records for the current function and data set?

   The XFACILIT resource names that are used by Z Data Tools to determine whether audit records should be written depend on the Z Data Tools function being executed and the data set being accessed.

1. Refers to the level of access the user has to SAF FACILITY rule 1 in Table 15: Z Data Tools auditing FACILITY class resource names on page 101.
2. Refers to the level of access the user has to SAF FACILITY rule 2 in Table 15: Z Data Tools auditing FACILITY class resource names on page 101.
3. Refers to the level of access the user has to SAF FACILITY rule 3 in Table 15: Z Data Tools auditing FACILITY class resource names on page 101.
4. The visibility of the "Create audit trail" option does not influence whether a user can write audit log records, although the user must have access to write audit log records (to either a data set or SMF), for the option to be visible.

Table 14: Z Data Tools function codes that can be audited using SAF on page 98 shows the function codes which are supported.

Table 14. Z Data Tools function codes that can be audited using SAF

Function code	Online option	Description
DSB	BROWSE prefix command	Browse
DSV	1	View
DSE	2	Edit
DSC	3.3	Copy
DSG	3.1	Create
DSP	3.2	Print
DSM	3.11	Compare
FCH	3.6	Find/Change
DSCMP		Batch compare
DSEB		Batch data set edit
DSU		Batch data set update

These restrictions apply:

- When copying data sets and SAF XFACILIT rules have been supplied for both data sets and these rules are different, the most inclusive rule is applied. For example, if the SAF XFACILIT rule for the old data set specifies UPDATE logging, and the rule for the new data set specifies FUNCTION logging, UPDATE logging is applied.
- When comparing data sets and SAF XFACILIT rules have been supplied for both data sets and these rules are different, the most inclusive rule is applied. For example, if the SAF XFACILIT rule for the old data set specifies ALL logging, and the rule for the new data set specifies FUNCTION logging, ALL logging is applied.
- SAF XFACILIT rules do not check for data sets which are written to using the WRITE function in a REXX procedure. The SAF XFACILIT rule supplied for the function and data sets is applied to any records written using the WRITE function.
- If a data set can be copied using DFSORT, but SAF logging is required, DFSORT is not used. If PDS members can be copied using IEBCOPY, but SAF logging is required for one or more members, IEBCOPY is not used.
- If the "Use Z Data Tools editor" option has not been selected, and a member is selected from a selection list to be browsed, edited, or viewed, and there is a SAF logging rule which affects this function and resource, the ISPF editor is not used. Instead, the Z Data Tools function is invoked to ensure the requested logging is performed.

SAF rule examples

This section shows SAF rule examples under different conditions.

Controlling where Z Data Tools writes audit log records

You can use SAF to control whether Z Data Tools writes audit log records to SMF, the user's audit log data set, or to both.

Table 15: Z Data Tools auditing FACILITY class resource names on page 101 shows the SAF FACILITY class resource names used to control Z Data Tools to logging and the user's audit log data set.

Example 1

- Disable audit logging to a user data set for all Z Data Tools users.
- Enable Z Data Tools audit logging to SMF for the PROD logonid.

You could write the following RACF® rules:

```
RDEL FACILITY FILEM.AUDIT.TOSMF[5]
RDEL FACILITY FILEM.AUDIT.TODSN[5]
RDEF FACILITY FILEM.AUDIT.TOSMF UACC(NONE) OWNER(XXXXXXX)[6]
RDEF FACILITY FILEM.AUDIT.TODSN UACC(NONE) OWNER(XXXXXXX)[7]
PE FILEM.AUDIT.TOSMF ACC(READ) ID(PROD) CLASS(FACILITY)[8]
```

Example 2

- Enable audit logging to a user data set for all Z Data Tools users.
- Enable demand logging for the following users, PROD1, PROD2, PROD3.

You could write the following RACF® rules:

```
RDEL FACILITY FILEM.AUDIT.TOSMF[5]
RDEL FACILITY FILEM.AUDIT.TODSN[5]
RDEF FACILITY FILEM.AUDIT.TOSMF UACC(NONE) OWNER(XXXXXXX)[6]
RDEF FACILITY FILEM.AUDIT.TODSN UACC(READ) OWNER(XXXXXXX)[9]
PE FILEM.AUDIT.TODSN ACC(UPDATE) ID(PROD1) CLASS(FACILITY)[10]
PE FILEM.AUDIT.TODSN ACC(UPDATE) ID(PROD2) CLASS(FACILITY)[10]
PE FILEM.AUDIT.TODSN ACC(UPDATE) ID(PROD3) CLASS(FACILITY)[10]
```

Example 3

- Disable audit logging completely for all Z Data Tools users.
- Enable dual logging for all Z Data Tools users.

You could write the following RACF® rules:

5. Delete any existing facility rule.
6. Define the facility rule for audit logging to SMF (TOSMF suffix). UACC(NONE) is used so that any user, for which there is no specific rule, has no access.
7. Define the facility rule for audit logging to the user's audit log data set (TODSN suffix). UACC(NONE) is used so that any user, for which there is no specific rule, has no access.
8. Allow logonid PROD to write audit log records (ACC(READ)) to SMF.
9. Define the facility rule for audit logging to the user's audit log data set (TODSN suffix). UACC(READ) is used so that any user, for which there is no specific rule, has read access, and can therefore write audit log records.
10. Allow logonids PROD1, PROD2, PROD3 to write audit log records with automatic printing of the audit report ("Demand logging") (ACC(UPDATE)), to SMF.

```
RDEL FACILITY FILEM.AUDIT.TOSMF[5]
RDEL FACILITY FILEM.AUDIT.TODSN[5]
RDEF FACILITY FILEM.AUDIT.TOSMF UACC(READ) OWNER(XXXXXXX)[11]
RDEF FACILITY FILEM.AUDIT.TODSN UACC(READ) OWNER(XXXXXXX)[12]
```

Controlling auditing of Z Data Tools functions

You can use SAF to control whether Z Data Tools writes audit log records for functions which access resources. Table 14: Z Data Tools function codes that can be audited using SAF on page 98 shows Z Data Tools function codes which may be logged.

Example 1

- Enable audit logging of all modifications to data set HFM.TEST.DATA using the Z Data Tools Edit function for all users except TSO logonid MAINT1.

You could write the following RACF® rules:

```
RDEL XFACILIT FILEM.AUDIT.DSE.UPDATE.HFM.TEST.DATA[13]
RDEF XFACILIT FILEM.AUDIT.DSE.UPDATE.HFM.TEST.DATA OWNER(XXXXXXXX) UACC(READ)[14]
PE FILEM.AUDIT.DSE.UPDATE.HFM.TEST.DATA CLASS(XFACILIT) ID(MAINT1) ACC(NONE)[15]
```

Example 2

- Enable audit logging of all records which are read or modified for data set HFM.TEST.DATA using the Z Data Tools Edit function for user SERVIC1.

You could write the following RACF® rules:

```
RDEL XFACILIT FILEM.AUDIT.DSE.ALL.HFM.TEST.DATA[13]
RDEF XFACILIT FILEM.AUDIT.DSE.ALL.HFM.TEST.DATA OWNER(XXXXXXXX) UACC(NONE)[16]
PE FILEM.AUDIT.DSE.ALL.HFM.TEST.DATA CLASS(XFACILIT) ID(SERVIC1) ACC(READ)[17]
```

Example 3

- Enable audit logging of functional information for member MEM1 in library HFM.TEST.DATA.PDS using the Z Data Tools Print utility for all users.

11. Define the facility rule for audit logging to SMF (TOSMF suffix). UACC(READ) is used so that any user, for which there is no specific rule, has access (and can therefore write audit records to SMF).
12. Define the facility rule for audit logging to the user's audit log data set (TODSN suffix). UACC(READ) is used so that any user, for which there is no specific rule, has access (and can therefore write audit records to the user's audit log data set).
13. Delete any existing XFACILIT rule.
14. Define the XFACILIT rule to log all modifications to data set HFM.TEST.DATA using the Z Data Tools Edit function (DSE). UACC(READ) allows all TSO user IDs to write audit log records (in the absence of any over-riding more specific rule).
15. A specific rule for logonid MAINT1 to prevent audit log records being written.
16. Define the XFACILIT rule to log all records which are read or modified for data set HFM.TEST.DATA using the Z Data Tools Edit function (DSE). UACC(NONE) specifies that no TSO user IDs write audit log records (in the absence of any over-riding more specific rule).
17. A specific rule for logonid SERVICE1 to write audit log records.

You could write the following RACF® rules:

```
RDEL XFACILIT FILEM.AUDIT.DSP.FUNCTION.HFM.TEST.DATA.PDS.MEM1[13]
RDEF XFACILIT FILEM.AUDIT.DSP.FUNCTION.HFM.TEST.DATA.PDS.MEM1
     OWNER(XXXXXXXX) UACC(READ)[18]
```

Example 4

- Enable audit logging of functional information for all access to WebSphere MQ Queue HFM.TEST.QUEUE which is managed by WebSphere MQ Queue Manager HFM1 for all users.

You could write the following RACF® rules:

```
RDEL XFACILIT FILEM.AUDIT.*.FUNCTION.HFM1:HFM.TEST.QUEUE[13]
RDEF XFACILIT FILEM.AUDIT.*.FUNCTION.HFM1:HFM.TEST.QUEUE
     OWNER(XXXXXXXX) UACC(READ)[19]
```

Z Data Tools auditing FACILITY and XFACILIT class resource names

These two tables list FACILITY and XFACILIT class resource names and details.

Table 15. Z Data Tools auditing FACILITY class resource names

Rule Number	Resource Name	Purpose
1	FILEM.AUDIT.TODSN	Allows a user to write audit log records to the user's audit log data set.
2	FILEM.AUDIT.TOSMF	Allows a user to write audit log records to SMF.
3	FILEM.AUDIT.OPTION	Allows the user access to the "Create audit trail" option on the Z Data Tools Edit panels.

Table 16. Z Data Tools auditing XFACILIT class resource names

Resource Name[20]	Purpose
FILEM.AUDIT.functioncode.ALL[21].resource[22]	Allows users to write audit log records for all records which are read and modified for the specified data

18. Define the XFACILIT rule to log function information when member MEM1 in library HFM.TEST.DATA.PDS is printed using the Z Data Tools Print utility (DSP). UACC(READ) allows all TSO user IDs to write audit log records (in the absence of any over-riding more specific rule).
19. Define the XFACILIT rule to log function information when WebSphere MQ Queue HFM1:HFM.TEST.QUEUE is accessed using any Z Data Tools function (*). UACC(READ) allows all TSO user IDs to write audit log records (in the absence of any over-riding more specific rule).
20. You cannot define a SAF XFACILIT rule containing any of these:
    - Lowercase characters
    - Embedded spaces

Table 16. Z Data Tools auditing XFACILIT class resource names

(continued)

Resource Name[20]	Purpose
	set (resource) using the Z Data Tools function (function code).

20. You cannot define a SAF XFACILIT rule containing any of these:

    - Lowercase characters
    - Embedded spaces
    - Unprintable characters
    - Most special characters
    - Unprintable characters
    - Most special characters
21. Use this option with caution. The size of the data set being accessed and the editing technique used will influence the number of read records logged, and this may affect the performance of Z Data Tools.
22. When specifying the resource:

    - If a member name is to be included, it must be period-qualified and not enclosed in brackets.
    - A WebSphere Queue name must be prefixed with the WebSphere MQ Queue Manager name followed by a colon.
    -

    For CICS® resources other than files, the resource name is in the format:

    ```
    queuetype:cicsapplid:resourcename
    ```

    Where:

    queuetype

    Queue type. Can be one of these values:

    TS

    For temporary storage queues.

    TD

    For internal transient data queues.

    External transient data queues are controlled using the associated data set name.

    cicsapplid

    The CICS® applid where the resource is defined.

    resourcename

    The name of the temporary storage queue or internal transient data queue.

Table 16. Z Data Tools auditing XFACILIT class resource names

(continued)

Resource Name[20]	Purpose
FILEM.AUDIT.functioncode.UPDATE.resource	Allows users to write audit log records for all modifications to the specified data set (resource) using the Z Data Tools function (function code).
FILEM.AUDIT.functioncode.FUNCTION.resource	Allows users to write audit log records containing information for the specified data set (resource) using the Z Data Tools function (function code).

20. You cannot define a SAF XFACILIT rule containing any of these:

    - Lowercase characters
    - Embedded spaces
    - Unprintable characters
    - Most special characters
    - For CICS® files and external transient data queues, specify only the data set name.
    - If accessing CICS® resources from Z Data Tools Base component running under ZDT/CICS, omit the *cicsapplid* parameter from the resource name:

      ```
      queuetype:resourcename
      ```

Chapter 7. Customizing Z Data Tools for national languages

You can customize Z Data Tools for national languages other than English.

If you are using Japanese and you have installed the Z Data Tools Japanese component, you might not need to perform any other customization for the Japanese national language. If you are using a language other than English or Japanese, you will need to perform all of the steps listed in Table 17: Summary of steps for customizing Z Data Tools for a national language on page 104.

Table 17. Summary of steps for customizing Z Data Tools for a national language

Step	Description
__ 1	Set the LANGUAGE option for batch processing, if required. See Setting the default national language on page 55.
__ 2	Change the TERMTYPE option in HFM0POPT, if you are using a DBCS language. See TERMTYPE on page 418.
__ 3	Confirm that you have the correct terminal type set in ISPF (ISPF option 0).
__ 4	Create a print and display translation table for your language. See Changing the print and display translation tables for languages other than English on page 104.
__ 5	Translate the Z Data Tools message text to your language. See Translating the message text on page 106.
__ 6	Provide a version of HFM0MENU for your language. See Providing a multicultural version of HFM0MENU on page 106.
__ 7	Translate the Z Data Tools ISPF messages to your language. See Translating the ISPF messages text on page 107.
__ 8	Translate the Z Data Tools panels to your language. See Translating the panel text on page 107.

Changing the print and display translation tables for languages other than English

If you plan to use Z Data Tools with a national language other than English, you might need to provide print and display tables specific to your language.

You use the usermod HFMUMODX, and the member HFMTRTBS to do this. HFMUMODX and HFMTRTBS are distributed in HFM.SHFMSAM1.

A translation table is also provided for German. This is HFMTRDEU, which is distributed in HFM.SHFMMOD1. You can modify this table, if necessary, using the usermod HFMUMODG, which is distributed in HFM.SHFMSAM1.

A sample translation table is provided for Thai. This is HFMTRTHT, which is distributed in HFM.SHFMSAM1. To build the Thai translation table, copy this sample member to HFMTRTBS and use the usermod HFMUMODX.

This step is essential if you are using a DBCS language other than Japanese.

You do this as follows:

1. Check that the terminal on which you want to display Z Data Tools panels supports the display of special characters, or that the universal character buffer (UCB) of your printer has the characters you want to use.
2. Change the Z Data Tools options to specify PRTTRANS=ON. See Changing the default options on page 55 for information on how to do this.
3. If you are using any DBCS language, change the Z Data Tools options to specify TERMTYPE=3270KN.
4. Create a multicultural support version of the translation table as follows:
    a. Copy the HFMTRTBS member from HFM.SHFMSAM1 into your own source library, with the name HFMTR*yyy*, where *yyy* is one of the following language codes:

        FRA

            French

        DEU

            German

        ITA

            Italian

        JPN

            Japanese

        PTG

            Portuguese

        ESP

            Spanish

        DAN

            Danish

        ENP

            Upper case English

        KOR

            Korean

        DES

            Swiss German

        CHT

            Traditional Chinese

CHS

Simplified Chinese

XXX

Other

b. Change the translation table definition statements in your source member, HFMTR*yyy*, in your source library, as required.

Z Data Tools provides support under ISPF for command verbs and command keywords entered in lower or uppercase as long as they are valid SBCS-only strings, and the a - z codepoints in the host codepage used correspond to the standard EBCDIC a-z codepoints. If this is not the case, and you want to be able to enter lower or mixed case command verbs and command keywords, then you will need to modify the TRUPC translation table in HFMTRTBS, to specify the uppercase A - Z EBCDIC codepoints at locations corresponding to the a - z codepoints in the codepage used.

c. Modify the HFMUMODX member in HFM.SHFMSAM1 to meet your site's requirements. Refer to the usermod for information about changes you might need to make.
d. Install SMP/E usermod HFMUMODX.

Translating the message text

All Z Data Tools Base function messages are stored in the HFM0MENU source member.

This CSECT is part of the root module so that an English version of the messages is always available. In addition, all messages used by Z Data Tools under ISPF are provided in the library, HFM.SHFMMENU. Using both HFM0MENU and members in HFM.SHFMMENU, you can provide your own set of translated messages in another language. To use the messages you have translated, see Using the translated messages and panels on page 108.

To provide translated versions of the messages, you must provide a version of HFM0MENU for your language, as described in Providing a multicultural version of HFM0MENU on page 106, **and** provide translated versions of the appropriate members in HFM.SHFMMENU, as described in Translating the ISPF messages text on page 107.

You should not need to provide a Japanese version of the messages if you have installed the Z Data Tools Japanese component.

Providing a multicultural version of HFM0MENU

HFM0MENU contains the assembler source for the messages. To provide translated versions of the messages:

1. Copy the member HFM0MENU from HFM.SHFMSAM1 to your own source library with the name HFM0M*yyy*, where *yyy* is the same language code that you specified to change your print and display translation tables. (See step 4.a on page 105 above.)
2. Change the message text in HFM0M*yyy* in your library.

3. Modify the HFMUMODM member in HFM.SHFMSAM1 to meet your site's requirements, using the same language code as above. Refer to the usermod for information about other changes you might need to make.
4. Install SMP/E usermod HFMUMODM.

Translating the ISPF messages text

All Z Data Tools ISPF messages are provided in English.

They are provided in Japanese if you have installed the Japanese component of Z Data Tools. You can translate some or all of these messages into other languages.

All Z Data Tools ISPF messages are stored in HFM.SHFMMENU. You translate a message as follows:

1. Find the members in HFM.SHFMMENU that contain the messages you want to translate. The message members specific to the Z Data Tools Base function are all named HFMB*zzzz* or HFMM*nn*.
2. Create a library with the same characteristics as HFM.SHFMMENU, with the name HFM.SHFMM*yyy*, where *yyy* is the same language code that you specified to change your print and display translation tables. (See step 4.a on page 105.) Copy the required message members from HFM.SHFMMENU to this library.
3. Change the required message texts in these members in your library.

To use the messages you have translated, see Using the translated messages and panels on page 108.

Note: Be sure to include **ALL** the messages in the message members you copy to your own library. When Z Data Tools needs to display an ISPF message, it uses ISPF services to do this. Therefore the search for a message is made according to ISPF rules. Thus, if ISPF finds the message member it requires in your library, but the message number required is not in that member, ISPF will not look in any other library for the message, but will give an error. However, if you omit a complete message member from your library, then ISPF will use the English message member from the next library in the ISPMLIB concatenation. For more information about defining and using ISPF messages, see z/OS ISPF Dialog Developer's Guide.

Translating the panel text

All Z Data Tools panels are provided in English.

They are provided in Japanese if you install the Z Data Tools Japanese component. .

You can translate some or all of these panels into another language. (If no translated version of a particular panel is available, Z Data Tools uses the English version.)

All Z Data Tools panels are stored in HFM.SHFMPENU. You translate a panel as follows:

1. Find the members in HFM.SHFMPENU that you want to translate. The panel members specific to the Z Data Tools Base function are all named HFM*xzzzz* or HFM0*zzzz* where *x* is an alphabetic character (A—Z). (ZDT/IMS panels are named HFM1*zzzz* and ZDT/Db2 panels are named HFM2*zzzz*.)
2. Create a library with the same characteristics as HFM.SHFMPENU, with the name HFM.SHFMP*yyy*, where *yyy* is the same language code that you specified to change your print and display translation tables. (See 4.a on page 105.) Copy the required panel members from HFM.SHFMPENU to this library.
3. Change the required panel text in the members in your library. A panel may reference a **help** panel by means of a `.HELP` statement. If any panel you are changing contains any of these `.HELP` statements, also copy and change these referenced members in your library.

To use the panels you have translated, see Using the translated messages and panels on page 108.

Using the translated messages and panels

To use your translated messages in a batch job, specify the appropriate language with the LANGUAGE processing option, using the keywords shown in Table 18: Keyword values for the LANGUAGE option on page 108. See Changing the default options on page 55 for information on how to do this.

Table 18. Keyword values for the LANGUAGE option

Language	Code	Specify on LANGUAGE option...
French	FRA	FRENCH
German	DEU	GERMAN
Italian	ITA	ITALIAN
Japanese	JPN	JAPANESE
Portuguese	PTG	PORTUGUESE
Spanish	ESP	SPANISH
Danish	DAN	DANISH
Upper case English	ENP	UPPERENG
Korean	KOR	KOREAN
Swiss German	DES	SGERMAN
Traditional Chinese	CHT	CHINESET
Simplified Chinese	CHS	CHINESES
Other	XXX	OTHER

For example, to use French messages, specify LANGUAGE=FRENCH.

Under ISPF, the language used in messages and panels is determined by the setting of the national language for ISPF, for the current ISPF session. For information on changing the national language setting for your ISPF session, see *z/OS ISPF Dialog Developer's Guide*.

When your ISPF session is set up to use your language, you need to add your libraries to the appropriate ISPF concatenation, in front of any Z Data Tools English libraries. For example, to use your translated messages, add HFM.SHFMM*yyy* to ISPMLIB, in front of HFM.SHFMMENU. To use your translated panels, add HFM.SHFMP*yyy* to ISPPLIB, in front of HFM.SHFMPENU.

Customizing for the Japanese national language

Modifying the Japanese translation tables

The Z Data Tools Japanese component provides translation tables for display and print that are used by Z Data Tools when the Japanese language is selected. You can change these Japanese translation tables if necessary. You use the usermod HFMUMODJ and the member HFMTRJPN to change the Japanese translation tables. HFMUMODJ and HFMTRJPN are distributed in HFM.SHFMSAM1.

To do this:

1. Copy the member HFMTRJPN from HFM.SHFMSAM1 to your own source library.
2. Change the translation table definition statements in the HFMTRJPN source member in your library, according to your requirements.

   Z Data Tools provides support under ISPF for command verbs and command keywords entered in lower or uppercase as long as they are valid SBCS-only strings, and the a - z codepoints in the host codepage used correspond to the standard EBCDIC a-z codepoints. If this is not the case, and you want to be able to enter lower or mixed case command verbs and command keywords, then you will need to modify the TRUPC translation table in HFMTRJPN, to specify the uppercase A - Z EBCDIC codepoints at locations corresponding to the a - z codepoints in the codepage used.
3. Modify the HFMUMODJ member in HFM.SHFMSAM1 to meet your site's requirements. Refer to the usermod for information about changes you might need to make.
4. Install SMP/E usermod HFMUMODJ.



Note: Under ISPF, to enable user DBCS data to be displayed or edited, the terminal type must be set to 3277KN or 3278KN, using ISPF option 0.

Changing the Japanese message text

If you have installed the Z Data Tools Japanese component, all Z Data Tools Base function Japanese messages are stored in the HFM0MJPN source member. Normally you should not need to modify this module. However, if you do want to modify it, you can do so by means of the usermod, HFMUMODN.

To do this:

1. Copy the member HFM0MJPN from HFM.SHFMSAM1 to your own source library.
2. Change the message text in HFM0MJPN in your library.
3. Modify the HFMUMODN member in HFM.SHFMSAM1 to meet your site's requirements. Refer to the usermod for information about other changes you might need to make.
4. Install SMP/E usermod HFMUMODN.

Chapter 8. Customizing Z Data Tools Service Provider for z/OS® Connect

Z Data Tools Service Provider for IBM® z/OS® Connect Enterprise Edition enables client applications to access IBM® Z data sources using z/OS® Connect.

Using z/OS® Connect tooling, customers can create their own RESTful APIs and access IBM® Z data sources through the service provider. Clients can read data sequentially, by relative record position, or by key.

Installation and customization of the service provider is optional. It is only necessary for customers who plan to use this feature to access IBM® Z data sources through z/OS® Connect.

Data sources

The supported data sources include:

- MVS data sets (including VSAM)
- UNIX System Services files
- CICS® files (including VSAM and CICS® TS and CICS® TD queues)
- WebSphere MQ queues
- Db2®
- IMS™

Prerequisites

Z Data Tools Service Provider has the following prerequisites:

- Z Data Tools V17R1 and later
- Z Common Components V17R1.

  Z Common Components is provided with the Z Data Tools product and must be installed and configured as a prerequisite. The ZCC server must be running on the same z/OS system as z/OS Connect for the service provider to perform its function.

  Refer to *Z Common Components Customization Guide and User Guide* for more information.
- IBM® z/OS® Connect Enterprise Edition V3.0 (5655-CE3) or later

  z/OS® Connect Enterprise Edition V3.0 or later must be installed and configured before installing and configuring the Z Data Tools Service Provider. Refer to z/OS® Connect EE documentation in IBM® Knowledge Center for installation and configuration information.

Installing Z Data Tools Service Provider for z/OS® Connect

The Service Provider feature is extracted to UNIX® System Services, then installed into z/OS® Connect and configured in ZCC server.

The Z Data Tools Service Provider for z/OS® Connect feature is shipped as a binary archive in the SHFMSAM1 data set as member HFMZCEE.

Generic or specific mapping of data sources

z/OS® Connect Enterprise Edition provides tools to create APIs that use the Z Data Tools Service Provider. During API creation, a service archive (SAR file) is associated with the API which identifies the target service provider.

Z Data Tools service archives (zdt*.sar) and the Z Data Tools z/OS Connect Build Toolkit plug-in (com.hcl.zosconnect.buildtoolkit.zdatatools.jar) are provided for this purpose.

zdtService.sar, zdtIMSService.sar, and zdtDB2Service.sar are ready-to-use SAR files that can be associated with z/OS® Connect APIs during API creation. They provide a *generic* mapping of file data to a prescribed response format, that is, one format for all data sources. When using generic mappings, you do not need to create SAR files.

Alternatively, customers who require a response mapping *specific* to a data resource can create their own SAR files. To create and manage your own SAR files you must install the Z Data Tools z/OS Connect Build Toolkit plug-in.

To understand if you will use the *specific* or *generic* response mapping methods, and therefore whether you need to install the Build Toolkit plug-in or the provided SAR files or both, refer to *"Methods for accessing IBM Z data sources"* in the *Z Data Tools User's Guide and Reference*.

Installing Z Data Tools Service Provider on z/OS®

The Service Provider feature is extracted to UNIX System Services, then installed into z/OS Connect within WLP, and then configured in ZCC server.

Before you begin

The installation procedure is conducted in the UNIX® System Services environment and has the following environmental prerequisites:

- The shell environment must have a z/OS® Connect EE-compatible version of Java™ available on the environment PATH variable. You can check the PATH variable by using the UNIX® System Services echo $PATH command.
- The z/OS® Connect installation directory `<ZCON_INST_DIR>/v3r0/wlp/bin` must be in the PATH environment variable.

About this task

The Z Data Tools Service Provider for z/OS® Connect feature is shipped as a binary archive in the SHFMSAM1 data set as member HFMZCEE. The archive contains the following files:

com.hcl.zosconnect.zdt.provider.feature_n.n.n.n.esa	Feature archive
zdtService.sar	Z Data Tools Base component service archive
zdtIMSService.sar	IMS™ service archive

zdtDB2Service.sar	Db2® service archive
zdatatools.properties	Properties file
com.hcl.zosconnect.buildtoolkit.zdatatools.jar	Build Toolkit plug-in

1. Extract the Z Data Tools Service Provider for z/OS® Connect archive from the SHFMSAM1 data set using the UNIX System Services pax command.
   Example
   ```
   pax -rf "//'<hlq>.SHFMSAM1(HFMZCEE)'"
   ```
   Result
   This command extracts the HFMZCEE archive into the current working directory.

2. (Optional) Edit the productInstall option in the properties file if you want to change the installation directory.
   Example
   ```
   com.ibm.websphere.productInstall=</your/preferred/directory>
   ```
   This is the location where the Z Data Tools Service Provider is installed. The directory must be fully qualified. That is, it must begin with a forward slash (/). The default installation directory is /var/zdatatools. If you elect to change the default directory, it is recommended that you choose one that does not already contain existing files or sub-directories. The productInstall directory is created if it does not already exist.

3. Copy the properties file to the z/OS® Connect *<WLP_USER_DIR>*`/v3r0/extensions` directory using the UNIX System Services cp command.
   Example
   For example:
   ```
   cp zdatatools.properties /var/zosconnect/v3r0/extensions
   ```

4. Copy the service archive files to the z/OS® Connect service directory.
   Example
   For example:
   ```
   cp zdt*.sar
      /var/zosconnect/servers/<yourWLPServer>/resources/zosconnect/services
   ```
   Ensure that the service archive files are readable by the z/OS® Connect WLP started task user.

5. Install the Z Data Tools Service Provider feature into z/OS® Connect using the installUtility command. For example:
   Example
   ```
   installUtility install
   com.hcl.zosconnect.zdt.provider.feature_2.0.0.0.esa --to=zdatatools
   ```
   Ensure that the ESA file version number matches the file name in the archive.

6. Edit the z/OS Connect WLP `server.xml` configuration.

   The server.xml requires the following changes:

a. Add the Z Data Tools feature to the featureManager stanza:

Example

```
<featureManager>
 <feature>zdatatools:zdtProvider-2.0</feature>
</featureManager>
```

b. Add the runport and maximum timeout configuration for the feature:

Example

```
<ZDataTools_Connection id="default" runport="2800" max_timeout="1800" />
```

where runport represents the ZCC server port that is used to establish Z Data Tools sessions, and max_timeout represents the maximum number of seconds for which a client can hold a Z Data Tools session open without activity.

7. Change the ZCC server JCL to support z/OS Connect REST requests.

Example

In the HFISRV CONFIG DD statement, add REST=YES after the CONFIG=ZDT stanza. For example:

```
//CONFIG DD *
 CONFIG=ZDT
 REST=YES
 …
/*
```

8. Restart or refresh the z/OS Connect WLP task and ZCC server task.

What to do next

To verify a successful installation of the Z Data Tools Service Provider, do the following steps.

1. Check the JOBLOG of the z/OS Connect WLP started task and verify that the CWWKF0012I message lists `zdtProvider-2.0` as one of the installed features. For example:

```
CWWKF0012I: The server installed the following features: ssl-1.0,
jdbc-4.1, zdatatools:zdtProvider-2.0, …
```

2. Invoke the Z Data Tools Service Provider manually, using an HTTP PUT request to `https://zceehost:zceeport/zosConnect/services/zdtService?action=invoke` with the following payload:

```
{
    "action" : "verify"
}
```

This can be done from a browser or a REST API tool such as POSTMAN. The HTTP verb must be PUT or POST, and a `Content-Type: application/json` header is required. A HTTP status code of 200 indicates that the Z Data Tools Service Provider has been installed correctly.

If installation verification fails, check that all installation steps have been completed in the designated order. The *Z Data Tools User's Guide and Reference* also provides a troubleshooting topic for the Z Data Tools Service Provider for z/OS Connect.

Installing the service archive files

z/OS® Connect API creation tools run outside z/OS® in an Eclipse environment. If you want to use the generic Z Data Tools service archive files (zdtService.sar, zdtIMSService.sar, or zdtDB2Service.sar) you must make them available to the Eclipse environments in which they will run. The SAR files must be copied to the relevant Eclipse environments, or to a network folder where the relevant Eclipse environments can access them.

The service archive files reside in the UNIX System Services directory where the pax archive was expanded. They must be copied in binary mode.

Installing the Build Toolkit plug-in

This topic explains how to install the Build Toolkit plug-in for use with the zconbt command.

About this task

z/OS® Connect provides a Build Toolkit (zconbt) for creating service archive (SAR) files. SAR files are used during z/OS® Connect API creation to associate APIs with service providers such as Z Data Tools Service Provider.

The z/OS® Connect EE Build Toolkit is bundled with the z/OS® Connect host product. It provides the zconbt command, which runs in Windows, Linux, and z/OS® environments.

1. If you have not already done so, extract the z/OS® Connect Build Toolkit to the environment (Windows, Linux or z/OS®) where you intend to build your SAR files.
2. Copy the Z Data Tools Build Toolkit plug-in to the Build Toolkit `/plugins` directory.
   Example
   For example:
   ```
   cp com.hcl.zosconnect.buildtoolkit.zdatatools.jar /var/zconbtv3/plugins
   ```
3. Edit the `plugin.properties` file in the z/OS® Connect Build Toolkit `/lib` directory, and add an entry for the Z Data Tools plug-in.
   Example
   For example, edit `/var/zconbtv3/lib/plugin.properties` and then add the following entry:
   ```
   zdatatools=com.hcl.zosconnect.buildtoolkit.zdatatools.ZDataToolsSarGenerator
   ```
4. You can now use the zconbt command to create SAR files for Z Data Tools APIs.
   Example
   For example:
   ```
   cd /var/zconbtv3/bin
   zconbt.zos -p=yourSAR.properties -f=./yourSAR.sar
   ```
5. All service archives created using the Z Data Tools Build Toolkit plug-in must be copied to the z/OS Connect `/zosconnect/services` directory.
   Example
   For example:

```
cp yourService.sar /var/zosconnect/servers/<yourWLPServer>/resources/zosconnect/services
```

Copied archives must be readable by the z/OS Connect WLP STC user.

Related information

The z/OS Connect EE build toolkit

Creating services with the build toolkit

Using the Build Toolkit plug-in

TLS for the Build Toolkit plug-in

During SAR creation, the Build Toolkit plug-in needs to connect to an instance of the ZCC server.

If your HFISRV server is configured to require clients to connect using transport layer security (TLS), you will need to install the trust chain of your server's digital certificate in the environment where the Build Toolkit runs.

The Z Data Tools Build Toolkit plug-in uses the default trust store of the JVM environment in which it runs, for example: `$JAVA_HOME/jre/lib/security/cacerts`

If your default trust store does not already contain the Certificate Authority (CA) trust chain of your server certificate, you will need to export your CA certificate chain and import it into the default trust store. For example:

```
RACDCERT SITE EXPORT(LABEL('ZCC Server
                       Certificate')) DSN(ZCC.CERT)
   FORMAT(CERTDER)
```

Note that there may be multiple certificates in the trust chain.

Once you have the CA certificates of your server certificate, you can import them into the JVM default trust store using your preferred tooling. For example, using Java™ keytool:

```
keytool -import -trustcacerts -keystore jre\lib\security\cacerts -storepass changeit
 -noprompt -alias ZCC -file C:\certs\zcc.cert
```

With the CA trust chain in place, the Z Data Tools Build Toolkit plug-in will be able to establish a TLS connection with your HFISRV server.

Maintaining Z Data Tools Service Provider

If service is applied, some maintenance steps are necessary.

Before you begin

Note: If the service provider is not already installed into the z/OS Connect environment, then maintenance is not applicable, and the installation steps should be followed instead.

The maintenance procedure is conducted in the UNIX® System Services environment and has the following environmental prerequisites:

- The shell environment must have a z/OS® Connect EE-compatible version of Java™ available on the environment PATH variable. You can check the PATH variable by using the UNIX® System Services echo $PATH command.
- The z/OS® Connect installation directory *<ZCON_INST_DIR>*`/v3r0/wlp/bin` must be in the PATH environment variable.

About this task

The Z Data Tools Service Provider for z/OS® Connect feature is shipped as a binary archive in the SHFMSAM1 data set as member HFMZCEE. The archive contains the following files:

com.hcl.zosconnect.zdt.provider.feature_n.n.n.n.esa	Feature archive
zdtService.sar	Z Data Tools Base component service archive
zdtIMSService.sar	IMS™ service archive
zdtDB2Service.sar	Db2® service archive
zdatatools.properties	Properties file
com.hcl.zosconnect.buildtoolkit.zdatatools.jar	Build Toolkit plug-in

The following procedure includes stopping and restarting the z/OS® Connect WLP server. In high-availability environments this may be undesirable. For alternative procedures refer to the following topics:

- Stopping and starting z/OS Connect EE servers in an HA environment
- Management of z/OS Connect EE APIs, services, and API requesters in an HA environment

1. Extract the Z Data Tools Service Provider for z/OS® Connect archive from the SHFMSAM1 data set using the UNIX System Services pax command.
   Example
   ```
   pax -rf "//'<hlq>.SHFMSAM1(HFMZCEE)'"
   ```
   Result
   This command extracts the HFMZCEE archive into the current working directory.
2. Copy the service archive files to the z/OS® Connect service directory.
   Example
   For example:

```
cp zdt*.sar
   /var/zosconnect/servers/<yourWLPServer>/resources/zosconnect/services
```

Ensure that the service archive files are readable by the z/OS® Connect WLP started task user.

3. (Optional) If you use the zconbt command to create SAR files, copy the Z Data Tools Build Toolkit plug-in to the Build Toolkit `/plugins` directory.
   Example
   For example:

   ```
   cp com.hcl.zosconnect.buildtoolkit.zdatatools.jar /var/zconbtv3/plugins
   ```

4. Check APAR HOLDDATA before continuing to the next step in case there are additional steps that need to occur at this point.

5. Stop the z/OS Connect WLP server.

6. Uninstall the current Z Data Tools Service Provider using the installUtility command.
   Example
   For example:

   ```
   installUtility uninstall zdatatools:zdtProvider-2.0
   ```

7. Reinstall the Z Data Tools Service Provider feature into z/OS Connect using the installUtility command.
   Example
   For example:

   ```
   installUtility install
   com.hcl.zosconnect.zdt.provider.feature_2.0.0.0.esa --to=zdatatools
   ```

   Note that the .esa filename may vary by version.

8. Restart the z/OS Connect WLP server.

Uninstalling Z Data Tools Service Provider

Uninstall the Service Provider from z/OS® Connect using the installUtility command.

Before you begin

The uninstall procedure is conducted in the UNIX® System Services environment and has the following environmental prerequisites:

- The shell environment must have a z/OS® Connect EE-compatible version of Java™ available on the environment PATH variable. You can check the PATH variable by using the UNIX® System Services echo $PATH command.
- The z/OS® Connect installation directory <*ZCON_INST_DIR*>`/v3r0/wlp/bin` must be in the PATH environment variable.

About this task

The following procedure includes stopping and restarting the z/OS® Connect WLP server. In high-availability environments this may be undesirable. For alternative procedures refer to the following topics:

- Stopping and starting z/OS Connect EE servers in an HA environment
- Management of z/OS Connect EE APIs, services, and API requesters in an HA environment

1. Stop the z/OS Connect WLP server.
2. Uninstall the Z Data Tools Service Provider feature using the installUtility command.
   Example
   For example:
   ```
   installUtility uninstall zdatatools:zdtProvider-2.0
   ```
3. Remove the properties file from the z/OS® Connect *<WLP_USER_DIR>*`/v3r0/extensions` directory using the UNIX System Services rm command.
   Example
   For example:
   ```
   rm /var/zosconnect/v3r0/extensions/zdatatools.properties
   ```
4. Remove the service archive files from the z/OS® Connect service directory.
   Example
   For example:
   ```
   rm /var/zosconnect/servers/<yourWLPServer>/resources/zosconnect/services/zdt*.sar
   ```
   Tip: You might also decide to remove any other Z Data Tools SAR files from this directory.
5. Edit the z/OS Connect WLP `server.xml` configuration.

   The server.xml requires the following changes:

   a. Remove the entry for the Z Data Tools feature from the `<featureManager>` stanza.
   b. Remove the `ZDataTools_Connection` configuration stanza.
   c. If you have added any other `ZDataTools_Connection` configuration stanzas for IDs other than *"default"*, remove these also.
6. Restart the z/OS Connect WLP server.

Chapter 9. Customizing Z Data Tools to use library management system libraries

Z Data Tools provides the facility to allow you to access source code stored in either library management (LM) or source code management (SCM) systems.

The difference between library management systems and source code management systems is not well defined, but in general, SCM systems tend to provide software packaging, version management of packages, and build management in addition to the typical LM functions. LM systems tend to be simpler and focus on efficiently storing data (files) in a library and managing versions or levels of the individual files. An example of a library management system is CA-Panvalet. An example of a source code management system is SCLM.

In this chapter both LMs and SCMs are referred to as library management systems or LMSs.

There are two possible scenarios to accessing source code in LMSs:

- The LMS is CA-Panvalet.
- The LMS (not CA-Panvalet) provides a SUBSYS interface.

If you want to use Z Data Tools to access COBOL copybooks, PL/I include books, or HLASM copybooks stored in LMS libraries, the customization tasks you must perform are different, depending on which of the above scenarios applies to your LMS. The possible customization tasks are listed in Table 19: Summary of steps for customizing Z Data Tools to use LMS libraries on page 120.

Table 19. Summary of steps for customizing Z Data Tools to use LMS libraries

Step	Description
__ 1	Determine if your LMS provides a SUBSYS interface.
__ 2	Set the LMS and LMSUBSYS options, as necessary. See LMS on page 404.
__ 3	Ensure that the correct release of Language Environment® is available to Z Data Tools. See the third bullet point in Accessing source code in an LMS with SUBSYS interface on page 121.

Accessing source code in CA-Panvalet libraries

The CA-Panvalet interface enables Z Data Tools to work with COBOL copybooks, PL/I include books, and HLASM copybooks stored in CA-Panvalet libraries.

Before you can use this, you must change the default options to specify LMS=PANVALET.

If you want to access only CA-Panvalet libraries, specify LMS=PANVALET.

For information about the LMS option, see LMS on page 404. For more information on how to change the default options, see Changing the default options on page 55.

If you do not change the LMS option to LMS= PANVALET, you will receive an error message if you attempt to access copybooks in a CA-Panvalet library.

Note:

1. This support is provided for CA-Panvalet release 14 and above.
2. Ensure that your CA-Panvalet library is defined either with DSORG=DA, or with DSORG=PS and LRECL=0.

Other library management systems

Accessing source code in an LMS with SUBSYS interface

If your LMS provides a SUBSYS interface, you need to perform tasks #2 and #3 from Table 19: Summary of steps for customizing Z Data Tools to use LMS libraries on page 120, as described here.

- Set the LMSUBSYS option to the value for the SUBSYS interface provided by your LMS vendor, (for example, LMSUBSYS=LAM).
- Ensure that the LMS option is set to NO.
- Ensure Language Environment® is available to Z Data Tools, if you have not already done so.

  You can do this by one of the following methods:
  - Add the Language Environment® runtime library, SCEERUN, to your concatenated LINKLIST.
  - Place SCEERUN in the STEPLIB DD statement of the logon proc.
  - Your users can use the TSO command, TSOLIB, to add SCEERUN to the search list.

Note: You cannot access source code from an LMS with a SUBSYS interface and also from CA-Panvalet. The two are mutually exclusive.

Chapter 10. Customizing Z Data Tools to use the Optim™ Data Privacy Provider API

To enable Z Data Tools to specify and run ODPP commands, you must first allocate or define a DD name HFMODPP. This name points to a data set that contains the environment variables that are required by the ODPP to run as a TSO or batch application.

Example

```
//HFMODPP DD DISP=SHR,DSN=OPTIM.ODPP.SAMPLIB(ENVVARS)
```

Where the data set member ENVVARS contains the following lines:

```
LIBPATH=/usr/local/odpp/odppbin
ODPPLL=/usr/local/odpp/licensefiles
ODPPERRL=/usr//local/odpp/odppbin
ODPPTRCL=/usr/local/odpp/tracefiles
ODPPTRC=N
```

Customize the names to match the path names that are specified in the installation of the ODDP API. Refer to the relevant ODPP User's Guide and ODPP Installation Guide for more details on the environment variables required for your installation.

If the HFMODPP DD is allocated by JCL or TSO allocation statement, Z Data Tools attempts to initialize the framework that is required for running ODPP commands. If the framework initialization is successful then you are able to define the ODPP commands as a scrambling attribute for any given field in a Z Data Tools template. The commands are run when you run a copy operation with an input and output template and the receiving field has the scrambling option of ODPP with the associated ODPP command. For more information, see the *Z Data Tools User's Guide and Reference*.



Note:

1. For TSO/ISPF users. Only one instance of Z Data Tools can have the framework initialized. Therefore, when running split sessions under ISPF, only one session can have an active Z Data Tools session that can support specifying and running ODPP commands.
2. You might need to increase your region sizes when running Z Data Tools to accommodate the ODPP API. You receive a Language Environment® message that indicates *"not enough storage"* when you start Z Data Tools if your region size is insufficient. Run with a minimum of 100 MB.
3. The IBM® Optim™ Data Privacy Solution on z/OS® needs to be Version 11 or above.

Chapter 11. Customizing Z Data Tools to use an I/O exit

Z Data Tools allows you to provide a user I/O exit for use when processing data. This enables you to process records in data sets which are subject to any kind of pre- or post-processing not offered directly by Z Data Tools. Such processing could include compression, encryption, or other site-specific activities performed on records in a data set.

There are a number of ways that data sets can be processed, and how the processing is made visible to users. Therefore Z Data Tools allows you to provide an I/O exit, to be associated with a particular data set, and to specify the name of the exit. The I/O exit is a program that interfaces with Z Data Tools using a predefined communication area.

This I/O exit is not intended to fully support compressed or encrypted data sets. Specifically, it does not provide member name list access to data sets which are in other than PDS or PDSE format. However, you can access members of a proprietary compressed library (PDS or other format) through the I/O exit, as long as the member name is known and is provided to Z Data Tools.

The Z Data Tools I/O exit operates only on the data, not the data set, and supports the following functions:

- Interactive view, edit and browse in the Z Data Tools Base component
- Batch and interactive utilities:
    - Data set copy (DSC)
    - Data set compare (DSM)
    - Data set edit in batch (DSEB)
    - Data set generate (DSG)
    - Data set print (DSP)
    - Data set update (DSU)
    - Find/change (FCH)

You cannot use the Z Data Tools I/O exit to operate directly on COBOL copybook or PL/I include data sets, or on template data sets. The I/O exit, when specified, can apply only to the data being processed, not to the template or copybook data sets used to format or select the processed data. However, the I/O exit can be used when the template or copybook data set is itself the subject of a function.

You use the library management system (LMS) exit to process COBOL copybook or PL/I include data sets, or template data sets, when associated with an input or output data set. For information on providing an LMS exit, see Customizing Z Data Tools to use library management system libraries on page 120 and The library management system exit on page 539.

To specify that an I/O exit can be used, set the USEIOX option, in HFM0POPT. If required, you can also provide the name of a site-specific I/O exit. For more information about USEIOX, see USEIOX on page 422. For information about changing the options in HFM0POPT, see Changing the default options on page 55. This exit name can be overridden by the user interactively or in batch. *Z Data Tools User's Guide and Reference* describes how to use the I/O exit.

Exit protocol

The I/O exit provides record-level support. Z Data Tools performs all the I/O functions and the exit can process the record after it is read or before it is written. The exit supports the following protocol:

Initialization

The exit indicates that the specified data set is to be processed by the exit and returns information about supported functions (once per data set).

Open

Z Data Tools opens the data set and provides vital information about the data (multiple possible per data set).

A series of record access calls implementing the following functions:

- Record read
- Record write

The record to be processed is provided in an input buffer. The exit should process the data and return it in the output buffer while setting the output (record) length. A length of zero for the output record indicates that the processing did not change the record and the input record can be used "as is".

At any time during read/write processing the exit can indicate that it should no longer process the current data set. Z Data Tools will then stop invoking the exit for read/write processing and continue processing without the involvement of the exit. The Close and Terminate calls will still occur.

Note: The read and write requests may occur in random record order. No assumption about data processing should be made based on such order. This is important with some encryption technologies which require a sequentially ordered data stream.

Error codes can be set and messages returned by the exit to be displayed or printed for the user. Some codes immediately terminate the current function while some allow for continued processing. The messages are displayed regardless of the return code every time there is a non-blank message returned by the exit. See also Information that should always be returned from the I/O Exit to Z Data Tools on page 134 for information about messages returned by your exit.

Close

Z Data Tools closes the data set after returning from the exit.

Termination

The exit performs any required 'cleanup' processing.

Writing your exit

Z Data Tools does not supply a default I/O exit. If you plan to use an I/O exit to process your data, you must provide one or more for your installation. You must also set the USEIOX option in HFM0POPT. For information about the USEIOX option, see USEIOX on page 422.

Any exit you provide must be in the form of a load module, in any load library available to Z Data Tools, either by a STEPLIB DD statement, or in LINKLIST, or LPALIST. If Z Data Tools attempts to load an exit and is unable to find it, an error message is displayed.

You can write your own exit in any high level language, for example, COBOL, PL/I, or High Level Assembler. Sample exits for COBOL, PL/I, and HLASM are provided in HFM.SHFMSAM1. They are:

HFMIOXEA

Sample HLASM exit

HFMIOXEC

Sample COBOL exit

HFMIOXEP

Sample PL/I exit

HFMIOXHF

Sample HLASM exit using z/FS

Copybooks providing the I/O exit control blocks for COBOL, PL/I, and HLASM are distributed in the Z Data Tools macro library, HFM.SHFMMAC1. They are:

HFMIOXCB

Control block for HLASM programs

HFMIOXCC

Control block for COBOL programs

HFMIOXCP

Control block for PL/I programs

These copybooks are described in Exit control block data names on page 126. See Using the I/O exit control block on page 133 for information on how to write your exit using the copybooks, and the processing your exit should perform.



Note:

1. The Z Data Tools I/O exit does not support 24-bit addressing mode.
2. Z Data Tools supports an exit written in any supported release of COBOL and PL/I.
3. Z Data Tools returns general information about the data set, which can be interpreted by your exit routine. Some data in the exit control blocks, (for example, DSORG, RECFM), mimic data used in z/OS® system control blocks. Therefore, for convenience, in the COBOL copybook, HFMIOXCC, level-88 values are defined.
4. Z Data Tools also processes z/FS files as simulated QSAM files. For more information about Z Data Tools and z/FS files, refer to section *"Using UNIX™ System Services and the Hierarchical File System"* in the *Z Data*



Tools User's Guide and Reference. There are no major differences on the exit routine level. An exit routine will get an address to a path name rather than the data set name in this case, but the way a (simulated) record is processed remains the same.

Exit control block data names

Table 20. Exit data names

Description	Data type	Set by:	Value
Eye catcher	CL8	ZDT initialization	"UIOEXCB"
Interface version	F	ZDT initialization	**1** Initial value
Level support	F	I/O exit initialization	**0** Process without exit **1** I/O performed by Z Data Tools **2** Reserved for future use
Function code	F	ZDT, for every call to the exit	**1** Initialize once per data set **2** Terminate once per data set **3** Open, multiple possible per data set **4** Close, multiple possible per data set **5** Read, required **6** Write, required **7** Reserved for future use **8** Reserved for future use

Table 20. Exit data names (continued)

Description	Data type	Set by:	Value
			9 Reserved for future use
Exit return code	F	I/O exit	**0** OK, default. See note 1. **4** Warning **8** Error, terminate function **12** Severe error, terminate function **16** Fatal error, terminate function
Warning code	F	I/O exit	**0** Normal warning, default. See note 1. **1** Reserved for future use **2** Complete, continue processing without exit
Functions supported by exit	X	I/O exit initialization	**1** Read, required **2** Write, required **3** Read + write, default **32** Reserved for future use **64** Reserved for future use

Table 20. Exit data names (continued)

Description	Data type	Set by:	Value
			128 Reserved for future use
Operating mode	X	ZDT initialization	**1** TSO in batch **8** Keyword mode **16** Command mode **32** Query/answer mode **64** Fullscreen ISPF **128** Batch, without TSO
DDNAME	CL8	ZDT initialization	DDname allocated by Z Data Tools
DSN	CL44	ZDT initialization	fully qualified DSN
Member name	CL8	ZDT initialization	Member name, only if the data set is opened as sequential, not PO
VOLSER	CL6	ZDT initialization	First VOLSER of the data set, set to blank if not used
DSORG	2X	ZDT open	Data set organization: 1st byte: **1** Unmovable **2** PDS(E) **32** DA **64** PS

Table 20. Exit data names (continued)

Description	Data type	Set by:	Value
			128 IS 2nd byte: **1** z/FS **8** VSAM **16** IAM involved
VSAM catalog entry	X	ZDT open	**8** Path **16** Alternate index **32** Index **64** Data **128** Cluster
VSAM type information	X	ZDT open	**4** IAM **8** LDS **16** VRRDS **32** RRDS **64** ESDS

Table 20. Exit data names (continued)

Description	Data type	Set by:	Value
			128 KSDS
RECFM	X	ZDT open	**2 - M** Machine control character **4 - A** ASA control character **8 - S** Spanned **16 - B** Blocked **32 - T** Track overflow **64 - V** Variable **128 - F** Fixed **128+64 - U** Undefined
Allocation mode	X	ZDT initialization	**1** Old **2** Mod **4** New **8** (new,catlg) **16** Shr **128** JCL/DD allocated

Table 20. Exit data names (continued)

Description	Data type	Set by:	Value
Open mode	X	ZDT open	**1** Input **2** Output **4** Update/inout **8** Initial load (VSAM)
Processing mode	X	ZDT initialization	**128** Temporary file used in support edit of data (dsn - original name preserved)
LRECL	F	ZDT open	record length (non-VSAM)
BLKSIZE	F	ZDT open	block size (non-VSAM)
MLRECL	F	ZDT open	maximum record length (VSAM)
CISZ	F	ZDT open	CI size (VSAM)
KEYLEN	F	ZDT open	key length
RKP	F	ZDT open	relative key position (relative to zero offset into the IO area)
RBA	XL8	ZDT or I/O exit read	XRBA (VSAM)
SLOT	F	ZDT or I/O exit read	slot number (VSAM, RRDS/VRRDS)
MAXL	F	I/O exit open	maximum record length ever to be returned by the exit after read and decompression/decoding. The initial value is the same as that for the file on DASD.
Estimated raw (that is, uncompressed) data set size, as a percentage of the original size. See note 2.	F	I/O exit open	**0** Unknown **nnn** Percentage of the original size **100** Initial value

Table 20. Exit data names (continued)

Description	Data type	Set by:	Value
Current input record pointer	A	ZDT, all IO	I/O area address passed to the exit (size at least as big as the maximum record length) holds the record to be processed by read/write
Current input record length	F	ZDT, all IO	length of record passed to the exit
Current output record pointer	A	ZDT, all IO	I/O area address passed to the exit (size at least as big as the maximum record length) for the output record, holds the record being the result of processing by read/write
Current output record length	F	I/O exit, all IO	Length of record passed from the exit, if zero then the input record will be assumed to be unchanged and the output record will be ignored
z/FS path name pointer	A	ZDT initialization, open	Address of an z/FS path name
z/FS path name length	F	ZDT initialization, open	Length of an z/FS path name
z/FS object type	X	ZDT initialization, open	z/FS object type specified by the z/FS path: **4** Socket **8** Symbolic link **16** FIFO **32** Regular file **64** Special character file **128** Directory **Note:** Only regular files are supported. Directory can be set at initialization only. Other types are added for completeness.
z/FS processing mode	X	ZDT initialization, open	z/FS processing mode used in an z/FS file processing:

Table 20. Exit data names (continued)

Description	Data type	Set by:	Value
			64 Binary mode (records of fixed, arbitrary defined, length) **128** Text mode (record boundaries determined by delimiters)
Message	CL80	I/O exit, all	(Error) message - can be provided for output after each operation, non-error messages will be suppressed after 100 messages per session. See note 3.
User scratchpad area	CL1024	UIOEX, all	Maintained by the exit from INIT to TERM

Note:

1. Zero is the default. This code is initialized to zero on every call from Z Data Tools to the exit, before the control block is passed to the exit.
2. This is a compression ratio, where 100 means the same size as the original data. 100 is the initial value. See Open call discussion on page 135 for more information, and examples of a compression ratio.
3. See Information that should always be returned from the I/O Exit to Z Data Tools on page 134 for more information about message suppression.

Using the I/O exit control block

The I/O exit control block is a single parameter that is passed with each call from Z Data Tools to the I/O exit and returns data from the I/O exit to Z Data Tools.

Function codes and the flow of I/O exit processing

The I/O exit control block contains a function code field. This is set by Z Data Tools before the I/O exit is called. It indicates the type of I/O function being performed by Z Data Tools. The I/O exit is given the opportunity to perform actions at the points named by the function codes during Z Data Tools processing. The function codes are also discussed in Exit protocol on page 124. They are: Initialization, Termination, Open, Close, Read and Write.

Note that Z Data Tools makes the "real" I/O calls to the operating system. As a result, the I/O exit does not get control at exactly the time of the real I/O operation. Instead, the I/O exit gets control, as follows:

- **Open:** After the "real" data set open
- **Close:** Before the "real" close

- **Read:** Immediately after a read from DASD (before Z Data Tools truncation or padding or other changes)
- **Write:** Immediately before a write to DASD (after Z Data Tools truncation or padding or other changes)

The Function Code is the key input that your I/O exit should use to determine what actions to take. The following sections discuss the flow of control and data during I/O exit processing by focussing on the call type, as determined by the function code.

Information that should always be returned from the I/O Exit to Z Data Tools

The I/O exit should set any non-zero exit return code as appropriate. The exit return code field is initialized to zero on every call from Z Data Tools to the exit.

The I/O exit can also set an informational, warning, or error message into the message field of the I/O exit control block. You can code your exit to return any number of messages, however Z Data Tools will suppress all informational (RC=0) and warning (RC=4) messages after the first 100. Error messages (RC=8 or higher) are never suppressed. The Z Data Tools message:

```
Number of messages issued by exit reached the limit. Remaining non-error (RC < 8)
messages will be suppressed
```

is issued after 100 non-error messages have been produced. The limit of 100 messages cannot be changed.

The I/O exit can set the warning code to 2 at any time, to tell Z Data Tools to continue processing and not call the I/O exit again. The I/O exit will still be called for termination and, if an open was made, for close. For Z Data Tools to check the warning code, the I/O exit must also set the exit return code to 4 (warning).

Initialization call discussion

The I/O exit is always called, for any particular data set, first with one (and only one) initialization call. This is an opportunity for the exit to indicate the type of processing it will provide for the data set, for example, to allocate storage.

At initialization, the exit should decide whether to process the data set, and, if necessary, obtain dynamic storage.

- **Deciding whether to process the data set:** the data set name and other information is available in the I/O exit control block. If the exit "knows" from the data set name and other information that it should not process the data set, the exit should indicate that Z Data Tools should not call the exit again. There are two ways to do this:
  - Only at initialization, you can set the level support field to 0. This indicates that Z Data Tools should not call the exit.
  - In response to any I/O exit call, you can set the exit return code to 4 (warning), and set the warning code to 2, to avoid further calls, and let Z Data Tools handle the remaining I/O without the exit.

  Otherwise set the level support field to 1 to indicate that the exit will process the data set.
- **Obtaining dynamic storage:**

- Typically only an assembler exit needs to obtain storage. COBOL and PL/I storage and addressability is preserved between calls to the exit. However, Z Data Tools provides both a current input and a current output record buffer, so even an assembler exit may not need to allocate storage.
- Pointers to allocated storage and other information can be saved in the user scratchpad area at the end of the I/O control block, and will be maintained between initialization and termination calls.

There is no reason to change the values in the **Functions supported by exit** field. Both read and write support are required, and this is the default.

Termination call discussion

The last call for a given data set is always one (and only one) termination call. The termination call is intended primarily to allow de-allocation of resources allocated at initialization time. Z Data Tools makes the termination call even if exit processing was terminated by an error or a request from the exit to continue without calling the exit again.

Open call discussion

The I/O exit may receive multiple consecutive open calls, for various reasons internal to Z Data Tools. If the exit receives at least one open call, then the exit will receive at least one close call.

The open function code means the exit is being called immediately after a "real" file open. Additional information is now available (compared to what was available at initialization). The exit should:

- Decide whether to continue processing the data set. (See Initialization call discussion on page 134.)
- Set the MAXL (maximum data length) field in the I/O exit control block. There is no need to change this value if the record length is not to be changed by the exit.
- Set the estimated raw data set size, as a percentage of the original. For example, setting the estimated raw data set size to:

  50

  Indicates that an unencrypted data set is 50% of the encrypted data.

  100

  Indicates that an unencrypted data set will be the same size as the encrypted data set.

  200

  Indicates that the unencrypted data set will be twice the size of the encrypted data set (for example, as a result of decompression).

  This allows Z Data Tools to estimate the amount of in-memory storage that will be required. The sample exits in HFM.SHFMSAM1 use a setting of 200.

Close call discussion

The I/O exit may receive multiple consecutive close calls, for various reasons internal to Z Data Tools. If the exit receives at least one open call, then the exit will receive at least one close call.

The close function code means the exit is being called immediately prior to a "real" file close.

Read and Write call discussion

In this section:

- The I/O exit control block field "open mode" refers to the data set open operation, therefore "input" means open for input, and "output" means open for output.
- References in I/O exit control block fields to "current input record" refer to the data in the input buffers provided by Z Data Tools to the I/O exit. Similarly, references to "current output record" refer to the data in the output buffers.

Processing flow for read and write function codes:

- **During a read operation**, Z Data Tools reads a record from a data set and places it in the I/O exit current input record. Z Data Tools then calls the exit with a read function code. The exit should then copy the current input record to the current output record buffer, performing any required processing. On return from the exit, Z Data Tools uses the processed current output record from the exit.
- **During a write operation**, Z Data Tools takes an in-memory record and places it into the I/O exit current input record. The exit should then copy the current input record to the current output record buffer, performing any required processing. On return from the exit, Z Data Tools writes the processed current output record from the exit to DASD.

Important additional information to note about Read and Write

- In addition to setting the current output record data, the I/O exit should also set the current output record length during a read or write operation. If the current output record length is not set, or is set explicitly to zero, any changes in the current output record data buffer will be ignored, and the current input record will be used as the current output record.
- When processing a variable length sequential file, the data portion **does not** include the record descriptor word (RDW) containing the record length. The length is stored in a separate field in the control block.
- The I/O exit code should not assume that it is operating on records in sequential order. For example, the Z Data Tools editor might read or write individual records out of order.
- The I/O exit could receive read and write calls in any sequence. For example, the Z Data Tools Find/Change utility might read a number of records and then write a number of records, and then go back to reading.

Installing your exit

You can provide more than one user I/O exit. For example, you might want to provide a different exit for each user, or different exits for use with different data sets. You can only specify one exit name for the option USEIOX, so you could make this your site-specific or default exit. Individual users can specify different exits interactively or in batch.

Two usermods are provided, HFMUMODU and HFMUMODH, to enable you to install High Level Assembler exits under the control of SMP/E. HFMUMODU installs a basic exit written in HLASM. HFMUMODH installs an exit written in HLASM, which enables Z Data Tools to process data in an z/FS. Other exits written in HLASM, and exits written in other high level languages must be provided outside of SMP/E.

To provide a site-specific exit in HLASM, using HFMUMODU or HFMUMODH:

1. Change the USEIOX option in HFM0POPT. For more information on how to change the default options, see Changing the default options on page 55. For information about the USEIOX option, see USEIOX on page 422.
2. Code your own version of HFMIOXEA or HFMIOXHF, in your own source library. Refer to the information in Exit control block data names on page 126 and Using the I/O exit control block on page 133. You can use HFMIOXEA or HFMIOXHF in HFM.SHFMSAM1 as examples.
3. Modify the HFMUMODU or HFMUMODH members in HFM.SHFMSAM1 to meet your site's requirements. Refer to the usermods for information about changes you may need to make.
4. Install SMP/E usermods HFMUMODU or HFMUMODH. Your exit will be installed in HFM.SHFMMOD1 if you installed Z Data Tools into the default libraries.

To provide further exits in HLASM:

1. Code your own exit in your own source library. You can use HFMIOXEA or HFMIOXHF in HFM.SHFMSAM1 as examples.
2. Assemble and link your exit into a load library available to Z Data Tools, for example, HFM.SHFMMOD1.

JCL is provided in the job HFMIOXHA to help you to assemble and link your exit. Refer to the instructions in the job for changes you need to make. HFMIOXHA is distributed in HFM.SHFMSAM1.

To provide your own exit in COBOL or PL/I:

1. If you plan to use this exit as your site-specific or default exit, change the USEIOX option in HFM0POPT.
2. Code your own exit in your own source library. Refer to the information in Exit control block data names on page 126 and Using the I/O exit control block on page 133. You can use HFMIOXEC or HFMIOXEP in HFM.SHFMSAM1 as examples.
3. Compile and link your exit into a load library available to Z Data Tools, for example, HFM.SHFMMOD1.

JCL is provided in the job HFMIOXCO to help you to compile and link your exit in COBOL. JCL is provided in the job HFMIOXPL to help you to compile and link your exit in PL/I. Refer to the instructions in the jobs for changes you need to make. HFMIOXCO and HFMIOXPL are distributed in HFM.SHFMSAM1.

Chapter 12. Customizing Z Data Tools to use a scrambling exit

You can use Z Data Tools to create test data based upon production data with the option to change the values of certain fields that may contain sensitive information. This facility is known as field scrambling.

You can also write your own scrambling exit, to perform scrambling operations that are not provided in Z Data Tools.

To specify a scrambling exit, select **Scramble Type** 4 ("Exit") from the Field Attributes panel of the template editing process, where you have specified the fields to be scrambled. Z Data Tools displays the Scramble Exit Specification panel where you can provide the exit name, a constant to be passed to the exit, and specify whether numeric fields are to be formatted before the value is passed to the exit. The exit can be an assembler or LE-enabled program. The *Z Data Tools User's Guide and Reference* describes how to use the scrambling exit.

Exit protocol

The scrambling exit is called during a copy operation with an input and output template, and where the output field has been defined with **Scramble Type** 4 ("Exit") and mapped to an input field. The exit is called in these situations:

Initialization

The first call to the exit for each field is an initialization call that causes the exit to perform set-up procedures. A 1024-byte user area is provided, in addition to the user constant for the exit to use.

Scramble

This call is repeated for each instance of the field being copied. On this call, the exit is expected to provide Z Data Tools with the scrambled output value for the field, or a directive for Z Data Tools to scramble the field.

Termination

The last call for each field is a termination call that causes the exit to perform clean-up procedures.

Writing your exit

Z Data Tools does not supply a default scrambling exit. If you plan to use a scrambling exit you must provide one or more for your installation.

Any exit you provide must be in the form of a load module, in any load library available to Z Data Tools, either by a STEPLIB DD statement, or in LINKLIST, or LPALIST. If Z Data Tools attempts to load an exit and is unable to find it, an error message is displayed.

You can write your own exit in any high level language, for example, COBOL, PL/I, or High Level Assembler. Sample exits for COBOL, PL/I, and HLASM are provided in HFM.SHFMSAM1. They are:

HFMSCXEA

Sample HLASM exit

HFMSCXEC

Sample COBOL exit

HFMSCXEP

Sample PL/I exit

These samples perform a simple scrambling algorithm of changing all uppercase characters to 'A', all lowercase characters to 'a', and all numbers to '1'. They are designed purely to demonstrate the capabilities of an exit. Refer to the prolog in each sample for more information about coding your exit.

Copybooks providing the scrambling exit control blocks for COBOL, PL/I, and HLASM are distributed in the Z Data Tools macro library, HFM.SHFMMAC1. They are:

HFMSCXCA

Control block for HLASM programs

HFMSCXCC

Control block for COBOL programs

HFMSCXCP

Control block for PL/I programs

These copybooks are described in Exit control block description on page 139. See Using the scrambling exit control block on page 145 for information on how to write your exit using the copybooks.

Note:

1. The Z Data Tools I/O exit does not support 24-bit addressing mode.
2. Z Data Tools supports an exit written in any supported release of COBOL and PL/I.
3. Z Data Tools provides information concerning the input and output field formats as described by the exit control block. See Table 21: Exit control block on page 139

Exit control block description

Table 21. Exit control block

Description	Data type	Set by:	Value
Eye catcher	CL8	ZDT initialization	'UFLDXCB'
Version	F	ZDT initialization	**1** Initial value
Function Code	F	ZDT for every call to the exit	**1** **Initialize** - once per field **2** **Scramble** - each time the field is copied

Table 21. Exit control block (continued)

Description	Data type	Set by:	Value
			3 **Terminate** - once per field
Input Field Address	A	ZDT, for every scramble call	Field area address of the input field value
Input Field Length	F	ZDT, for every scramble call	Field length of the input field value area
Input Field Subscript Address	A	ZDT, for every scramble call	For dimensioned fields this contains the address of an array of fullword subscripts values
Input Field Dimensions	F	ZDT, for every scramble call	Number of subscripts for dimensioned fields
Input Field Name Address	A	ZDT, for every scramble call	Address of template or copybook field name
Input Field Name Length	H	ZDT, for every scramble call	Template or copybook field name length
Input field type	H	ZDT for every scramble call	**1** Alphanumeric (includes alphabetic, and group items). **2** Binary. **3** Packed decimal. **4** Zoned decimal. **5** Numeric edit. **6** External floating point. **7** Floating point. **8** Decimal floating point.

Table 21. Exit control block (continued)

Description	Data type	Set by:	Value
			9 Date. **10** Time. **11** Timestamp. **12** DBCS.
Input nullable	CL1	ZDT for every scramble call	(Y/N). Y is for Db2® nullable columns, otherwise N.
Input varying	CL1	ZDT for every scramble call	(Y/N). Y is for varying field or column, otherwise N.
Input varying	CL1	ZDT for every scramble call	(Y/N). Y is for varying null terminated strings, otherwise N.
Input signed	CL1	ZDT for every scramble call	For COBOL: **0** No sign clause. **1** Leading sign. **2** Leading sign separate. **3** Trailing. **4** Trailing sign separate. For PL/I: **Y** Signed. **N** Not signed.
Input Db2® null indicator	H	ZDT for every scramble call	Db2® null indicator. Only valid for Db2® nullable columns.

Table 21. Exit control block (continued)

Description	Data type	Set by:	Value
Input varying field length	H	ZDT for every scramble call	The length value for a varying field or column.
Input field precision	H	ZDT for every scramble call	This is the precision. That is, the number of significant digits of a binary, zoned or packed decimal field.
Input field scale	H	ZDT for every scramble call	The scale. That is, the number of digits to the right of the decimal point (it indicates where the decimal point would be positioned).
Output field address	A	ZDT for every scramble call	Field area address of the output field value.
Output field subscript address	A	ZDT for every scramble call	For dimensioned fields, this contains the address of an array of fullword subscript values.
Output field dimensions	F	ZDT for every scramble call	Number of subscripts for dimensioned fields.
Output field name address	A	ZDT for every scramble call	Address of Template or copybook field name.
Output field name length	H	ZDT for every scramble call	Template or copybook field name length.
Output field type	H	ZDT for every scramble call	**1** Alphanumeric (includes alphabetic, and group items). **2** Binary. **3** Packed decimal. **4** Zoned decimal. **5** Numeric edit. **6** External floating point.

Table 21. Exit control block (continued)

Description	Data type	Set by:	Value
			7 Floating point. **8** Decimal floating point. **9** Date. **10** Time. **11** Timestamp. **12** DBCS.
Output nullable	CL1	ZDT for every scramble call	(Y or N). Y is for Db2® nullable columns, otherwise N.
Output varying	CL1	ZDT for every scramble call	(Y or N). Y is for varying field or column, otherwise N.
Output varying	CL1	ZDT for every scramble call	(Y or N). Y is for varying null terminated strings, otherwise N.
Output signed	CL1	ZDT for every scramble call	For COBOL: **0** No sign clause. **1** Leading sign. **2** Leading sign separate. **3** Trailing. **4** Trailing sign separate. For PL/I: **Y** Signed.

Table 21. Exit control block (continued)

Description	Data type	Set by:	Value
			N Not signed.
Output field precision	H	ZDT for every scramble call	The precision. That is, the number of significant digits of a binary, zoned, or packed decimal field.
Output field scale	H	ZDT for every scramble call	The scale. That is, the number of digits to the right of the decimal point (it indicates where the decimal point would be positioned).
Return Code	F	ZDT sets to zero, exit sets on return	**0** Successful. For Scramble option, the output value is used for the output field. **1** Random. Request for ZDT to perform random scrambling. Only valid for Scramble option. **2** Repeat. Request for ZDT to perform repeatable scrambling. Only valid for Scramble option. **3** Translate. Request for ZDT to perform translatable scrambling. This requires that a data set name and column in and column out values have been provided on the field attributes panel for this field. **8** Unsupported input field type. **10** Invalid input value. **16** Severe Error terminate.
Output field length	F	ZDT for every scramble call. Exit to set on return for formatted fields.	This value should be provided when returning a formatted numeric. The output area length is 80 and the returned length must be less than or equal to this.

Table 21. Exit control block (continued)

Description	Data type	Set by:	Value
Output null indicator	H	ZDT for every scramble call. Exit to set when returning for nullable field.	The exit can change the null indicator for nullable fields by setting this value.
Output varying length	H	ZDT for every scramble call. Exit can set when returning varying field.	The exit can set this value for a varying field on return. For any other type of field, the value is ignored.
User constant	CL80	ZDT for every scramble call	This contains the constant defined during scramble exit specification in template edit.
User Area	CL1024	Exit to set the values. Same area is referenced for each call.	This is a user area for the exit to use and is available for every call to the exit.

Using the scrambling exit control block

Function codes

The function code determines the action of the scramble exit. The initialization and termination calls are once for each field. The scramble calls are once for each instance. If a field is an array, then the exit is called for each element of the array.

Return codes

For initialization and termination, the return code is either zero for success, or non-zero indicating a severe error has occurred and Z Data Tools is to be terminated.

For scramble, the exit can return codes as described in the control block.

0

Instructs Z Data Tools to update the output field with the value return. If the value passed to the exit was formatted numeric, then Z Data Tools treats the value returned from the exit as a formatted numeric which is converted to the internal format before updating the field.

1

Instructs Z Data Tools to perform random scrambling on the field. Any scrambling value or range specifications are used. The output field value is ignored.

2

Instructs Z Data Tools to perform repeatable scrambling on the field. Any scrambling value or range specifications are used. The output field value is ignored.

3

Instructs Z Data Tools to perform translate scrambling on the field. For this return code to be valid, you must have specified a value data set (**Dsn**), column in and column out values (**In** and **Out**), as well as selecting the **Value** option. If you have not provided these options for the field, then the copy process is terminated with an error.

8

The scramble exit being called does not support the input field type. The process is terminated with an error message.

10

The scramble exit being called has found an invalid input field value. The process is terminated with an error message.

16

Severe error - terminate.

Note: Any other return code is interpreted as a severe error that terminates processing.

Return output field values

When a return code of zero is returned with the function code for scramble, the output field is updated with the output value provided by the scramble exit. The output value should be valid for the field being updated.

Installing your exit

You can provide more than one user scrambling exit. For example, you might want to provide a different exit for each user, or different exits for use with different data sets. Also, you could define a different exit for each field or column you have selected for scrambling. Thus one copy, export or import operation could load multiple exit programs to satisfy the scrambling requirements of different fields or columns. Individual users can also specify different exits.

To provide an exit in HLASM, COBOL or PL/I:

1. Code your own exit in your own source library. See the information in Exit control block description on page 139 and Using the scrambling exit control block on page 145. You can use HFMSCXEA as a HLASM example, HFMSCXEC as a COBOL example, or HFMSCXEP as a PL/I example. These samples are distributed in HFM.SHFMSAM1.
2. Assemble or compile and link your exit into a load library available to Z Data Tools, for example, HFM.SHFMMOD1.

Sample jobs are provided to help you do this. These are:

HFMSCXHA

Assemble and link a HLASM exit

HFMSCXCO

Compile and link a COBOL exit

HFMSCXPL

Compile and link a PL/I exit

See the instructions in the jobs for changes you need to make. These jobs are distributed in HFM.SHFMSAM1.

Chapter 13. Verifying the customization of Z Data Tools

After you have completed the initial installation and customization of Z Data Tools Base function, you can perform the following steps to verify your customization.

You might need to refer to the *Z Data Tools User's Guide and Reference for IMS Data*.

1.

   Edit and submit the batch installation verification program (IVP) job, HFMVERIF, as described in the Z Data Tools Program Directory.

   Check the results to ensure the job ran successfully.

2. Log on to a TSO user ID that is enabled to access and use Z Data Tools.
3. Start Z Data Tools:
   - If you have added an option for Z Data Tools to your ISPF Primary Options menu (see Adding Z Data Tools to the ISPF menu on page 36), type the option value you have assigned to Z Data Tools and press Enter. For example, if you have assigned Z to Z Data Tools, type Z and press Enter.
   - If you defined Z Data Tools to the ISPF command table (see Defining Z Data Tools in an ISPF command table on page 37), verify that Z Data Tools can be started by entering the command HFM on any ISPF command line.

   The Z Data Tools Primary Option Menu is displayed.

   Note: The first time you use Z Data Tools, a Copyright panel appears. After reading the panel text, press the Cancel key (PF12). In subsequent Z Data Tools sessions, this panel will not automatically appear.

4.

   Enter VER on the command line to display the release level and PTF level of Z Data Tools. A panel is displayed similar to this:

```
HCL Z Data Tools
(not APF authorized)
Service Levels of installed components
             Base       IMS        Db2        CICS
English      -NONE-     -NONE-     -NONE-     -NONE-
```

   Note:

   a. This will always show Z Data Tools as `not APF authorized`, even if you have made Z Data Tools APF-authorized, as Z Data Tools cannot run APF-authorized under ISPF.
   b. When you first install Z Data Tools, `-NONE-` will be shown under each component that you have installed. Subsequently, when you have applied service to Z Data Tools, a PTF number will be shown,



indicating the PTF level of each component you have installed. If you have not installed a component, that component will not be shown at all.

If you have installed the Japanese language component, another line will be displayed indicating the service level of that component.

5. Press the Exit key (PF3) to end the Z Data Tools session and return to your ISPF menu.

Several other sample batch jobs are distributed in HFM.SHFMSAM1 to help you verify the installation of Z Data Tools and demonstrate its functions. They are:

Sample job	Function
HFMSAMC1	Copy all records from one file to another, modifying a record in the process.
HFMSAMF1	Find JCL exec statements that run a particular program.
HFMSAMF2	Find JCL EXEC statements that run a particular program, and rename the program in each member in place.
HFMSAMP1	Print the JCL members that execute a particular program.
HFMSAMP2	Print selected records and drop all other records, formatting the output from a template.
HFMSAMRS	Perform a simple record selection, and copy the selected records to a file.
HFMSAMSP	Split selected records to two output files.
HFMSAMU1	Make selective updates in a member of a data set, in place, using function DSEB and REXX.
HFMSAMU2	Make selective updates in a member of a data set, in place, using function DSU and FASTREXX.

You will have to make changes to these jobs before you can run them. See the sample jobs for information about the changes you need to make.

You can now complete the installation of Z Data Tools by performing the ACCEPT processing. The steps involved are described in the Z Data Tools Program Directory.

Verifying the Japanese component

After you have completed all the necessary steps involved in the initial installation and customization of the Z Data Tools Japanese component, you need to perform the following steps to verify your customization, before completing the installation process. You might need to refer to the *Z Data Tools User's Guide and Reference for IMS Data*.

Note: The Z Data Tools Japanese component requires the Z Data Tools Base function product to have already been installed.

1. Log on to a TSO user ID that is enabled to access and use Z Data Tools. Ensure the ISPF terminal type is set to 3277KN or 3278KN in ISPF option 0.
2. Start Z Data Tools:
   - If you have added an option for Z Data Tools to your ISPF Primary Options menu (see Adding Z Data Tools to the ISPF menu on page 36), enter the option value you have assigned to Z Data Tools and press **Enter**. For example, if you have assigned 'F' to Z Data Tools, enter 'F' and press **Enter**.
   - If you defined Z Data Tools to the ISPF command table (see Defining Z Data Tools in an ISPF command table on page 37), verify that Z Data Tools can be started by entering the command 'FM' on any ISPF command line.

   The English or Japanese Z Data Tools Primary Option Menu is displayed. depending on the default defined.

   Note: The first time you use Z Data Tools, a Copyright panel appears. After reading the panel text, press the Cancel key (PF12). In subsequent Z Data Tools sessions, this panel will not automatically appear.

3. If the English menu appears:
   - Check the ISPPLIB concatenation and ensure that the Japanese panel library (HFM.SFMNPJPN)) is allocated in front of the English panel library (HFM.SFMNPENU).
   - Check the ISPMLIB concatenation and ensure that the Japanese message library (HFM.SFMNMJPN) is allocated in front of the English message library (HFM.SFMNMENU).
4. Enter `ver` on the command line to display the release level and PTF level of Z Data Tools. If you are verifying the Japanese component, a panel is displayed similar to this:

```
HCL Z Data Tools
                              Version 17 Release
                                    1 Modification 0

(not APF authorized)

Service Levels of installed components

             Base       IMS        DB2        CICS
English      -NONE-     -NONE-     -NONE-     -NONE
Japanese     -NONE-     -NONE-     -NONE-     -NONE-
```

   Check that the information displayed conforms with the level of Z Data Tools that you are verifying. If you have not installed a component in Japanese (for example, you have not installed the ZDT/IMS Japanese component), that component will not be shown at all.

5. Press the Exit key (PF3) to end the Z Data Tools session and return to your ISPF menu.

You can now complete the installation of Z Data Tools by performing the ACCEPT processing. The steps involved are described in the Z Data Tools Program Directory.

Part II. Customizing Z Data Tools Db2® Component

Chapter 14. Preparing to customize ZDT/Db2

Before you can install and customize ZDT/Db2 you must have installed the Z Data Tools Base function. These topics assume you have installed ZDT/Db2 into the same target and distribution libraries as Z Data Tools Base function. You could encounter problems using Z Data Tools if you do not install ZDT/Db2 in the same libraries as the Z Data Tools Base function.

Furthermore, before you can use ZDT/Db2 you must customize ZDT/Db2 and the operating environment. You may also need to customize Z Data Tools Base function. Customizing Z Data Tools on page 22 describes the customization of Z Data Tools Base function.

Attention: Granting access to Db2® resources is a Db2® task, not a Z Data Tools task. Consult your site's Db2® administrator or security administrator before proceeding. If, after careful consideration of your site's security requirements and the requirement for ZDT/Db2 to access the Db2® catalog, you are unable to provide ZDT/Db2 users with the appropriate level of access, do not attempt to install and use ZDT/Db2.

To function correctly, ZDT/Db2 requires access to various Db2® catalog tables. The configuration of Db2® security/authorization is described in the *Db2® Administration Guide*. Only general information is provided on how you might allow ZDT/Db2 users SELECT access to the Db2® catalog tables. Consider your site's security requirements and how Db2® security is implemented and administered when selecting the most appropriate method for your installation. For example, many sites have a standard that SELECT access on some or all of the Db2® catalog tables must not be granted to PUBLIC.

There are three customization tasks you **must** perform for ZDT/Db2, otherwise you will not be able to use ZDT/Db2.

- Granting access to the Db2® catalogs, described in Granting access to the Db2 catalog (required) on page 161.
- Binding the ZDT/Db2 packages and plans for Db2®, described in Binding Db2 (required) on page 172.
- Customizing the **Db2 Subsystem Selection** panel, described in Defining all Db2 systems that ZDT/Db2 will access in HFM2POPT (required) on page 174.

Table 22: Summary of steps for customizing ZDT/Db2 and the operating environment on page 153 lists the customization tasks that you can perform for ZDT/Db2. Read the referenced sections to see if you need to perform the customization described.

Checklist for installing and customizing ZDT/Db2

Table 22. Summary of steps for customizing ZDT/Db2 and the operating environment

	Description
__ 1	Concatenate ZDT/Db2 libraries to the LINKLIST. See Concatenating libraries to the LINKLIST on page 155.
__ 2	Modify the TSO logon procedure. See Modifying the TSO logon procedure on page 155.
__ 3	Customize to improve ZDT/Db2 performance. See Improving ZDT/Db2 performance on page 156.

Table 22. Summary of steps for customizing ZDT/Db2 and the operating environment (continued)

	Description
__ 4	Add ZDT/Db2 to the ISPF menu. See Adding ZDT/Db2 to the ISPF menu on page 159.
__ 5	Define ZDT/Db2 in an ISPF command table. See Defining ZDT/Db2 in an ISPF command table on page 159.
__ 6	Customize the ZDT/Db2 Primary Option Menu. See Customizing the ZDT/Db2 Primary Option Menu on page 160.
__ 7	Grant access to the Db2® catalogs - **required.** See Granting access to the Db2 catalog (required) on page 161.
__ 8	Add ZDT/Db2 to the Db2® Admin Launchpad See Adding ZDT/Db2 to the Db2 Administration Launchpad on page 165.
__ 9	Add commands to start ZDT/Db2 functions from Db2® Admin Tool See Adding commands to start ZDT/Db2 functions from the Db2 Administration Tool on page 166.
__ 10	Bind Db2® - **required**. See Binding Db2 (required) on page 172.
__ 11	Set up to run multiple versions of ZDT/Db2. See Running multiple versions of ZDT/Db2 on page 173.
__ 12	Identify all the Db2® systems that ZDT/Db2 will access - define these in the HFM2POPT module - **required**. See Defining all Db2 systems that ZDT/Db2 will access in HFM2POPT (required) on page 174.
__ 13	Customize ZDT/Db2 options. See Customizing the ZDT/Db2 options on page 178.
__ 14	Change the default options. See Changing the default options on page 180.
__ 15	Change the batch JCL skeletons. See Changing the JCL skeletons for batch mode on page 181.
__ 16	Customize to protect update functions. See Customizing to protect update functions in ZDT/Db2 on page 183.
__ 17	Decide how to customize the ZDT/Db2 audit facility. See Alternatives for controlling ZDT/Db2 auditing on page 157.
__ 18	Customize ZDT/Db2 for national languages. See Customizing ZDT/Db2 for national languages on page 211.

Alternatives for making ZDT/Db2 available

You can make ZDT/Db2 available to users either by concatenating HFM.SHFMMOD1 to your LINKLIST or adding it to the STEPLIB DD statement in your TSO logon procedure.

To make ZDT/Db2 readily available from ISPF, configure your ISPF environment as described in Customizing the operating environment for Z Data Tools on page 30.

Concatenating libraries to the LINKLIST

To make ZDT/Db2 commonly available, add the HFM.SHFMMOD1 library to your concatenated LINKLIST. If you did not do this for Z Data Tools Base function, add this library to either your LNKLST*xx* or PROG*xx* member in SYS1.PARMLIB.

You can also choose to add your Db2® load and run libraries to your LINKLIST. If you do this, you might not need to specify these libraries in your HFM2SSDM macros. See DB2LLIB on page 431 and DB2RLIB on page 432.

Modifying the TSO logon procedure

If you made Z Data Tools Base function available to TSO, then you do not need to make any further changes to TSO for ZDT/Db2. Otherwise, add the Z Data Tools libraries to your TSO logon procedure, as described in Modifying the TSO logon procedure on page 27.

Using LIBDEFs to allocate the ZDT/Db2 Libraries

If you choose to use a CLIST or a REXX exec to dynamically define the required ZDT/Db2 libraries using LIBDEFs, also copy the REXX exec HFM2RESS from HFM.SHFMEXEC to a library that is defined in the SYSPROC or SYSEXEC concatenation at the time ISPF is started. If possible, this library should be the same library that contains the EXEC or CLIST that you initially use to start ZDT/Db2.



Note: Do not place HFM2RESS in a library defined by the TSO ALTLIB command. HFM2RESS is used during the invocation of a second (or subsequent) ZDT/Db2 edit session (related edit). The invocation of the related edit session will fail if this exec cannot be accessed at related edit invocation time. Libraries defined by the TSO ALTLIB command are not propagated to the second ISPF session started for the related edit, so it is not sufficient to place this exec in a library defined by the ALTLIB command.

Ensuring the Db2® libraries are available to ZDT/Db2

ZDT/Db2 requires access to the same Db2® libraries as the DB2I online utility.

You can allocate the Db2® libraries required by ZDT/Db2 using one of the following methods:

- You can include the Db2® libraries in your TSO logon procedure; or in a CLIST or EXEC run as part of the TSO logon procedure; or in a CLIST or EXEC executed prior to the initialization of ZDT/Db2. This method is not appropriate if ZDT/Db2 will be connecting to Db2® systems at different version/maintenance levels. If you use this method you

might not need to specify the Db2® libraries in the HFM2SSDM macros, but you will still need to review the contents of the HFM2POPT options module. See Changing the default options on page 180.

- Specify the Db2® libraries in the HFM2SSDM macro entry for the Db2® system. This is the preferred method and allows ZDT/Db2 to connect to Db2® systems at different version/maintenance levels. The Db2® libraries are allocated as part of the ZDT/Db2 connect process, and de-allocated at the end of the ZDT/Db2 session, or when ZDT/Db2 connects to a different Db2® system. See HFM2SSDM on page 426.

Planning for running ZDT/Db2 with APF-authorization

If you intend to use SMF for recording audit trail information for ZDT/Db2, you must ensure that the Z Data Tools load module, HFMSMF, is APF-authorized.

If you have not made Z Data Tools Base function APF-authorized, use one of the following methods to ensure that HFMSMF is authorized.

- Add HFM.SHFMMOD1 to your list of authorized libraries.
- Copy HFMSMF into an authorized library.

Note that library HFM.SHFMMODA contains Z Data Tools modules that may need to run authorized.

Improving ZDT/Db2 performance

ZDT/Db2 performance can be improved by creating additional indexes on various Db2® catalog tables. A sample job HFM2CIX is provided in HFM.SHFMSAM1 to help you do this. This job contains additional indexes that are useful in improving performance when connected to Db2® version 12 or later systems.

Take a copy of the job and make the changes described in the job. You should review the indexes to be created against the current indexes defined for the Db2® catalog tables. It is possible that some of the indexes to be defined already exist, having been created as part of the installation of other products, or during the installation of previous versions of Z Data Tools Db2®.

After the indexes have been successfully created, run the RUNSTATS utility against database DSNDB06. See your site Db2® System Administrator for additional information on how to do this.

The Z Data Tools Db2® editor can operate in two modes: "normal" and "large". The mode of operation is determined by the value entered in the **row count** field for those ZDT/Db2 functions that use the ZDT/Db2 editor to display data. The characteristics of the two editor modes are documented in the *Z Data Tools User's Guide and Reference for Db2® Data*.

The use of "large" editor mode may have negative Db2® performance implications. When the ZDT/Db2 editor operates in "large" mode, it uses a Db2® scrollable cursor for access to Db2® data. This minimizes the memory usage in the ZDT/Db2 user's TSO address space, but might require Db2® to build a temporary copy of the entire result table in a Db2® temporary database. For large tables, this can lead to SQLCODE-904 (unavailable resource) on table spaces defined within the Db2® temporary database. For these reasons, providing access to the ZDT/Db2 editor in production Db2® environments, where there are large Db2® tables, should be carefully considered.

The product installer can disable the use of "large" editor mode, by Db2® subsystem. This is achieved by setting the HFM2SSDM macro parameter EDIT_MAX_ROWS to a non-zero value. See EDIT_MAX_ROWS on page 434. The HFM2SSDM macro is discussed further in Defining all Db2 systems that ZDT/Db2 will access in HFM2POPT (required) on page 174.

Alternatives for controlling ZDT/Db2 auditing

ZDT/Db2 auditing is an optional facility. ZDT/Db2 works if auditing is not implemented. You should consider:

- Whether user access to Db2® data and other resources using Z Data Tools Db2® component requires auditing.
- The information that Z Data Tools audit log records can provide.
- The information that Z Data Tools audit log records cannot provide, and possible alternatives to obtaining that information.
- If you do decide to use Z Data Tools auditing, how you will handle any issues associated with large audit log data sets, or additional SMF records.
- How you will use the information provided by Z Data Tools audit log records.

If your site requires a record of a user's read access to Db2® data, an external security product such as RACF® can be configured to log access by some or all users, and may be a better alternative. Db2® also provides an audit facility which may be an alternative to Z Data Tools auditing.

Z Data Tools audit of read access to Db2® data does not write audit log records for every row processed, rather the name of the Db2® object and how many rows were processed are written to the audit log. Z Data Tools audit of changes to Db2® data typically writes two log records, a before and after image of the row that was changed. If you intend to log update changes to Db2® tables that are subject to heavy update activity, consider the performance impact of writing many audit log records as well the size of any audit log data sets that might be produced.

You have two choices with respect to auditing of ZDT/Db2 audit activities. These are:

1. Use HFM2POPT controlled auditing.

   The following points summarize the facilities available with HFM2POPT controlled auditing:

   - Different audit settings can be specified for each Db2® system that ZDT/Db2 might access
   - Auditing occurs for key ZDT/Db2 functions only (edit and copy). Other ZDT/Db2 functions are not subject to audit (for example print, import, export, and issuing Db2® commands).
   - The audit settings for any Db2® system apply equally to all ZDT/Db2 users connected to that Db2® system.
   - The audit settings for any Db2® system apply equally to all Db2® objects within that Db2® system.
   - The "Create audit trail" option is visible to the user for those ZDT/Db2 functions where auditing might occur.
   - You can specify auditing to the user's audit log data set, to the user's audit log data set with automatic (mandatory) printing of the audit log at the completion of the session, or to SMF.
2. Use SAF-rule controlled auditing. This relies on various SAF FACILITY and XFACILIT resource rules, which you define with an external security product, such as RACF® (or equivalent product).

   The following points summarize the facilities available with SAF-rule controlled auditing:

- Different audit settings can be specified for each Db2® system that ZDT/Db2 might access.
- Auditing can be (optionally) specified for all ZDT/Db2 functions, however audit records are not written for the SQL statements used by ZDT/Db2 to build templates, or for other internal purposes.
- Different auditing requirements can be specified for different TSO user IDs.
- Different auditing requirements can be specified for access to different Db2® objects, or to different types of SQL statements, or when issuing Db2® commands.
- You can provide ZDT/Db2 users with a "Create audit trail" option for some ZDT/Db2 functions. This is also SAF-rule controlled. The presence of the "Create audit trail" option does not guarantee that the user can switch off auditing, since this depends on the level of access the user has to the appropriate SAF resource names. When a user has access to the "Create audit trail" option, they can always turn on auditing, even if the relevant SAF resource rules do not require auditing.
- You can specify auditing to the user's audit log data set, to the user's audit log data set with automatic (mandatory) printing of the audit log at the completion of the session, or to SMF. Dual logging (to the user's audit log data set and to SMF) can also be specified.

Some other points to consider are:

- Auditing to the user's audit log data set can result in large numbers of audit log data sets, and this may have disk space implications. You may need to consider implementing automatic purging or archiving of audit log data sets.
- Auditing to SMF (only) requires additional set-up, but provides a more reliable and secure environment for capturing audit information than audit logging to the user's audit log data set.
- With SAF-rule controlled auditing in effect to SMF only, a Z Data Tools/Db2 user can determine that SAF-rule controlled auditing is in effect, but there is no indication to the user which activities are actually being audited.
- With SAF-rule controlled auditing in effect, failure to write records to the SMF audit log - when this is required - will result in the termination of the current ZDT/Db2 function.

If you implement SAF-rule controlled auditing you need to decide how Z Data Tools auditing will be enabled. This is described in more detail in Customizing the audit facility for ZDT/Db2 on page 189. There are two alternatives. One requires an enabling SAF rule and the presence of a member in SYS1.PARMLIB, the other requires an enabling SAF rule but has no requirement for a member in SYS1.PARMLIB. The use of a member in SYS1.PARMLIB provides additional facilities compared with the alternative that does not require the use of SYS1.PARMLIB. The additional facilities are documented in Z Data Tools options specified in PARMLIB members on page 515.

When you have determined the appropriate type of auditing for your installation, follow the instructions in Customizing the audit facility for ZDT/Db2 on page 189.

Chapter 15. Customizing the operating environment for ZDT/Db2

This chapter describes how to customize the operating environment for ZDT/Db2. You do this after you have installed ZDT/Db2.

Modifying the ISPF environment

To make it easy to start ZDT/Db2 under ISPF, configure your ISPF environment as described in the following sections.

Adding ZDT/Db2 to the ISPF menu

To add ZDT/Db2 to your ISPF Primary Option Menu panel (ISR@PRIM), insert the additional lines (◄ **New**) as shown in Figure 14: on page 159. You could add ZDT/Db2 in your Primary Option Menu after the Z Data Tools Base function.

Figure 14.

```
⋮
)BODY  CMD(ZCMD)
⋮
 9  IBM Products  IBM program development products
 10 SCLM          SW Configuration Library Manager
 11 Workplace     ISPF Object/Action Workplace
 Z  Z Data Tools  Z Data Tools
 ZD ZDT/Db2/       ZDT/Db2             ◄ New
⋮
)PROC
⋮
 &ZSEL = TRANS( TRUNC (&ZCMD,'.')
⋮
  9,'PANEL(ISRDIIS) ADDPOP'
 10,'PGM(ISRSCLM) SCRNAME(SCLM) NOCHECK'
 11,'PGM(ISRUDA) PARM(ISRWORK) SCRNAME(WORK)'
  Z,'PANEL(HFMSTASK) SCRNAME(ZDTOOLS) NEWAPPL(HFM)' /* Z Data Tools */
 ZD,'PANEL(HFM2ST00) SCRNAME(HFMDB2) NEWAPPL(HFM2)' /* ZDT/Db2       */ ◄ New

⋮
```

To invoke ZDT/Db2 with LIBDEFs, see Example 1. Invoking Z Data Tools, ZDT/IMS, and ZDT/Db2 primary options from a selection panel on page 35.

For information about configuring your ISPF Primary Option Menu panel, see z/OS ISPF Planning and Customizing.

Defining ZDT/Db2 in an ISPF command table

ISPF supports four different command tables where you can define an ISPF command to invoke ZDT/Db2:

- Application command table
- User command table
- Site command table
- System command table

You can use the ISPF Command Table Utility (ISPF option 3.9) to create or change a command table that is not currently in use (the system command table, ISPCMDS, is always in use). When you add a command for ZDT/Db2 to one of these command tables, you can invoke ZDT/Db2 from any ISPF panel without prefixing the command with TSO.

Add the following entry to a command table to enable ZDT/Db2 to be run from any ISPF panel by entering ZD on the command line.

Verb	ZD
Action	SELECT PANEL(HFM2ST00) OPT(&ZPARM) SCRNAME(HFMDB2) SUSPEND NEWAPPL(HFM2)
Description	Z Data Tools Db2® Component

To invoke ZDT/Db2 with LIBDEFs, see #unique_68_Connect_42_libdef-invoke-ex2.

For information about ISPF command tables, see z/OS ISPF Planning and Customizing and *z/OS ISPF User's Guide Vol II*.

Customizing the ZDT/Db2 Primary Option Menu

This customization step is optional. Review this information to determine if you need to perform this step.

The ZDT/Db2 Primary Option Menu panel (HFM2ST00) provides an option to select Db2® Interactive. If the DB2I option is selected, the exec HFM2DB2I is called.

If you have customized the method used to start DB2I, for example, if your site displays a Db2® authorization panel before starting DB2I, you may need to modify the HFM2ST00 panel or the HFM2DB2I exec.

Owing to the wide range of possible customizations the information provided here is not specific to any particular method.

Panel HFM2ST00, as supplied, includes the following:

```
&ZSEL = TRANS(TRUNC(&ZCMD,'.')
  0,'PANEL(&HFM2STSC)'
  1,'CMD(HFMDB2 D2B) MODE(FSCR)'
  2,'CMD(HFMDB2 D2E) MODE(FSCR)'
  3,'PANEL(HFM2STUT)'
  4,'PANEL(HFM2STSS)'
  5,'CMD(HFM2DB2I SSID(&HFM2SSIS)) MODE(FSCR)'    ◄ DB2I
  X,'EXIT'
  ' ',' '
```

DB2I indicates where HFM2DB2I is called when Option 5, DB2I, is selected.

The exec HFM2DB2I, as supplied, allocates a data set to DDNAME DSNETBLS, before invoking DB2I using the following ISPF SELECT statement:

```
"SELECT CMD(DSNECPRI SSID("ssid")) NEWAPPL(DSNE) PASSLIB MODE(FSCR)"
```

You may need to customize either the panel HFM2ST00, or the exec HFM2DB2I, if your site has a customized exec or panels called prior to starting DB2I.

Note:



1. ZDT/Db2 provides two methods for allocating the Z Data Tools libraries. See Modifying the TSO logon procedure on page 155 for further information. You must ensure that any processing in your site's Db2® invocation procedure does not de-allocate the ZDT/Db2 libraries.
2. If your site's Db2® customization uses LIBDEF statements to allocate the required Db2® libraries, ZDT/Db2 allocates the required Db2® libraries (as defined in the HFM2SSDM macro for the connected Db2® system) at connect time: that is, when ZDT/Db2 is first started, or whenever the Db2® SSID is changed. ZDT/Db2 is connected to the Db2® system shown against "Db2® SSID" on the Primary Option Menu panel, HFM2ST00. This means that it is unnecessary for the DB2I invocation procedure to allocate the Db2® libraries prior to starting DB2I. You can determine the current LIBDEF allocation by starting ZDT/Db2 and issuing the ISPLIBD command on the command line.
3. HFM2ST00 is distributed in HFM.SHFMPENU. HFM2DB2I is distributed in HFM.SHFMEXEC. If you decide to modify these or any other ZDT/Db2 parts to complete your customization for invoking DB2I, create a usermod to do this, to keep track of the changes to SMP/E controlled parts. For information about modifying panels, refer to the z/OS ISPF Dialog Developer's Guide. For information about modifying execs, refer to the z/OS ISPF Services Guide.

Granting access to the Db2® catalog (required)

ZDT/Db2 is a *"full-function"* Db2® application, intended to provide access to every Db2® catalog table, including those catalog tables that may contain sensitive information. ZDT/Db2 uses dynamic SQL, issued against the Db2® catalog, as part of its processing. To make ZDT/Db2 available and to ensure correct and optimum operation, you must ensure that each ZDT/Db2 user has SELECT access against the Db2® catalog tables. This is a non-negotiable requirement.

ZDT/Db2 includes an option (Db2® Privileges utility) that enables information in the various *AUTH Db2® catalog tables to be viewed, and possibly changed. If your installation restricts access to the *AUTH tables, you can disable the Db2® Privileges utility entirely, and remove the need to grant SELECT access on the *AUTH tables at install time. You can disable the ZDT/Db2 Privileges utility for some Db2® systems, but not for others as required.

To disable access to the Db2® Privileges utility you need to:

- Code `AUTH_ACCESS=N` in the HFM2SSDM macro entry for each Db2® system where no SELECT access to the *AUTH tables is allowed. See the parameter description AUTH_ACCESS on page 428 for more information.
- Remove or comment out references to the *AUTH tables in the sample job used to grant SELECT access on the Db2® catalog tables to ZDT/Db2 users. See Sample jobs to grant SELECT access on the Db2 catalog tables on page 163 for additional information.

You can also further restrict access to some Db2® catalog tables - see Sample jobs to grant SELECT access on the Db2 catalog tables on page 163 for additional information. ZDT/Db2 is designed to tolerate incomplete access to most Db2® catalog tables, although the functionality of the product is reduced when access to some Db2® catalog tables is reduced. Restricting access to certain key Db2® catalog tables renders the product inoperative - for an indicative list of the key tables and columns see the *"minimal subset"* sample members described below.

Db2® authorization configuration can only be achieved using Db2®, or an external security server (or both). Since these are external products, only general guidance is provided here on the authorization and security issues associated with the use of ZDT/Db2. You must determine the best approach to providing the required level of access for ZDT/Db2 users, based on your installation's unique requirements. You must ensure SELECT access against the Db2® catalog tables is given for all Db2® systems that are accessible to ZDT/Db2 users.

ZDT/Db2 requires the use of DYNAMICRULES(RUN) when the ZDT/Db2 plan is bound. See Binding Db2 (required) on page 172 for further information. The effect of DYNAMICRULES(RUN) behavior is that ZDT/Db2 uses the *run behavior* for dynamic SQL statements, as summarized here:

- Db2® uses the authid of the application process and the SQL authid (the value of the CURRENT SQLID special register) for authorization checking of dynamic SQL statements.
- Db2® uses the authid of the application process and the SQL authid (the value of the CURRENT SQLID special register) as the implicit qualifier of table, view, index, and alias names.
- Dynamic SQL statements use the values of application programming options that were specified during installation. The installation option USE FOR DYNAMICRULES has no effect.
- GRANT, REVOKE, CREATE, ALTER, DROP, and RENAME statements can be executed dynamically.

The important point here is that the authid of the application process is used for authorization checking of dynamic SQL statements, including access to the Db2® catalog tables.

In a simple Db2® installation, the user's TSO logon ID is used as the Db2® authid, so SELECT access on the appropriate Db2® catalog tables needs to be granted to a list of Db2® authids which would be the same IDs as the list of the TSO logon IDs used by ZDT/Db2 users. You can achieve the same result by granting SELECT access on the Db2® catalog tables to PUBLIC.

In a more complex Db2® installation, an external security server may be used to control access to Db2® resources, and a Db2® authorization exit may be used to convert a user TSO logon ID into a primary and one or more secondary Db2® authids. See the *Db2® Administration Guide* for the appropriate version of Db2® for more information.

When an external security server is used to manage Db2® authorization, and a Db2® authorization exit is available, one approach to granting ZDT/Db2 users SELECT access to Db2® catalog tables is shown here:

- Determine an otherwise unused, generic Db2® authid for use by ZDT/Db2 users. For example, USER. You can select any Db2® authid as required.
- Use the ZDT/Db2 sample jobs (see Sample jobs to grant SELECT access on the Db2 catalog tables on page 163) to grant SELECT access on the Db2® catalog tables to the selected Db2® authid. You must modify the sample jobs to achieve this.
- Configure the security software and Db2® authorization exit to:

1. Validate that the user has authority to use the USER Db2® authid.
2. Convert the user's TSO logon ID into Db2® authid USER during the connection to Db2®. This means that authorization checking for dynamic SQL statements issued by ZDT/Db2 are made using Db2® authid USER, rather than the user TSO logon ID.

- As an additional security measure, grant EXECUTE on the ZDT/Db2 plans to the generic Db2® authid, rather than PUBLIC.

When implemented, this approach can be used to ensure:

1. SELECT access on the Db2® catalog tables is not granted to PUBLIC.
2. Only the generic Db2® authid need have SELECT access on the Db2® catalog tables.
3. The external security system can be used to validate and control which users are allowed access to the generic Db2® authid, and hence to ZDT/Db2 functionality.
4. Individual ZDT/Db2 users may have no direct access - using a Db2® authid that is the same as their TSO logon ID - to the Db2® catalog tables.
5. Only those users with access to the generic Db2® authid can run the ZDT/Db2 plans.

Sample jobs to grant SELECT access on the Db2® catalog tables

These jobs are provided in HFM.SHFMSAM1 to grant SELECT access on the Db2® catalog tables to PUBLIC:

1. HFM2GSC2 (Db2® V12)
2. HFM2GSC3 (Db2® V13)
3. HFM2GVW2 (Db2® V12)
4. HFM2GVW3 (Db2® V13)
5. HFM2GV22 (Db2® V12)
6. HFM2GV23 (Db2® V13)

You should review the GRANT statements in these members whenever you install a later version of Db2®. New versions of Db2® may include new or updated catalog tables, and ZDT/Db2 may require access to these new or updated tables.

Sample jobs HFM2GSC*n* grant SELECT access on the Db2® catalog tables to PUBLIC. This is the simplest method of ensuring ZDT/Db2 users have access to the Db2® catalog tables. It is also the least secure.

Sample jobs HFM2GVW*n*

- Create views on the Db2® catalog tables with a different owner, but the same name as the Db2® catalog tables.
- Grant access on the newly created views to PUBLIC.

If you intend to use these views, you must set the CATOWNER parameter in the HFM2POPI to match the value of the owner specified in these sample jobs.

For information about HFM2POPI, see HFM2POPI on page 446. For information about changing the options in HFM2POPT, see Changing the default options on page 180.

Sample jobs HFM2GV2*n*

- Create views on the Db2® catalog tables with a different owner, but the same name as the Db2® catalog tables.
- These views include all the Db2® catalog tables normally accessed by ZDT/Db2. Some Db2® catalog tables are not referenced and therefore no views for these Db2® catalog tables are defined.
- Not every column of the referenced Db2® catalog tables is included in the views. The columns that are included are required for the correct functioning of ZDT/Db2. Omitting additional columns may impact the functioning of ZDT/Db2, in some cases rendering the product unusable.
- Grant access on the newly created views to PUBLIC.

If you intend to use these views, you must set the CATOWNER parameter in the HFM2POPI to match the value of the owner specified in these sample jobs.

For information about HFM2POPI, see HFM2POPI on page 446. For information about changing the options in HFM2POPT, see Changing the default options on page 180.

Note:

1. If you create views of the Db2® catalog tables and change the CATOWNER parameter in the HFM2POPI macro, the following example outlines how ZDT/Db2 attempts to access the Db2® catalog. For this example, it is assumed that the owner of the views (CATOWNER value) is SYSIBMV.
    - ZDT/Db2 generates an SQL statement like SELECT * FROM **SYSIBMV**.SYSTABLES.
    - ZDT/Db2 attempts to run the SQL statement.
    - If this attempt fails, ZDT/Db2 does NOT attempt to access the Db2® catalog using, for example, SELECT * FROM SYSIBM.SYSTABLES.
    - In summary, ZDT/Db2 attempts to access the Db2® catalog once only, using the specified CATOWNER value as the owner for the relevant catalog tables.
2. If you do not want to grant SELECT access against the Db2® catalog tables or views to all users (that is, to PUBLIC), you can customize the GRANT statements in the sample jobs to list individual user IDs.
3. In situations where you do not want to expose all the information in the Db2® catalog tables, you can define views (a "minimal subset") that refer only to the Db2® catalog tables, and columns within those tables, that are needed by ZDT/Db2. Sample jobs that do this are provided in HFM.SHFMSAM1:
    - **HFM2GV22** for Db2® version 12 subsystems.
    - **HFM2GV23** for Db2® version 13 subsystems.
4. The view definitions in these sample jobs refer only to the columns needed by ZDT/Db2, and not all catalog tables are included. There are consequences to using views that include only some columns of the actual Db2® catalog table. For example, option 3.4 of ZDT/Db2 provides an option to show all columns of certain catalog tables. The columns shown will reflect the columns of the view and not necessarily all columns of the catalog table, reducing the usefulness of the product.
5. If your installation security standards prevent access to the SYSIBM.*AUTH tables, you can omit these views, or create the views, and then not grant access. This has the effect of disabling the Object Privileges utility (Option 3.5).

6. You can define different views for different Db2® subsystems. For example, you could define the views in HFM2GVW2 against a development system, but the views in HFM2GV22 against a production system. You must ensure, however, that the owner (CATOWNER value) used is consistent across all Db2® subsystems accessed by ZDT/Db2.
7. When you define views on the Db2® catalog table for use by ZDT/Db2, you cannot change the object name of the catalog table; you can only change the owner. For example, defining a view **SYSIBMV.SYSTABLES** on the table **SYSIBM.SYSTABLES** is valid. However, defining a view **SYSIBM.TABLES** on the table **SYSIBM.SYSTABLES** is invalid. You cannot rename the columns of the catalog tables in the views you define for use by ZDT/Db2. The owner (CATOWNER value) of the views you create must be the same for every view.

Adding ZDT/Db2 to the Db2® Administration Launchpad

You can add ZDT/Db2 to the list of applications on the Db2® Admin Launchpad - see *DB2 Administration Tool for z/OS User's Guide and Reference*, Chapter 3, for a description of the Db2® Admin Launchpad and how it is customized.

You need to complete the following tasks:

1. Define a Launchpad entry for ZDT/Db2 using the instructions in the Db2® Administration Tool manual. Use the values shown in Table 23: Values for Db2 Admin Launchpad definition on page 166.
2. Locate the sample exec, HFM2ADIN, (distributed in HFM.SHFMSAM1.)
3. Copy HFM2ADIN to a library that is allocated to SYSEXEC DD NAME when the Db2® Admin Launchpad is active. This could be the SADBEXEC library.
4. In your copy of HFM2ADIN, change the names for the Z Data Tools libraries in the following statements to the appropriate values, if you are not using the default values.

```
shfmexec = "'HFM.SHFMEXEC'"
shfmllib = "'HFM.SHFMMOD1'"
shfmmlib = "'HFM.SHFMMENU'"
shfmplib = "'HFM.SHFMPENU'"
shfmslib = "'HFM.SHFMSLIB'"shfmtlib = "'HFM.SHFMTENU'"
```

Note:

1. These instructions assume you will run ZDT/Db2 using LIBDEF/ALTLIB statements to allocate the Z Data Tools application libraries.
2. If all the Z Data Tools application libraries are available when the Db2® Admin Launchpad is active, for example, when the libraries have been allocated in the user's TSO logon procedure, you can remove all the statements to assign the Z Data Tools libraries and to initiate and terminate the LIBDEFs and ALTLIB libraries. In this situation you should copy the sample exec to the Z Data Tools exec (SHFMEXEC) library.

Table 23. Values for Db2® Admin Launchpad definition

Admin Tool Field	Value
PID	19OP1220
REL	1710
NAME	Z Data Tools Db2®
CDE	HFM
GRP	Choose a value (1..4)
STAT	Y
CMD	SELECT MODE(FSCR) CMD(%HFM2ADIN)

Adding commands to start ZDT/Db2 functions from the Db2® Administration Tool

You can add one or more line commands to start the ZDT/Db2 editor and other ZDT/Db2 functions to the list of line commands that may be issued against a list of Db2® objects in Db2® Admin Tool. See the *DB2 Administration Tool for z/OS User's Guide and Reference*, Chapter 2 for a detailed description of how to do this. One reason to do this is to allow the ZDT/Db2 editor to be invoked for a Db2® object (table, view or alias) that appears in an Admin Tool object list.

You need to complete the following tasks:

1. Locate the sample exec HFM2ADLC (distributed in HFM.SHFMSAM1).
2. Review the codes and descriptions in the sample exec. You can change the line command code or description if required. Be careful to avoid conflicts with any existing line command codes in the Admin Tool table.
3. Follow the directions in the *DB2 Administration Tool for z/OS User's Guide and Reference* to add the commands to the list of available commands for the Admin Tool "Tables, Views and Aliases" display. Be sure to check that the appropriate Admin Tool tables are updated for the various types of tables that may be displayed by Db2® Admin Tool. See the *DB2 Administration Tool for z/OS User's Guide and Reference*.
4. Copy HFM2ADIE to a library that is allocated to SYSEXEC DD NAME when the Db2® Admin Launchpad is active. This could be the SADBEXEC library.
5. In your copy of HFM2ADIE, change the names for the Z Data Tools libraries in the following statements to the appropriate values, if you are not using the default values.

```
shfmexec = "'HFM.SHFMEXEC'"
shfmllib = "'HFM.SHFMMOD1'"
shfmmlib = "'HFM.SHFMMENU'"
shfmplib = "'HFM.SHFMPENU'"
shfmslib = "'HFM.SHFMSLIB'"
shfmtlib = "'HFM.SHFMTENU'"
```

Note:

1. These instructions assume you will run ZDT/Db2 using LIBDEF/ALTLIB statements to allocate the Z Data Tools application libraries.
2. If all the Z Data Tools application libraries are available when the Db2® Admin Tool is active, for example, when the libraries have been allocated in the user's TSO logon procedure, you can remove all the statements to assign the Z Data Tools libraries and to initiate and terminate the LIBDEFs and ALTLIB libraries. In this situation you should copy the sample exec to the Z Data Tools exec (SHFMEXEC) library.

When a Z Data Tools/Db2 function is started via a Db2® Admin Tool line command, the following restrictions apply:

- You cannot issue a command to change the currently connected Db2® system.
- You cannot issue Db2® commands.
- The SQLID used when ZDT/Db2 is started is the same as the users TSO logon ID, unless modified by a Db2® authorization exit.
- Exiting the first panel displayed after the Db2® Admin Tool line command is issued returns to the Db2® Admin Tool Object List display. It is not possible to navigate to other parts of Z Data Tools. You can, however, use the pull-down options to perform ZDT/Db2 functions.

Starting ZDT/Db2 functions from an external application

You can use the ZDT/Db2 exec HFM2INEX to start a Z Data Tools/Db2 function from an external application such as a REXX exec or CLIST.

HFM2INEX is distributed in HFM.SHFMEXEC. When using this interface you need to ensure the following:

- The Z Data Tools libraries must be allocated before calling HFM2INEX.
- If you use LIBDEFs and ALTLIB statements to allocate the Z Data Tools libraries, invoke HFM2INEX using the following command:

  ```
  "SELECT MODE(FSCR) CMD(%HFM2INEX "parms") NEWAPPL(HFM2) PASSLIB SUSPEND"
  ```

  This will ensure that the Z Data Tools libraries are available to HFM2INEX.

The parameters that can be passed to HFM2INEX are shown in Table 24: Parameters for HFM2INEX on page 168.

The function codes that can be specified are shown in Table 25: Function codes for HFM2INEX on page 168.

For example, referring to Table 24: Parameters for HFM2INEX on page 168 and Table 25: Function codes for HFM2INEX on page 168, you might build the following parameter string:

```
SSID(DFE2) OW(DSN81210) NM(EMP) F(4.5)
```

In this case your select statement would be:

```
"SELECT MODE(FSCR) CMD(%HFM2INEX SSID(DFE2) OW(DSN81210) NM(EMP) F(4.5)),
NEWAPPL(HFM2) PASSLIB SUSPEND"
```

Table 24. Parameters for HFM2INEX

Parameter	Description
SSID	The Db2® subsystem (or data sharing group) ID that ZDT/Db2 should connect to.
OW	The owner (creator) of a Db2® object that ZDT/Db2 should access. The owner is only passed if the function will accept an owner. Owner values are not passed to menu panels, or any option panel. See Table 25: Function codes for HFM2INEX on page 168 for additional exceptions.
NM	The name of a Db2® object that ZDT/Db2 should access. The name is only passed if the function will accept a name. Name values are not passed to menu panels, or any option panel. See Table 25: Function codes for HFM2INEX on page 168 for additional exceptions.
F	• A code that indicates which ZDT/Db2 function should be executed. The code corresponds to the ZDT/Db2 main menu option used to select the desired function (eg 3.6 for import). • For editor functions only, if the function code has 'O' as a suffix, the editor function entry panel will be displayed. • For editor functions without the 'O' suffix, the first data displayed will be the data for the specified Db2® object.

Table 25. Function codes for HFM2INEX

Function code	Description
B	ZDT/Db2 Browse function. First panel displayed shows data for the specified Db2® object.
BO	As for B, except the editor function entry panel is the first panel displayed.
1	ZDT/Db2 View function. First panel displayed shows data for the specified Db2® object.
1O	As for 1, except the editor function entry panel is the first panel displayed.
2	ZDT/Db2 Edit function. First panel displayed shows data for the specified Db2® object.
2O	As for 1, except the editor function entry panel is the first panel displayed.

Table 25. Function codes for HFM2INEX (continued)

Function code	Description
3	Displays the ZDT/Db2 Utilities menu.
3.1	ZDT/Db2 Print function. First panel displayed is the print function entry panel.
3.2	Z Data Tools Object functions. First panel displayed is the Object Functions function entry panel. Any owner or name values are NOT passed.
3.3	ZDT/Db2 Copy function. First panel displayed is the copy (from) function entry panel.
3.4	Z Data Tools Object List function. First panel displayed in the Object List function entry panel.
3.5	Z Data Tools Object Privileges function. First panel displayed in the Object Privileges function entry panel.
3.6	ZDT/Db2 Import function. First panel displayed is the import (from) function entry panel.
3.7	ZDT/Db2 Export function. First panel displayed is the export function entry panel.
3.8	ZDT/Db2 Data Create function. First panel displayed is the data create function entry panel.
3.9	ZDT/Db2 Utilities function. First panel displayed is the Db2® utilities function entry panel. Any owner or name values are NOT passed.
3.10	ZDT/Db2 Audit Log function. First panel displayed is the audit log function entry panel. Any owner or name values are NOT passed.
4	Displays the ZDT/Db2 SQL Prototyping, Execution and Analysis menu.
4.1	Z Data Tools Basic SQL Prototyping function. First panel displayed is the basic SQL prototyping function entry panel.
4.2	Z Data Tools Advanced SQL Prototyping function. First panel displayed is the advanced SQL prototyping function entry panel. Any owner or name values are NOT passed.
4.3	Z Data Tools "Enter, Execute and Explain SQL Statements" function. First panel displayed is the "Enter, Execute and Explain SQL Statements" function entry panel. Any owner or name values are NOT passed.
4.4	Z Data Tools "Edit and Execute SQL Statements from a Data Set" function. First panel displayed is the "Edit and Execute SQL Statements from a Data Set" function entry panel. Any owner or name values are NOT passed.
4.5	Z Data Tools Explain Utilities function. First panel displayed is the explain utilities function entry panel. Any owner or name values are NOT passed.
6	Z Data Tools Enter and Execute Db2® Commands function. First panel displayed in the enter and execute Db2® commands function entry panel. Any owner or name values are NOT passed.
0	Displays the ZDT/Db2 Options menu.

Table 25. Function codes for HFM2INEX (continued)

Function code	Description
0.0	Displays the ZDT/Db2 System Options menu.
0.0.1	Displays the ZDT/Db2 Print Settings panel.
0.0.2	Displays the first ZDT/Db2 Systems Options panel.
0.0.3	Displays the ZDT/Db2 Batch Job Card information panel.
0.0.4	Displays the ZDT/Db2 Compiler Language selection panel.
0.0.5	Displays the ZDT/Db2 COBOL Processing Options panel.
0.0.6	Displays the ZDT/Db2 HLASM Processing Options panel.
0.0.7	Displays the ZDT/Db2 PL/I Processing Options panel.
0.0.8	Displays the ZDT/Db2 Temporary Data Set Allocations panel.
0.0.9	Displays the ZDT/Db2 Output Data Set Allocation Options panel.
0.0.10	Displays the ZDT/Db2 Trace Options panel.
0.1 (0.2)	Displays the first ZDT/Db2 Editor options panel.
0.3	Displays the ZDT/Db2 Utility Processing Options panel.
0.3.1	Displays the ZDT/Db2 Print Utility Options panel.
0.3.3	Displays the ZDT/Db2 Copy Utility Options panel.
0.3.4	Displays the ZDT/Db2 Object list utility options panel.
0.3.7	Displays the ZDT/Db2 Export utility options panel.
0.3.L	Displays the ZDT/Db2 Db2® LOAD utility options panel.
0.3.UL	Displays the ZDT/Db2 Db2® Utility LISTDEF options panel.
0.3.UO	Displays the ZDT/Db2 Db2® Utility OPTIONS options panel.
0.3.UT	Displays the ZDT/Db2 Db2® Utility TEMPLATE options panel.
0.3.UU	Displays the ZDT/Db2 Db2® Unload utility options panel.

When a Z Data Tools/Db2 function is started using HFM2INEX, the following restrictions apply:

- You cannot issue a command to change the currently connected Db2® system.
- The SQLID used when ZDT/Db2 is started is the same as the users TSO logon ID, unless modified by a Db2® authorization exit.
- Exiting the first panel displayed by the HFM2INEX invocation returns to the caller. It is not possible to navigate to other parts of Z Data Tools. You can however use the pull-down options to perform ZDT/Db2 functions.

These restrictions do not apply when no option code is specified for the HFM2INEX invocation.

Only the function codes listed in Table 25: Function codes for HFM2INEX on page 168 are supported. Any attempt to use any other code will result in the primary ZDT/Db2 menu being displayed; the erroneous function code will be discarded.

Chapter 16. Customizing ZDT/Db2

This chapter describes how to customize ZDT/Db2. You do this after you have installed ZDT/Db2. In particular, there are two tasks you **must** perform, "Binding Db2®" and "Customizing the **Db2 Subsystem Selection** panel".

Binding Db2® (required)

Before you can use ZDT/Db2, you must bind the ZDT/Db2 packages and application plans, and grant EXECUTE privilege on these plans to ZDT/Db2 users. The sample jobs shown here are provided in HFM.SHFMSAM1:

- **HFM2BN1K** Db2® V12 FL 100 binds the ZDT/Db2 DBRM members into collections used by the plans, HFM2PLAN and HFM2GEN, in a Db2 V12 FL 100 system.
- **HFM2BN2P** Db2® V12 binds the ZDT/Db2 plans, HFM2PLAN and HFM2GEN, into a Db2 V12 subsystem.
- **HFM2BN2K** Db2® V12 binds the ZDT/Db2 DBRM members into collections used by the plans, HFM2PLAN and HFM2GEN, in a Db2 V12 system.
- **HFM2BN3P** Db2® V13 binds the ZDT/Db2 plans, HFM2PLAN and HFM2GEN, into a Db2 V13 subsystem.
- **HFM2BN3K** Db2® V13 binds the ZDT/Db2 DBRM members into collections used by the plans, HFM2PLAN and HFM2GEN, in a Db2 V13 system.

The HFM2BN*n*K jobs must be run once when you install ZDT/Db2 into a Db2® subsystem, and then rerun for each Db2 subsystem that uses ZDT/Db2 when the Z Data Tools service PTF ++HOLD data specifies that a bind is required.

Note:

- The running of HFM2BN*n*K jobs is not required when applying Db2 maintenance.
- If binding into Db2 12 Function Level 100 (compatibility mode with Version 11), use sample HFM2BN1K instead of HFM2BN2K.

It is a requirement that ZDT/Db2 is bound using the DYNAMICRULES(RUN) bind option (the default). The use of DYNAMICRULES(BIND) prevents ZDT/Db2 from being able to dynamically execute SQL statements such as GRANT/REVOKE/ALTER and RENAME, and is therefore not recommended. The DEFINERUN and DEFINEBIND options of the DYNAMICRULES bind option are not supported. See the *Db2® Command Reference* for the appropriate version of Db2®, section "Bind and Rebind options" for a detailed explanation of the DYNAMICRULES bind options.

In HFM2BN*n*K you will see that the members bound into the packages are called ADBxxxx. These members are aliases of DBRMs and are provided in HFM.SHFMDBRM.

Select the appropriate bind jobs, depending on the version of the target Db2® system. Run both the "bind plan" and "bind package" jobs, to bind the packages and plans, and to grant the Db2® EXECUTE privilege on the ZDT/Db2 plans to PUBLIC. If you do not grant EXECUTE privilege on the ZDT/Db2 plans to public, you will encounter Db2® authority errors, including SQLCODE-551 errors, when you attempt to run ZDT/Db2.

You **must** run a version of this job for each Db2® subsystem you want ZDT/Db2 to use, either by direct or remote connection. Select the appropriate bind plan and bind package jobs for each Db2® subsystem you want ZDT/Db2 to use. Refer to the comments in the job for information on modifications you need to make to the job to do this. A return code of 4 is acceptable from this job.

When there are multiple Db2® systems at your site, on possibly different LPARs or different physical machines, and the Db2® systems are configured so that remote access is possible from one Db2® system to another using the Db2® DDF facility, you must ensure that:

- You run a version of the bind jobs against **every Db2® system that ZDT/Db2 will access**, either directly or remotely. The HFM2BN*n*K jobs require access to the DBRMs shipped with the product (in SHFMDBRM). You may need to copy this DBRM library so that the DBRMs are available to the HFM2BN*n*K jobs when these are run. You need to do this for each LPAR or physical machine where a Db2® system, that ZDT/Db2 is to access, is located.
- The bind jobs (HFM2BN*n*P and HFM2BN*n*K as appropriate) complete successfully; that is the ZDT/Db2 plans and packages are bound correctly. See the comments in the HFM2BIND job, for the appropriate version of Db2®, for acceptable return codes.
- EXECUTE privilege on the Db2® plans is successfully granted to PUBLIC. If this step is not successful it is possible that ZDT/Db2 will work correctly when connected to a local Db2® system, but fail when an attempt is made to access a remote Db2® system.

You can change the plan names for the ZDT/Db2 plan and the ZDT/Db2 reverse engineering plan. To do this modify HFM2BN*n*P as appropriate. The default names for these plans are HFM2PLAN andHFM2GEN respectively. If you choose to change these plan names, you **must** change the values for `PLAN` and `PLAN2` to your chosen names, in your HFM2SSDM macros, when you define your Db2® subsystems to ZDT/Db2.

For information about ZDT/Db2 and Db2® subsystems, see Defining all Db2 systems that ZDT/Db2 will access in HFM2POPT (required) on page 174. For information about the HFM2SSDM macro, see HFM2SSDM on page 426.

Running multiple versions of ZDT/Db2

This topic explains how you can install multiple different versions of ZDT/Db2.

Attention: You must ensure that each ZDT/Db2 installation is completely independent of any others.

In particular, you must ensure that:

- Different (and unique) plan and package names are used for each ZDT/Db2 installation.
- The ZDT/Db2 plan and packages are bound using the appropriate DBRM modules for each version of ZDT/Db2.
- Each version of ZDT/Db2 uses a unique HFM2POPT module, and that module must specify the correct plan name for that version of ZDT/Db2.
- When ZDT/Db2 is used to access remote Db2® systems, the plan and packages for each ZDT/Db2 version must be bound on every Db2® system that might be accessed using ZDT/Db2.

The notes below cover the situation where an existing version (say version n) of ZDT/Db2 is installed. A new version (say n+1) of ZDT/Db2 is available and will be installed. The two versions are to be run concurrently while acceptance testing is completed for the newer version.

The default plan names and package names are used for the existing version of ZDT/Db2 that is already installed:

- HFM2PLAN
- HFM2GEN
- HFM2PLPK
- HFM2GNPK

For the new version of ZDT/Db2, select new plan and package names, for example:

- HFM2PLND (instead of HFM2PLAN)
- HFM2GEND (instead of HFM2GEN)
- HFM2PLPD (instead of HFM2PLPK)
- HFM2GNPD (instead of HFM2GNPK)

Any new names can be selected as long as they are different to all other ZDT/Db2 plan and package names that are already in use.

For the new version of ZDT/Db2, modify the sample bind jobs HFM2BN*n*P and HFM2BN*n*K as appropriate.

Run the modified bind jobs, both the bind plan and bind package, on every Db2® system. Ensure that the DBRM library specified in the new bind job is for the new version of ZDT/Db2.

Modify the sample HFM2POPT job for the new version of ZDT/Db2. Specify the new plan names in the HFM2SSDM macro definition for each Db2® system that ZDT/Db2 will access, either directly or remotely, as shown below:

```
HFM2SSDM  SSID=XXXX,PLAN=HFM2PLND,PLAN2=HFM2GEND
```

Assemble the new HFM2POPT module, and ensure that this is available when a user runs the new version of Db2®.

For information about modifying HFM2POPT, see Changing the default options on page 180. For information about HFM2SSDM, see HFM2SSDM on page 426.

Defining all Db2® systems that ZDT/Db2 will access in HFM2POPT (required)

Before you can use ZDT/Db2, you **must** define to ZDT/Db2 the Db2® subsystems, or Db2® data sharing groups, which ZDT/Db2 will use.

To do this you must provide an HFM2SSDM macro for each required subsystem, to be included in the ZDT/Db2 options macro, HFM2POPT. If you are using Db2® data sharing groups, you must also provide an HFM2SSDM macro for each data sharing group id.

If your site uses Db2® data sharing groups, review the explanatory information for the LIST parameter of the HFM2POPI macro in LIST on page 452. This information explains how ZDT/Db2 determines the Db2® systems that are candidates for connection on the local z/OS® server.

You must then install HFM2POPT on your ZDT/Db2 system, using the usermod HFM2UMDP. HFM2SSDM on page 426 describes the syntax of the options in HFM2SSDM. See Changing the default options on page 180 for information about other changes to make to HFM2POPT and how to install your version of HFM2POPT.

Each HFM2SSDM macro describes one subsystem or data sharing group. You can provide as many HFM2SSDM macros as you require in one HFM2POPT macro. You can specify an HFM2SSDM macro for any supported version of Db2® in the same HFM2POPT macro.

If your environment includes many Db2® subsystems, you can optionally specify whether ZDT/Db2 displays all Db2® subsystems on the subsystem selection menu, or only those systems that are currently active. You can also restrict which Db2® subsystems ZDT/Db2 will attempt to connect to. To do this, you specify an HFM2POPI macro and include it in HFM2POPT. HFM2POPI on page 446 describes the syntax of the options in HFM2POPI.

You can also optionally provide site-specific initial values for LISTDEF and TEMPLATE libraries, and an initial OPTIONS EVENT statement, which will be used in every batch utility job generated by ZDT/Db2. You can also specify default template names to be used in various clauses in several Db2® utility statements.

You provide these site-specific values, also, in an HFM2SSDM macro in HFM2POPT. For more information, see HFM2SSDM on page 426. For information on the use of the site-specific LISTDEF and TEMPLATE libraries, and the OPTIONS EVENT statement, refer to the *Z Data Tools User's Guide and Reference for DB2 Data*.

See Examples of HFM2SSDM macros on page 175 for sample code showing the HFM2SSDM macros you would code to define a Db2® subsystem, and a Db2® group attachment identifier for a data sharing group.

Examples of HFM2SSDM macros

The following sample code shows the HFM2SSDM macro statements you would code to define Db2® subsystems to ZDT/Db2, for Db2® version 12, and data sharing groups for Db2® version 12. You would code HFM2SSDM macros similar to these, and include them in your version of HFM2POPT. (Do not change HFM2SSDM in HFM.SHFMMAC1.) For an explanation of the statements, see the notes at the end of the example.

```
HFM2SSDM SSID=DF52,PLAN=HFM2PLAN,PLAN2=HFM2GEN,              +   DF52
      TYPE=SUBSYS,                                           +
      AUDIT=(REQUIRED,SMF),                                  +
      EDIT_MAX_ROWS=0,                                       +
      DB2LLIB=('DB2VC10.DF52.SDSNEXIT','DB2.VC10.SDSNLOAD'), +
      DB2RLIB=('DB2VC10.DF52.RUNLIB.LOAD'),                  +
      DB2CLIB=('DB2.VC10.SDSNCLST'),                         +
      DB2MLIB=('DB2.VC10.SDSNSPFM'),                         +
      DB2PLIB=('DB2.VC10.SDSNSPFP','DB2.VC10.SDSNPFPE'),     +
      DB2SLIB=('DB2.VC10.SDSNSPFS'),                         +
      DB2TLIB=('DB2.VC10.SDSNSPFT'),                         +
      DB2PROC=('DB2VC10.DF52.PROCLIB'),                      +
      DESC='DB2 Version 12.1 - Production'
```

```
*
         HFM2SSDM SSID=DF61,PLAN=HFM2PLAN,PLAN2=HFM2GEN,               +   DF61
               DISPLAY=HIDDEN,                                         +
               AUDIT=OPTIONAL,                                         +
               DB2LLIB=('DB2VC10.DF61.SDSNEXIT','DB2.VC10.SDSNLOAD'),  +
               DB2RLIB=('DB2VC10.DF61.RUNLIB.LOAD'),                   +
               DB2CLIB=('DB2.VC10.SDSNCLST'),                          +
               DB2MLIB=('DB2.VC10.SDSNSPFM'),                          +
               DB2PLIB=('DB2.VC10.SDSNSPFP','DB2.VC10.SDSNPFPE'),      +
               DB2SLIB=('DB2.VC10.SDSNSPFS'),                          +
               DB2TLIB=('DB2.VC10.SDSNSPFT'),                          +
               DB2PROC=('DB2VC10.DF61.PROCLIB'),                       +
               DESC='DB2 Version 12.1 - Development'
*
         HFM2SSDM SSID=DF13,PLAN=HFM2PLAN,PLAN2=HFM2GEN,               +   DF13
               TYPE=SUBSYS,                                            +
               AUDIT=REQUIRED,                                         +
               EDIT_MAX_ROWS=0,                                        +
               FORCE_WITH_UR=Y,                                        +
               TABLE_LOCKING=NO,                                       +
               DB2LLIB=('DB2VC10.DF13.SDSNEXIT','DB2.VC10.SDSNLOAD'),  +
               DB2RLIB=('DB2VC10.DF13.RUNLIB.LOAD'),                   +
               DB2CLIB=('DB2.VC10.SDSNCLST'),                          +
               DB2MLIB=('DB2.VC10.SDSNSPFM'),                          +
               DB2PLIB=('DB2.VC10.SDSNSPFP','DB2.VC10.SDSNPFPE'),      +
               DB2SLIB=('DB2.VC10.SDSNSPFS'),                          +
               DB2TLIB=('DB2.VC10.SDSNSPFT'),                          +
               DB2PROC=('DB2VC10.DF13.PROCLIB'),                       +
               TMPDDLN=DF13TMDD,                                       +
               STMJCL1='DISP=SHR,DSN=FIRST.TEMPLATE.LIBRARY,',         +
               STMJCL2='DISP=SHR,DSN=SECOND.TEMPLATE.LIBRARY',         +
               OPTEVT1='ITEMERROR,HALT',                               +
               CPYCPYN=DF13CPYC,                                       +
               LODINDN=DF13LDIN,                                       +
               DESC='DB2 Version 12.1 - Production'
*
         HFM2SSDM SSID=DG03,PLAN=HFM2PLAN,PLAN2=HFM2GEN,               +   DG03
               TYPE=GROUP,                                             +
               DISPLAY=YES,                                            +
               DB2LLIB=('DB2VC10.DF61.SDSNEXIT','DB2.VC10.SDSNLOAD'),  +
               DB2RLIB=('DB2VC10.DF61.RUNLIB.LOAD'),                   +
               DB2CLIB=('DB2.VC10.SDSNCLST'),                          +
               DB2MLIB=('DB2.VC10.SDSNSPFM'),                          +
               DB2PLIB=('DB2.VC10.SDSNSPFP','DB2.VC10.SDSNPFPE'),      +
               DB2SLIB=('DB2.VC10.SDSNSPFS'),                          +
               DB2TLIB=('DB2.VC10.SDSNSPFT'),                          +
               DB2PROC=('DB2VC10.DF61.PROCLIB')                        +
               DESC='DB2 Group Attach ID number 1'
*
         HFM2SSDM SSID=DEFAULT                                           DEFAULT
```

Note: In this example the + characters are in column 72.

DF52

1. These statements define a Db2® 12 subsystem, called `DB2 Version 12.1 – Production`.
2. The SSID is DF52.
3. Auditing is required. Audit records are written as SMF records.
4. EDIT_MAX_ROWS is set to 0. This disables *"large"* editor mode for all users.
5. The Db2® library names are specified using DB2LLIB, DB2RLIB, and so on.

DF61

1. These statements define a Db2® 12 subsystem, called `DB2 Version 12.1 – Development`.
2. The SSID is DF61.
3. Auditing is optional.
4. DISPLAY=HIDDEN is specified. This Db2® subsystem will not appear on the Z Data Tools Db2® subsystem selection list.
5. The Db2® library names are specified using DB2LLIB, DB2RLIB, and so on.

DF13

1. These statements define a Db2® 12 subsystem, called `DB2 Version 12.1 – Production`.
2. The SSID is DF13.
3. Auditing is required. Audit records are written to data sets.
4. EDIT_MAX_ROWS is set to 0. This disables *"large"* editor mode for all users.
5. FORCE_WITH_UR is set to Y. This adds the `WITH UR` clause to any SQL statements executed by the ZDT/Db2 editor.
6. TABLE_LOCKING is set to NO. This prevents any user specifying the *"Lock table"* option when using the ZDT/Db2 editor.
7. The TMPDDLN keyword specifies a ddname for the site-specific template library of DF13TMDD. The STMJCL1 and STMJCL2 keywords specify two concatenated data sets named FIRST.TEMPLATE.LIBRARY and SECOND.TEMPLATE.LIBRARY for this ddname. The JCL generated by these keywords will be:

   ```
   //DF13TMDD DD DISP=SHR,DSN=FIRST.TEMPLATE.LIBRARY
   //                  DD DSN=SECOND.TEMPLATE.LIBRARY
   ```

8. The OPTEVT1 keyword will generate this OPTIONS EVENT statement:

   ```
   OPTIONS EVENT(ITEMERROR,HALT)
   ```

9. The CPYCPYN keyword specifies a template name of DF91CPYC to be used in a COPYDDN clause in a COPY utility statement.
10. The LODINDD keyword specifies a template name of DF13LDIN to be used in an INDDN clause in the LOAD utility statement.
11. The Db2 library names are specified using DB2LLIB, DB2RLIB, and so on.

DG03

1. These statements define a Db2® 12 data sharing group called `DB2 group attach ID number 1`.
2. The SSID is DG03.

3. TYPE=GROUP specifies that the entry is for a Db2® group, not a Db2® subsystem.
4. The Db2® library names are specified using DB2LLIB, DB2RLIB, and so on.

DEFAULT

This statement is required. Its only parameter is DEFAULT. The last (or only if there is only one) HFM2SSDM macro invocation must specify DEFAULT.

Related information

db2alib.dita

Customizing the ZDT/Db2 options

ZDT/Db2 also provides a number of processing options that you can customize to suit your installation requirements. You specify these options in HFM2POPI. For a description of the options in HFM2POPI, and the values you can specify, see HFM2POPI on page 446. These options determine how your users will access the Db2® subsystems available to ZDT/Db2.

You then include your HFM2POPI macros in HFM2POPT, using the usermod HFM2UMDP. See Changing the default options on page 180 for information about other changes you can make to HFM2POPT and how to install HFM2POPT.

Parameters for HFM2POPI are optional. If you do not want to change any of the options in HFM2POPI, you do not need to provide any parameters for HFM2POPI. However, you must supply an HFM2POPI statement, without parameters if appropriate.

Examples of HFM2POPI macros

The following examples show the HFM2POPI statements you would code to change the default values for some ZDT/Db2 options. You include this code in your version of HFM2POPT. (Do not change HFM2POPI in HFM.SHFMMAC1.)

```
Example 1
         HFM2POPI                                                +
               LIST=ACTIVE,                                      +
               CONNECT=DEFINED,                                  +
               SSIDCMD1=CONNECT,                                 +
               SSIDCMD2=SSID,                                    +
               CATOWNER=SYSIBMV,                                 +
               EDITCAPS=(INITON,PROFILE,CAPSCMD)
Example 2
         HFM2POPI                                                +
               LIST=ALL,                                         +
               CONNECT=ANY,                                      +
               SSIDCMD1=SSID,                                    +
               SSIDCMD2=DB2SYS,                                  +
               CATOWNER=SYSIBM,                                  +
               EDITCAPS=(INITON,FIXED,NOCAPSCMD)
```

Note: In these examples the + character is in column 72.

Example 1

1. LIST=ACTIVE specifies that only Db2® subsystems that are active (on the local z/OS® system), and any group items, on the subsystem selection list, will be shown.
2. CONNECT=DEFINED specifies that ZDT/Db2 will only attempt to connect to active Db2® subsystems or groups for which there is an HFM2SSDM macro entry. The Db2® subsystem or group must be active for ZDT/Db2 to attempt the connection.
3. SSIDCMD1=CONNECT specifies that the word CONNECT can be used as the first command to change the currently connected Db2® subsystem, when entered from ZDT/Db2 panels that support the command. This replaces the default value for SSIDCMD1 of SSID. CONNECT can be abbreviated to CONN.

   SSIDCMD2=SSID specifies that the word SSID can be used as the second command to change the currently connected Db2® subsystem, when entered from ZDT/Db2 panels that support the command. This replaces the default value for SSIDCMD2 of DB2SYS.

   The net effect of these two statements is to define two commands, CONNECT and SSID. CONNECT effectively replaces the DB2SYS command. The SSID command remains available.
4. CATOWNER=SYSIBMV specifies that the catalog owner for Db2® catalog tables will be SYSIBMV, replacing the default value of SYSIBM. You must ensure that suitable Db2® views, with an owner of SYSIBMV, have been defined for all Db2® catalog tables accessed by ZDT/Db2.
5. EDITCAPS=(INITON,PROFILE,CAPSCMD) sets the ZDT/Db2 editor caps option as follows:

   For new users the CAPS edit option is set to ON. The initial setting is stored in the user's ISPF profile and can be changed using either the global or local edit option's panels. Each ZDT/Db2 edit session starts with the CAPS command set to the current edit option's setting. The CASE command can be used within any ZDT/Db2 edit session. The only difference between this, and the installation default, (INITOFF,PROFILE,CAPSCMD), is that new users begin with the CAPS edit option set to ON. With the default setting new users begin with the CAPS edit option set to off.

Example 2

1. LIST=ALL specifies that all Db2® subsystems defined with an HFM2SSDM macro entry on the subsystem selection list, whether active or not, will be shown. The exception is any Db2® subsystem defined with DISPLAY=HIDDEN in the appropriate HFM2SSDM macro. Other Db2® systems defined on the local z/OS® system, but for which there is no HFM2SSDM macro entry, will also appear on the subsystem selection list.
2. CONNECT=ANY specifies that ZDT/Db2 will attempt to connect to any active Db2® subsystem or group that appears on the subsystem selection list, regardless of whether there is an HFM2SSDM macro entry for the Db2® subsystem or group.
3. SSIDCMD1=SSID specifies that the default value (SSID) is to be used as a command that can be used to change the currently connected Db2® subsystem, when entered from ZDT/Db2 panels that support the command.

SSIDCMD2=DB2SYS specifies that the default value (DB2SYS) is to be used as a second command that can be used to change the currently connected Db2® subsystem, when entered from ZDT/Db2 panels that support the command.

4. CATOWNER=SYSIBM specifies that the catalog owner for Db2® catalog tables will be SYSIBM, which is the default. ZDT/Db2 will access the Db2® catalog tables directly.
5. EDITCAPS=(INITON,FIXED,NOCAPSCMD) sets the ZDT/Db2 editor caps option as follows:

   Every ZDT/Db2 edit session starts with the CAPS set to ON. The initial setting cannot be altered using the Edit Options. The editor CAPS command is disabled. The editor CASE command is still available. Lower case data can only be entered by first issuing the editor CASE MIXED command.

Usage tips

The three most important HFM2POPI and HFM2SSDM parameters for controlling the ZDT/Db2 subsystem selection list are:

- CONNECT (HFM2POPI)
- LIST (HFM2POPI)
- DISPLAY (HFM2SSDM)

The default for the CONNECT parameter is DEFINED. Consider changing this to ANY. With CONNECT=DEFINED, ZDT/Db2 will only attempt connection to Db2 systems that are defined in the HFM2POPT.

The default for the LIST parameter is ALL. This can be changed to show any combination of active, inactive, or unavailable subsystems or groups.

The default for the DISPLAY parameter is YES. A Db2® subsystem or group will be displayed if it is eligible to be displayed based on how the HFM2POPI LIST parameter is set.

You can prevent inactive or unavailable Db2® systems or groups being displayed on the ZDT/Db2 subsystem selection list by specifying LIST=ACTIVE.

You can prevent an *individual* Db2® system or group being displayed on the ZDT/Db2 subsystem selection list by specifying DISPLAY=HIDDEN. A typical use of this feature is to hide one or more Db2® subsystems that are also part of a Db2® data sharing group, leaving the Db2® data sharing group visible.

Changing the default options

Default processing options are supplied with ZDT/Db2 in the module HFM2POPT. You can change these options to suit your installation requirements. You also use HFM2POPT to define your Db2® subsystems, and, in a Db2® data sharing environment, your Db2® group attachment identifiers. You use the usermod HFM2UMDP to install your version of HFM2POPT.

Note: The options in HFM2POPT are also available to Z Data Tools Base function in HFM0POPT. However, any option changed in the ZDT/Db2 options macro only takes effect in ZDT/Db2.

You change the options as follows:

1. Copy the member HFM2POPT from HFM.SHFMSAM1 into your own source library.
2. Change the options in HFM2POPT in your library according to your requirements. For a description of the options in HFM2POPT, and the values you can specify, see Z Data Tools options on page 383.
3. Include your HFM2SSDM macros, (as described in Defining all Db2 systems that ZDT/Db2 will access in HFM2POPT (required) on page 174, and HFM2SSDM on page 426), in the HFM2POPT member in your library.
4. Include your HFM2POPI macros (if required), (as described in Customizing the ZDT/Db2 options on page 178, and HFM2POPI on page 446), in the HFM2POPT member in your library.
5. Modify the HFM2UMDP member in HFM.SHFMSAM1 to meet your site's requirements. Refer to the usermod for information about changes you might need to make.
6. Install SMP/E usermod HFM2UMDP.

Note: You can also use the sample job HFM2POPH to assemble HFM2POPT if you do not want to use SMP/E.

Setting the default national language

You can change the default national language for your installation that is used by ZDT/Db2 for batch processing.

The language used by ZDT/Db2 under ISPF depends on the language setting for your ISPF session. To change the language for ZDT/Db2 batch processing, set the LANGUAGE option in HFM2POPT to your language. See LANGUAGE on page 403 for more information about the LANGUAGE option.

For other customization you can do for ZDT/Db2 for national languages, refer to Customizing ZDT/Db2 for national languages on page 211.

Changing the JCL skeletons for batch mode

Several functions in ZDT/Db2 are available in batch mode. For these functions to run successfully, appropriate job control must be provided. This is done by means of the set processing options and job control skeletons. The job control skeletons for ZDT/Db2 are the members HFM2FTSL and HFM2FTEX in HFM.SHFMSLIB.

HFM2FTSL

HFM2FTSL assigns a STEPLIB DD concatenation which comprises the ZDT/Db2 load library and the Db2® load libraries for the Db2® subsystem you are using. (The Db2® load libraries are specified by DB2LLIB in the HFM2SSDM macro for the Db2® subsystem you are using, in HFM2POPT. For more information, see HFM2SSDM on page 426.)

HFM2FTSL assumes that you have installed ZDT/Db2 into the default target libraries, and that the load library is HFM.SHFMMOD1. If you have installed ZDT/Db2 into a different library, or if you do not want the load library in a STEPLIB DD concatenation (for example, if HFM.SHFMMOD1 is in your LINKLIST), you must modify the HFM2FTSL skeleton accordingly.

You modify HFM2FTSL using the usermod HFM2UMDB. HFM2UMDB is distributed in HFM.SHFMSAM1. To do this:

1. Copy the member HFM2FTSL from HFM.SHFMSLIB to your own source library.
2. To change the name of the load library, change the line:

   ```
   )SET SHFMMOD1 = HFM.SHFMMOD1
   ```

   to read:

   ```
   )SET SHFMMOD1 = your.loadlib
   ```

   where *your.loadlib* is the name of the load library where you installed ZDT/Db2.

   To change the skeleton so that no STEPLIB DD statement is generated for the load library, change the line:

   ```
   )SET SHFMMOD1 = HFM.SHFMMOD1
   ```

   to read:

   ```
   )SET SHFMMOD1 = &Z
   ```

   Note: A STEPLIB DD statement will always be generated for the Db2® load libraries for the Db2® subsystem you are using.

3. Modify the HFM2UMDB member in HFM.SHFMSAM1 to meet your site's requirements. Refer to the usermod for information about changes you might need to make.
4. Install SMP/E usermod HFM2UMDB.

Note: ZDT/Db2 does not provide support for the automatic generation of job routing control statements in the JCL generated by HFM2FTSL.

HFM2FTEX

HFM2FTEX provides printer information to HFM2FTSL. You might also need to modify HFM2FTEX, for example, to change the DCB information for SYSPRINT.

Modify HFM2FTEX using the usermod HFM2UMDE, which is distributed in HFM.SHFMSLIB. To do this:

1. Copy the member HFM2FTEX from HFM.SHFMSLIB to your own source library.
2. Modify one or more of the following three statements in HFM2FTEX in your library, as required. Do not change any other statements in HFM2FTEX.

   ```
   //SYSPRINT DD SYSOUT=*
   //HFMTSPRT DD SYSOUT=*
   //SYSTERM  DD SYSOUT=*
   ```

   For example, you might want to change

   ```
   //SYSPRINT DD SYSOUT=*
   ```

   to

   ```
   //SYSPRINT DD SYSOUT=*,DCB=(RECFM=FBA,LRECL=133)
   ```
3. Modify the HFM2UMDE member in HFM.SHFMSAM1 to meet your site's requirements. Refer to the usermod for information about changes you might need to make.
4. Install SMP/E usermod HFM2UMDE.

Customizing to protect update functions in ZDT/Db2

You can use an external security product to write facility class rules to protect update functions within ZDT/Db2.

The following ZDT/Db2 functions are considered to be update functions:

Table 26. ZDT/Db2 update functions

Function	Menu option	Description
D2E	2	Db2® edit
DBC	3.3	Copy utility
D2I	3.6	Import utility
D2G	3.8	Db2® data create
DBSBSP	4.1	Basic select prototyping
DBSASP	4.2	Advanced select prototyping
DBSEDX	4.4	Db2® edit and execute SQL
DBSENX	4.3	Db2® enter and execute SQL

To protect update functions, specify SEC=YES in the HFM2POPT module (see the SEC parameter in Z Data Tools options on page 383).

The facility class rules that are required are:

FILEM.DB2.UPDATE
FILEM.FUNCTION.*function_code*

Example 1: To protect all Db2® update functions

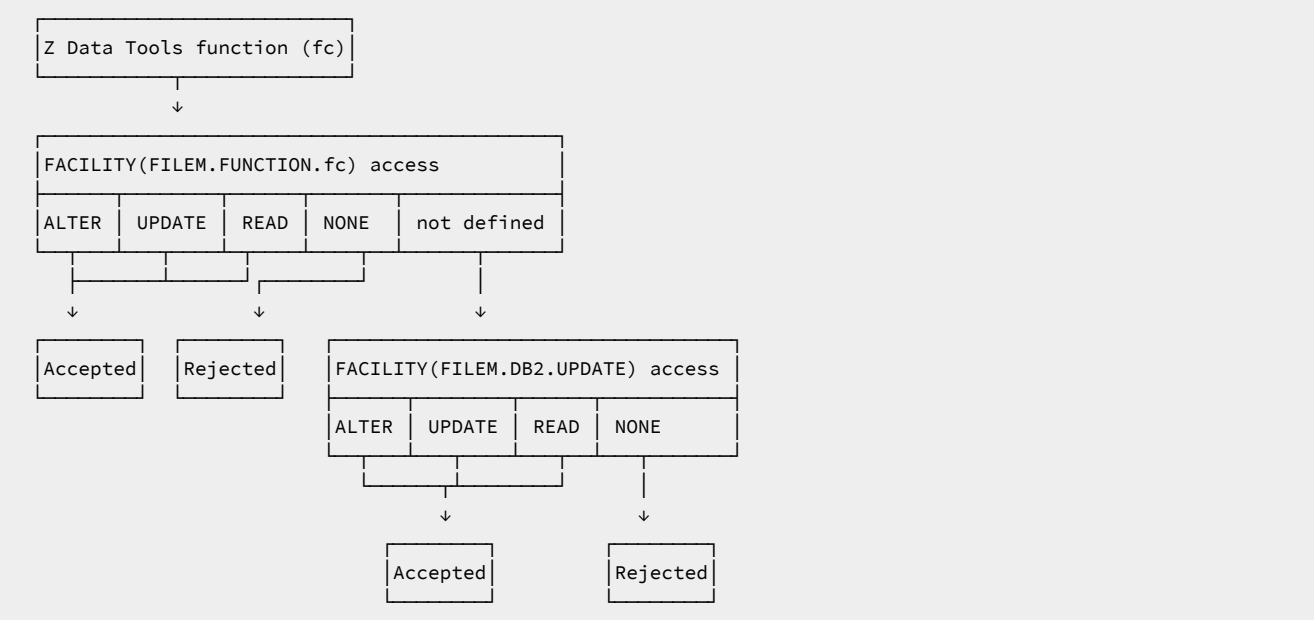
- Specify SEC=YES in the HFM2POPT
- Write a facility class rule for FILEM.DB2.UPDATE

Example 2: To protect the Db2® editor function only

- Specify SEC=YES in the HFM2POPT
- Write a facility class rule for FILEM.FUNCTION.D2E

The following diagram shows the processing that is used when function rules, update facility class rules, or both are specified.

Figure 15. Security system validation for update functions



Note: In most cases it is preferable to use Db2® security, with or without an external security server, to control update access to Db2® objects. Db2® security allows access to be specified for individual Db2® objects at various levels of access (SELECT, INSERT, UPDATE, DELETE), and to individual Db2® authids. The same level of control is not possible using ZDT/Db2.

Customizing ZDT/Db2 for use in production environments

If you intend to use ZDT/Db2 to access Db2® systems that support a high throughput online transaction processing environment, you should consider the potential for adverse performance and concurrency problems arising from the use of ZDT/Db2, particularly the ZDT/Db2 editor.

This section outlines concurrency and performance issues that might arise:

- The use of *"large"* mode in the ZDT/Db2 editor.

  The Z Data Tools Db2® editor can operate in two modes: *"normal"* and *"large"*. The mode of operation is determined by the value entered in the **row count** field for those ZDT/Db2 functions that use the ZDT/Db2 editor to display data. The characteristics of the two editor modes are documented in the *Z Data Tools User's Guide and Reference for DB2 Data*.

  The use of *"large"* editor mode might have negative Db2® performance implications. When the ZDT/Db2 editor operates in *"large"* mode, it uses a Db2® scrollable cursor for access to Db2® data. This minimizes the memory usage in the ZDT/Db2 user's TSO address space, but might require Db2® to build a temporary copy of the entire result table in a Db2® temporary database. For large tables, this can lead to SQLCODE-904 (unavailable resource) on table spaces defined within the Db2® temporary database. For these reasons, providing access to the ZDT/Db2 editor in production Db2® environments, where there are large Db2® tables, should be carefully considered.

  The product installer can disable the use of *"large"* editor mode, by Db2® subsystem. This is achieved by setting the HFM2SSDM macro parameter
  - EDIT_MAX_ROWS to a non-zero value. See EDIT_MAX_ROWS on page 434.
  - PROD_EDIT to YES. See PROD_EDIT on page 439. Note that setting PROD_EDIT=YES over-rides EDIT_MAX_ROWS=0, if specified, replacing it with EDIT_MAX_ROWS=1000. The HFM2SSDM macro is discussed further in Defining all Db2 systems that ZDT/Db2 will access in HFM2POPT (required) on page 174.
- The use of table locking options

  The Z Data Tools Db2® editor has an option that allows a table to be locked for the duration of the editor session. The table can be locked in *"SHARED"* or *"EXCLUSIVE"* mode, see the description of the 'LOCK TABLE' statement in the SQL Reference manual for the appropriate version of Db2® for a description of these options.

  The use of either the *"SHARED"* or *"EXCLUSIVE"* options is inconsistent with the requirements for a high throughput online transaction processing environment. You can disable the editor table locking option, by Db2® subsystem. This is achieved by setting the HFM2SSDM macro parameters:
  - TABLE_LOCKING to NO. See TABLE_LOCKING on page 443.
  - PROD_EDIT to YES. See PROD_EDIT on page 439. Note that setting PROD_EDIT=YES will over-ride TABLE_LOCKING=YES, if specified, replacing it with TABLE_LOCKING=NO. The HFM2SSDM macro is discussed further in Defining all Db2 systems that ZDT/Db2 will access in HFM2POPT (required) on page 174.
- User specified SELECT statements and ZDT/Db2 editor choice

  Various ZDT/Db2 functions allow users to create and execute SELECT statements. The ZDT/Db2 editor is used to display the result table for these statements. There is a system option controlling whether the result table is displayed in browse, view or edit modes.

  ZDT/Db2 functions that process SELECT statements include the following:

  Prototype SELECT statements (basic)

Prototype SELECT statements (advanced)
Enter, execute and explain SQL statements
Edit and execute SQL statements from a data set

You can disable the use of the ZDT/Db2 editor, in edit mode, when processing the result table for user specified SELECT statements, by Db2® system. This is achieved by setting the HFM2SSDM macro parameter USER_SELECT_EDIT to NO. See USER_SELECT_EDIT on page 445.

The HFM2SSDM macro is discussed further in Defining all Db2 systems that ZDT/Db2 will access in HFM2POPT (required) on page 174.

- The use of Cursor concurrency options

The ZDT/Db2 editor has an option that allows a *"WITH"* clause to be added to the declaration for the cursor used to access Db2® data. See the description for *"isolation-clause"* in the SQL Reference manual for the appropriate version of Db2®.

The default for this option is to not add any WITH clause, which results in Db2® taking locks based on system options, and options specified for the table spaces that contain the Db2® object. The following editor options are also relevant:

Read-only access
Commit after data fetch

The *"Uncommitted read"* concurrency option can be specified when you need to avoid any locks being taken by ZDT/Db2 users when they read data prior to display in an editor session. You can force the use of *"uncommitted read"*, by Db2® system. This is achieved by setting the HFM2SSDM macro parameter FORCE_WITH_UR to Y. See FORCE_WITH_UR on page 436.

The HFM2SSDM macro is discussed further in Defining all Db2 systems that ZDT/Db2 will access in HFM2POPT (required) on page 174.

- The PROD_EDIT parameter

The HFM2SSDM macro parameter PROD_EDIT has multiple effects on the behavior of the Z Data Tools Db2® editor, as described below:

  - An inactivity timer applies to all ZDT/Db2 editor sessions. If EDITOR_TIMEOUT=0 is specified in the same HFM2SSDM macro, this value is replaced with EDITOR_TIMEOUT=60.
  - Table locking is disabled. If TABLE_LOCKING=YES is specified in the same HFM2SSDM macro, this value is replaced with TABLE_LOCKING=NO. When a ZDT/Db2 user displays the Editor Locking option, the value is set to '1' (None) and cannot be altered. This prevents the user changing the option to either SHARED or EXCLUSIVE.
  - *"Large"* edit mode is disabled. If EDIT_MAX_ROWS=0 is specified in the same HFM2SSDM macro, this value is replaced with EDIT_MAX_ROWS=1000. When a ZDT/Db2 user enters 0 in the row count field on an editor-

related function entry panel, the value is ignored and replaced with the value specified for EDIT_MAX_ROWS. This prevents the user accessing the ZDT/Db2 editor in *"large"* mode.
    - The *"Commit after data fetch"* option is fixed and is selected.
    - The *"Commit when save issued"* option is fixed and is selected.
    - The *"Commit when no save errors"* option is fixed and is not selected.

The *"Commit after data fetch"* option ensures that any user of the ZDT/Db2 editor has no locks against Db2® when the data in the editor session is initially displayed, and the user cannot change the option. The other two options ensure that a Db2® commit is always issued when a user attempts to save changes made in the current editor session.

The commit is always issued, whether there are save errors or not. When considered in combination, the PROD_EDIT behavior reduces locking against the Db2® object being edited to a minimum. The only time locks are held is when:

- The data is initially accessed, prior to the display of the data.
- Whenever changes made in the editor session are presented to Db2® for validation.

- Restricting the use of the ZDT/Db2 editor and educating users

Setting the HFM2SSDM parameters as described in above reduces the potential for unwanted locking or resource constraints arising from the use of the ZDT/Db2 editor, it does not eliminate it.

The ZDT/Db2 editor is a powerful tool. You should consider carefully which users, if any, have access to the ZDT/Db2 editor in production Db2® environments.

You should consider educating any users of the ZDT/Db2 editor in production environments. The main issues to be considered are:

- Minimizing the number of rows of data retrieved for any editor session. The fewer rows that are retrieved, the smaller the number of locks that might be taken against production data bases.
- Minimizing the duration of any editor sessions against production data. The Db2® *"unit of work"* concept is relevant here, it is recommended that any users of the ZDT/Db2 editor against production data are familiar with the concept, and potential problems that might arise when the elapsed time for a unit of work becomes long.

The following example demonstrates the potential problem.

The user starts a Z Data Tools/Db2 editor session of a production table. They have the editor option *"uncommitted read"* on and use row selection criteria to limit the number of rows in the editor session to 1000.

The user issues a change command that affects 800 rows. The user issues the SAVE command to validate the changes made. The editor option *"Commit when save issued"* is turned off. The changes are successful: all of the 800 rows that were changed are updated in Db2®.

The user is called away, goes to lunch or finishes work for the day, leaving the editor session active.

In this scenario the ZDT/Db2 editor session starts with no locks against the production database. When the SAVE command is issued, the 800 rows that are changed are marked in Db2® with update locks. The editor option *"Commit*

when save issued" option is off, so there is no COMMIT issued and the Db2® unit of work remains active. The Db2® locks against the 800 rows remain active until the Db2® unit of work ends, which occurs when the user cancels the editor session, the user ends the editor session, or the user's TSO session is cancelled. This might be hours, even days, in the scenario outlined above. During that time other users of the production database may not be able to access those rows.

The issues outlined in this scenario can be avoided if PROD_EDIT=YES is specified.

Users of the ZDT/Db2 editor in production environments should be educated to minimize the elapsed time for any editor sessions that make changes to production data. This minimizes the duration of any locks taken against production data.

Chapter 17. Customizing the audit facility for ZDT/Db2

ZDT/Db2 can write audit log records to either SMF or an audit log data set.

If auditing is not required, you should:

- Set AUDIT=NONE in the HFM2SSDM macro specification of the HFM2POPT module. There is one HFM2SSDM specification for each Db2® system accessed by ZDT/Db2. Specify AUDIT=NONE for each HFM2SSDM specification where auditing is not required.

  See AUDIT on page 427 for more information.
- Set SMFNO=0 in the HFM0POPI macro specification of the HFM2POPT module. See SMFNO on page 416 for more information.

You can skip the customization described in the rest of this chapter.

Z Data Tools provides two different methods (HFM2POPT-controlled and SAF-controlled audit) for controlling whether audit records are written for ZDT/Db2. These are described in detail in Alternatives for controlling ZDT/Db2 auditing on page 157.

Use the checklist to determine the customization required for Z Data Tools Db2® audit facility.

Table 27. Checklist for customizing ZDT/Db2 auditing. This table lists choices and decisions.

Audit customization choice	Decision (Yes|No|Not applicable)
1. Control auditing using the HFM2POPT options module	
2. Control auditing using SAF rules and a member in SYS1.PARMLIB	
3. Control auditing using SAF rules, without any changes to SYS1.PARMLIB	
4. Audit records are to be written to a data set	
5. Audit records are to be written to SMF	

For choices 1 to 3, you should answer YES for one choice only. Mark the other two choices as 'Not applicable'.

If you answer YES for choice 1, you can answer YES for one of choices 4 and 5. Mark the other choice as 'Not applicable'.

If you answer YES for choices 2 or 3, you can answer YES for one or both of choices 4 and 5. If you answer YES for both choices you are implementing dual-logging, which is only available with SAF-controlled auditing.

Once you have completed the checklist, use the table below to identify the customization that is required for each customization choice. Only complete the customization if your decision was *"YES"* in the checklist.

Table 28. Customization steps for audit customization choices. This table lists choices and related actions.

Customization choice	Sections to complete
1. Control auditing using the HFM2POPT options module	• HFM2POPT-controlled auditing on page 190
2. Control auditing using SAF rules and a member in SYS1.PARMLIB	• SAF-controlled auditing for Z Data Tools Db2 component on page 192 • Implementing SAF-rule controlled auditing on page 195
3. Control auditing using SAF rules, without any changes to SYS1.PARMLIB	• SAF-controlled auditing for Z Data Tools Db2 component on page 192 • Implementing SAF-rule controlled auditing on page 195
4. Audit records are to be written to a data set	• Audit data set configuration on page 191
5. Audit records are to be written to SMF	• Customizing Z Data Tools to write audit records to SMF on page 86

HFM2POPT-controlled auditing

Use the AUDIT option in HFM2SSDM to customize how ZDT/Db2 records an audit trail.

- No audit trail produced under any circumstances. Specify AUDIT=NONE.
- Optional for the current ZDT/Db2 session. An audit trail is produced if the individual user wants to record audit information, but the audit option is reset to "not selected" when a Z Data Tools/Db2 session ends. Specify AUDIT=(OPTIONAL,OFF).
- Optional for this and subsequent ZDT/Db2 sessions. An audit trail is produced if the individual user wants to record audit information. The current setting of the audit option is saved in the ISPF profile between ZDT/Db2 sessions. Specify AUDIT=(OPTIONAL,PROFILE).
- Required. An audit trail is produced for each user, regardless of whether the user wants to record audit information. Specify AUDIT=(REQUIRED,....).
- Demand. An audit trail is produced for each user, regardless of whether the user wants to record audit information. An audit report job is submitted when the edit function ends or when you change SSIDs. Specify AUDIT=(DEMAND,....).

  The job submitted is determined by the skeleton member HFM.SHFMSLIB(HFM2FTAD). Customize the job card and JCL to specify the reporting options you require. To change the audit report options in the skeleton see "AUD (Print Audit Trail Report)" in the *Z Data Tools User's Guide and Reference*.

When recording of audit information is optional, this is determined by the individual setting of Create an audit trail in the user's Editor global options.

For optional auditing, AUDIT=(OPTIONAL,OFF) is suggested as the best option when audit data is written to data sets, rather than SMF, and when auditing is an exceptional, rather than a normal process.

Note: The AUDIT option is active only when data is modified via the editor (view, browse or edit) or copy utility. The setting of this option is ignored for any other Z Data Tools function that allows data to be modified, and for data modified by the ISPF editor, even though this is from within Z Data Tools.

Auditing can be selectively disabled when using ZDT/Db2 view or browse by specifying AUDITBROWSE=N in the HFM2SSDM macro definition for a Db2® system. Specifying this option does not affect the audit records produced when using ZDT/Db2 edit. See AUDITBROWSE on page 428 for information about AUDITBROWSE.

Audit data set configuration

The format of the audit log data set name is determined by the setting of the AUDITHLQ parameter in the HFM0POPI definition in HFM2POPT.

The following data set name formats may be generated:

- `userid.HFM2AUD.<ssid>.Dyymmdd.Thhmmss` (when AUDITHLQ= (blank))
- *`audithlq`*`.HFM2AUD.<ssid>.Dyymmdd.Thhmmss` (when AUDITHLQ=*audithlq*)
- `qual1.<qual2.><qual3.>Dyymmdd.Thhmmss` (when AUDITHLQ=*qual1.<qual2.><qual3>*)

where:

audithlq

Any 1-8 character constant that is valid in the context of a data set name.

ssid

The name of the currently connected (local) Db2® system.

userid

The user ID creating the data set.

Dyymmdd

The date of the activity.

Thhmmss

The time of the activity.

When AUDITHLQ contains one or more periods, the AUDITHLQ value is treated as a data set prefix, with one, two or three levels. Each level of the prefix can be:

XXX

Any 1-8 character constant that is valid in the context of a data set name.

&&PREFIX

Indicates that the user's TSO prefix should by used. This is null if TSO NOPREFIX is in effect and, after substitution, the appropriate level of the audit log data set name prefix is also null.

&&USER

Indicates that the user's logonid (ISPF system variable ZUSER, stored in the shared pool) should be used.

&&UID

Indicates that the user's TSO prefix should be used, when the value is non-blank. When TSO NOPREFIX is in effect, the user's TSO logonid (ISPF system variable ZUSER, stored in the shared pool) should be used.

&&FUNCOD

Indicates that the Z Data Tools internal function code should be used. Specifying this parameter allows the Z Data Tools function that generated the audit log data set to be included in the audit log data set name.

&&SSID

Indicates that the currently connected (local) Db2® subsystem name should be used. This data set can be printed using the ZDT/Db2 Print Audit Trail utility, (option 3.10 from the ZDT/Db2 Primary Option menu).

In this case, audit data sets are produced for each user, and are subject to the usual rules controlling user data sets. For this reason, if you intend that audit logging should always take place and be reported on, it is suggested that you select `AUDIT=(REQUIRED,SMF)` in your HFM2SSDM macro.

Set the AUDITHLQ parameter in the HFM0POPI macro to the required value, based on your site's requirements. See AUDITHLQ on page 386 for more information and examples.

You can print the information in an audit data set using the ZDT/Db2 Print Audit Trail utility. To do this select option 3.10 from the ZDT/Db2 Primary Option menu.

SAF-controlled auditing for Z Data Tools Db2® component

There are two methods for implementing SAF-controlled auditing for ZDT/Db2. These are:

1. Using an enabling SAF Facility class rule and a member in SYS1.PARMLIB.

   To use this method complete the customization described in SAF-controlled auditing using SYS1.PARMLIB on page 193.

2. Using an enabling SAF Facility class rule, without any changes to SYS1.PARMLIB.

   To use this method complete the customization described in SAF-controlled auditing without SYS1.PARMLIB on page 195.

Important: If you use a security product other than RACF®, review the information in When a security product other than RACF is in use on page 94 to avoid one possible cause of S047 abends.

SAF-controlled auditing using SYS1.PARMLIB

You need to define an enabling SAF facility profile as described below:

Define SAF facility profile

```
FILEM.PARMLIB.DB2
```

and ensure all Z Data Tools Db2® users to be audited have at least read access to that facility. See the example below:

Example

User PROD1 to have SAF-rule controlled auditing using SYS1.PARMLIB.

Write this RACF® rule:

```
RDEF FACILITY FILEM.PARMLIB.DB2 AUDIT(NONE) UACC(NONE) OWNER(ownerid)
PE FILEM.PARMLIB.DB2 ACC(READ) ID(PROD1) CLASS(FACILITY)
```

Add member HFM2PARM to SYS1.PARMLIB (or any other library in the logical parmlib concatenation). Defining the HFM2PARM member on page 193.

Once the above SAF rule is defined and activated, auditing for Z Data Tools Db2® component users is controlled by the FMAUDIT parameter in the HFM2PARM member. See ZDT/Db2 options specified in HFM2PARM on page 526 for more information. If audit log records are to be written to SMF, the SMF record number is specified as an FMAUDIT parameter option. See FMAUDIT on page 527, and SMF_NO on page 527.

Note: Z Data Tools Db2® component does not start if a user has read access to the above facility and the HFM2PARM member does not exist in the logical parmlib concatenation.

If SAF processing is not active, or the rule is not defined, or the rule is defined and the user has no access, then no parmlib processing is performed.

Defining the HFM2PARM member

If auditing is to be controlled from parmlib (user has read access to FILEM.PARMLIB.DB2, see SAF-controlled auditing for Z Data Tools Db2 component on page 192) then member HFM2PARM must be defined in SYS1.PARMLIB (or any other library in the logical parmlib concatenation) as follows.

Default parmlib member HFM2PARM is provided in the SHFMSAM1 library. Copy this member to the appropriate system parmlib library. See below for details of methods that can be used to make this change.

Note: The sample HFM2PARM member supplied in SHFMSAM1 also includes a FMSECRTY statement. This option is not used in ZDT/Db2 and can be either omitted or commented out. It has no effect.

There are two methods that can be used to include the HFM2PARM member in a library in the logical parmlib concatenation. The choice of method depends on whether the installation's security software is configured to allow ZDT/Db2 users READ access to data set SYS1.PARMLIB.

Method 1 can only be used when ZDT/Db2 users have read access to all libraries in the logical parmlib concatenation.

Method 2 can be used regardless of whether ZDT/Db2 users have READ access to the libraries in the logical parmlib concatenation.

Method 2 must be used when ZDT/Db2 users do not have READ access to one or more libraries in the logical parmlib concatenation.

Method 1

Place the HFM2PARM member in any library in the current logical parmlib concatenation. No IPL or other action is required to active the new member (unless a new library was added to the logical parmlib concatenation).

Note: HFM2POPT controlled auditing cannot be used in any situation where ZDT/Db2 users do not have READ access to all of the libraries in the logical parmlib concatenation.

For example, when:

- There are six libraries in the logical parmlib concatenation, for simplicity: libraries A, B, C, D, E and F.
- ZDT/Db2 users have read access to five of these libraries: A, B, D, E, F.
- Library C may be SYS1.PARMLIB, or any other library in the logical parmlib concatenation.

This will not work, the attempt by a Z Data Tools/Db2 user to access the logical parmlib concatenation will fail with a security-related (913) abend.

Method 2

This method must be used when ZDT/Db2 users do not have READ access to all of the libraries in the logical parmlib concatenation.

1. Create a new library with dataset attributes similar to SYS1.PARMLIB.

   The library name for this data set must include the string "HFMPARM" in one of the qualifiers. You can choose any data set name that meets this requirement. Examples of suitable data set names are:

   SYS1.PARMLIB.HFMPARM
   SYS8.HFMPARM.PARMLIB
   HFMPARM.SYS8.PARMLIB

SYS2.HFMPARMS.LIB
SYS8.XHFMPARM.PARMLIB

2. Add member HFM2PARM to the new library, specifying the appropriate FMAUDIT parameter.
3. Add the new library to the logical parmlib concatenation. This can be done dynamically or via a system IPL.

Note: When Method 2 is used, the HFM2PARM member must be located in the library created in step 1 on page 194. If the HFM2PARM member specifies any include statements (see Facilities for customizing the HFM2PARM definitions on page 528), all of the included members must also reside in the same library.

You use the HFM2PARM member to define the following:

- Whether Z Data Tools will use SAF to control Z Data Tools audit logging.
- The SAF resource name prefix to be used by Z Data Tools when determining access to various resources.
- Whether Z Data Tools should load the HFM2POPT module from a specific library.

See ZDT/Db2 options specified in HFM2PARM on page 526 for more information.

SAF-controlled auditing without SYS1.PARMLIB

You need to define an enabling SAF facility profile as described below:

Define SAF facility profile

```
FILEM.SAFAUDIT.DB2
```

and ensure that all ZDT/Db2 users to be audited have at least read access to that facility. See the example below.

Example

User PROD2 to have SAF-rule controlled auditing without using SYS1.PARMLIB.

Write this RACF® rule:

```
RDEF FACILITY FILEM.SAFAUDIT.DB2 AUDIT(NONE) UACC(NONE) OWNER(ownerid)
PE FILEM.SAFAUDIT.DB2 ACC(READ) ID(PROD2) CLASS(FACILITY)
```

If you use this method and intend to write audit records to SMF, the required SMF number is specified in the HFM2POPT module. See Customizing Z Data Tools to write audit records to SMF on page 86 for more information.

Implementing SAF-rule controlled auditing

Use the following checklist to implement SAF-rule controlled auditing:

1. Determine the SAF FACILITY and XFACILIT rules that will be required. See Understanding how ZDT/Db2 uses SAF rules to control auditing on page 196 for more information.
2. Write the relevant SAF rules. See the examples in SAF rule examples on page 203.
3. Activate SAF auditing for a specific logon using the chosen method of activating SAF-controlled auditing. See SAF-controlled auditing for Z Data Tools Db2 component on page 192.
4. Using the selected logon, test the configuration to ensure that auditing is occurring as required.
5. When testing is complete, activate SAF-controlled auditing for all ZDT/Db2 users.

Understanding how ZDT/Db2 uses SAF rules to control auditing

SAF (System Authorization Facility) allows applications, such as Z Data Tools, to define "resources" that might need to be protected. The "resource" to be protected need not be something specific, such as a data set; it can be essentially any type of resource or facility that the application considers to be important. For ZDT/Db2 and auditing, the "resource" is the ability to write audit log records. The resource names reflect either the type of auditing that is to occur (eg to SMF), or the type of Db2® object, SQL statement, or Db2® command that is being processed, for example, a Db2® object name.

ZDT/Db2 uses two types of SAF resource names to control auditing. Note that a user's ability to write audit log records under SAF control is independent of, for example, the user's ability to access a particular Db2® object, issue a particular Db2® command and so on. A user may be able to write audit records when using the ZDT/Db2 editor to look at a particular Db2® object, but lack the Db2® authority to actually look at the object.

The SAF resource rules used by ZDT/Db2 to control auditing are shown in Table 33: ZDT/Db2 auditing FACILITY class resource names on page 207 and Table 34: ZDT/Db2 auditing XFACILIT class resource names on page 207).

Understanding SAF rule access levels

SAF provides for five levels of access to any FACILITY or XFACILIT resource. The levels of access form a hierarchy, so that a user with the highest level of access to a resource also has access to all the lower levels. The levels of access are specified in RACF® rules using the following mnemonics:

NONE

No access

READ

Level 1 access

UPDATE

Level 2 access

CONTROL

Level 3 access

ALTER

Level 4 access.

It is important to understand that the mnemonics used (READ, ALTER and so on) can and do mean different things, depending on the context in which the SAF resource name is used. This can be confusing since READ and UPDATE have obvious meanings when it comes to, for example, accessing a data set. For SAF rules used to control ZDT/Db2 audit, it may aid understanding to think of the mnemonics as indicating level 1 access, level 2 access and so on.

For the SAF resource rules used by ZDT/Db2, the meanings of the various levels of access are:

NONE

The user does not have access to the resource; this typically means the user cannot write audit log records.

READ

The user has level 1 access to the resource; this typically means that the user can write audit log records.

UPDATE

The user has level 2 access to the resource. This level of access only has meaning for FACILITY rule 2 (see Table 33: ZDT/Db2 auditing FACILITY class resource names on page 207). A user with level 2 access can write audit log records to the user's audit log data set, and the audit log data set will be printed at the end of the user's session (online execution only). This is equivalent to the DEMAND audit option in the non-SAF case.

CONTROL

The user has level 3 access to the resource. This level of access only has meaning with the XFACILIT rules described in Table 34: ZDT/Db2 auditing XFACILIT class resource names on page 207. A user with level 3 access can change the auditing requirement specified in the current resource name. Note that, in order to do so, the user must also have at least level 1 access to FACILITY rule 3 (see Table 33: ZDT/Db2 auditing FACILITY class resource names on page 207).

ALTER

The user has level 4 access to the resource. This level of access is not used by ZDT/Db2.

How ZDT/Db2 determines whether audit log records should be written

The determination of whether audit records are to be written for a particular ZDT/Db2 function and a given TSO logonid follows this three step process:

1. Step 1.
   - If auditing is to be controlled by means of parmlib, the HFMAUDIT specification of the HFM2PARM member is used as follows.

     The FMAUDIT specification setting in the HFM2PARM member (in SYS1.PARMLIB or any other library in the logical parmlib concatenation) is the "main" switch for SAF-rule controlled auditing. Note that there are facilities available to specify different settings in the HFM2PARM member for different TSO logonids, see Z Data Tools options specified in PARMLIB members on page 515 for more information. For any given TSO logonid, there are two possibilities:

SAF_CTRL=NO

SAF-rule controlled auditing is not in effect. Auditing is determined by the settings in the HFM2POPT module, see Customizing the audit facility for ZDT/Db2 on page 189.

SAF_CTRL=YES

SAF-rule controlled auditing is in effect. Processing continues to Step 2.

- If auditing is being controlled using the method which does not access the logical parmlib concatenation, the TSO logonid has READ access to the SAF FACILITY rule FILEM.SAFAUDIT.DB2 for processing to continue to Step 2 on page 198.

2. Step 2.

   Does the user have access to write audit records?

   This is determined by the user's access to rules 1 and 2 in Table 33: ZDT/Db2 auditing FACILITY class resource names on page 207, the various outcomes are summarized in Table 29: Determination of a user's ability to write audit log records on page 198.

Table 29. Determination of a user's ability to write audit log records

TODSN access[23]	TOSMF access[24]	OPTION access[25]	Can write audit records?	Demand logging?	"Create audit trail" option[26]
NONE	NONE	ANY	No	No	Not visible
READ	NONE	NONE	Yes, data set only	No	Not visible
READ	NONE	READ	Yes, data set only	No	Visible
UPDATE	NONE	NONE	Yes, data set only	Yes	Not visible
UPDATE	NONE	READ	Yes, data set only	Yes	Visible
NONE	READ	NONE	Yes, SMF only	No	Not visible
NONE	READ	READ	Yes, SMF only	No	Visible
READ	READ	NONE	Yes, to data set and SMF	No	Not visible

23. Refers to the level of access the user has to SAF FACILITY rule 1 in Table 33: ZDT/Db2 auditing FACILITY class resource names on page 207.
24. Refers to the level of access the user has to SAF FACILITY rule 2 in Table 33: ZDT/Db2 auditing FACILITY class resource names on page 207.
25. Refers to the level of access the user has to SAF FACILITY rule 3 in Table 33: ZDT/Db2 auditing FACILITY class resource names on page 207.
26. The visibility of the "Create audit trail" option does not influence whether a user can write audit log records, although the user must have access to write audit log records (to either a data set or SMF), for the option to be visible.

TODSN access[23]	TOSMF access[24]	OPTION access[25]	Can write audit records?	Demand logging?	"Create audit trail" option[26]
READ	READ	READ	Yes, to data set and SMF	No	Visible
UPDATE	READ	NONE	Yes, to data set and SMF	Yes	Not visible
UPDATE	READ	READ	Yes, to data set and SMF	Yes	Visible

If the user does not have the ability to write audit log records, then no check of SAF resource names in Step 3 occurs.

A user's access to write audit log records at Step 2 only indicates that auditing might occur, the final decision depends on the user's level of access to the XFACILIT resource name (or names) that apply to the particular ZDT/Db2 function.

3. Step 3.

   Does the user have access to write audit records for the current function?

   The XFACILIT resource names used by ZDT/Db2 to determine whether audit records should be written depend on the ZDT/Db2 function being executed.

   The types of SQL statements and Db2® commands that might be issued by each ZDT/Db2 function are shown in Table 30: Types of SQL (and Db2) statements issued by ZDT/Db2 functions on page 199.

   The relationship between various SQL statements or Db2® commands and XFACILIT resource names is shown in Table 31: Relationship between SQL statement type and SAF resource names on page 201.

Table 30. Types of SQL (and Db2®) statements issued by ZDT/Db2 functions

ZDT/Db2 function	Option number	SQL/DB2 statements
Browse	B	SELECT
View	1	SELECT
Edit	2	SELECT, DELETE, INSERT, UPDATE
Print	3.1	SELECT
Db2® Objects	3.2	CREATE, DROP

23. Refers to the level of access the user has to SAF FACILITY rule 1 in Table 33: ZDT/Db2 auditing FACILITY class resource names on page 207.
24. Refers to the level of access the user has to SAF FACILITY rule 2 in Table 33: ZDT/Db2 auditing FACILITY class resource names on page 207.
25. Refers to the level of access the user has to SAF FACILITY rule 3 in Table 33: ZDT/Db2 auditing FACILITY class resource names on page 207.
26. The visibility of the "Create audit trail" option does not influence whether a user can write audit log records, although the user must have access to write audit log records (to either a data set or SMF), for the option to be visible.

Table 30. Types of SQL (and Db2®) statements issued by ZDT/Db2 functions (continued)

ZDT/Db2 function	Option number	SQL/DB2 statements
Copy	3.3	SELECT (source object) DELETE (target object) INSERT (target object) UPDATE (target object)
Object List	3.4	DROP, GRANT, REVOKE, FREE, BIND, REBIND
Object Privileges	3.5	GRANT, REVOKE
Import	3.6	DELETE, INSERT, UPDATE
Export	3.7	SELECT
Create	3.8	INSERT
Basic SELECT prototyping	4.1	SELECT (any editor mode), DELETE, INSERT and UPDATE (only when the Editor option for "Arbitrary SQL Select Statements" is set to edit).
Advanced SELECT prototyping	4.2	SELECT (any editor mode), DELETE, INSERT and UPDATE (only when the Editor option for "Arbitrary SQL Select Statements" is set to edit).
Enter, Execute and Explain SQL	4.3	For SELECT statements as per "Basic SELECT prototyping". Any other SQL statement that can be issued.
Edit/Execute SQL (Data Set)	4.4	For SELECT statements as per "Basic SELECT prototyping". Any other SQL statement that can be issued.
Db2® commands	6	-ACCESS -ALTER -ARCHIVE -CANCEL -DISPLAY -MODIFY -RECOVER -REFRESH -RESET -SET -START -STOP -TERM

Table 31. Relationship between SQL statement type and SAF resource names

SQL Statement Type	Audit resource name suffix [27]
ALTER	DDL.<object_type>[28]
COMMENT	OTHER.ADHOCSQL
COMMIT	OTHER.ADHOCSQL
CREATE	DDL.<object_type>[28]
DELETE	UPDATE.ADHOCSQL
DROP	DDL.<object_type>[28]
EXCHANGE	OTHER.ADHOCSQL
EXPLAIN	OTHER.ADHOCSQL
GRANT	AUTH.<auth_type>[29]
INSERT	UPDATE.ADHOCSQL
LABEL	OTHER.ADHOCSQL
LOCK	OTHER.ADHOCSQL
MERGE	OTHER.ADHOCSQL
REFRESH	OTHER.ADHOCSQL
RENAME	DDL.<object_type>[28]
REVOKE	AUTH.<auth_type>[29]
ROLLBACK	OTHER.ADHOCSQL
SELECT	READ.<object>[30] UPDATE.<object>[31] READ.ADHOCSQL[32] [33] UPDATE.ADHOCSQL[34] [33]

27. The prefix for all resource names in this table is FILEM.AUDIT.<ssid>, where ssid is the Db2® subsystem or group ID.
28. See Table 35: Resource name suffixes for Db2 object types (DDL SQL statements) on page 209.
29. See Table 36: Resource name suffixes for Db2 privileges (GRANT and REVOKE SQL statements) on page 209.
30. This resource name is used when processing a Db2® object via options B and 1.
31. This resource name is used when processing a Db2® object using option 2.
32. This resource name is used when processing a Db2® object (or objects) using a SELECT statement entered via options 4.1, 4.2, 4.3 or 4.4, when ZDT/Db2 browse or view is used to display the result table for the SELECT.

Table 31. Relationship between SQL statement type and SAF resource names

(continued)

SQL Statement Type	Audit resource name suffix [27]
SET	OTHER.ADHOCSQL
TRUNCATE	UPDATE.ADHOCSQL
UPDATE	UPDATE.ADHOCSQL

For some ZDT/Db2 functions only a single SAF XFACILIT rule needs to be checked to determine whether audit log records should be written. An example is the ZDT/Db2 editor, which processes a single Db2® object in either READ or UPDATE modes.

For other ZDT/Db2 functions multiple SAF XFACILIT rules may be checked.

Example 1: ZDT/Db2 Copy utility

- The access to write audit records for READ access to the source Db2® object is checked.
- The access to write audit records for UPDATE access to the target Db2® object is checked.

Example 2: ZDT/Db2 Edit/Execute SQL (Data Set) utility

This utility allows the execution of SQL statements coded in a data set. ZDT/Db2 checks the access to write audit records for each SQL statement in the data set as it is executed and audit records are written (or not) as appropriate.

Controlling where ZDT/Db2 writes audit log records

You can use SAF to control whether ZDT/Db2 writes audit log records to SMF, the user's audit log data set, or to both.

The following table shows the SAF FACILITY class resource names used to control ZDT/Db2 logging to SMF and the user's audit log data set.

Table 32. SAF FACILITY class resource names controlling disposition of ZDT/Db2 audit records

FACILITY Class Name	Purpose
FILEM.AUDIT2.<ssid>.TOSMF	Enables or disables auditing to SMF for ZDT/Db2.
FILEM.AUDIT2.<ssid>.TODSN	Enables or disables auditing to the user's data set for ZDT/Db2.

27. The prefix for all resource names in this table is FILEM.AUDIT.<ssid>, where ssid is the Db2® subsystem or group ID.
33. The choice of ZDT/Db2 editor mode (browse, view or edit) is determined by the setting of the "Arbitrary SQL Select Statements", Editor option, which can be found on the second ZDT/Db2 system options panel. This is accessed by typing 0.0.2 from the ZDT/Db2 main menu.
34. This resource name is used when processing a Db2® object (or objects) using a SELECT statement entered via options 4.1, 4.2, 4.3 or 4.4, when ZDT/Db2 edit is used to display the result table for the SELECT.

See the examples in Controlling where ZDT/Db2 writes audit log records on page 203.

SAF rule examples

This section shows SAF rule examples under different conditions.

Controlling where ZDT/Db2 writes audit log records

You can use SAF to control whether ZDT/Db2 writes audit log records to SMF, the user's audit log data set, or to both.

Table 33: ZDT/Db2 auditing FACILITY class resource names on page 207 shows the SAF FACILITY class resource names used to control ZDT/Db2 to logging and the user's audit log data set.

Example 1

- Disable audit logging to a user data set for all ZDT/Db2 users, for Db2® system DSNA.
- Enable ZDT/Db2 audit logging to SMF for the PROD logonid, for Db2® system DSNA.

You could write the following RACF® rules:

```
RDEL FACILITY FILEM.AUDIT2.DSNA.TOSMF[35]
RDEL FACILITY FILEM.AUDIT2.DSNA.TODSN[35]

RDEF FACILITY FILEM.AUDIT2.DSNA.TOSMF UACC(NONE) OWNER(XXXXXXX)[36]
RDEF FACILITY FILEM.AUDIT2.DSNA.TODSN UACC(NONE) OWNER(XXXXXXX)[37]

PE FILEM.AUDIT2.DSNA.TOSMF ACC(READ) ID(PROD) CLASS(FACILITY)[38]
```

Example 2

- Enable audit logging to a user data set for all ZDT/Db2 users, for Db2® system DSNB.
- Enable demand logging for the following users, PROD1, PROD2, PROD3
- Disable audit logging to SMF for all ZDT/Db2 users, for Db2® system DSNB.

You could write the following RACF® rules:

```
RDEL FACILITY FILEM.AUDIT2.DSNB.TOSMF [35]
RDEL FACILITY FILEM.AUDIT2.DSNB.TODSN [35]

RDEF FACILITY FILEM.AUDIT2.DSNB.TOSMF UACC(NONE) OWNER(XXXXXXX)[39]
RDEF FACILITY FILEM.AUDIT2.DSNB.TODSN UACC(READ) OWNER(XXXXXXX)[40]
```

35. Delete any existing facility rule
36. Define the facility rule for Db2® system DSNA and audit logging to SMF (TOSMF suffix). UACC(NONE) is used so that any user, for which there is no specific rule, has no access.
37. Define the facility rule for Db2® system DSNA and audit logging to the user's audit log data set (TODSN suffix). UACC(NONE) is used so that any user, for which there is no specific rule, has no access.
38. Allow logonid PROD to write audit log records (ACC(READ)), to SMF.
39. Define the facility rule for Db2® system DSNB and audit logging to SMF (TOSMF suffix). UACC(NONE) is used so that any user, for which there is no specific rule, has no access.
40. Define the facility rule for Db2® system DSNB and audit logging to the user's audit log data set (TODSN suffix). UACC(READ) is used so that any user, for which there is no specific rule, has read access, and can therefore write audit log records.

```
PE FILEM.AUDIT2.DSNB.TODSN ACC(UPDATE) ID(PROD1) CLASS(FACILITY)[41]
PE FILEM.AUDIT2.DSNB.TODSN ACC(UPDATE) ID(PROD2) CLASS(FACILITY)[41]
PE FILEM.AUDIT2.DSNB.TODSN ACC(UPDATE) ID(PROD3) CLASS(FACILITY)[41]
```

Example 3

- Disable audit logging completely for all ZDT/Db2 users, for Db2® system DSND.
- Enable dual logging for all ZDT/Db2 users for Db2® system DSNP.

You could write the following RACF® rules:

```
RDEL FACILITY FILEM.AUDIT2.DSND.TOSMF [35]
RDEL FACILITY FILEM.AUDIT2.DSND.TODSN [35]
RDEL FACILITY FILEM.AUDIT2.DSNP.TOSMF [35]
RDEL FACILITY FILEM.AUDIT2.DSNP.TODSN [35]

RDEF FACILITY FILEM.AUDIT2.DSND.TOSMF UACC(NONE) OWNER(XXXXXXX)[42]
RDEF FACILITY FILEM.AUDIT2.DSND.TODSN UACC(NONE) OWNER(XXXXXXX)[43]
RDEF FACILITY FILEM.AUDIT2.DSNP.TOSMF UACC(READ) OWNER(XXXXXXX)[44]
RDEF FACILITY FILEM.AUDIT2.DSNP.TODSN UACC(READ) OWNER(XXXXXXX)[45]
```

Controlling auditing of update access to Db2® objects

You can use SAF to control whether ZDT/Db2 writes audit log records for ZDT/Db2 functions that update, or have the potential to update, Db2® objects. Examples of such functions are:

- The ZDT/Db2 editor operating in edit mode (but not view or browse)
- ZDT/Db2 Copy, for the target Db2® object
- ZDT/Db2 Import, for the target Db2® object
- ZDT/Db2 Data Create

Table 34: ZDT/Db2 auditing XFACILIT class resource names on page 207 shows the SAF XFACILIT class resource names used to control ZDT/Db2 audit logging.

Example 1

- Enable audit logging for update access to DSN8910.EMP in Db2® system DSNA, for all users except TSO logonid MAINT1.

41. Allow logonids PROD1, PROD2, PROD3 to write audit log records with automatic printing of the audit report ("Demand logging") (ACC(UPDATE)), to SMF.
42. Define the facility rule for Db2® system DSND and audit logging to SMF (TOSMF suffix). UACC(NONE) is used so that any user, for which there is no specific rule, has no access.
43. Define the facility rule for Db2® system DSND and audit logging to the user's audit log data set (TODSN suffix). UACC(NONE) is used so that any user, for which there is no specific rule, has no access.
44. Define the facility rule for Db2® system DSNP and audit logging to SMF (TOSMF suffix). UACC(READ) is used so that any user, for which there is no specific rule, has access (and can therefore write audit records to SMF).
45. Define the facility rule for Db2® system DSNP and audit logging to the user's audit log data set (TODSN suffix). UACC(READ) is used so that any user, for which there is no specific rule, has access (and can therefore write audit records to the user's audit log data set).

You could write the following RACF® rules:

```
RDEL XFACILIT FILEM.AUDIT.DSNA.UPDATE.OBJ.DSN8910.EMP[46]

RDEF XFACILIT FILEM.AUDIT.DSNA.UPDATE.OBJ.DSN8910.EMP +
     OWNER(XXXXXXXX) UACC(READ)[47]

PE FILEM.AUDIT.DSNA.UPDATE.OBJ.DSN8910.EMP +
     CLASS(XFACILIT) ID(MAINT1) ACC(NONE)[48]
```

Example 2

- Enable audit logging for update access to remote object DSN8910.EMP. This object is accessed from Db2® system DSNA. The location of the remote Db2® system is TEXAS. Audit logging is to be performed for all users except SERVIC1, SERVIC2.

You could write the following RACF® rules:

```
RDEL XFACILIT FILEM.AUDIT.DSNA.UPDATE.REMOBJ.TEXAS.DSN8910.EMP[46]

RDEF XFACILIT FILEM.AUDIT.DSNA.UPDATE.REMOBJ.TEXAS.DSN8910.EMP +
     OWNER(XXXXXXXX) UACC(READ)[49]

PE FILEM.AUDIT.DSNA.UPDATE.REMOBJ.TEXAS.DSN8910.EMP +
     CLASS(XFACILIT) ID(SERVIC1) ACC(NONE)[50]
PE FILEM.AUDIT.DSNA.UPDATE.REMOBJ.TEXAS.DSN8910.EMP +
     CLASS(XFACILIT) ID(SERVIC2) ACC(NONE)[50]
```

Controlling auditing of read access to Db2® objects

You can use SAF to control whether ZDT/Db2 writes audit log records for ZDT/Db2 functions that read data from Db2® objects. Examples of such functions are:

- The ZDT/Db2 editor operating in view or browse modes (but not edit mode)
- ZDT/Db2 print.
- ZDT/Db2 Copy, for the source Db2® object
- ZDT/Db2 Export

Table 34: ZDT/Db2 auditing XFACILIT class resource names on page 207 shows the SAF XFACILIT class resource names used to control ZDT/Db2 audit logging.

Example 1

46. Delete any existing XFACILIT rule
47. Define the XFACILIT rule for Db2® system DSNA and UPDATE access to Db2® object (OBJ) DSN8910.EMP. UACC(READ) allows all TSO user IDs to write audit log records (in the absence of any over-riding more specific rule).
48. A specific rule for logonid MAINT1 to prevent audit log records being written.
49. Define the XFACILIT rule for Db2® system DSNA and UPDATE access to remote Db2® object (REMOBJ) TEXAS.DSN8910.EMP. UACC(READ) allows all TSO user IDs to write audit log records (in the absence of any over-riding more specific rule).
50. Specific rules for logonids SERVIC1, SERVIC2 to prevent audit log records being written.

- Enable audit logging for read access to DSN8910.DEPT in Db2® system DSNC, for all users except TSO logonid MAIN1.

You could write the following RACF® rules:

```
RDEL XFACILIT FILEM.AUDIT.DSNC.READ.OBJ.DSN8910.DEPT[51]

RDEF XFACILIT FILEM.AUDIT.DSNC.READ.OBJ.DSN8910.DEPT +
     OWNER(XXXXXXXX) UACC(READ)[52]

PE FILEM.AUDIT.DSNC.READ.OBJ.DSN8910.DEPT +
     CLASS(XFACILIT) ID(MAIN1) ACC(NONE)[53]
```

Example 2

- Enable audit logging for update access to remote object DSN8910.ACT. This object is accessed from Db2® system DSNP. The location of the remote Db2® system is MONTANA. Audit logging is to be performed for all users except DEV1.

You could write the following RACF® rules:

```
RDEL XFACILIT FILEM.AUDIT.DSNP.READ.REMOBJ.MONTANA.DSN8910.ACT[51]

RDEF XFACILIT FILEM.AUDIT.DSNP.READ.REMOBJ.MONTANA.DSN8910.ACT +
     OWNER(XXXXXXXX) UACC(READ)[54]

PE FILEM.AUDIT.DSNP.READ.REMOBJ.MONTANA.DSN8910.ACT +
     CLASS(XFACILIT) ID(DEV1) ACC(NONE)[55]
```

Testing SAF-rule controlled auditing

You can test your SAF rules achieve the desired result using the following method:

1. Select a "test" TSO logonid, for example "TEST1".
2. Write the appropriate SAF FACILITY and XFACILIT rules for this logonid.
3. Activate SAF auditing for logonid TEST1 using the chosen method of activating SAF-controlled auditing. See SAF-controlled auditing for Z Data Tools Db2 component on page 192.
4. Logon using TEST1.

   From the ZDT/Db2 main menu select the Help pull-down menu, option 8 (About Db2®). This should show:

   ```
   Auditing  . . . . . : SAF-RULE CONTROLLED
   ```

   If it does not then TEST1 does not have access to the TOSMF or TODSN FACILITY rules.

51. Delete any existing XFACILIT rule
52. Define the XFACILIT rule for Db2® system DSNC and READ access to Db2® object (OBJ) DSN8910.EMP. UACC(READ) allows all TSO user IDs to write audit log records (in the absence of any over-riding more specific rule).
53. A specific rule for logonid MAIN1 to prevent audit log records being written.
54. Define the XFACILIT rule for Db2® system DSNP and READ access to remote Db2® object (REMOBJ) MONTANA.DSN8910.ACR. UACC(READ) allows all TSO user IDs to write audit log records (in the absence of any over-riding more specific rule).
55. Specific rule for logonid DEV1 to prevent audit log records being written.

5. Test individual ZDT/Db2 functions to confirm that audit log records are written (or not) as specified in the appropriate SAF rules.
6. When testing is complete, activate auditing for all users using your chosen method.

ZDT/Db2 auditing FACILITY and XFACILIT class resource names

These two tables (and associated tables) list FACILITY and XFACILIT class resource names and details.

Table 33. ZDT/Db2 auditing FACILITY class resource names

Rule Number	Resource Name[56]	Purpose
1	<pfx>.TODSN	Allows a user to write audit log records to the user's audit log data set.
2	<pfx>.TOSMF	Allows a user to write audit log records to SMF.
3	<pfx>.OPTION	Allows the user access to the "Create audit trail" option on selected ZDT/Db2 panels.

Table 34. ZDT/Db2 auditing XFACILIT class resource names

Rule Number	Resource Name suffix[57]	Purpose
1	<pfx>.READ.OBJ.<object>	Allows a user to write audit log records for functions that read data from the specified local object (object) in the specified Db2® system (ssid).
2	<pfx>.UPDATE.OBJ.<object>	Allows a user to write audit log records for functions that change data from the specified local object (object) in the specified Db2® system (ssid).
3	<pfx>.READ.REMOBJ.<object>	Allows a user to write audit log records for functions that read data from the specified remote object (object), when accessed from the specified Db2® system (ssid).

56. The prefix <pfx> for all resource names in this table is FILEM.AUDIT2.<ssid>, where ssid is the Db2® subsystem or group ID.
57. The prefix <pfx> for all resource names in this table is FILEM.AUDIT.<ssid>, where ssid is the Db2® subsystem or group ID.

Table 34. ZDT/Db2 auditing XFACILIT class resource names

(continued)

Rule Number	Resource Name suffix[57]	Purpose
4	<pfx>.UPDATE.REMOBJ.<object>	Allows a user to write audit log records for functions that update data from the specified remote object (object), when accessed from the specified Db2® system (ssid).
5	<pfx>.READ.ADHOCSQL	Allows a user to write audit log records for functions that read data from some result table in the specified Db2® system (ssid).
6	<pfx>.UPDATE.ADHOCSQL	Allows a user to write audit log records for functions that update data for some result table in the specified Db2® system (ssid). Alternatively, allows a user to write audit log records for SQL statements that might update data in the specified Db2® system (ssid).
7	<pfx>.OTHER.ADHOCSQL	Allows a user to write audit log records for functions that issue SQL statements that are not covered by the READ or UPDATE ADHOCSQL rules, or the DDL and AUTH rules, in the specified Db2® system (ssid).
8	<pfx>.DDL.<objecttype>[58]	Allows a user to write audit log records for functions that issue DDL statements (such as CREATE, DROP, ALTER and RENAME) in the specified Db2® system (ssid). The type of Db2® object is specified using the <object type> suffix.
9	<pfx>.AUTH.<privilege type>[59]	Allows a user to write audit log records for functions that issue SQL statements (such as GRANT, REVOKE) that explicitly alter Db2® privileges in the specified Db2® system (ssid). The type of Db2® privilege is specified using the <privilege type> suffix.
10	<pfx>.DB2CMD.<command type>[60]	Allows a user to write audit log records for functions that issue Db2® commands in the specified Db2® system (ssid).

57. The prefix <pfx> for all resource names in this table is FILEM.AUDIT.<ssid>, where ssid is the Db2® subsystem or group ID.
58. See Table 35: Resource name suffixes for Db2 object types (DDL SQL statements) on page 209.
59. See Table 36: Resource name suffixes for Db2 privileges (GRANT and REVOKE SQL statements) on page 209.
60. See Table 37: Resource name suffixes for Db2 commands on page 210.

Table 35. Resource name suffixes for Db2® object types (DDL SQL statements)

Db2® Object Type	Resource Rule Name suffix
ALIAS	ALIAS
AUXILIARY TABLE	AUXTABLE
DATABASE	DATABASE
FUNCTION	FUNCTION
GLOBAL TEMPORARY TABLE	GBLTABLE
INDEX	INDEX
PROCEDURE	PROC
ROLE	ROLE
SEQUENCE	SEQUENCE
STOGROUP	STOGROUP
SYNONYM	SYNONYM
TABLE	TABLE
TABLESPACE	TBSPACE
TRIGGER	TRIGGER
TRUSTED CONTEXT	CONTEXT
TYPE	TYPE
VIEW	VIEW

Table 36. Resource name suffixes for Db2® privileges (GRANT and REVOKE SQL statements)

Db2® Authorization Type	Resource Rule Name suffix
COLLECTION	COLLECT
DATABASE	DATABASE
TYPE	TYPE
JAR	JAR
FUNCTION	FUNCTION
PACKAGE	PACKAGE
PLAN	PLAN
PROCEDURE	FUNCTION

Table 36. Resource name suffixes for Db2® privileges (GRANT and REVOKE SQL statements) (continued)

Db2® Authorization Type	Resource Rule Name suffix
SCHEMA	SCHEMA
SEQUENCE	SEQUENCE
SYSTEM	SYSTEM
TABLE	TABLE
USE	USE

Table 37. Resource name suffixes for Db2® commands

Db2® Command	Resource Rule Name suffix
ACCESS	ACCESS
ALTER	ALTER
ARCHIVE	ARCHIVE
BIND	BIND
CANCEL	CANCEL
DISPLAY	DISPLAY
FREE	FREE
MODIFY	MODIFY
REBIND	REBIND
RECOVER	RECOVER
REFRESH	REFRESH
RESET	RESET
RUN	RUN
SET	SET
START	START
STOP	STOP
TERM	TERM

Chapter 18. Customizing ZDT/Db2 for national languages

You can customize ZDT/Db2 for national languages other than English.

If you are using Japanese and you have installed the ZDT/Db2 Japanese component, you might not need to perform any other customization for the Japanese national languages.

If you are using a language other than English or Japanese, you will need to perform the customization tasks listed in Table 38: Summary of steps for customizing ZDT/Db2 for a national language on page 211. Perform **ALL** these steps (except Step 9).

You can also, optionally, translate the object list utility panel titles, column headings, and SQL statements. If you want to do this, perform step 9.

Table 38. Summary of steps for customizing ZDT/Db2 for a national language

Step	Description
__ 1	Set the LANGUAGE option for batch processing, if required. See Setting the default national language on page 181.
__ 2	Change the TERMTYPE option in HFM2POPT, if you are using a DBCS language. See TERMTYPE on page 418.
__ 3	Confirm that you have the correct terminal type set in ISPF (ISPF option 0).
__ 4	Create a print and display translation table for your language. See Changing the print and display translation tables for languages other than English on page 211.
__ 5	Translate the Z Data Tools message text to your language. See Translating the message text on page 212.
__ 6	Provide a version of HFM2MENU for your language. See Providing a multicultural version of HFM2MENU on page 212.
__ 7	Translate the ZDT/Db2 ISPF messages to your language. See Translating the ISPF messages text on page 213.
__ 8	Translate the ZDT/Db2 panels to your language. See Translating the panel text on page 214.
__ 9	Translate the ZDT/Db2 object list utility headings and statements (HFM2DENU) to your language. See Providing a multicultural version of HFM2DENU on page 215.

Changing the print and display translation tables for languages other than English

If you plan to use ZDT/Db2 with a national language other than English, you might need to provide a print and display translation table for your language.

You do this as part of your customization for Z Data Tools Base function. See Changing the print and display translation tables for languages other than English on page 104.

You should also specify PRTTRANS=ON in HFM2POPT. If you are using any DBCS language you might also need to specify TERMTYPE=3270KN in HFM2POPT.

This step is essential if you are using a DBCS language other than Japanese.

Translating the message text

All ZDT/Db2 messages are stored in the HFM2MENU source member. This CSECT is part of the root module so that an English version of the messages is always available. In addition, all messages used by ZDT/Db2 under ISPF are provided in the library, HFM.SHFMMENU. Using both HFM2MENU and members in HFM.SHFMMENU, you can provide your own set of translated messages. To use the messages you have translated, see Using the translated messages and panels on page 214.

To provide translated versions of the messages, you must provide a version of HFM2MENU for your language, as described in Providing a multicultural version of HFM2MENU on page 212, **and** provide translated versions of the appropriate members in HFM.SHFMMENU, as described in Translating the ISPF messages text on page 213.

You should not need to provide a Japanese version of the messages if you have installed the ZDT/Db2 Japanese component.

Providing a multicultural version of HFM2MENU

HFM2MENU contains the assembler source for the ZDT/Db2 messages. To provide translated versions of the messages:

1. Copy the member HFM2MENU from HFM.SHFMSAM1 to your own source library with the name HFM2M*yyy*, where *yyy* is one of the following language codes:

   FRA

   French

   DEU

   German

   ITA

   Italian

   JPN

   Japanese

   PTG

   Portuguese

   ESP

   Spanish

DAN

Danish

ENP

Upper case English

KOR

Korean

DES

Swiss German

CHT

Traditional Chinese

CHS

Simplified Chinese

XXX

Other

2. Change the message text in HFM2M*yyy* in your library.
3. Modify the HFM2UMDM member in HFM.SHFMSAM1 to meet your site's requirements, using the same language code as above. Refer to the usermod for information about other changes you might need to make.
4. Install SMP/E usermod HFM2UMDM.

Translating the ISPF messages text

All ZDT/Db2 ISPF messages are provided in English.

They are provided in Japanese if you have installed the ZDT/Db2 Japanese component.

You can translate some or all of these messages into another language.

All ZDT/Db2 ISPF messages are stored in HFM.SHFMMENU. You translate a message as follows:

1. Find the members in HFM.SHFMMENU that contain the messages you want to translate. The message members specific to ZDT/Db2 are all named HFMD*zzzz* or HFM2*n*.
2. Create a library with the same characteristics as HFM.SHFMMENU, with the name HFM.SHFMM*yyy*, where *yyy* is the same language code you specified when you modified HFM2MENU. If you have already created a library with this name for translated Z Data Tools Base function messages, use that library. Copy the required message members from HFM.SHFMMENU to this library.
3. Change the required message texts in these members in your library.

To use the messages you have translated, see Using the translated messages and panels on page 214. For more information about defining and using ISPF messages, see *z/OS ISPF Dialog Developer's Guide*.

Note: Be sure to include **ALL** the messages in the message members you copy to your own library. When ZDT/Db2 needs to display an ISPF message, it uses ISPF services to do this. Therefore the search for a message is made according to ISPF rules. Thus, if ISPF finds the message member it requires in your library, but the message number required is not in that member, ISPF will not look in any other library for the message, but will give an error. However, if you omit a complete message member from your library, then ISPF will use the English message member from the next library in the ISPMLIB concatenation. For more information about defining and using ISPF messages, see *z/OS ISPF Dialog Developer's Guide*.

Translating the panel text

All ZDT/Db2 ISPF panels are provided in English.

They are also provided in Japanese if you have installed the ZDT/Db2 Japanese component. You can translate some or all of these panels into another language. (If no translated version of a particular panel is available, ZDT/Db2 uses the English version.)

All ZDT/Db2 panels are stored in HFM.SHFMPENU. You translate a panel as follows:

1. Find the panel members in HFM.SHFMPENU that you want to translate. The panel members specific to ZDT/Db2 are all named HFM2*zzzz*.
2. Create a library with the same characteristics as HFM.SHFMPENU, with the name HFM.SHFMP*yyy*, where *yyy* is the same language code you specified when you modified HFM2MENU. If you have already created a library with this name for translated Z Data Tools Base function panels, use that library. Copy the required panel members from HFM.SHFMPENU to this library.
3. Change the required panel text in the members in your library. A panel may reference a **help** panel by means of a `.HELP` statement. If any panel you are changing contains any of these `.HELP` statements, also copy and change these referenced members in your library.

To use the panels you have translated, see Using the translated messages and panels on page 214.

Using the translated messages and panels

To use your translated messages in a batch job, specify the appropriate language with the LANGUAGE processing option, using the keywords shown in Table 39: Keyword values for the LANGUAGE option on page 214. See Changing the default options on page 180 for information on how to do this.

Table 39. Keyword values for the LANGUAGE option

Language	Code	Specify on LANGUAGE option...
French	FRA	FRENCH
German	DEU	GERMAN
Italian	ITA	ITALIAN

Table 39. Keyword values for the LANGUAGE option (continued)

Language	Code	Specify on LANGUAGE option...
Japanese	JPN	JAPANESE
Portuguese	PTG	PORTUGUESE
Spanish	ESP	SPANISH
Danish	DAN	DANISH
Upper case English	ENP	UPPERENG
Korean	KOR	KOREAN
Swiss German	DES	SGERMAN
Traditional Chinese	CHT	CHINESET
Simplified Chinese	CHS	CHINESES
Other	XXX	OTHER

For example, to use French messages, specify LANGUAGE=FRENCH.

Under ISPF, the language used in messages and panels is determined by the setting of the national language for ISPF, for the current ISPF session. For information on changing the national language setting for your ISPF session, see *z/OS ISPF Dialog Developer's Guide*.

If your ISPF session is set up to use your language, you need to add your libraries to the appropriate ISPF concatenation, in front of any Z Data Tools English libraries. For example, to use your translated messages, add HFM.SHFMM*yyy* to ISPMLIB, in front of HFM.SHFMMENU. To use your translated panels, add HFM.SHFMP*yyy* to ISPPLIB, in front of HFM.SHFMPENU.

Providing a multicultural version of HFM2DENU

HFM2DENU contains the assembler source for the ZDT/Db2 object list utility panel titles, column headings, and text used in expanded SQL statements. These are the titles, headings, and statements displayed when you select ZDT/Db2 option 3.4 and following options. These titles, headings, and statements can be translated to your national language.

Note: If you have installed the ZDT/Db2 Japanese component, these titles, headings and statements are provided in Japanese, in the module HFM2DJPN.

To translate these titles, headings or statements to your national language:

1. Copy the member HFM2DENU from HFM.SHFMSAM1 to your own source library with the name HFM2D*yyy*, where *yyy* is one of the following language codes:

   FRA

   French

DEU

German

ITA

Italian

JPN

Japanese

PTG

Portuguese

ESP

Spanish

DAN

Danish

ENP

Upper case English

KOR

Korean

DES

Swiss German

CHT

Traditional Chinese

CHS

Simplified Chinese

XXX

Other

2. Change the text in HFM2D*yyy* in your library as required, according to the following information:
   - Change the statement at about line 419, which reads:

     ```
     DC     CL8'HFM2DENU'
     ```

     to:

     ```
     DC     CL8'HFM2D yyy'
     ```

     where *yyy* is the language code you selected in Step 1.
   - To change the panel headings, translate the wording in the statements starting `HFM2@TTL` and containing `TTL=...`

- To change the column headings, translate the wording in the statements starting `HFM2@CHD` and containing `CHD=...`

  Where a ! separates two words, these two words appear on separate lines in the column heading.

- To change the text used in expanded SQL statements, translate the wording between the quotation marks on the SETC statements, starting at about line 63.

Do not change any statements other than those listed above.

For example, in the `Tables, Views and Aliases` panel (HFM2POLT), to translate the panel title, you would change the statement:

```
HFM2@TTL ISPFV=QSTBPT,TTL='Tables, Views and Aliases'
```

To translate the column heading, TABLE SPACE NAME (where the words appear on three separate lines), you would change the statement:

```
HFM2@CHD XRF=SCTSNAME,CHD='TABLE!SPACE!NAME'
```

To translate the words Alias and Table for the OBJECT TYPE column, you would change the statements:

```
&QTXT(4)   SETC 'Alias'
```

and

```
&QTXT(297) SETC 'Table'
```

3. Modify the HFMUMDD member in HFM.SHFMSAM1 to meet your site's requirements, using the same language code as above. Refer to the usermod for information about other changes you might need to make.
4. Install SMP/E usermod HFMUMDD.

Customizing for the Japanese national language

The other customization task you might need to do for the Japanese national language is to modify the supplied Japanese translation tables. If you want to do this, do so as part of the customization of the Z Data Tools Base function. See Modifying the Japanese translation tables on page 109.

Changing the Japanese message text

If you have installed the ZDT/Db2 Japanese component, all ZDT/Db2 Japanese messages are stored in the HFM2MJPN source member. Normally you should not need to modify this module. However, if you do want to modify it, you can do so by means of the usermod, HFM2UMDN.

To do this:

1. Copy the member HFM2MJPN from HFM.SHFMSAM1 to your own source library.
2. Change the message text in HFM2MJPN in your library.
3. Modify the HFM2UMDN member in HFM.SHFMSAM1 to meet your site's requirements. Refer to the usermod for information about other changes you might need to make.
4. Install SMP/E usermod HFM2UMDN.

Chapter 19. Verifying the customization of ZDT/Db2

After you have completed all the necessary steps to install and customize ZDT/Db2, you can perform the following steps to verify your customization.

1. Use the ZDT/Db2 *"Edit/Execute SQL (Data Set)"* function to execute sample DDL. This creates a small database that is used in subsequent IVP steps.
2. Run a batch job to execute the Db2® CHECK DATA utility against the Db2® objects in the ZDT/Db2 IVP database.
3. Use the ZDT/Db2 *"Db2® Utilities"* function to generate and run a job to take an image copy of the Db2® objects in the ZDT/Db2 IVP database.
4. Use the ZDT/Db2 editor to display and modify data in the ZDT/Db2 IVP tables.
5. Use the ZDT/Db2 *"Basic SELECT Prototyping"* function to create and execute a simple join of the tables in the ZDT/Db2 IVP database.
6. Use the ZDT/Db2 *"Copy utility"* function to copy data from one IVP table to another.
7. Use the ZDT/Db2 *"Export utility"* function to copy the data in one of the ZDT/Db2 IVP tables to a sequential data set.
8. Use the ZDT/Db2 *"Import utility"* function to copy the sequential data set (created in step 7) into a Z Data Tools/Db2 IVP table.
9. Use the ZDT/Db2 *"Object list utility"* function to display information about the ZDT/Db2 IVP database and objects.

To review the customization and verification steps, refer to:

- Checklist for installing and customizing ZDT/Db2 on page 153
- *Z Data Tools User's Guide and Reference for DB2 Data*.

Step 1. Define Db2® objects to be used during verification

The first step in the ZDT/Db2 verification is to define the Db2® objects to be used.

Take a copy of the sample IVP member HFM2VER from the sample library.

Note: The figures in this chapter are from a Db2® version 12 system.

Follow the instructions at the top of the sample. You need to review, and possibly change, the following values:

SG name

Change this to the name of a Db2® storage group of your choice. You can use an existing Db2® storage group name. To do this, change *SG_name* to the existing storage group name, and comment out the CREATE STOGROUP statement. If you use an existing Db2® storage group name, you can skip the customization for *Vol_list* and *DB2_VCAT_name*.

VOL_list

Change this, if required, to a list of valid disk volume names.

DB2_VCAT_name

Change this to the VCAT name used for Db2® user data sets, for the appropriate Db2® system.

To execute the DDL statements:

1. Log on to a TSO user ID that is enabled to access and use ZDT/Db2.
2. Start ZDT/Db2.
   - If you have added an option for ZDT/Db2 to your ISPF Primary Options menu (see Adding ZDT/Db2 to the ISPF menu on page 159), type this option value, and press **Enter**. For example, if you have assigned ZD to ZDT/Db2, type ZD and then press **Enter**.
   - If you defined ZDT/Db2 in an ISPF command table (see Defining ZDT/Db2 in an ISPF command table on page 159), verify that ZDT/Db2 can be started by entering the command ZD on any ISPF command line.
3. If you have previously selected a Db2® subsystem, you immediately see the Primary Option Menu.

   If this is the first time you have used ZDT/Db2:

   - The Copyright panel appears. After reading the panel text, press the Cancel key, PF12. In subsequent sessions this panel does not automatically appear.
   - The **Db2 Subsystem Selection** menu is displayed, showing the Db2® subsystems you customized in your HFM2SSDM macro. Select the Db2® subsystem you want to use, or run the ZDT/Db2 IVP against, and press **Enter** to go to the ZDT/Db2 Primary Option Menu.
4. Enter VER on the command line to display the release level and PTF level of ZDT/Db2. A panel is displayed that gives you, as example, the following information:

```
HCL Z Data
                        Tools
                              Version 17 Release
                                    1 Modification 0
Db2 Component
(not APF authorized)

Service Levels of installed components

             Base       IMS        Db2        CICS
English      -NONE-     -NONE-     -NONE-     -NONE-
```

   - ZDT/Db2 is always shown as `APF not authorized`, even if you have made Z Data Tools APF-authorized, as Z Data Tools cannot run as APF-authorized under ISPF.
   - When you first install Z Data Tools, `-NONE-` is shown against each component. Subsequently, when you have applied service to Z Data Tools, a PTF number is shown, indicating the PTF level of each component you have installed. If you have not installed a component, that component is not be shown at all.

     If you have installed the Japanese language component of ZDT/Db2, another line is displayed indicating the service level of that component.
5. Select the ZDT/Db2 system options by typing '0.0.2' on the ZDT/Db2 main menu panel and pressing Enter. See Figure 16: ZDT/Db2 System Options panel on page 220. Ensure that:
   - The *"Translate Db2® object names"* option is **not selected**.
   - The *"Translate input SQL statements"* option is **not selected**.
6. Press PF3 to return to the ZDT/Db2 main menu.

Figure 16. ZDT/Db2 System Options panel

```
 Process   Options   Utilities   Help
_____________________________________________________________________________
ZDT/Db2 (DFF2)          ZDT/Db2 System Options (1 of 4)          Global Settings
Command ===> _____________________________________________________________

Uppercase Translation:
   Enter "/" to select option
   _  Translate Db2 object names
   _  Translate input SQL statements

Encapsulation of SQL Identifiers in Double Quotes:
   Use double quotes
   3  1. For Db2 SQL reserved words
      2. For IBM and Db2 SQL reserved words
      3. Always

More Options:
   Enter "/" to select option
   /  Arbitrary select statements, time display format, template creation
   /  CCSID warning option
   /  Editor choice for viewing output data, members

 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F11=NxtPage  F12=Cancel
```

7. Select the *"Edit/Execute SQL (Data Set)"* function by typing '4.4' on the ZDT/Db2 main menu panel and pressing Enter.

   See Figure 17: Edit/Execute SQL (Data Set) panel on page 220.

   Enter a data set name, member name, and execution options, as shown in Figure 17: Edit/Execute SQL (Data Set) panel on page 220. Modify the data set and member names to reflect the data set and member name containing your modified copy of the HFM2VER sample.

   Figure 17. Edit/Execute SQL (Data Set) panel

```
 Process   Options   Utilities   Help
_____________________________________________________________________________
ZDT/Db2 (DFF2)            Edit/Execute SQL (Data Set)
Command ===> _____________________________________________________________

Input Data Set:
   Data set name . . . 'HFM.V17R1M0.IVP'_____________________________
   Member  . . . . . . HFM2VER
   Volume  . . . . . . ______

SELECT statement options:
   Row count . . . . . ALL_____     Number of rows to browse

Processing Options:
   Execution options                Enter "/" to select option
   1    First column                /  Edit data set
   72   Last column                 /  Execute SQL from data set
   --   Comment chars               _  Re-edit data set after execution



 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

8. Press Enter to edit the sample DDL. When you have made any required changes, press PF3 to run the DDL statements.

When the DDL samples run successfully, two SQL warning messages are displayed. See Figure 18: SQL Warning (1) panel on page 221 and Figure 19: SQL Warning (2) panel on page 221. These warnings are expected. To continue running, press Enter.

Figure 18. SQL Warning (1) panel

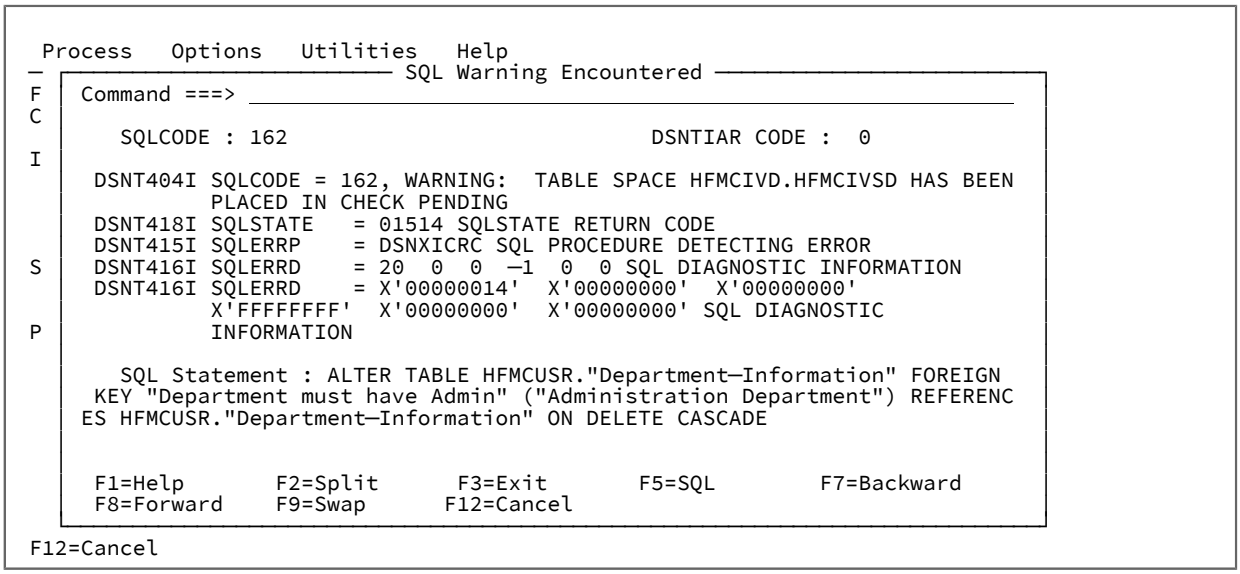
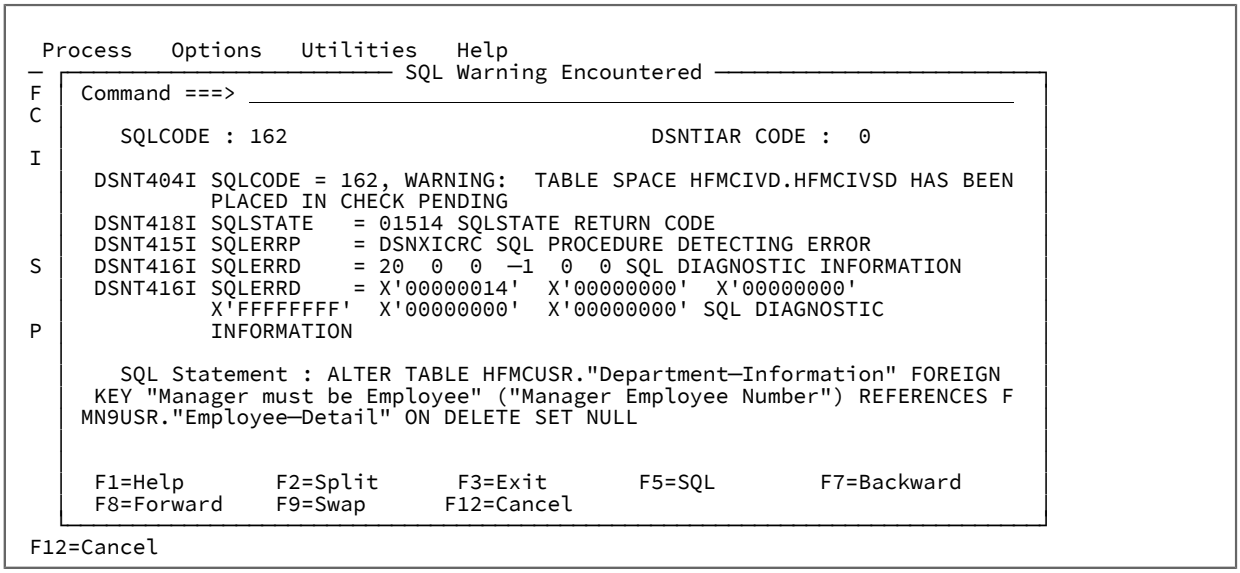


Figure 19. SQL Warning (2) panel



When the SQL has run successfully, a box appears at the bottom of the screen with *"173 statements executed"*.

Move onto Step 2. Run the HFM2CHCK sample job on page 222 of the IVP.

Step 2. Run the HFM2CHCK sample job

The Db2® objects created in Step 1. Define Db2 objects to be used during verification on page 218 have referential constraints defined. Before these objects can be accessed, you need to run the Db2® CHECK DATA utility against the objects to ensure that the data does not violate any data constraints.

Take a copy of the *"check data"* member HFM2CHCK from the sample library.

Follow the instructions at the top of the sample. You need to review and possibly change the following values:

Job card

Specify a valid job card.

DB2LLIB

Change the default value DSN.SDSNLOAD to the name of the Db2® load library.

DB2PROC

Change the default value DSN.SDSNPROC to the name of the Db2® procedure library. If a Db2® proclib is required, comment out the JCLLIB statement.

SSID

Change DSN to the name of the Db2® subsystem.

SORTLIB

Change the default value SYS1.SORTLIB to the library name used by DFSORT or an equivalent sort product.

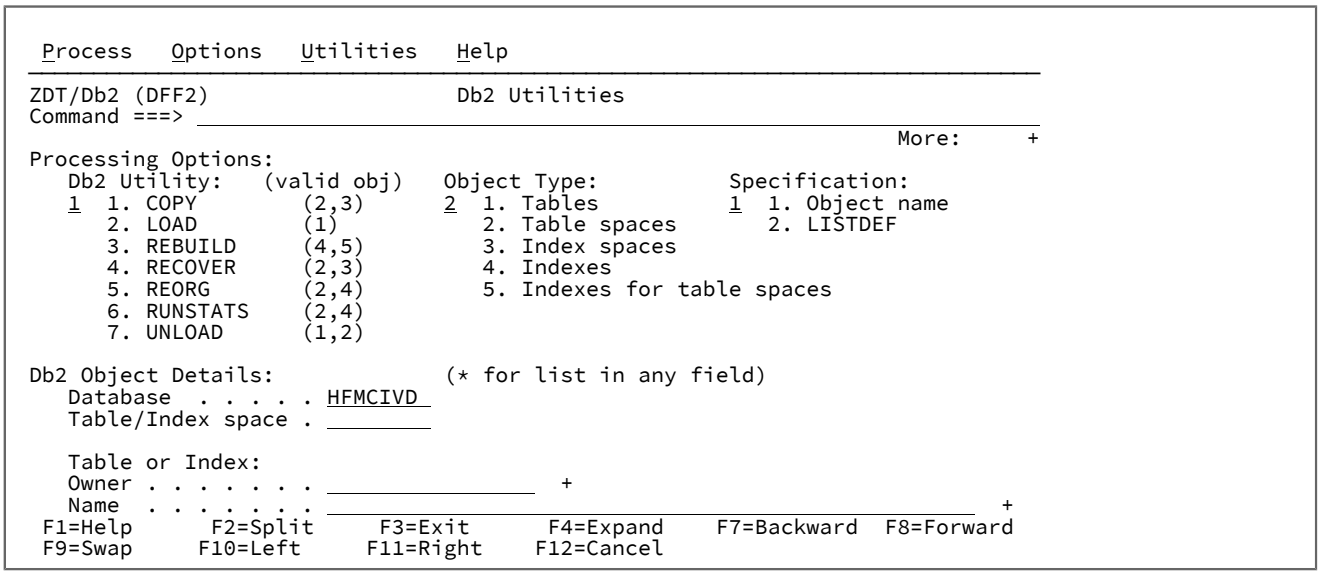
Submit the job. The expected return code is 0.

Step 3. Take a copy of the ZDT/Db2 IVP tables

Log on to ZDT/Db2 and connect to the same Db2® system specified in Step 1. Define Db2 objects to be used during verification on page 218. Select the *"Db2® Utilities"* function by typing '3.9' on the ZDT/Db2 main menu and press **Enter**. See Figure 20: Db2 Utilities panel on page 223.

Enter the name of the ZDT/Db2 IVP database in the Database field, select the COPY utility and Table spaces. Press **Enter**.

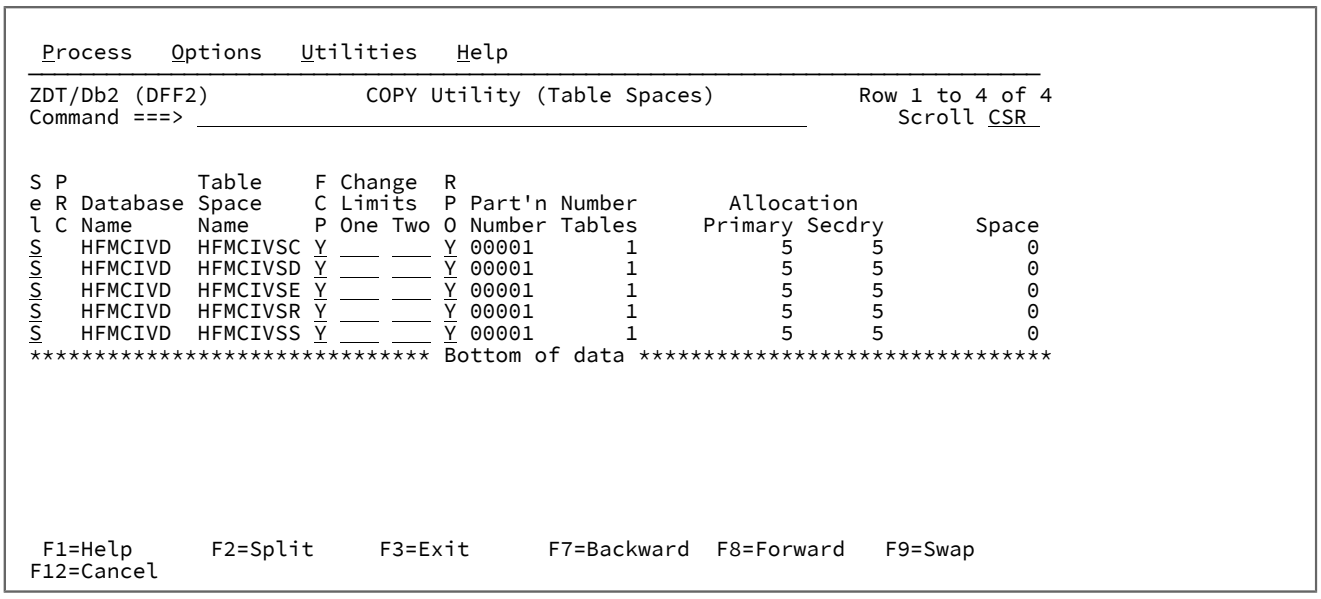
Figure 20. Db2® Utilities panel



See Figure 21: COPY Utility panel (Table Spaces) on page 223.

This display shows the table spaces defined in the ZDT/Db2 IVP data base. Type *"s"* against every entry and press Enter.

Figure 21. COPY Utility panel (Table Spaces)



ZDT/Db2 displays an ISPF edit session with a job that takes an image copy of the selected table spaces. Check the following carefully:

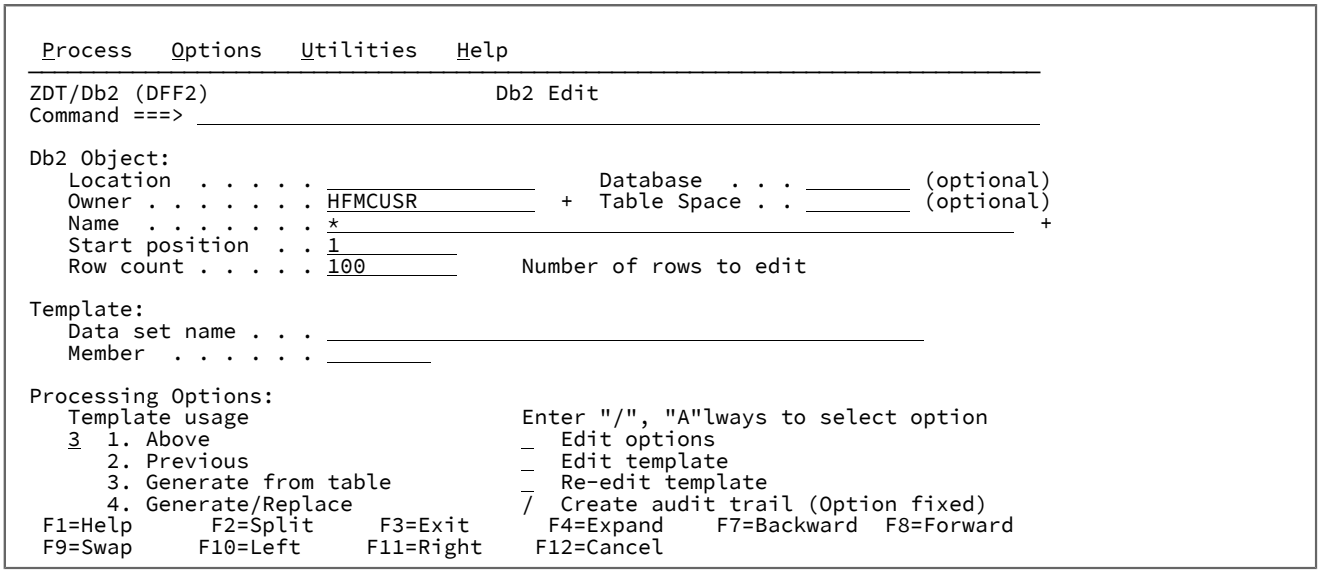
- Job card.
- DSNUPROC.STEPLIB libraries. The values for these libraries are those specified in the HFM2SSDM macro entry for the connected Db2® system in the HFM2POPT. For more information, see Defining all Db2 systems that ZDT/Db2 will access in HFM2POPT (required) on page 174.

Submit the job. The expected return code is RC=0.

Step 4. Use the ZDT/Db2 editor (normal and related edit)

Log on to ZDT/Db2, and connect to the same Db2® system specified in Step 1. Define Db2 objects to be used during verification on page 218. Select the ZDT/Db2 editor by typing '2' on the ZDT/Db2 main menu. Press **Enter**.

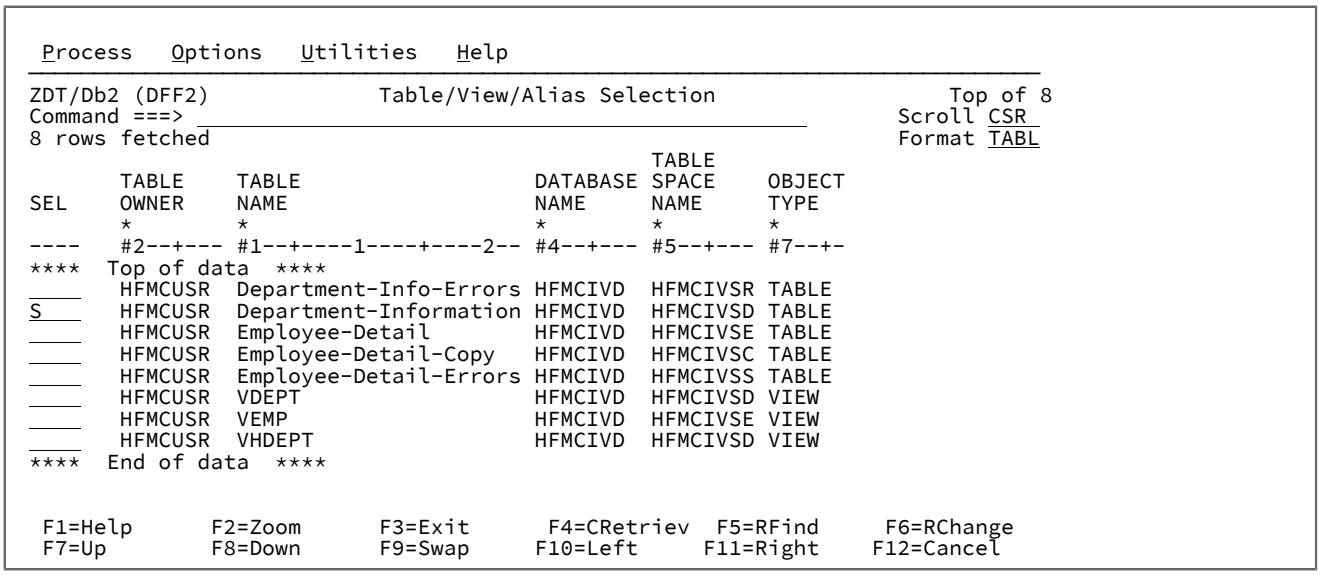
Figure 22. Db2® Edit panel



See Figure 22: Db2 Edit panel on page 224. Enter the values shown in the figure, HFMCUSR in the *"Owner"* field and '*' in the *"Name"* field.

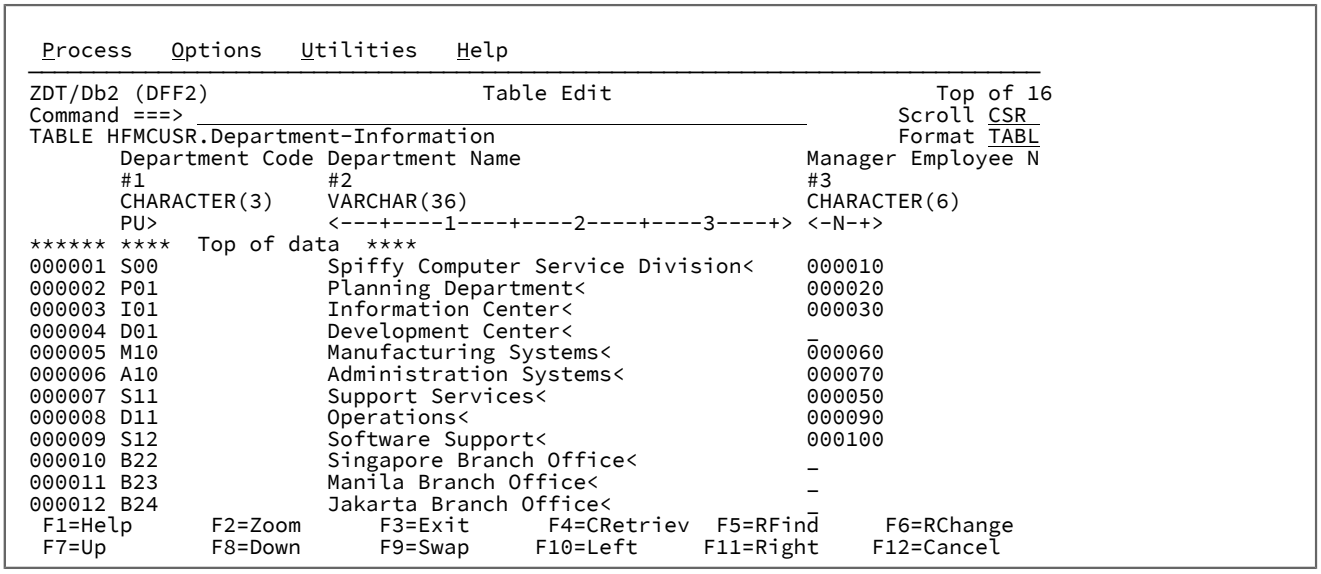
Press **Enter** to display the *"Table/View/Alias Selection"* panel. See Figure 23: Table/View/Alias Selection panel on page 224.

Figure 23. Table/View/Alias Selection panel



Type 'S' against the *"Department-Information"* field in the table and press **Enter**. See Figure 24: Table Edit panel on page 225.

Figure 24. Table Edit panel



Type over the *"Manager Employee Number"* value in the second row, as shown in Figure 25: Table Edit panel (showing typeover Manager Employee number) on page 225. Change the value to 'XXXXXX' and press **PF3**. See Figure 26: Table Edit panel (showing amended Manager Employee number) on page 226 to see the ZDT/Db2 response when the PF3 (EXIT) key is pressed.

Figure 25. Table Edit panel (showing typeover Manager Employee number)

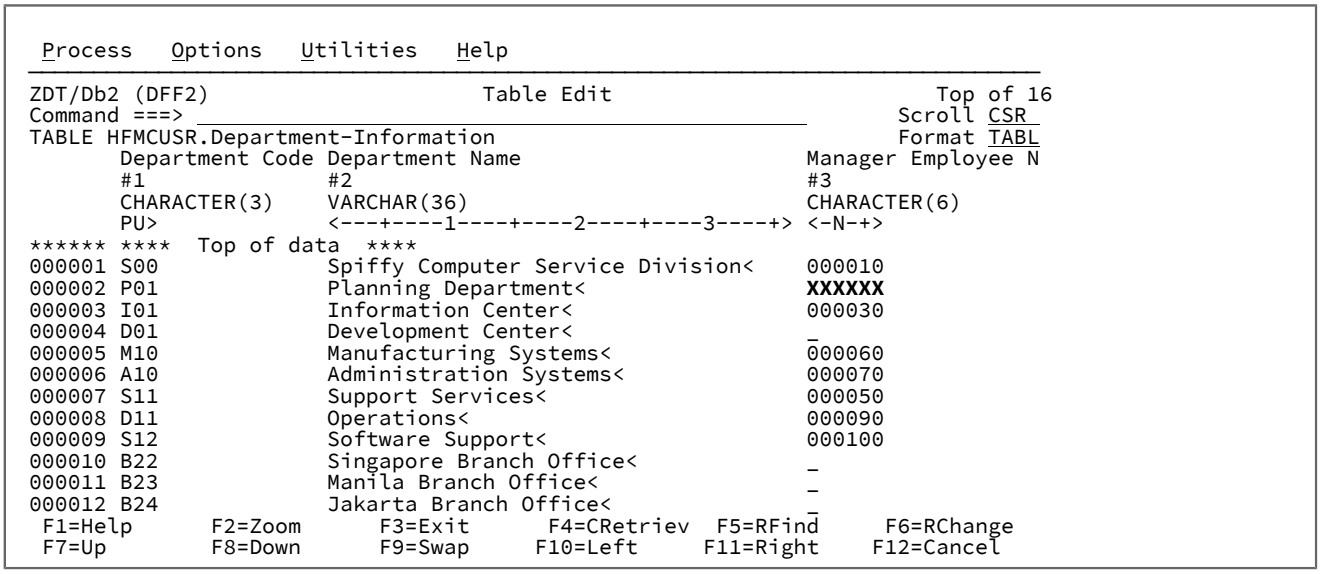
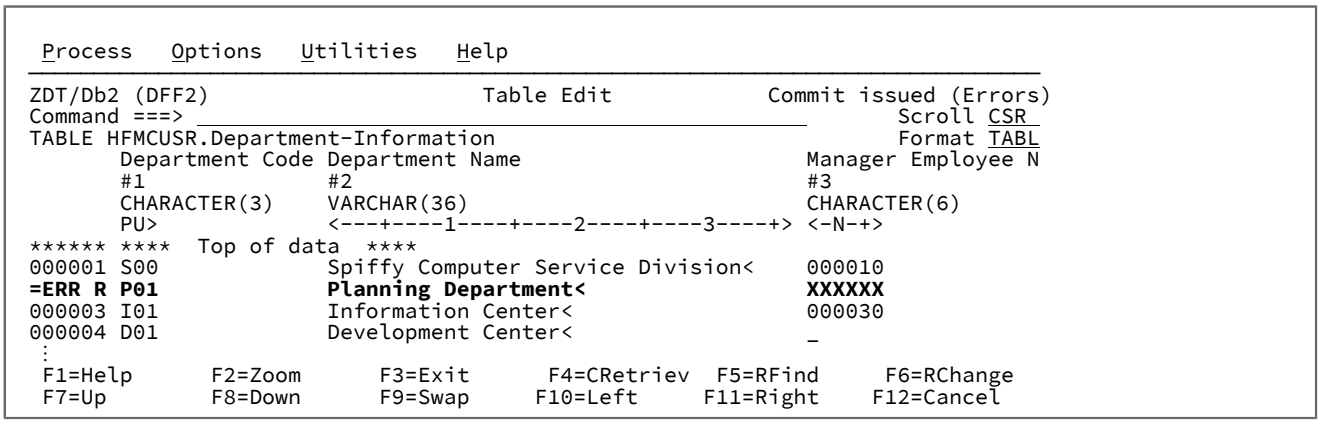
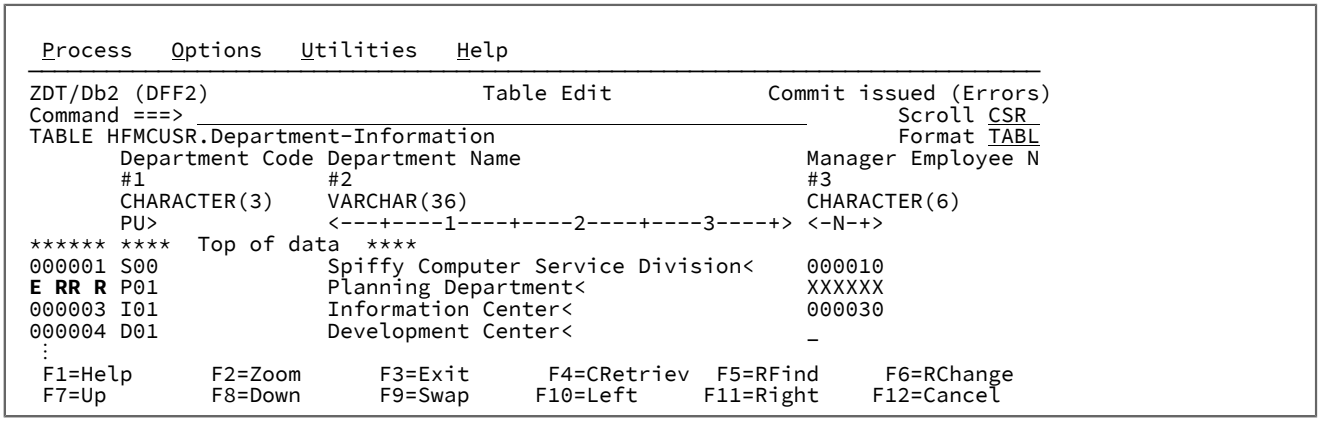


Figure 26. Table Edit panel (showing amended Manager Employee number)



The change to the *"Manager Employee Number"* could not be saved. The row in error is marked with `'=ERR R'`. Overtype 'e' in the prefix area for this row and press **Enter**. See Figure 27: Table Edit with amended Manager Employee number on page 226.

Figure 27. Table Edit with amended Manager Employee number



ZDT/Db2 displays the error panel shown in Figure 28: Db2 Save Error Action panel on page 227. Type the REDIT command on the command line, and press **Enter**. See Figure 29: Table Edit (related) panel on page 227 to display the parent table for the relationship.

Figure 28. Db2® Save Error Action panel

```
 Process   Help
--------------------------------------------------------------------------------
ZDT/Db2 (DFF2)              Db2 Save Error Action
Command ===> ____________________________________________________________________

Db2 reported a No Primary Key error while attempting to save this row.  See
below for key column details.

Relationship    : Manager must be Employee
Parent table    : HFMCUSR.Employee-Detail
Dependent table : HFMCUSR.Department-Information

Explanation: The insert or update operation on this line would have resulted
in a foreign key value for which there is no corresponding primary key value.

Instructions: Type REDIT on the command line to edit the parent table shown
above.  Press ENTER or enter EXIT to return to the edit session and correct
the error.  Press the CANCEL key to terminate the edit session.  Any changes
made since the last commit point will be lost.

Parent Column Name   Depndnt Column Name  Value
Employee Number      Manager Employee Num XXXXXX
 F1=Help      F2=Split     F3=Exit      F4=Expand    F7=Backward  F8=Forward
 F9=Swap     F10=Left     F11=Right    F12=Cancel
```

Figure 29. Table Edit (related) panel

```
 Process   Options   Utilities   Help
--------------------------------------------------------------------------------
ZDT/Db2 (DFF2)               Table Edit (related)                Parent table
48 rows - End of object.                                         Format TABL
       Employee Number First Name   Middle Initial Last Name      Work Depart
       #1              #2           #3             #4            #5
       CHARACTER(6)    VARCHAR(12)  CHARACTER(1)   VARCHAR(15)   CHARACTER(3
       PU-->           <---+----1-> -              <---+----1----> <-N
****** ****  Top of data  ****
000001 000010          Xena<        B              Howard<       S00
000002 000020          Michelle<    J              Jackson<      P01
000003 000030          James<       Z              Jones<        I01
000004 000050          Jo-anne<     G              Gratten<      S11
000005 000060          Jackson<     P              Costello<     M10
000006 000070          Charles<     A              Abercrombie<  A10
000007 000090          Eleni<                      MacMahon<     D11
000008 000100          Junichi<     K              Funahashi<    S12
000009 000110          Luigi<       V              Andretti<     S00
000010 000120          Patrick<     O              O'Farrelly<   S00
000011 000130          Megumi<                     Takami<       I01
Command ===>                                                    Scroll CSR
 F1=Help      F2=Zoom      F3=Exit      F4=CRetriev  F5=RFind     F6=RChange
 F7=Up        F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
```

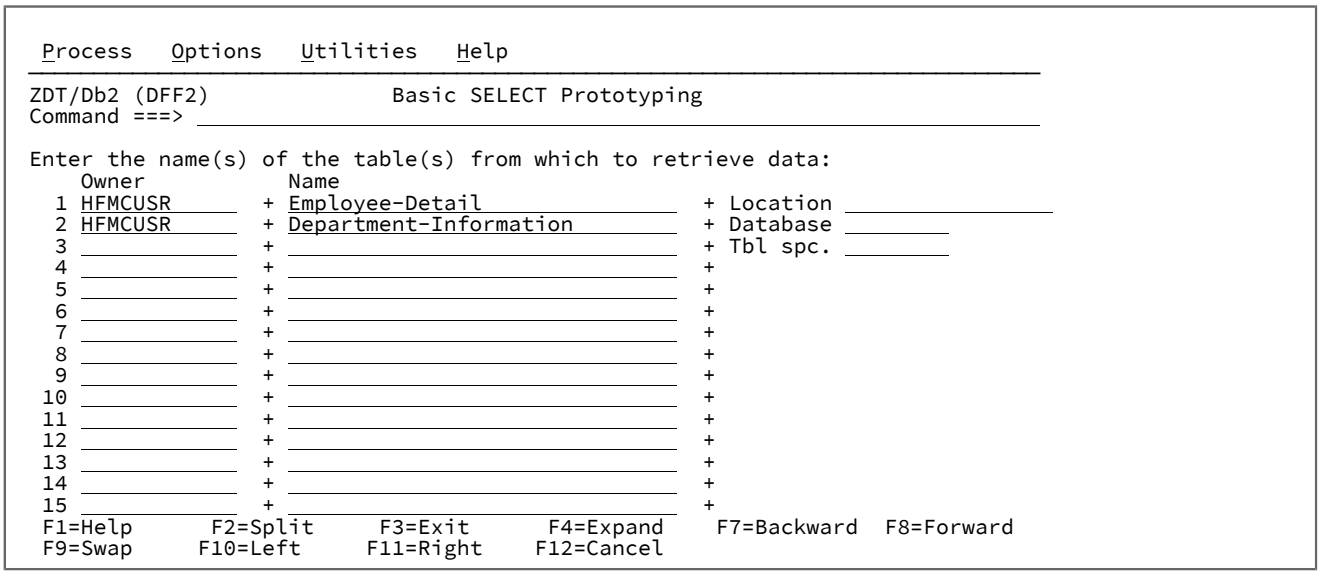
Press **PF3** to exit from the ZDT/Db2 edit session of the *"Employee-Detail"* table and return to the ZDT/Db2 edit session of the *"Department-Information"* table. Press **PF12** (CANCEL) to return to the ZDT/Db2 Editor function entry panel. A pop-up panel is displayed asking for confirmation to cancel the operation. Press **Enter** to continue.

Step 5. Use the ZDT/Db2 Basic SELECT Prototyper

Log on to ZDT/Db2, and connect to the same Db2® system specified in Step 1. Define Db2 objects to be used during verification on page 218. From the ZDT/Db2 main menu, enter "4.1" to display the *"Basic SELECT Prototyping"* function.

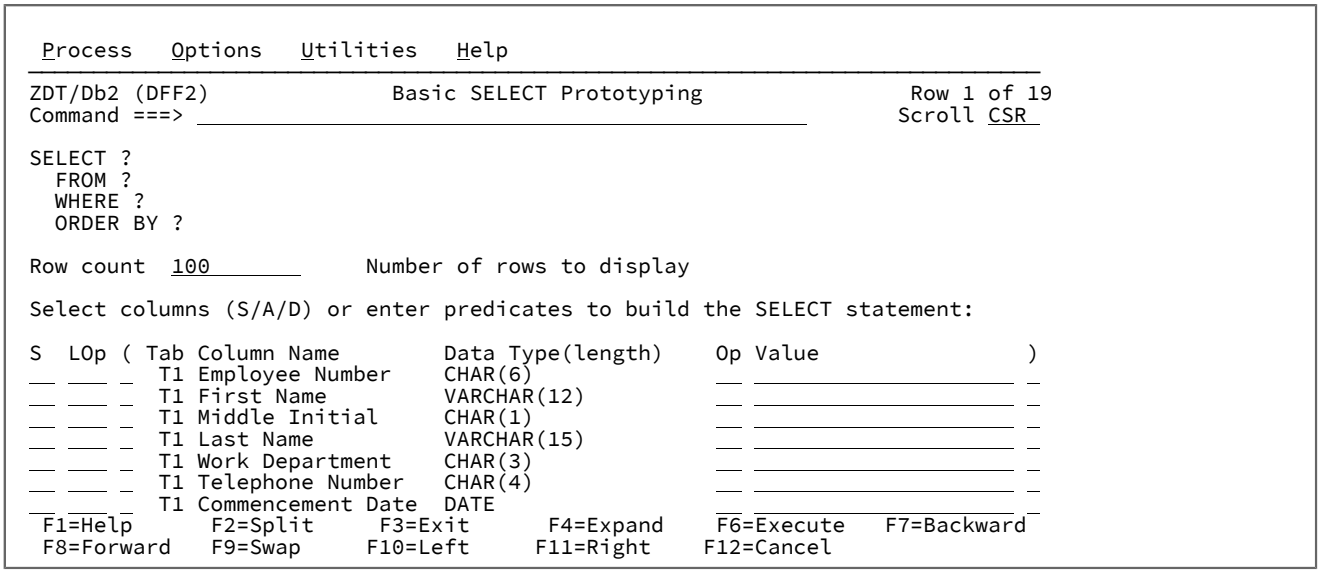
Enter the names of the two ZDT/Db2 IVP tables as shown in Figure 30: Basic SELECT Prototyping panel on page 228.

Figure 30. Basic SELECT Prototyping panel



Press **Enter** to display the second *"Basic SELECT Prototyping"* panel. See Figure 31: Basic SELECT Prototyping panel (2) on page 228.

Figure 31. Basic SELECT Prototyping panel (2)



In this example, an SQL query is developed that shows the Employees (Names and Employee number only), and the Department name; for Department Code = "A10". Complete the following steps to prototype this statement:

1. Select the following columns by typing 'S' next to the column name:
   - First Name
   - Middle Initial
   - Last Name

   Press **Enter**.

The selected columns are added to the SELECT clause, which is displayed at the top of the panel.

Select the following additional columns:

- Employee Number
- Department Code

Press **Enter**. The additional columns are added to the SELECT clause, after the columns selected previously.

See the result in Figure 32: Basic SELECT Prototyping panel (3) on page 229.

Figure 32. Basic SELECT Prototyping panel (3)

```
 Process   Options   Utilities   Help
──────────────────────────────────────────────────────────────────────────────
ZDT/Db2 (DFF2)              Basic SELECT Prototyping               Row 1 of 19
Command ===> ____________________________________________________  Scroll CSR

SELECT "First Name", "Middle Initial", "Last Name", "Employee Number", "Dep  +
  FROM "HFMCUSR"."Employee-Detail", "HFMCUSR"."Department-Information"
  WHERE ?
  ORDER BY ?

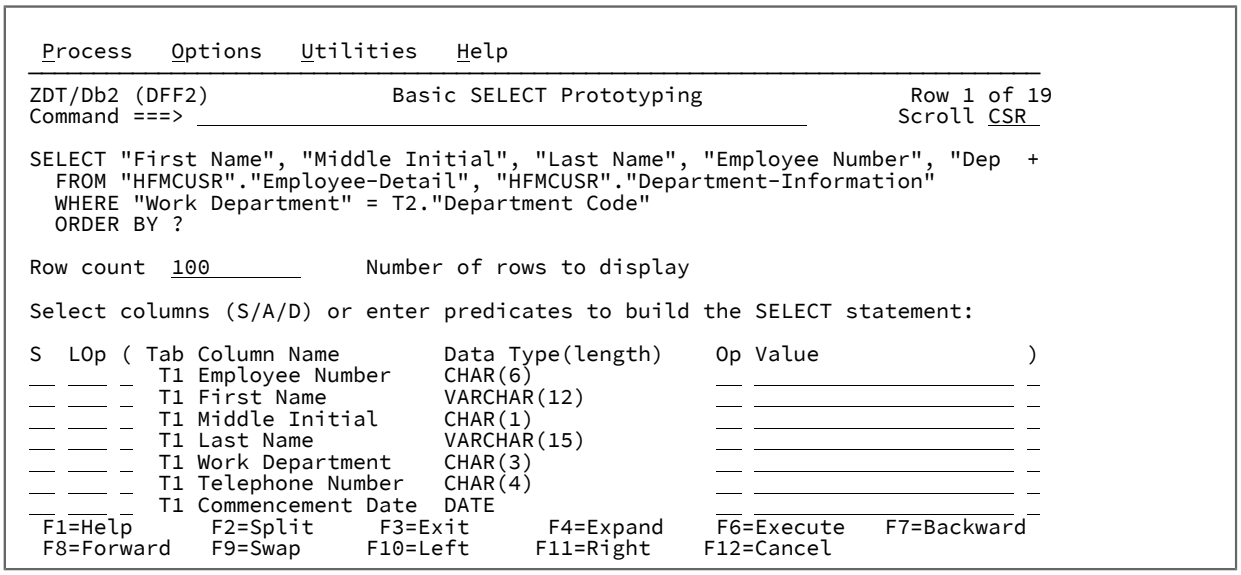
Row count  100_______     Number of rows to display

Select columns (S/A/D) or enter predicates to build the SELECT statement:

S  LOp ( Tab Column Name        Data Type(length)    Op Value                 )
__ ___ _  T1 Employee Number    CHAR(6)              __ ____________________ _
__ ___ _  T1 First Name         VARCHAR(12)          __ ____________________ _
__ ___ _  T1 Middle Initial     CHAR(1)              __ ____________________ _
__ ___ _  T1 Last Name          VARCHAR(15)          __ ____________________ _
__ ___ _  T1 Work Department    CHAR(3)              __ ____________________ _
__ ___ _  T1 Telephone Number   CHAR(4)              __ ____________________ _
__ ___ _  T1 Commencement Date  DATE                 __ ____________________ _
 F1=Help      F2=Split     F3=Exit      F4=Expand    F6=Execute   F7=Backward
 F8=Forward   F9=Swap     F10=Left     F11=Right    F12=Cancel
```

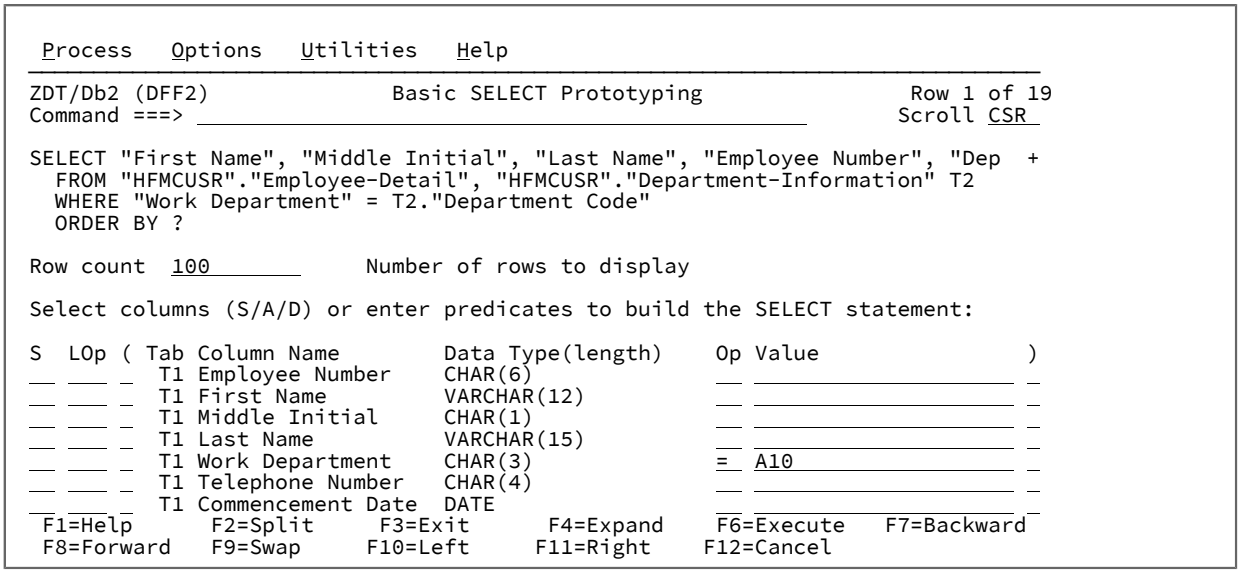
2. Specify the join between the two tables. To do this, type T2."Department Code" in the "Value" column next to the entry for "Work Department" in the "Employee-Detail" table. You should also specify an '=' in the "Op" column. Press **Enter**. The WHERE clause is updated to include "Work Department" = T2."Department Code". See Figure 33: Basic SELECT Prototyping panel (4) on page 230.

Figure 33. Basic SELECT Prototyping panel (4)



3. Add a WHERE clause to show only the employees in the "A10" Department by typing "A10" in the *"Work Department"* column. See Figure 34: Basic SELECT Prototyping panel (5) on page 230.

   Figure 34. Basic SELECT Prototyping panel (5)



4. Press **Enter** to update the WHERE clause to include only the employees in the "A10" Department. See Figure 35: Basic SELECT Prototyping panel (6) on page 231

Figure 35. Basic SELECT Prototyping panel (6)

```
 Process   Options   Utilities   Help
──────────────────────────────────────────────────────────────────────────────
ZDT/Db2 (DFF2)              Basic SELECT Prototyping                Row 1 of 19
Command ===> ____________________________________________________   Scroll CSR

SELECT "First Name", "Middle Initial", "Last Name", "Employee Number", "Dep  +
  FROM "HFMCUSR"."Employee-Detail", "HFMCUSR"."Department-Information" T2
  WHERE "Work Department" = T2."Department Code" AND "Work Department" = 'A10'
  ORDER BY ?

Row count  100_______      Number of rows to display

Select columns (S/A/D) or enter predicates to build the SELECT statement:

S  LOp ( Tab Column Name        Data Type(length)    Op Value                )
__ ___ _  T1 Employee Number    CHAR(6)              __ ____________________ _
__ ___ _  T1 First Name         VARCHAR(12)          __ ____________________ _
__ ___ _  T1 Middle Initial     CHAR(1)              __ ____________________ _
__ ___ _  T1 Last Name          VARCHAR(15)          __ ____________________ _
__ ___ _  T1 Work Department    CHAR(3)              __ ____________________ _
__ ___ _  T1 Telephone Number   CHAR(4)              __ ____________________ _
__ ___ _  T1 Commencement Date  DATE                 __ ____________________ _
 F1=Help      F2=Split     F3=Exit      F4=Expand    F6=Execute   F7=Backward
 F8=Forward   F9=Swap      F10=Left     F11=Right    F12=Cancel
```

5. Either type EXECUTE on the command line, or press the **PF6** key to run the SQL statement and display the result table. See Figure 36: Select Statement Browse panel on page 231.

   Figure 36. Select Statement Browse panel

```
 Process   Options   Utilities   Help
──────────────────────────────────────────────────────────────────────────────
ZDT/Db2 (DFF2)              Select Statement Browse                   Top of 6
Command ===> ____________________________________________________   Scroll CSR
6 rows - End of object.                                            Format TABL
First Name   Middle Initial Last Name       Employee Number Department Code
#2           #3             #4              #1              #5
VARCHAR(12)  CHARACTER(1)   VARCHAR(15)     CHARACTER(6)    CHARACTER(3)
<---+----1-> -              <---+----1----> PU--+>          <->
****  Top of data  ****
Charles<     A              Abercrombie<    000070          A10
Kyle<        B              Giddens<        000230          A10
Vincent<     E              Gomez<          000240          A10
Larry<       M              Kuntz<          000250          A10
Kathleen<    L              Miller<         000260          A10
Diane<                      Konyn<          000270          A10
****  End of data  ****




 F1=Help      F2=Zoom      F3=Exit      F4=CRetriev  F5=RFind     F6=RChange
 F7=Up        F8=Down      F9=Swap      F10=Left     F11=Right    F12=Cancel
```

This completes the Basic SQL Prototyping part of the IVP. Press **PF3** repeatedly to return to the ZDT/Db2 main menu.

Step 6. Use the ZDT/Db2 Copy function

Log on to ZDT/Db2, and connect to the same Db2® system specified in Step 1. Define Db2 objects to be used during verification on page 218.

Set the Copy options as shown in Figure 37: Copy Options panel on page 232. Enter '0.3.3' on the ZDT/Db2 main menu to access the *"Copy Options"* panel.

Note that the "Create audit trail" option may not be displayed, or may be displayed with different text, depending on choices made during the product install. The auditing setting does not affect the IVP.

Figure 37. Copy Options panel

```
 Process   Options   Utilities   Help
──────────────────────────────────────────────────────────────────────────────
ZDT/Db2 (DFF2)                    Copy Options                  Global Settings
Command ===> ____________________________________________________________

From Table Concurrency Option:        To Table Locking Option:
   Enter "/" to select option            Locking
   _  Use uncommitted read               1  1. None
                                            2. Share mode
                                            3. Exclusive mode

Processing Options:
   Duplicate key processing              Enter "/" to select option
   1  1. Ignore                          /  Delete existing rows
      2. Update                          _  Ignore RI/Constraint errors
      Max duplicates   ALL               _  Native unicode processing
                                         /  Create audit trail (Option fixed)




 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Return to the ZDT/Db2 main menu by pressing **PF3**. Select the ZDT/Db2 *"Copy function"* by typing '**3.3**' on the ZDT/Db2 main menu and pressing **Enter**.

Type 'HFMCUSR' and 'Employee-Detail' on the *"From Db2 Object"* field as shown in Figure 38: Copy Utility panel on page 232. Press **Enter** to show the COPY *"To"* field in the panel.

Figure 38. Copy Utility panel

```
 Process   Options   Utilities   Help
──────────────────────────────────────────────────────────────────────────────
ZDT/Db2 (DFF2)                    Copy Utility
Command ===> ____________________________________________________________

From Db2 Object:
   Location  . . . . . ________________     Database  . . . ________ (optional)
   Owner . . . . . . . HFMCUSR          +  Table Space . . ________ (optional)
   Name  . . . . . . . Employee-Detail                                        +

   Copy count  . . . . ALL          Number of rows to copy

From Template:
   Data set name . . . ____________________________________________
   Member  . . . . . . ________

Processing Options:
   Template usage                    Enter "/", "A"lways to select option
   3  1. Above                       _  Edit template
      2. Previous                    _  Copy panel values
      3. Generate from table         /  Create audit trail (Option fixed)
      4. Generate/Replace
 F1=Help      F2=Split     F3=Exit      F4=Expand    F7=Backward  F8=Forward
 F9=Swap     F10=Left     F11=Right    F12=Cancel
```

Type 'HFMCUSR' and 'Employee-Detail-Copy' on the ZDT/Db2 COPY *"To"* field panel, and select the *"Batch execution"* option under the *"Processing Options"* field, as shown in Figure 39: Copy Utility panel (destination table) on page 233.

Figure 39. Copy Utility panel (destination table)

```
 Process   Options   Utilities   Help
------------------------------------------------------------------------------
COPY        From HFMCUSR.EMPLOYEE-DETAIL
Command ===> ____________________________________________________________

To Db2 Object:
   Location  . . . . . ________________     Database  . . . _________ (optional)
   Owner . . . . . . . HFMCUSR_________  +  Table Space . . _________ (optional)
   Name  . . . . . . . Employee-Detail-Copy_____________________________  +

To Template:
   Data set name . . . ____________________________________________
   Member  . . . . . . ________

Processing Options:
   Template usage                    Enter "/", "A"lways to select option
   3  1. Above                       _  Edit copy options
      2. Previous                    _  Edit template mapping
      3. Generate from table         /  Batch execution
      4. Generate/Replace


 F1=Help      F2=Split     F3=Exit      F4=Expand    F7=Backward  F8=Forward
 F9=Swap     F10=Left     F11=Right    F12=Cancel
```

Press Enter to display the generated JCL for the ZDT/Db2 batch copy job.

Review the generated JCL. It should not be necessary to change the STEPLIB DD statements if the correct Db2® libraries are specified in the HFM2SSDM macro entry for the currently connected Db2® system.

Submit the job. The expected return code is RC=0, with 48 rows copied.

You can examine the data copied to the *"Employee-Detail-Copy"* table using the ZDT/Db2 editor. Note that the columns are similar to those in the *"Employee-Detail"* table, but are in a different order.

Step 7. Use the ZDT/Db2 Export function

Log on to ZDT/Db2, and connect to the same Db2® system specified in Step 1. Define Db2 objects to be used during verification on page 218.

For this step of the IVP a PDS or PDS/E is required, where the ZDT/Db2 template for the exported data will be stored. The examples given assume a name of

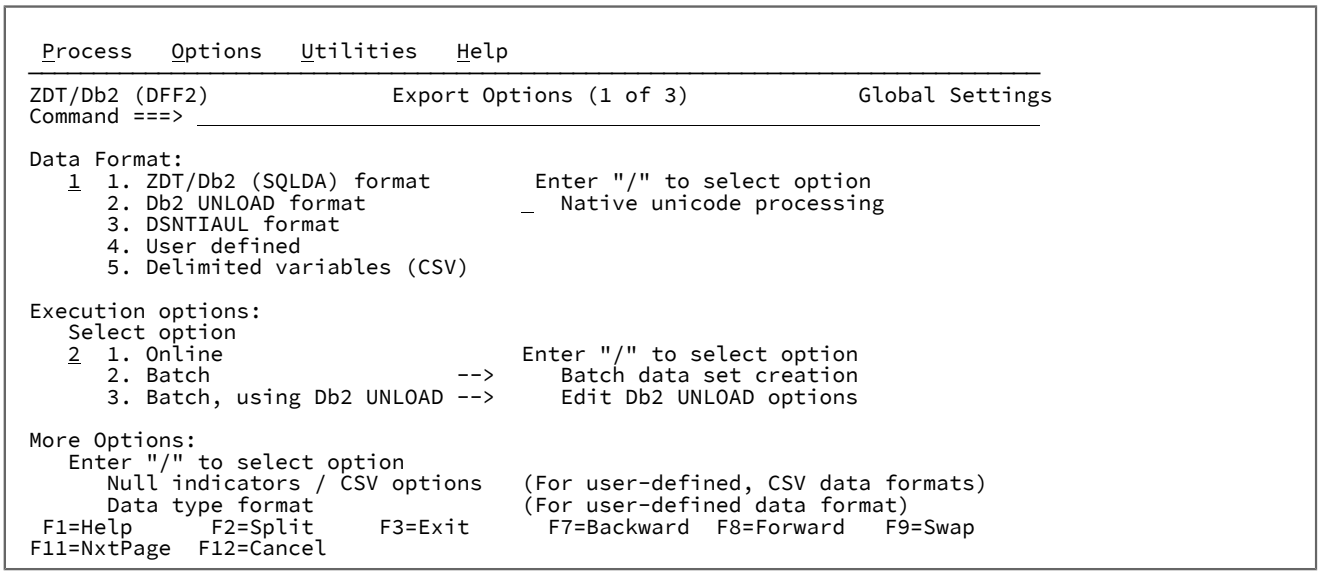
TSO logonid.HFM.TEMPLATE

You can specify any valid data set name that your TSO logon ID has update access to. You can use ISPF option 3.2, or a batch job, to create the data set. Create this data set with these attributes:

```
Organization..:PO
Record format...:FB
Record length...:80
Data set name type   PDS or Library
```

Set the Export options as shown in Figure 40: Export Options (1 of 3) on page 234. Enter '0.3.7' on the ZDT/Db2 main menu to access the *"Export Options panel"*.

Figure 40. Export Options (1 of 3)



Return to the ZDT/Db2 main menu by pressing **PF3**.

Select the ZDT/Db2 Export function by typing '3.7' on the ZDT/Db2 main menu and pressing **Enter**.

Type 'HFMCUSR' and 'Employee-Detail' on the *"From Db2® object"* field as shown in Figure 41: Export Utility From... panel on page 235. Press **Enter** to show the ZDT/Db2 *"Export To"* panel.

Enter these values on the EXPORT *"To partitioned..."* panel:

Export data set name: `EXPORT.EMPDET`

Template data set name: `HFM.TEMPLATE`

Template member name: `EMPUL`

Template usage: `4`

Disposition: `2`

Figure 41. Export Utility From... panel

```
 Process   Options   Utilities   Help
------------------------------------------------------------------------------
ZDT/Db2 (DFF2)                  Export Utility
Command ===> ____________________________________________________________

From Db2 Object:
   Location  . . . . . ________________     Database  . . . _________ (optional)
   Owner . . . . . . . HFMCUSR_________  +  Table Space . . _________ (optional)
   Name  . . . . . . . Employee-Detail_____________________________________ +

   Export count  . . . ALL_______     Number of rows to export

From Template:
   Data set name . . . ____________________________________________
   Member  . . . . . . ________

Processing Options:
   Template usage                     Enter "/", "A"lways to select option
   3  1. Above                        _  Edit options
      2. Previous                     _  Edit template
      3. Generate from table
      4. Generate/Replace
 F1=Help      F2=Split     F3=Exit      F4=Expand    F7=Backward  F8=Forward
 F9=Swap     F10=Left     F11=Right    F12=Cancel
```

Figure 42. Export Utility To... panel

```
 Process   Options   Utilities   Help
------------------------------------------------------------------------------
EXPORT      From HFMCUSR.EMPLOYEE-DETAIL                          Top of data
Command ===> ____________________________________________________________

To Partitioned, Sequential or VSAM Data Set:
   Data set name . . . EXPORT.EMPDET_______________________________________
   Member  . . . . . . ________
   Volume  . . . . . . ______

To Copybook or Template:
   Data set name . . . HFM.TEMPLATE________________________________________
   Member  . . . . . . EMPUL___

Processing Options:
   Template usage                     Disposition
   4  1. Above                        2  1. Old or Reuse
      2. Previous                        2. Mod
      3. Generate from table
      4. Generate/Replace             Enter "/", "A"lways to select option
      5. None. (CSV output)           _  View options
                                      _  Edit template mapping
 F1=Help      F2=Split     F3=Exit      F4=Expand    F7=Backward  F8=Forward
 F9=Swap     F10=Left     F11=Right    F12=Cancel
```

Press **Enter**. An allocation panel for the export data set is displayed. See Figure 43: Allocation panel for Export data on page 236. Type '6' for a *"Non VSAM"* data set and press **Enter** to move to the second allocation panel. See Figure 44: Allocation panel for Export data (2) on page 236.

Note:

- If you execute this IVP step on subsequent occasions you may see a "Template replace" pop-up panel. This is a warning panel displayed when a Z Data Tools/Db2 function is about to replace a template in a member of PDS that already exists. Press **Enter** to continue.
- On subsequent executions of this IVP step the allocation panel for the export data set is not shown.

Figure 43. Allocation panel for Export data

```
 Process   Options   Utilities   Help
------------------------------------------------------------------------------
Allocate PERTHAP.EXPORT.EMPDET                                 Template saved
Command ===> ____________________________________________________________

New Data Set Organization:
   Select option         Instructions
   6 1. KSDS             The above data set does not exist.
     2. ESDS             To define or allocate a new data set select a data
     3. RRDS             set organization and press ENTER or press PF3/EXIT
     4. VRRDS            or PF12/CANCEL to return without allocation.
     5. LDS
     6. Non VSAM
     7. IAM KSDS         For a new data set, enter a data set name
     8. IAM ESDS         below to copy existing allocation attributes.


Existing Data Set:
   Like data set . . . . . ____________________________________________
   Volume serial . . . . . ______


 F1=Help      F2=Split     F3=Exit      F4=CRetriev  F7=Backward  F8=Forward
 F9=Swap     F10=Actions  F12=Cancel
```

Figure 44. Allocation panel for Export data (2)

```
 Process   Options   Utilities   Help
------------------------------------------------------------------------------
Allocate PERTHAP.EXPORT.EMPDET
Command ===> ____________________________________________________________
                                                                More:     +
Specify a model data set, volume, SMS class names, or leave blank for defaults:
   Like data set . . ____________________________________________
   Volume serial . . ______          (Blank for system default volume)
   Device type . . . ________        (Generic unit or device address)
   Data class  . . . ________        (Leave blank for default)
   Storage class . . ________        (Leave blank for default)
   Management class  ________        (Leave blank for default)

Space Requirements:
   Space unit  . . . CYLS             BLKS, TRKS, CYLS, KB, or MB
   Primary units . . 1________        quantity of above units
   Secondary units . 5________        quantity of above units
   Directory blocks  0_____           leave blank for SMS default
   Record format . . FB               if new format: U,F,V, or D, with B,S,A,M
   Record length . . 151
   Block size  . . . _____            physical output block size
   Data set type . . _______          LIBRARY, PDS, BASIC, EXTR, EXTP or LARGE
 F1=Help      F2=Split     F3=Exit      F4=CRetriev  F7=Backward  F8=Forward
 F9=Swap     F10=Actions  F12=Cancel
```

Review the generated JCL. It should not be necessary to change the STEPLIB DD statements if the correct Db2® libraries are specified in the HFM2SSDM macro entry for the currently connected Db2® system.

Submit the job. The expected return code is RC=0, with 48 rows exported.

You can browse the exported data set using the View option in the Z Data Tools Base component.

Step 8. Use the ZDT/Db2 Import function

Log on to ZDT/Db2, and connect to the same Db2® system specified in Step 1. Define Db2 objects to be used during verification on page 218.

Set the Import options as shown in Figure 45: Import Options panel on page 237. Enter '0.3.6' on the ZDT/Db2 main menu to access the "Import Options" panel.

Figure 45. Import Options panel

```
 Process   Options   Utilities   Help
_____________________________________________________________________________
ZDT/Db2 (DFF2)                  Import Options                Global Settings
Command ===> _____________________________________________________________

Import dataset:
   Data format
   1  1. ZDT/Db2 (SQLDA) format
      2. Db2 UNLOAD format
      3. DSNTIAUL format
      4. User defined

Import Options:
   Duplicate key processing             Enter "/" to select option
   1  1. Ignore                         /  Delete existing rows
      2. Update
      Max duplicates   ALL

Auto Commit (Changes):
   Auto-commit count . 0


 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Return to the ZDT/Db2 main menu by pressing PF3. Select the ZDT/Db2 Import function by typing '3.6' on the ZDT/Db2 main menu and pressing Enter.

Enter these values on the ZDT/Db2 Import *"From"* panel (See Figure 46: Import Utility panel on page 238):

From data set name: `EXPORT.EMPDET`

Template data set name: `HFM.TEMPLATE`

Template member name: `EMPUL`

Template usage: `1`

Note: The template data set and member name should match the values you entered in Step 7. Use the ZDT/Db2 Export function on page 233.

Figure 46. Import Utility panel

```
 Process   Options   Utilities   Help
------------------------------------------------------------------------------
ZDT/Db2 (DFF2)                  Import Utility
Command ===> ____________________________________________________________

From Partitioned, Sequential or VSAM Data Set:
   Data set name . . . EXPORT.EMPDET                          
   Member  . . . . . . ________
   Volume  . . . . . . ______
   Start position  . . 1________
   Import count  . . . ALL_____    Number of rows to import

From Copybook or Template:
   Data set name . . . HFM.TEMPLATE                           
   Member  . . . . . . EMPUL___

Processing Options:
   Template usage                    Enter "/", "A"lways to select option
   1  1. Above                       _  Edit options
      2. Previous                    _  Edit template


 F1=Help      F2=Split     F3=Exit      F4=Expand    F7=Backward  F8=Forward
 F9=Swap     F10=Left     F11=Right    F12=Cancel
```

Press **Enter** to display the ZDT/Db2 Import "To" panel.

Type 'HFMCUSR' and 'Employee-Detail-Copy' on the ZDT/Db2 Import *"To"* field panel as shown in Figure 47: Import Utility To... panel on page 238.

Press **Enter** to confirm the Delete all rows option is set (See Figure 48: Confirmation panel for Delete all rows on page 239) and then press **Enter** to start the Import process. The *"Import Report"* panel is displayed. See Figure 49: Import Report panel on page 239.

Figure 47. Import Utility To... panel

```
 Process   Options   Utilities   Help
------------------------------------------------------------------------------
IMPORT      From PERTHAP.EXPORT.EMPDET
Command ===> ____________________________________________________________

To Db2 Object:
   Location  . . . . . ________________     Database  . . . _________ (optional)
   Owner . . . . . . . HFMCUSR_________  +  Table Space . . _________ (optional)
   Name  . . . . . . . Employee-Detail-Copy                                    +

To Template:                      From  PERTHAP.TEMPLATE(EMPUL)
   Data set name . . . ____________________________________________
   Member  . . . . . . ________

Processing Options:
   Template usage                    Enter "/", "A"lways to select option
   3  1. Above                       _  Edit options
      2. Previous                    _  Edit template mapping
      3. Generate from table         _  Batch execution
      4. Generate/Replace
                                     _  Use REXX proc
                                        REXX proc name  . . ________
 F1=Help      F2=Split     F3=Exit      F4=Expand    F7=Backward  F8=Forward
 F9=Swap     F10=Left     F11=Right    F12=Cancel
```

Figure 48. Confirmation panel for Delete all rows

```
Delete Confirmation
You have requested to delete all existing rows of the
DB2 object:
    HFMCUSR.Employee-Detail-Copy

Press ENTER to proceed with the DB2 table import.
Press EXIT or CANCEL to abort import.


  F1=Help      F2=Split     F3=Exit      F4=Expand    F7=Backward
  F8=Forward     F9=Swap     F12=Cancel
```

Figure 49. Import Report panel

```
 Process   Options   Utilities   Help
──────────────────────────────────────────────────────────────────────────────
ZDT/Db2 (DFF2)                  Import Report                48 rows imported
Command ===> ____________________________________________________________
                                                                More:     +
Source and target:
   Source dataset  . . : 'PERTHAPC.EXPORT.EMPDET'
   Db2 object  . . . . : "HFMCUSR.Employee-Detail-Copy"
Rows imported:
   Records read (Start, First error) . . . : 48 (1)
   Committed (successful) changes  . . . . : 48
   Uncommitted (unsuccessful) changes  . . : 0
   Total successful changes  . . . . . . . : 48
Row update not selected



Errors:
   Duplicate keys/Max duplicates . . . . . : 0/NO LIMIT
   Updates . . . . . . . . . . . . . . . . : 0
   Record selection  . . . . . . . . . . . : 0
   Dropped (REXX proc) . . . . . . . . . . : 0
 F1=Help      F2=Split     F3=Exit      F4=Expand    F7=Backward  F8=Forward
 F9=Swap     F10=Left     F11=Right    F12=Cancel
```

You can use the ZDT/Db2 editor to display the contents of the HFMCUSR.*"Employee-Detail-Copy"* table.

Step 9. Use the ZDT/Db2 Object List utility

Log on to ZDT/Db2, and connect to the same Db2® system specified in Verifying the customization of ZDT/Db2 on page 218.

Select the ZDT/Db2 Object List utility by typing '3.4' on the ZDT/Db2 main menu panel and press **Enter**.

Type 'HFMCI*' in the Name field and change the *"Object type"* value to '1'. This will show all databases that start with *"HFMCI"*. See Figure 50: Object List Utility panel on page 240

Figure 50. Object List Utility panel

```
 Process   Options   Utilities   Help
------------------------------------------------------------------------------
ZDT/Db2 (DFF2)                Object List Utility
Command ===> ____________________________________________________________
                                                                More:     +
   blank Display object list                 P Print object list

Object Identification Criteria:
   Location  . . . . . . . . ________________    Enter * for list
   Owner . . . . . . . . . . ________________  +
   Name  . . . . . . . . . . HFMCI*                                        +
   Database/collect/schema ____________________________________________    +

Additional Selection Criteria:
 . . . . . . . . . . . . . . . . . . . . Enter * to list catalog columns
   Column  . . . . . . . . . ____________________________________  +
   Operator  . . . . . . . . ____________
   Value . . . . . . . . . . ____________________________________________  +

Processing Options:
   Object Type                                Enter ? to list all object types
   1   1. Database          8.  Package           15. Trigger
 F1=Help      F2=Split     F3=Exit      F4=Expand    F7=Backward  F8=Forward
 F9=Swap      F10=Left     F11=Right    F12=Cancel
```

Press Enter to display the list of databases. See Figure 51: Databases panel on page 240

Figure 51. Databases panel

```
 Process   Options   Utilities   Help
------------------------------------------------------------------------------
ZDT/Db2 (DFF2)                    Databases                        Top of 1
Command ===> ____________________________________________________  Scroll CSR
1 rows fetched                                                     Format TABL
                          STORAGE  BUFFER     INTERNAL AUTHID
       DATABASE DATABASE  GROUP    POOL       DATABASE THAT     TYPE OF  GROU+
SEL    NAME     CREATOR   NAME     NAME     IDENTIFIER CREATED  DATABASE MEMB+
       *________ *________ *________ *________      ____* *________ *_______ *
----   #1--+--- #2--+--- #3--+--- #4--+---     ----#5 #7--+--- #23-+--- #11-+
****  Top of data  ****
____    HFMCIVD  PERTHAP  SYSDEFLT BP0           16716 PERTHAP
****  End of data  ****









 F1=Help      F2=Zoom      F3=Exit      F4=CRetriev  F5=RFind     F7=Up
 F8=Down      F9=Swap      F10=Left     F11=Right    F12=Cancel
```

Type '?' against the HFMCIVD database entry and press Enter. See Figure 52: Object List Line Commands panel on page 241.

Figure 52. Object List Line Commands panel

```
 Process   Options   Utilities   Help
------------------------------------------------------------------------------
ZDT/Db2 (DFF2)            Object List Line Commands              Row 1 of 15
Command ===> _____________________________________________       Scroll CSR

Type S against the required line command and press Enter.

Sel Command  Description

___ A        Alter database
___ CR       Create a database
___ CRS      Create a table space
___ DR       Drop database
___ DS       Show database structure
___ G        Grant privileges
___ I        Details about database
___ ICS      Show recovery information (image copies)
___ P        Show privileges
___ ROW      Show all columns for this row
___ S        Show table spaces
___ SG       Show storage group
 F1=Help     F2=Split    F3=Exit     F4=CRetriev  F7=Backward  F8=Forward
 F9=Swap     F10=Actions F12=Cancel
```

Type an 'S' against the "DS" entry to show the database structure. Press **Enter**. See Figure 53: Database Structure for HFMCIVD panel on page 241.

Figure 53. Database Structure for HFMCIVD panel

```
 Process   Options   Utilities   Help
------------------------------------------------------------------------------
ZDT/Db2 (DFF2)          Database Structure for HFMCIVD               Top of 15
Command ===> _____________________________________________       Scroll CSR
15 rows fetched                                                  Format TABL

              OBJECT                    OBJECT
SEL    TYPE   NAME                      OWNER    DBID   PSID   OBID
       *      *                         *        *      *      *
----   #10-   #12-+----1----+----2----+- #13-+--- ----#5 ---#19 ---#20
****  Top of data  ****
_____   D---   HFMCIVD----------------    PERTHAP   379      0      0
_____    S       HFMCIVSC                PERTHAP   379      6      0
_____     T        Employee-Detail-Copy   HFMCUSR   379      0     26
_____    S       HFMCIVSD                PERTHAP   379      2      0
_____     T        Department-Information HFMCUSR   379      0     11
_____      I         Dept Info Index N2   HFMCUSR   379      0     15
_____      I         Dept Info Index N3   HFMCUSR   379      0     17
_____      I         Dept Info Index U1   HFMCUSR   379      0     13
_____    S       HFMCIVSE                PERTHAP   379      4      0
_____     T        Employee-Detail        HFMCUSR   379      0     18
_____      I         Employee Index N2    HFMCUSR   379      0     25
_____      I         Employee Index U1    HFMCUSR   379      0     23
_____    S       HFMCIVSR                PERTHAP   379      8      0
_____     T        Department-Info-Errors HFMCUSR   379      0     29
_____    S       HFMCIVSS                PERTHAP   379     10      0
_____     T        Employee-Detail-Errors HFMCUSR   379      0     30
 F1=Help      F2=Zoom      F3=Exit      F4=CRetriev  F5=RFind     F7=Up
 F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
```

Press **PF7/PF8** to scroll the content and **PF3** repeatedly to return to the ZDT/Db2 main menu.

This completes the ZDT/Db2 IVP.

Chapter 20. Re-installing ZDT/Db2 after migration of a Db2® system

When migration of an existing Db2® system to a new version of Db2® is complete, you should revisit the following ZDT/Db2 installation tasks:

- Granting access to the Db2 catalog (required) on page 161
- Binding Db2 (required) on page 172

Verifying ZDT/Db2 has access to any new/changed Db2® catalog tables.

Depending on the method used to grant access to the Db2® catalog tables, you made need to execute additional grant statements, or drop and re-create views on the Db2® catalog tables.

The important point here is that you should start with the samples for the new version, rather than the old version.

Binding Db2®

Locate the sample bind plan and bind package jobs for the new version of Db2®. Customize these jobs as required and run them.

The important point here is that you should start with the samples for the new version, rather than the old version.

Note: Z Data Tools provides sample bind jobs only for supported versions of Db2® running in new-function mode (NFM).

Part III. Customizing Z Data Tools IMS™ Component

Chapter 21. Preparing to customize ZDT/IMS

Before you can install and customize ZDT/IMS, you must have installed the Z Data Tools Base function. These topics assume you have installed ZDT/IMS into the same target and distribution libraries as Z Data Tools Base function. You could encounter problems using Z Data Tools if you do not install ZDT/IMS in the same libraries as the Z Data Tools Base function.

Furthermore, before you can use ZDT/IMS you must customize ZDT/IMS and the operating environment. You may also need to customize the Z Data Tools Base function. For information on customizing the Z Data Tools Base function, see Customizing Z Data Tools on page 22.

Table 40: Summary of steps for customizing ZDT/IMS and the operating environment on page 244 lists the customization tasks you can perform for ZDT/IMS. Review the following sections in this chapter, and read the referenced sections in the table, to determine the customization you need to do.

Checklist for installing and customizing ZDT/IMS

Table 40. Summary of steps for customizing ZDT/IMS and the operating environment

	Description
__ 1	Concatenate ZDT/IMS libraries to the LINKLIST. See Concatenating libraries to the LINKLIST on page 258.
__ 2	Modify the TSO logon procedure See Modifying the TSO logon procedure on page 258.
__ 3	Add ZDT/IMS to the ISPF menu. See Adding ZDT/IMS to your ISPF menu on page 261.
__ 4	Define ZDT/IMS in an ISPF command table. See Defining ZDT/IMS in an ISPF command table on page 261.
__ 5	Customize IMS™ to support the use of dynamic PSBs. See Customizing IMS to support the use of dynamic PSBs on page 262.
__ 6	Customize the ZDT/IMS installation options module See Customizing the ZDT/IMS installation options module on page 264.
__ 7	Tailor the job control skeletons. See Tailoring the job control skeletons on page 279.
__ 8	Customize the ZDT/IMS security exit, HFM1SXT. See Customizing the ZDT/IMS security exit on page 289.
__ 9	Provide access to protected functions using RACF®. See IMS subsystems and ZDT/IMS functions access control facility on page 284.
__ 10	Decide how to customize the ZDT/IMS audit facility. See Alternatives for controlling ZDT/IMS auditing on page 258.

Table 40. Summary of steps for customizing ZDT/IMS and the operating environment (continued)

	Description
__ 11	Customize ZDT/IMS for national languages. See Customizing ZDT/IMS for national languages on page 310.

Supported databases

ZDT/IMS supports the following databases:

HDAM

Hierarchical Direct Access Method

HIDAM

Hierarchical Index Direct Access Method

HISAM

Hierarchical Index Sequential Access Method

SHISAM

Simple Hierarchical Index Sequential Access Method

HSAM

Hierarchical Sequential Access Method (BSAM or QSAM only)

SHSAM

Simple Hierarchical Sequential Access Method (BSAM or QSAM)

INDEX

Index database

DEDB

Data Entry Database (Fast Path)

MSDB

Main Storage database (Fast Path)

PHDAM

Partitioned Hierarchical Direct Access Method (HALDB)

PHIDAM

Partitioned Hierarchical Index Direct Access Method (HALDB)

PSINDEX

Partitioned Secondary Index (HALDB)

Region types

ZDT/IMS offers two modes of accessing IMS™ databases - DLI mode and BMP mode.

DLI mode

When a Z Data Tools/IMS function is run in DLI mode, a DL/I batch processing program issues the DL/I calls. The DL/I batch processing program executes as a subtask of the ZDT/IMS function.

You use DLI mode when databases are offline.

If you plan to run ZDT/IMS functions in DLI mode, you must specify the following details when you are customizing the ZDT/IMS installation options module:

- The IDs of the IMS™ subsystems you plan to access in DLI mode.
- The DLIBATCH parameters that are to be passed to the IMS™ region controller when accessing each subsystem.
- Whether ZDT/IMS functions which are using PSBs that have update intent use an IMS™ log when accessing each subsystem and, if they do use a log, whether the log is kept when the function ends.
- The name pattern that ZDT/IMS functions use to generate IMS™ log data set names when accessing each subsystem.
- The names of the DFSVSAMP, RESLIB, IMS™ macros and staging ACBLIB data sets for each subsystem.
- The names of the PSB, DBD and Template libraries for each subsystem.
- Various processing options that you want the functions to use when accessing each subsystem in DLI mode.

For information on how you do this, see Customizing the ZDT/IMS installation options module on page 264.

BMP mode

When a Z Data Tools/IMS function is run in BMP mode, a batch message processing (BMP) program issues the DL/I calls. The BMP executes as a subtask of the ZDT/IMS function.

You use BMP mode when databases are online.

If you plan to run ZDT/IMS functions in BMP mode, you must specify the following details when you are customizing the ZDT/IMS installation options module:

- The IDs of the IMS™ subsystems you plan to access in BMP mode.
- The IMSBATCH parameters that are to be passed to the IMS™ region controller when accessing each subsystem.
- The names of the RESLIB, IMS™ macros and DOPT ACBLIB data sets for each subsystem.
- The names of the PSB, DBD and Template libraries for each subsystem.
- Various processing options that you want the functions to use when accessing each subsystem in BMP mode.

For information on how you do this, see Customizing the ZDT/IMS installation options module on page 264.

PSB types

ZDT/IMS supports the use of two types of PSBs - dynamic PSBs and static PSBs.

Dynamic PSBs

Dynamic PSBs are temporary PSBs generated by ZDT/IMS when a function is started and deleted when it finishes. ZDT/IMS generates a PSB that is sensitive to all the segments in the specified DBD. If you don't want your function to be sensitive to all segments in a physical DBD, you can use a logical DBD that only includes the physical segments that you want the function to process. For the Extract and Load functions, the PSB is generated with PCBs for the primary database and all logically related databases.

Advantages in using a dynamic PSB

- A database administrator is not required to generate PSBs for the ZDT/IMS functions.
- ZDT/IMS generates a PSB that best suits the ZDT/IMS function.

Disadvantages in using a dynamic PSB

- The same processing option (PROCOPT) is used for all segments in the DBD. For example, you cannot prevent the Edit from updating one segment type in the DBD.
- A dynamic PSB can access any database in the subsystem, so unless special measures are taken they can be a security risk. See Controlling access to databases by ZDT/IMS functions on page 248 for a description of some special measures that can be taken to remove this security risk.
- The PSB is not available for use if Batch Backout is required.

For information on how to customize IMS to support dynamic PSBs, see Customizing IMS to support the use of dynamic PSBs on page 262.

Static PSBs

Static PSBs are existing PSBs generated by the database administrator.

Advantages of using a static PSB

- It is easier to exclude segments from processing.
- Processing options (PROCOPTs) can be used to limit the update capability of the user.
- They are easily secured.
- The PSB is available for use if Batch Backout is required.

Disadvantages of using a static PSB

- A database administrator must generate the PSB.

IMS™ region controller parameters

ZDT/IMS functions that access databases invoke the IMS™ region controller as a subtask. You specify the default values for the parameters passed to the IMS™ region controller when you customize the ZDT/IMS installation options module.

In the ZDT/IMS installation options module customization, you specify defaults for the following options:

- Whether or not ZDT/IMS functions use Database Recovery Control (DBRC) when they are run in DLI mode.
- Whether or not ZDT/IMS functions use an Internal Resource Lock Manager (IRLM) when they are run in DLI mode.
- The 4-byte z/OS® subsystem name assigned to the Internal Resource Lock Manager.
- Whether or not dynamic backout is to be performed when an IMS™ pseudo-abend occurs in a Z Data Tools/IMS function running in DLI mode.
- (If the IMS™ subsystem is part of a Remote Site Recovery (RSR) complex) the Global Service Group (GSG) name and the Transport Manager Instance (TMI) name that ZDT/IMS functions are to use when they are run in DLI mode.
- The parallel DL/I option that ZDT/IMS functions are to use when they are run in BMP mode.
- The number of Fast Path database buffers to be made available in the Common Service Area (CSA) when a Fast Path region is activated.
- The number of additional page-fixed buffers to be made available to a Fast Path region if the normal allotment is used.
- The maximum number of locks that a Z Data Tools/IMS function is allowed to hold at one time.

In the ZDT/IMS installation options module customization, you also specify whether or not the user can override the values that you specify for the above options.

For more information on specifying the IMS™ region controller parameters, see Specifying the parameters passed to the IMS region controller on page 267.

Security

This section outlines how you can protect IMS™ resources that can be accessed through ZDT/IMS functions.

Controlling access to databases by ZDT/IMS functions

Security administrators can control the access that users have to databases when using ZDT/IMS functions.

Different methods of controlling access are required depending on whether the functions run in DLI or BMP mode.

For functions that run in DLI mode, you must control users' access to the database data sets. There are two ways you can do this:

1. You can use RACF® (or an equivalent security product) data set profiles. At most installations, the database data sets are already protected by RACF® data set profiles. If this is not the case, you need to define RACF® profiles for all database data set resources. One advantage of this method is that the protection that it provides is not just restricted to when the access is through ZDT/IMS functions.
2. You can use the ZDT/IMS Database Access Control facility. For information on how to customize this facility, see The Database Access Control facility on page 282.

For functions that run in BMP mode, there are several different ways of controlling users' access to databases.

The standard way is to use Resource Access Security (RAS) and the IIMS and JIMS RACF® security classes to control which PSBs each user can use. An advantage of this method is that the protection that it provides is not just restricted to when the access is through ZDT/IMS functions. However, the method has one serious limitation: it can only be used to control access by functions that use static PSBs. One of the following methods must be used to control access by functions that use dynamic PSBs.

1. You can use the ZDT/IMS Database Access Control facility.

   For information on how to customize this facility, see The Database Access Control facility on page 282.
2. You can provide a version of the HFM1SXT security exit that prevents users accessing databases that they don't have authority to access.

   For information on writing your own version of HFM1SXT, see Customizing the ZDT/IMS security exit on page 289.

Both of these methods can also be used to control access by functions that use static PSBs.

Controlling access to IMS™ subsystems and ZDT/IMS functions

Through the IMS™ subsystem and ZDT/IMS Function Access Control facility, you can control access to these resources:

- The update or read-only functions
- Individual ZDT/IMS functions
- Individual IMS™ subsystems by the update or read-only functions
- Individual IMS™ subsystems by individual functions

A number of functions in ZDT/IMS are protected by default. They are listed in Table 48: Protected ZDT/IMS Functions on page 284. If you want to use these functions, you must give your users access to them using RACF® (or an equivalent security product).

For more information, see IMS subsystems and ZDT/IMS functions access control facility on page 284.

Resource access security and AGN security considerations

Resource access security (RAS) prevents an application program that is running in a dependent region from using IMS™ resources unless it is authorized to do so.

This means that an ZDT/IMS function running in BMP mode is prevented from using a PSB unless the user is authorized to use that PSB. This in turn means that users cannot use a dynamic PSB unless they are authorized to use all of the dynamic PSBs that ZDT/IMS can use. See Declaring the dynamic PSBs on page 262.

You might have to customize the IIMS and JIMS class profiles to provide ZDT/IMS users with the access they require. No special customization is required for ZDT/IMS to be able to access subsystems that use RAS.

Before IMS™ Version 9, Application Group Names (AGNs) were used to secure dependent regions. In IMS Version 9 RAS superseded AGN security. Support for AGNs was removed from IMS in Version 10.

ZDT/IMS still allows the user to specify an AGN on the entry panel or in batch JCL. However, if the IMS™ subsystem you are accessing is Version 10 or later, IMS ignores the specified AGN value.

If you do use ZDT/IMS to access IMS™ subsystems that use AGNs to secure BMP regions, you must specify that the subsystem uses AGN security when customizing the ZDT/IMS installation options module. You also have the option of specifying the AGNs that you want ZDT/IMS to use. For information on both of these tasks, see Specifying AGNS on page 272.

RACF® PADS security considerations

Program Access to Data Sets (PADS) allows users or groups of users to access data sets with higher authority than they would normally have, but only while running a specified program.

To run ZDT/IMS functions, users generally require more access to IMS™ data sets than they would ordinarily have. If you want this access to be restricted to when they are using ZDT/IMS functions, you should use PADS.

To run a Z Data Tools/IMS function in DLI mode, users must be given UPDATE access to the RECON data sets, and if you want this function to use an IMS™ log, users must be given ALTER access to the IMS™ log data set.

When ZDT/IMS functions only require READ access to an IMS™ data set, you may choose not to restrict the access to when ZDT/IMS functions are used. However, when ZDT/IMS functions require UPDATE or ALTER access to the data set, as is the case for the RECON and IMS™ log data sets, the consequences of not restricting access can be more serious. So, access to these data sets should always be provided through PADS.

If PADS is used in the environment in which ZDT/IMS functions are run, you should take the following actions:

- Set PADS=Y when you customize the ZDT/IMS installation options module.
- Identify all the subsystems in the ZDT/IMS installation options module with ACBMGMT=ACBLIB or USEDDL=N specified.
- Create a PDSE program library for each subsystem with the above settings, and set the DYNPSB parameter for the subsystem to the name of the library.

For more information on how to code the PADS and DYNPSB parameters, see ZDT/IMS options on page 457.

IMS™ subsystem access

ZDT/IMS allows you to control the way in which your IMS™ subsystems are accessed as described in the following sections.

For more information on these controls, see Controlling access to the subsystems on page 268 and ZDT/IMS options on page 457.

Read only subsystems

ZDT/IMS allows you to define a subsystem as read-only. If a subsystem is defined as read-only, only the ZDT/IMS non-update functions - Browse, Batch Browse, Extract and Print - can be run against databases in the subsystem; ZDT/IMS prevents users running the update functions - Edit, Batch Edit, Load, Initialize and Delete/Define.

BMP mode only and DLI mode only subsystems

ZDT/IMS allows you to define a subsystem as BMP mode only or DLI mode only. If a subsystem is defined as BMP mode only, databases in the subsystem can only be accessed in BMP mode. If a subsystem is defined as DLI mode only, databases in the subsystem can only be accessed in DLI mode.

Static PSB only and dynamic PSB only subsystems

ZDT/IMS allows you to define a subsystem as static PSB only or dynamic PSB only. If a subsystem is defined as static PSB only, then only static PSBs can be used to access databases in the subsystem. If a subsystem is defined as dynamic PSB only, then only dynamic PSBs can be used to access databases in the subsystem.

Dynamic allocation only subsystems

ZDT/IMS allows you to define a subsystem as dynamic allocation only. If a subsystem is defined as dynamic allocation only, ZDT/IMS enforces the use of the database data sets specified in the DFSMDA dynamic allocation modules, when accessing databases in the subsystem in DLI mode.

Enforcing the usage and retention of IMS logs

ZDT/IMS allows you to force ZDT/IMS functions to use an IMS™ log when they are run in DLI mode and use a PSB that has update intent. ZDT/IMS also allows you to force the user to keep the log when the function ends.

Avoiding resource contention

This section describes how to avoid resource contention between ZDT/IMS Edit/Browse sessions and other concurrently executing processes at your installation.

When a user starts an Edit or Browse session, ZDT/IMS starts a batch message processing (BMP) program or a DL/I batch processing program. Ordinarily, the BMP or DL/I batch processing program continues to run until the user exits from the Edit or Browse session. These programs have periods of inactivity when they are waiting for a user response and periods of activity when they are processing the user response.

We need to consider measures you can take to reduce resource contention when the programs are active and when they are inactive. However, the latter measures are more important because, for the majority of the time, the programs just wait for a user response.

Inactive Edit and Browse sessions

Two types of resource contention are considered here:

- Lock contention
- Contention with processes that use the /DBRECOVERY command

These are discussed in turn.

Lock contention

Lock contention occurs when a concurrently executing program attempts to acquire a lock that the inactive Edit or Browse session is holding. The way to avoid it, is to get ZDT/IMS to release all locks before it returns control to the user.

There are two types of locks that need to be considered:

- Locks for unmodified data
- Locks protecting updates

Before returning control to the user, ZDT/IMS issues a RLSE call which releases all locks held for unmodified data, so we can restrict our attention to locks protecting updates.

Only Edit sessions hold locks protecting updates. If you want ZDT/IMS to release any locks protecting updates that the Edit session holds, before returning control to the user, then you should set AUTOSAVE=Y, EDITFREQ=1 and UAUTOSAV=N when you customize the ZDT/IMS installation module. You can specify these settings for some IMS™ subsystems, so all locks protecting updates are released when the user is editing databases in those subsystems, and specify different settings for other IMS™ subsystems, where it is not as critical that these locks are released.

You can also specify different settings for BMP and DLI mode. For more information on how to code these parameters, see Specifying selected processing options on page 269 and ZDT/IMS options on page 457.

Note: If you set AUTOSAVE=Y, EDITFREQ=1 and UAUTOSAV=N, editing changes are committed when the user presses Enter. Thus these settings preclude users from using the UNDO and CANCEL commands to discard their editing changes.

Contention with processes that use the /DBRECOVERY command

The contention described in this section occurs when the Edit or Browse session runs in BMP mode.

A process that uses the /DBRECOVERY command to deallocate a database cannot be run while an inactive Edit or Browse BMP is accessing this database. The process can be run when the user exits out of the Edit or Browse session. However, at most installations, processes that use the /DBRECOVERY command to deallocate databases are run at night after the user has gone home, so they won't always be available to do this.

Some utilities, such as the Online Reorganization Facility, support pausing BMPs and this allows them to use the /DBRECOVERY command to deallocate the databases that the BMPs are accessing. However, a BMP can only be paused when it issues a checkpoint, so if the inactive Edit or Browse BMP is to be paused, it would have to be awakened periodically to issue a checkpoint.

There are two installation options that you can use to stop this type of resource contention occurring - a time-based checkpointing option and a timeout option.

If the time-based checkpointing option is selected, then the Edit/Browse BMP issues checkpoints while it is waiting for a user response. With this option selected, utilities that support pausing BMPs are able to pause inactive Edit/Browse BMPs, allowing them to use the /DBRECOVERY command to deallocate the databases that the Edit/Browse BMPs are accessing.

To select the time-based checkpointing option for an IMS™ subsystem, use the CHKPINTVL parameter on the HFM1POPI macro statement to specify the time interval between checkpoints. For information on how to code this parameter, see ZDT/IMS options on page 457.

If the timeout option is selected, then the Edit/Browse session is timed out after a period of inactivity. With this option selected, you are able to run processes that use the /DBRECOVERY command to deallocate databases that inactive Edit/Browse BMPs are accessing, once the Edit/Browse BMPs are timed out.

To select the timeout option for an IMS™ subsystem, use the TIMEOUTI parameter on the HFM1POPI macro statement to specify the time period that a user has to respond before an Edit/Browse BMP is timed out. For information on how to code this parameter, see ZDT/IMS options on page 457.

Active Edit and Browse sessions

Typically locks held while the Edit or Browse session is active are not responsible for any resource contention, because the locks are usually held for a very short period of time and a very small number of resources are locked at any time. There are exceptions, however. The CHANGE ALL and REPEAT ALL commands can hold a large number of locks for long periods of time, so their usage needs to be controlled. If you want to control their usage for an IMS™ subsystem, you should set AUTOSAVE=Y, and UAUTOSAV=N and use the CHGAFREQ parameter to specify the checkpoint frequency that is to be used during Change All and Repeat All operations. For more information on how to code these parameters, see Specifying selected processing options on page 269 and ZDT/IMS options on page 457. Also, you can use the MAXGN parameter to limit the number of DL/I calls that can be issued when searching a database. For more information on how to code this parameter, see Controlling access to the subsystems on page 268 and ZDT/IMS options on page 457.

IMS™ management of ACBs

This topic describes how ZDT/IMS operates when IMS™ is configured to manage the runtime application control blocks (ACBs).

When management of ACBs by IMS™ is enabled:

- The IMS™ catalog is the trusted source for DBDs and PSBs.
- IMS™ mostly no longer requires DBD, PSB, and ACB libraries.

In keeping with this, ZDT/IMS retrieves the PSBs and DBDs from the IMS catalog.

The IMS Catalog API (DFS3CATQ) is used to retrieve these control blocks.

Restriction: ZDT/IMS requires IMS APAR PH40873/UO06108 to access logical databases in an IMS managed ACB environment.

Configuring ZDT/IMS when IMS™ management of ACBs is enabled

Set ACBMGMT=CATALOG on the HFM1POPI macro statement to specify that IMS™ management of ACBs is enabled.

Also set the following HFM1POPI parameters:

- Use the BSDSHLQ parameter to specify the high level qualifier for the IMS™ bootstrap data set.
- Use the DFSDF parameter to specify the 3-character suffix of the DFSDFxxx member of the IMS™ PROCLIB data set that contains the settings and attributes of the IMS™ catalog.
- Use the PROCLIB parameter to specify the name of the IMS™ PROCLIB data set that contains the required DFSDFxxx member.
- Use the REGCATLG parameter to specify whether the IMS catalog is registered with DBRC.

For more information on how to code these parameters, see ZDT/IMS options on page 457.

Note: The DBDLIBn, DBRC, and PSBLIBn parameters are not used when IMS management of ACBs is enabled.

Dynamic PSB support when IMS™ management of ACBs is enabled

Dynamic PSBs are supported differently when IMS management of ACBs is enabled.

When USEDDL is set to Y on the HFM1POPI macro statement, dynamic PSBs are generated by submitting Data Definition Language (DDL) SQL statements to the IMS Data Definition utility (DFS3ID00). When they are no longer required, they are deleted in the same way — by submitting DDL SQL statements to the IMS Data Definition utility.

Prerequisites: Support for running the IMS Data Definition utility in a BMP region was delivered with APAR PH45367/PTF UI81894 and enabled by IMS 15.3. Support for running it in a DL/I batch region was delivered with APAR PH51761/PTF UI91330 and enabled by IMS 15.4. Ensure you are at or above these levels to generate and delete dynamic PSBs this way.

When USEDDL is set to N on the HFM1POPI macro statement, ZDT/IMS generates dynamic PSBs by submitting macro instructions to the PSB generation utility and then uses the ACB Maintenance utility and the IMS Catalog Populate utility (DFS3PU00) to add the PSB to the IMS catalog. However, this method of generating dynamic PSBs has been deprecated, so using it is not recommended.

For more information on how dynamic PSBs are generated when IMS management of ACBs is enabled, see USEDDL on page 507.

Templates

ZDT/IMS uses templates to format database segments into their individual fields. One template is required for each database.

Templates are used by the Edit, Browse, Extract, Print, Batch Edit, and Batch Browse functions.

Static templates

Static templates are created from COBOL copybooks or PL/I include members that define the fields of the database segments. They must be created manually and kept up to date.

Use the Template function (option 4.1), to create individual static templates from scratch or to update individual static templates. The function runs in the foreground.

Use the Template Update Utility (option 4.4), to update one or more static templates in either foreground or batch.

Use the TPLLIBn parameters on the HFM1POPI macro statement to specify the names of the data sets that contain the static templates that you want the above functions to use. If you do not want users to override the values you specify for these parameters, set UTPLLIB=N on the HFM1POPI macro statement.



Note: Static templates are simply called templates in the *Z Data Tools User's Guide and Reference for IMS Data*.

Dynamic templates

Installations that have added application-defined fields to their DBDs have the option of using dynamic templates. Dynamic templates are templates that ZDT/IMS dynamically generates from the field definitions in the DBDs.

ZDT/IMS gets the field definitions from the IMS catalog when ACBs are managed by IMS, or from the DBD libraries for the subsystem when ACBs are managed by your installation. The advantage of dynamic templates over static templates is that they don't have to be created manually nor do they have to be kept up to date.

To enable dynamic generation of templates, set DYNTPLT=Y on the HFM1POPI macro statement. When dynamic generation of templates is enabled, functions that have requested a new or an existing view or criteria set attempt to generate a template for the database that the function is accessing. If the function is unable to generate a dynamic template for the database, it reverts to using the static template.

The Edit, Browse, Print, Batch Edit, and Batch Browse functions use new and existing views. The Extract function uses new and existing criteria sets.

How static and dynamic templates are used

How the static or dynamic template is used depends on whether the function requests a new or an existing view or criteria set.

If the function requests a new view or criteria set, the static or dynamic template is used to format database segments into their individual fields. The function creates a temporary view or criteria set that is based on the static or dynamic template. You can save the temporary view or criteria set for future use.

If the function has requested an existing view or criteria set, how the static or dynamic template is used depends on which *usage rules* have been specified.

View and criteria set usage rules

Usage rules specify whether a view or criteria set can be used with a database and, if it can, whether the function uses the view or criteria set specified by the user or a composite view or criteria set created by adding the view or criteria set's selection information to the static or dynamic template.

When a composite view or composite criteria set is used, the template is used to format database segments into their individual fields. Otherwise, the view or criteria set is used to format database segments into their individual fields.

You can specify one set of rules for when the function generates a template for the database and another set of rules for when it doesn't.

When the function generates a template

The VCURUDT parameter specifies the usage rules that apply to views and criteria sets when the function generates a template for the database.

Use the VCURUDT parameter on the HFM1POPI macro statement to set one of the following values:

C

A view or criteria set can be used with a database only if it provides one or more layouts for each segment in the database. If it does provide one or more layouts for each segment in the database, the function uses a composite view or criteria set that is created by adding the view or criteria set's selection information to the template that was generated for the database. So, the generated template is used to format the database segments into their individual fields.

R

A view or criteria set can be used with a database only if it has the same fields as the template that was generated for the database. If it does have the same fields as the generated template, the function uses the view or criteria set that is specified by the user. So, the view or criteria set is used to format the database segments into their individual fields. However, as the view or criteria set and the generated template have the same fields, this is equivalent to using the template to format the database segments into their individual fields.

VCURUDT=C is the default setting and the recommended setting.

When the function does not generate a template

The VCURULE parameter specifies the usage rules for views and criteria sets that apply when the function does not generate a template for the database.

Use the VCURULE parameter on the HFM1POPI macro statement to set one of the following values:

C

A view or criteria set can be used with a database only if it provides one or more layouts for each segment in the database. If it does provide one or more layouts for each segment in the database, the function uses a composite view or criteria set that is created by adding the view or criteria set's selection information to the static template for the database. So, the static template for the database is used to format the database segments into their individual fields.

R

A view or criteria set can be used with a database only if it has the same fields as the static template for the database. If it does have the same fields as the static template for the database, the function uses the view or criteria set that is specified by the user. So, the view or criteria set is used to format the database segments into their individual fields. However, as the view or criteria set and the static template for the database have the same fields, this is equivalent to using the template to format the database segments into their individual fields.

S

A view or criteria set can be used with a database only if it provides one or more layouts for each segment in the database. If it does provide one or more layouts for each segment in the database, the function uses the view or criteria set that is specified by the user. So, the view or criteria set is used to format the database segments into their individual fields.

The static template for the database is not used.

T

A view or criteria set can be used with a database only if both of these conditions are true:

- The view or criteria set provides one or more layouts for each segment in the database.
- The view or criteria set was created for the database OR the view or criteria set and the static template for the database have the same fields.

If the view or criteria set satisfies these conditions, the function uses the view or criteria set that is specified by the user. So, the view or criteria set is used to format the database segments into their individual fields.

Note: VCURULE=T is the default setting, but VCURULE=C is the recommended setting.

Alternatives for making ZDT/IMS available

You can make ZDT/IMS available to your users either by concatenating HFM.SHFMMOD1 to your linklist, or adding it to the STEPLIB DD statement in your TSO logon procedure.

To make ZDT/IMS readily available from ISPF, configure your ISPF environment as described in Customizing the operating environment for Z Data Tools on page 30.

Concatenating libraries to the LINKLIST

To make ZDT/IMS commonly available, add the HFM.SHFMMOD1 library to your concatenated LINKLIST. If you did not do this for Z Data Tools Base function, add this library to either your LNKLST*xx* or PROG*xx* member in SYS1.PARMLIB.

Modifying the TSO logon procedure

If you made Z Data Tools Base function available to TSO, then you do not need to make any further changes to TSO for ZDT/IMS. Otherwise, add the Z Data Tools libraries to your TSO logon procedure, as described Modifying the TSO logon procedure on page 27.

Alternatives for controlling ZDT/IMS auditing

ZDT/IMS auditing is an optional facility. There is no requirement to implement it and ZDT/IMS works if auditing is not implemented. You should consider:

- Whether user access to IMS™ databases using Z Data Tools IMS™ component requires auditing.
- The information that Z Data Tools audit log records can provide.
- The information that Z Data Tools audit log records cannot provide, and possible alternatives to obtaining that information.
- If you do decide to use Z Data Tools auditing, how you will handle any issues associated with large audit log data sets, or additional SMF records.
- How you will use the information provided by Z Data Tools audit log records.

If your site requires a record of a user's read access to IMS™ databases, an external security product such as RACF® can be configured to log access by some or all users, and may be a better alternative.

Z Data Tools audit of read access to IMS™ data does not write audit log records for every segment processed, rather the name of the database and how many segments were processed are written to the audit log.

Z Data Tools audit of changes to IMS™ data typically writes two log records, a before and after image of the segment that was changed. If you intend to log update changes to IMS™ databases that are subject to heavy update activity you need to consider the performance impact of writing many audit log records, also the size of any audit log data sets that may be produced

You have two choices as to how you control auditing of ZDT/IMS activities:

Use HFM1POPT controlled audit logging

This was the original method of controlling auditing and as such only provides limited functionality.

With this method, you control audit logging by specifying the required audit settings in the ZDT/IMS installation options module.

These points summarize the facilities available with HFM1POPT controlled auditing:

- The ZDT/IMS Edit function provides audit logging support, but the other ZDT/IMS functions do not create audit trails.
- You can specify different audit settings (such as whether or not auditing is required) for each IMS™ subsystem that ZDT/IMS accesses.
- The audit settings specified for any IMS™ subsystem apply equally to all ZDT/IMS users accessing that IMS™ subsystem.
- The audit settings specified for any IMS™ subsystem apply equally to all databases within that IMS™ subsystem.
- The **Create audit trail** option on the Edit Entry panel allows users to request audit logging of their Edit sessions when audit logging is not required.
- You can specify audit logging to SMF or to the user's audit log data set, but this is an installation-wide setting and you can only get logging to both the user's log data set and SMF if you specify logging to SMF and you request that the audit log is printed at the end of the Edit session.

Use System Authorization Facility (SAF) controlled audit logging

With this method, audit logging is controlled by RACF® (or an equivalent security product) and FACILITY and XFACILIT class profiles that you define.

These points summarize the facilities available with SAF-rule controlled auditing:

- All ZDT/IMS functions that access IMS™ databases provide audit logging support.
- You can specify different audit settings (such as whether or not auditing is required) for each IMS™ subsystem that ZDT/IMS accesses.
- You can specify different audit settings for different ZDT/IMS users.
- You can specify different audit settings for different databases.
- You can specify different audit settings for each ZDT/IMS function.
- You can control whether or not the **Create audit trail** option on the Edit Entry can be used:
  - To request an audit trail when one is not required.
  - To stop an audit trail being created when one is required.
- You can specify audit logging to SMF, to the user's audit log data set or, for Edit and Browse only, to the user's audit log data set with automatic (mandatory) printing of the audit log at the end of the session. You can also specify dual logging (to the user's audit log data set and to SMF).

Some other points to consider are:

- Audit logging to SMF requires additional set-up, but provides a more reliable and secure environment for capturing audit information than audit logging to the user's audit log data set.
- If an attempt to write an audit log record to SMF or the user's log data set fails, the ZDT/IMS function terminates.
- If you implement SAF-rule controlled auditing you need to decide how Z Data Tools auditing will be enabled. This is described in more detail in Customizing the Z Data Tools audit facility for IMS component on page 297. There are two alternatives. One requires an enabling SAF rule and the presence of a member in SYS1.PARMLIB. The other requires an enabling SAF rule but has no requirement for a member in SYS1.PARMLIB. The use of a member

in SYS1.PARMLIB provides additional facilities compared with the alternative that does not require the use of SYS1.PARMLIB. The additional facilities are documented in Z Data Tools options specified in PARMLIB members on page 515.

When you have determined the appropriate type of auditing for your installation, follow the relevant instructions in Customizing the Z Data Tools audit facility for IMS component on page 297.

Chapter 22. Customizing the operating environment for ZDT/IMS

This chapter describes how to customize the operating environment for ZDT/IMS. You do this after you have installed ZDT/IMS.

Modifying the ISPF environment

To make it easy to start ZDT/IMS under ISPF, configure your ISPF environment as described in the following sections.

Adding ZDT/IMS to your ISPF menu

To add ZDT/IMS to your ISPF Primary Option Menu panel (ISR@PRIM), insert the additional lines (◄ **New**) as shown in Figure 54: on page 261. You could add ZDT/IMS in your Primary Option Menu after the Z Data Tools Base function.

Figure 54.

```
⋮
)BODY  CMD(ZCMD)
⋮
 9  IBM Products  IBM program development products
 10 SCLM          SW Configuration Library Manager
 11 Workplace     ISPF Object/Action Workplace
 Z  Z Data Tools  Z Data Tools
 ZI ZDT/IMS       ZDT/IMS              ◄ New
⋮
)PROC
⋮
 &ZSEL = TRANS( TRUNC (&ZCMD,'.')
⋮
  9,'PANEL(ISRDIIS) ADDPOP'
 10,'PGM(ISRSCLM) SCRNAME(SCLM) NOCHECK'
 11,'PGM(ISRUDA) PARM(ISRWORK) SCRNAME(WORK)'
  Z,'PANEL(HFMSTASK) SCRNAME(ZDTOOLS) NEWAPPL(HFM)' /* Z Data Tools */
 ZI,'PANEL(HFM1ST00) SCRNAME(HFMIMS) NEWAPPL(HFM1)'  /* Z Data Tools IMS */ ◄New
⋮
```

To invoke ZDT/IMS with LIBDEFs, see Example 1. Invoking Z Data Tools, ZDT/IMS, and ZDT/Db2 primary options from a selection panel on page 35.

For information about configuring your ISPF Primary Option Menu panel, see z/OS ISPF Planning and Customizing.

Defining ZDT/IMS in an ISPF command table

ISPF supports four different command tables where you can define an ISPF command to invoke ZDT/IMS:

- Application command table
- User command table
- Site command table
- System command table

You can use the ISPF Command Table Utility (option 3.9) to create or change a command table that is not currently in use (the system command table, ISPCMDS, is always in use). When you add a command for ZDT/IMS to one of these command tables, you can invoke ZDT/IMS from any ISPF panel without prefixing the command with TSO.

Add the following entry to a command table to enable ZDT/IMS to be run from any ISPF panel by entering ZI on the command line.

Verb	ZI
Action	SELECT PANEL(HFM1ST00) OPT(&ZPARM) SCRNAME(HFMIMS) SUSPEND NEWAPPL(HFM1)
Description	Z Data Tools IMS™ component

To invoke ZDT/IMS with LIBDEFs, see #unique_68_Connect_42_libdef-invoke-ex2.

For information about ISPF command tables, see z/OS ISPF Planning and Customizing and *z/OS ISPF User's Guide Vol II*.

Customizing IMS™ to support the use of dynamic PSBs

If you plan to use dynamic PSBs to access databases, you might have to declare the dynamic PSBs and provide a DOPT ACBLIB data set.

Declaring the dynamic PSBs

If you do not plan to use dynamic PSBs to access databases in BMP mode or dynamic resource definition for MODBLKS resources is enabled and you plan to use the IMS Data Definition utility to generate dynamic PSBs, declaring the dynamic PSBs is not required.

Otherwise, you must include in the system definition an APPLCTN macro statement for each dynamic PSB name used by ZDT/IMS functions when run in BMP mode.

You use:

- The DYNPRFN parameter on the HFM1POPI macro statement to specify the number of dynamic PSB names that ZDT/IMS can use to access the subsystem. (See DYNPRFN on page 466).
- The DYNPRFX parameter on the HFM1POPI macro statement to specify the first 1- to 5-characters of the dynamic PSB name. (See DYNPRFX on page 467).

The dynamic PSB names that ZDT/IMS uses are obtained by combining the 1- to 5-characters you specify in the DYNPRFX parameter with a three-digit number in the range 001 to the number specified in the DYNPRFN parameter.

For example, if you specify DYNPRFN=3 and DYNPRFX=HFM on the HFM1POPI macro statement for the subsystem, you will need to include the following APPLCTN macro statements in the system definition:

```
APPLCTN PSB=HFM001,PGMTYPE=BATCH,DOPT
APPLCTN PSB=HFM002,PGMTYPE=BATCH,DOPT
APPLCTN PSB=HFM003,PGMTYPE=BATCH,DOPT
```

PGMTYPE=BATCH is required because the PSBs are for BMPs. The DOPT parameter specifies that it is a dynamic PSB, so it is also required.

Providing a DOPT ACBLIB data set

If you plan to use the IMS Data Definition utility to generate dynamic PSBs, or ACBs are managed by your installation (ACBMGMT=ACBLIB) and you do not plan to use dynamic PSBs to access databases in BMP mode, a DOPT ACBLIB data set is not required.

Otherwise, provide an ACBLIB data set into which the ACB Maintenance utility can generate DOPT PSBs. Use the DYNACB parameter on the HFM1POPI macro statement to specify the name of this data set.

When the ACBs are managed by your installation (ACBMGMT=ACBLIB), concatenate the specified data set with the primary ACBLIB data set in the IMS™ execution JCL (that is, the JCL that starts the IMS™ control region). The DOPT ACBLIB data set must be concatenated to both the IMSACBA and IMSACBB DDs in the IMS™ execution JCL.

When IMS™ management of ACBs is enabled (ACBMGMT=CATALOG), you do not have to concatenate the specified data set with the primary ACBLIB data set. Instead, the specified data set is used as input for the IMS Catalog Populate utility (DFS3PU00), which adds the DOPT PSBs to the IMS catalog.

Chapter 23. Customizing ZDT/IMS

This chapter describes how to customize ZDT/IMS. You do this after you have installed ZDT/IMS.

Customizing the ZDT/IMS installation options module

Before you can use ZDT/IMS, you must customize the installation options module, HFM1POPT. In HFM1POPT you specify:

- The IDs of the IMS™ subsystems that are to be accessed by ZDT/IMS.
- The parameters that are to be passed to the IMS™ region controller when a function starts a BMP, Fast Path or DL/I batch processing region.
- The names of the PSBLIB, DBDLIB, DFSVSAMP, RESLIB, IMS™ macros, staging ACBLIB, RECON, DOPT ACBLIB and Template data sets for each subsystem.
- Selected processing options that you want the ZDT/IMS functions to use.
- Whether or not the user can override the values you specify for selected parameters, data sets, and processing options.

A version of HFM1POPT is distributed with ZDT/IMS, but you will need to replace this module with your own version of HFM1POPT.

You use the usermod HFM1UMDP to install your version of HFM1POPT. HFM1UMDP and the source for HFM1POPT are distributed in HFM.SHFMSAM1. To create and install your own version of HFM1POPT:

1. Copy the member HFM1POPT from HFM.SHFMSAM1 into your own source library. The member HFM1POPT contains the sample source for the HFM1POPT module.
2. Modify your copy of the HFM1POPT sample source, as described in the following sections.
3. Modify the HFM1UMDP member in HFM.SHFMSAM1 to meet your site's requirements. Refer to the usermod for information about changes you might need to make.
4. Install SMP/E usermod HFM1UMDP.

Note: You can also use the sample job HFM1POPH to assemble HFM1POPT if you do not want to use SMP/E.

Migration considerations

If you are migrating from an earlier version of ZDT/IMS, you cannot use the HFM1POPT module that was created for the earlier version. If you do, you may get unpredictable results.

However, you can use the source code from which the earlier version of the HFM1POPT module was created. Add any new parameters that you need to specify, then assemble the source using the latest version of the Z Data Tools macro library (HFM.SHFMMAC1) and lastly link-edit the module.

ZDT/IMS macro statements

The HFM1POPT sample source includes five ZDT/IMS macro statements:

- HFM0POPI
- HFM1POPD
- HFM1POPI
- HFM1AGNT
- HFM1END

You use these macro statements to specify the details that ZDT/IMS requires.

HFM0POPI macro

You use the HFM0POPI macro statement, to specify the non-IMS options. This macro statement is also included in the installation option modules for the other Z Data Tools components. But you do not have to use the same options in ZDT/IMS as you do in the other components.

The HFM0POPI macro statement is required and it must be the first statement in the HFM1POPT module. You can only have one HFM0POPI macro statement in the HFM1POPT module.

The parameters on the HFM0POPI macro statement in the sample HFM1POPT specify the default values. The parameters that ZDT/IMS does not use are omitted. All the parameters are optional.

For detailed information on these parameters, see Z Data Tools options on page 383. You use this information to decide the required settings for these parameters at your installation.

HFM1POPD macro

You use the HFM1POPD macro statement to specify the ZDT/IMS installation defaults. The HFM1POPD macro statement is required and it must be the second statement in the HFM1POPT module. You can only have one HFM1POPD macro statement in the HFM1POPT module.

The parameters on the HFM1POPD macro statement in the sample HFM1POPT specify the default values. All the parameters are optional. Each one is described in this chapter. For more detailed information, see ZDT/IMS options on page 457.

HFM1POPI macro

You use the HFM1POPI macro statement to specify the values that are to be used when ZDT/IMS accesses an IMS™ subsystem. One HFM1POPI macro statement is required for each IMS™ subsystem that is to be accessed by ZDT/IMS. They must be included after the HFM1POPD macro statement and before all HFM1AGNT macro statements.

All the parameters on the HFM1POPI macro statement except the SSID parameter are optional. Each one is described in this chapter. For more detailed information, see ZDT/IMS options on page 457.

HFM1AGNT macro

If a subsystem uses AGNs (Application Group Names) to secure dependent regions, you use HFM1AGNT macro statements to specify the AGNs that you want ZDT/IMS to use when it accesses this subsystem in BMP mode. One HFM1AGNT macro statement is required for each AGN you want to specify.

This macro statement is optional. Specifying AGNS for a subsystem is only required when you want to either:

- Limit the AGNS that ZDT/IMS functions can use when accessing this subsystem in BMP mode, or
- Allow users to be able to select from a list of AGNs when they specify this subsystem (and a BMP region type) on the ZDT/IMS functions entry panel.

If you do want to include HFM1AGNT macro statements in the HFM1POPT module, include them after all the HFM1POPI macro statements and before the HFM1END macro statement.

All the parameters on the HFM1AGNT macro statement except the DESC parameter are required. Each one is described in this chapter. For more detailed information, see ZDT/IMS options on page 457.

HFM1END macro

The HFM1END macro statement is required. HFM1END can only be specified once and it must be the last statement in the HFM1POPT module. This macro statement does not have any parameters.

Specifying the IMS™ subsystems

Include in the HFM1POPT module, one HFM1POPI macro statement for each IMS™ subsystem that is to be accessed by ZDT/IMS and use the **SSID** parameter on each statement to specify the subsystem ID. The **SSID** parameter is required.

Use the **DESC** parameter to specify a description of the subsystem. The description is displayed on the Subsystem Selection panel. The **DESC** parameter is optional.

You can use the HFM1POPD macro statement to specify an installation default for all the other parameters of the HFM1POPI macro statement. These parameters are optional. When a parameter is not specified on the HFM1POPI macro statement, ZDT/IMS uses the installation default (if specified) or the ZDT/IMS default (otherwise). So you only need to specify a parameter on the HFM1POPI macro statement if it differs from the installation or ZDT/IMS default.

Specifying the IMS Group

If you want ZDT/IMS to identify the active IMS™ subsystem based on its associated `IMSGROUP`, specify `IMSGROUP=` in the HFM1POPI entry for that SSID. This support is available only in BMP mode.

If multiple IMS™ subsystems in the same `IMSGROUP` are active, ZDT/IMS connects to the first active IMS™ subsystem in the sequence defined for that `IMSGROUP`. The sequence is determined by the order of the HFM1POPI entries in the ZDT/IMS installation options module.

Note: The `IMSGROUP` value specified in the HFM1POPI entry must match the value defined for the corresponding IMSID in the DFSPBxxx member.

Specifying the parameters passed to the IMS™ region controller

Use the parameters in column 2 of Table 41: DLIBATCH parameters on page 267 to specify values for the DLIBATCH parameters in column 1. The values you specify are passed to the IMS™ region controller when a Z Data Tools/IMS function is run in DLI mode. Where applicable, use the parameters in column 3 to specify whether or not the user can override the values you specify.

Table 41. DLIBATCH parameters

DLIBATCH parameter	HFM1POPD/HFM1POPI parameter	Use this parameter to allow override
BKO	IMSBKO	UIMSBKO
BUF		UBUF
DBRC	DBRC[61]	UDBRC
DFSDF	DFSDF[62]	Cannot override
GSGNAME	GSGNAME	URSR
IRLM	IRLM	UIRLM
IRLMNM	IRLMNAME	UIRLM
LOCKMAX	LOCKMAX	ULOCKMAX
TMINAME	TMINAME	URSR

Use the parameters in column 2 of Table 42: IMSBATCH parameters on page 267 to specify values for the IMSBATCH parameters in column 1. The values you specify are passed to the IMS™ region controller when a Z Data Tools/IMS function is run in BMP mode. Use the parameters in column 3 to specify whether or not the user can override the values you specify.

Table 42. IMSBATCH parameters

IMSBATCH parameter	HFM1POPD/HFM1POPI parameter	Use this parameter to allow override
LOCKMAX	LOCKMAX	ULOCKMAX
PARDLI	PARDLI	UPARDLI
NBA	IMSNBA	UIMSNBA
OBA	IMSOBA	UIMSNBA

61. Not used when IMS management of ACBs is enabled (ACBMGMT=CATALOG).
62. Not used when the ACBs are managed by your installation (ACBMGMT=ACBLIB).

All the parameters in columns 2 and 3 of Table 41: DLIBATCH parameters on page 267 and Table 42: IMSBATCH parameters on page 267 can be specified on the HFM1POPD and HFM1POPI macro statements. For detailed information on how to code these parameters, see ZDT/IMS options on page 457.

PARDLI considerations

The PARDLI parameter specifies the parallel DL/I option that ZDT/IMS functions use when they are run in BMP mode.

When this parameter is not specified on the HFM1POPD and HFM1POPI macro statements, ZDT/IMS passes a null value to the IMS™ region controller and IMS™ uses the default value, PARDLI=0.

With PARDLI=0, a system X22 abend in the BMP region causes the IMS™ control region to terminate with a U0113 abend.

A system X22 abend will occur in a Z Data Tools/IMS BMP region when:

- A TSO session executing an Edit or Browse in BMP mode is cancelled or times out.
- A batch function running in BMP mode is cancelled or times out.

If you want to prevent the control region being terminated with abend U0113 in the above-mentioned instances, you should set PARDLI=1. But note that this setting can degrade performance.

Controlling access to the subsystems

Use the parameters in Table 43: Control options on page 268 to control the access that ZDT/IMS functions have to each subsystem.

Table 43. Control options

Parameter	Description
CHKPINTVL	Issue checkpoints while Edit/Browse BMPs are waiting for a user response so that utilities that support pausing BMPs are able to pause inactive Edit/Browse BMPs.
DYNALLOC	Enforce the use of the database data sets specified in the DFSMDA dynamic allocation members when accessing the subsystem in DLI mode.
IMSAUDLG	Enforce audit logging when editing databases in the subsystem.
MAXGN	Limit the number of DL/I calls that can be issued when searching a database in the subsystem.
PSBTYPES	Prevent dynamic PSBs or static PSBs being used to access databases in the subsystem.
READONLY	Prevent users from running update functions (Edit, Batch Edit, Load, Initialize and Delete/Define) against databases in the subsystem.
REGTYPES	Prevent users from accessing the subsystem in BMP mode or DLI mode.
TIMEOUTI	Time out Edit/Browse BMPs after a period of inactivity.

Table 43. Control options (continued)

Parameter	Description
VCURULE	Control the views and criteria sets that can be used with a database in the subsystem.

Use the parameters in Table 44: IMS log usage parameters on page 269 to control the IMS™ log usage of functions which run in DLI mode and use PSBs that have update intent.

Table 44. IMS™ log usage parameters

Parameter	Description
LOGUSAGE	Whether the functions use an IMS™ log data set and, if they do use a log, whether the log is kept when the function ends.
ULOGUSAG	Whether the user can override the value that is specified for the LOGUSAGE parameter.
LOGDSN	The name pattern that the functions use to generate IMS™ log data set names.
ULOGDSN	Whether the user can override the name pattern that is specified in the LOGDSN parameter.
UIEFRDER	Whether batch functions use the IEFRDER DD in the JCL.

If you want to force ZDT/IMS functions to use an IMS™ log when they are run in DLI mode and use a PSB that has update intent, then specify ULOGUSAG=N, ULOGDSN=N, UIEFRDER=N and one of the following options:

LOGUSAGE=KEEP

If you want to force the user to keep the log when the function ends.

LOGUSAGE=KEEPUPD

If you want to force the user to keep the log if the function updates databases or does not end normally.

LOGUSAGE=DELETE

If you want to force the user to keep the log if the function does not end normally.

All the parameters in Table 43: Control options on page 268 and Table 44: IMS log usage parameters on page 269 can be specified on the HFM1POPD and HFM1POPI macro statements. For detailed information on how to code these parameters, see ZDT/IMS options on page 457.

Specifying selected processing options

Use the parameters in column 1 of Table 45: Processing options on page 269 to specify the processing options that you require. Use the parameters in column 3 to specify whether or not the user can override the values you specify.

Table 45. Processing options

Parameter	Description	Use this parameter to allow override
AUTOSAVE	Whether or not the automatic save function is set on during Edit.	UAUTOSAV

Table 45. Processing options (continued)

Parameter	Description	Use this parameter to allow override
CHGAFREQ	The AUTOSAVE checkpoint frequency during Change All and Repeat All operations.	UAUTOSAV
EDITFREQ	The AUTOSAVE checkpoint frequency during an Edit.	UAUTOSAV
IEBFREQ	The checkpoint frequency during a Batch Edit.	UIEBFREQ
LOADFREQ	The checkpoint frequency during a database load.	ULOADFRQ
PROCOPTB	The PSB processing option (PROCOPT) that a dynamic PSB generated for a Browse uses to read databases.	UPROCOPB
PROCOPTP	The PSB processing option (PROCOPT) that a dynamic PSB generated for a Print uses to read databases.	UPROCOPP
PROCOPTX	The PSB processing option (PROCOPT) that a dynamic PSB generated for an Extract uses to read databases.	UPROCOPX
PROCOPTY	The PSB processing option (PROCOPT) that a dynamic PSB generated for a Batch Browse uses to read databases.	UPROCOPY
PSBTYPE	The type of PSB that batch functions use to access databases.	UPSBTYPE

All the parameters in columns 1 and 3 of Table 45: Processing options on page 269 except PSBTYPE and UPSBTYPE can be specified on the HFM1POPD and HFM1POPI macro statements. You can only specify PSBTYPE and UPSBTYPE on the HFM1POPD macro statement.

For detailed information on how to code these parameters, see ZDT/IMS options on page 457.

Specifying IMS™ and ZDT/IMS data sets

Use the parameters in column 2 of Table 46: IMS and ZDT/IMS data sets on page 270 to specify the data sets in column 1. Where applicable, use the parameters in column 3 to specify whether or not the user can override the values you specify.

Table 46. IMS™ and ZDT/IMS data sets

Data set	Parameter	Use this parameter to allow override
DBDLIB	DBDLIB*n*[63]	UDBDLIB
DFSVSAMP data set	DFSVSAMP	UDFSVSMP
DFSVSAMP member	VSMPMEM	UDFSVSMP
DOPT ACBLIB	DYNACB	Cannot override

63. Not used when IMS management of ACBs is enabled (ACBMGMT=CATALOG).

Table 46. IMS™ and ZDT/IMS data sets (continued)

Data set	Parameter	Use this parameter to allow override
Dynamic PSBLIB	DYNPSB	Cannot override
IMS™ bootstrap	BSDSHLQ[64]	Cannot override
IMS™ macros	MACLIB	UMACLIB
IMS™ PROCLIB	PROCLIB[64]	Cannot override
RESLIB	RESLIB*n*	URESLIB
RECON		URECON
Staging ACBLIB	ACBLIB	UACBLIB
Static PSBLIB	PSBLIBn[63]	UPSBLIB
Template	TPLLIB*n*	UTPLLIB
SDSNLOAD	SDSNLOAD*n*	Cannot override

All the parameters in columns 2 and 3 of Table 46: IMS and ZDT/IMS data sets on page 270 can be specified on the HFM1POPD and HFM1POPI macro statements. For detailed information on how to code these parameters, see ZDT/IMS options on page 457.

Specifying miscellaneous IMS™ subsystem details

Use the parameters in Table 47: Miscellaneous IMS subsystem details on page 271 to specify some miscellaneous IMS™ subsystem details that ZDT/IMS requires.

Table 47. Miscellaneous IMS™ subsystem details

Parameter	Description
ACBMGMT	Whether IMS™ manages the active application control blocks (ACBs) in the IMS catalog or your installation manages the ACBs in ACB libraries (ACBLIBs).
ACBSHR	Whether the ACBs that this IMS subsystem uses are also used by other IMS™ subsystems.
CATALIAS	The alias for the IMS™ catalog.
DYNPRFN	The maximum number of dynamic PSBs required by concurrent ZDT/IMS users (when the PSBs are generated by the IMS Data Definition utility) or by concurrent ZDT/IMS BMP users (otherwise).
DYNPRFX	The first 1 - 5 characters of the dynamic PSB names used by ZDT/IMS functions.
DFSRRC00	The name of the program that ZDT/IMS attaches to invoke IMS™.
IMSGROUP	The 1- to 4-character identifier of the IMSGROUP in which the IMS subsystem is a member.

64. Not used when the ACBs are managed by your installation (ACBMGMT=ACBLIB).

Table 47. Miscellaneous IMS™ subsystem details (continued)

Parameter	Description
IMSPLEX	When the IMS subsystem is a member of an IMSplex, the 1 to 5-character IMSplex identifier.
PADS	Whether or not Program Access to Data Sets (PADS) is used in the environments in which ZDT/IMS is run.
REGCATLG	Whether the IMS™ catalog is registered with DBRC.
UAGNS	Whether or not the subsystem uses AGNs to secure dependent regions.
USEDDL	Whether or not ZDT/IMS uses the IMS Data Definition utility (DFS3ID00) to generate dynamic PSBs when IMS management of ACBs is enabled (ACBMGMT=CATALOG).
XDOPTLB	When APAR PH17975 is not applied: Whether the ACB Maintenance utility deletes all PSBs and DBDs from the DOPT ACBLIB data set and makes all their space available for reuse, before building control blocks for the dynamic PSB. When APAR PH17975 is applied: This parameter is ignored. The ACB Maintenance utility deletes all PSBs and DBDs from the DOPT ACBLIB data set and makes all their space available for reuse, before building control blocks for the dynamic PSB.

All the parameters in Table 47: Miscellaneous IMS subsystem details on page 271 can be specified on the HFM1POPD and HFM1POPI macro statements except for the PADS parameter, which can only be specified on the HFM1POPD macro statement. For detailed information on how to code these parameters, see ZDT/IMS options on page 457.

Specifying AGNS

If a subsystem uses AGNs to secure dependent regions, you can use HFM1AGNT macro statements to specify the AGNs that you want ZDT/IMS to use when it accesses this subsystem in BMP mode. One HFM1AGNT macro statement is required for each AGN you want to specify. You use:

- The **AGN** parameter to specify the Application Group Name.
- The **SSID** parameter to specify the ID of the subsystem that the AGN is for.
- The **DESC** parameter to specify a description of the AGN.

The **AGN** and **SSID** parameters are required. The **DESC** parameter is optional. For detailed information on how to code these parameters, see ZDT/IMS options on page 457.

Note: Specifying AGNs for a subsystem is optional unless you want to limit the AGNs that ZDT/IMS functions can use when accessing this subsystem in BMP mode, or you want users to be able to select from a list of AGNs when they specify this subsystem (and a BMP region type) on the ZDT/IMS function's entry panel.

You must, however, specify whether or not each subsystem uses AGNs to secure dependent regions. If a subsystem uses AGNs, specify UAGNS=Y on the HFM1POPI macro statement, or specify UAGNS=Y on the HFM1POPD macro statement and do not specify the UAGNS parameter on the HFM1POPI macro statement.

If a subsystem does not use AGNs, specify UAGNS=N on the HFM1POPI macro statement, or specify UAGNS=N on the HFM1POPD macro statement and do not specify the UAGNS parameter on the HFM1POPI macro statement, or do not specify the UAGNS parameter on the HFM1POPI and HFM1POPD macro statements.

Note: If the UAGNS parameter is not specified on the HFM1POPD and HFM1POPI macro statement, ZDT/IMS uses the default value, UAGNS=N, and any HFM1AGNT macro statements for the subsystem included in the HFM1POPT module are ignored.

Setting the default national language

If you installed the Japanese component of ZDT/IMS, (or you provided other locally translated messages and panels), you can change the default national language, for ZDT/IMS batch functions. The language used by ZDT/IMS under ISPF depends on the language setting for your ISPF session. To change the language for ZDT/IMS batch functions, set the LANGUAGE parameter on the HFM0POPI macro statement, to your language. See LANGUAGE on page 403 for more information about the LANGUAGE parameter.

For other customization you can do for ZDT/IMS for national languages, refer to Customizing ZDT/IMS for national languages on page 310.

Examples of HFM1POPD, HFM1POPI and HFM1AGNT macros

The following sample code shows the HFM1POPD, HFM1POPI and HFM1AGNT macros you would code to define IMS™ subsystems to ZDT/IMS. Five subsystems are defined, of which two use AGNs. For an explanation of the statements see the notes at the end of the example. See ZDT/IMS options on page 457 for more information about HFM1POPD, HFM1POPI and HFM1AGNT.

Where a parameter value already coded in HFM1POPD is to be used by a subsystem, that parameter is not coded in the HFM1POPI macro for that subsystem.

HFM1POPD

All BMP values are protected, all DLI values are unprotected. These will be the defaults unless overridden in an HFM1POPI macro for a subsystem.

HFM1POPI

SSID=IF52, un-protect all BMP values such that everything is unprotected, and use Dynamic PSBs only.

SSID=IF42, protect all DLI values such that everything is protected, and use Static PSBs only.

SSID=IF32, only use database data sets that are dynamically allocated, and use AGNs.

SSID=IFA2, BMP only, use all values from HFM1POPD except data set names.

SSID=IFB2, READONLY, all other values from HFM1POPD.

```
HFM1POPD ACBLIB=IMSV11.IFB2.ACBLIB,                          +
               AUTOSAVE=(Y,Y),                               +
               CHGAFREQ=(100,100),                           +
               DBRC=Y,                                       +
               DBDLIB1=HFM1.PROD.DBDLIB,                     +
               DBDLIB2=HFMA.PROD.DBDLIB2,                    +
               DBDLIB3=HFM2.PROD.DBDLIB,                     +
               DBDLIB4=HFM5.PROD.DBDLIB,                     +
               DBDLIB5=HFMX.TEST.DBDLIB,                     +
               DBDLIB6=HFMY.TEST.DBDLIB,                     +
               DFSVSAMP=IMSV11.IFB2.PROCLIB,                 +
               DYNACB=IMSV11.IFB2.DOPTLIB,                   +
               DYNALLOC=N,                                   +      ◄1
               DYNPRFN=50,                                   +
               DYNPRFX=HFM0,                                 +
               EDITFREQ=(1,1),                               +
               IEBFREQ=(100,100),                            +
               IMSAUDLG=N,                                   +
               IMSBKO=Y,                                     +
               IMSNBA=10,                                    +
               IMSOBA=6,                                     +
               IRLM=N,                                       +
               IRLMNAME=,                                    +
               LOADFREQ=(100,100),                           +
               MACLIB=IMS.V10.SDFSMAC,                       +
               MAXGN=(100,100),                              +
               PARDLI=0,                                     +
               PSBLIB1=HFM1.PROD.PSBLIB,                     +
               PSBLIB2=HFMA.PROD.PSBLIB2,                    +
               PSBLIB3=HFM2.PROD.PSBLIB,                     +
               PSBLIB4=HFM5.PROD.PSBLIB,                     +
               PSBLIB5=HFMX.TEST.PSBLIB,                     +
               PSBLIB6=HFMY.TEST.PSBLIB,                     +
               PROCOPTB=(G,G),                               +
               PROCOPTP=(G,G),                               +
               PROCOPTX=(G,G),                               +
               PROCOPTY=(G,G),                               +
               PSBTYPE=DYNAMIC,                              +
               PSBTYPES=BOTH,                                +      ◄2
               READONLY=N,                                   +
               REGTYPES=BOTH,                                +      ◄3
               RESLIB1=IMSV11.IFB2.SDFSRESL,                 +
               RESLIB2=IMSV11.IFB2.USERLIB,                  +
               TPLLIB1=HFM1.PROD.TEMPLATE,                   +
               TPLLIB2=HFMA.PROD.TEMPLATE,                   +
               TPLLIB3=HFM2.PROD.TEMPLATE,                   +
               TPLLIB4=HFM5.PROD.TEMPLATE,                   +
               TPLLIB5=HFMX.TEST.TEMPLATE,                   +
               TPLLIB6=HFMY.TEST.TEMPLATE,                   +
```

```
               UACBLIB=Y,                                      +
               UAUTOSAV=(N,Y),                                 +
               UBUF=Y,                                         +
               UDBRC=Y,                                        +
               UDFSVSMP=Y,                                     +
               UIEBFREQ=(N,Y),                                 +
               UIMSBKO=Y,                                      +
               UIMSNBA=N,                                      +
               UIRLM=Y,                                        +
               ULOADFRQ=(N,Y),                                 +
               ULOCKMAX=(N,Y),                                 +
               UMACLIB=Y,                                      +
               UPARDLI=N,                                      +
               UPROCOPB=(N,Y),                                 +
               UPROCOPP=(N,Y),                                 +
               UPROCOPX=(N,Y),                                 +
               UPROCOPY=(N,Y),                                 +
               UPSBTYPE=Y,                                     +
               URECON=Y,                                       +
               URESLIB=Y,                                      +
               URSR=Y,                                         +
               VCURULE=T,                                      +
               VSMPMEM=DFSVSMDB,                               +
               DOPTLB=N
*
HFM1POPI SSID=IF52,                                            +         ◄ IF52
               DESC="Dynamic PSB only, unprotected",           +
               ACBLIB=IMSV910.IF52.ACBLIB,                     +      ◄ 7
               DBDLIB1=HFM.IF52.DBDLIB,                        +      ◄ 29
               DYNACB=IMSV910.IF52.DOPTLIB,                    +      ◄ 8
               IRLM=Y,                                         +      ◄ 6
               IRLMNAME=IRLM,                                  +
               MACLIB=IMS.V910.SDFSMAC,                        +      ◄ 9
               MAXGN=(0,0),                                    +
               PSBTYPES=DYNAMIC,                               +      ◄ 4
               RESLIB1=IMSV910.IF52.SDFSRESL,                  +      ◄ 10
               UAUTOSAV=Y,                                     +
               UIEBFREQ=Y,                                     +
               UIMSNBA=Y,                                      +
               ULOADFRQ=Y,                                     +
               ULOCKMAX=Y,                                     +      ◄ 5
               UPARDLI=Y,                                      +
               UPROCOPB=Y,                                     +
               UPROCOPP=Y,                                     +
               UPROCOPX=Y,                                     +
               UPROCOPY=Y,                                     +
               XDOPTLB=                                        +
*
         HFM1POPI SSID=IF42,                                   +         ◄ IF42
               DESC="Static PSB only, all protected",          +
               ACBLIB=IMSV910.IF42.ACBLIB,                     +      ◄ 15
               DBRC=N,                                         +      ◄ 14
               DYNACB=IMSV910.IF42.DOPTLIB,                    +      ◄ 16
               PSBLIB1=HFM.IF42A.PSBLIB,                       +      ◄ 333
               PSBLIB2=HFM.IF42B.PSBLIB,                       +
               PSBTYPES=STATIC,                                +      ◄ 11
               RESLIB1=IMSV910.IF42.SDFSRESL,                  +      ◄ 17
               RESLIB2=IMSV910.IF42.USERLIB,                   +      ◄ 18
```

```
               UACBLIB=N,                                  +      ◄ 12
               UAUTOSAV=(,N),                              +      ◄ 13
               UBUF=N,                                     +
               UDBRC=N,                                    +
               UDFSVSMP=N,                                 +
               UIEBFREQ=(,N),                              +
               UIMSBKO=N,                                  +
               UIRLM=N,                                    +
               ULOADFRQ=(,N),                              +
               ULOCKMAX=(,N),                              +
               UMACLIB=N,                                  +
               UPROCOPB=(,N),                              +
               UPROCOPP=(,N),                              +
               UPROCOPX=(,N),                              +
               UPROCOPY=(,N),                              +
               URECON=N,                                   +
               URESLIB=N,                                  +
               URSR=N                                      +
*
         HFM1POPI SSID=IF32,                               +         ◄ IF32
               DESC="Dynamic allocation ds only",          +
               ACBLIB=IMSV810.IF32.ACBLIB,                 +
               AUTOSAVE=N,                                 +      ◄ 222
               CHGAFREQ=(44,44),                           +
               DBRC=Y,                                     +      ◄ 21
               DFSVSAMP=IMSV810.IF32.PROCLIB,              +
               DYNACB=IMSV810.IF32.DOPTLIB,                +
               DYNALLOC=Y,                                 +      ◄ 19
               RESLIB1=IMSV810.IF32.SDFSRESL,              +
               RESLIB2=,                                   +
               TPLLIB1=IFA2.XXX1.TEMPLATE,                 +      ◄ 31
               TPLLIB2=IFA2.XXX2.TEMPLATE,                 +
               TPLLIB3=IFA2.XXX3.TEMPLATE,                 +

               UACBLIB=N,                                  +
               UAGNS=Y,                                    +      ◄ 22
               XDOPTLB=Y                                   +      ◄ 23
*
         HFM1POPI SSID=IFA2,                               +         ◄ IFA2
               DESC="BMP only system",                     +
               ACBLIB=IMSV11.IFA2.ACBLIB,                  +
               DYNACB=IMSV11.IFA2.DOPTLIB,                 +
               IMSAUDLG=Y,                                 +      ◄ 26
               MACLIB=IMS.V10.BETA.SDFSMAC,                +
               REGTYPES=BMP,                               +      ◄ 24
               RESLIB1=IMSV11.IFA2.SDFSRESL,               +
               RESLIB2=IMSV11.IFA2.USERLIB,                +
               UAGNS=Y                                     +      ◄ 25
*
         HFM1POPI SSID=IFB2,                               +         ◄ IFB2
               DESC="READONLY system",                     +
               READONLY=Y                                  +      ◄ 27
*
         HFM1AGNT SSID=IF52,AGN=HFM52                      +      ◄ 28
         HFM1AGNT SSID=IFA2,AGN=HFMA0001,DESC="HR"         +
         HFM1AGNT SSID=IFA2,AGN=HFMA0002,DESC="Finance"    +
*
         HFM1END                                                  END
```

Note:

1. In this example, the + characters are in column 72.
2. This example shows the statements you would code to define five IMS™ subsystems to be used by ZDT/IMS. Code for an HFM1POPD macro is also shown. Application group name security is in use in two of these IMS™ subsystems.
3. Notice that all the HFM1POPI macros are specified together, followed by the HFM1AGNT macros.
4. An HFM1END macro statement is coded at **END**.

HFM1POPD:

1. These parameters will serve as defaults for the parameters in the HFM1POPI macros, if not specified therein. For parameters that have two values, the first is for BMP mode, the second for DLI mode.
2. As a default, all subsystems can run in BMP and DLI mode [3], all subsystems can run with a Static or Dynamic PSB [2] and dynamic allocation [1] is not enforced.

HFM1POPI:

IF52:

1. These statements define an IMS™ subsystem, called **Dynamic PSB only, unprotected**, as specified on the DESC keyword. The SSID is **IF52**. This is the value that will be shown on the ZDT/IMS entry panels.
2. To force the use of dynamic PSBs, the parameter PSBTYPES is set to "DYNAMIC" [**4**].
3. To unprotect all parameters that have a default value of "protect" all of the usage parameters must be set to "Y" [**5**].
4. Furthermore this subsystem will use IRLM [**6**].
5. All data set names for this subsystem are different from those that are set as default and as such need to be specified [**7**, **8**, **9**, **10**].
6. When this one parameter is included [**29**], it overrides all six DBDLIB parameters set up in HFM1POPD.
7. The UAGNS keyword is not specified, and UAGNS=NO is specified in the HFM1POPD macro; therefore **IF52** will not use application group name security. The AGNs in the first HFM1AGNT macro will be ignored.

IF42:

1. These statements define an IMS™ subsystem, called **Static PSB only, all protected**, as specified on the DESC keyword. The SSID is **IF42**. This is the value that will be shown on the ZDT/IMS entry panels.
2. To force the use of static PSBs, the parameter PSBTYPES is set to "STATIC" [**11**].

3. To protect all parameters that have a default value of "unprotect" all of the usage parameters must be set to "N" [**12**] or "(,N)" [**13**], depending on whether or not that specific parameter has one or two values.
4. Furthermore, this subsystem will not use DBRC [**14**]. This is only possible if the IMS™ system definitions allow for it.
5. Several data set names for this subsystem are different from those that are set as default and as such need to be specified [**15**, **16, 17**, **18**].
6. When these parameters [**15**, **16,**] are included, they override all six PSBLIB parameters set up in HFM1POPD.

IF32:

1. These statements define an IMS™ subsystem, called **Dynamic allocation ds only**, as specified on the DESC keyword. The SSID is **IF32**. This is the value that will be shown on the ZDT/IMS entry panels.
2. To force the use of dynamic allocation of all database data sets in DLI mode, the parameter DYNALLOC is set to "Y" [**19**].
3. When these parameters [**31**] are included, they override all six TPLLIB parameters set up in HFM1POPD.
4. The AUTOSAVE option is turned off [**222**].
5. DBRC is set to "Yes" [**21**], but is not needed as the default value is "Yes" also.
6. This subsystem will use AGNs [**22**].
7. When dynamic PSBs are used, the DOPT ACBLIB is cleared of all members. This is done as XDOPTLB is set to "Y" [**23**].

IFA2:

1. These statements define an IMS™ subsystem, called **BMP only system**, as specified on the DESC keyword. The SSID is **IFA2**. This is the value that will be shown on the ZDT/IMS entry panels.
2. To force the use of BMP mode, the parameter REGTYPES is set to "BMP" [**24**].
3. This subsystem will use AGNs [**25**].
4. To force the use of audit logging, the parameter IMSAUDLG is set to "Y" [**26**].

IFB2:

1. These statements define an IMS™ subsystem, called **READONLY system**, as specified on the DESC keyword. The SSID is **IFB2**. This is the value that will be shown on the ZDT/IMS entry panels.
2. To force this system to be a read-only system, the parameter READONLY is set to "Y" [**27**].

HFM1AGNT:

1. The first statement refers to subsystem **IF52**, however, this subsystem will not use AGNs and as such this statement will be ignored [28].
2. The next two statements refer to subsystem **IFA2** where UAGNS is set to "Y" and it will use these two and only these two AGNs.
3. Note that there is no HFM1AGNT macro for IF32. It is specified to use AGNs. Since there is no HFM1AGNT macro for IF32 the validity of the names will be controlled by IMS™ only and ZDT/IMS will allow any name in the **AGN** field.
4. Note that the advantage of specifying the HFM1AGNT macros is be able to display a selection list if the user is not sure about the name to use. This is not possible when these macros are not included.

Tailoring the job control skeletons

Several dialogs in ZDT/IMS generate JCL for batch functions. If you want these dialogs to generate JCL that will run at your installation, appropriate job control must be provided. This is done by means of the installation options, the Settings panels, and a job control skeleton.

HFM1FTEX

A sample job control skeleton for ZDT/IMS is in HFM.SHFMSLIB(HFM1FTEX). You might need to change the following DD statements in the sample skeleton:

1. The STEPLIB DD statement specifies DSN=&HFMSMOD1. ZDT/IMS sets &HFMSMOD1 to either the value specified for the LOADLIB parameter in the HFM1POPT module (when specified) or HFM.SHFMMOD1 (otherwise). So if you haven't installed Z Data Tools into the default target libraries, you need to specify the name of the target Z Data Tools load library in the LOADLIB parameter or modify this DD statement.
2. The HFM1JIN DD statement specifies DSN=&HFM1SLIB. ZDT/IMS sets &HFM1SLIB to either the value specified for the SKELLIB parameter in the HFM1POPT module (when specified) or HFM.SHFMSLIB (otherwise). So if you haven't installed Z Data Tools into the default target libraries, you need to specify the name of the target Z Data Tools skeleton library in the SKELLIB parameter or modify this DD statement.
3. The STEPLIB DD statement in this skeleton also concatenates two other statements

```
//*        DD DSN=IGY.SIGYCOMP,DISP=SHR
//*HFMCOB  DD DUMMY     Uncomment to force use of ZDT COBOL Compiler
```

which appear as comments in HFM1FTEX.

In the first statement shown above, IGY.SIGYCOMP is the supported, licensed COBOL compiler library. Some batch functions which make use of COBOL require this library. If you did not add your supported COBOL compiler library to your LINKLIST, remove the * to uncomment the line, and change the DSN to the name of your COBOL compiler library. All currently supported versions of IBM® Enterprise COBOL for z/OS® and OS/390® are supported by Z Data Tools.

If you have created a COBOL compiler library for Z Data Tools with a special version of IGYCDOPT, you can use this DD statement to make it available to ZDT/IMS for all batch jobs using COBOL templates. See Using COBOL compiler options with Z Data Tools on page 46 for information about a special version of IGYCDOPT for Z Data Tools.

Z Data Tools provides an internal version of the COBOL compiler, for use in preference to a supported COBOL compiler, or when a supported COBOL compiler is not available to Z Data Tools. (See Using the Z Data Tools COBOL compiler on page 44.)

If you need to modify the HFM1FTEX job control skeleton, do this:

1. Copy the member HFM1FTEX from HFM.SHFMSLIB to your own source library.
2. Modify the HFM1FTEX member in your own library as required.
3. Modify the HFM1UMDB member in HFM.SHFMSAM1 to meet your site's requirements. Refer to the usermod for information about changes you might need to make.
4. Install SMP/E usermod HFM1UMDB.

Note: ZDT/IMS does not provide support for the automatic generation of job routing control statements in the JCL generated by HFM1FTEX.

Customizing for DEDB randomizing modules

If your installation uses DEDB randomizing modules that abend or cause an IMS™ abend, you can create your own HFM1RNDM load module.

In the HFM1RNDM load module, you provide the names, or name patterns, of the DEDB randomizing modules at your installation that abend, or cause an IMS™ abend, when they receive key field values that are not valid packed decimal numbers.

If the key field of the root segment of a DEDB database is defined as TYPE=P in the DBD, and the name of the randomizing module that the DEDB database uses is included in the HFM1RNDM load module, then ZDT/IMS checks that the root key values specified in DL/I calls are valid packed decimal numbers before issuing the calls.

Providing your own HFM1RNDM module

ZDT/IMS does not provide a default HFM1RNDM module. You can create your own using the sample HFM1RNDM and usermod HFM1UMD1.

To create your own HFM1RNDM module:

1. Copy the member HFM1RNDM from HFM.SHFMSAM1 to your own source library.
2. Code the names of the DEDB randomizing routines that you want to include, on the HFM1RAND RANDNAME statements, in your copy of HFM1RNDM. You can provide as many statements as you need. You can specify wildcards, using * (multiple characters) and % (single character).

3. Modify the HFM1UMD1 member in HFM.SHFMSAM1 to meet your site's requirements. Refer to the usermod for information about the changes you need to make.
4. Install SMP/E usermod HFM1UMD1.

Chapter 24. Customizing the ZDT/IMS security environment

This chapter describes:

- How you can control access to databases when functions are run in BMP mode or DLI mode.
- How you can control access to the ZDT/IMS functions and IMS™ subsystems.
- How you can use user-written security exit routines to control access to various IMS™ resources.

The Database Access Control facility

Use the Database Access Control facility to control users' access to databases when running ZDT/IMS functions. You have the option of controlling users' access to databases when functions run in BMP or DLI mode.

Depending on which profiles you define, access checking may be performed when the function uses a dynamic PSB, when the function uses a static PSB, or when the function uses a dynamic or static PSB.

To activate access checking for functions using a dynamic PSB, do the following:

1. Define a FACILITY class profile named `FILEM.IMS1.DBDYNAM`
2. Give all users whose access you want checked UPDATE or READ access to this resource:
    - UPDATE access, if you want access checking to be performed when functions run in BMP or DLI mode.
    - READ access, if you want access checking to be performed only when functions run in BMP mode.

To activate access checking for functions using a static PSB, do the following:

1. Define a FACILITY class profile named `FILEM.IMS1.DBSTATIC`
2. Give all users whose access you want checked UPDATE or READ access to this resource:
    - UPDATE access, if you want access checking to be performed when functions run in BMP or DLI mode.
    - READ access, if you want access checking to be performed only when functions run in BMP mode.

Once access checking is activated, ZDT/IMS issues RACROUTE calls to determine whether the user is authorized to access the database specified on the entry panel or in the batch JCL. The RACROUTE call is for the XFACILIT class resource `FILEM.IMS1.`*`ssid`*`.DB.`*`dbname`*, where *ssid* is the IMS™ subsystem name and *dbname* is the name of the database that the user is attempting to access.

- For access by a read-only function, a user requires a minimum of READ access.
- For access by an update function, a user requires a minimum of UPDATE access.

Create corresponding profiles to protect this resource.

If you plan to use the Database Access Control facility to control users' access to databases when running functions in DLI mode, there is one additional requirement. You must define the IMS subsystems to be dynamic allocation only. You do so by setting the **DYNALLOC** parameter to `Y` when you customize the ZDT/IMS options module.

For more information on the **DYNALLOC** parameter, see ZDT/IMS options on page 457.

Logging unauthorized access attempts

The Database Access Control facility supports writing RACF audit records to SMF.

To activate logging for an IMS subsystem, do the following:

1. Define a FACILITY class profile named `FILEM.IMS1.ssid.DBLOG` where, *ssid* is the IMS subsystem name.
2. Give all users whose access you want logged READ access to this profile.

To activate logging for all IMS subsystems or IMS subsystems whose name match a pattern, replace the *ssid* in the profile name in step 1 on page 283 with an asterisk (*) or the name pattern.

RACF® examples

1. Activating access checking for functions that use a dynamic PSB and run in BMP mode.

   With these commands, checking is activated for all users:

   ```
   RDEFINE FACILITY FILEM.IMS1.DBDYNAM AUDIT(NONE) +
           UACC(READ) OWNER(ownerid)
   SETROPTS RACLIST(FACILITY) REFRESH
   ```

2. Activating access checking for functions that use a static PSB and run in BMP or DLI mode.

   With these commands, checking is activated for all users:

   ```
   RDEFINE FACILITY FILEM.IMS1.DBSTATIC AUDIT(NONE) +
           UACC(UPDATE) OWNER(ownerid)
   SETROPTS RACLIST(FACILITY) REFRESH
   ```

3. Ensure users do not obtain access to databases through profiles that are less specific than the profile FILEM.IMS1.*.DB.*

   ```
   RDEFINE XFACILIT FILEM.IMS1.*.DB.* AUDIT(NONE) +
           UACC(NONE) OWNER(ownerid)
   SETROPTS RACLIST(XFACILIT) REFRESH
   ```

4. Allow user *dbadmin* to update all databases in subsystem IF52, and allow all other users to read these databases.

   ```
   RDEFINE XFACILIT FILEM.IMS1.IF52.DB.* AUDIT(NONE) +
           UACC(READ) OWNER(ownerid)
   PERMIT FILEM.IMS1.IF52.DB.* CLASS(XFACILIT) +
          ID(dbadmin) ACC(UPDATE)
   SETROPTS RACLIST(XFACILIT) REFRESH
   ```

5. Allow user *user1* to update database DJ1E in subsystem IF52, and allow all other users to read this database.

   ```
   RDEFINE XFACILIT FILEM.IMS1.IF52.DB.DJ1E AUDIT(NONE) +
           UACC(READ) OWNER(ownerid)
   PERMIT FILEM.IMS1.IF52.DB.DJ1E CLASS(XFACILIT) +
          ID(user1) ACCESS(UPDATE)
   SETROPTS RACLIST(XFACILIT) REFRESH
   ```

6. Activating logging for subsystem IF52.

   With these commands, logging is activated for all users:

```
RDEFINE FACILITY FILEM.IMS1.IF52.DBLOG AUDIT(NONE) +
        UACC(READ) OWNER(ownerid)
SETROPTS RACLIST(FACILITY) REFRESH
```

7. Activating logging for all subsystems.

   With these commands, logging is activated for all users:

```
RDEFINE FACILITY FILEM.IMS1.*.DBLOG AUDIT(NONE) +
        UACC(READ) OWNER(ownerid)
SETROPTS RACLIST(FACILITY) REFRESH
```

IMS™ subsystems and ZDT/IMS functions access control facility

ZDT/IMS allows you to control which IMS™ subsystems a user can access when using each of the functions listed in Table 48: Protected ZDT/IMS Functions on page 284. These functions are protected by default when you receive ZDT/IMS.

Table 48. Protected ZDT/IMS Functions

Function code	Description	UPDATE or READONLY
DBI	Initialize dialog - generates JCL for the initialize function	UPDATE
DDD	Delete or define dialog - generates JCL to delete or define database data sets	UPDATE
DIB	Initialize - initialize databases (batch)	UPDATE
IB	Browse - browse a database	READONLY
IBBO	Batch browse dialog - generate JCL for the batch browse function	READONLY
IBB	Batch browse - read a database in batch (batch)	READONLY
IE	Edit - edit a database	UPDATE
IEBO	Batch edit dialog - generates JCL for the batch edit function	UPDATE
IEB	Batch edit - edit a database in batch (batch)	UPDATE
IPRO	Print dialog - generates JCL for the print function	READONLY
IPR	Print - print data from databases (batch)	READONLY
IX	Extract dialog - generates JCL for the extract function	READONLY
IXB	Extract - extract data from databases (batch)	READONLY
IL	Load dialog - generates JCL for the load function	UPDATE
ILB	Load - load data into databases (batch)	UPDATE

You can grant or deny some or all users access to:

1. Individual IMS™ subsystems by individual functions in Table 48: Protected ZDT/IMS Functions on page 284.
2. Individual functions in Table 48: Protected ZDT/IMS Functions on page 284. When you grant or deny users access to individual functions, they are granted or denied access to all IMS™ subsystems when using these functions.
3. Individual IMS™ subsystems by the update or read-only functions.
4. The update or read-only functions. When you grant or deny users access to the update or read-only functions, they are granted or denied access to all IMS™ subsystems when using the update or read-only functions.

ZDT/IMS provides security for these functions, in one of two ways, either through RACF® (or an equivalent security product).

If Security Server RACF® or an equivalent security product is active, the System Authorization Facility (SAF) with the Z Data Tools enhanced security facility is used for access control and authorization verification. Authorization is controlled by ZDT/IMS-specific profiles in the FACILITY class. This chapter describes the ZDT/IMS-specific profiles that you must define to RACF® or your equivalent security product. It also describes how you define these profiles to RACF®. If you use another security product, consult the documentation for your product to determine how to define these profiles to your product.

Note: ZDT/IMS functions that are not listed in Table 48: Protected ZDT/IMS Functions on page 284 cannot be protected by RACF® (or an equivalent security product).

The rest of this section describes how you implement security controls in RACF® (or an equivalent security product) for the functions in Table 48: Protected ZDT/IMS Functions on page 284.

Controlling access to the update or read-only functions

The update functions in Table 48: Protected ZDT/IMS Functions on page 284 are protected by the profile FILEM.IMS.UPDATE. The read-only functions in this table are protected by the profile FILEM.IMS.RDONLY. As a minimum, you need to define these profiles and grant or deny users access to them.

Enter the following RACF® commands to define these profiles in the FACILITY class:

```
RDEFINE FACILITY FILEM.IMS.UPDATE UACC(READ or NONE)
RDEFINE FACILITY FILEM.IMS.RDONLY UACC(READ or NONE)
```

Specify:

- UACC(READ), if you want users or groups to be granted access to these resources, unless they are specifically denied access.
- UACC(NONE), if you want users or groups to be denied access to these resources, unless they are specifically granted access.

In the following, assume that:

- The RDEFINE for the FILEM.IMS.UPDATE profile specifies UACC(NONE), so users and groups are denied access to the update functions unless they are specifically granted access.
- The RDEFINE for the FILEM.IMS.RDONLY profile specifies UACC(READ), so users and groups are granted access to the read-only functions unless they are specifically denied access.

To grant a user (with user ID *userid*) or a group (with groupid *groupid*) access to the update functions, you enter one of the following RACF® commands:

```
PERMIT FILEM.IMS.UPDATE CLASS(FACILITY) ID(userid) ACCESS(READ)
PERMIT FILEM.IMS.UPDATE CLASS(FACILITY) ID(groupid) ACCESS(READ)
```

To deny a user (with user ID *userid*) or a group (with groupid *groupid*) access to the read-only functions, you enter one of the following RACF® commands:

```
PERMIT FILEM.IMS.RDONLY CLASS(FACILITY) ID(userid) ACCESS(NONE)
PERMIT FILEM.IMS.RDONLY CLASS(FACILITY) ID(groupid) ACCESS(NONE)
```

Controlling access to individual IMS™ subsystems by the update or read-only functions

Access to individual IMS™ subsystems by the update functions in Table 48: Protected ZDT/IMS Functions on page 284 is protected by the profiles FILEM.IMS.UPDATE.ssid, where ssid is the IMS™ subsystem ID. Access to individual IMS™ subsystems by the read-only functions in Table 48: Protected ZDT/IMS Functions on page 284 is protected by the profiles FILEM.IMS.RDONLY.ssid, where ssid is the IMS™ subsystem ID.

If you want to grant or deny some or all users access to individual IMS™ subsystems by the update or read-only functions in Table 48: Protected ZDT/IMS Functions on page 284, you will need to define the aforementioned profiles.

Enter the following RACF® commands to define these profiles in the FACILITY class:

```
RDEFINE FACILITY FILEM.IMS.UPDATE.ssid UACC(READ or NONE)
RDEFINE FACILITY FILEM.IMS.RDONLY.ssid UACC(READ or NONE)
```

Specify:

- UACC(READ), if you want users or groups to be granted access to these resources, unless they are specifically denied access
- UACC(NONE), if you want users or groups to be denied access to these resources, unless they are specifically granted access.

You use the PERMIT commands to grant or deny users and groups access to these resources, in the same way as for the FILEM.IMS.UPDATE and FILEM.IMS.RDONLY profiles described in Controlling access to the update or read-only functions on page 285.

Controlling access to individual functions

Individual functions in Table 48: Protected ZDT/IMS Functions on page 284 are protected by the profiles FILEM.FUNCTION.*fc*, where *fc* is the function code. If you want to grant or deny some or all users access to individual functions in Table 48: Protected ZDT/IMS Functions on page 284, you will need to define the profiles for these functions.

Enter the following RACF® command to define these profiles in the FACILITY class:

```
RDEFINE FACILITY FILEM.FUNCTION.fc UACC(READ or NONE)
```

Specify:

- UACC(READ), if you want users or groups to be granted access to this function, unless they are specifically denied access
- UACC(NONE), if you want users or groups to be denied access to this function, unless they are specifically granted access.

You use the PERMIT commands to grant or deny users and groups access to these function, in the same way as for the *FILEM.IMS.UPDATE and FILEM.IMS.RDONLY* profiles.

Controlling access to individual IMS™ subsystems by individual functions

Access to individual IMS™ subsystems by individual functions in Table 48: Protected ZDT/IMS Functions on page 284 is protected by the profiles FILEM.FUNCTION.*fc*.*ssid*, where *fc* is the function code and *ssid* is the IMS™ subsystem ID.

If you want to grant or deny some or all users access to individual IMS™ subsystems by individual functions in Table 48: Protected ZDT/IMS Functions on page 284, you will need to define the aforementioned profiles.

Enter the following RACF® command to define these profiles in the FACILITY class:

```
RDEFINE FACILITY FILEM.FUNCTION.fc,ssid UACC(READ or NONE)
```

Specify:

- UACC(READ), if you want users or groups to be granted access to these resources, unless they are specifically denied access
- UACC(NONE), if you want users or groups to be denied access to these resources, unless they are specifically granted access.

You use the PERMIT commands to grant or deny users and groups access to these resources in the same way as for the FILEM.IMS.UPDATE and FILEM.IMS.RDONLY profiles described in Controlling access to the update or read-only functions on page 285.

What governs whether access is granted or denied

Access to a subsystem *ssid* by an *update* function in Table 48: Protected ZDT/IMS Functions on page 284 with function code *fc* is governed by the first profile in this list

- FILEM.FUNCTION.*fc*.*ssid*
- FILEM.FUNCTION.*fc*
- FILEM.IMS.UPDATE.*ssid*
- FILEM.IMS.UPDATE

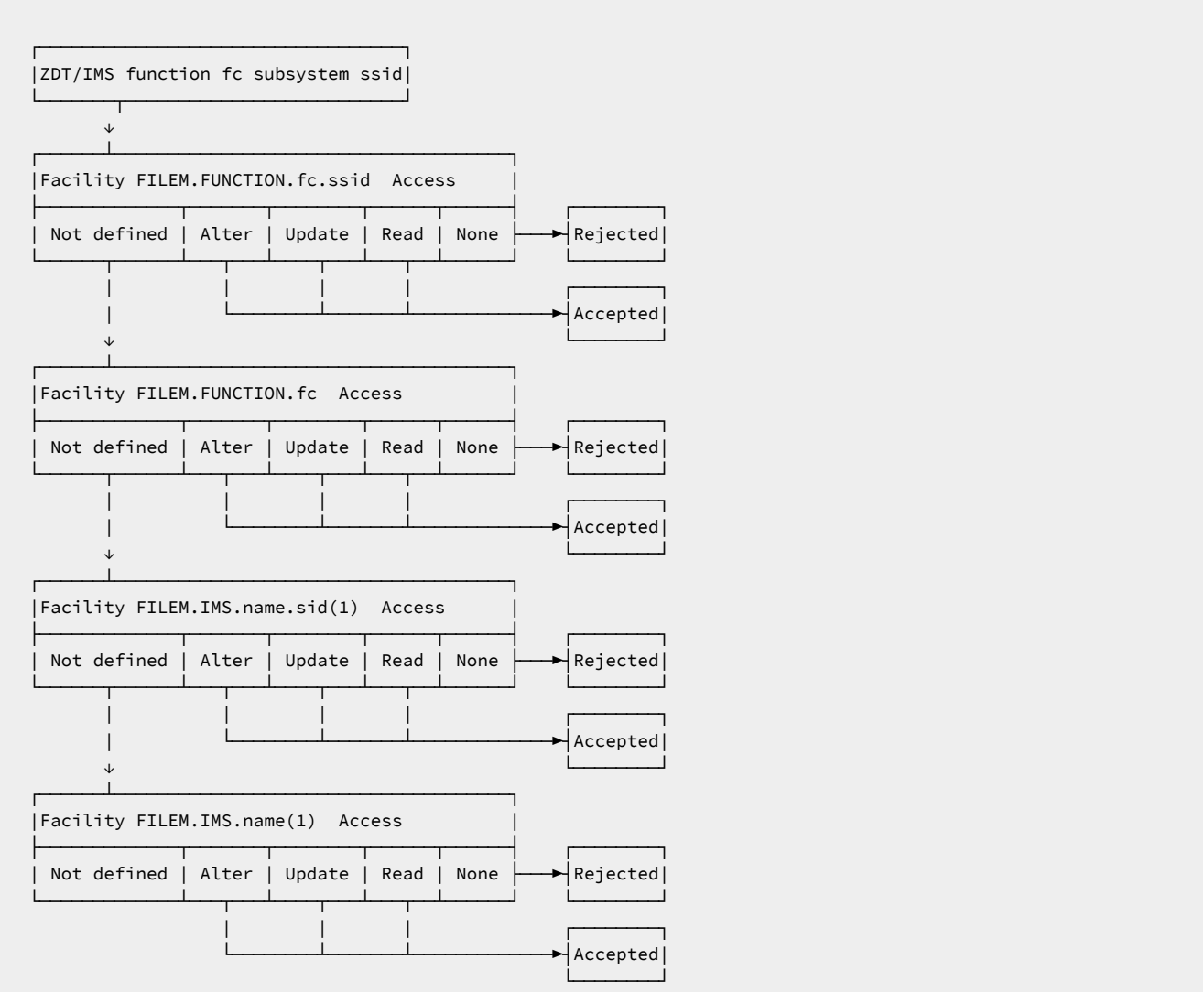
Access to a subsystem *ssid* by a *read-only* function in Table 48: Protected ZDT/IMS Functions on page 284 with function code *fc* is governed by the first profile in this list that has been defined in the FACILITY class:

- FILEM.FUNCTION.*fc*.*ssid*
- FILEM.FUNCTION.*fc*

- FILEM.IMS.RDONLY.*ssid*
- FILEM.IMS.RDONLY

Figure 55: Security checking for ZDT/IMS functions on page 288 illustrates the security checking that ZDT/IMS performs when a function in Table 48: Protected ZDT/IMS Functions on page 284 attempts to access an IMS™ subsystem.

Figure 55. Security checking for ZDT/IMS functions



Note:

1. FILEM.IMS.name is either FILEM.IMS.UPDATE or FILEM.IMS.RDONLY

ALTER, UPDATE or READ access means that the user can use the function. Access NONE means that the user cannot use the function.

Important information for users of non-IBM security products

ZDT/IMS issues a RACROUTE TYPE=AUTH for FILEM.FUNCTION.*fc.ssid*, the first profile in the list. If this profile is not defined RACF® returns RC=4. If RC=4 is returned, ZDT/IMS issues a RACROUTE for the second, third, and fourth profile in the list, in turn, until RACF® returns something other than RC=4.

Note that not all non-IBM security products issue RC=4 when a RACROUTE TYPE=AUTH is issued for a profile that is not defined; in this case ZDT/IMS only issues a RACROUTE for FILEM.FUNCTION.*fc.ssid* and, if this profile has not been defined, the request fails immediately. If this applies to your security product, your only option is to use the FILEM.FUNCTION.*fc.ssid* profiles to control access to your ZDT/IMS functions and IMS™ subsystems.

Customizing the ZDT/IMS security exit

ZDT/IMS provides a security exit module, HFM1SXT. HFM1SXT is called from four different points during processing.

- Exit Type A - Prior to allocating an audit trail data set when editing a database.
- Exit Type D - When allocating a database data set (DLI mode only).
- Exit Type I - Prior to invoking the IMS™ region controller.
- Exit Type T - After the IMS™ region controller terminates.

The version of HFM1SXT that is distributed with ZDT/IMS performs no security checking. Each exit type returns control immediately, thus allowing normal processing to continue.

You can provide your own version of HFM1SXT using either of the sample source decks, HFM1XITA (High Level Assembler) or HFM1XITC (COBOL), as a base. HFM1XITA and HFM1XITC are distributed in HFM1.SHFMSAM1.

Tip: Do not use ISPLINK calls to invoke ISPF services from an HFM1SXT security exit. This is because the ISPF environment does not exist when ZDT/IMS is invoked from batch and HFI clients.

Types of security exits

This topic describes what you can use each security exit type for.

Audit Trail Exit - Type A

The Audit Trail exit can be used to:

- Force the creation of an audit trail for a certain database or group of databases or for a certain user or group of users, or both.
- Override the standard audit trail data set name that is constructed by ZDT/IMS.
- Force the use of System Management Facilities (SMF) recording for the audit trail instead of using an audit trail data set. If you want to use SMF recording, the SMF record ID to be used must have been specified in the options macro, HFM1POPT.
- This exit is called only for the Edit function.

Database Data set Allocation Exit - Type D

The database data set allocation exit is called when the function runs in DLI mode. For example, this exit can be used to:

- Control access to database data sets.
- Override the database data set allocation status from OLD to SHR during database edit if IMS™ data sharing is used at your installation.

IMS™ Initialization Exit - Type I

The IMS™ Initialization exit can be used to:

- Control access to databases.
- Validate the value entered by the user for the IMS™ log data set. You can either accept, override, or disallow the value entered by the user.
- Override the standard ZDT/IMS log data set naming conventions.
- Override the Profile Option MAXGN.

IMS™ Termination Exit - Type T

The IMS™ Termination exit can be used to perform post-IMS processing of the log data set. This exit type has no parameters, it is provided as a point where you can add your own REXX code for termination processing

Invoking the security exit

The security exit program is invoked as follows:

```
CALL HFM1SXT (FM_SECURITY_PARAMETERS, FM_SECURITY_WORKAREA, FM_IMS_SECURITY_PARAMETERS)
```

The security exit interface parameters are:

- The security parameter list passed to the security exit. This parameter list is documented in the following pages.
- A security area work area which is a 256-byte area initialized to binary zeroes. This area is unchanged by ZDT/IMS and can be used to pass information between multiple calls to the security exit.
- IMS™ security parameters which are passed to the IMS™ Initialization and Termination exits.

Common Exit Parameters

Table 49. Common Parameters - All Exit Types

Field	Update	Size	Description
Request Type	N	Char(1)	**A** Audit Trail allocation **D** Database data set allocation

Table 49. Common Parameters - All Exit Types (continued)

Field	Update	Size	Description
			I IMS™ Initialization **T** IMS™ Termination
Option	N	Char(1)	**B** Browse **E** Edit **L** Extract/Load **P** Batch printing **U** Utilities
User Id	N	Char(7)	TSO user ID
Permitted	Y	Char(1)	**Y** Allow intended action **N** Disallow intended action **Blank** N/A Not used for Exit Type A.

Security Exit Parameters

Table 50. Parameters - Exit Type A.

Field	Update	Size	Description
DB DSN	N	Char(44)	This field is obsolete and no longer populated.
DSN	Y	Char(44)	The Audit Trail DSN to be allocated.
SMF Record Id	Y	Binary(16)	Default is 00 (SMF not used for audit trail). Choose a number between 128 and 255. Only applies if the 'Create Indicator' (below) is set to Y.

Table 50. Parameters - Exit Type A. (continued)

Field	Update	Size	Description
Create	Y	Char(1)	Input value from the Edit Entry Panel. Override the value with one of the following: **Y** Create an audit trail. **N** Do not create an audit trail. **D** See IMSAUDLG=D description.
Keep	Y	Char(1)	**Y** Keep the audit trail data set. **N** Delete the audit trail data set after printing.
Report	Y	Char(1)	**Y** Produce the Audit Trail report at the end of the edit session. **N** Do not produce the Audit Trail report at the end of the edit session.
Job Type	N	Char(1)	**B** DLI mode **M** BMP mode
DBD DSN	N	Char(44)	The DBD library that has been allocated.
DBD Name	N	Char(8)	The DBD being processed.
IMS™ System Id	N	Char(4)	The IMS™ SYSTEM ID entered.
Appl. Group Name	N	Char(8)	The AGN used.
PSB Type	N	Char(1)	**S** Static **D** Dynamic

Table 50. Parameters - Exit Type A. (continued)

Field	Update	Size	Description
PSB Name	N	Char(8)	The name of the Program Specification Block (PSB).
PSB DSN	N	Char(44)	The name of the PSB library data set (static DLI).

Table 51. Parameters - Exit Type D.

Field	Update	Size	Description
Primary DBD Name	N	Char(8)	The primary DBD being processed.
Physical DBD Name	N	Char(8)	The physical DBD that has the DATASET= statement for this DDNAME.
DDNAME	N	Char(8)	The DDNAME of the database being allocated.
DSN Status	Y	Char(3)	SHR or OLD. Default is SHR for Browse and OLD for Edit.
DSN	N	Char(44)	The DSN of the database.

Table 52. Parameters - Exit Type I

Field	Update	Size	Description
DBD DSN	N	Char(44)	The DBD library that has been allocated.
DBD Name	N	Char(8)	The DBD being processed.
Subfunction	N	Char(1)	Option subfunction. Function L, Extract/Load: **E** Extract **L** Load Function U, Utilities: **D** Delete/define database data sets **I** Initialize IMS™ databases

Table 52. Parameters - Exit Type I (continued)

Field	Update	Size	Description
IMS™ Log Indicator	Y	Char(1)	The user entered IMS™ log indicator. Only applicable during the online portion of the Edit and Load functions. Values are: **K** Allocate and keep log data set. **D** Allocate and delete log data set. **N** Do not use a log data set
IEFRDER DSN	Y	Char(44)	The IMS™ log data set name as constructed by ZDT/IMS when the 'IMS log data set' option has been selected in the Edit and Load online functions.
Job Type	N	Char(1)	**B** DLI mode **M** BMP mode
IMS™ System Id	N	Char(4)	The IMS™ SYSTEM ID entered.
Appl. Group Name	N	Char(8)	The AGN used.
Processing Option	N	Char(1)	The Database Load Processing Option as specified by the user. **1** Update and insert segments. **2** Insert new segments only.
Time	N	Char(6)	The time the exit call is made. Format is HHMMSS.
Date	N	Char(8)	The date the exit call is made. Format is YYYYMMDD.
PSB Type	N	Char(1)	**S** Static **D** Dynamic

Table 52. Parameters - Exit Type I (continued)

Field	Update	Size	Description
PSB Name	N	Char(8)	The name of the Program Specification Block (static PSB).
PSB DSN	N	Char(44)	The name of the PSB library data set (static DLI).
MAXGN Value	Y	Binary(16)	This value is only applicable during the Edit and Browse functions. The maximum number of GN (Get Next) calls allowed to satisfy a FIND or CHANGE command. Used to override the Profile Option MAXGN value.

Sample programs for a security exit

Copybooks, sample program source and JCL for the security exit are supplied for High Level Assembler and COBOL. The sample program source and JCL are distributed in HFM.SHFMSAM1, and the copybooks in HFM.SHFMMAC1. They are:

HFM1AXIT

HLASM copybook for security exit parameters.

HFM1XITA

Sample HLASM code for program HFM1SXT.

HFM1UMDS

Usermod to install an HLASM version of HFM1SXT.

HFM1CXIT

COBOL copybook for security exit parameters.

HFM1XITC

Sample COBOL code for program HFM1SXT.

HFM1SECC

Job control to install a COBOL version of HFM1SXT.

To provide your version of HFM1SXT in HLASM you use the usermod HFM1UMDS as follows:

1. Copy the member HFM1XITA from HFM.SHFMSAM1 to your own source library.
2. Code your version of HFM1XITA in your source library, using HFM1XITA from HFM.SHFMSAM1 as a base.
3. Modify the usermod HFM1UMDS member in HFM.SHFMSAM1 to meet your requirements. Refer to the usermod for information about changes you might need to make.
4. Install SMP/E usermod HFM1UMDS.

To provide your version of HFM1SXT in COBOL:

1. Copy the member HFM1XITC from HFM.SHFMSAM1 to your own source library.
2. Code your version of HFM1XITC in your source library, using HFM1XITC from HFM.SHFMSAM1 as a base.
3. Modify the sample job HFM1SECC in HFM.SHFMSAM1 to meet your site's requirements. Refer to the sample job for information about any changes you might need to make.
4. Run the job to HFM1SECC to compile and link your version of HFM1SXT. This job will link HFM1SXT into HFM.SHFMMOD1.

To implement your exit, add HFM.SHFMMOD1 to your LINKLIST or to the STEPLIB DD statement in your TSO logon procedure.



Note: If the security exit program is written in COBOL, the performance of the application may be impacted.

Chapter 25. Customizing the Z Data Tools audit facility for IMS™ component

ZDT/IMS can optionally write audit log records to either SMF or an audit log data set.

If auditing is not required:

- Set IMSAUDLG=N in the HFM1POPD macro specification of the HFM1POPT module. See AUDITLOG on page 388.
- Set SMFNO=0 in the HFM0POPI macro specification of the HFM1POPT module. See SMFNO on page 416 for more information.
- Skip the customization described in the rest of this chapter.

Z Data Tools provides two different methods (HFM1POPT-controlled and SAF-controlled audit) for controlling whether audit records are written for Z Data Tools IMS™ component. These are described in detail in Alternatives for controlling ZDT/IMS auditing on page 258.

You should determine which of the two methods is appropriate for your site's requirements.

Use the checklist to determine the customization required for the Z Data Tools IMS™ audit facility.

Table 53. Checklist for customizing ZDT/IMS auditing. This table lists choices and decisions.

Audit customization choice	Decision (Yes|No|Not applicable)
1. Control auditing using the HFM1POPT options module	
2. Control auditing using SAF rules and a member in SYS1.PARMLIB	
3. Control auditing using SAF rules, without any changes to SYS1.PARMLIB	
4. Audit records are to be written to a data set	
5. Audit records are to be written to SMF	

For choices 1 to 3, you should answer YES for one choice only. Mark the other two choices as 'Not applicable'.

If you answer YES for choice 1, you can answer YES for one of choices 4 or 5. Mark the other choice as 'Not applicable'.

If you answer YES for choices 2 or 3, you can answer YES for one or both of choices 4 and 5. If you answer YES for both choices you are implementing dual-logging.

Table 54. Customization steps for audit customization choices. This table lists choices and related actions.

Customization choice	Sections to complete
1. Control auditing using the HFM1POPT options module	• HFM1POPT-controlled audit logging on page 298
2. Control auditing using SAF rules and a member in SYS1.PARMLIB	• SAF-controlled auditing for Z Data Tools IMS component on page 300 • Implementing SAF-rule controlled auditing on page 303
3. Control auditing using SAF rules, without any changes to SYS1.PARMLIB	• SAF-controlled auditing for Z Data Tools IMS component on page 300 • Implementing SAF-rule controlled auditing on page 303
4. Audit records are to be written to a data set	• Audit data set configuration on page 299
5. Audit records are to be written to SMF	• Customizing Z Data Tools to write audit records to SMF on page 86

HFM1POPT-controlled audit logging

This was the original method of controlling auditing and, as such, only provides limited functionality.

With this method, you control audit logging by specifying the required values for the IMSAUDLG parameter on the HFM1POPI macro statement and the SMFNO parameter on the HFM0POPI macro statement.

You use the IMSAUDLG parameter to specify whether or not auditing is enforced for ZDT/IMS Edit and, if it is, whether or not an audit report is generated at the end of the Edit session. For information on how to code the IMSAUDLG parameter, see IMSAUDLG on page 470.

You use the SMFNO to specify whether the audit records are written to SMF or an audit log data set, and, if they are written to SMF, the SMF record type. For information on how to code the SMFNO parameter, see SMFNO on page 416.

If auditing of an Edit session is not enforced, users can still get ZDT/IMS to create an audit trail of their updates, by selecting the Create audit trail option on the Edit Entry panel.

You can further customize how ZDT/IMS records an audit trail by means of the Audit Trail exit in the ZDT/IMS security exit, HFM1SXT. For example, using this exit you can:

- Force the creation of an audit trail for a certain database or group of databases or for a certain user or group of users, or both.
- Override the standard audit trail data set name that is constructed by ZDT/IMS.
- Override Create audit trail as set by your users.
- Force the use of SMF recording for the audit trail instead of using an audit trail data set.
- Force audit logging during Edit and at the conclusion of the Edit session submit an audit report job to report on the changes. The job submitted is determined by skeleton member HFM1FTAD found in HFM.SHFMSLIB. Customize the job card and JCL to specify the reporting options you require. For changing the audit report options in the skeleton see "AUD (Print Audit Trail Report)" in the *Z Data Tools Users Guide and Reference*. For more information, see IMSAUDLG on page 470.

Note: You do not use the AUDITLOG option in HFM0POPT to determine whether or not to produce an audit trail in ZDT/IMS. The AUDITLOG option has no effect in ZDT/IMS.

Audit data set configuration

The format of the audit log data set name is determined by the setting of the AUDITHLQ parameter in the HFM0POPI definition in HFM1POPT. See AUDITHLQ on page 386 for more information about the AUDITHLQ option.

The following data set name formats may be generated:

- `userid.IMSAUDIT.Dyymmdd.Thhmmss` (when AUDITHLQ= (blank))
- *`audithlq`*`.IMSAUDIT.Dyymmdd.Thhmmss` (when AUDITHLQ=*audithlq*)
- `qual1.<qual2.><qual3.>Dyymmdd.Thhmmss` (when AUDITHLQ=*qual1.<qual2.><qual3>*)

where:

audithlq

Any 1-8 character constant that is valid in the context of a data set name.

userid

The user ID creating the data set

Dyymmdd

The date of the activity.

Thhmmss

The time of the activity.

When AUDITHLQ contains one or more periods, the AUDITHLQ value is treated as a data set prefix, with one, two or three levels. Each level of the prefix can be:

XXX

Any 1-8 character constant that is valid in the context of a data set name.

&&PREFIX

Indicates that the user's TSO prefix should be used. This is null if TSO NOPREFIX is in effect and, after substitution, the appropriate level of the audit log data set name prefix is also null.

&&USER

Indicates that the user's logonid (ISPF system variable ZUSER, stored in the shared pool) should be used.

&&UID

Indicates that the user's TSO prefix should be used, when the value is non-blank. When TSO NOPREFIX is in effect, the user's TSO logonid (ISPF system variable ZUSER, stored in the shared pool) should be used.

&&FUNCOD

Indicates that the Z Data Tools internal function code should be used. Specifying this parameter allows the Z Data Tools function that generated the audit log data set to be included in the audit log data set name.

&&SSID

Indicates that the currently connected IMS™ subsystem name should be used.

Set the AUDITHLQ parameter in the HFM0POPI macro for the HFM1POPT to the required value, based on the above information and your site's requirements.

You can print the information in a Z Data Tools IMS™ audit data set using the ZDT/IMS Print Audit Trail utility. To do this select option 3.8 from the ZDT/IMS Primary Option Menu.

SAF-controlled auditing for Z Data Tools IMS™ component

There are two methods for implementing SAF-controlled auditing for Z Data Tools IMS™ component. These are:

1. Control auditing for Z Data Tools IMS™ component using an enabling SAF Facility class rule and a member in SYS1.PARMLIB.

   To use this method complete the customization described in SAF-controlled auditing using SYS1.PARMLIB on page 300.

2. Control auditing for Z Data Tools IMS™ component using an enabling SAF Facility class rule, without any changes to SYS1.PARMLIB.

   To use this method complete the customization described in SAF-controlled auditing without SYS1.PARMLIB on page 303.

Important: If you use a security product other than RACF®, review the information in When a security product other than RACF is in use on page 94 to avoid one possible cause of S047 abends.

SAF-controlled auditing using SYS1.PARMLIB

You need to define an enabling SAF facility profile as described below:

Define SAF facility profile

```
FILEM.PARMLIB.IMS
```

and ensure all Z Data Tools IMS™ users to be audited have at least read access to that facility. See the example below:

Example

User PROD1 to have SAF-rule controlled auditing using SYS1.PARMLIB.

Write this RACF® rule:

```
RDEF FACILITY FILEM.PARMLIB.IMS AUDIT(NONE) UACC(NONE) OWNER(ownerid)
PE FILEM.PARMLIB.IMS ACC(READ) ID(PROD1) CLASS(FACILITY)
```

Add member HFM1PARM to SYS1.PARMLIB (or any other library in the logical parmlib concatenation). See Defining the HFM1PARM member on page 301.

Once the above SAF rule is defined and activated, auditing for Z Data Tools IMS™ component users is controlled by the FMAUDIT parameter in the HFM1PARM member. See ZDT/IMS options specified in HFM1PARM on page 521 for more information. If audit log records are to be written to SMF, the SMF record number is specified as an FMAUDIT parameter option. See FMAUDIT on page 521, and SMF_NO on page 521.

Note: Z Data Tools IMS™ component does not start if a user has read access to the above facility and the HFM1PARM member does not exist in the logical parmlib concatenation.

If SAF processing is not active, or the rule is not defined, or the rule is defined and the user has no access, then no parmlib processing is performed.

Defining the HFM1PARM member

If auditing is to be controlled from parmlib (user has read access to FILEM.PARMLIB.IMS, see SAF-controlled auditing for Z Data Tools IMS component on page 300), then member HFM1PARM must be defined in SYS1.PARMLIB (or any other library in the logical parmlib concatenation) as follows.

Default parmlib member HFM1PARM is provided in the SHFMSAM1 library. Copy this member to the appropriate system parmlib library.

Note: The sample HFM1PARM member supplied in SHFMSAM1 also includes a FMSECRTY statement. This option is not used at present, and can be either omitted, or commented out. It has no effect.

There are two methods that can be used to include the HFM1PARM member in a library in the logical parmlib concatenation. The choice of method depends on whether the installation's security software is configured to allow ZDT/IMS users READ access to the data set SYS1.PARMLIB.

Method 1 can only be used when ZDT/IMS users have read access to SYS1.PARMLIB.

Method 2 can be used regardless of whether ZDT/IMS users have READ access to SYS1.PARMLIB or not, and must be used when ZDT/IMS users do not have READ access to SYS1.PARMLIB.

Method 1

Place the HFM1PARM member in any library in the current logical parmlib concatenation. No IPL or other action is required to activate the new member (unless a new library was added to the logical parmlib concatenation).

Note:

1. Method 1 cannot be used in any situation where ZDT/IMS users do not have READ access to SYS1.PARMLIB. For example, when ZDT/IMS users have READ access to another library in the logical parmlib concatenation, and the HFM1PARM member is placed in the latter library. This will not work. The key issue is whether the ZDT/IMS user has READ access to SYS1.PARMLIB.

Method 2

This method must be used when ZDT/IMS users do not have READ access to SYS1.PARMLIB.

1. Create a new library with dataset attributes similar to SYS1.PARMLIB.

   The library name for this data set must include the string "HFMPARM" in one of the qualifiers. You can choose any data set name that meets this requirement. Examples of suitable data set names are:

   SYS1.PARMLIB.HFMPARM
   SYS8.HFMPARM.PARMLIB
   HFMPARM.SYS8.PARMLIB
   SYS2.HFMPARMS.LIB
   SYS8.XHFMPARM.PARMLIB
2. Add member HFM1PARM to the new library, specifying the appropriate FMAUDIT parameter.
3. Add the new library to the logical parmlib concatenation. This can be done dynamically, or by means of a system IPL.

Note: When Method 2 is used, the HFM1PARM member must be located in the library created in step 1 on page 302. If the HFM1PARM member specifies any include statements (see Facilities for customizing the HFM1PARM definitions on page 522), all of the included members must also reside in the same library.

You use the HFM1PARM member to define:

- Whether ZDT/IMS uses SAF to control ZDT/IMS audit logging.
- The SAF resource name prefix to be used by ZDT/IMS when determining access to various resources.
- Whether ZDT/IMS loads the HFM1POPT module from a specific library.

For more information, see ZDT/IMS options specified in HFM1PARM on page 521.

SAF-controlled auditing without SYS1.PARMLIB

You need to define an enabling SAF facility profile as described below:

Define SAF facility profile

```
FILEM.SAFAUDIT.IMS
```

and ensure that all Z Data Tools IMS™ users to be audited have at least read access to that facility. See the example below.

Example

User PROD2 to have SAF-rule controlled auditing without using SYS1.PARMLIB.

Write this RACF® rule:

```
RDEF FACILITY FILEM.SAFAUDIT.IMS AUDIT(NONE) UACC(NONE) OWNER(ownerid)
PE FILEM.SAFAUDIT.IMS ACC(READ) ID(PROD2) CLASS(FACILITY)
```

If you use this method and intend to write audit records to SMF, the required SMF number is specified in the HFM1POPT module. See Customizing Z Data Tools to write audit records to SMF on page 86 for more information.

Implementing SAF-rule controlled auditing

Use the checklist shown here to implement SAF-rule controlled auditing:

1. Determine the FACILITY and XFACILIT class profiles required to control ZDT/IMS audit logging at your installation. The information provided in this section should help you to do that:
   - Understanding how SAF controls ZDT/IMS audit logging on page 304.
2. Determine which level of access to these profiles your users will require to meet the audit logging requirements at your installation. The information provided in these sections should help you to do that:
   - Controlling where ZDT/IMS writes audit log records on page 305.
   - Controlling whether an audit trail is created on page 307.
   - Controlling if users can request an audit trail when one is not required (Edit function only) on page 308.
3. Define the required profiles to RACF® or your equivalent security product and provide users with the required access to these profiles. The information provided in these sections describes how you do this when using RACF®:
   - FILEM.AUDIT1.ssid.TOSMF on page 306.
   - FILEM.AUDIT1.ssid.TODSN on page 306.
   - FILEM.AUDIT1.ssid.fc.db. on page 308.
   - FILEM.AUDIT1.ssid.OPTION on page 308.

   If you use another security product, consult the documentation for your product.
4. Activate SAF-rule controlled auditing for the user IDs that you intend to use in Step 5. The information provided in these sections describe how you do that:
   - SAF-controlled auditing for Z Data Tools IMS component on page 300.
   - ZDT/IMS options specified in HFM1PARM on page 521.

5. Test your configuration to ensure that audit logging occurs when, and only when, it is required.
6. When you are satisfied that ZDT/IMS audit logging is only occurring when it should, activate SAF controlled audit logging for all ZDT/IMS users. The information provided in these sections describe how you do that:
    - SAF-controlled auditing for Z Data Tools IMS component on page 300.
    - ZDT/IMS options specified in HFM1PARM on page 521.

Understanding how SAF controls ZDT/IMS audit logging

Table 55: ZDT/IMS functions that support audit logging when audit logging is controlled by SAF on page 304 lists the ZDT/IMS functions that can be made to create an audit trail when audit logging is controlled by SAF.

Table 55. ZDT/IMS functions that support audit logging when audit logging is controlled by SAF

Function code	Function name	Description
IB	Browse	Browse a database
IBB	Batch Browse	Read a database in batch
IE	Edit	Edit a database
IEB	Batch Edit	Edit a database in batch
ILB	Load	Load data into databases (batch)
IPR	Print	Print data from a database (batch)
IXB	Extract	Extract data from databases (batch)

The following describes how SAF controls ZDT/IMS audit logging.

When functions in Table 55: ZDT/IMS functions that support audit logging when audit logging is controlled by SAF on page 304 are started, SAF is invoked to answer these audit queries:

- Whether audit logging is required.
- If audit logging is required, whether audit records should be written to SMF, the user's audit log data set, or both.
- For Edit and Browse function only, whether the user's audit log data set should be printed at the end of the Edit/Browse session.
- For Edit function only, whether the **Create audit trail** option on the Edit Entry panel can be used:
    - To request an audit trail when one is not required.
    - To stop an audit trail being created when one is required.

The responses to these queries are controlled by FACILITY and XFACILIT class profiles that you define.

Table 56: SAF profiles that control ZDT/IMS audit logging on page 305 lists the profiles that control the responses when the query is from a given ZDT/IMS function that is being used to access a given database in a given IMS™ subsystem, where:

- *ssid* is the IMS™ subsystem ID.
- *fc* is the function code.
- *db* is the database name.

Table 56: SAF profiles that control ZDT/IMS audit logging on page 305 lists the profile name (column 1), the class in which the profile must be defined (column 2), and what the profile controls (column 3).

Table 56. SAF profiles that control ZDT/IMS audit logging

SAF profile	Class	Description
FILEM.AUDIT1.*ssid*.TOSMF	FACILITY	Controls whether audit log records are written to SMF.
FILEM.AUDIT1.*ssid*.TODSN	FACILITY	1. Controls whether audit log records are written to the user's audit log data set. 2. Controls whether the user's audit log data set is printed at the end of the session (for Edit and Browse function only).
FILEM.AUDIT1.*ssid*.OPTION	FACILITY	Controls whether the **Create audit trail** option on the Edit Entry panel can be used to request an audit trail when one is not required (Edit function only).
FILEM.AUDIT1.*ssid.fc.db*	XFACILIT	1. Controls whether audit logging is required. 2. Controls whether the **Create audit trail** option on the Edit Entry panel can be used to stop an audit trail being created when one is required (Edit function only).

The following sections describe how you use these profiles to control ZDT/IMS audit logging and how you define these profiles to RACF®.

Controlling where ZDT/IMS writes audit log records

When a function in Table 55: ZDT/IMS functions that support audit logging when audit logging is controlled by SAF on page 304 accesses a given IMS™ subsystem, two profiles control where ZDT/IMS writes audit log records. These profiles are FILEM.AUDIT1.*ssid*.TOSMF and FILEM.AUDIT1.*ssid*.TODSN, where *ssid* is the IMS™ subsystem ID. Each of these profiles are discussed in turn.

FILEM.AUDIT1.*ssid*.TOSMF

This profile controls whether any audit log records that this function may create are written to SMF. If the user has no access to this profile, then the function does not write audit log records to SMF. If the user has READ (or higher) access to this profile, then any audit log records that this function may create are written to SMF.

FILEM.AUDIT1.ssid.TODSN

This profile controls whether any audit log records that this function may create are written to the user's audit log data set. For the Edit and Browse function, it also controls whether or not the user's audit log data set

is printed at the end of the Edit/Browse session. If the user has no access to this profile, then the function does not write audit log records to the user's audit log data set. If the user has READ (or higher) access to this profile, then any audit log records that this function may create are written to the user's audit log data set. If the user has UPDATE (or higher) access to this profile, and it is the Edit or Browse function, and the function created an audit trail, then the user's audit log data set is printed at the end of the Edit/Browse session.

If the user does not have access to either of these profiles, then ZDT/IMS does not create an audit trail under any circumstances. So, apart from controlling where the audit log is written, these profiles can also affect whether or not an audit trail is created.

The following describes how you define these profiles to RACF®.

FILEM.AUDIT1.*ssid*.TOSMF

You define this profile in the FACILITY class. You do this by entering this RACF® command:

```
RDEFINE FACILITY FILEM.AUDIT1.ssid.TOSMF UACC(READ or NONE)
```

`UACC(READ)`

Specify this if you want audit log records from users and groups to be written to SMF, unless they are specifically denied access to this profile.

`UACC(NONE)`

Specify this if you do not want audit log records from users and groups to be written to SMF, unless they are specifically granted access to this profile.

In the following, assume that the RDEFINE for the profile specifies UACC(NONE), so audit log records from users and groups are not written to SMF unless they are specifically granted access.

To specify that the audit log records from a user (with user ID *userid*) or a group (with groupid *groupid*) are to be written to SMF, you enter one of these RACF® commands:

```
PERMIT FILEM.AUDIT1.ssid.TOSMF CLASS(FACILITY) ID(userid) ACCESS(READ)
PERMIT FILEM.AUDIT1.ssid.TOSMF CLASS(FACILITY) ID(groupid) ACCESS(READ)
```

FILEM.AUDIT1.*ssid*.TODSN

You define this profile in the FACILITY class. You do this by entering this RACF® command:

```
RDEFINE FACILITY FILEM.AUDIT1.ssid.TODSN UACC(UPDATE or READ or NONE)
```

`UACC(UPDATE)`

Specify this if you want audit log records from users and groups to be written to their audit log data sets and, if it is an Edit or Browse, you want the audit log data sets printed at the end of the Edit/Browse session, unless they are specifically given some other access to this profile.

`UACC(READ)`

Specify this if you want audit log records from users and groups to be written to their audit log data sets, unless they are specifically given some other access to this profile.

`UACC(NONE)`

Specify this if you do not want audit log records from users and groups to be written to their audit log data set, unless they are specifically granted access to this profile.

In the following, assume that the RDEFINE for the profile specifies UACC(READ), so audit log records from users and groups are written to their audit log data set unless they are specifically denied access.

To specify that the audit log records from a user (with user ID *userid*) or a group (with groupid *groupid*) are not to be written to their audit log data set, you enter one of these RACF® commands:

```
PERMIT FILEM.AUDIT1.ssid.TODSN CLASS(FACILITY) ID(userid) ACCESS(NONE)
PERMIT FILEM.AUDIT1.ssid.TODSN CLASS(FACILITY) ID(groupid) ACCESS(NONE)
```

To specify that Browse and Edit audit log data sets from a user (with user ID *userid*) or a group (with groupid *groupid*) are to be printed at the end of the Edit/Browse session, you enter one of these RACF® commands:

```
PERMIT FILEM.AUDIT1.ssid.TODSN CLASS(FACILITY) ID(userid) ACCESS(UPDATE)
PERMIT FILEM.AUDIT1.ssid.TODSN CLASS(FACILITY) ID(groupid) ACCESS(UPDATE)
```

Controlling whether an audit trail is created

When a function in Table 55: ZDT/IMS functions that support audit logging when audit logging is controlled by SAF on page 304 accesses a given database in a given IMS™ subsystem, three profiles control whether ZDT/IMS creates an audit trail.

These profiles are:

FILEM.AUDIT1.*ssid*.TOSMF
FILEM.AUDIT1.*ssid*.TODSN
FILEM.AUDIT1.*ssid.fc.db*

where *ssid* is the IMS™ subsystem ID, *fc* is the function code, and *db* is the database name.

The FILEM.AUDIT1.*ssid*.TOSMF and FILEM.AUDIT1.ssid.TODSN profiles are discussed in Controlling where ZDT/IMS writes audit log records on page 305 where it mentions that if the user does not have access to either of these profiles, then no audit log is created under any circumstances. In the following, we assume that the user *does* have READ (or higher) access to one or both of these profiles.

The FILEM.AUDIT1.*ssid.fc.db* profile controls whether the function creates an audit trail.

If the user has no access to this profile, then the function does not create an audit trail. If the user has READ (or higher) access to this profile, then the function does create an audit trail. If the user has CONTROL (or higher) access to this profile and it is the Edit function, then the function creates an audit trail if, and only if, the user has selected the **Create audit trail** option on the Edit Entry panel.

The following describes how you define this profile to RACF®.

FILEM.AUDIT1.*ssid*.*fc*.*db*.

You define this profile in the XFACILIT class. You do this by entering this RACF® command:

```
RDEFINE XFACILIT FILEM.AUDIT1.ssid.fc.db UACC(READ or NONE)
```

UACC(READ)

Specify this if you want audit trails created for the users and groups, unless they are specifically denied access to this profile.

UACC(NONE)

Specify this if you do not want audit trails created for the users and groups, unless they are specifically granted access to this profile.

In the following, assume that the RDEFINE for the profile specifies UACC(NONE), so audit trails are not created for the users and groups, unless they are specifically granted access.

To specify that audit trails are to be created for a user (with user ID *userid*) or a group (with groupid *groupid*), you enter one of these RACF® commands:

```
PERMIT FILEM.AUDIT1.ssid.fc.db CLASS(XFACILIT) ID(userid) ACCESS(READ)
PERMIT FILEM.AUDIT1.ssid.fc.db CLASS(XFACILIT) ID(groupid) ACCESS(READ)
```

Controlling if users can request an audit trail when one is not required (Edit function only)

When the ZDT/IMS Edit function accesses a given IMS™ subsystem, the FILEM.AUDIT1.*ssid*.OPTION profile controls whether or not the **Create audit trail** option on the Edit Entry panel can be used to request an audit trail when one is not required.

If the user has no access to this profile, then the **Create audit trail** option cannot be used to request an audit trail when one is not required.

If the user has READ (or higher) access to this profile, then the **Create audit trail** option can be used to request an audit trail when one is not required, but only when the user has READ (or higher) access to either the FILEM.AUDIT1.*ssid*.TOSMF or the FILEM.AUDIT1.*ssid*.TODSN profiles. If the user does not have access to either of these profiles, then no audit log is created under any circumstances.

The following describes how you define this profile to RACF®.

FILEM.AUDIT1.*ssid*.OPTION

You define this profile in the FACILITY class. You do this by entering this RACF® command:

```
RDEFINE FACILITY FILEM.AUDIT1.ssid.OPTION UACC(READ or NONE)
```

UACC(READ)

Specify this if you want to allow users and groups to use the **Create audit trail** option to request an audit trail when one is not required. unless they are specifically denied access to this profile.

`UACC(NONE)`

Specify this if you don't want to allow users and groups to use the **Create audit trail** option to request an audit trail, unless they are specifically granted access to this profile.

Chapter 26. Customizing ZDT/IMS for national languages

You can customize ZDT/IMS for national languages other than English.

If you are using Japanese and have installed the ZDT/IMS Japanese component, you might not need to perform any other customization for the Japanese national languages.

If you are using a language other than English or Japanese, you will need to perform **all** of the customization tasks listed in Table 57: Summary of steps for customizing ZDT/IMS for a national language on page 310.

Table 57. Summary of steps for customizing ZDT/IMS for a national language

Step	Description
__ 1	Set the LANGUAGE option for batch processing, if required. See Setting the default national language on page 273.
__ 2	Change the TERMTYPE option in HFM1POPT, if you are using a DBCS language. See TERMTYPE on page 418.
__ 3	Confirm that you have the correct terminal type set in ISPF (ISPF option 0).
__ 4	Create a print and display translation table for your language. See Changing the print and display translation tables for languages other than English on page 310.
__ 5	Translate the Z Data Tools message text to your language. See Translating the message text on page 311.
__ 6	Provide a version of HFM1MENU for your language. See Providing a multicultural version of HFM1MENU on page 311.
__ 7	Translate the ZDT/IMS ISPF messages to your language. See Translating the ISPF messages text on page 312.
__ 8	Translate the ZDT/IMS panels to your language. See Translating the panel text on page 313.

Changing the print and display translation tables for languages other than English

If you plan to use ZDT/IMS with a national language other than English, you might need to provide a print and display translation table for your language.

You do this as part of your customization for Z Data Tools Base function. See Changing the print and display translation tables for languages other than English on page 104.

You should also specify PRTTRANS=ON in HFM1POPT. If you are using any DBCS language you might also need to specify TERMTYPE=3270KN in HFM1POPT.

This step is essential if you are using a DBCS language other than Japanese.

Translating the message text

All ZDT/IMS messages are stored in the HFM1MENU source member. This CSECT is part of the root module so that an English version of the messages is always available. In addition, all messages used by ZDT/IMS under ISPF are provided in the library, HFM.SHFMMENU. Using both HFM1MENU and members in HFM.SHFMMENU, you can provide your own set of translated messages. To use the messages you have translated, see Using the translated messages and panels on page 313.

To provide translated versions of the messages, you must provide a version of HFM1MENU in your language, as described in Providing a multicultural version of HFM1MENU on page 311 **and** provide translated versions of the appropriate members in HFM.SHFMMENU, as described in Translating the ISPF messages text on page 312.

You should not need to provide a Japanese version of the messages if you have installed the ZDT/IMS Japanese component.

Providing a multicultural version of HFM1MENU

HFM1MENU contains the assembler source for the ZDT/IMS messages. To provide translated versions of the messages:

1. Copy the member HFM1MENU from HFM.SHFMSAM1 to your own source library with the name HFM1M*yyy*, where *yyy* is one of the following language codes:

   FRA

   French

   DEU

   German

   ITA

   Italian

   JPN

   Japanese

   PTG

   Portuguese

   ESP

   Spanish

   DAN

   Danish

   ENP

   Upper case English

KOR

Korean

DES

Swiss German

CHT

Traditional Chinese

CHS

Simplified Chinese

XXX

Other

2. Change the message text in HFM1M*yyy* in your library.
3.

   Modify the HFM1UMDM member in HFM.SHFMSAM1 to meet your site's requirements, using the same language code as above. Refer to the usermod for information about other changes you might need to make.
4. Install SMP/E usermod HFM1UMDM.

Translating the ISPF messages text

All ZDT/IMS ISPF messages are provided in English.

They are provided in Japanese if you have installed the ZDT/IMS Japanese component

You can translate some or all of these messages into another language.

All ZDT/IMS ISPF messages are stored in HFM.SHFMMENU. You translate a message as follows:

1. Find the members in HFM.SHFMMENU that contain the messages you want to translate. The message members specific to ZDT/IMS are all named HFMI*zzzz*.
2. Create a library with the same characteristics as HFM.SHFMMENU, with the name HFM.SHFMM*yyy*, where *yyy* is the same language code you specified when you modified HFM1MENU. If you have already created a library with this name for translated Z Data Tools Base function messages, use that library. Copy the required message members from HFM.SHFMMENU to this library.
3. Change the required message texts in these members in your library.

To use the messages you have translated, see Using the translated messages and panels on page 313. For more information about defining and using ISPF messages, see *z/OS ISPF Dialog Developer's Guide*.

Note: Be sure to include **ALL** the messages in the message members you copy to your own library. When ZDT/IMS needs to display an ISPF message, it uses ISPF services to do this. Therefore the search for a message is made

according to ISPF rules. Thus, if ISPF finds the message member it requires in your library, but the message number required is not in that member, ISPF will not look in any other library for the message, but will give an error. However, if you omit a complete message member from your library, then ISPF will use the English message member from the next library in the ISPMLIB concatenation. For more information about defining and using ISPF messages, see *z/OS ISPF Dialog Developer's Guide*.

Translating the panel text

All ZDT/IMS ISPF panels are provided in English.

They are also provided in Japanese if you have installed the ZDT/IMS Japanese component. . You can translate some or all of these panels into another language. (If no translated version of a particular panel is available, ZDT/IMS uses the English version.)

All ZDT/IMS panels are stored in HFM.SHFMPENU. You translate a panel as follows:

1. Find the panel members in HFM.SHFMPENU that you want to translate. The panel members specific to ZDT/IMS are all named HFM1*zzzz*.
2. Create a library with the same characteristics as HFM.SHFMPENU, with the name HFM.SHFMP*yyy*, where *yyy* is the same language code you specified when you modified HFM1MENU. If you have already created a library with this name for translated Z Data Tools Base function panels, use that library. Copy the required panel members from HFM.SHFMPENU to this library.
3. Change the required panel text in the members in your library. A panel may reference a **help** panel by means of a `.HELP` statement. If any panel you are changing contains any of these `.HELP` statements, also copy and change these referenced members in your library.

To use the panels you have translated, see Using the translated messages and panels on page 313.

Using the translated messages and panels

To use your translated messages in a batch job, specify the appropriate language with the LANGUAGE processing option, using the keywords shown in Table 58: Keyword values for the LANGUAGE option on page 313. See Customizing the ZDT/IMS installation options module on page 264 for information on how to do this.

Table 58. Keyword values for the LANGUAGE option

Language	Code	Specify on LANGUAGE option...
French	FRA	FRENCH
German	DEU	GERMAN
Italian	ITA	ITALIAN
Japanese	JPN	JAPANESE
Portuguese	PTG	PORTUGUESE

Table 58. Keyword values for the LANGUAGE option (continued)

Language	Code	Specify on LANGUAGE option...
Spanish	ESP	SPANISH
Danish	DAN	DANISH
Upper case English	ENP	UPPERENG
Korean	KOR	KOREAN
Swiss German	DES	SGERMAN
Traditional Chinese	CHT	CHINESET
Simplified Chinese	CHS	CHINESES
Other	XXX	OTHER

For example, to use French messages, specify LANGUAGE=FRENCH.

Under ISPF, the language used in messages and panels is determined by the setting of the national language for ISPF, for the current ISPF session. For information on changing the national language setting for your ISPF session, see *z/OS ISPF Dialog Developer's Guide*.

If your ISPF session is set up to use your language, you need to add your libraries to the appropriate ISPF concatenation, in front of any Z Data Tools English libraries. For example, to use your translated messages, add HFM.SHFMM*yyy* to ISPMLIB, in front of HFM.SHFMMENU. To use your translated panels, add HFM.SHFMP*yyy* to ISPPLIB, in front of HFM.SHFMPENU.

Customizing for the Japanese national language

The other customization task you might need to do for the Japanese national language is to modify the supplied Japanese translation tables. If you want to do this, do so as part of the customization of the Z Data Tools Base function. See Modifying the Japanese translation tables on page 109.

Changing the Japanese message text

If you have installed the ZDT/IMS Japanese component, all ZDT/IMS Japanese messages are stored in the HFM1MJPN source member. Normally you should not need to modify this module. However, if you do want to modify it, you can do so by means of the usermod, HFM1UMDN.

To do this:

1. Copy the member HFM1MJPN from HFM.SHFMSAM1 to your own source library.
2. Change the message text in HFM1MJPN in your library.
3. Modify the HFM1UMDN member in HFM.SHFMSAM1 to meet your site's requirements. Refer to the usermod for information about other changes you might need to make.
4. Install SMP/E usermod HFM1UMDN.

Chapter 27. Verifying the customization of ZDT/IMS

After you have completed the initial installation and customization of ZDT/IMS, you can perform the following steps to verify your installation, before completing the installation process. You might need to refer to the *Z Data Tools User's Guide and Reference for IMS Data*.

Step 1. Build the sample IMS™ databases to be used during verification

Member HFM1IVP in HFM.SHFMSAM1 contains a sample job that builds sample IMS™ databases. These databases are used in the installation verification process. Instructions in the sample job describe the modifications you must make to the JCL before you submit it.

1. Following the instructions in the sample job, make the required modifications to the HFM1IVP job.
2. Run the HFM1IVP job. The expected return code is zero.

Member HFM1IVP1 in HFM.SHFMSAM1 contains a sample job that copies the sample database segment layouts to your COBOL and PL/I COPYLIB data sets. Instructions in the sample job describe the modifications you must make to the JCL before you submit it.

1. Following the instructions in the sample job, make the required changes to the HFM1IVP1 job.
2. Run the HFM1IVP1 job. The expected return code is zero.

Note: View and Template data sets are empty.

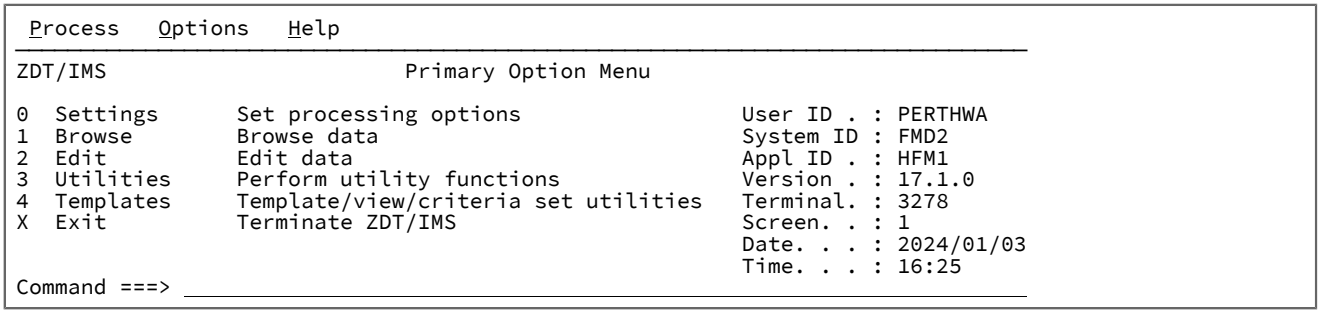
Step 2. Start ZDT/IMS

If you have added an option for ZDT/IMS to your ISPF Primary Options menu (see Adding ZDT/IMS to your ISPF menu on page 261), enter the option value you have assigned to ZDT/IMS and press Enter. For example, if you have assigned ZI to ZDT/IMS, enter ZI and press Enter.

If you defined ZDT/IMS in an ISPF command table (see Defining ZDT/IMS in an ISPF command table on page 261) verify that ZDT/IMS can be started by entering the command ZI on any ISPF command line.

The ZDT/IMS Primary Option Menu should be displayed.

Figure 56. Primary Option Menu



Note: The first time you use ZDT/IMS, a Copyright panel is displayed. After reading the panel text, press Enter. In subsequent ZDT/IMS sessions, this panel will not automatically appear.

Enter VER on the command line to display the release level and PTF level of ZDT/IMS. A panel is displayed, similar to this:

```
HCL Z Data Tools
                              Version 17 Release
                                    1 Modification 0
IMS Component
(not APF authorized)

Service Levels of installed components

             Base       IMS        Db2        CICS
English      -NONE-     -NONE-     -NONE-     -NONE-
```

Note:

1. This will always show ZDT/IMS as `not APF authorized`, even if you have made Z Data Tools APF-authorized, as Z Data Tools cannot run APF-authorized under ISPF.
2. When you first install Z Data Tools, `-NONE-` will be shown against each component. Subsequently, when you have applied service to Z Data Tools, a PTF number will be shown, indicating the PTF level of each component. If you have not installed a component, that component will not be shown at all. If you have installed the Japanese language component of ZDT/IMS, another line will be displayed indicating the service level of that component.

Step 3. Display settings menu

On the Primary Option Menu, enter **0** on the command line, for Settings, and press Enter.

The Settings Menu is displayed.

Figure 57. Settings Menu

```
 Process   Options   Help
──────────────────────────────────────────────────────────────────────────────
ZDT/IMS                          Settings Menu

1  Print      Print settings
2  System     System settings
3  Batch      Job card specifications
4  Compiler   Language and compiler specifications
5  EDIT       Editor options
6  Subsystem  IMS subsystem settings
7  Temporary  Temporary Data Set Allocations
8  Output     Output Data Set Allocations
9  Trace      Trace options

Command ===> ____________________________________________________________
```

Step 4. Display DLI mode settings menu

Enter **6** on the command line for Subsystem settings, and press Enter. The Subsystem Settings Menu is displayed.

Figure 58. Subsystem Settings Menu

```
 Process   Options   Help
--------------------------------------------------------------------------------
ZDT/IMS                      Subsystem Settings Menu

DLI mode settings
1  Parameters   Parameters passed to the IMS region controller
2  Data sets 1  DFSVSAMP, RESLIB and IMS macros data set names
3  Data sets 2  RECON and ACBLIB data set names
4  Options      Autosave, checkpoint frequencies and PROCOPTs

BMP mode settings
5  Parameters   Parameters passed to the IMS region controller
6  Options      Autosave, checkpoint frequencies and PROCOPTs

DLI and BMP mode settings
7  Data sets 1  PSB and DBD data set names
8  Data sets 2  Template data set names

IMS Subsystem Name   ____

Command ===> _____________________________________________________________________
```

Enter **1** on the command line, for Parameters, and leave the **IMS™ Subsystem Name** field blank, and press Enter. The DLI Mode Parameters : Subsystem Selection panel is displayed.

Step 5. Verify subsystem details

Figure 59. DLI Mode Parameters

```
 Process   Options   Help
--------------------------------------------------------------------------------
            DLI Mode Parameters : Subsystem Selection

                     Read  PSB    Region  AGNs
Cmd  SSID  Status    Only  Types  Types   Used  Description
___  IFA2  ACTIVE     N    BOTH   BMP      N    IFA2 - BMP only
___  IFB2  INACTIVE   Y    BOTH   BOTH     N    IFB2 - READONLY
___  IF92  ACTIVE     N    BOTH   BOTH     Y    IF92 - UAGNS=Y and 3 AGNs
___  IF82  ACTIVE     N    BOTH   BOTH     Y    IF82 - UAGNS=Y and no AGNs
___  IF72  ACTIVE     N    BOTH   BOTH     N    IF72 - UAGNS=N, READONLY=N
___  IF62  ACTIVE     N    BOTH   BOTH     Y    IF62 - UAGNS not spec, 6 AGNs
___  IF52  ACTIVE     N    DYN    BOTH     N    IF52 - DYNAMIC, all unprotect
___  IF42  ACTIVE     N    STAT   BOTH     N    IF42 - STATIC, all protected
___  IF32  ACTIVE     N    BOTH   BOTH     N    IF32 - DYNALLOC=Y
___  IF22  INACTIVE   N    BOTH   BOTH     N    IF22 - IMS V7
___  IF12  ACTIVE     Y    BOTH   BOTH     N    IF12 - UAGNS=N, READONLY=Y
    ****  End of data  ****

Command ===> _____________________________________________________________________
```

Check that all the subsystems that you specified in the ZDT/IMS installation options module are listed, and that the correct details are listed for each.

Select the subsystem that you are using for the verification process and press Enter. The DLI Mode Parameters panel is displayed.

Step 6. Verify DLI mode parameters

Figure 60. DLI Mode Parameters

```
 Process   Options   Help
--------------------------------------------------------------------------------
 ZDT/IMS                       DLI Mode Parameters

 Subsystem IF52  IF52 - DYNAMIC, all unprotect

 Options:
    Enter "/" to select option
    /  Dynamic backout

    DBRC                          IRLM
    2  1. Use DBRC                2  1. Use IRLM
       2. Do not use DBRC            2. Do not use IRLM
       3. Subsystem default          3. Subsystem default

 Parameters:
    IRLMNM  . . . IRLM
    GSGNAME . . .
    TMINAME . . .
    BUF . . . . .
    LOCKMAX . . . 555

Command ===> ____________________________________________________________
```

Enter **RESET** on the command line and press Enter.

Check that the settings displayed for the subsystem are the settings you specified in the ZDT/IMS installation options module.

Press **PF3** to save the settings in your profile and exit the panel. The Subsystem Settings Menu is again displayed.

Step 7. Verify DLI mode data sets

Enter **2** on the command line, for Data Sets 1, and press Enter. The DLI Mode Data Sets 1 panel is displayed.

Figure 61. DLI Mode Data Sets 1

```
 Process   Options   Help
--------------------------------------------------------------------------------
ZDT/IMS                       DLI Mode Data Sets 1

Subsystem IF52  IF52 - DYNAMIC, all unprotect

DFSVSAMP:
   Data set name . . 'IMSV11.IFB2.PROCLIB'
   Member  . . . . . DFSVSMDB

RESLIB:
   Data set name #1  'IMSV910.IF52.SDFSRESL'
   Data set name #2  'IMSV910.IF52.USERLIB'
   Data set name #3
   Data set name #4
   Data set name #5
   Data set name #6

IMS Macros:
   Data set name . . 'IMS.V910.SDFSMAC'

Command ===> ____________________________________________________________
```

Enter **RESET** on the command line and press Enter.

Check that the data set names displayed for the subsystem are the data set names you specified in the ZDT/IMS installation options module.

Press **PF3** to save the data set names in your profile and exit the panel. The Subsystem Settings Menu (Figure 58: Subsystem Settings Menu on page 317) is redisplayed

Enter **3** on the command line, for Data Sets 2, and press Enter. The DLI Mode Data Sets 2 is displayed.

Figure 62. DLI Mode Data Sets 2

```
 Process   Options   Help
--------------------------------------------------------------------------------
ZDT/IMS                        DLI Mode Data Sets 2

Subsystem IF52  IF52 - DYNAMIC, all unprotect

RECON:
   Primary data set  'IMSV910.IF52.RECON1'
   Secondary . . . . 'IMSV910.IF52.RECON2'
   Spare . . . . . . 'IMSV910.IF52.RECON3'

ACBLIB:
   Data set name . . 'IMSV910.IF52.ACBLIB'

Command ===> ____________________________________________________________________
```

Enter **RESET** on the command line and press Enter.

Check that the RECON data set name fields are blank. Also check that the ACBLIB data set name displayed for the subsystem is the data set name you specified in the ZDT/IMS installation options module.

Press **PF3** to save the settings in your profile and exit the panel. The Subsystem Settings Menu is again displayed.

Step 8. Verify DLI mode options

Enter **4** on the command line, for Options, and press Enter. The DLI Mode Options panel is displayed.

Figure 63. DLI Mode Options

```
 Process   Options   Help
--------------------------------------------------------------------------------
ZDT/IMS                           DLI Mode Options

Subsystem IF52  IF52 - DYNAMIC, all unprotect

Options:
   Enter "/" to select option
   _  Autosave

Checkpoint Frequencies:
   Edit . . . . . . . . . 5
   Change All/Repeat All  55
   Load . . . . . . . . . 155
   Batch Edit . . . . . . 255

PSB Processing Options:
   Browse         Extract        Print          Batch Browse
   1  1. G        1  1. G        2  1. G        2  1. G
      2. GO          2. GO          2. GO          2. GO


Command ===> ____________________________________________________________________
```

Enter **RESET** on the command line and press Enter.

Check that the settings displayed for the subsystem are the settings you specified in the ZDT/IMS installation options module.

Press **PF3** to save the settings in your profile and exit the panel. The Subsystem Settings Menu is again displayed.

Step 9. Verify DBD library

Enter **7** on the command line, for Data sets 1 PSB and DBD data set names, and press Enter. The PSB and DBD Data Sets panel is displayed.

Figure 64. PSB and DBD Data Sets panel

```
 Process   Options   Help
--------------------------------------------------------------------------------
ZDT/IMS                      PSB and DBD Data Sets

Subsystem IF52    IF52 - DYNAMIC, all unprotect

PSB:
   Data set name #1  'HFM.PSBLIB'
   Data set name #2
   Data set name #3
   Data set name #4
   Data set name #5
   Data set name #6

DBD:
   Data set name #1  'HFM.DBDLIB'
   Data set name #2
   Data set name #3
   Data set name #4
   Data set name #5
   Data set name #6

Command ===>
```

Press PF3 to save the settings in your profile and exit the panel. The Subsystem Settings Menu is again displayed.

Press PF3 again to return to the Settings Menu (Figure 57: Settings Menu on page 316).

Press **PF3** again to return to the ZDT/IMS Primary Option Menu (Figure 56: Primary Option Menu on page 315).

Step 10. Start the browse dialog

Enter **1** on the command line, for Browse data, and press Enter. The Browse Entry Panel (Dynamic PSB) is displayed.

Figure 65. Browse Entry Panel (Dynamic PSB)

```
 Process   Options   Help
--------------------------------------------------------------------------------
ZDT/IMS                        Browse Entry Panel

IMS:
   Subsystem name  . . IF52          PSB name  . . . ________  (If static PSB)
   Database name . . . ________      AGN name  . . . ________  (If BMP)
View:
   Data set name . . . ____________________________________________
   Member  . . . . . . ________

Processing Options:
   PSB type            Region type         Fetch DB dsnames from (if DLI)
   1  1. Dynamic       1  1. DLI           1  1. User profile
      2. Static           2. BMP              2. DFSMDA members

   View usage                              Enter "/" to select option
   3  1. New                               _  Secondary index (if dynamic PSB)
      2. Existing                          _  Skip DB data set panel (if DLI)
      3. None

Command ===> ____________________________________________________________
```

The subsystem name you selected in Step 5. is displayed in the IMS™ **Subsystem name** field.

Enter **DJ1E** in the **Database name** field.

Enter **1** in the **PSB type** field, **1** in the **Region type** field, **2** in the **Fetch DB dsnames from (if DLI)** field, a space in the **Secondary index (if dynamic PSB)** field, **3**in the **View usage** field, and press Enter.

Either the Browse : Database Data Set Specification panel or the Browse : Database Data Set Display panel is displayed. (The Browse Database Data Set Display panel is displayed if the HFM1POPI macro statement for the subsystem specifies DYNALLOC=Y. Otherwise, the Browse Database Data Set Specification panel is displayed.)

Figure 66: Browse : Database Data Set Specification on page 321 shows the Browse : Database Data Set Specification panel. Figure 67: Browse : Database Data Set Display on page 322 shows the Browse : Database Data Set Display panel.

Step 11. Verify database data sets

Figure 66. Browse : Database Data Set Specification

```
 Process   Options   Help
--------------------------------------------------------------------------------
ZDT/IMS             Browse : Database Data Set Specification

Subsystem IF52  Database DJ1E

DBD name  DD name   Data set name
DJ1E      DJ1E      'HFMIMS.DB.DJ1E'
DJ2E      DJ2E      'HFMIMS.DB.DJ2E'
DJ2F      DJ2F      'HFMIMS.DB.DJ2F'
DJ3E      DJ3E      'HFMIMS.DB.DJ3E'
DJ3F      DJ3F      'HFMIMS.DB.DJ3F'
DJ1F      DJ1F      'HFMIMS.DB.DJ1F'
****  End of data  ****


Processing Options:
   Fetch dsnames from            Enter "/" to select option
   1  1. Profile                 _  Save dsnames in profile
      2. Dynamic Allocation data set


Command ===> ____________________________________________________________
```

Figure 67. Browse : Database Data Set Display

```
 Process   Options   Help
--------------------------------------------------------------------------------
                    Browse : Database Data Set Display

Subsystem IF32  Database DJ1E

DBD name  DD name   Data set name
DJ1E      DJ1E      'HFMIMS.XTEST.DB.DJ1E'
DJ2E      DJ2E      'HFMIMS.XTEST.DB.DJ2E'
DJ2F      DJ2F      'HFMIMS.XTEST.DB.DJ2F'
DJ3E      DJ3E      'HFMIMS.XTEST.DB.DJ3E'
DJ3F      DJ3F      'HFMIMS.XTEST.DB.DJ3F'
DJ1F      DJ1F      'HFMIMS.XTEST.DB.DJ1F'
****  End of data  ****



Command ===> ____________________________________________________________
```

Check that the data set names displayed on the panels are the names given to the database data sets in the HFM1IVP job. If necessary, change the data set names to those shown in Figure 66: Browse : Database Data Set Specification on page 321 and Figure 67: Browse : Database Data Set Display on page 322. Then press Enter. The Browse: Database Positioning panel is displayed.

Step 12. Database positioning

Figure 68. Browse : Database Positioning

```
 Process   Options   Help
--------------------------------------------------------------------------------
ZDT/IMS                  Browse : Database Positioning

Subsystem IF52  Database DJ1E      Key sequence                    Format CHAR
View      None                                Key
Cmd SXE Level     Segment   Description     len  Key value
___ SX  1         SHIRE                      20  BROOME
___  X   2        SHIRENP                     0
___  X   2        LINKSUB                    18  ..................
   ****  End of data  ****


Command ===> ____________________________________________________________
```

Enter an **S** in the **Cmd** field for the SHIRE segment and press Enter. The Browse : IMS™ Database panel is displayed.

Step 13. Browse the database

Figure 69. Browse : IMS™ Database

```
 Process   Options   Help
──────────────────────────────────────────────────────────────────────────────<
 ZDT/IMS                  Browse : IMS Database DJ1E
                                                 Scope DB  Col 1       Format CHAR
 Cmd Level Segment   ----+----1----+----2----+----3----+----4----+----5---
                     ****   Top of window   ****
 ___ 1     SHIRE     BROOME              100100998.......p.......k......f.
 ___  2    SHIRENP   ..U........
 ___  2    LINKSUB   BROOME
 ___  2    LINKSUB   BROOME SHIRE
 ___ 1     SHIRE     DENMARK             130500244.h.....g/..............
 ___  2    SHIRENP   ...........
 ___  2    LINKSUB   BOW BRIDGE
 ___  2    LINKSUB   DENMARK
 ___  2    LINKSUB   DENMARK SHIRE
 ___  2    LINKSUB   GOLDEN HILL
 ___  2    LINKSUB   HAZELVALE
 ___  2    LINKSUB   KENT RIVER
 ___  2    LINKSUB   KENTDALE
 ___  2    LINKSUB   KENTON
 ___  2    LINKSUB   KORDABUP
 ___  2    LINKSUB   MOUNT SHADFORTH
 ___  2    LINKSUB   PARRY INLET

Command ===> ____________________________________________________________
```

This concludes the verification process.

Enter **=x** on the command line to return you to the z/OS® Primary Option Menu. Alternatively, press PF3 repeatedly until you return to this panel. In the process, the message:

```
DFS627I IMS RTM CLEANUP ( EOT ) COMPLETE FOR ST PERTHWA .ISPFPROC.        ,RC=00
```

is displayed. Ensure that RC=00 is displayed. This message verifies that the connection between IMS™ and your ISPF session has terminated normally.

You can now complete the installation of ZDT/IMS by performing the ACCEPT processing. The steps involved are described in the Z Data Tools Program Directory.

Part IV. Customizing Z Data Tools CICS® Component

Chapter 28. Preparing to access CICS resources from Z Data Tools

Z Data Tools can access files, transient data queues, or temporary storage queues defined to and owned by a CICS® subsystem.

Installation requirements for accessing CICS resources from Z Data Tools depend on the way in which Z Data Tools is being invoked. Z Data Tools can use ZDT/CICS to access CICS resources from the following environments:

- CICS via the ZCC server
- CICS via a batch job
- TSO/ISPF
- Batch JCL

CICS via the ZCC server

HFM3POPT specifies the option **START=TASK**, and ZDT/CICS is run from a CICS terminal via the CICS transaction defined in hlq.SHFMSAM1(HFMCINST). The CICS transaction communicates with the common server via the **PORT** and, optionally, the **HOST** address specified in HFM3POPT to spawn an address space to run Z Data Tools that communicates with the CICS transaction via TCP/IP. The spawned address space runs until the ZDT/CICS session is terminated.

The ZDT/CICS user interface is an ISPF-like interface which uses the ZCC feature called Interactive Panel Viewer (IPV) to display panels under CICS.

See Checklist for installing and customizing Z Data Tools to access CICS resources on page 326 for the tasks that need to be completed.

CICS via a batch job

HFM3POPT specifies the option **START=BATCH**, and ZDT/CICS is run from a CICS terminal via the CICS transaction defined in hlq.SHFMSAM1(HFMCINST). The CICS transaction submits a batch job that executes a procedure based upon sample member hlq.SHFMSAM1(HFM3CICB) that runs Z Data Tools to communicate with the CICS transaction via TCP/IP. The batch job runs until the ZDT/CICS session is terminated.

The ZDT/CICS user interface is an ISPF-like interface which uses the ZCC feature called Interactive Panel Viewer (IPV) to display panels under CICS.

See Checklist for installing and customizing Z Data Tools to access CICS resources on page 326 for the tasks that need to be completed.

TSO/ISPF

You can access CICS resources from a Z Data Tools ISPF invocation or keyword from REXX under ISPF. ISPF communicates directly with the CICS region to access its resources.

See Checklist for installing and customizing Z Data Tools to access CICS resources on page 326 for the tasks that need to be completed.

Batch JCL

You can access CICS resources by executing Z Data Tools from batch JCL.

See Checklist for installing and customizing Z Data Tools to access CICS resources on page 326 for the tasks that need to be completed.

There are a number of installation steps you need to perform to access CICS resources from Z Data Tools. Some of the steps are common regardless of the environment from which Z Data Tools is invoked, and others are specific to the environment. See Checklist for installing and customizing Z Data Tools to access CICS resources on page 326.

Before you can install and customize Z Data Tools to access CICS resources, you must have installed the Z Data Tools Base function. These topics assume you have installed Z Data Tools CICS component into the same target and distribution libraries as Z Data Tools Base function, with one exception: Z Data Tools CICS load modules are installed by default into HFM.SHFMMOD2, whereas Z Data Tools Base function and the other Z Data Tools component load modules are installed into HFM.SHFMMOD1.

Checklist for installing and customizing Z Data Tools to access CICS resources

Z Data Tools can use ZDT/CICS to access CICS resources from different environments. The following checklist defines the required installation and customization tasks for each environment.

For more information about these environments, see Preparing to access CICS resources from Z Data Tools on page 325.

The following table outlines for each environment from which Z Data Tools can be invoked, the installation steps required so that Z Data Tools can use ZDT/CICS to access CICS resources.

The preferred environment to access CICS resources using Z Data Tools is via the TSO/ISPF environment. For installations using CICS via the ZCC server environment, HFM3PRFD should specify ***PASSWORD=PASSTICKET**. For further information, see Modifying and submitting HFM3INST and HFM3PRFD on page 337.

Table 59. Summary of installation steps required based on the environment

Step	Description	CICS via the ZCC server	CICS via a batch job	TSO/ISPF	Batch JCL
__ 1	Update your CICS® startup procedures - **required**. See Updating the CICS startup procedures on page 330.	✓	✓	✓	✓

Table 59. Summary of installation steps required based on the environment (continued)

Step	Description	CICS via the ZCC server	CICS via a batch job	TSO/ISPF	Batch JCL
__ 2	Customize TCP/IP, if necessary. See Customizing for CICS TCP/IP on page 330.	✓	✓	✓	✓
__ 3	Provide OMVS segments for the CICS® region user ID and other users - **required**. See Providing OMVS segments for ZDT/CICS users on page 332.	✓	✓		
__ 4	Customize ZCC server. See Customizing the ZCC server on page 381.	✓			
__ 5	Customize to run Base ZDT, ZDT/IMS and ZDT/Db2 through ZDT/CICS, if necessary. See Accessing other Z Data Tools functions from the primary option menu on page 332.	✓	✓		
__ 6	Customize for ZDT/CICS on interconnected CICS® regions. See Providing CICS resource definitions for ZDT/CICS on interconnected regions on page 332.	✓	✓	✓	✓

Table 59. Summary of installation steps required based on the environment (continued)

Step	Description	CICS via the ZCC server	CICS via a batch job	TSO/ISPF	Batch JCL
__ 7	Set up for External CICS Interface (EXCI) access. See Setting up the External CICS interface (EXCI) access on page 333			✓	✓
__ 8	Customize the ZDT/CICS batch procedure. See Customizing the batch procedure on page 336.		✓		
__ 9	Modify and run the installation job, HFMCINST. See Modifying and submitting HFMCINST on page 337.	✓	✓	✓	✓
__ 10	Modify and run the installation job, HFM3INST - **required**. See Modifying and submitting HFM3INST and HFM3PRFD on page 337.	✓	✓		
__ 11	Change the default options. See Changing the default options on page 341.	✓	✓		
__ 12	Customize the ZDT/CICS security environment. See Customizing the ZDT/CICS security	✓	✓		

Table 59. Summary of installation steps required based on the environment (continued)

Step	Description	CICS via the ZCC server	CICS via a batch job	TSO/ISPF	Batch JCL
	environment on page 345.				
__ 13	Customize CICS® security for ZDT/CICS. See CICS security and ZDT/CICS on page 347.	✓	✓	✓	✓
__ 14	Determine how to customize the ZDT/CICS audit facility. See Alternatives for controlling Z Data Tools CICS auditing on page 352.	✓	✓		
__ 15	Customize ZDT/CICS for national languages. See Customizing ZDT/CICS for national languages on page 368.	✓	✓		
__16	Verify the installation and customization of ZDT/CICS. See Verifying the customization of ZDT/CICS on page 373	✓	✓		

Chapter 29. Customizing the operating environment for ZDT/CICS

This chapter describes how to customize the operating environment for ZDT/CICS.

CICS® resource definitions for ZDT/CICS

For required definitions for all environments, see Modifying and submitting HFMCINST on page 337.

For definitions required to run Z Data Tools as a CICS transaction, see Modifying and submitting HFM3INST and HFM3PRFD on page 337.

Updating the CICS® startup procedures

You need to make the following changes to your CICS® startup procedures, before you can use ZDT/CICS.

- Add the group HFMCICS to a group list that is installed at CICS® startup.
- Add the ZDT/CICS load library to the DFHRPL DD in your CICS® startup procedure. This will be HFM.SHFMMOD2 if you installed ZDT/CICS into the default libraries.

  If you have installed the ZDT/CICS Japanese component, add the ZDT/CICS Japanese load library to the DFHRPL DD in front of HFM.SHFMMOD2. The default Japanese load library is HFM.SHFMMODJ.
- Add the following statement for the definition of the internal reader for job submission, to your startup procedure:

  ```
  //HFMRDR   DD   SYSOUT=(,INTRDR)
  ```

  If you change the name of the HFMRDR resource on the DEFINE TDQUEUE(HFMJ) statement in HFM3INST, then modify this statement in your startup procedure accordingly. See Modifying and submitting HFM3INST and HFM3PRFD on page 337 for information about HFM3INST.

  You might also have to modify your CICS® startup procedures for TCP/IP customization. See Customizing for CICS TCP/IP on page 330 for more information.

You should also ensure that your CICS® system is set up to run with SEC=YES. Without this option the ZDT/CICS logon panel will not be able to verify or change passwords. For information about ZDT/CICS security, see CICS security and ZDT/CICS on page 347.

When you have done all this, restart the CICS® region.

Customizing for CICS® TCP/IP

ZDT/CICS communicates with the CICS® transaction via TCP/IP. Therefore you must set up and configure CICS® TCP/IP if you have not already done so for another application. The following is an outline of the steps required to do this. For detailed information, see the *IP CICS Sockets Guide*.

1. Modify the startup JCL for the CICS® region to include the DD statements required for TCP/IP. The required DD statements are listed in "Modifying CICS® startup" in the *IP CICS Sockets Guide*.
2. Define the required TCP/IP resources to CICS®. Refer to the member EZACICCT in the SEZAINST TCP/IP target library, or the RDO definitions listed in "Defining CICS® TCP/IP Resources" in the *IP CICS Sockets Guide* for a complete list of all required resource definitions.
3. Optionally add the IP CICS® entries to the PLT to automatically startup or shutdown the CICS® Sockets Interface during CICS® startup and shutdown. See "CICS® Program List Table (PLT)" in the *IP CICS Sockets Guide* for a description of the IP CICS® PLT entries.

   Note: If the IP CICS® entries are not added to the PLT, then the CICS® Sockets interface must be started manually through the EZAO,START,CICS transaction after every time the CICS® region is restarted.

4. Record the value of the TCPIPJOBNAME parameter specified in the TCPIP.DATA data set. This is the name of the started procedure used to start the TCP/IP Services address space and is required for the next step.
5. Create the TCP/IP configuration file, and then configure it using the EZACICD macro. A complete description and sample JCL for creating and building the configuration file can be found in "Configuring the CICS® TCP/IP Environment" in the IP CICS® Sockets Guide.
6. Restart the CICS® region.
7. If you did not add the IP CICS® entries to the PLT, start the CICS® sockets interface by issuing the EZAO,START,CICS transaction (if available).

IPv6 support

The ZDT/CICS transaction communicates with either the ZCC server, or a batch job via TCP/IP sockets and supports both IPv4 and IPv6 networks.

If TCP/IP on the system where the CICS® region is running supports Internet Protocol version 6 (IPv6), the following additional requirements exist:

- TCP/IP and the resolver on the system where the CICS® region is running must be properly configured so ZDT/CICS is able to obtain the correct host name of the system.

  You can test your TCP/IP configuration by running the HFMCHKH REXX exec member in HFM.SHFMSAM1.

- TCP/IP and the resolver on the system where the Z Data Tools batch job is submitted must then be able to translate the host name that was obtained and passed by the CICS® transaction into an appropriate IP address for the system where the CICS® region is running.

Note: The requirements listed above apply for IPv6 networks even if the CICS® region and Z Data Tools batch job run on the same system.

See *z/OS Communications Server: IP Configuration Guide* for further details.

Providing OMVS segments for ZDT/CICS users

The CICS® region and each individual ZDT/CICS user must have a user ID defined to z/OS® with an OMVS segment. This allows usage of TCP/IP by ZDT/CICS.

If an OMVS segment is not defined, message HFMCA021 may be issued with the `INITAPI` function when a user attempts to run the ZDT/CICS transaction.

Accessing other Z Data Tools functions from the primary option menu

You can allow users to invoke Z Data Tools Base function, or ZDT/Db2 or ZDT/IMS (if installed), from the ZDT/CICS primary option menu.

To do this you customize your Z Data Tools Base function security, using RACF® (or your equivalent security product). For more information refer to Customizing the ZDT/CICS security environment on page 345.

If you plan to access ZDT/Db2 from ZDT/CICS, then ensure that the SYSPROC DD statement in HFM3CICB references the Z Data Tools CLIST library, SHFMCLIB. You should also ensure that the user's ZDT/CICS startup JCL specifies or defaults to a region size of at least 8M. For information about modifying HFM3CICB, see Customizing the batch procedure on page 336.

Providing CICS® resource definitions for ZDT/CICS on interconnected regions

ZDT/CICS can be used to edit, list, and so on, CICS® resources that are owned by remote CICS® regions. An active connection must be available on the region where the ZDT/CICS interface is running, to the resource-owning CICS® region. In addition, on each of the remote resource-owning CICS® regions, you must:

- Add the ZDT/CICS load library to the DFHRPL DD statement in the CICS® startup procedure. This will be HFM.SHFMMOD2 if you installed ZDT/CICS into the default libraries.

  If you have installed the ZDT/CICS Japanese component, add the ZDT/CICS Japanese load library to the DFHRPL DD in front of HFM.SHFMMOD2. The default Japanese load library is HFM.SHFMMODJ.
- Define and install the required ZDT/CICS load modules, HFM3CICS and HFMLVL, in the CSD group, HFMCICS. You can do this using the sample job HFM3INSR. Modify and run this job against the CSD of the interconnected regions to define the required ZDT/CICS resources on the remote regions. You do this as follows:
  1. Copy the member HFM3INSR from HFM.SHFMSAM1 into your own JCL library.
  2. Modify HFM3INSR in your library.
     - Change the job card to meet your site's requirements.
     - Change *#cicshlq* to the high level qualifier for your CICS® libraries.
     - Change *#cee* to the high level qualifier for your Language Environment® libraries. Remove the reference if Language Environment® is available from LINKLIST.
     - Change *#csddsn* to the data set name of the DFHCSD file.

After you have done this, a resource owned by a remote CICS® region can be processed through the ZDT/CICS interface by specifying the SYSID of the remote CICS® region.

Note: The service level (PTF level) of the ZDT/CICS load modules must be the same (or higher) on all interconnected regions that you want the ZDT/CICS interface to link to, as on the local region.

This will automatically be true if you use the same ZDT/CICS load library for all your interconnected CICS® regions. If you do not use the same load library, you must ensure that all the load libraries have the same service level applied. If ZDT/CICS calls another region to process remote files, an error message is produced if the remote region is not at the required service level.

Limiting the regions to which ZDT/CICS can connect

By default, ZDT/CICS will attempt to link to each system that has an active connection defined on the region where the ZDT/CICS interface is running. You can limit the regions that ZDT/CICS attempts to link to by performing the following tasks:

1. Create a physical sequential data set (fixed, blocked, with 80 byte records) and enter the APPLID of each CICS® region that ZDT/CICS should link to in the first column of a new line. Sample member HFM3CONN is provided in HFM.SHFMSAM1 which you can copy into your own sequential data set and modify for your requirements. Refer to the sample for more information.
2. An active connection must exist and ZDT/CICS must be installed on each CICS® region entered in this data set.
3. Define the data set created in step 1 on page 333 as an extrapartition transient data queue (tdqueue) to the CICS® region where the ZDT/CICS interface is running. Refer to member HFM3INST in HFM.SHFMSAM1 library for a sample definition (tdqueue HFMC).
4. Specify the name of the extrapartition tdqueue defined in step 3 on page 333 on the CONN option of the ZDT/CICS options module, HFM3POPT. See Changing the default options on page 341 for information about the ZDT/CICS options module.
5. Re-install usermod HFM3UMDP, and refresh HFM3POPT in your CICS® region.

After performing these tasks, ZDT/CICS will only attempt to link to connected CICS® regions that:

- Have been listed in the transient data queue defined to the CONN option.
- Have acquired a valid connection on the CICS® region where ZDT/CICS is running.

Setting up the External CICS interface (EXCI) access

To access a CICS® resource from the Z Data Tools Base function via ISPF and batch, you must perform each of the steps shown here:

1. **Enable CICS® Interregion communications**

   Specify the system initialization parameter IRCSTRT=YES, or ensure IRC is started dynamically with an OPEN IRC command.

2. **Define connections to CICS®**

Copy member HFMCCONN from HFM.SHFMSAM1 into your JCL library and customize the JCL as described in the member. Ensure the group is defined and included in the start up list for the CICS® regions to be connected to.

Note: If you change the transaction name of HFMX, ensure you customize the Z Data Tools Base component and server options modules to specify the new transaction name via the EXCITRAN parameter (see Step 3 on page 334).

3. **Option module changes**

   If the transaction ID HFMX was changed in Step 2 on page 333, then HFM0POPT must be modified to specify the new transaction ID by means of the EXCITRAN keyword.

4. **EXCI Load library**

   The CICS® external interface library *cicshlq*.SDFHEXCI must be made available to TSO, the batch job, or the ZCC server, either in the linklist, or as a part of the STEPLIB in the TSO procedure, or batch JCL, or as a part of the HFMLIB in the configuration file used by the ZCC server startup procedure. If you are using a STEPLIB, then you should customize the batch job submission skeleton HFMFTEXC to include the library.

   Note: For an HFMELIBD invocation of Z Data Tools, the EXCI load library can be specified using the SDFHEXCI variable in the HFMELIBD exec. See Preparing Z Data Tools to run with LIBDEFs on page 31 for more details.

5. **HFMCICS DD**

   Copy member HFMCAPPL from HFM.SHFMSAM1 into your own PDS(E) data set that has the attributes of record format FB and logical record length 80. Edit this member to provide a list of CICS® regions that are accessible to Z Data Tools. This is used to support generic queries from the Z Data Tools ISPF interface.

   The layout of the member is:

   1 - 8

   CICS® VTAM® applid

   10 - 72

   CICS® region description

   All users must have read access to the data set and the HFMCICS DD should be defined to the TSO procedure and the ZCC server Z Data Tools configuration.

   Note: For an HFMELIBD invocation of Z Data Tools, the HFMCICS DD can be allocated by specifying the HFMCICS variable in the HFMELIBD exec. See Preparing Z Data Tools to run with LIBDEFs on page 31 for more details.

6. **Security**

All CICS® communications are performed under the security defined for the user running the Z Data Tools function. A user must have update access to the FACILITY class profile DFHAPPL.user ID to be authorized to use the EXCI Call interface. You must define the FACILITY class profiles shown here:

```
RDEFINE FACILITY (DFHAPPL.userid) UACC(NONE)
```

Where *userid* is either a generic name or a specific user name. We recommend you use an asterisk (*) and control access to the facility class using the permissions.

Users requiring CICS® access should be given UPDATE access to the FACILITY class as shown here:

```
PERMIT DFHAPPL.userid  CLASS(FACILITY) ID(USERID)   +
             ACCESS(UPDATE)
```

For more information about setting up the CICS® external interface, refer to the *CICS® Transaction Server for z/OS® CICS® External Interfaces Guide*.

Chapter 30. Customizing ZDT/CICS

This chapter describes how to customize ZDT/CICS.

Customizing the batch procedure

Customizing the batch procedure is only required if you have specified START=BATCH in the HFM3POPT module or you plan to use the START=BATCH invocation option on the HFM transaction.

The batch procedure, HFM3CICB, is required to run the ZDT/CICS batch job. You will need to customize HFM3CICB before you can use it. You do this as follows:

1. Copy the member HFM3CICB from HFM.SHFMSAM1 into your own procedure library.
2. Modify HFM3CICB in your library.
    - Change *#hfmhlq* to the high level qualifier for the Z Data Tools target libraries.
    - Change *#hfihlq* to the high level qualifier for the ZCC target libraries. ZCC is required for the operation of ZDT/CICS. Information about installing ZCC is included in the Z Data Tools Program Directory.
    - If the name of the TCP/IP address space is not the default value of 'TCPIP', uncomment the SYSTCPD DD statement and change *#tcpparms* to the data set and member name that contains the appropriate TCPIPJOBNAME parameter.
    - If you plan to access WebSphere® MQ queues from ZDT/CICS, uncomment the three MQ target library statements in the STEPLIB concatenation and change *#mqhlq* to the high level qualifier of your MQ libraries. The required MQ libraries are: SCSQAUTH, SCSQANLE, and SCSQLOAD.
    - If you plan to access ZDT/Db2 from ZDT/CICS and use the Db2® DSN command processor, uncomment the Db2® load library statement in the STEPLIB concatenation and change *#db2hlq* to the high level qualifier of your Db2® load library. The Db2® load library is: SDSNLOAD

      If you are using more than one version of Db2®, then specify the load library of the earliest version of Db2® you are using.

Notes:

1. The member name HFM3CICB is used on the **PROCNAME* statement in HFM3PRFD, which you customize for the job HFM3INST. If you change the name of the member here, you must also change the name specified on the **PROCNAME* statement to match when customizing HFM3INST. See Modifying and submitting HFM3INST and HFM3PRFD on page 337 for information about HFM3PRFD and HFM3INST.
2. For customization you can perform for ZCC, see Customizing the ZCC server on page 381.

Profile data sets

HFMPROF (*#hfm.profile.dsn*) is used to populate the ZDT/CICS logon panel.

The data set specified on the *PROFILE statement in HFM3PRFD and in the ZDT/CICS logon panel is the ZCC profile data set used to store information about the ZDT/CICS session for the user (similar to an ISPF profile data set).

Modifying and submitting HFMCINST

All of the CICS® definitions in this task are required for any of the ways Z Data Tools accesses CICS resources.

You modify HFMCINST as follows:

1. Copy the member HFMCINST from HFM.SHFMSAM1 into your own JCL library.
2. Change HFMCINST in your library as follows:
    a. Change the job card to meet your site's requirements.
    b. Change **CICSHLQ=** to the high-level qualifier for your CICS libraries.
    c. Change **LEHLQ=** to the high-level qualifier for your Language Environment® libraries. remove the reference if Language Environment is available from LINKLIST.
    d. Change **CS=** to the data set name of the DFHCSD file.

You can also change the CICS resource definition names listed in Table 61: ZDT/CICS resource definition names on page 340 in the DFHCSDUP step in HFMCINST. Note that changing some of these names also requires a change to the ZDT/CICS options in HFM3POPI. See HFM3POPI on page 342 for information about changing HFM3POPI.

If necessary, change **RESSEC** for the transactions HFM or HFMV to **RESSEC(YES)**, if you plan to use CICS security to protect resources accessed by these transactions.

Table 60. ZDT/CICS resource definition names

Resource name	Description	HFM3POPI change required?
HFM	Transaction - ZDT/CICS invocation	No
HFMV	Transaction - verify user ID	Yes
HFMM	Transient Data queue for ZDT/CICS message log.	Yes

Modifying and submitting HFM3INST and HFM3PRFD

You use the job HFM3INST (and HFM3PRFD) to provide the default job card and other default values that will be used to populate the logon panel the first time the ZDT/CICS transaction is run for any given user. After a user logs on for the first time the information will be saved in the profile data set for subsequent invocations.

HFM3INST will define the required CICS® table entries, define the ZDT/CICS profile data set (a VSAM KSDS), and initialize this profile data set with the logon defaults.

The CICS® resources defined by HFM3INST must be defined and installed locally on each CICS® region where the ZDT/CICS interface will run.

The VSAM data set HFMPROF (*#hfm.profile.dsn*) that is created and loaded by the last two steps of HFM3INST specifies SHAREOPTIONS(2). However, you can share this HFMPROF data set between multiple CICS® systems if you change this to SHAREOPTIONS(4). In this case, you need only run the delete/define and load of the VSAM data set once. An example of how this might be useful is if you use a common DFHCSD data set across more than one CICS® region.

Note: HFMPROF is used to populate the ZDT/CICS logon panel.

There are three steps to modifying and running HFM3INST:

- Modify HFM3PRFD
- Modify HFM3INST
- Run HFM3INST

HFM3PRFD

You modify HFM3PRFD as follows:

1. Copy the member HFM3PRFD from HFM.SHFMSAM1 into your own JCL library.
2. Change the following values in HFM3PRFD in your library, to customize the job card and other logon defaults:

   ***JOBCARD only applies to HFM3POPT START=BATCH**

   Defines the default job card values.

   You can use up to four lines to provide the job card values. There are two variables you can use:
   - **&TERM** is substituted with the 1 - 4 character terminal ID when the job is submitted.
   - **&USER** is substituted with the 1 - 7 character user ID required by ZDT/CICS when the job is submitted.

   Note:
   a. You can terminate the variable with a period which will be removed during substitution.
   b. You can use these variables to ensure the default job card is unique for each user.

   ***PROCNAME=HFM3CICB only applies to HFM3POPT START=BATCH**

   The procedure name is the name of the procedure to be used by the submitted batch job. The HCL-supplied value is HFM3CICB. If you change the name of the procedure then change this statement accordingly. (See Customizing the batch procedure on page 336 for information about HFM3CICB.)

   ***PROFILE=*profile_dataset_name* applies to both HFM3POPT START=BATCH and START=TASK**

   Specifies the default data set name for the user's Z Common Components profile data set. This value will be used to populate the profile data set name on the logon panel the first time a user invokes ZDT/CICS. The user can overwrite the system default on the logon panel. The submitted job will allocate the data set if it does not already exist. If it does exist it must be a partitioned data set with an LRECL of 80. The current user ID will be substituted for the variable &USER referenced in the name.

Note:

a. This profile data set is used to store information about the ZDT/CICS session for the user (similar to an ISPF profile data set).
b. It is recommended that you define this data set as a PDSE.
c. This is not the same data set as HFMPROF.

***USERID=SIGNON applies to both HFM3POPT START=BATCH and START=TASK**

ZDT/CICS requires a user ID to be provided when you run the transaction. If you specify *USERID=SIGNON then the user ID will default to the CICS® signed on user ID value when a user ID is not provided on transaction invocation. If you remove this statement and you invoke ZDT/CICS without a user ID then ZDT/CICS will prompt you for a user ID. If you specify *PASSWORD=REMEMBER or *PASSWORD=PASSTICKET you can remove this statement.

***PASSWORD=REMEMBER applies to both HFM3POPT START=BATCH and START=TASK**

ZDT/CICS initially displays a logon panel where you provide user ID and password and other logon information. Specify this statement if you want to bypass the logon panel for subsequent invocations of ZDT/CICS. ZDT/CICS will remember your password, and if you sign on to CICS® with the user ID and simply enter the transaction HFM, you will bypass the logon panel as long as the logon information recorded by ZDT/CICS is still valid. To invoke ZDT/CICS to get the logon panel enter:

```
HFMuserid
```

The logon panel will also appear if the ZDT/CICS job does not respond to the CICS® region for any reason.

***PASSWORD=PASSTICKET applies to HFM3POPT START=TASK**

Specify this parameter to direct ZDT/CICS to request a PassTicket for signed-on users. and, if the PassTicket is successful, bypass the logon panel. See *"Introduction to CICS security with RACF"* in the CICS® documentation for more information on generating and using PassTickets for secure sign-on.

Note: *PASSWORD=PASSTICKET is recommended over *PASSWORD=REMEMBER as it is the most secure way of bypassing the logon panel for signed-on users. *PASSWORD=REMEMBER will be deprecated in a future release.

HFM3INST

You modify HFM3INST as follows:

1. Copy the member HFM3INST from HFM.SHFMSAM1 into your own JCL library.
2. Change HFM3INST in your library as follows:
   a. Change the job card to meet your site's requirements.
   b. Change **CICSHLQ=** to the high-level qualifier for your CICS® libraries.

c. Change **LEHLQ=** to the high-level qualifier for your Language Environment® libraries. Remove the reference if Language Environment® is available from LINKLIST.
d. Change **CSD=** to the data set name of the DFHCSD file.
e. Change **HFMPROF=** to the data set name to be used for the VSAM logon profile data set for ZDT/CICS.
f. Change `HFMHLQ=` to the high-level qualifier for the Z Data Tools target libraries.
g. Change `HFMPROFD=` to refer to your customized version of HFM3PRFD.
h. If you are supplying ZDT/CICS with a list of APPLIDs of the connected CICS® regions where ZDT/CICS is installed, change *#hfm.conn.list* to the data set name that contains the list. A sample list is provided in HFM.SHFMSAM1(HFM3CONN).
i. Change the SHAREOPTIONS value in the second last step, or remove the last two steps completely, if you are sharing the HFMPROF profile data set. See SHAREOPTIONS on page 338 for more information.
j. You can also change the CICS® resource definition names, listed in Table 61: ZDT/CICS resource definition names on page 340, in the DFHCSDUP step in HFM3INST. Note that changing some of these names also requires a change to the ZDT/CICS options in HFM3POPI. See HFM3POPI on page 342 for information about changing HFM3POPI.

Table 61. ZDT/CICS resource definition names

Resource name	Description	HFM3POPI change required?
HFMPROF	File - logon profile data set	Yes
HFMJ	Transient Data queue name for job submission.	Yes
HFMRDR	DDNAME for transient data queue HFMJ.	No
HFMC	Transient data queue with a list of APPLIDs of connected CICS® regions.	Yes

3. When you have made all the required changes to HFM3INST, submit the job. It should finish with return code zero.

Modifying and submitting HFM3PRDU

If you need to update the parameters for a CICS® region after HFM3INST has been run to create the profile data set, use the job HFM3PRDU.

There are three steps to modifying and running HFM3PRDU:

- Modify HFM3PRFD (see HFM3PRFD on page 338)
- Modify HFM3PRDU
- Run HFMPRDU

HFM3PRDU

You modify HFM3PRDU as follows:

1. Copy the member HFM3PRDU from HFM.SHFMSAM1 into your own JCL library.
2. Change the following values in your copy of HFM3PRDU:
    - Change the job card to meet your site's requirements.
    - Change `HFMHLQ=` to the high level qualifier for the Z Data Tools target libraries.
    - Change `HFMPROF=profile.dsn` to the data set name to be used for the VSAM logon profile data set for ZDT/CICS.
    - Change `HFMPROFD=` to refer to your customized version of HFM3PRFD.

Changing the default options

Default processing options are supplied with ZDT/CICS in the module HFM3POPT. You can change these options to suit your installation requirements.

The first part of this module invokes the HFM0POPI macro and allows you to change options related to Z Data Tools processing (for example, audit logging and the print data set). See Z Data Tools options on page 383 for a description of the available Z Data Tools options and the values you can specify.

Any option changed in HFM3POPT will be used by the ZDT/CICS interface and when running the Z Data Tools Base component interface under ZDT/CICS.

Note:

1. If you have previously customized the Z Data Tools Base function, the options specified in the HFM0POPT options module (not HFM3POPT) are used when running Z Data Tools Base function under ISPF or batch. Therefore, if you want to run Z Data Tools Base component under ISPF and ZDT/CICS with similar installation requirements, you must ensure that the options specified in HFM0POPT match the options specified in HFM3POPT.

   For example, if you want Z Data Tools running through ISPF and ZDT/CICS to use the same audit log HLQ, ensure that the value specified for the AUDITHLQ option is identical in both HFM0POPT and HFM3POPT.

   If you do not plan to run Z Data Tools Base function under ISPF or batch, then only HFM3POPT needs to be configured.
2. When running ZDT/IMS through ISPF or through ZDT/CICS, all ZDT/IMS options are taken from HFM1POPT. Therefore you must ensure that the ZDT/IMS component is installed and configured to your requirements. See Customizing the ZDT/IMS installation options module on page 264 for information about changing options in HFM1POPT.

The second part of the HFM3POPT options module invokes the HFM3POPI macro and allows you to change options specific to the CICS® environment.

You use the usermod HFM3UMDP to install your version of HFM3POPT. HFM3UMDP is distributed in HFM.SHFMSAM1.

The options you can change in HFM3POPI are:

- The file name of the CICS® profile data set (PROF on page 342 option).
- The queue name of the transient data queue required for submission of CICS® batch jobs (QRDR on page 342 option).
- The transaction name for the user ID verification program (TRANVU on page 343 option).
- The queue name of the ZDT/CICS message log transient data queue (MSGL on page 343 option).
- The name of the transient data queue with a list of APPLIDs of connected CICS® regions (CONN on page 343 option).
- The ZDT/CICS start type (START on page 343 option).
- The transaction ID for HFM3CICS (TRAN on page 343 option).
- The port of ZCC server if START=TASK is specified (PORT on page 344 option).
- The host name of ZCC server (HOST on page 344 option).

You change the options as follows:

1. Copy the member HFM3POPT from HFM.SHFMSAM1 into your own source library.
2. Change the options in HFM3POPT in your library, according to your requirements. For a description of the options in HFM3POPT, and the values you can specify, see Z Data Tools options on page 383.
3. Modify HFM3POPI in the copy of HFM3POPT in your library, if required. See HFM3POPI on page 342 for information about HFM3POPI.
4. Modify the HFM3UMDP member in HFM.SHFMSAM1 to meet your site's requirements. Refer to the usermod for information about changes you might need to make.
5. Install SMP/E usermod HFM3UMDP.



Note: You can also use the sample job HFM3POPH to assemble HFM3POPT if you do not want to use SMP/E.

HFM3POPI

You use the options in HFM3POPI to override the ZDT/CICS defaults for the profile data set file name, the transient data queue name for job submission, and the transaction identifier for the user verification program. Refer to the description of each option for details.

PROF

```
[ { PROF=profile_name | PROF = HFMPROF } ]
```

PROF

Specifies the 1-8 character file name used to define the profile data set to CICS®. The default is HFMPROF.

QRDR

```
[ { QRDR=td_queue_name | QRDR = HFMJ } ]
```

QRDR

Specifies the 1-4 character transient data queue name that is used by ZDT/CICS for job submission.

TRANVU

```
[ { TRANVU=transaction_name | TRANVU = HFMV } ]
```

TRANVU

Specifies the 1-4 character transaction ID that is used to run the user verification program.

MSGL

```
[ { MSGL=td_queue_name | MSGL = HFMM } ]
```

MSGL

Specifies the 1-4 character transient data queue name that is used by ZDT/CICS as the message log.

CONN

```
CONN = [ td_queue_name ]
```

CONN

Specifies the 1-4 character transient data queue name that contains the list of APPLIDs of permitted CICS® regions for ZDT/CICS to access. This parameter is optional.

START

```
[ { START = BATCH | START = TASK } ]
```

START

Specifies how the ZDT/CICS address space is to be started.

TASK

The address space will be started by connecting to a running ZCC server task.

BATCH

A batch job will be submitted to start the address space.

Note: If you specify START=TASK then you must specify a value for PORT and, optionally, a value for HOST.

TRAN

```
[ { TRAN=tranid | TRAN = HFM } ]
```

TRAN

Specifies the 1-4 character transaction ID for HFM3CICS. The default is HFM.

PORT

```
[ { PORT=nnnnn | PORT=0 } ]
```

PORT

Specifies a 1 - 5 digit port number to be dedicated to the ZCC server task. ZDT/CICS will attempt to connect to the task using the specified port.

Note: If you specified TASK for the START option, then you must provide a value for PORT. If you specified START=BATCH then you will need to provide a value for PORT if your users will use the option (START=TASK) to invoke Z Data Tools. Otherwise, any value you specify for PORT is ignored.

HOST

```
HOST = [ host_name ]
```

HOST

Specifies the host name of the system where the ZCC server is running when START=TASK has also been specified. The maximum length of the name is 255 characters. The default, if not coded, is the domain name of the localhost.

Note: If you specified START=BATCH then any value you specify for HOST is ignored.

Chapter 31. Customizing the ZDT/CICS security environment

Security can be provided for Z Data Tools functions, including the functions available through the ZDT/CICS interface, using RACF® (or an equivalent security product). If you have not already configured your Z Data Tools Base function security, see Customizing the Z Data Tools security environment on page 65 for further information.

Customizing to protect update functions in ZDT/CICS

You can protect update functions in ZDT/CICS by means of the facility class FILEM.CICS.UPDATE, together with the SEC option in HFM3POPT.

If you set SEC=YES in HFM3POPT and then give or deny users access to the FILEM.CICS.UPDATE facility class, you can protect the following functions when running ZDT/CICS.

DSC

Data set copy

CDE

CICS® transient data edit

CFE

CICS® file edit

CTE

CICS® temporary storage edit

DSE

Data set edit

DSEB

Data set edit batch

DSG

Data set create

DSU

Data set update

FCH

Find/change - only protected if the function attempts the update command

The normal security checking of FILEM.FUNCTION.*function_code* applies to these functions when SEC=YES has been specified in HFM3POPT. If you do not specify SEC=YES, then no checking is done for these functions.

Customizing to invoke other Z Data Tools components from ZDT/CICS

You can allow users to invoke Z Data Tools Base function, or ZDT/Db2 or ZDT/IMS (if installed), from the ZDT/CICS primary option menu. To do this you customize your Z Data Tools Base function security, using RACF® (or your equivalent security

product), to give the user IDs running ZDT/CICS a minimum of read access to the facility groups, FILEM.CICS.BASE, FILEM.CICS.IMS or FILEM.CICS.DB2, as appropriate. For information about these facility groups, see Controlling access on page 66. For more information about ZDT/IMS security, see Customizing the ZDT/IMS security environment on page 282.

If you provide this access then the ZDT/CICS primary option menu will include the functions HFM (for Z Data Tools Base function), ZDT/IMS, and ZDT/Db2. These functions can then be invoked through the ZDT/CICS interface.

If you do not provide this access, then these functions will not appear on the menu, and will not be available.



Note: If you plan to run the Z Data Tools Base function interface through the ZDT/CICS session, you must first complete the installation and customization of Z Data Tools Base functionas described in Customizing Z Data Tools on page 22. If you plan to run the ZDT/IMS interface through the ZDT/CICS session, you must first complete the installation and customization of ZDT/IMS as described in Customizing Z Data Tools IMS Component on page 243. If you plan to run the ZDT/Db2 interface through the ZDT/CICS session, you must first complete the installation and customization of ZDT/Db2 as described in Customizing Z Data Tools Db2 Component on page 152.

Chapter 32. CICS® security and ZDT/CICS

Introduction

This chapter describes CICS® security as it applies to ZDT/CICS. For complete information about CICS® security, refer to the *CICS Transaction Server for z/OS RACF Security Guide* for your CICS® release.

To allow ZDT/CICS to process user IDs and passwords correctly, the CICS® system should be set up to run with SEC=YES. Without this option the ZDT/CICS logon panel will not be able to verify or change passwords.

This chapter provides an overview of CICS® security that applies to the functions performed by ZDT/CICS. It is intended to help you:

- Ensure proper security is implemented in your environment to prevent unauthorized access to CICS® resources or commands through ZDT/CICS.
- Determine why a NOTAUTH response was returned from a Z Data Tools/CICS function.

  Note that a resp2 value of 100 indicates a command security failure, whereas a resp2 value of 101 indicates a resource security failure.

Note:

1. CICS® Security is turned on for the region by specifying SEC=YES as a SIT parameter.
2. To use CICS® Security to protect resources accessed in a transaction, you must specify RESSEC(YES) for the transactions HFM or HFMV. You can do this in HFMCINST (see Modifying and submitting HFMCINST on page 337). The default value for RESSEC in HFMCINST for HFM and HFMV is RESSEC(NO).

CICS® security considerations for ZDT/CICS

To provide the necessary security for your CICS® regions, CICS® uses the z/OS® system authorization facility (SAF) to route authorization requests to an external security manager (ESM), such as RACF®, at appropriate points within CICS® transaction processing. ZDT/CICS does not use any undocumented or unsupported interfaces that bypass CICS® security. Therefore, CICS® security can be utilized to protect your system from unauthorized access to CICS® resources through ZDT/CICS.

When CICS® Security prevents successful execution of a Z Data Tools/CICS function, a NOTAUTH response is displayed. To ensure proper authorization to CICS® resources through ZDT/CICS, consider the following aspects of CICS® Security in your environment:

Logon and user security

CICS® security must be turned on in the region (SEC=YES) for ZDT/CICS to verify and change passwords from the ZDT/CICS logon panel.

Resource security

All files, temporary storage queues, and transient data queues can be protected through CICS® resource security. Appropriate authority must be assigned to the user ID for a user to process (edit and view) resources through ZDT/CICS.

Note the following exception to CICS® resource security for extrapartition transient data queues. When browsing an extrapartition transient data queue through ZDT/CICS, the batch HFM address space will read the records directly from the data set the queue is defined to. Therefore, the z/OS® user ID supplied to ZDT/CICS must have a minimum of READ authority to the appropriate RACF® data set profile.

Command security

CICS® command security allows CICS® security checking to be performed on system programming commands. ZDT/CICS issues the following commands that are subject to command security checking when listing and processing CICS® resources:

- INQUIRE
- SET

Therefore, the user ID must have sufficient authority to issue the commands listed above to successfully perform the ZDT/CICS function.

Note: A user ID must have authority to issue the CICS® INQUIRE command to run the ZDT/CICS interface.

Transaction security

CICS® transaction security prevents unauthorized users from running specific CICS® transactions. Therefore, a user ID must have appropriate authority to run the transaction that executes the ZDT/CICS program. Also, ZDT/CICS will link to other CICS® regions to process remote resources. This requires the user ID to have appropriate authority to run the mirror transaction that executes the ZDT/CICS program on the remote region.

Intercommunication security

When multiple CICS® regions are interconnected, ZDT/CICS can be used to access (list and edit) resources on multiple regions. To process remote resources, ZDT/CICS will function ship requests and link to the remote CICS® regions. Therefore, the user ID must pass the CICS® Link Security check and optional User Security check on the remote region to successfully complete the ZDT/CICS function.

Controlling ZDT/CICS processing

Z Data Tools has the ability to read, modify and change the status of CICS® resources. If the resources are not protected in the CICS® environment then there might be a requirement to control what functions Z Data Tools for CICS® users can perform.

If Security Server, RACF® 1.9 (or later) or an equivalent security product is active, the System Authorization Facility (SAF) with the Z Data Tools enhanced security facility is used for access control and authorization verification. Authorization is controlled by Z Data Tools-specific profiles in the FACILITY and XFACILIT class as follows.

Activating ZDT/CICS resource checking

The following facility class profile is used to determine whether Z Data Tools checks access for any given CICS® resource.

```
FILEM.CICS.RESOURCE
```

Here is an example of activating ZDT/CICS resource checking.

```
RDEF FACILITY FILEM.CICS.RESOURCE AUDIT(NONE)      +
        UACC(READ) OWNER(TYRONED)
SETROPTS RACLIST(FACILITY) REFRESH
```

If this profile has been defined and the user has an access of read or more then ZDT/CICS perform resource security checking using the XFACILIT class profiles described below.

Defining access to CICS® resources

Define XFACILIT class profiles in the form:

```
FILEM.sysplex_name.cics_applid.resource_type.resource_name
```

Where

sysplex_name

The z/OS® sysplex name.

cics_applid

The CICS® VTAM® application id for the CICS® region

resource_type

One of these values:

FILE

CICS® files

TD

CICS® transient data queues

TS

CICS® temporary storage queues

ENQ

CICS® enqueue resource name

resource_name

The CICS® file name, transient data queue name or temporary storage queue name. This level doesn't apply to the resource type ENQ.

Z Data Tools checks the level of access as follows to determine what functions can be performed.

READ

This allows read only functions like browse, print and view to run. The user is not allowed to modify a CICS® resource.

UPDATE

This allows update functions like edit, data create, copy to, and the ability to delete TS queues and empty TD queues from the resource list displays.

CONTROL

This allows CICS® SET function processing to change the status of a resource and the ability to purge tasks with outstanding enqueues for the XFACILIT class with resource_type ENQ. If the user does not have CONTROL access then the status fields that were modifiable on the resource list panels are protected for resources they are not allowed to modify.

Note: If the XFACILIT class for CICS® files has been defined and the user is performing a Z Data Tools function that can read or update the data set, then an additional check is performed to validate whether the user has the required level of access to the data set name associated with the CICS® file.

Examples for RACF® definitions

Case 1. Ensure all files on CICSDEV can only be accessed read

```
RDEF XFACILIT FILEM.SYSPLEXA.CICSDEV.FILE.**  AUDIT(NONE) +
              UACC(READ) OWNER(userid)
```

Case 2. Ensure all CICS® resources on CICSDEV can only be accessed read

```
RDEF XFACILIT FILEM.SYSPLEXA.CICSDEV.**  AUDIT(NONE)   +
              UACC(READ) OWNER(userid)
```

Case 3. Allow update against all CICS® resources on CICSDEV and allow SET processing to the systems programmer userid

```
RDEF XFACILIT FILEM.SYSPLEXA.CICSDEV.**  AUDIT(NONE)  +
              UACC(UPDATE) OWNER(userid)

PE FILEM.SYSPLEXA.CICSDEV.**  +
        CLASS(XFACILIT) ID(sysprog) ACC(CONTROL)
```

Case 4. Allow a specific user full access to FILE names beginning with FM

```
RDEF XFACILIT FILEM.SYSPLEXA.CICSDEV.FILE.FM* AUDIT(NONE) +
               UACC(NONE) OWNER(TYRONED)
PE FILEM.SYSPLEXA.CICSDEV.FILE.FM* +
         CLASS(XFACILIT) ID(user1) ACC(CONTROL)
```

Chapter 33. Customizing the Z Data Tools audit facility for CICS® component

You can optionally use Z Data Tools audit logging to create an audit trail of editing activity against resources processed through the ZDT/CICS interface. You can create your audit trail using either SMF or recording the activity to an audit log data set.

Alternatives for controlling Z Data Tools CICS® auditing

ZDT/CICS auditing is an optional facility. There is no requirement to implement it and ZDT/CICS works if auditing is not implemented. You should consider:

- Whether user access to data sets and other resources using Z Data Tools CICS® requires auditing.
- The information that Z Data Tools audit log records can provide.
- The information that Z Data Tools audit log records cannot provide, and possible alternatives to obtaining that information.
- If you do decide to use Z Data Tools auditing, how you will handle any issues associated with large audit log data sets, or additional SMF records.
- How you will use the information provided by Z Data Tools audit log records.

If your site requires a record of a user's read access to data sets, an external security product such as RACF® can be configured to log access by some or all users, and may be a better alternative.

CICS® also provides logging facilities that may be a better alternative. Z Data Tools audit of read access to data sets does not write audit log records for every record processed, rather the name of the data set and how many records were processed are written to the audit log.

Z Data Tools audit of changes to data sets typically writes two log records, a before and after image of the record that was changed. If you intend to log update changes to data sets that are subject to heavy update activity you need to consider the performance impact of writing many audit log records, also the size of any audit log data sets that may be produced.

You have two choices with respect to auditing of ZDT/CICS audit activities:

HFM3POPT controlled auditing

The facilities available with HFM3POPT controlled auditing are that you can specify auditing to the user's audit log data set, to the user's audit log data set with automatic (mandatory) printing of the audit log at the completion of the session, or to SMF.

SAF-rule controlled auditing

This relies on various SAF FACILITY and XFACILIT resource rules which you define with an external security product, such as RACF® (or equivalent product).

These points summarize the facilities available with SAF-rule controlled auditing:

- Auditing can be (optionally) specified for all ZDT/CICS functions.
- Different auditing requirements can be specified for different TSO user IDs.

- Different auditing requirements can be specified for access to different resources.
- You can provide ZDT/CICS users with a "Create audit trail" option for the ZDT/CICS edit functions. This is also SAF-rule controlled. The presence of the "Create audit trail" option does not guarantee that the user can switch off auditing, since this depends on the level of access the user has to the appropriate SAF resource names. When a user has access to the "Create audit trail" option, they can always turn on auditing, even if the relevant SAF resource rules do not require auditing.
- You can specify auditing to the user's audit log data set, to the user's audit log data set with automatic (mandatory) printing of the audit log at the completion of the session, or to SMF. Dual logging (to the user's audit log data set and to SMF) can also be specified.

Some other points to consider are:

- Auditing to the user's audit log data set can result in large numbers of audit log data sets. This may have disk space implications. You may need to consider implementing automatic purging or archiving of audit log data sets.
- Auditing to SMF (only) requires additional set-up, but provides a more reliable and secure environment for capturing audit information than audit logging to the user's audit log data set.
- If you implement SAF-rule controlled auditing you need to decide how Z Data Tools auditing will be enabled. This is described in more detail in Implementing SAF-rule controlled auditing on page 360. There are two alternatives. One requires an enabling SAF rule and the presence of a member in SYS1.PARMLIB. The other requires an enabling SAF rule but has no requirement for a member in SYS1.PARMLIB.

  The use of a member in SYS1.PARMLIB provides additional facilities compared with the alternative that does not require the use of SYS1.PARMLIB. The additional facilities are documented in Z Data Tools options specified in PARMLIB members on page 515.

When you have determined the appropriate type of auditing for your installation, follow the instructions in Customizing the Z Data Tools audit facility for CICS component on page 352.

Audit logging under ZDT/CICS and CICS® logging

ZDT/CICS uses standard CICS® APIs to process CICS® resources. Therefore, any established CICS® logging is performed when running ZDT/CICS. However, ZDT/CICS audit logging is completely independent of CICS® logging. Therefore, CICS® logging (including CICS® audit logging) can be used with ZDT/CICS. You could use CICS® audit logging as an alternative to using ZDT/CICS audit logging.

Z Data Tools provides two different methods for controlling whether audit records are written for Z Data Tools CICS® component. These are described in detail in Alternatives for controlling Z Data Tools CICS auditing on page 352.

You should determine which of the two methods is appropriate for your site's requirements.

Use the checklist to determine the customization required for the Z Data Tools CICS® audit facility.

Table 62. Checklist for audit customization, ZDT/CICS component. This table lists choices and decisions.

Audit customization choice	Decision (Yes|No|Not applicable)
1. Control auditing using the HFM3POPT options module	
2. Control auditing using SAF rules and a member in SYS1.PARMLIB	
3. Control auditing using SAF rules, without any changes to SYS1.PARMLIB	
4. Audit records are to be written to a data set	
5. Audit records are to be written to SMF	

For choices 1 to 3, you should answer YES for one choice only. Mark the other two choices as 'Not applicable'.

If you answer YES for choice 1, you can answer YES for one of choices 4 and 5. Mark the other choice as 'Not applicable'.

If you answer YES for choices 2 or 3, you can answer YES for one or both of choices 4 and 5. If you answer YES for both choices you are implementing dual-logging, which is only available with SAF-controlled auditing.

Once you have completed the checklist, use the table below to identify the customization that is required for each customization choice. Only complete the customization if your decision was *"YES"* in the checklist.

Table 63. Customization steps for audit customization choices. This table lists choices and related actions.

Customization choice	Sections to complete
1. Control auditing using the HFM3POPT options module	• HFM3POPT-controlled auditing on page 355
2. Control auditing using SAF rules and a member in SYS1.PARMLIB	• SAF-controlled auditing for Z Data Tools CICS component on page 357 • Implementing SAF-rule controlled auditing on page 360
3. Control auditing using SAF rules, without any changes to SYS1.PARMLIB	• SAF-controlled auditing for Z Data Tools CICS component on page 357 • Implementing SAF-rule controlled auditing on page 360
4. Audit records are to be written to a data set	• Audit data set configuration on page 355

Table 63. Customization steps for audit customization choices. This table lists choices and related actions.

(continued)

Customization choice	Sections to complete
5. Audit records are to be written to SMF	• Customizing Z Data Tools to write audit records to SMF on page 86

HFM3POPT-controlled auditing

To implement HFM3POPT-controlled auditing you need to set the AUDITLOG option in the HFM0POPI macro of the HFM3POPT module.

For information about the options see Z Data Tools options on page 383.

- If you want to produce an audit trail, specify AUDITLOG=YES in HFM3POPI macro.
- If you want to produce an audit trail and report on the changes made at the conclusion of an edit function, specify AUDITLOG=DEMAND. The job submitted will be determined by the skeleton member HFM.SHFMSLIB(HFM3FTAD). Customize the job card and JCL to specify the reporting options you require.

If audit logging is enabled, audit log records are produced for all edit activity against resources through the ZDT/CICS interface. This includes both CICS® resources and non-CICS data sets (if running the Z Data Tools Base function under ZDT/CICS).

Note:

- If you plan to enable audit logging for ZDT/CICS edit activity or for Z Data Tools Base function edit activity when running under the ZDT/CICS interface, then you must specify values for AUDITLOG and (if required) AUDITHLQ and SMFNO, in HFM3POPT.
- If you want to enable audit logging for the Z Data Tools Base function in batch and under ISPF, then you must specify values for AUDITLOG, AUDITHLQ and SMFNO in the Z Data Tools Base function options module, HFM0POPT.

Audit data set configuration

The format of the audit log data set name is determined by the setting of the AUDITHLQ parameter in the HFM0POPI definition in HFM3POPT. See AUDITHLQ on page 386 for more information about the AUDITHLQ option.

The following data set name formats may be generated:

- `userid.HFMLOG.Dyymmdd.Thhmmss` (when AUDITHLQ= (blank))
- *`audithlq`*`.userid.HFMLOG.Dyymmdd.Thhmmss` (when AUDITHLQ=audithlq)
- `qual1.<qual2.><qual3.>Dyymmdd.Thhmmss` (when AUDITHLQ=*qual1.<qual2.><qual3>*)

where:

audithlq

Any 1-8 character constant that is valid in the context of a data set name.

userid

The user ID creating the data set.

Dyymmdd

The date of the activity.

Thhmmss

The time of the activity.

When AUDITHLQ contains one or more periods, the AUDITHLQ value is treated as a data set prefix, with one, two or three levels. Each level of the prefix can be:

XXX

Any 1-8 character constant that is valid in the context of a data set name.

&&PREFIX

Indicates that the user's TSO prefix should by used. This is null if TSO NOPREFIX is in effect and, after substitution, the appropriate level of the audit log data set name prefix is also null.

&&USER

Indicates that the user's logonid (ISPF system variable ZUSER, stored in the shared pool) should be used.

&&UID

Indicates that the user's TSO prefix should be used, when the value is non-blank. When TSO NOPREFIX is in effect, the user's TSO logonid (ISPF system variable ZUSER, stored in the shared pool) should be used.

&&FUNCOD

Indicates that the Z Data Tools internal function code should be used. Specifying this parameter allows the Z Data Tools function that generated the audit log data set to be included in the audit log data set name.

Set the AUDITHLQ parameter in the HFM0POPI macro for the HFM3POPT to the required value, based on the above information and your site's requirements.

You can print the information in a Z Data Tools CICS® audit data set using the ZDT/CICS Print Audit Trail utility. To do this select option 3.6 from the ZDT/CICS Primary Option Menu.

Note:

- The format of the data set name for audit log data sets created under ZDT/CICS is the same as that for audit log data sets created by the Z Data Tools Base function. To distinguish between the two, use the AUDITHLQ option in HFM0POPT and HFM3POPT as appropriate.
- If you specify a value for AUDITHLQ in HFM3POPT, this value will apply to audit data sets for ZDT/CICS and for data sets for Z Data Tools Base function running under ZDT/CICS.

SAF-controlled auditing for Z Data Tools CICS® component

There are two methods for implementing SAF-controlled auditing for Z Data Tools CICS® component. These are:

1. Control auditing for Z Data Tools CICS® component using an enabling SAF Facility class rule and a member in SYS1.PARMLIB.

   To use this method complete the customization described in SAF-controlled auditing using SYS1.PARMLIB on page 357.

2. Control auditing for Z Data Tools CICS® component using an enabling SAF Facility class rule, without any changes to SYS1.PARMLIB.

   To use this method complete the customization described in SAF-controlled auditing without SYS1.PARMLIB on page 359.

Important: If you use a security product other than RACF®, review the information in When a security product other than RACF is in use on page 94 to avoid one possible cause of S047 abends.

SAF-controlled auditing using SYS1.PARMLIB

You need to define an enabling SAF facility profile as described below:

Define SAF facility profile

```
FILEM.PARMLIB.CICS
```

and ensure all ZDT/CICS users to be audited have at least read access to that facility. See the example below:

Example

User PROD1 to have SAF-rule controlled auditing using SYS1.PARMLIB.

Write this RACF® rule:

```
RDEF FACILITY FILEM.PARMLIB.CICS AUDIT(NONE) UACC(NONE) OWNER(ownerid)
PE FILEM.PARMLIB.CICS ACC(READ) ID(PROD1) CLASS(FACILITY)
```

Add member HFM3PARM to SYS1.PARMLIB (or any other library in the logical parmlib concatenation). See Defining the HFM3PARM member on page 358.

Once the above SAF rule is defined and activated, auditing for ZDT/CICS component users is controlled by the FMAUDIT parameter in the HFM3PARM member. See ZDT/CICS options specified in HFM3PARM on page 532 for more information. If audit log records are to be written to SMF, the SMF record number is specified as an FMAUDIT parameter option. See FMAUDIT on page 533, and SMF_NO on page 533.

Note: ZDT/CICS does not start if a user has read access to the above facility and the HFM3PARM member does not exist in the logical parmlib concatenation.

If SAF processing is not active, or the rule is not defined, or the rule is defined and the user has no access, then no parmlib processing is performed.

Defining the HFM3PARM member

If auditing is to be controlled from parmlib (user has read access to FILEM.PARMLIB.CICS, see SAF-controlled auditing for Z Data Tools CICS component on page 357), then member HFM3PARM must be defined in SYS1.PARMLIB (or any other library in the logical parmlib concatenation) as follows.

Default parmlib member HFM3PARM is provided in the SHFMSAM1 library. Copy this member to the appropriate system parmlib library. See below for details of methods that can be used to make this change.

Note: The sample HFM3PARM member supplied in SHFMSAM1 also includes a FMSECRTY statement. This option is not used at present, and can be either omitted, or commented out. It has no effect.

There are two methods that can be used to include the HFM3PARM member in a library in the logical parmlib concatenation. The choice of method depends on whether the installation's security software is configured to allow ZDT/CICS users READ access to the data set SYS1.PARMLIB.

Method 1 can only be used when ZDT/CICS users have read access to SYS1.PARMLIB.

Method 2 can be used regardless of whether ZDT/CICS users have READ access to SYS1.PARMLIB or not, and must be used when ZDT/CICS users do not have READ access to SYS1.PARMLIB.

Method 1

Place the HFM3PARM member in any library in the current logical parmlib concatenation. No IPL or other action is required to activate the new member unless a new library was added to the logical parmlib concatenation.

Notes:

1. Method 1 cannot be used in any situation where ZDT/CICS users do not have READ access to SYS1.PARMLIB. For example, when ZDT/CICS users have READ access to another library in the logical parmlib concatenation, and the HFM3PARM member is placed in the latter library. This will not work. The key issue is whether the ZDT/CICS user has READ access to SYS1.PARMLIB.
2. Using this method results in message IEE252I being written to the system log whenever a ZDT/CICS user accesses SYS1.PARMLIB. These messages cannot be suppressed. To avoid these messages use Method 2.

Method 2

This method must be used when ZDT/CICS users do not have READ access to SYS1.PARMLIB, or when suppression of the IEE252I messages is required.

1. Create a new library with dataset attributes similar to SYS1.PARMLIB.

   The library name for this data set must include the string "HFMPARM" in one of the qualifiers. You can choose any data set name that meets this requirement. Examples of suitable data set names are:

   SYS1.PARMLIB.HFMPARM
   SYS8.HFMPARM.PARMLIB
   HFMPARM.SYS8.PARMLIB
   SYS2.HFMPARMS.LIB
   SYS8.XHFMPARM.PARMLIB
2. Add member HFM3PARM to the new library, specifying the appropriate FMAUDIT parameter.
3. Add the new library to the logical parmlib concatenation. This can be done dynamically, or by means of a system IPL.

Note: When Method 2 is used, the HFM3PARM member must be located in the library created in step 1 on page 359. If the HFM3PARM member specifies any include statements (see Facilities for customizing the HFM3PARM definitions on page 534), all of the included members must also reside in the same library.

You use the HFM3PARM member to define:

- Whether ZDT/CICS uses SAF to control ZDT/CICS audit logging.
- The SAF resource name prefix to be used by ZDT/CICS when determining access to various resources.
- Whether ZDT/CICS loads the HFM3POPT module from a specific library.

For more information, see ZDT/CICS options specified in HFM3PARM on page 532.

SAF-controlled auditing without SYS1.PARMLIB

You need to define an enabling SAF facility profile as described below:

Define SAF facility profile

```
FILEM.SAFAUDIT.CICS
```

and ensure that all ZDT/CICS users to be audited have at least read access to that facility. See the example below.

Example

User PROD2 to have SAF-rule controlled auditing without using SYS1.PARMLIB.

Write this RACF® rule:

```
RDEF FACILITY FILEM.SAFAUDIT.CICS AUDIT(NONE) UACC(NONE) OWNER(ownerid)
PE FILEM.SAFAUDIT.CICS ACC(READ) ID(PROD2) CLASS(FACILITY)
```

If you use this method and intend to write audit records to SMF, the required SMF number is specified in the HFM3POPT module. See Customizing Z Data Tools to write audit records to SMF on page 86.

Implementing SAF-rule controlled auditing

Use the following checklist to implement SAF-rule controlled auditing:

1. Determine the SAF FACILITY and XFACILIT rules that will be required. See Understanding how ZDT/CICS uses SAF rules to control auditing on page 360 for more information.
2. Write the relevant SAF rules. See the examples in SAF rule examples on page 364.
3. Activate SAF auditing for a specific logon using the chosen method of activating SAF-controlled auditing. See SAF-controlled auditing for Z Data Tools Base component on page 91.
4. Using the selected logon, test the configuration to ensure that auditing is occurring as required.
5. When testing is complete, activate SAF-controlled auditing for all ZDT/CICS users.

Understanding how ZDT/CICS uses SAF rules to control auditing

SAF (System Authorization Facility) allows applications, such as ZDT/CICS, to define "resources" that might need to be protected. The "resource" to be protected need not be something specific, such as a data set; it can be essentially any type of resource or facility that the application considers to be important. For ZDT/CICS and auditing, the "resource" is the ability to write audit log records. The resource names reflect either the type of auditing that is to occur (eg to SMF), or the ZDT/CICS function and resource.

ZDT/CICS uses two types of SAF resource names to control auditing. The SAF resource rules used by ZDT/CICS to control auditing are shown in Table 67: ZDT/CICS auditing FACILITY class resource names on page 367 and Table 68: ZDT/CICS auditing XFACILIT class resource names on page 367).

Understanding SAF rule access levels

SAF provides for five levels of access to any FACILITY or XFACILIT resource. The levels of access form a hierarchy, so that a user with the highest level of access to a resource also has access to all the lower levels. The levels of access are specified in RACF® rules using the following mnemonics:

NONE

No access

READ

Level 1 access

UPDATE

Level 2 access

CONTROL

Level 3 access

ALTER

Level 4 access.

It is important to understand that the mnemonics used (READ, UPDATE and so on) can and do mean different things, depending on the context in which the SAF resource name is used. This can be confusing since READ and UPDATE have obvious meanings when it comes to, for example, accessing a data set. For SAF rules used to control ZDT/CICS audit, it may aid understanding to think of the mnemonics as indicating level 1 access and level 2 access.

For the SAF resource rules used by ZDT/CICS, the meanings of the various levels of access are:

NONE

The user does not have access to the resource; this typically means the user cannot write audit log records.

READ

The user has level 1 access to the resource; this typically means that the user can write audit log records.

UPDATE

The user has level 2 access to the resource. This level of access only has meaning for FACILITY rule 2 (see Table 67: ZDT/CICS auditing FACILITY class resource names on page 367). A user with level 2 access can write audit log records to the user's audit log data set, and the audit log data set will be printed at the end of the user's session (online execution only). This is equivalent to the DEMAND audit option in the non-SAF case.

CONTROL

The user has level 3 access to the resource. This level of access is not used by ZDT/CICS.

ALTER

The user has level 4 access to the resource. This level of access is not used by ZDT/CICS.

How ZDT/CICS determines whether audit log records should be written

The determination of whether audit records are to be written for a particular ZDT/CICS function and a given TSO logonid follows this three step process:

1. Step 1.
   - If auditing is being controlled by means of parmlib, the HFMAUDIT specification of the HFM3PARM member is used as follows.

     The FMAUDIT specification setting in the HFM3PARM member (in SYS1.PARMLIB or any other library in the logical parmlib concatenation) is the "main" switch for SAF-rule controlled auditing. Note that there are facilities available to specify different settings in the HFM3PARM member for different TSO logonids. See ZDT/CICS options specified in HFM3PARM on page 532 for more information. For any given TSO logonid, there are two possibilities:

     SAF_CTRL=NO

     SAF-rule controlled auditing is not in effect. Auditing is determined by the settings in the HFM3POPT module, see Customizing the Z Data Tools audit facility for CICS component on page 352.

     SAF_CTRL=YES

     SAF-rule controlled auditing is in effect. Processing continues to Step 2.
   - If auditing is being controlled using the method which does not access the parmlib concatenation, the TSO logonid has READ access to the DAF FACILITY rule FILEM.SAFAUDIT.CICS for processing to continue to Step 2 on page 362.
2. Step 2.

   Does the user have access to write audit records?

   This is determined by the user's access to rules 1 and 2 in Table 67: ZDT/CICS auditing FACILITY class resource names on page 367; the various outcomes are summarized in Table 64: Determination of a user's ability to write audit log records on page 362.

   Table 64. Determination of a user's ability to write audit log records

TODSN access[65]	TOSMF access[66]	OPTION access[67]	Can write audit records?	Demand logging?	"Create audit trail" option[68]
NONE	NONE	ANY	No	No	Not visible
READ	NONE	NONE	Yes, data set only	No	Not visible

65. Refers to the level of access the user has to SAF FACILITY rule 1 in Table 67: ZDT/CICS auditing FACILITY class resource names on page 367.
66. Refers to the level of access the user has to SAF FACILITY rule 2 in Table 67: ZDT/CICS auditing FACILITY class resource names on page 367.
67. Refers to the level of access the user has to SAF FACILITY rule 3 in Table 67: ZDT/CICS auditing FACILITY class resource names on page 367.
68. The visibility of the "Create audit trail" option does not influence whether a user can write audit log records, although the user must have access to write audit log records (to either a data set or SMF), for the option to be visible.

TODSN access[65]	TOSMF access[66]	OPTION access[67]	Can write audit records?	Demand logging?	"Create audit trail" option[68]
READ	NONE	READ	Yes, data set only	No	Visible
UPDATE	NONE	NONE	Yes, data set only	Yes	Not visible
UPDATE	NONE	READ	Yes, data set only	Yes	Visible
NONE	READ	NONE	Yes, SMF only	No	Not visible
NONE	READ	READ	Yes, SMF only	No	Visible
READ	READ	NONE	Yes, to data set and SMF	No	Not visible
READ	READ	READ	Yes, to data set and SMF	No	Visible
UPDATE	READ	NONE	Yes, to data set and SMF	Yes	Not visible
UPDATE	READ	READ	Yes, to data set and SMF	Yes	Visible

If the user does not have the ability to write audit log records, then no check of SAF resource names in Step 3 occurs.

A user's access to write audit log records at Step 2 only indicates that auditing *might* occur. The final decision depends on the user's level of access to the XFACILIT resource name (or names) that apply to the particular ZDT/CICS function.

3. Step 3.

   Does the user have access to write audit records for the current function and data set?

   The XFACILIT resource names used by ZDT/CICS to determine whether audit records should be written depend on the ZDT/CICS function being executed and the data set being accessed.

   Table 65: ZDT/CICS function codes that can be audited using SAF on page 363 shows the function codes which are supported.

Table 65. ZDT/CICS function codes that can be audited using SAF

Function code	Online option	Description
CSL	Delete prefix command	Delete queue

65. Refers to the level of access the user has to SAF FACILITY rule 1 in Table 67: ZDT/CICS auditing FACILITY class resource names on page 367.
66. Refers to the level of access the user has to SAF FACILITY rule 2 in Table 67: ZDT/CICS auditing FACILITY class resource names on page 367.
67. Refers to the level of access the user has to SAF FACILITY rule 3 in Table 67: ZDT/CICS auditing FACILITY class resource names on page 367.
68. The visibility of the "Create audit trail" option does not influence whether a user can write audit log records, although the user must have access to write audit log records (to either a data set or SMF), for the option to be visible.

Function code	Online option	Description
CTB	Browse prefix command	Browse temporary storage queue
CTE	2	Edit temporary storage queue
CTV	1	View temporary storage queue
CTP	3.2	Print temporary queue
CDB	Browse prefix command	Browse transient data queue
CDE	2	Edit transient data queue
CDV	1	View transient data queue
CDP	3.2	Print transient data queue
CFB	Browse prefix command	Browse file
CFE	2	Edit file
CFV	1	View file
CFP	3.2	Print file

Controlling where ZDT/CICS writes audit log records

You can use SAF to control whether ZDT/CICS writes audit log records to SMF, the user's audit log data set, or to both.

The following table shows the SAF FACILITY class resource names used to control ZDT/CICS logging to SMF and the user's audit log data set.

Table 66. SAF FACILITY class resource names controlling disposition of ZDT/CICS audit records

FACILITY Class Name	Purpose
FILEM.AUDIT3.TOSMF	Enables or disables auditing to SMF for ZDT/CICS.
FILEM.AUDIT3.TODSN	Enables or disables auditing to the user's data set for ZDT/CICS.

SAF rule examples

This section shows SAF rule examples under different conditions.

Controlling where ZDT/CICS writes audit log records

You can use SAF to control whether ZDT/CICS writes audit log records to SMF, the user's audit log data set, or to both.

Table 67: ZDT/CICS auditing FACILITY class resource names on page 367 shows the SAF FACILITY class resource names used to control ZDT/CICS to logging and the user's audit log data set.

Example 1

- Disable audit logging to a user data set for all ZDT/CICS users.
- Enable ZDT/CICS audit logging to SMF for the PROD logonid.

You could write the following RACF® rules:

```
RDEL FACILITY FILEM.AUDIT3.TOSMF[69]
RDEL FACILITY FILEM.AUDIT3.TODSN[69]
RDEF FACILITY FILEM.AUDIT3.TOSMF UACC(NONE) OWNER(XXXXXXX)[70]
RDEF FACILITY FILEM.AUDIT3.TODSN UACC(NONE) OWNER(XXXXXXX)[71]
PE FILEM.AUDIT3.TOSMF ACC(READ) ID(PROD) CLASS(FACILITY)[72]
```

Example 2

- Enable audit logging to a user data set for all ZDT/CICS users.
- Enable demand logging for the following users, PROD1, PROD2, PROD3.

You could write the following RACF® rules:

```
RDEL FACILITY FILEM.AUDIT3.TOSMF[69]
RDEL FACILITY FILEM.AUDIT3.TODSN[69]
RDEF FACILITY FILEM.AUDIT3.TOSMF UACC(NONE) OWNER(XXXXXXX)[70]
RDEF FACILITY FILEM.AUDIT3.TODSN UACC(READ) OWNER(XXXXXXX)[73]
PE FILEM.AUDIT3.TODSN ACC(UPDATE) ID(PROD1) CLASS(FACILITY)[74]
PE FILEM.AUDIT3.TODSN ACC(UPDATE) ID(PROD2) CLASS(FACILITY)[74]
PE FILEM.AUDIT3.TODSN ACC(UPDATE) ID(PROD3) CLASS(FACILITY)[74]
```

Example 3

- Disable audit logging completely for all ZDT/CICS users.
- Enable dual logging for all ZDT/CICS users.

You could write the following RACF® rules:

```
RDEL FACILITY FILEM.AUDIT3.TOSMF [69]
RDEL FACILITY FILEM.AUDIT3.TODSN [69]
RDEF FACILITY FILEM.AUDIT3.TOSMF UACC(READ) OWNER(XXXXXXX)[75]
RDEF FACILITY FILEM.AUDIT3.TODSN UACC(READ) OWNER(XXXXXXX)[76]
```

69. Delete any existing facility rule.
70. Define the facility rule for audit logging to SMF (TOSMF suffix). UACC(NONE) is used so that any user, for which there is no specific rule, has no access.
71. Define the facility rule for audit logging to the user's audit log data set (TODSN suffix). UACC(NONE) is used so that any user, for which there is no specific rule, has no access.
72. Define the facility rule for audit logging to the user's audit log data set (TODSN suffix). UACC(NONE) is used so that any user, for which there is no specific rule, has no access.
73. Define the facility rule for audit logging to the user's audit log data set (TODSN suffix). UACC(READ) is used so that any user, for which there is no specific rule, has read access, and can therefore write audit log records.
74. Allow logonids PROD1, PROD2, PROD3 to write audit log records with automatic printing of the audit report ("Demand logging") (ACC(UPDATE)), to SMF.
75. Define the facility rule for audit logging to SMF (TOSMF suffix). UACC(READ) is used so that any user, for which there is no specific rule, has access (and can therefore write audit records to SMF).
76. Define the facility rule for audit logging to the user's audit log data set (TODSN suffix). UACC(READ) is used so that any user, for which there is no specific rule, has access (and can therefore write audit records to the user's audit log data set).

Controlling auditing of ZDT/CICS functions

You can use SAF to control whether ZDT/CICS writes audit log records for functions which access resources. Table 65: ZDT/CICS function codes that can be audited using SAF on page 363 shows ZDT/CICS function codes which may be logged.

Example 1

- Enable audit logging of all modifications to temporary storage queue TS01 using the ZDT/CICS temporary storage queue edit function for all users except TSO logonid MAINT1.

You could write the following RACF® rules:

```
RDEL XFACILIT FILEM.AUDIT3.CICSAPLD.CTE.UPDATE.TS01[77]
RDEF XFACILIT FILEM.AUDIT3.CICSAPLD.CTE.UPDATE.TS01 OWNER(XXXXXXXX) UACC(READ)[78]
PE FILEM.AUDIT3.CTE.UPDATE.CICSAPLD.TS01 CLASS(XFACILIT) ID(MAINT1) ACC(NONE)[79]
```

Example 2

- Enable audit logging of all modifications together with all records read for transient data set TD01 using the ZDT/CICS transient data queue edit function (CDE) for user SERVIC1.

You could write the following RACF® rules:

```
RDEL XFACILIT FILEM.AUDIT3.CICSAPLD.CDE.ALL.TD01[77]
RDEF XFACILIT FILEM.AUDIT3.CICSAPLD.CDE.ALL.TD01 OWNER(XXXXXXXX) UACC(NONE)[80]
PE FILEM.AUDIT3.CICSAPLD.CDE.ALL.TD01 CLASS(XFACILIT) ID(SERVIC1) ACC(READ)[81]
```

Example 3

- Enable audit logging of functional information for data set HFM.CICS.KSDS using the ZDT/CICS file print utility for all users.

You could write the following RACF® rules:

```
RDEL XFACILIT FILEM.AUDIT3.CICSAPLD.CFP.FUNCTION.HFM.CICS.KSDS[77]
RDEF XFACILIT FILEM.AUDIT3.CICSAPLD.CFP.FUNCTION.HFM.CICS.KSDS
     OWNER(XXXXXXXX) UACC(READ)[82]
```

ZDT/CICS auditing FACILITY and XFACILIT class resource names

These two tables (and associated tables) list FACILITY and XFACILIT class resource names and details.

77. Delete any existing XFACILIT rule.
78. Define the XFACILIT rule to log all modifications to temporary storage queue TS01 using the ZDT/CICS temporary storage queue edit function (CTE). UACC(READ) allows all TSO user IDs to write audit log records (in the absence of any over-riding more specific rule).
79. A specific rule for logonid MAINT1 to prevent audit log records being written.
80. Define the XFACILIT rule to log all modification and all records read for transient data queue TD01 using the ZDT/CICS transient data queue edit function (CDE). Uacc(NONE) specified that no TSO user IDs write audit log records (in the absence of any over-riding more specific rule).
81. A specific rule for logonid SERVIC1 to write audit log records.
82. Define the XFACILIT rule to log function information when data set HFM.CICS.KSDS is printed using the ZDT/CICS file print utility (CFP). UACC(READ) allows all TSO user IDs to write audit log records (in the absence of any over-riding more specific rule).

Table 67. ZDT/CICS auditing FACILITY class resource names

Rule Number	Resource Name[83]	Purpose
1	FILEM.AUDIT3.TODSN	Allows a user to write audit log records to the user's audit log data set.
2	FILEM.AUDIT3.TOSMF	Allows a user to write audit log records to SMF.
3	FILEM.AUDIT3.OPTION	Allows the user access to the "Create audit trail" option on the ZDT/CICS Edit panels.

Table 68. ZDT/CICS auditing XFACILIT class resource names

Resource Name	Purpose
FILEM.AUDIT3.cicsapplid.functioncode.ALL[83].resource[84]	Allows users to write audit log records for all records which are read and modified for the specified data set (resource) using the ZDT/CICS function (function code).
FILEM.AUDIT3.cicsapplid.functioncode.UPDATE.resource	Allows users to write audit log records for all modifications to the specified data set (resource) using the ZDT/CICS function (function code).
FILEM.AUDIT3.cicsapplid.functioncode.FUNCTION.resource	Allows users to write audit log records containing information for the specified data set (resource) using the ZDT/CICS function (function code).

83. Use this option with caution. The size of the data set being accessed and the editing technique used will influence the number of read records logged, and this may affect the performance of ZDT/CICS.
84. If the resource is a file, the data set name and not the CICS® resource name must be specified. For temporary storage queues, transient data queues, and enqueues, the CICS® resource name must be specified.

Chapter 34. Customizing ZDT/CICS for national languages

You can customize ZDT/CICS for national languages other than English.

If you are using a language other than English, you will need to perform some or all of the customization tasks listed in Table 69: Summary of steps for customizing ZDT/CICS for a national language on page 368.

Unlike other Z Data Tools components, if you are using Japanese, you will have to perform the first task, *Setting the LANGUAGE option in HFM3POPT*, even if you have installed the ZDT/CICS Japanese component.

You will also have to do this if you plan to use the Z Data Tools Base function interface under ZDT/CICS.

Table 69. Summary of steps for customizing ZDT/CICS for a national language

Step	Description
__ 1	Set the LANGUAGE option in HFM3POPT. See Setting the LANGUAGE option on page 368.
__ 2	Verify that your CICS® terminal is defined as DBCS-capable, if necessary. See Verifying that the CICS terminal is DBCS-capable on page 369.
__ 3	Translate the ZDT/CICS logon messages to your language. See Translating the ZDT/CICS logon messages on page 369.
__ 4	Provide a version of HFM3MENU (ZDT/CICS logon messages) for your language. See Providing a multicultural version of HFM3MENU on page 370.
__ 5	Translate the ZDT/CICS panels to your language. See Translating the panel text on page 371.

Changing the print and display translation tables for languages other than English

If you plan to use ZDT/CICS with a national language other than English, you might need to provide a print and display translation table for your language.

You do this as part of your customization for the Z Data Tools Base function. See Changing the print and display translation tables for languages other than English on page 104.

You should also specify PRTTRANS=ON in HFM3POPT. If you are using any DBCS language you might also need to specify TERMTYPE=3270KN in HFM3POPT.

This step is essential if you are using a DBCS language other than Japanese. .

Setting the LANGUAGE option

ZDT/CICS uses the LANGUAGE option set in the ZDT/CICS options module, HFM3POPT. This setting is used for all ZDT/CICS messages and panels.

Therefore, if you are using a language other than English, you will need to set the LANGUAGE option in HFM3POPT to one of the values shown in Table 1. For example, to use French messages, specify LANGUAGE=FRENCH.

You will need to do this if you are using Japanese, even if you have installed the Z Data Tools and ZDT/CICS Japanese component.

See Changing the default options on page 55 and Changing the default options on page 341 for information on how to do this.

Table 70. Keyword values for the LANGUAGE option

Language	Code	Specify on LANGUAGE option...
French	FRA	FRENCH
German	DEU	GERMAN
Italian	ITA	ITALIAN
Japanese	JPN	JAPANESE
Portuguese	PTG	PORTUGUESE
Spanish	ESP	SPANISH
Danish	DAN	DANISH
Upper case English	ENP	UPPERENG
Korean	KOR	KOREAN
Swiss German	DES	SGERMAN
Traditional Chinese	CHT	CHINESET
Simplified Chinese	CHS	CHINESES
Other	XXX	OTHER

Verifying that the CICS® terminal is DBCS-capable

If you are using a DBCS language, verify that the CICS® terminals are defined as DBCS-capable. To do this, issue the following CICS® transaction and verify that the *SOSI* data area contains X'FF':

```
CECI ASSIGN
```

Translating the ZDT/CICS logon messages

The language used in ZDT/CICS logon messages is determined by the setting of the LANGUAGE option in HFM3POPT. All ZDT/CICS messages are stored in the HFM3MENU source member. This CSECT is part of the root ZDT/CICS module so that an English version of the messages is always available.

If you are using a language other than English or Japanese, you must provide a version of HFM3MENU for your language, as described in Providing a multicultural version of HFM3MENU on page 370.

Providing a multicultural version of HFM3MENU

HFM3MENU contains the assembler source for the ZDT/CICS messages. To provide translated versions of the ZDT/CICS logon messages:

1. Copy the member HFM3MENU from HFM.SHFMSAM1 to your own source library with the name HFM3M*yyy*, where *yyy* is one of the following language codes:

   FRA

   French

   DEU

   German

   ITA

   Italian

   JPN

   Japanese

   PTG

   Portuguese

   ESP

   Spanish

   DAN

   Danish

   ENP

   Upper case English

   KOR

   Korean

   DES

   Swiss German

   CHT

   Traditional Chinese

   CHS

   Simplified Chinese

XXX

Other

2. Change the message text in HFM3M*yyy* in your library.
3. Modify the HFM3UMDM member in HFM.SHFMSAM1 to meet your site's requirements, using the same language code as above. Refer to the usermod for information about other changes you might need to make.
4. Install SMP/E usermod HFM3UMDM.
5. Define and install program HFM3M*yyy* to your CICS® region, where *yyy* is the same language code as above. You can do this by updating and running the ZDT/CICS installation job HFM3INST. See Modifying and submitting HFM3INST and HFM3PRFD on page 337 for information about HFM3INST.

To use your translated messages, see Using the translated messages and panels on page 371.

Translating the panel text

All ZDT/CICS panels are provided in English.

They are also provided in Japanese if you have installed the ZDT/CICS Japanese component.

You can translate some or all of these panels into another language. (If no translated version of a particular panel is available, ZDT/CICS uses the English version.)

All ZDT/CICS panels are stored in HFM.SHFMPENU. You translate a panel as follows:

1. Find the panel members in HFM.SHFMPENU that you want to translate. The panel members specific to ZDT/CICS are all named HFM3*zzzz*.
2. Create a library with the same characteristics as HFM.SHFMPENU, with the name HFM.SHFMP*yyy*, where *yyy* is the same language code you specified when you modified HFM3MENU. If you have already created a library with this name for other translated Z Data Tools panels, use that library. Copy the required panel members from HFM.SHFMPENU to this library.
3. Change the required panel text in the members in your library. A panel may reference a **help** panel by means of a `.HELP` statement. If any panel you are changing contains any of these `.HELP` statements, also copy and change these referenced members in your library.

To use your translated panels, see Using the translated messages and panels on page 371.

Using the translated messages and panels

The language used in messages and panels displayed through the ZDT/CICS interface is determined by the setting of the LANGUAGE option defined in HFM3POPT. To use your translated messages and panels you must include your libraries in the message and panel concatenations in front of the Z Data Tools English libraries, in HFM3CICB.

For example, to use your translated messages, add HFM.SHFMM*yyy* to HFM3CICB, to the HFIPLIB DD statement, in front of HFM.SHFMMENU. To use your translated panels, add HFM.SHFMP*yyy* to HFM3CICB, to the HFIPLIB DD statement, in front of HFM.SHFMPENU.

See Customizing the batch procedure on page 336 for information about HFM3CICB.

Customizing for the Japanese national language

The other customization task you might need to do for the Japanese national language is to modify the supplied Japanese translation tables. If you want to do this, do so as part of the customization of the Z Data Tools Base function. See Modifying the Japanese translation tables on page 109.

Changing the Japanese logon message text

If you have installed the ZDT/CICS Japanese component, all ZDT/CICS Japanese logon messages are stored in the HFM3MJPN source member. Normally you should not need to modify this module. However, if you do want to modify it, you can do so by means of the usermod, HFM3UMDN.

To do this:

1. Copy the member HFM3MJPN from HFM.SHFMSAM1 to your own source library.
2. Change the message text in HFM3MJPN in your library.
3. Modify the HFM3UMDN member in HFM.SHFMSAM1 to meet your site's requirements. Refer to the usermod for information about other changes you might need to make.
4. Install SMP/E usermod HFM3UMDN.

Chapter 35. Verifying the customization of ZDT/CICS

After you have completed the initial installation and customization of ZDT/CICS, you can perform the following steps to verify your customization. You might need to refer to the Z Data Tools User's Guide and Reference for CICS.

You verify the installation and customization of ZDT/CICS by logging on to a CICS® system where you plan to run ZDT/CICS. Before you can do this, you must have customized ZDT/CICS, your CICS® system, and, if necessary, TCP/IP.

1. Log on to the CICS® system where you have implemented ZDT/CICS, and enter the ZDT/CICS transaction, HFM.
2. Enter your user ID and press Enter.
3. The ZDT/CICS logon panel is displayed:

```
──────────────────────── Z Data Tools for CICS Logon ────────────────────────

     Enter Logon parameters

     Userid . . . . . . . . USER

     Password . . . . . . .

     New Password . . . . .

     Node . . . . . . . . . FMNODE      (Machine the job is to be run on)

     Procedure. . . . . . . FMC62FM4    (Procedure to run Z Data Tools)

     Profile data set . . . USER.HFI.HFIPROF

     Prefix . . . . . . . . USER      (Default prefix for data sets)

     Jobcard
     //FM&TERM.B JOB (,,,,),&USER,
     //     MSGCLASS=A,MSGLEVEL=(1,1),CLASS=A


     F1=Help   F3=Logoff   F4=Reset   F5=Wait   F6=Default   F12=Cancel
     Enter=Submit
```

   If you specified START=TASK in the parameters in HFM3POPT, or you used (START=TASK) to initiate your ZDT/CICS session, and a valid port number has been provided, then you will see a logon panel similar to this.

```
──────────────────────── Z Data Tools for CICS Logon ────────────────────────

     Enter Logon parameters

     Userid . . . . . . . . USER

     Password . . . . . . .

     New Password . . . . .

     Profile data set . . . USER.HFI.HFIPROF

     Prefix . . . . . . . . USER      (Default prefix for data sets)

     Port . . . . . . . . . 2054

     Host name



     F1=Help   F3=Logoff   F4=Reset   F6=Default   F12=Cancel
```

4. Enter your password and press Enter.

5. Unless using the START=TASK option, the batch job to start the ZDT/CICS session will be submitted, and then the ZDT/CICS Primary Option Menu should be displayed, as shown in Figure 70: The ZDT/CICS Primary Option Menu on page 374.
    - If after submitting the ZDT/CICS batch job, message HFMCA016 is displayed (Job *jobname* not responding), the ZDT/CICS job may have failed or may be queued. Examine the job log for further details and possible error messages.

      If the job log cannot be found (the ZDT/CICS job was never started) verify that the HFMRDR DD statement has been properly defined in the CICS® startup JCL and that your system has sufficient initiators. See Updating the CICS startup procedures on page 330 for further information.

      Some of the most common reasons the batch job may have failed or been queued are:
        - Insufficient authority to access one of the libraries in the ZDT/CICS procedure. Check the system log for security messages.
        - TCP/IP is not active on the CICS® region. Check for TCP/IP error messages in the job log and verify that TCP/IP is active on the CICS® region.
        - The CICS® Sockets interface has not been started
        - Duplicate job name. Verify if another user may have specified the same job name in the ZDT/CICS **Jobcard** input field, or a previous ZDT/CICS job is still active.
        - JCL errors. Check the job log for further information.

Figure 70. The ZDT/CICS Primary Option Menu

```
 Process   Options   Help
--------------------------------------------------------------------------------

ZDT/CICS                      Primary Option Menu

0  Settings      Set processing options                 User ID . : XYUSER
1  View          View data                              CICS User : XYC5DFLT
2  Edit          Edit data                              CICS Appl : C62D2FM5
3  Utilities     Perform utility functions              Date. . . : 2022/10/10
4  Templates     Template and copybook utilities        Time. . . : 11:32
B  ZDT           HCL Z Data Tools
I  ZDT/IMS       HCL Z Data Tools IMS component
D  ZDT/Db2       HCL Z Data Tools Db2 component
X  Exit          Terminate ZDT/CICS

Processing Options:
 CICS Resource
 1  1. File
    2. Temporary Storage
    3. Transient Data


Command ===> ____________________________________________________________
F1=Help     F3=Exit     F4=CRetriev F7=Backward F8=Forward  F10=Actions
F12=Cancel
```

Note: The options B, I, and D allow the user to invoke Z Data Tools and ZDT/IMS from this primary option menu. These options will not appear if you did not customize Z Data Tools Base function security to allow

this access. For more information, see Accessing other Z Data Tools functions from the primary option menu on page 332.

6. Enter VER on the command line to display the release level and PTF level of ZDT/CICS and Z Common Components. A panel is displayed similar to this:

```
HCL Z Data Tools
                              Version 17 Release
                                    1 Modification 0
CICS Component
(not APF authorized)

Service Levels of installed components

             Base       IMS        Db2        CICS       ZCC
English      -NONE-     -NONE-     -NONE-     -NONE-     -NONE-
```

Notes:

a. ZDT/CICS is always shown as `not APF authorized`, even if you have made Z Data Tools APF-authorized, as ZDT/CICS cannot run APF-authorized.

b. When you first install ZDT/CICS, `-NONE-` will be shown under each component that you have installed. Subsequently, when you have applied service to ZDT/CICS, a PTF number will be shown, indicating the PTF level of each component you have installed. If you have not installed a component, that component will not be shown at all.

   If you have installed the Japanese language component, another line will be displayed indicating the service level of that component.

7. Press the Exit key (PF3) to exit from the VER display.
8. Enter `VERCICS` on the command line to display the level and version of the running Z Data Tools batch job and ZDT/CICS on all connected CICS® regions. A panel is displayed similar to this:

```
 Process   Options   Help
------------------------------------------------------------------------------
ZDT/CICS                       Primary Option Menu
                                  CICS Levels

 HCL Z Data Tools Version 1 Release 1 CICS Component

 Current PTF: -NONE-      Level: 1

    Connected ZDT/CICS Levels
    Sysid    Applid      Version    PTF        Level
             C62D2FM5    V17R1      -NONE-         1

 Command ===>                                                 Scroll PAGE
 F1=Help     F3=Exit     F4=CRetriev F7=Backward F8=Forward  F10=Actions
 F12=Cancel
```

Notes:



a. The version, PTF level, and buffer layout level for the running Z Data Tools batch job, and ZDT/CICS on all interconnected CICS® regions (if the HFM3LVL program is available on those regions), is displayed.
b. The level of the Z Data Tools CICS® component and all ZDT/CICS programs should be the same, otherwise some functions will fail.
c. If more than one interconnected CICS® region is displayed, each region should be at the same level as the running ZDT/CICS batch job. If the level or version is different, these values will be highlighted in red, indicating a possible error.
d. A difference in version or level is likely to be due to maintenance not being applied to all instances of ZDT/CICS, or the ZDT/CICS programs not being refreshed in the CICS® region after maintenance was applied.
e. If HFM3LVL is not available on a connected system, the text `LOAD ERROR` *condition* will be displayed. Ensure that HFM3LVL can be loaded on the connected CICS® region and retry the VERCICS command.
f. If an entry for an interconnected CICS® region where ZDT/CICS has been installed is not displayed in the list, this indicates one of the following:
    - An active connection to the region was not acquired when the VERCICS command was issued.
    - The ZDT/CICS program HFM3CICS could not be linked to and loaded on the connected CICS® region.
    - A TD queue has been specified for the CONN option in HFM3POPT and the APPLID of the connected CICS® region has not been added to this queue.

    Verify that an active connection to the remote region has been acquired and HFM3CICS is available, then retry the VERCICS command.

9. Press the Exit key (PF3) to exit from the VERCICS display.
10. Press the Exit key (PF3) twice more to exit from ZDT/CICS.

You can now complete the installation of ZDT/CICS by performing the ACCEPT processing. The steps involved are described in the Z Data Tools Program Directory.

Part V. Z Data Tools on VS Code

HCL Z Data Tools is a host component that supports a Microsoft Visual Studio Code IDE.

The HCL Z Data Tools VS Code extension works in conjunction with the Zowe Explorer Microsoft Visual Studio Code extension. HCL Z Data Tools provides flexible, user-friendly tools for enhanced file processing when working with z/OS data sets.

This comprehensive solution provides intuitive interfaces for accessing, editing, and managing data sets and files on z/OS systems, and is a modern alternative to working with HCL Z Data Tools.

To download the Z Data Tools VS Code extension, visit the download site My HCLSoftware.

Chapter 36. Installation requirements

Host and client requirements for HCL Z Data Tools VS Code extension Microsoft Visual Studio Code extension.

Host requirements

- HCL Z Common Components 17.1.0 or later
- HCL Z Data Tools V17.1.0 or later

Client requirements

Z Common Components Microsoft Visual Studio Code extension (always use the same version as the IBM Fault Analyzer for z/OS® Microsoft Visual Studio Code extension)

Zowe™ Explorer V3.2.1

Note: To get the most up-to-date information on supported operating systems, detailed system requirements, and hardware requirements, you can generate a dynamic software product compatibility report.

1. Go to the Software Product Compatibility Reports page.
2. Type the product name in the Search by product name field and click the product name in the list.
3. In the search results, select a release.

Part VI. Preparing for Z Data Tools Remote Services

Z Data Tools supports a number of services using resources accessed on a remote system via the ZCC server connection. When the remote ZCC server is configured for SSL/TLS, the local Z Data Tools system validates the remote host's server certificate during the SSL/TLS handshake by verifying the Certificate Authority (CA) of the server's certificate is registered as trusted.

By default, Z Data Tools searches local SITE certificates for the CA certificate of the remote system and verifies that it is trusted. Consequently, when importing a CA certificate for remote services, you should import it as a SITE certificate. For example, using RACF®:

```
RACDCERT ADD('hlq.ZCC.CA.EXPORT') SITE TRUST WITHLABEL('your label')
```

Alternatively, you can use the CERTRUST keyword of the HFM4POPT module to specify a trust store other than SITE. Note that all users of remote services need authority to access the nominated key store. See Customizing miscellaneous options in HFM4POPT on page 383 for more information about the CERTRUST keyword and the HFM4POPT module.

HFMAUTH DD usage and security

When using Z Data Tools to create a remote connection through the menu option 11, the entered details are stored (in an internal format) in a file allocated to the HFMAUTH DD. If such an allocation does not pre-exist, as is normally the case, a data set is created as `Userid.HFMAUTH` and allocated to the HFMAUTH DD.

When running batch functions and specifying remote resources, the HFMAUTH DD needs to be included in JCL to provide the stored connection details.

Similarly, if there is a requirement to share remote connection details amongst users, you may pre-allocate the HFMAUTH DD in TSO/ISPF and Z Data Tools reads the currently allocated HFMAUTH.

Define profiles for data sets allocated to the HFMAUTH DD with UACC(NONE). Set up storage management routines so that these data sets are encrypted. For information on setting up data set encryption, see the z/OS documentation on "Using DFSMS functions", section "Data set encryption".

Part VII. Customizing Z Common Components server

Chapter 37. Customizing the ZCC server

These topics describe how to customize the ZCC server.

Do this after you have installed Z Data Tools if you plan to use ZDT/CICS.

This server runs a z/OS® UNIX™ process that will listen for a connection request on a specific port. The server can be started manually or during IPL by execution of a customized procedure.

When a user runs the HFM transaction using the ZDT/CICS interface, a connection request will be received by the server from the client. The server will then verify the credentials of the client (user ID and password, or passphrase) and if valid, create a new HFM process for that user. The user ID of the newly created process will be changed to that of the client and all further communication will occur between the client and the newly created HFM process using TCP/IP. The server will then wait for another connection request from a new client.

Z Common Components server configuration

The default name of the ZCC server is HFISRV1. You can change this name to suit your site's requirements.

You must provide and configure the server options in a data set specified on the CONFIG DD statement in the server procedure (HFISRV1). The data set defined, or concatenated, to this DD statement must be sequential, or a member from a PDS or PDSE.

Sample configuration data is provided in the HFM.SHFMSAM1 member HFMSAMCS. The sample configuration data needs customizing to your environment.

The Z Data Tools sample HFI configuration member HFMSAMCS specifies SPAWN_PROGRAM=HFMCSEP. This program allows Z Data Tools to run authorized and ensures that ZDT/CICS can:

- Perform auditing
- Access tapes
- Use remote services

High level qualifiers for both Z Data Tools and the ZCC server should be adjusted according to your site installation details. To simplify the creation of this configuration data, a sample REXX exec is supplied in the HFM.SHFMEXEC library, called HFMCSTCS. Running this exec from TSO prompts for the various data set names required, and produces a usable configuration for the ZCC server.

Refer to the *Z Common Components Customization Guide and User Guide* for any additional customization which may be required.

Additional security considerations

All the libraries listed in the SPAWN_STEPLIB statement must be APF-authorized, as the spawned address space needs to attach TSO to support authorized services such as SMF logging, IEBCOPY invocation, and Db2® CLIST processing.

Note: The user ID the HFISRV1 procedure runs under must also have an OMVS RACF® segment defined.

See *z/OS Security Server RACF Security Administrator's Guide*, or equivalent documentation for your security product, for more information about started tasks and security.

Address space timeout

When an address space is launched for a client and has completed its current function, the address space will be waiting for TCP/IP communications from the ZDT/CICS client.

Accordingly, the client address space may be subject to an s522 abend if waiting longer than the active site settings for job wait time. The job wait time is controlled by the **JWT** parameter of the **SMFPRM*xx*** member, but may also be set to never time out by the site settings for **MAXCPUTIME** in the site's **BPXPRM*xx*** member. You should consider what settings are appropriate for your site and your use of ZCC server.

Appendix A. Z Data Tools options

This section describes the Z Data Tools options. The syntax described here applies to the HFM0POPI macro, which is installed by default in HFM.SHFMMAC1. You can modify these Z Data Tools options to suit your requirements, for Z Data Tools Base function, ZDT/Db2, ZDT/IMS, and ZDT/CICS.

However, if you want to change an option setting for Z Data Tools Base function, and also for ZDT/Db2, ZDT/IMS, or ZDT/CICS, you must change the option in each of HFM0POPT, HFM1POPT, HFM2POPT, HFM3POPT, and HFM4POPT and apply the appropriate usermods. The options only take effect in the component where they are applied.

For information about changing Z Data Tools miscellaneous options, see Changing the default options on page 384.

For information about changing Z Data Tools Base function options, see Changing the default options on page 55.

For information about changing ZDT/Db2 options, see Changing the default options on page 180.

For information about changing ZDT/IMS options, see Customizing the ZDT/IMS installation options module on page 264.

For information about changing ZDT/CICS options, see Changing the default options on page 341.

Customizing miscellaneous options in HFM4POPT

Miscellaneous processing options are supplied with Z Data Tools in the module HFM4POPT. This options module supports the following keywords:

REPOS

REPOS = *template.repostry.dsn*

Where *template.repostry.dsn* is the name of the template repository file that is defined using Z Data Tools ISPF option 7.7.1.

CERTRUST

CERTRUST = *keystore-pattern*

Specifies the trust key store for Z Data Tools remote services certificate validation. Z Data Tools remote services connects to remote systems using the ZCC server. When the common server is configured for SSL/TLS, the local Z Data Tools system validates the certificate of the remote host server by verifying that the Certificate Authority (CA) of the certificate is registered as trusted in the key store(s) specified by the *keystore-pattern*. Consequently, the *keystore-pattern* identifies one or more key stores that contain relevant CA certificates of all remote servers.

The *keystore-pattern* identifies the key store owner and the key store name, separated by a forward slash (/). For example:

```
CERTRUST=userid/keyring
```

The *keystore-pattern* can include asterisks (*) as wild cards. The maximum length you can specify for the value of the CERTRUST keyword is 246 bytes. CERTRUST is optional. If omitted, the default pattern used is `CERTRUST=*SITE*/*`.

Changing the default options

You can find the HFM4POPI macro statement in HFM.SHFMSAM1(HFM4POPT). Change the options as follows:

1. Copy the member HFM4POPT from HFM.SHFMSAM1 into your own source library.
2. Change the default options in your copy of the HFM4POPT member as required.
3. Modify the HFM4UMDP member in HFM.SHFMSAM1 to meet your site's requirements. Refer to the usermod for information about changes you might need to make.
4. Install SMP/E usermod HFM4UMDP.

Note: You can also use the sample job HFM4POPH to assemble HFM4POPT if you do not want to use SMP/E.

ABENDCC

```
ABENDCC= { NONE | nnnnn }
```

ABENDCC

Specifies how Z Data Tools processing resulting in a non-zero return code (including customized return codes) will be completed in batch.

NONE

The job step will terminate with a non-zero return code. The default is NONE.

nnnnn

If the return code is greater than or equal to *nnnnn*, Z Data Tools will abend with U999, Reason Code=888 (hex: 378). *nnnnn* can be any integer between 1 and 32767. *nnnnn = 0* is the same as NONE. *nnnnn* can be selected as appropriate, for specific return codes (including customized return codes) issued by Z Data Tools functions.

Note: Specification of any non-zero integer will prevent Z Data Tools from intercepting abnormal terminations (system abends).

ALTCOB

```
ALTCOB = { NO  | YES }
```

ALTCOB

Determines which Z Data Tools COBOL compiler is to be used.

NO (default)

Use the default internal compiler.

YES

Use the alternate internal compiler

The following table details the available Z Data Tools COBOL compilers.

Table 71. Available COBOL compilers

Language	Option	Version	Notes
COBOL	ALTCOB=NO	6.4	50 MB region size minimum is recommended.
COBOL	ALTCOB=YES	4.2	

ALTPLI

```
ALTPLI = { NO  | YES }
```

ALTPLI

Determines which Z Data Tools PL/I compiler is to be used.

NO (default)

Use the default internal compiler.

YES

Use the alternate internal compiler

The following table details the available Z Data Tools PL/I compilers.

Table 72. Available PL/I compilers

Language	Option	Version
PL/I	ALTPLI=NO	6.1
PL/I	ALTPLI=YES	5.3

ASCII

```
ASCII= { NO  | YES  | IN  | OUT  | BOTH }
```

ASCII

Specifies the requirements for translation of tape data between ASCII and EBCDIC.

NO

No translation is performed.

IN or YES

Translates tape input from ASCII format to EBCDIC format.

OUT

Translates tape output from EBCDIC format to ASCII format before writing it to the output tape.

BOTH

Translates tape input from ASCII format to EBCDIC format, and translates tape output from EBCDIC format to ASCII format.

Use this option to convert ASCII tapes from one ASCII character set to another.

Input translation is done using the Z Data Tools ASCII/EBCDIC translate table; output translation is done using the Z Data Tools EBCDIC/ASCII translate table.

Note: Changing the ASCII option has no effect for ZDT/Db2, ZDT/IMS, or ZDT/CICS.

AUDDATAC

```
AUDDATAC=XXXXXXXX
```

AUDDATAC

Specifies the SMS data class that Z Data Tools uses when allocating the Z Data Tools Audit Log. The maximum length you can specify for AUDDATAC is eight (8) bytes. AUDDATAC is optional. If omitted the SMS data class specified in PDATAC is used, if supplied.

AUDITHLQ

```
AUDITHLQ= { user-defined audithlq  | user-defined prefix }
```

AUDITHLQ

Specifies an optional high-level qualifier, or data set prefix, for the Z Data Tools audit log data sets.

The format of the generated data set names is shown in Table 73: The format of generated data set names on page 386.

Table 73. The format of generated data set names

ZDT component	AUDITHLQ value	Audit log data set name format
Base, ZDT/CICS	blank (none)	`userid.HFMLOG.Dyymmdd.Thhmmss`
Base, ZDT/CICS	*audithlq*	`audithlq.userid.HFMLOG.Dyymmdd.Thhmmss`
Base, ZDT/CICS	*prefix*	`qual1.<qual2.><qual3.>Dyymmdd.Thhmmss`
ZDT/Db2	blank (none)	`userid.HFM2AUD.<ssid>.Dyymmdd.Thhmmss`
ZDT/Db2	*audithlq*	`audithlq.HFM2AUD.<ssid>.Dyymmdd.Thhmmss`

Table 73. The format of generated data set names (continued)

ZDT component	AUDITHLQ value	Audit log data set name format
ZDT/Db2	*prefix*	`qual1.<qual2.><qual3.>Dyymmdd.Thhmmss`
ZDT/IMS	blank (none)	`userid.IMSAUDIT.Dyymmdd.Thhmmss`
ZDT/IMS	*audithlq*	`audithlq.IMSAUDIT.Dyymmdd.Thhmmss`
ZDT/IMS	*prefix*	`qual1.<qual2.><qual3.>Dyymmdd.Thhmmss`

where:

audithlq

Is any 1-8 character constant that is valid in the context of a data set name

userid

Is the user ID creating the data set

Dyymmdd

Is the date of the activity

Thhmmss

Is the time of the activity

prefix

`qual1.<qual2.><qual3.>`, as defined below

The following values can be specified for each level (qual1, qual2, or qual3) of the data set name prefix.

XXX

Any 1-8 character constant that is valid in the context of a data set name.

&&PREFIX

Indicates that the user's TSO prefix should be used. This will be null if TSO NOPREFIX is in effect.

&&USER

Indicates that the user's logonid (ISPF system variable ZUSER, stored in the shared pool) should be used.

&&UID

Indicates that the user's TSO prefix should be used, when the value is non-blank. When TSO NOPREFIX is in effect, the user's logonid (ISPF system variable ZUSER, stored in the shared pool) should be used.

&&FUNCOD

Indicates that the Z Data Tools internal function code should be used. Specifying this parameter allows the Z Data Tools function that generated the audit log data set to be included in the audit log data set name.

&&SSID

Indicates that the currently connected subsystem name (Db2® or IMS™ only) should be used. If this symbolic parameter is specified in HFM0POPT or HFM3POPT, the value is ignored, the substituted value is null.

Notes:

- When a symbolic parameter is specified (any value preceded by &&), a trailing period is required, except for qual3.
- You cannot specify a combination of characters and symbolic parameters at the same level of the data set name prefix.
- You cannot use more than one symbolic parameter at the same level of the data set name prefix.
- You cannot specify a prefix value with more than 3 levels.
- You can specify a trailing period to distinguish between a high level qualifier value and a prefix value comprising a single level.

Examples:

AUDITHLQ=

Default

AUDITHLQ=HFMAUD,

HLQ value is HFMAUD

AUDITHLQ=HFMAUD.,

Prefix value is HFMAUD

AUDITHLQ=&&USER..HFM1AUD

Two-level prefix

AUDITHLQ=HFMAUD.&&FUNCOD..&&USER

Three-level prefix

AUDITHLQ=HFMAUD.&&USER..&&SSID

Three-level prefix, SSID non-blank (Db2® and IMS™ only)

AUDITLOG

```
AUDITLOG={NO | YES | DEMAND}
```

AUDITLOG

Specifies whether or not audit trail logging for the Z Data Tools Base function or for ZDT/CICS is mandatory.

NO

Audit trail logging is not mandatory.

YES

Audit trail logging is mandatory.

DEMAND

Audit trail logging is mandatory and a job described by skeleton HFM0FTAD (HFM3FTAD for ZDT/CICS) will be submitted to report on the changes made by a user when exiting the edit function.

Note: Changing the AUDITLOG option has no effect if the relevant HFM*n*PARM PARMLIB member has FMAUDIT **SAF_CTRL=YES**.

AUDMGMTC

```
AUDMGMTC=XXXXXXXX
```

AUDMGMTC

Specifies the default SMS management class that Z Data Tools uses when allocating the Z Data Tools Audit Log. The maximum length you can specify for AUDMGMTC is eight (8) bytes. AUDMGMTC is optional. If omitted the SMS management class specified in PMGMT is used, if supplied.

AUDPQTY

```
AUDPQTY= { 16 | nnnnnnnn }
```

AUDPQTY

Specifies the amount of DASD space to be used for primary space allocation of the Z Data Tools Audit Log. The range depends on the space unit specified and the DASD device type.

AUDSQTY

```
AUDSQTY= { 8 | nnnnnnnn }
```

AUDSQTY

Specifies the amount of DASD space to be used for secondary space allocation of the Z Data Tools Audit Log. The range depends on the space unit specified and the DASD device type.

AUDSTORC

```
AUDSTORC=XXXXXXXX
```

AUDSTORC

Specifies the default SMS storage class that Z Data Tools uses when allocating the Z Data Tools Audit Log. The maximum length you can specify for AUDSTORC is eight (8) bytes. AUDSTORC is optional. If omitted the SMS storage class specified in PSTORC is used, if supplied.

AUDSUNIT

```
AUDSUNIT={CYL | REC | KB | MB | TRK}
```

AUDSUNIT

Specifies the unit of primary and secondary space to be allocated to the Z Data Tools Audit Log.

REC

Record of average size.

KB

Kilobyte, a kilobyte is 1024 bytes.

MB

Megabyte, a megabyte is 1048576 bytes.

TRK

Track of a direct access storage device (DASD).

CYL

Cylinder of a DASD.

AUDUNIT

```
AUDUNIT=XXXXXXXX
```

AUDUNIT

Specifies the default permanent unit that Z Data Tools uses when allocating the Z Data Tools Audit Log. The maximum length you can specify for AUDUNIT is eight (8) bytes. AUDUNIT is optional. If omitted the SMS storage class specified in PUNIT is used.

AUXDATAC

```
AUXDATAC=XXXXXXXX
```

AUXDATAC

Specifies the default SMS data class that Z Data Tools uses when allocating the auxiliary data set for a full function edit session. AUXDATAC is optional. The maximum length you can specify for AUXDATAC is eight (8) bytes. If not supplied, no SMS data class will be specified when allocating the auxiliary data set.

AUXDSN

```
AUXDSN=dsn
```

AUXDSN

Specifies an installation-specific data set name to be used as the auxiliary file for a full function edit session.

You can specify any valid name for this data set. If you specify &&USER. as a high-level qualifier for AUXDSN, the data set name has a high-level qualifier of your user ID. If you specify &&PREFIX. as the high-level qualifier for AUXDSN, the data set name has a high-level qualifier of the user's TSO PREFIX.

The auxiliary file must be a REUSEable RRDS file, whose record length definitions are compatible with the file being edited. If this field is left blank, an RRDS file is defined for the edit session and subsequently deleted.

AUXHLQ

AUXHLQ=*aux-hlq*

AUXHLQ

Specifies an installation-specific high-level qualifier to be used in the name of the auxiliary data set for a full function edit session.

You can specify any valid qualifier, including multilevel if required, up to a total length of 24 characters. If you specify &&USER. for any part of the qualifier in your HFM0POPT macro, this is replaced with your user ID. If you specify &&PREFIX. for any part of the qualifier, this is replaced with the user TSO prefix.

&&USER. and &&PREFIX. can also be concatenated with a character string. A multilevel qualifier with symbols follows the same rules as in the specification of a data set name with symbolic parameters in JCL procedures. Thus, a single period used after a symbol combines the symbol with the text following it (See Example 1 below); a period separating parts of a multilevel qualifier must be doubled (see Example 2 below).

Example 1

If the user ID is XXXX and the TSO prefix is YYYY, then specifying

```
&&USER.1.TEMP.&&PREFIX
```

results in a high-level-qualifier of

```
XXX1.TEMP.YYYY
```

Example 2

If the user ID is ZZZZ then specifying

```
HFM.&&USER..ABCD
```

results in a high-level-qualifier of

```
HFM.ZZZZ.ABCD
```

Note:

1. The AUXHLQ parameter can be up to 24 characters long. However, temporary data sets can have different lengths, and the system part of the data set name may be longer than 20 characters. In this case, minor levels of AUXHLQ are ignored, to the extent that the final data set name is less than or equal to 44 characters.
2. Concatenation of &&USER.string, or &&PREFIX.string, could result in a part of the high-level qualifier being longer than eight characters. In this case, the string is shortened to the extent that the final part of the qualifier is eight characters long.

AUXMDSN

```
AUXMDSN=dsn
```

AUXMDSN

Specifies the data set name to be used as the model file when defining an the auxiliary file for a full function edit session. A model file should be used to determine the volume placement of the auxiliary data set in a non-SMS environment. The model file must be a VSAM cluster.

AUXMGMTC

```
AUXMGMTC=XXXXXXXX
```

AUXMGMTC

Specifies the default SMS management class that Z Data Tools uses when allocating the auxiliary data set for a full function edit session. AUXMGMTC is optional. The maximum length you can specify for AUXMGMTC is eight (8) bytes. If not supplied, no SMS management class will be specified when allocating the auxiliary data set.

AUXSTORC

```
AUXSTORC=XXXXXXXX
```

AUXSTORC

Specifies the default SMS storage class that Z Data Tools uses when allocating the auxiliary data set for a full function edit session. AUXSTORC is optional. The maximum length you can specify for AUXSTORC is eight (8) bytes. If not supplied, no SMS storage class will be supplied when allocating the auxiliary data set.

BDY

```
BDY = { NO  | YES }
```

BDY

When data is displayed or printed in SNGL format, this option specifies whether fields that start beyond the current record boundary should be suppressed.

NO

Fields starting beyond the record's boundary are not suppressed.

YES

Fields starting beyond the record's boundary are suppressed.

CCSID

```
CCSID= { 00037 | nnnnn }
```

CCSID

Specifies the CCSID to be used as the default CCSID when Z Data Tools is run in batch. The CCSID specified should be a valid CCSID and a CCSID which is recognised by the Z/OS system the batch job will be run on. This option does not apply to ZDT/Db2.

COBDBCS

```
COBDBCS= { NO | YES }
```

COBDBCS

Specifies if the **DBCS** COBOL compiler option is to be specified when compiling COBOL copybooks.

YES

The **DBCS** COBOL compiler option is specified

NO

The **DBCS** COBOL compiler option is not specified.

COBDPC

```
COBDPC= { NO | YES }
```

COBDPC

Specifies if the **Decimal-point is comma** SPECIAL-NAMES paragraph is included when compiling COBOL copybooks.

YES

The paragraph is included.

NO

The paragraph is not included.

COBEXTND

```
COBEXTND= { NO | YES }
```

COBEXTND

Specifies if the **Arith(extend)** COBOL compiler option is to be specified when compiling COBOL copybooks.

YES

The **Arith(extend)** COBOL compiler option is specified.

NO

The **Arith(extend)** COBOL compiler option is not specified.

COBMAXRTN

```
COBMAXRTN= { 4 | nn }
```

COBMAXRTN

Specifies the maximum warning or error code received from the COBOL compiler, when compiling a copybook. Any code higher than specified stops the current Z Data Tools function. You can specify a value from 0 to 99. The default value is 4.

COBMCASE

```
COBMCASE= { NO | YES }
```

COBMCASE

Specifies if the original case of the field name as coded in the COBOL copybooks is to be retained. This feature is only available if you are running with the Z Data Tools COBOL compiler or a minimum compiler level of Enterprise COBOL V4R1.

YES

The original case of the field name as coded is to be retained.

NO

The field name is translated to uppercase.

COMMADEC

```
COMMADEC= { NO | YES }
```

COMMADEC

Specifies whether to use a comma character (,) as the decimal separator.

NO

Default. A period character is used as the decimal separator.

YES

A comma character is used as the decimal separator when processing numeric values with the following exceptions where a period character (.) is always used instead:

- Numeric edited fields. (Type ZE and ZA).
- External floating-point fields. (Type FE).
- Any numeric value specified in a template. For example: Criteria or data create.

COMPLANG

```
COMPLANG={ COBOL | PL/I | HLASM | AUTO }
```

COMPLANG

Specifies the language which is to be used for compiling a source member to create a template.

COBOL

Use the COBOL compiler

PL/I

Use the PL/I compiler

HLASM

Use the HLASM compiler

AUTO

Z Data Tools analyzes the source to determine whether the language of the source is COBOL or PL/I and creates a template based on the result. It does not recognize HLASM source.

Note: The COMPLANG setting in HFM0POPT is the equivalent of the LANG parameter in batch functions.

- If COMPLANG is set it determines the installation default language for online and batch compilation.
- If COMPLANG is not specified then COBOL is the installation default for online compilation and AUTO is the default for batch compilation.
- If a value of COBOL, HLASM, PL/1, or AUTO is specified (in the **Compiler Language Selection** panel or through the LANG parameter in a batch job) it overrides the default language.

CREPLACE*n*

```
CREPLACE n = ( from_string , [to_string] )
```

CREPLACE*n*

Where *n* is a number in the range 1 to 5, specifies the from pseudo-text character string that you want to be replaced by another character string in your COBOL copybooks via a COBOL REPLACE statement. The *from_string* must be 1 to 30 characters in length. The *to_string* is optional. If supplied it must be 1 to 30 characters in length. If it is not supplied a comma must still be used to delimit the *from* string.

CSYSLIB*nn*

```
CSYSLIBnn=dsname
```

CSYSLIB*nn*

Where *nn* is a number in the range 1 to 10, specifies the names of SYSLIB data sets which are searched in the order specified for COPY or INCLUDE members for the COBOL compilation.

CYLHD

```
CYLHD={ ABSOLUTE | RELATIVE }
```

CYLHD

Defines the way you specify the location of a disk data set in disk functions.

ABSOLUTE

Z Data Tools interprets cylinder-head and track-number values as actual physical addresses.

RELATIVE

Z Data Tools interprets cylinder-head and track-number values as relative to the start of the data set.

Note: Changing the CYLHD option has no effect for ZDT/Db2, ZDT/IMS, or ZDT/CICS.

DATAHDR

```
DATAHDR={ YES | NO }
```

DATAHDR

Specifies whether record header information (that is, record number and length) is added to each record printed.

YES

Record header information is included in the output.

NO

Record header information is not included in the output.

Note:

1. This option affects only DSP and FCH processing.
2. This option affects only output in character format. When printing in hexadecimal format, the output always includes a data header.

DSINFO

```
DSINFO = { NO  | YES }
```

DSINFO

Specifies whether additional data set information for input and output sources should be produced in batch reports for the DSP, DSM, DSC, and FCH commands.

NO

No additional data set information is generated.

YES

Generates data set information including DSORG, RECFM, LRECL, BLKSIZE, and VSAM attributes including average record length, maximum record length, key offset, key length and reuse, for input and output data sets when applicable.

DSPINC

```
DSPINC= { 2048  | nnnnnnn }
```

DSPINC

Specifies the amount (in KB) by which a clipboard will expand when it becomes full. Default is 2048 (2MB). Maximum is 4194303 (4GB).

DSPMAX

```
DSPMAX= { 2097152  | nnnnnnn }
```

DSPMAX

Specifies the maximum size in KB, of any one clipboard. Default is 2097152 (2GB). Maximum is 4194303 (4GB). Note that dataspace limits may be otherwise determined by the installation IEFUSI system exit. See *z/OS® MVS™ Installation Exits*, SA22-7593, for more information about system exits.

DSPMIN

```
DSPMIN={ 1024 | nnnnnnn }
```

DSPMIN

Specifies the starting size in KB of clipboards created by Z Data Tools. Default is 1024 (1MB). Maximum is 4194303 (4GB).

DSPNUM

```
DSPNUM={ 11 | nn }
```

DSPNUM

Specifies the maximum number of clipboards that Z Data Tools can create. The default, and maximum, is 11.

DUMP

```
DUMP={ UPDOWN | ACROSS }
```

DUMP

Specifies the format of hexadecimal print output (for example, when you use Tape Print with **Print format** set to HEX).

UPDOWN

The two digits making up the hexadecimal representation of each EBCDIC character are displayed vertically directly under that character. This is the Z Data Tools dump format.

ACROSS

The hexadecimal digits are displayed as 2 groups of 4 fullwords resulting in 32 hexadecimal digits followed by the EBCDIC character display to the right of the hexadecimal display. This corresponds to the system dump format.

EDMAXVIRT

```
EDMAXVIRT={ 0 | nnnnnnn }
```

EDMAXVIRT

Specifies the limit in KB, that an editor session should use. The default is 0, meaning it is limited only by available region. Note that the editor generally allocates virtual storage in 512KB blocks, so the actual usage may be rounded up to the next 512KB block. Specifying a value larger than the available virtual storage has no net effect on the editor's virtual storage usage. Maximum is 4194303 (4GB).

EDITCAPS

```
EDITCAPS=( [{ INITOFF | INITON }] , [{ PROFILE | INITFIXED }] , [{ CAPSCMD | NOCAPSCMD }] )
```

EDITCAPS

Controls the setting of the CAPS option in edit. A number of options are provided, which allow you to:

- Set the initial value (either CAPS ON or CAPS OFF) for new Z Data Tools users.
- Force all users of the Z Data Tools editor to commence an edit session with either CAPS ON or CAPS OFF; alternatively to allow the user to control the initial CAPS setting using a new editor option, that is saved in the user's ISPF profile.
- Disable the CAPS command within a Z Data Tools edit session.

  You can use certain settings of the EDITCAPS parameters to create an environment where every Z Data Tools edit session starts with CAPS ON, and this cannot be altered using the CAPS OFF command.

There are three sub-parameters to EDITCAPS. You do not have to specify them all in HFM0POPT, but if you do so, they must be in the order shown in the syntax diagram. If you specify more than one, they must be separated by commas. If you specify any sub-parameters, you must also provide the parentheses.

The default is `(INITOFF,PROFILE,CAPSCMD)`.

These sub-parameters are described below.

INITOFF, INITON

Determines the initial CAPS setting in the user's profile, when the profile is first created.

For users with an existing Z Data Tools profile, the effect depends on the setting of the second parameter (described below). If INITFIXED is specified for the second parameter, then every Z Data Tools edit session begins with either CAPS OFF if INITOFF is specified, or CAPS ON if INITON is specified.

PROFILE, INITFIXED

Determines if the user can alter the initial setting of CAPS in Z Data Tools edit. INITFIXED prevents the user from altering the initial setting of CAPS. PROFILE allows the user to change this initial setting and it is preserved between Z Data Tools sessions in the user's profile.

CAPSCMD, NOCAPSCMD

Determines if the user is permitted to use the CAPS command within a Z Data Tools edit session. NOCAPSCMD does not affect the CASE command, which remains available for use.

When CAPSCMD is specified, the user can turn CAPS on or off in a Z Data Tools edit session, as required. When NOCAPSCMD is specified, the CAPS command is disabled, and is fixed at whatever setting was selected when the Z Data Tools edit session commenced.

EOD

EOD=*delimiter*

EOD

Specifies the end-of-data delimiter. Must be between one and eight characters. For character data, enclose the string in quotation marks if it contains blanks, commas, or lowercase letters. For hexadecimal data, enter an X followed by the string enclosed in quotation marks (for example, X'04').

EXCITRAN

```
EXCITRAN= { HFMX | transid }
```

EXCITRAN

Specifies the transaction name used for the mirror transaction required for the external CICS® interface support.

transid

This value must be a valid CICS® transaction ID and must correspond to the transaction ID provided in the customized HFMCCONN job used to define the connection transaction to the CICS® regions.

For more information, see Setting up the External CICS interface (EXCI) access on page 333.

FMEDITOR

```
FMEDITOR= { YES | NO }
```

FMEDITOR

Specifies which editor is to be used when viewing or browsing a member from a member selection panel or when viewing generated output.

YES

The Z Data Tools editor is used.

NO

The ISPF editor is used.

Note: In a CICS® environment, only the Z Data Tools editor is available.

HEADERPG

```
HEADERPG= { YES | NO }
```

HEADERPG

Specifies whether a header page (a title page with Z Data Tools on it) is included in print output. A header page does not appear on output routed to the terminal, even if HEADERPG=YES is set.

YES

A header page is included in the print output.

If you specify YES then the output will also show the Z Data Tools installed components, version and PTF level. This is the same information as is produced by the $$FILEM VER control statement.

NO

A header page is not included in the print output.

Note: Changing the HEADERPG option has no effect for ZDT/Db2, ZDT/IMS, or ZDT/CICS.

HLDBCS

```
HLDBCS= { NO | YES }
```

HLDBCS

Specifies if the **DBCS** HLASM compiler option is to be specified when compiling assembler copybooks.

YES

The **DBCS** HLASM compiler option is specified

NO

The **DBCS** HLASM compiler option is not specified.

HLMAXRTN

```
HLMAXRTN= { 4 | nn }
```

HLMAXRTN

Specifies the maximum warning or error code received from the HLASM compiler, when compiling a copybook. Any code higher than specified stops the current Z Data Tools function. You can specify a value from 0 to 99. The default value is 4.

HLNOALIGN

```
HLNOALIGN= { NO | YES }
```

HLNOALIGN

Specifies if the **NOALIGN** HLASM compiler option is be specified when compiling assembler copybooks.

YES

The **NOALIGN** HLASM compiler option is specified.

NO

The **NOALIGN** HLASM compiler option is not specified.

HSYSLIB*nn*

```
HSYSLIBnn=dsname
```

HSYSLIB*nn*

Where *nn* is a number in the range 1 to 10, specifies the names of SYSLIB data sets which are searched in the order specified for COPY members for the HLASM compilation.

ISPFGLBL

```
Global_Disable_Pack = { NO  | YES }
```

ISPFGLBL

Specifies whether Z Data Tools considers the current global settings for ISPF or ignores them.

Global_Disable_Pack

Enabling the option, Z Data Tools will only consider the following ISPF global settings:

YES

The ISPF global settings should affect the Z Data Tools function.

NO (Default)

Z Data Tools ignores the settings.

Note: The settings are considered according to the ISPF standard module (ISPCFIGU) content.

ISPFPACK

```
ISPFPACK= { YES  | NO }
```

ISPFPACK

Specifies whether Z Data Tools is to check the data set or member to determine if it has been written with the ISPF PACK option.

YES

Z Data Tools is to check the data set or member to determine if it has been written with the ISPF PACK option. If so, Z Data Tools then unpacks the data to allow it to be edited or viewed in the normal fashion. If the data set is too large to be contained in memory, Z Data Tools is not able to edit or view the data set in unpacked form, but instead provides the data set in its packed form.

NO

Z Data Tools does not check the data and the editor operates on the record data regardless of the data being in ISPF pack form or not.

JCL

```
JCL = { COMPAT  | RECORD }
```

JCL

Defines the method by which JCL data is processed when running FCH or DSC with JCL=YES.

COMPAT

Z Data Tools reformats the physical JCL records into a single logical JCL statement for processing and then builds a new set of physical JCL records.

RECORD

Z Data Tools processes each physical JCL record as is, unless it contains a parameter field enclosed in apostrophes. If this parameter continues across multiple records, the records are flowed together to join the parameter before processing.

LANGUAGE

```
LANGUAGE= { ENGLISH | language }
```

LANGUAGE

Specifies the language for Z Data Tools batch message text. The possible values depend on which translated message texts you installed. For information on installing translated messages for the various Z Data Tools components, see the relevant chapter on customizing for national languages. See also the following notes.

ENGLISH

Messages will appear in English.

language

Messages will appear in the language of your choice.



Note:

1. See Table 18: Keyword values for the LANGUAGE option on page 108 to determine what value to use for *language* for your requirements.
2. Specifying a language here only affects batch processing. The language used by Z Data Tools under ISPF depends on the language setting for your ISPF session.

   However, ZDT/Db2 also uses the value of *language* to determine the data module to use when displaying Db2® catalog table information. See OP34MOD on page 451 for more information.

3. If you specify a language other than English, you must also customize the translation tables, the message modules, and (in some cases) the ISPF messages for your chosen language. See Customizing Z Data Tools for national languages on page 104, or Customizing ZDT/Db2 for national languages on page 211, or Customizing ZDT/IMS for national languages on page 310, or Customizing ZDT/CICS for national languages on page 368, for information about customizing for national languages.
4. Specifying LANGUAGE=*language*, causes the relevant language translate tables and message tables to be loaded for batch processing.
5. You must specify LANGUAGE=JAPANESE (or another DBCS-capable language) to print DBCS characters in batch.
6. You must specify LANGUAGE=JAPANESE (or another DBCS-capable language) for batch processing, if you want to use the FMT function, and field type of DB, to describe COBOL or PL/I Graphic fields.

LMS

```
LMS = { NO  |  PANVALET }
```

LMS

Specifies whether or not Z Data Tools will use COBOL copybooks, PL/I include books, or HLASM copybooks stored in CA-Panvalet libraries or other Library Management System libraries.

NO

Z Data Tools will not use any library management system libraries.

PANVALET

Z Data Tools will use copybooks or include books stored in CA-Panvalet libraries.

Note: You cannot specify LMS=PANVALET together with LMSUBSYS=xxxx, since you cannot access source from CA-Panvalet together with source from an LMS that provides a SUBSYS interface.

LMSUBSYS

```
LMSUBSYS=XXXX
```

LMSUBSYS

Specifies that Z Data Tools will, when accessing COBOL copybooks, PL/I include books, or HLASM copybooks, attempt the access via an I/O subsystem using the DFSMSdfp SUBSYS=xxxx allocation parameter.

This access is only available to copybooks and include books that exist in data sets controlled by the LMS. Data set organizations of BDAM, VSAM, and PS (sequential) are supported.

This facility is used to access copybooks or include books in OEM library management system data sets.

Specify *xxxx* as provided by your library management system vendor.

Note: You cannot specify LMSUBSYS=xxxx if you have specified LMS=PANVALET, since you cannot specify a combination of LMS and LMSUBSYS that equates to using CA-Panvalet with another LMS that provides a SUBSYS interface.

LOADLIB

```
LOADLIB = { HFM.SHFMMOD1 | dsname }
```

LOADLIB

The data set name of the Z Data Tools load library which is used in the HFMFTEXC, HFM1FTEX, and HFM2FTSL job control skeletons.

MQREPHLQ

```
MQREPHLQ= mqrep-hlq
```

MQREPHLQ

Specifies an installation-specific high-level qualifier to be used in the name pattern of the dynamic reply queue for Websphere MQ related functions. The specific queue name used is returned by the Queue Manager being used at the time.

mqrep-hlq

You can specify any valid qualifier, including multilevel if required, up to a total length of 38 characters. If you specify &&USER. for any part of the qualifier in your HFM0POPT macro, this is replaced with your user ID. If you specify &&PREFIX. for any part of the qualifier, this is replaced with your TSO prefix. Note that the current user ID is always appended to this high-level qualifier when requesting a dynamic queue.

Also note that the characters '#' (X'7C'), '@' (X'7B') and '$' (X'5B'), while allowed in user IDs, are not permitted in Websphere MQ. Although specifying them in this parameter is accepted, they are changed at runtime to the allowable MQ character '_'.

&&USER. and &&PREFIX. can also be concatenated with a character string. A multilevel qualifier with symbols follows the same rules as in the specification of a data set name with symbolic parameters in JCL procedures. Thus, a single period used after a symbol combines the symbol with the text following it (see Example 2 below). A period separating parts of a multilevel qualifier must be doubled (see Example 3 below).

Example 1

If the user ID is XXXX, specifying the value HFMTMQL. results in a dynamic name request for the pattern:

HFMTMQL.XXXX.*

Example 2

If the user ID is XX#XX and the TSO prefix is YYYY, then specifying &&USER.1.TEMP.&&PREFIX results in a dynamic name request for the pattern:

XX_XX1.TEMP.YYYY.XX_XX.*

Example 3

If the user ID is ZZZZ, then specifying HFM.&&USER..ABCD. results in a dynamic name request for the pattern:

HFM.ZZZZ.ABCD.ZZZZ.*

Note: Concatenation of &&USER.string, or &&PREFIX.string, could result in a part of the high-level qualifier being longer than eight characters. In this case, the string is shortened to the extent that the final part of the qualifier is eight characters long.

MSGUPPER

```
MSGUPPER= { NO | YES }
```

MSGUPPER

Specifies that all message text is to be translated to uppercase.

NO

No messages are translated to uppercase.

YES

All messages are translated to uppercase.

NOTRUNC

```
NOTRUNC= { NO | YES }
```

NOTRUNC

Specifies that, if the PAD option has been selected, no truncation is performed when copying or writing records to a variable-length data set.

NO

Truncation performed.

YES

No truncation performed.

OPSCRAM

```
OPSCRAM = { OFF | ON | MAX }
```

OPSCRAM

Specifies the optimization method used when scrambling data in fields using the data set copy function.

OFF

No optimization is performed.

ON

This option is designed to improve performance when scrambling large amounts of data where the Repeat or Random option has been specified for various fields.

MAX

This option is designed to provide the maximum performance improvement.



Note: If Repeat or Random has been specified for a field and a value list is not provided then duplicate alphanumeric characters within a field will yield the same scrambled character.

PAD

```
PAD={ OFF | ON | X'hh | c }
```

PAD

Specifies whether records are padded or truncated when being copied.

OFF

Records are not padded or truncated.

ON

Records are padded with blanks, or truncated.

***c* or X'*hh*'**

The same as PAD=ON except that the specified alphanumeric or hexadecimal character is used as the pad character instead of a blank.

PAGESIZE

```
PAGESIZE={ 60 | nnn }
```

PAGESIZE

Specifies the number of lines printed on each page of print output. You can specify a value from 1 to 999. The default is 60.

PAGESKIP

```
PAGESKIP={ NO | YES }
```

PAGESKIP

Specifies whether print output from each function begins on a new page.

NO

Print output from each function does not begin on a new page.

YES

Print output from each function begins on a new page.

PDATAC

```
PDATAC=XXXXXXXX
```

PDATAC

Specifies the default temporary SMS data class that Z Data Tools will use when allocating work data sets for internal use. PDATAC is optional. The maximum length you can specify for PDATAC is eight (8) bytes.

PLI31DIGIT

```
PLI31DIGIT={ NO | YES }
```

PLI31DIGIT

Specifies if the **LIMITS(FIXEDDEC(31))** PL/I compiler option is to be specified when compiling PL/I copybooks.

YES

The **LIMITS(FIXEDDEC(31))** PL/I compiler option is specified

NO

The **LIMITS(FIXEDDEC(31))** PL/I compiler option is not specified.

PLI63BIT

```
PLI63BIT={ NO | YES }
```

PLI63BIT

Specifies if the **LIMITS(FIXEDBIN(63))** PL/I compiler option is to be specified when compiling PL/I copybooks.

YES

The **LIMITS(FIXEDBIN(63))** PL/I compiler option is specified

NO

The **LIMITS(FIXEDBIN(63))** PL/I compiler option is not specified.

PLIGRAPHIC

```
PLIGRAPHIC={ NO | YES }
```

PLIGRAPHIC

Specifies if the **GRAPHIC** PL/I compiler option is to be specified when compiling PL/I copybooks.

YES

The **GRAPHIC** PL/I compiler option is specified.

NO

The **GRAPHIC** PL/I compiler option is not specified.

PLIMAXRTN

```
PLIMAXRTN= { 4 | nn }
```

PLIMAXRTN

Specifies the maximum warning or error code received from the PL/I compiler, when compiling a copybook. Any code higher than specified stops the current Z Data Tools function. You can specify a value from 0 to 99. The default value is 4.

PLIUNALIGN

```
PLIUNALIGN= { NO | YES }
```

PLIUNALIGN

Specifies if the statement **DEFAULT RANGE(*) UNALIGNED;** is to be added when compiling PL/I copybooks.

YES

The statement is to be added.

NO

The statement is not to be added.

PMGMTC

```
PMGMTC=XXXXXXXX
```

PMGMTC

Specifies the default permanent SMS management class that Z Data Tools will use when allocating permanent data sets, (for example, a print file). PMGMTC is optional. The maximum length you can specify for PMGMTC is eight (8) bytes.

PRINTDSN

```
PRINTDSN= { userid .HFM.LIST | dsn }
```

PRINTDSN

Specifies the name of the temporary print data set where Z Data Tools print output is directed when PRINTDSN is specified for the **PRINTOUT** option.

You can specify any valid name for this data set. If you specify &&USER. as a high-level qualifier for PRINTDSN, the data set name will have a high-level qualifier of your user ID. If you specify &&PREFIX. as the high-level qualifier for PRINTDSN, the data set name will have a high-level qualifier of your TSO PREFIX.

To view the contents of the print data set, use the PB command (Print Browse), or option 3.9. For information on Print Browse, see *"Printing from Z Data Tools"* in the *Z Data Tools User's Guide and Reference*.

PRINTLEN

```
PRINTLEN={ 132 | 80 }
```

PRINTLEN

Specifies the line length of printed output.

132

The line length of printed output is 132 characters, suitable for a line printer.

80

The line length of printed output is 80 characters, suitable for a terminal.

PRINTLEN is not applicable when the data is printed in TABL format. When data is printed in TABL format, the line length is determined from the number and size of the fields printed. If this length is greater than the record length specified for the data set, the print line is truncated. The maximum print line length is 32760, unless limited to 132 characters by setting WIDEPRT=NO. For information about WIDEPRT, see WIDEPRT on page 424.

PRINTOUT

```
PRINTOUT={ SYSPRINT | TERMINAL | PRINTDSN | REXX | SYSOUT=class }
```

PRINTOUT

Specifies where print output is sent, except for batch execution.

SYSPRINT

Send print output to the current SYSPRINT allocation.

PRINTDSN

Send print output to the temporary data set specified by the PRINTDSN option (see below).

TERMINAL

Send print output to the terminal.

REXX

Send print output to the REXX stem variable FILEM.

SYSOUT=*class*

Send print output to the temporary data set specified by the PRINTDSN option (see below) and direct output generated by the PRINT Primary command to the JES spool queue class *class*. (It is recommended that you do not use this parameter. Instead, use the PRINTDSN parameter together with the PRTCLASS.)

PRTCLASS

```
PRTCLASS= { A | class }
```

PRTCLASS

Specifies the class of the JES spool queue to be used when the PRINT primary command is issued.

PRTDATAC

```
PRTDATAC=XXXXXXXX
```

PRTDATAC

Specifies the SMS data class that Z Data Tools uses when allocating the Z Data Tools print data set. The maximum length you can specify for PRTDATAC is eight (8) bytes. PRTDATAC is optional. If omitted the SMS data class specified in PDATAC is used, if supplied.

PRTDISP

```
PRTDISP= { MOD | OLD }
```

PRTDISP

Specifies the disposition of a print data set.

MOD

Print output is appended to the print data set.

OLD

A print data set is cleared before each print operation, and print output is written from the beginning of the data set.

PRTMGMTC

```
PRTMGMTC=XXXXXXXX
```

PRTMGMTC

Specifies the default SMS management class that Z Data Tools uses when allocating the Z Data Tools print data set. The maximum length you can specify for PRTMGMTC is eight (8) bytes. PRTMGMTC is optional. If omitted the SMS management class specified in PMGMT is used, if supplied.

PRTPQTY

```
PRTPQTY= { 64 | nnnnnnnn }
```

PRTPQTY

Specifies the amount of DASD space to be used for primary space allocation of the Z Data Tools print data set. The range depends on the space unit specified and the DASD device type.

PRTSQTY

```
PRTSQTY= { 32 | nnnnnnnn }
```

PRTSQTY

Specifies the amount of DASD space to be used for secondary space allocation of the Z Data Tools print data set. The range depends on the space unit specified and the DASD device type.

PRTSTORC

```
PRTSTORC=XXXXXXXX
```

PRTSTORC

Specifies the default SMS storage class that Z Data Tools uses when allocating the Z Data Tools print data set. The maximum length you can specify for PRTSTORC is eight (8) bytes. PRTSTORC is optional. If omitted the SMS storage class specified in PSTORC is used, if supplied.

PRTSUNIT

```
PRTSUNIT= { CYL | REC | KB | MB | TRK }
```

PRTSUNIT

Specifies the unit of primary and secondary space to be allocated to the Z Data Tools print data set.

REC

Record of average size.

KB

Kilobyte, a kilobyte is 1024 bytes.

MB

Megabyte, a megabyte is 1048576 bytes.

TRK

Track of a direct access storage device (DASD).

CYL

Cylinder of a DASD.

PRTTRANS

```
PRTTRANS={ON | OFF}
```

PRTTRANS

Specifies how Z Data Tools should translate non-printable characters.

ON

Non-printable characters are translated to blanks using a translate table. This may make printing faster.

OFF

No translation is performed. Use PRTTRANS=OFF to support special print chains.

For information on altering the print translate table, see Changing the print and display translation tables on page 62 and Changing the print and display translation tables for languages other than English on page 104.

PRTUNIT

```
PRTUNIT=XXXXXXXX
```

PRTUNIT

Specifies the default permanent unit that Z Data Tools uses when allocating the Z Data Tools print data set. The maximum length you can specify for PRTUNIT is eight (8) bytes. PRTUNIT is optional. If omitted the SMS storage class specified in PUNIT is used.

PSTORC

```
PSTORC=XXXXXXXX
```

PSTORC

Specifies the default permanent SMS storage class that Z Data Tools will use when allocating permanent data sets, (for example, a print file). PSTORC is optional. The maximum length you can specify for PSTORC is eight (8) bytes.

PSYSLIB*nn*

```
PSYSLIBnn=dsname
```

PSYSLIB*nn*

Where *nn* is a number in the range 1 to 10, specifies the names of SYSLIB data sets which are searched in the order specified for %INCLUDE members for the PL/I compilation.

PUNIT

```
PUNIT= { SYSALLDA | XXXXXXXX }
```

PUNIT

Specifies the default permanent unit that Z Data Tools will use when allocating permanent data sets, (for example, a print file). The default is SYSALLDA. The maximum length you can specify for PUNIT is eight (8) bytes.

RECLIMIT

```
RECLIMIT= { (1, *) | (m,n ) }
```

RECLIMIT

For data printed in CHAR or LHEX print format, limits print output for each record (or OAM object).

(1,*)

The entire record (or block) is printed.

(*n*,*m*)

Print output is limited to the data from columns (bytes) *n* through *m*. An asterisk (*) specified for *m* indicates the end of the record.

Note: Changing the RECLIMIT option has no effect for ZDT/Db2, ZDT/IMS, or ZDT/CICS.

RLS

```
RLS = { YES  | NO }
```

RLS

Specifies whether VSAM RLS support is active or not.

NO

VSAM data sets are opened in non-RLS mode.

YES

VSAM data sets are opened in RLS mode if applicable.

Note: In the options module, if **RLS=YES** is specified on the HFM0POPI macro, Z Data Tools performs the following for the VSAM data set:

1. If IDCAMS specifies **RLS IN USE = YES** and **VSAM QUIESCED = NO**, then Z Data Tools opens the VSAM data set in RLS mode.
2. If IDCAMS specifies **RLS IN USE = YES** and **VSAM QUIESCED = YES**, then Z Data Tools opens the VSAM data set in non-RLS mode.
3. If IDCAMS specifies **RLS IN USE = NO**, then Z Data Tools opens the VSAM data set in non-RLS mode.

SEC

```
SEC={ NO | YES }
```

SEC

Specifies whether or not update functions are to be protected in Z Data Tools Base function or in ZDT/CICS.

NO

Update functions are not protected.

YES

Update functions can be protected.

Note: To protect update functions in Z Data Tools Base function set SEC=YES in HFM0POPT. To protect update functions in ZDT/Db2 set SEC=YES in HFM2POPT. To protect update functions in ZDT/CICS set SEC=YES in HFM3POPT. Setting this option in HFM1POPT has no effect in ZDT/IMS.

For more information about protecting update functions, see Unprotected functions and profile names for protected functions on page 76.

SHOWCOPY

```
SHOWCOPY={ NO | YES }
```

SHOWCOPY

Specifies whether the copybook name for a record layout is made visible during template edit and for an edit, view or browse session where a copybook or template is being used.

YES

The copybook name if applicable is visible.

NO

The copybook name is not visible. The default is NO.

SMFNO

```
SMFNO={ 0 | nnn }
```

SMFNO

Specifies the SMF record type for audit logging. You can specify a value from 128 to 255, or 0. The default is 0.

Note:

1. Specifying SMFNO=0 indicates that audit logging will not create SMF records.
2. Z Data Tools Base component only uses the value specified for this parameter, when the FMAUDIT parameter in the HFM0PARM member in the SYS1.PARMLIB (or its concatenations) specifies SAF_CTRL=NO.
3. ZDT/IMS only uses the value specified for this parameter, when the FMAUDIT parameter in the HFM1PARM member in the SYS1.PARMLIB (or its concatenations) specifies SAF_CTRL=NO.
4. ZDT/Db2 only uses the value specified for this parameter, when the FMAUDIT parameter in the HFM2PARM member in the SYS1.PARMLIB (or its concatenations) specifies SAF_CTRL=NO.
5. ZDT/CICS only uses the value specified for this parameter, when the FMAUDIT parameter in the HFM3PARM member in the SYS1.PARMLIB (or its concatenations) specifies SAF_CTRL=NO.
6. When the FMAUDIT parameter specifies SAF_CTRL=YES, audit logging is controlled by SAF and Z Data Tools ignores this parameter.

TAPELBL

```
TAPELBL={ SL | AL | AL4 }
```

TAPELBL

Specifies the type of tape labels that are created by the Initialize Tape function.

SL

EBCDIC labels are created.

AL

ANSI Version 3 labels are created. (*"Version 3"* refers to ANSI X3.27–1978, ISO 1001–1979, and FIPS 79 standards.)

AL4

ANSI Version 4 labels are created. (*"Version 4"* refers to ANSI X3.27–1987 level 4 and ISO 1001–1986(E) standards.)

This parameter only affects the Initialize Tape function. For information on Initialize Tape, see *"INT (Initialize Tape)"* in the *Z Data Tools User's Guide and Reference*.

Note: Changing the TAPELBL option has no effect for ZDT/Db2, ZDT/IMS or ZDT/CICS.

TDATAC

```
TDATAC=XXXXXXXX
```

TDATAC

Specifies the default temporary SMS data class that Z Data Tools will use when allocating work data sets for internal use. TDATAC is optional. The maximum length you can specify for TDATAC is eight (8) bytes.

TEMPHLQ

```
TEMPHLQ=temp_hlq
```

TEMPHLQ

Specifies an installation-specific high-level qualifier for the temporary data sets created during a Z Data Tools session.

Some Z Data Tools functions that create temporary data sets, for which the TEMPHLQ value is used if specified, are:

- ZDT/IMS processing the XKEY command
- ZDT/IMS creating the IMS™ LOG data set
- ZDT/IMS executing IXB (IMS™ Extract data in batch) function
- ZDT/IMS processing and building PSBs
- ZDT/Db2 editing and showing SQL statements

You can specify any valid qualifier, including multilevel if required, up to a total length of 24 characters. If you specify &&USER. for any part of the qualifier in your HFM0POPT macro, this is replaced with the user's user ID. If you specify &&PREFIX. for any part of the qualifier, this is replaced with the user's TSO prefix.

&&USER. and &&PREFIX. can also be concatenated with a character string. A multilevel qualifier with symbols follows the same rules as in the specification of a data set name with symbolic parameters in JCL procedures. Thus, a single period used after a symbol combines the symbol with the text following it (See Example 1 below); a period separating parts of a multilevel qualifier must be doubled (see Example 2 below).

Example 1

If the user ID is XXXX and the TSO prefix is YYYY, then specifying

```
&&USER.1.TEMP.&&PREFIX
```

will result in a high-level-qualifier of

```
XXXX1.TEMP.YYYY
```

Example 2

If the user ID is ZZZZ then specifying

```
HFM.&&USER..ABCD
```

will result in a high-level-qualifier of

```
HFM.ZZZZ.ABCD
```

Note:

1. The TEMPHLQ parameter can be up to 24 characters long. However, temporary data sets can have different lengths, and the system part of the data set name may be longer than 20 characters. In this case, minor levels of TEMPHLQ will be ignored, to the extent that the final data set name is less than or equal to 44 characters.
2. Concatenation of &&USER.*string*, or &&PREFIX.*string*, could result in a part of the high-level qualifier being longer than eight characters. In this case, the string will be shortened to the extent that the final part of the qualifier is eight characters long.

TERMTYPE

TERMTYPE= { **3270** | **3270KN** }

TERMTYPE

Specifies the terminal type.

3270

Specify 3270 for standard 3270 terminals.

3270KN

Specify 3270KN to support terminals that use Japanese Katakana characters or another DBCS language.

If 3270KN is specified, Z Data Tools translates message text to uppercase if you have LANGUAGE=ENGLISH, and translates panel text to uppercase regardless of the language.

TMGMTC

TMGMTC=*XXXXXXXX*

TMGMTC

Specifies the default temporary SMS management class that Z Data Tools will use when allocating work data sets for internal use. TMGMTC is optional. The maximum length you can specify for TMGMTC is eight (8) bytes.

TRACECLS

TRACECLS= { **A** | *class* }

TRACECLS

Specifies the class of the JES spool queue to be used when the PRINT primary command is issued when browsing the temporary trace data set.

TRACEDSN

```
TRACEDSN= { userid.HFM.TRACE | dsn }
```

TRACEDSN

The trace data set where output is directed when the TRACEOUT print option is TRACEDSN.

You can specify any valid name for this data set. If you specify &&USER as a high-level qualifier for TRACEDSN, the data set name will have a high-level qualifier of the user's user ID. If you specify &&PREFIX as the high-level qualifier for TRACEDSN, the data set name will have a high-level qualifier of the user's TSO PREFIX.

TRACELIM

```
TRACELIM= { 500 | nnn }
```

TRACELIM

Specifies the number of trace entries to be retained by the background trace mechanism. In the event of abnormal termination of a Z Data Tools function, or the DEBUG option is selected, these entries will be written to TRACEDSN. If this value is set to zero, the background trace mechanism is switched off. If a value is specified it must be zero, or in the range 500 to 9999.

TRACEOUT

```
TRACEOUT= { TRACEDSN | TERMINAL | HFMTRC }
```

TRACEOUT

Specifies where printed Z Data Tools debug trace output should be sent.

HFMTRC

Debug trace output is sent to the current HFMTRC allocation.

TERMINAL

Debug trace output is sent to the terminal.

TRACEDIN

Debug trace output is sent to the temporary trace data set specified by the TRACEDSN option (see above).

TRCDATAC

```
TRCDATAC=XXXXXXXX
```

TRCDATAC

Specifies the SMS data class that Z Data Tools uses when allocating the Z Data Tools Trace data set. The maximum length you can specify for TRCDATAC is eight (8) bytes. TRCDATAC is optional. If omitted the SMS data class specified in PDATAC is used, if supplied.

TRCMGMTC

```
TRCMGMTC=XXXXXXXX
```

TRCMGMTC

Specifies the default SMS management class that Z Data Tools uses when allocating the Z Data Tools Trace data set. The maximum length you can specify for TRCMGMTC is eight (8) bytes. TRCMGMTC is optional. If omitted the SMS management class specified in PMGMT is used, if supplied.

TRCPQTY

```
TRCPQTY= { 64 | nnnnnnnn }
```

TRCPQTY

Specifies the amount of DASD space to be used for primary space allocation of the Z Data Tools Trace data set. The range depends on the space unit specified and the DASD device type.

TRCSQTY

```
TRCSQTY= { 32 | nnnnnnnn }
```

TRCSQTY

Specifies the amount of DASD space to be used for secondary space allocation of the Z Data Tools Trace data set. The range depends on the space unit specified and the DASD device type.

TRCSTORC

```
TRCSTORC=XXXXXXXX
```

TRCSTORC

Specifies the default SMS storage class that Z Data Tools uses when allocating the Z Data Tools Trace data set. The maximum length you can specify for TRCSTORC is eight (8) bytes. TRCSTORC is optional. If omitted the SMS storage class specified in PSTORC is used, if supplied.

TRCSUNIT

```
TRCSUNIT= { CYL | REC | KB | MB | TRK }
```

TRCSUNIT

Specifies the unit of primary and secondary space to be allocated to the Z Data Tools Trace data set.

REC

Record of average size.

KB

Kilobyte, a kilobyte is 1024 bytes.

MB

Megabyte, a megabyte is 1048576 bytes.

TRK

Track of a direct access storage device (DASD).

CYL

Cylinder of a DASD.

TRCUNIT

```
TRCUNIT=XXXXXXXX
```

TRCUNIT

Specifies the default permanent unit that Z Data Tools uses when allocating the Z Data Tools Trace data set. The maximum length you can specify for TRCUNIT is eight (8) bytes. TRCUNIT is optional. If omitted the SMS storage class specified in PUNIT is used.

TSTORC

```
TSTORC=XXXXXXXX
```

TSTORC

Specifies the default temporary SMS storage class that Z Data Tools will use when allocating work data sets for internal use. TSTORC is optional. The maximum length you can specify for TSTORC is eight (8) bytes.

TUNIT

```
TUNIT= { SYSALLDA | XXXXXXXX }
```

TUNIT

Specifies the default temporary unit that Z Data Tools will use when allocating work data sets for internal use. The default is SYSALLDA. The maximum length you can specify for TUNIT is eight (8) bytes.

USEIOX

```
USEIOX= { DISABLE | ENABLE | (ENABLE,exit_name) }
```

USEIOX

Specifies whether or not Z Data Tools will load an I/O exit For information on providing a user I/O exit, see Customizing Z Data Tools to use an I/O exit on page 123.

DISABLE

The exit function is disabled. Z Data Tools will not use an I/O exit when processing data sets, and the user will not be able to specify an exit name on any panel. The default is DISABLE.

ENABLE

The exit function is enabled. The User I/O Exit selection field is displayed on the relevant panels, allowing the user to provide the name of a user-written exit, if required. Alternatively, the name of an exit can be specified in the system settings.

(ENABLE,*exit_name*)

The exit function is enabled and a default exit, *exit_name*, will be displayed in the User I/O Exit selection field, unless overwritten by the user. If a value is specified for *exit_name*, then *exit_name* should exist as a module in a load library available to Z Data Tools.

VSAUTO

```
VSAUTO= { 0 | nnnnnnnn }
```

VSAUTO

Specifies the frequency of periodic SAVEs when operating upon a VSAM shared file. If 0 is specified, Z Data Tools will not automatically issue a SAVE command. If a value is specified, Z Data Tools automatically issues a SAVE command for the file being operated upon after the given number of updates have occurred. You can specify a value from 0 to 99999999. The default value is 0.

VIEWAUX

```
VIEWAUX = { NO | YES }
```

VIEWAUX

Defines the default setting for the usage of an auxiliary data set when viewing a large non-VSAM data set that does not fit into memory.

YES

An auxiliary data set will be used.

NO

An auxiliary data set will not be used.

VSCHGAUTO

```
VSCHGAUTO= { NO | YES }
```

VSCHGAUTO

Specifies if during a CHANGE command with the ALL operand, when a nonzero figure has been used for the VSCHGFRQ option, this determines if an automatic retry of the change is to take place when a record in a VSAM shared file has been updated by another user before the Change process has saved the changed record.

YES

Z Data Tools automatically refreshes records that it had attempted to save and re-applies the change update to them. This is the most effective way of using the Change ALL command upon a shared and heavily updated file.

NO

Z Data Tools displays a Record Integrity Warning panel for each instance of a record becoming out of date. The user then has the choice to reapply the update, skip the update or abort the Change process at that point.

VSCHGFRQ

```
VSCHGFRQ= { 0 | nnnnnnnn }
```

VSCHGFRQ

Specifies the frequency of periodic SAVEs of the VSAM shared file during a CHANGE command with the ALL operand. If 0 is specified, Z Data Tools does not automatically issue a SAVE command. If a value is specified, Z Data Tools automatically issues a SAVE command for the file during a CHANGE command with the ALL operand after the given number of updates have occurred. You can specify a value from 0 to 99999999. The default value is 0.

VSSAVE

```
VSSAVE= { NO | YES }
```

VSSAVE

Specifies if when operating upon a VSAM shared file, during a CHANGE command with the ALL operand, a SAVE is issued before commencing the CHANGE.

NO

Z Data Tools does not issue a SAVE before commenting the CHANGE.

YES

Z Data Tools issues a SAVE command which ensures that any updates made to the file so far are saved. During the CHANGE ALL process, when using VSCHGAUTO (and a VSCHGFRQ value), the change process may need to refresh records as it goes (due to concurrent updates by other

users). That refresh may discard other updates that had been made in this edit session to records on the file. Leaving the setting at YES, to issue a SAVE before the CHANGE commences, is therefore recommended as it will ensure the state of the records that have been updated.

WBLKSIZE

```
WBLKSIZE={ 32760 | nnnnn }
```

WBLKSIZE

Specifies the blocksize of the to-be-allocated (new) print output data sets (online) or for SYSPRINT allocation (in batch) if the WIDEPRT=YES option is specified.

nnnnn

For the to-be-allocated (new) print output data or the SYSPRINT allocation (in batch), Z Data Tools uses the blocksize specified with a record format of VBA. The record size used is taken from the WLRECL option.

WIDEPRT

```
WIDEPRT={ YES | NO }
```

WIDEPRT

Specifies the record length and blocksize of new print output data sets.

YES

For to-be-allocated (new) print output data sets (online) or for SYSPRINT allocation (in batch), Z Data Tools will use the maximum record length/blocksize of 32756/32760 and record format of VBA. This is the default and is the same as previous behaviour.

The maximum record length/block size can be overwritten by the WBLKSIZE and WLRECL Z Data Tools options.

NO

For to-be-allocated (new) print output data sets (online) or for SYSPRINT allocation (in batch), Z Data Tools will use the record length/blocksize of 133/137 and record format of VBA.

If the user has pre-allocated the print output data set, as specified in the PRINTDSN parameter, this data set's record length and record format are not changed. Also note that once the print output data set has been allocated and opened by Z Data Tools, the data set's attributes cannot be changed again, for example, by using a SET command.

If WIDEPRT is set to NO, all printed output exceeding the current record length of the output data set will be truncated without any warning.

WLRECL

```
WLRECL={ 32756 | nnnnn }
```

WLRECL

Specifies the record size of the to-be-allocated (new) print output data sets (online) or for SYSPRINT allocation (in batch) if the WIDEPRT=YES option is specified.

nnnnn

For the to-be-allocated (new) print output data or the SYSPRINT allocation (in batch), Z Data Tools uses the record size specified with a record format of VBA. The blocksize used is taken from the WBLKSIZE option.

Appendix B. ZDT/Db2 options

This section describes the ZDT/Db2 options. The syntax described here applies to HFM2SSDM and HFM2POPI. You modify these ZDT/Db2 options to suit your requirements. See Changing the default options on page 180 for information on how to do this.

HFM2SSDM

In particular, before you can use ZDT/Db2, you must specify the Db2® subsystems, or Db2® data sharing groups, which ZDT/Db2 will use. To do this you must provide HFM2SSDM macro statements in HFM2POPT. For information about specifying Db2® subsystems and Db2® data sharing groups, and HFM2SSDM, see Defining all Db2 systems that ZDT/Db2 will access in HFM2POPT (required) on page 174.

You must code at least one HFM2SSDM macro entry in HFM2POPT. When there is only one entry it should specify SSID=DEFAULT (see the description of the SSID option). When there are multiple entries, the final HFM2SSDM macro entry must specify SSID=DEFAULT. If this entry is omitted an assembler error will occur.



Note: All library names specified in HFM2SSDM must be enclosed in quotation marks, and each set of library names must be further enclosed in parentheses. A maximum of sixteen (16) library names can be specified for one set of library names.

ATTACH

```
ATTACH= { CAF  | RRSAF }
```

ATTACH

Specifies which attachment facility is to be used to connect to and use the Db2® subsystem defined by SSID, to process SQL statements, commands or instrument facility interface (IFI) calls.

The properties and requirements of the various Db2® attachment facilities are documented in the *Db2® Application Programming and SQL Guide*.

CAF

The call attachment facility will be used to connect to and use the Db2® subsystem defined by SSID.

CAF is the default and recommended value. The CAF attachment works concurrently with other Db2® applications that use the DSNELI (DSN) attachment, the DSNALI (CAF) attachment, but not the DSNRLI (RRSAF) attachment.

RRSAF

The Db2® Resource Recovery Services attachment facility (RRSAF) will be used to connect to and use the Db2® subsystem defined by SSID.

RRSAF uses z/OS® Transaction Management and Recoverable Resource Manager Services (z/OS® RRS) and this service must be active.

RRSAF should only be used in situations where there will be no conflicts with other Db2® applications that use other attachment facilities, especially the DSNELI attachment. For example, it is not possible to run Db2's SPUFI and ZDT/Db2 on separate ISPF split screens when ZDT/Db2 is using the RRSAF attachment (because of the conflict with the DSN attachment used by Db2's SPUFI).

The default is CAF.

AUDIT

```
AUDIT={(OPTIONAL[,SMF]{,PROFILE|,OFF}) |NONE|(REQUIRED[,SMF])|(DEMAND[,SMF])}
```

AUDIT

Specifies the audit logging requirements for the Db2® subsystem defined by SSID. Note that this option is ignored when SAF-rule controlled auditing is active.

NONE

Disregards the setting of `Create an audit trail` in the editor options and no audit trail is produced.

OPTIONAL

If `Create an audit trail` in the editor options is selected, an audit trail is produced. If `Create an audit trail` is not selected, no audit trail is produced. The default is **OPTIONAL**.

(OPTIONAL,SMF)

If the suboperand SMF is specified, then if an audit trail is to be produced, it will be written to SMF. You can also optionally specify the suboperands **PROFILE** or **OFF**. In this case, the default is **PROFILE**.

When **PROFILE** is specified, ZDT/Db2 stores the current value of the `Create an audit trail` setting from the global options in the ISPF profile at the end of every ZDT/Db2 session.

When **OFF** is specified, ZDT/Db2 ignores any value for the `Create an audit trail` setting stored in the user's ISPF profile. At the start of every ZDT/Db2 session the `Create an Audit trail` option is reset to **OFF**. Changes made to the option using either the global options, or local edit options last for the life of the current ZDT/Db2 session only. This option may be useful when audit data is written to data sets, rather than SMF, and taking an audit trail is an exceptional, rather than normal activity, as it will minimize the number of audit data sets produced.

REQUIRED

An audit trail is produced, regardless of the setting of `Create an audit trail` in the editor options. If **AUDIT=(REQUIRED,SMF)** is specified, then the audit trail is written to SMF.

DEMAND

An audit trail is produced, regardless of the setting of `Create an audit trail` in the editor options. An audit report job will be submitted at the conclusion of an edit function. If **AUDIT=(DEMAND,SMF)** is specified, then the audit trail is also written to SMF.

Note:

1. If you specify SMF, you must also specify a value for the SMFNO option in HFM2POPT.
2. The AUDIT option is active only when data is modified via the editor (view, browse or edit) or copy utility. The setting of this option will be ignored for any other Z Data Tools function that allows data to be modified, and for data modified by the ISPF editor, even though this is from within Z Data Tools.

   Auditing can be selectively disabled when using ZDT/Db2 view or browse by specifying AUDITBROWSE=N in the HFM2SSDM macro definition for a Db2® system. Specifying this option does not affect the audit records produced when using ZDT/Db2 edit. See AUDITBROWSE on page 428 for information about AUDITBROWSE.

AUDITBROWSE

```
AUDITBROWSE={Y | N}
```

AUDITBROWSE

Specifies whether or not audit log records are to be written for ZDT/Db2 view or browse sessions. Note that this option is ignored when SAF-rule controlled auditing is active.

Y

Specifies that audit log records are to be written for ZDT/Db2 view or browse sessions. The default is AUDITBROWSE=Y.

N

Audit log records are not written for ZDT/Db2 view or browse sessions.

Specify AUDITBROWSE=N if audit records are not required for users accessing Db2® data using ZDT/Db2 view or browse.

AUTH_ACCESS

```
AUTH_ACCESS={Y | N}
```

AUDITBROWSE

Specifies whether or not access to the *AUTH Db2® catalog tables is available to ZDT/Db2 users.

Y

Specifies that SELECT access to the *AUTH Db2® catalog tables is available.

This is the default. Specify Y if ZDT/Db2 users are to have access to the ZDT/Db2 Privileges utility, and are able to view, and possibly change, information the *AUTH Db2® catalog tables.

N

Specifies that no SELECT access to the *AUTH Db2® catalog tables is available.

Specify N in environments where the information in the *AUTH is considered confidential. Specify AUTH_ACCESS=N when you do not grant SELECT access on the *AUTH Db2® catalog tables to ZDT/Db2 users.

The ZDT/Db2 Privileges utility is disabled when N is specified, along with line commands (G and P) that may be issued against a list of Db2® objects in the ZDT/Db2 Object List utility.

AUTO_COMMIT

```
AUTO_COMMIT = { 0 | n }
```

AUTO_COMMIT

Specifies a value for the editor **Auto commit (Changes)** option that is to apply to all users accessing the Db2® subsystem specified in the SSID parameter. Allowable values are in the range 0 - 10000.

The default is 0, meaning that every ZDT/Db2 editor user can change the value for the **Auto commit (Changes)** option using the **Editor Options (6 of 8)** panel. Allowable values are in the range 0 - 2147483647.

Specify a positive value (*n*) to:

- Set the **Auto commit (Changes)** value for every ZDT/Db2 editor user to the specified value. The value appears on the **Editor Options (6 of 8)** panel as a protected field.
- Remove the ability of each ZDT/Db2 editor user to change the value for the **Auto commit (Changes)** option.

CCSIDWARNIGNORE

```
CCSIDWARNIGNORE= [ ( (plan_ccsid,terminal_ccsid) ) ]
```

CCSIDWARNIGNORE

Controls whether the CCSID Warning message will be displayed when ZDT/Db2 connects to a Db2® subsystem, if the CCSID of the terminal differs from the CCSID of the plan. ZDT/Db2 connects to a Db2® subsystem when the application is first entered or when the Db2® SSID is changed.

This option is used in conjunction with the CCSIDWARN option in the HFM2POPI macro.

An option is provided to allow you to stop the displaying of the CCSID Warning message when there is a mismatch between the terminal CCSID and the SSID plan CCSID.

Each parameter contains two CCSIDS. These are the CCSID of the plan and the CCSID of the terminal. These CCSIDs are separated by a comma and the whole parameter is enclosed in parentheses.

Up to five pairs of CCSIDs may be specified. If more than one pair is specified the parameters must be separated by a comma and the parameters must be enclosed in an outer pair of parentheses.

If you do not specify CCSIDWARNIGNORE in your HFM2SSDM macro, the default is no parameters.

CPYCPYN

```
CPYCPYN=cc_name
```

CPYCPYN

Specifies the template name to be used in a COPYDDN clause which is added to the COPY utility statement. *cc_name* can be up to eight characters in length. The CPYCPYN keyword is optional.

DB2ALIB

```
DB2ALIB = ( 'load_lib' )
```

DB2ALIB

Optionally specifies the Db2 Administration Tool® load libraries to be used to externally support the ZDT/Db2 GEN command.

The Db2 Administration Tool product must be installed and configured for the specified Db2® subsystem (SSID). If DB2ALIB is specified, the PLAN2 on page 439 option must also specify the plan name used by the Db2 Administration Tool.

If DB2ALIB is not specified, the ZDT/Db2 GEN command uses its default internal support.

DB2CLIB

```
DB2CLIB=( 'clist_lib' )
```

DB2CLIB

Specifies the Db2® CLIST libraries for the Db2® subsystem defined by SSID.

You only need to specify a value for DB2CLIB if the libraries have not been allocated prior to starting ZDT/Db2. You can do this in several ways, for example, in the TSO logon procedure as JCL statements, or in an initialization clist or exec run as part of the TSO logon procedure, or in a clist or exec executed prior to the initialization of ZDT/Db2.

DB2ELIB

```
DB2ELIB=( 'rexx_lib' )
```

DB2ELIB

Specifies the Db2® REXX libraries for the Db2® subsystem defined by SSID.

You only need to specify a value for DB2ELIB if the libraries have not been allocated prior to starting ZDT/Db2. You can do this in several ways, for example, in the TSO logon procedure as JCL statements, or in an initialization clist or exec run as part of the TSO logon procedure, or in a clist or exec executed prior to the initialization of ZDT/Db2.

DB2LLIB

```
DB2LLIB=( 'load_lib' )
```

DB2LLIB

Specifies the Db2® load libraries for the Db2® subsystem defined by SSID. All data sets defined to the DB2LLIB keyword must be APF-authorized.

You only need to specify a value for DB2LLIB if:

- The Db2® load libraries have not been added to the LINKLIST, and the libraries have not been allocated as part of the TSO logon procedure, or any initialization clist or exec run prior to starting ZDT/Db2.

  or:

- Db2® load libraries have been added to the LINKLIST, but the libraries in the LINKLIST are for a different version or release of Db2® to the Db2® system defined by this HFM2SSDM macro.

Note:

1. All Db2® load libraries, however specified to ZDT/Db2, must be APF-authorized.
2. If you code Db2® load library names in the HFM2SSDM macro entries for a Db2® system, ZDT/Db2 will use an ISPF LIBDEF ISPLLIB service to allocate the Db2® load libraries. This requires the user to have either READ or EXECUTE access to the Db2® load libraries. When EXECUTE access only is given, the environment must be a 'controlled environment' in RACF® terms, and you may need to write additional security rules. For information about establishing and using a RACF® 'controlled environment', see the *z/OS Security Server RACF Security Administrator's Guide*, or equivalent documentation for your security product, if you do not use RACF®.

See Alternatives for making ZDT/Db2 available on page 155 for information about the LINKLIST and TSO logon procedures for ZDT/Db2.

DB2MLIB

```
DB2MLIB=( 'msg_lib' )
```

DB2MLIB

Specifies the Db2® message libraries for the Db2® subsystem defined by SSID.

You only need to specify a value for DB2MLIB if the libraries have not been allocated prior to starting ZDT/Db2. You can do this in several ways, for example, in the TSO logon procedure as JCL statements, or in an initialization clist or exec run as part of the TSO logon procedure, or in a clist or exec executed prior to the initialization of ZDT/Db2.

DB2PLIB

```
DB2PLIB=( 'pnl_lib' )
```

DB2PLIB

Specifies the Db2® panel libraries for the Db2® subsystem defined by SSID.

You only need to specify a value for DB2PLIB if the libraries have not been allocated prior to starting ZDT/Db2. You can do this in several ways, for example, in the TSO logon procedure as JCL statements, or in an initialization clist or exec run as part of the TSO logon procedure, or in a clist or exec executed prior to the initialization of ZDT/Db2.

DB2PROC

```
DB2PROC=( 'proc_lib' )
```

DB2PROC

Specifies the Db2® procedure libraries for the Db2® subsystem defined by SSID.

You only need to specify a value for DB2PROC for a Db2® subsystem if the standard Db2® procedures (for example, DSNUPROC) are not stored in a system procedure library. Specifying a DB2PROC statement will mean that any batch JCL generated by ZDT/Db2 will include a JCLLIB statement, when ZDT/Db2 is connected to that subsystem.

DB2RLIB

```
DB2RLIB=( 'run_lib' )
```

DB2RLIB

Specifies the Db2® run libraries for the Db2® subsystem defined by SSID.

You only need to specify a value for DB2RLIB if:

- The Db2® run libraries have not been added to the LINKLIST, and the libraries have not been allocated as part of the TSO logon procedure, or any initialization clist or exec run prior to starting ZDT/Db2.

  or:

- Db2® run libraries have been added to the LINKLIST, but the libraries in the LINKLIST are for a different version or release of Db2® to the Db2® system defined in this HFM2SSDM macro.

Note: If you add your Db2® run libraries to the same TSO logon procedure, or initialization clist or exec, as your Db2® load libraries, then the Db2® run libraries must also be APF-authorized.

See Alternatives for making ZDT/Db2 available on page 155 for information about the LINKLIST and TSO logon procedures for ZDT/Db2.

DB2SLIB

```
DB2SLIB=( 'skel_lib' )
```

DB2SLIB

Specifies the Db2® skeleton libraries for the Db2® subsystem defined by SSID.

You only need to specify a value for DB2SLIB if the libraries have not been allocated prior to starting ZDT/Db2. You can do this in several ways, for example, in the TSO logon procedure as JCL statements, or in an initialization clist or exec run as part of the TSO logon procedure, or in a clist or exec executed prior to the initialization of ZDT/Db2.

DB2TLIB

```
DB2TLIB=( 'tbl_lib' )
```

DB2TLIB

Specifies the Db2® table libraries for the Db2® subsystem defined by SSID.

You only need to specify a value for DB2TLIB if the libraries have not been allocated prior to starting ZDT/Db2. You can do this in several ways, for example, in the TSO logon procedure as JCL statements, or in an initialization clist or exec run as part of the TSO logon procedure, or in a clist or exec executed prior to the initialization of ZDT/Db2.

DESC

```
DESC='description'
```

DESC

Is the description of the subsystem that will appear in the **Db2 Subsystem Selection** panel for this SSID. The maximum length of the description is 45 characters, enclosed in quotation marks.

DISPLAY

```
DISPLAY={ YES  | UNAVAIL  | HIDDEN }
```

DISPLAY

Controls the display of the SSID on the **Db2 Subsystem Selection** panel.

YES

The subsystem or group ID (SSID) will be displayed on the subsystem selection list. Inactive subsystems will not appear if **LIST=ACTIVE** is specified in the HFM2POPI macro. The default is YES.

UNAVAIL

Specify **DISPLAY=UNAVAIL** if ZDT/Db2 has not been installed on a subsystem or Db2® data sharing group, or you require that users will not be able to connect to the subsystem or group.

SSIDs specified UNAVAIL will appear on the subsystem selection list if **LIST=ALL** is specified in the HFM2POPI macro. The SSID appears with a status of UNAVAIL. and cannot be selected.

Unavailable SSIDs will not appear on the selection list if **LIST=ACTIVE** is specified in the HFM2POPI macro. However, ZDT/Db2 will attempt to connect to subsystems marked as UNAVAIL if **CONNECT=ANY** is specified in the HFM2POPI macro.

HIDDEN

Specify **DISPLAY=HIDDEN** when you require that an SSID should not appear on the subsystem selection list, and that users should not be able to connect to it.

Hidden subsystems will not appear on the subsystem selection list, regardless of the setting of the **LIST** option in the HFM2POPI macro. However, ZDT/Db2 will attempt to connect to "hidden" subsystems if **CONNECT=ANY** is specified in the HFM2POPI macro.

EDIT_MAX_ROWS

```
EDIT_MAX_ROWS = { 0 | n }
```

EDIT_MAX_ROWS

Controls the maximum number of rows that can be loaded into any ZDT/Db2 editor session, and can be used to disable "large" table support.

The default is 0, meaning that:

- "Large" editor mode is available to all ZDT/Db2 users. If a Z Data Tools/Db2 user specifies row count = 0 for an editor-related function, ZDT/Db2 may use a scrollable cursor to access Db2® data. This might require Db2® to build a copy of the result table in a Db2® temporary database.
- "Normal" editor mode is available to all ZDT/Db2 users. An ZDT/Db2 user can specify row count = *n*, *n*>0 for an editor-related function, ZDT/Db2 attempts to load up to *n* rows into the ZDT/Db2 editor session. There is no restriction on the value *n* that can be specified up to a maximum of 2G-1.

When the PROD_EDIT option is set to `YES`, the default value is enforced as 1000, which disables support for Large editor mode. See topic PROD_EDIT on page 439.

Specify a positive value (*nn*) to:

- Disable "large" editor mode. If a Z Data Tools/Db2 user specifies row count = 0 for an editor-related function, ZDT/Db2 ignores the 0 value, the resulting ZDT/Db2 editor session is a "normal" edit session and ZDT/Db2 loads at most *nn* rows.
- Set an absolute limit on the number of rows that any ZDT/Db2 can process in a Z Data Tools/Db2 editor session. If a Z Data Tools/Db2 user specifies row count = *m*, *m*>*nn* on a function entry panel, ZDT/Db2 ignores the *m* value and loads at most *nn* rows into the editor.

Tip:

- Specify a large value (for example, 2,000,000,000) to disable "large" editor mode, while setting no reasonable upper limit to the number of rows that can be loaded into a Z Data Tools/Db2 editor session.
- Specify a modest value (for example, 10,000) to limit TSO memory usage when using the ZDT/Db2 editor.

EDITOR_TIMEOUT

{ **EDITOR_TIMEOUT= 0** | **EDITOR_TIMEOUT=***n* }

EDITOR_TIMEOUT

Specifies whether ZDT/Db2 editor sessions are subject to an inactivity time-out, and the length of time that must elapse before the inactivity time-out is triggered.

Specify 0 to disable the inactivity time-out. This is the default behaviour when the PROD_EDIT option is not set or is set to `NO`. Otherwise, the value is enforced as `60` seconds. See topic PROD_EDIT on page 439.

Specify a non-zero value to enable the inactivity timer. **n** is integer, **n** > 0. The value specified is interpreted as HHMMSS. Here are some examples:

```
n=30        specifies an inactivity time-out interval of 30 seconds
n=100       specifies an inactivity time-out interval of 1 minute
n=10000     specifies an inactivity time-out interval of 1 hour
n=718191    specifies an inactivity time-out interval
                 of 71 hours, 81 minutes and 91 seconds
```

The inactivity time-out applies to the locally connected Db2® system. For example, suppose the HFM2POPT has these entries:

```
HFM2SSDM    SSID=DSNA,
            EDITOR_TIMEOUT=0
HFM2SSDM    SSID=DSNB,
            EDITOR_TIMEOUT=45
```

An ZDT/Db2 user connected locally to DSNA is not subject to the editor inactivity time-out. If the ZDT/Db2 user specifies a location value to access a Db2® object at DSNB, the resulting editor session IS NOT subject to the inactivity time-out.

An ZDT/Db2 used connected locally to DSNB is subject to the editor inactivity time-out. If the ZDT/Db2 user specifies a location value to access a Db2® object at DSNA, the resulting editor session **is** subject to the inactivity time-out.

FORCE_WITH_UR

```
FORCE_WITH_UR={N | Y}
```

FORCE_WITH_UR

Specifies whether ZDT/Db2 should always append *"WITH UR"* to any SELECT statement processed by the ZDT/Db2 editor. The default is N; ZDT/Db2 does not add the *"WITH UR"* clause to SELECT statements processed by the ZDT/Db2 editor.

Specify Y when there is a requirement to reduce the number of Db2® locks taken as a result of the use of the ZDT/Db2 editor.

Y

When Y is specified there are two effects:

- The Concurrency and Keep Locks editor options, which appear on the seventh editor options panel, are not displayed, and all ZDT/Db2 users are unable to specify either option.
- The *"WITH UR"* clause is automatically added to any SELECT statement processed by the Z Data Tools Db2® editor. This includes the Browse (B), View (1), Edit (2), Basic Prototyper (4.1), Advanced Prototyper (4.2), Enter, execute and explain SQL statements (4.3) and Edit and execute SQL statements from a data set (4.4).

N

When N is specified:

- The Concurrency and Keep Locks editor options, which appear on the seventh editor options panel, are displayed, and all ZDT/Db2 users are able to change either option to their preference. The values selected by each user are applied to ZDT/Db2 editor sessions arising from the Browse (B), View (1) and Edit (2) functions.
- Select statements entered using the following functions are not affected:
  - Basic Prototyper (4.1)
  - Advanced Prototyper (4.2)
  - Enter, execute and explain SQL statements (4.3)
  - Edit and execute SQL statements from a data set (4.4)

  ZDT/Db2 processes the SELECT statement from these functions without any additions.

LDFDDLN

```
LDFDDLN=ld_name
```

LDFDDLN

Specifies the DDNAME to be used in JCL to reference the site-specific listdef library. *ld_name* can be up to eight characters, and must conform to standard job control conventions. This library contains the listdef statements that establish the default site policy.

Specify a value for this keyword if you want to implement a site-specific policy for listdefs.

The LDFDDLN keyword is optional. If no value is specified for LDFDDLN then no installation value will be used as a default. The ZDT/Db2 default (SYSLISTD) will be used.

If you specify a value for LDFDDLN, also specify a value for at least SLDJCL1. Using the SLDJCL1 - SLDJCL4 keywords, you can build a DDNAME concatenation that comprises up to four libraries. See SLDJCL1 on page 441, SLDJCL2 on page 441, SLDJCL3 on page 441, and SLDJCL4 on page 441.

For an example, see Examples of HFM2SSDM macros on page 175.

LOCATION

```
LOCATION=location_name
```

LOCATION

Specifies the Db2® location name for the Db2® system or group. This is the LOCATION value entered in the DDF statement (Db2® change log inventory utility), when Db2® was installed. Alternatively you can specify the value returned by the following SQL statement:

```
SELECT CURRENT SERVER FROM SYSIBM.SYSDUMMY1.
```

The maximum length you can specify for *location_name* is 16 characters, and must conform to the naming conventions for a "location-name" as documented in the SQL Reference manual for your Db2® system.

Note: The LOCATION option is optional. ZDT/Db2 does not require the specification of the location name in the HFM2SSDM macro for successful access to remote Db2® systems. Specifying a Db2® location name enables the use of the Db2® subsystem or group identifier (for example, DSNA) as an alternative to the actual Db2® location name, for online functions only.

LOCATION_NICKNAME

```
LOCATION_NICKNAME=nickname
```

LOCATION_NICKNAME

Specifies a nickname for the Db2® system or group. This can be any value conforming to the rules for an SQL ordinary identifier. Examples include DEV, TEST, PROD, LA, NY.

The maximum length you can specify for *nickname* is 16 characters.

Note: The LOCATION_NICKNAME option is optional. If you specify a value for *nickname* for the Db2® system or group, you must also specify a value for the LOCATION option. Specifying a Db2® location nickname enables the use of that nickname (for example, DEV) as an alternative to the actual Db2® location name, for online functions only.

LODINDN

```
LODINDN=li_name
```

LODINDN

Specifies the template name to be used in an INDDN clause which is added to the LOAD utility statement. *li_name* can be up to eight characters in length. The LODINDN keyword is optional.

OPTEVT1

```
OPTEVT1=ev_line1
```

OPTEVT1

Specifies the first line to be placed in the EVENT clause of the OPTION utility statement. *ev_line1* can be up to 54 characters in length. If your clause includes one or more commas, enclose the clause within single quotation marks. The OPTEVT1 keyword is optional.

For an example, see Examples of HFM2SSDM macros on page 175.

OPTEVT2

```
OPTEVT2=ev_line2
```

OPTEVT2

Specifies the second line to be placed in the EVENT clause of the OPTION utility statement. *ev_line2* can be up to 54 characters in length. If your clause includes one or more commas, enclose the clause within single quotation marks. The OPTEVT2 keyword is optional.

OPTEVT3

```
OPTEVT3=ev_line3
```

OPTEVT3

Specifies the third line to be placed in the EVENT clause of the OPTION utility statement. *ev_line3* can be up to 54 characters in length. If your clause includes one or more commas, enclose the clause within single quotation marks. The OPTEVT3 keyword is optional.

OPTEVT4

```
OPTEVT4=ev_line4
```

OPTEVT4

Specifies the fourth line to be placed in the EVENT clause of the OPTION utility statement. *ev_line4* can be up to 54 characters in length. If your clause includes one or more commas, enclose the clause within single quotation marks. The OPTEVT4 keyword is optional.

PLAN

```
PLAN= { HFM2PLAN | plan_name }
```

PLAN

Specifies the name of the ZDT/Db2 plan for the Db2® subsystem defined by SSID. The default plan name is HFM2PLAN.

If you change HFM2PLAN you must make the same change in the sample jobs, HFM2BN*n*P. See Binding Db2 (required) on page 172.

PLAN2

```
PLAN2 = { HFM2GEN | rev_plan_name }
```

PLAN2

Specifies the name of the Db2® reverse engineering plan for the Db2® subsystem defined by SSID. The default reverse engineering plan name is HFM2GEN.

If you change HFM2GEN, you must make the same change in the sample jobs, HFM2BN*n*P. See Binding Db2 (required) on page 172.

If external Db2 Administration Tool® is installed and configured defined using the DB2ALIB on page 430 option for runtime libraries, the specified plan is the execution plan defined specifically for the product usage.

PROD_EDIT

```
{ PROD_EDIT=N | PROD_EDIT=Y }
```

PROD_EDIT

Specifies whether various restrictions should apply to any ZDT/Db2 editor sessions. These restrictions are intended to reduce the potential for locking and resource contention when accessing Db2® data. Collectively these restrictions are referred to as the *"PROD_EDIT"* (Production Edit) behavior.

Specify PROD_EDIT=N to disable the PROD_EDIT restrictions. This is the default.

Specify PROD_EDIT=Y to enable the PROD_EDIT restrictions. Specifying PROD_EDIT=Y has multiple effects, as described below:

- An inactivity time-out applies to every ZDT/Db2 editor session. If EDITOR_TIMEOUT=0 is also specified in the same HFM2SSDM macro invocation it is replaced with EDITOR_TIMEOUT=60. You can specify any value for EDIT_TIMEOUT except 0, and that value will not be over-ridden.
- *"Large"* editor mode is disabled for all users. If EDIT_MAX_ROWS=0 is also specified in the same HFM2SSDM macro invocation it is replaced with EDIT_MAX_ROWS=1000. You can specify any value for EDIT_MAX_ROWS except 0, and that value will not be over-ridden.
- Table locking is disabled for all users. If TABLE_LOCKING=YES is also specified in the same HFM2SSDM macro invocation it is replaced with TABLE_LOCKING=NO.
- The editor option *"Commit after data fetch"* is fixed and is selected for all ZDT/Db2 users.
- The editor option *"Commit when save issued"* is fixed and is selected for all ZDT/Db2 users.
- The editor option *"Commit when no save errors"* option is fixed and is not selected for all ZDT/Db2 users.

The PROD_EDIT behavior applies to the locally connected Db2® system.

For example, Suppose the HFM2POPT has these entries:

```
HFM2SSDM    SSID=DSNA,
            PROD_EDIT=N
HFM2SSDM    SSID=DSNB,
            PROD_EDIT=Y
```

An ZDT/Db2 user connected locally to DSNA is not subject to the PROD_EDIT behavior. If the ZDT/Db2 user specifies a location value to access a Db2® object at DSNB, the resulting editor session **is not** subject to the PROD_EDIT behavior.

An ZDT/Db2 user connected locally to DSNB is subject to the PROD_EDIT behavior. If the ZDT/Db2 user specifies a location value to access a Db2® object at DSNA, the resulting editor session IS subject to the PROD_EDIT behavior.

Note:

When enabled, this option minimizes conflicts with concurrent tasks by limiting the number and duration of locks held on resources. The option does not affect the connection or its associated Db2 thread. The connection and Db2 thread remain active until the session terminates, either by exiting FM/Db2 or by switching to another Db2 subsystem.

RBXWRKN

RBXWRKN=*rxw_name*

RBXWRKN

Specifies the template name to be used in an WORKDDN clause which is added to the REBUILD INDEX utility statement. *rxw_name* can be up to eight characters in length. The RBXWRKN keyword is optional.

ROGUNLN

```
ROGUNLN=rgu_name
```

ROGUNLN

Specifies the template name to be used in an UNLDDN clause which is added to the REORG TABLESPACE utility statement. *rgu_name* can be up to eight characters in length. The ROGUNLN keyword is optional.

SLDJCL1

```
SLDJCL1=ld_line1
```

SLDJCL1

Specifies the first line of the generated DD statement for the site-specific listdef library concatenation, as specified on the LDFDDLN keyword. *ld_line1* can be up to 54 characters in length, and must conform to standard job control conventions. If your JCL statement includes one or more commas, then enclose the whole statement within single quotation marks. The data set so referenced must exist.

You can specify a value for SLDJCL1 even if you have not specified a value for LDFDDLN. however, if you have specified a value for LDFDDLN, then also specify a value for SLDJCL1.

SLDJCL2

```
SLDJCL2=ld_line2
```

SLDJCL2

Specifies the second line of the generated DD statement for the site-specific listdef library concatenation, as specified on the LDFDDLN keyword. *ld_line2* can be up to 54 characters in length, and must conform to standard job control conventions. If your JCL statement includes one or more commas, then enclose the whole statement within single quotation marks. The data set so referenced must exist. The SLDJCL2 keyword is optional.

SLDJCL3

```
SLDJCL3=ld_line3
```

SLDJCL3

Specifies the third line of the generated DD statement for the site-specific listdef library concatenation, as specified on the LDFDDLN keyword. *ld_line3* can be up to 54 characters in length, and must conform to standard job control conventions. If your JCL statement includes one or more commas, then enclose the whole statement within single quotation marks. The data set so referenced must exist. The SLDJCL3 keyword is optional.

SLDJCL4

```
SLDJCL4=ld_line4
```

SLDJCL4

Specifies the fourth line of the generated DD statement for the site-specific listdef library concatenation, as specified on the LDFDDLN keyword. *ld_line4* can be up to 54 characters in length, and must conform to standard job control conventions. If your JCL statement includes one or more commas, then enclose the whole statement within single quotation marks. The data set so referenced must exist. The SLDJCL4 keyword is optional.

SSID

```
SSID= { DEFAULT | ssid }
```

SSID

Defines the (up to) 4-character Db2® subsystem or group ID for the Db2® system that ZDT/Db2 will connect to. The 4-character value will appear on the ZDT/Db2 primary option menu when ZDT/Db2 is connected to the Db2® system or group.

You must specify at least one HFM2SSDM macro definition, and the last (or only) HFM2SSDM macro definition must specify SSID=DEFAULT. If you have already specified a site-specific value for SSID in your last HFM2SSDM macro, you must supply further HFM2SSDM macro in which the only statement is SSID=DEFAULT. See example **DG02S** in Examples of HFM2SSDM macros on page 175.

STMJCL1

```
STMJCL1=tm_line1
```

STMJCL1

Specifies the first line of the generated DD statement for the site-specific template library concatenation, as specified on the TMPDDLN keyword. *tm_line1* can be up to 54 characters in length, and must conform to standard job control conventions. If your JCL statement includes one or more commas, then enclose the whole statement within single quotation marks. The data set so referenced must exist.

You can specify a value for STMJCL1 even if you have not specified a value for TMPDDLN. However, if you have specified a value for TMPDDLN, then also specify a value for STMJCL1.

For an example, see Examples of HFM2SSDM macros on page 175.

STMJCL2

```
STMJCL2=tm_line2
```

STMJCL2

Specifies the second line of the generated DD statement for the site-specific template library concatenation, as specified on the TMPDDLN keyword. *tm_line2* can be up to 54 characters in length, and must conform to standard job control conventions. If your JCL statement includes one or more commas, then enclose the whole

statement within single quotation marks. The data set so referenced must exist. The STMJCL2 keyword is optional.

STMJCL3

```
STMJCL3=tm_line3
```

STMJCL3

Specifies the third line of the generated DD statement for the site-specific template library concatenation, as specified on the TMPDDLN keyword. *tm_line3* can be up to 54 characters in length, and must conform to standard job control conventions. If your JCL statement includes one or more commas, then enclose the whole statement within single quotation marks. The data set so referenced must exist. The STMJCL3 keyword is optional.

STMJCL4

```
STMJCL4=tm_line4
```

STMJCL4

Specifies the fourth line of the generated DD statement for the site-specific template library concatenation, as specified on the TMPDDLN keyword. *tm_line4* can be up to 54 characters in length, and must conform to standard job control conventions. If your JCL statement includes one or more commas, then enclose the whole statement within single quotation marks. The data set so referenced must exist. The STMJCL4 keyword is optional.

TABLE_LOCKING

```
{ TABLE_LOCKING= YES | TABLE_LOCKING=NO }
```

TABLE_LOCKING

Specifies whether a table being edited can be explicitly locked. An explicit LOCK TABLE SQL statement is only issued if the "locking" option on the seventh editor options panel is set to either 2 (Share Mode) or 3 (Exclusive Mode).

YES

An explicit LOCK TABLE SQL statement might be issued.

NO

An explicit LOCK TABLE SQL statement is not issued, regardless of the setting of the "locking" editor option.

If the PROD_EDIT option is not specified or is set to `NO`, the default behavior is `YES`. Otherwise, the value is enforced as `NO`. See topic PROD_EDIT on page 439.

TMPDDLN

```
TMPDDLN=tm_name
```

TMPDDLN

Specifies the DDNAME to be used in JCL to reference the site-specific template library. *tm_name* can be up to eight characters, and must conform to standard job control conventions. This library contains the template statements that establish the default site policy.

Specify a value for this keyword if you want to implement a site-specific policy for templates.

The TMPDDLN keyword is optional. If no value is specified for TMPDDLN then no installation value will be used as a default. The ZDT/Db2 default (SYSTEMPL) will be used.

If you specify a value for TMPDDLN, also specify a value for at least STMJCL1. Using the STMJCL1 - STMJCL4 keywords, you can build a DDNAME concatenation that comprises up to four libraries. See STMJCL1 on page 442, STMJCL2 on page 442,STMJCL3 on page 443, and STMJCL4 on page 443.

For an example, see Examples of HFM2SSDM macros on page 175.

TYPE

```
TYPE= { SUBSYS  | GROUP }
```

TYPE

Specifies the type of Db2® subsystem defined by SSID.

SUBSYS

Specifies that this SSID is the name of a Db2® subsystem defined in the IEFSSNxx member of SYS1.PARMLIB. The default is SUBSYS.

GROUP

Specifies that this SSID is a Db2® group attachment identifier in a Db2® data sharing environment.

UNLPUNN

```
UNLPUNN=up_name
```

UNLPUNN

Specifies the template name to be used in an PUNCHDDN clause which is added to the UNLOAD utility statement. *up_name* can be up to eight characters in length. The UNLPUNN keyword is optional.

UNLUNLN

```
UNLUNLN=uu_name
```

UNLUNLN

Specifies the template name to be used in an UNLDDN clause which is added to the UNLOAD utility statement. *uu_name* can be up to eight characters in length. The UNLUNLN keyword is optional.

USER_SELECT_EDIT

```
USER_SELECT_EDIT={Y | N}
```

USER_SELECT_EDIT

Specifies whether a user can select the Edit option for the *"Arbitrary SQL SELECT Statements"* system option.

Y

Specifies that a user may select the Edit option for the *"Arbitrary SQL SELECT Statements"* system option. The default is USER_SELECT_EDIT=Y.

N

Specifies that a user may not select the Edit option for the *"Arbitrary SQL SELECT Statements"* system option.

Specify USER_SELECT_EDIT=N when it considered necessary to prevent users accessing the ZDT/Db2 editor - in edit mode - when processing the result table for any user-specified SELECT statements. The ZDT/Db2 editor - in either browse or view mode - is still used to display the result table from any user-specified SELECT statement. Since browse and view modes do not allow changes to be saved to Db2®, users are unable to make changes to Db2® data.

Note that this setting is applied when connecting to the local Db2® system only. Consider the following example:

In the HFM2POPT,

- HFM2SSDM for Db2® system DSNA has USER_SELECT_EDIT=N, the CURRENT SERVER special register for DSNA is LOCN_A.
- HFM2SSDM for Db2® system DSNB has USER_SELECT_EDIT=Y, the CURRENT SERVER special register for DSNB is LOCN_B.

An ZDT/Db2 user locally connected to Db2® system DSNA can issue a statement like:

```
SELECT * FROM DSN81010.EMP,
```

but can only browse or view the result table.

The same user at DSNA can issue a statement like:

```
SELECT * FROM LOCN_B.DSN81010.EMP,
```

but can only view or browse the result table.

An ZDT/Db2 user locally connected to Db2® system DSNB can issue a statement like:

```
SELECT * FROM DSN81010.EMP,
```

and can browse, view or edit the result table.

The same user at DSNB can issue a statement like:

```
SELECT * FROM LOCN_A.DSN81010.EMP
```

and can browse, view or edit the result table.

HFM2POPI

CATOWNERCDRM

```
CATOWNERCDRM={ROUND_CEILING | ROUND_DOWN | ROUND_FLOOR | ROUND_HALF_DOWN | ROUND_HALF_EVEN | ROUND_HALF_UP | ROUND_UP}
```

CATOWNERCDRM

Specifies the current DECFLOAT rounding mode (*CDRM*) to be used when accessing a Db2® version 12 (or later) system.

This option is only relevant when ZDT/Db2 is connected to a Db2® version 9 or later system.

The CDRM refers to the behaviour of Db2® when processing a DECFLOAT number, where rounding of the value is required. See the Db2® manuals for a more detailed explanation.

If you intend to specify a CATOWNER parameter to implement access to the Db2® catalog tables (that is, ZDT/Db2 will access views of the Db2® catalog tables, rather than the catalog tables directly), you need to take account of the CDRM constraint as noted above. It is recommended that:

- You create all of the views of the Db2® catalog table at the same time, and set the required CDRM before creating the views.

  The CDRM can be set using:

  ```
  SET CURRENT DECFLOAT ROUNDING MODE = value
  ```

  See the Db2® SQL Reference manual for a detailed explanation.
- The value used for the CDRM should be specified in the CATOWNERCDRM parameter.
- In the situation where ZDT/Db2 will access multiple Db2® version 9 (or later) systems, use the same CDRM value when creating the catalog table views for all target Db2® systems. Specify the selected value in the CATOWNERCDRM parameter.

If the CATOWNERCDRM parameter is omitted, or is incorrectly specified, ZDT/Db2 will auto-detect the CDRM mode used when attempting to access a view of the Db2® catalog table. This is an expensive procedure, requiring ZDT/Db2 to prepare up to 7 SQL statements each time a user accesses the Db2® system. It is therefore advisable for performance reasons to ensure that the CATOWNERCDRM value is correctly specified.

ZDT/Db2 will not function correctly if the required views of the Db2® catalog tables are created using more than one CDRM value.

ROUND_CEILING

Round towards positive infinity. If all of the discarded digits are zero or if the sign is negative, the result is unchanged other than the removal of discarded digits. Otherwise, the result coefficient is incremented by 1 (round up).

ROUND_DOWN

Round towards 0 (truncation). The discarded digits are ignored.

ROUND_FLOOR

Round towards negative infinity. If all of the discarded digits are zero or if the sign is positive, the result is unchanged other than the removal of discarded digits. Otherwise, the sign is negative and the result coefficient is incremented by 1 (round down).

ROUND_HALF_DOWN

Round to nearest value; if values are equidistant, rounds down. If the discarded digits represent greater than half (0.5) of the value of a number in the next left position, the result coefficient is incremented by 1 (round up). Otherwise, the discarded digits are ignored. This rounding mode is not recommended when creating a portable application because it is not supported by the IEEE draft standard for floating-point arithmetic.

ROUND_HALF_EVEN

Round to nearest value; if values are equidistant, round so that the final digit is even. If the discarded digits represent greater than half (0.5) of the value of a number in the next left position, the result coefficient is incremented by 1 (round up). If the discarded digits represent less than half of the value, the result coefficient is not adjusted (that is, the discarded digits are ignored). Otherwise, the result coefficient is unaltered if its rightmost digit is even, or is incremented by 1 (round up) if its rightmost digit is odd (to make an even digit).

ROUND_HALF_UP

Round to nearest value; if values are equidistant, round up. If the discarded digits represent greater than or equal to half (0.5) of the value of a number in the next left position, the result coefficient is incremented by 1 (round up). Otherwise the discarded digits are ignored.

ROUND_UP

Round away from 0. If all of the discarded digits are zero, the result is unchanged other than the removal of discarded digits. Otherwise, the result coefficient is incremented by 1 (round up).

Important information about CDRM

The CDRM becomes critical when processing **any** Db2® view in a Db2® version 9 or later system. This is because Db2® records the CDRM in effect when the view was created in the Db2® catalog tables. When the view is subsequently accessed, Db2® checks the CDRM when the view was created against the CDRM of the process attempting to access the view. If the two values do not match, Db2® disallows the access and issues SQLCODE-270. The CDRM for the process must be changed to be the same as the CDRM value when the view was originally created, in order for the view access to succeed.

This restriction applies to every view created in a Db2® version 12 system (or later), regardless of whether or not there are DECFLOAT columns defined in the view.

CATOWNER

```
CATOWNER={ SYSIBM | value }
```

CATOWNER

Specifies to ZDT/Db2 the owner of every Db2® catalog table or view accessed by ZDT/Db2.

The default is SYSIBM. This means the Db2® catalog tables are accessed directly. You can specify an alternative value if you want ZDT/Db2 to access a copy of the Db2® catalog tables, or views defined on the Db2® catalog tables.

Important: If you specify a *value* for CATOWNER, this value is used by ZDT/Db2 when accessing every Db2® subsystem defined in the HFM2POPT module (each subsystem is defined by a separate invocation of the HFM2SSDM macro). Therefore, if you allow ZDT/Db2 to access the Db2® catalog directly on one Db2® subsystem, you must do so for every Db2® subsystem accessed by ZDT/Db2. Similarly, if you define views on the Db2® catalog tables for one Db2® subsystem and direct ZDT/Db2 to access those views, you must define views, with the same owner, for every Db2® subsystem accessed by ZDT/Db2.

For more information about ZDT/Db2 and the Db2® catalog tables, see Granting access to the Db2 catalog (required) on page 161.

CCSIDWARN

```
{CCSIDWARN=(INITOFF,PROFILE)|CCSIDWARN=([{INITOFF | INITON}],[{PROFILE | INITFIXED}])}
```

CCSIDWARN

Controls whether a warning message will be displayed when ZDT/Db2 connects to a Db2® subsystem, if the CCSID of the terminal differs from the CCSID of the plan. ZDT/Db2 will connect to a Db2® subsystem when the application id first entered or when the Db2® SSID is changed.

This option is used in conjunction with the CCSIDWARNIGNORE option in the HFM2SSDM macro.

Options are provided to allow you to:

- Set the initial value of the system CCSID Warning message option for new ZDT/Db2 users.
- Force the system CCSID Warning message option for all ZDT/Db2 users; alternatively allow ZDT/Db2 users to control the initial setting using the new system option, that is saved in the user's ISPF profile.

There are two sub-parameters to CCSIDWARN. You do not have to specify them all in HFM2POPI, but if you do so, they must be in the order shown in the syntax diagram. If you specify more than one, they must be separated by commas and enclosed in parentheses.

If you do not specify CCSIDWARN at all in your HFM2POPI macro, the default setting of (INITOFF,PROFILE) is used.

Note: This option only applies to Db2® Release 7 and later.

The sub-parameters are described below.

INITOFF, INITON

Determines the initial system CCSID Warning message option setting in the user's profile, when the profile is first created.

For users with an existing ZDT/Db2 profile, the effect depends on the setting of the second parameter (described below). If INITFIXED is specified for the second parameter, then every ZDT/Db2 edit session begins with either the system CCSID Warning message option not being set if INITOFF is specified, or the system CCSID Warning message option being set if INITON is specified.

PROFILE, INITFIXED

Determines if the user can alter the initial setting of system CCSID Warning message option in ZDT/Db2 edit.

INITFIXED prevents the user from altering the initial setting of the system CCSID Warning message option.

PROFILE allows the user to change this initial setting and it is preserved between ZDT/Db2 sessions in the user's profile.

CONNECT

```
CONNECT={DEFINED | ANY|(EXCLUDE[({ | Hidden | Unavail})])}
```

CONNECT

Controls how ZDT/Db2 connects to Db2® subsystems or data sharing groups.

DEFINED

Connection will only be attempted to active Db2® subsystems or groups defined in the HFM2SSDM macro. The default is **DEFINED**.

ANY

Will allow ZDT/Db2 to attempt to connect to any specified SSID, whether defined in the HFM2SSDM macro or not. ZDT/Db2 will only attempt connection to an SSID (either Db2® subsystem or data sharing group) that is active on the local z/OS® system.

CONNECT=ANY over-rides DISPLAY=UNAVAIL and DISPLAY=HIDDEN specified in the HFM2SSDM macro, however the connection will only be attempted if the specified subsystem or group is active.

EXCLUDE

Excludes Hidden or Unavailable Db2® systems when attempting to connect to a Db2® system. If no parameters are specified for EXCLUDE, both Hidden and Unavailable systems are excluded. If both parameters are specified, separate them with a comma. The parameters may be abbreviated to H and U respectively.

EDITCAPS

```
EDITCAPS=[{(INITOFF,PROFILE,CAPSCMD)|([{INITOFF | INITON}],[{PROFILE | INITFIXED}],[{CAPSCMD | NOCAPSCMD}])}]
```

EDITCAPS

Controls the setting of the CAPS option in edit. A number of options are provided, which allow you to:

- Set the initial value (either CAPS ON or CAPS OFF) for new ZDT/Db2 users.
- Force all users of the ZDT/Db2 editor to commence an edit session with either CAPS ON or CAPS OFF; alternatively to allow the user to control the initial CAPS setting using a new editor option, that is saved in the user's ISPF profile.
- Disable the CAPS command within a Z Data Tools/Db2 edit session.

  You can use certain settings of the EDITCAPS parameters to create an environment where every ZDT/Db2 edit session starts with CAPS ON, and this cannot be altered using the CAPS OFF command.

There are three sub-parameters to EDITCAPS. You do not have to specify them all in HFM2POPI, but if you do so, they must be in the order shown in the syntax diagram. If you specify more than one, they must be separated by commas. If you specify any sub-parameters, you must also provide the parentheses.

If you do not specify EDITCAPS at all in your HFM2POPI macro, the default setting of `(INITOFF,PROFILE,CAPSCMD)` is used.

These sub-parameters are described below.

INITOFF, INITON

Determines the initial CAPS setting in the user's profile, when the profile is first created.

For users with an existing ZDT/Db2 profile, the effect depends on the setting of the second parameter (described below). If INITFIXED is specified for the second parameter, then every ZDT/Db2 edit session begins with either CAPS OFF if INITOFF is specified, or CAPS ON if INITON is specified.

PROFILE, INITFIXED

Determines if the user can alter the initial setting of CAPS in ZDT/Db2 edit. INITFIXED prevents the user from altering the initial setting of CAPS. PROFILE allows the user to change this initial setting and it is preserved between ZDT/Db2 sessions in the user's profile.

CAPSCMD, NOCAPSCMD

Determines if the user is permitted to use the CAPS command within a Z Data Tools/Db2 edit session. NOCAPSCMD does not affect the CASE command, which remains available for use.

When CAPSCMD is specified, the user can turn CAPS on or off in a Z Data Tools/Db2 edit session, as required. When NOCAPSCMD is specified, the CAPS command is disabled, and is fixed at whatever setting was selected when the ZDT/Db2 edit session commenced.

Note:

1. EDITCAPS can also be specified in the HFM0POPI macro, which appears at the start of the HFM2POPT definition. Specifying EDITCAPS in the HFM0POPI macro has no effect on the ZDT/Db2 editor. To change the EDITCAPS behavior in a Z Data Tools/Db2 edit session, the EDITCAPS option must be specified in the HFM2POPI macro.
2. The EDITCAPS option available for the Z Data Tools Base function has no effect in ZDT/Db2.

OP34MOD

```
OP34MOD=module_name
```

OP34MOD

Specifies the name of the data module ZDT/Db2 will use when displaying Db2® catalog table information, with ZDT/Db2 options 3.4, 3.5 and 4.5.

ZDT/Db2 uses a data module to display the Db2® catalog table information in a specific language. If no value is specified for OP34MOD, ZDT/Db2 determines which module to load from the value specified for the LANGUAGE option in HFM2POPT. You can specify a value for OP34MOD to override this automatic selection of the data module.

The name of the module is in the form HFM2D*yyy*.

The default for OP34MOD is OP34MOD=(blank). In this case ZDT/Db2 will load a module HFM2D*yyy* where *yyy* is the language code for the language specified on the LANGUAGE option in HFM2POPT. See Table 18: Keyword values for the LANGUAGE option on page 108, for doptop34mod.

Note:

1. If LANGUAGE=ENGLISH is specified in HFM2POPT and a value for OP34MOD is not specified, then the module used is HFM2DENU.
2. If a value is specified for LANGUAGE but no module HFM2D*yyy* exists with a value for *yyy* corresponding to the LANGUAGE option, then HFM2DENU is used.
3. Three data modules are shipped with ZDT/Db2 for English, uppercase English, and Japanese. These modules are HFM2DENU, FMN2DENP and FMN2DJPN respectively.

Example

A Japanese user is using ISPF ENGLISH, (and therefore has LANGUAGE=ENGLISH specified in HFM2POPT), but wants the Db2® catalog table information to be displayed in Japanese. Therefore they require HFM2DJPN, the Japanese version of the data module. In this case, specify the following option to ensure that the Japanese version of the data module is loaded.

```
OP34MOD=HFM2DJPN
```

LIST

```
LIST={ALL | ACTIVE|[({|GROUP[({ | Active | Inactive | Unavail})]|SUBSYS[({ | Active | Inactive | Unavail})]})]}
```

LIST

Controls which items appear in the subsystem selection list. ZDT/Db2 scans the sub-system vector table of the operating system when preparing the subsystem selection list. All Db2® systems found are considered **eligible** for inclusion on the subsystem selection list. If a Db2® system is not defined to z/OS® it **will not appear** in the sub-system vector table and will therefore **not appear** in the ZDT/Db2 sub-system selection list, even if there is an HFM2SSDM macro entry for it.

ALL

All eligible (see above) subsystems and group identifiers defined in the HFM2SSDM macro appear in the subsystem selection list, with the exception of any entries specified DISPLAY=HIDDEN in the HFM2SSDM macro. The default is **LIST=ALL**.

ACTIVE

Restricts the subsystem selection list to active Db2® subsystems and data sharing groups (on the local z/OS® system).

GROUP

Lists Db2® group entries. If no parameters are specified on the GROUP option, all Active, Inactive, and Unavailable group entries are listed. To select groups in a specific status, provide one or more parameters for the GROUP option. You can specify any combination of Active, Inactive and Unavailable. If more than one parameter is specified, separate them with a comma. The parameters may be abbreviated to A, I, or U respectively.

If SUBSYS is specified, but GROUP is not, no Db2® data sharing group items will appear on the subsystem selection list.

SUBSYS

Lists Db2® subsystem entries. If no parameters are specified on the SUBSYS option, all Active, Inactive, and Unavailable subsystem entries are listed. To select subsystems in a specific status, provide one or more parameters for the SUBSYS option. You can specify any combination of Active, Inactive and Unavailable. If more than one parameter is specified, separate them with a comma. The parameters may be abbreviated to A, I, and U respectively.

If GROUP is specified, but SUBSYS is not, no Db2® subsystems will appear on the subsystem selection list.

Some explanatory information

The following information is provided to assist you in understanding why items appear on the ZDT/Db2 subsystem selection list. It is not necessary to change the entries in member IEFSSNxx.

Db2® systems are defined to z/OS® using entries in member IEFSSNxx in SYS1.PARMLIB. Here are three examples:

```
SUBSYS SUBNAME(DFB2) INITRTN(DSN3INI) INITPARM('DSN3EPX,-DFB2,S')      ◄1
SUBSYS SUBNAME(DFA2) INITRTN(DSN3INI) INITPARM('DSN3EPX,-DFA2,S,DFA2') ◄2
SUBSYS SUBNAME(DFS2) INITRTN(DSN3INI) INITPARM('DSN3EPX,-DFS2,S,DFG2') ◄3
```

1 defines a Db2® subsystem DFB2 that is not part of a Db2® data sharing group.

2 defines a Db2® subsystem DFA2 that is also part of a Db2® data sharing group. The data sharing group is also called DFA2.

3 defines a Db2® subsystem DFS2 that is also part of a Db2® data sharing group. The data sharing group name is DFG2.

1 generates a single item on the ZDT/Db2 subsystem selection list. The corresponding HFM2SSDM macro entry should specify:

```
HFM2SSDM  SSID=DFB2,TYPE=SUBSYS, ...
```

2 and **3** generate two items each on the ZDT/Db2 subsystem selection list. For **2**, the corresponding HFM2SSDM macro entries are:

```
HFM2SSDM  SSID=DFA2,TYPE=SUBSYS, ...
HFM2SSDM  SSID=DFA2,TYPE=GROUP, ...
```

For **3**, the required HFM2SSDM macro entries are:

```
HFM2SSDM  SSID=DFS2,TYPE=SUBSYS, ...
HFM2SSDM  SSID=DFG2,TYPE=GROUP, ...
```

Note that when Db2® system DFA2 is active, Db2® group DFA2 is also active. Similarly when Db2® system DFS2 is active, Db2® group DFG2 is also active. This occurs regardless of whether Db2® data sharing groups DFA2 and DFG2 has been completed implemented or not. From the perspective of z/Os, definitions **2** and **3** in IEFSSNxx are all that is required to define the Db2® data sharing groups DFA2 and DFG2.

In the situation where a Db2® subsystem and Db2® group have the same name (for example, **2**), you can hide one of the entries using DISPLAY=HIDDEN on the corresponding HFM2SSDM macro definition. You can also use the SUBSYS or GROUP options to show only subsystems or groups, respectively, although the latter method applies to every subsystem or group.

For example, to show the group DFA2 but not the subsystem DFA2, the HFM2SSDM macro entries are:

```
HFM2SSDM SSID=DFS2,TYPE=SUBSYS,DISPLAY=HIDDEN
HFM2SSDM SSID=DFG2,TYPE=GROUP, ...
```

SHOWDATAC

SHOWDATAC=*XXXXXXXX*

SHOWDATAC

Specifies the SMS data class that Z Data Tools uses when allocating the data set created when the SQL command is issued. The maximum length you can specify for SHOWDATAC is eight (8) bytes. SHOWDATAC is optional. If omitted the SMS data class specified in PDATAC is used, if supplied.

SHOWMGMTC

SHOWMGMTC=*XXXXXXXX*

SHOWMGMTC

Specifies the default SMS management class that Z Data Tools uses when allocating the data set created when the SQL command is issued. The maximum length you can specify for SHOWMGMTC is eight (8) bytes. SHOWMGMTC is optional. If omitted the SMS management class specified in PMGMT is used, if supplied.

SHOWPQTY

SHOWPQTY= { **20** | *nnnnnnnn* }

SHOWPQTY

Specifies the amount of DASD space to be used for primary space allocation of the data set created when the SQL command is issued. The range depends on the space unit specified and the DASD device type.

SHOWSQTY

SHOWSQTY= { **20** | *nnnnnnnn* }

SHOWSQTY

Specifies the amount of DASD space to be used for secondary space allocation of the data set created when the SQL command is issued. The range depends on the space unit specified and the DASD device type.

SHOWSTORC

SHOWSTORC=*XXXXXXXX*

SHOWSTORC

Specifies the default SMS storage class that Z Data Tools uses when allocating the data set created when the SQL command is issued. The maximum length you can specify for SHOWSTORC is eight (8) bytes. SHOWSTORC is optional. If omitted the SMS storage class specified in PSTORC is used, if supplied.

SHOWSUNIT

```
SHOWSUNIT={ TRK | REC | KB | MB | CYL }
```

SHOWSUNIT

Specifies the unit of primary and secondary space to be allocated to the data set created when the SQL command is issued.

REC

Record of average size.

KB

Kilobyte, a kilobyte is 1024 bytes.

MB

Megabyte, a megabyte is 1048576 bytes.

TRK

Track of a direct access storage device (DASD).

CYL

Cylinder of a DASD.

SHOWUNIT

```
SHOWUNIT=XXXXXXXX
```

SHOWUNIT

Specifies the default permanent unit that Z Data Tools uses when allocating the data set created when the SQL command is issued. The maximum length you can specify for SHOWUNIT is eight (8) bytes. SHOWUNIT is optional. If omitted the SMS storage class specified in PUNIT is used.

SSIDCMD1

```
SSIDCMD1={ SSID | value1 }
```

SSIDCMD1

Specifies the ZDT/Db2 command to be used to dynamically change the currently connected Db2® subsystem. This keyword is optional.

SSID

Default command to dynamically change the currently connected Db2® subsystem.

value1

Installation-defined command to dynamically change the currently connected Db2® subsystem. If you specify a value for *value1* it must be 4-8 characters in length, and start with an alphabetic or national character. The remaining characters can be alphabetic, national, or numeric.

SSIDCMD2

```
SSIDCMD2= { DB2SYS | value2 }
```

SSIDCMD2

Specifies an additional ZDT/Db2 command to be used to dynamically change the currently connected Db2® subsystem, if required. This keyword is optional.

DB2SYS

Default alternative command to dynamically change the currently connected Db2® subsystem.

value2

An alternative installation-defined command to dynamically change the currently connected Db2® subsystem. If you specify a value for *value2* it must be 4-8 characters in length, and start with an alphabetic or national character. The remaining characters can be alphabetic, national, or numeric. SSIDCMD1 and SSIDCMD2 can have the same value.

Appendix C. ZDT/IMS options

This section describes the parameters of the HFM1POPD, HFM1POPI, and HFM1AGNT macro statements used in the ZDT/IMS installation options module. For more information on how to use these macro statements, see Customizing the ZDT/IMS installation options module on page 264.

HFM1POPD and HFM1POPI macros

The HFM1POPD macro statement supports all the parameters that the HFM1POPI macro statement supports except for the following:

- SSID
- DESC
- IMSPLEX

The HFM1POPI macro statement supports all the parameters that the HFM1POPD macro statement supports except for the following:

- COMPAT
- LKEYxxxx
- LOGxxxx
- PSBTYPE
- RKEYxxxx
- SKELLIB
- UPSBTYPE
- XKEYxxxx
- PADS

For the parameters that both macro statements support:

- The syntax of the parameter on the HFM1POPD macro statement is the same as the syntax of the parameter on the HFM1POPI macro statement.
- Each parameter is optional on both macro statements.
- Only the defaults are different. The default for the HFM1POPI parameter is the value specified on the HFM1POPD macro statement, while the default for the HFM1POPD macro statement is a fixed value.

Many parameters support the specification of either a single value, or two values in parentheses separated by a comma.

For each of these parameters, when two values are specified in parentheses separated by a comma, the first value is used for BMP mode and the second value is used for DLI mode. When a single value is specified, this value is used for both BMP and DLI mode.

The parameters of the HFM1POPD and HFM1POPI macro statements are described in the following sections. They are listed in alphabetical order.

ACBLIB

```
ACBLIB= { IMS.ACBLIB | dsname }
```

ACBLIB

Specifies the staging ACBLIB data set name. This parameter is used by the Initialize function when a Fast Path database is specified.

This parameter is optional. If it is not specified on the HFM1POPI macro statement, ZDT/IMS uses the value on the HFM1POPD macro statement (if one is specified), or IMS.ACBLIB.

If you do not want the user to override the value you specify for this parameter, set UACBLIB=N.

ACBMGMT

```
ACBMGMT= { ACBLIB | CATALOG }
```

ACBMGMT

Specifies how the active application control blocks (ACBs) are managed, as indicated by the ACBMGMT parameter in the CATALOG section of the DFSDFxxx PROCLIB member.

ACBLIB

The active ACBs are managed by your installation in ACBLIB data sets.

CATALOG

The active ACBs are managed by IMS™ in the IMS directory data sets of the IMS catalog.

When ACBMGMT=ACBLIB is specified:

- ZDT/IMS gets the DBDs and PSBs from the specified DBD and PSB libraries. (See DBDLIBn on page 462 and PSBLIBn on page 485.)
- The PSB Generation utility and the ACB Maintenance utility are used to generate dynamic PSBs.

When ACBMGMT=CATALOG is specified:

- ZDT/IMS gets the DBDs and PSBs from the IMS catalog through the IMS catalog API.
- When USEDDL=Y is specified, the IMS Data Definition utility is used to generate dynamic PSBs. Otherwise, the PSB Generation utility, the IMS Catalog Library Builder utility, the ACB Maintenance utility and the IMS Catalog Populate utility are used.

This parameter is optional. If it is not specified on the HFM1POPI macro statement, ZDT/IMS uses either the value specified on the HFM1POPD macro statement (if specified on that statement), or ACBLIB (otherwise).

The user cannot override the value you specify for this parameter.

ACBSHR

```
ACBSHR = { N  | Y }
```

ACBSHR

Specifies whether this IMS™ subsystem shares a common set of application control blocks (ACBs) with other IMS subsystems in an IMSplex.

N

This IMS™ subsystem uses its own dedicated set of ACBs.

Y

The set of ACBs that this IMS subsystem uses is shared by other IMS subsystems.

When IMS™ management of ACBs is enabled (ACBMGMT=CATALOG), the set of ACBs are in the IMS directory data sets of the IMS catalog. Set ACBSHR to Y when multiple IMS subsystems use the same IMS catalog.

When the ACBs are managed by your installation (ACBMGMT=ACBLIB), the set of ACBs are in ACBLIB data sets. Set ACBSHR to Y when multiple IMS subsystems use the same ACBLIB data sets.

When ACBSHR=Y is specified, you must use the IMSPLEX parameter to specify the IMSplex that this IMS subsystem is in. See IMSPLEX on page 473 for information on how you do this.

When ACBSHR=Y is specified, ZDT/IMS serializes the use of dynamic PSBs across the IMSplex. Otherwise, ZDT/IMS serializes the use of dynamic PSBs across the IMS subsystem.

This parameter is optional. If it is not specified on the HFM1POPI macro statement, ZDT/IMS uses either the value specified on the HFM1POPD macro statement (if specified on that statement), or N (otherwise).

The user cannot override the value you specify for this parameter.

AUTOSAVE

```
AUTOSAVE= { (Y,Y) | ( [{Y  | N}] , [{Y  | N}] )  | Y  | N }
```

AUTOSAVE

Specifies whether or not the automatic save function is set on during an Edit.

Y

The automatic save function is set on during an Edit.

N

The automatic save function is not set on during an Edit.

When you specify two values in parentheses, separated by a comma, the first value is used for BMP mode and the second value is used for DLI mode. When you specify a single value, this is used for both BMP and DLI modes.

This parameter is optional. If it is not specified on the HFM1POPI macro statement, ZDT/IMS uses either the value specified on the HFM1POPD macro statement (if specified on that statement), or "(Y,Y)" (otherwise).

If you do not want the user to override the value you specify for this parameter, set UAUTOSAV=N.

BSDSHLQ

```
BSDSHLQ= { IMS.DFSCD000 | bsds_hlq }
```

BSDSHLQ

Specifies the high level qualifier for the IMS™ bootstrap data set. The qualifier '.BSDS' is appended to the specified qualifier to form the full name of the IMS™ bootstrap data set.

When IMS™ management of ACBs is enabled (ACBMGMT=CATALOG), the specified high-level qualifier is passed to the IMS™ catalog API (DFS3CATQ) and the IMS™ Catalog Library Builder utility (DFS3LU00).

This parameter is optional. If it is not specified on the HFM1POPI macro statement, ZDT/IMS uses either the value on the HFM1POPD macro statement (if one is specified), or IMS.DFSCD000.

The user cannot override the value you specify for this parameter.

CATALIAS

```
CATALIAS = { DFSC | XXXX }
```

CATALIAS

The alias for the IMS™ catalog as specified on the ALIAS parameter in the CATALOG section of the DFSDFxxx PROCLIB member.

This parameter is optional. If it is not specified on the HFM1POPI macro statement, ZDT/IMS uses either the value specified on the HFM1POPD macro statement (if specified on that statement), or DFSC (otherwise).

The user cannot override the value you specify for this parameter.

CHGAFREQ

```
CHGAFREQ= { (100, 100) | (nnnn,mmmm ) | pppp }
```

CHGAFREQ

Specifies the frequency of the automatic save function during Change All and Repeat All operations; that is, the Change All/Repeat All checkpoint frequency.

When AUTOSAVE is set on, ZDT/IMS:

- Increments a count by 1 each time a segment is updated during a Change All operation or inserted during a Repeat All operation.
- Issues a checkpoint when the count is equal to the Change All/Repeat All checkpoint frequency.

When you specify two values in parentheses, separated by a comma, the first value is used for BMP mode and the second value is used for DLI mode. When you specify a single value, this is used for both BMP and DLI modes.

This parameter is optional. If it is not specified on the HFM1POPI macro statement, ZDT/IMS uses either the value specified on the HFM1POPD macro statement (if specified on that statement), or "(100,100)" (otherwise).

If you do not want the user to override the value you specify for this parameter, set UAUTOSAV=N.

CHKPINTVL

```
CHKPINTVL={ 0000 | MMSS }
```

CHKPINTVL

Specifies the time interval between checkpoints when the Edit/Browse BMP is waiting for a response from the user. The time interval is represented as zoned decimal digits of the form MMSS, where MM is minutes and SS is seconds.

The specified time interval must not be greater than 60 minutes and SS must be in the range 00 to 59.

If you specify 0000, then the BMP does not issue checkpoints when waiting for a response from the user.

CHKPINTVL equal to a non zero value is incompatible with each of these settings:

- EDITFREQ BMP value not equal to 1
- AUTOSAVE BMP value equal to N
- UAUTOSAV BMP value equal to Y

If you specify CHKPINTVL equal to a nonzero value and one or more of these incompatible settings, ZDT/IMS uses your CHKPINTVL setting and ignores the incompatible settings, using these compatible settings instead:

- EDITFREQ BMP value equal to 1
- AUTOSAVE BMP value equal to Y
- UAUTOSAV BMP value equal to N

The CHKPINTVL parameter is optional. If it is not specified on the HFM1POPI macro statement, ZDT/IMS uses either the value specified on the HFM1POPD macro statement (if specified on that statement), or "0000" otherwise.

The user cannot override the value you specify for this parameter.

COMPAT

```
COMPAT={ N | Y }
```

COMPAT

Specifies whether or not ZDT/IMS accepts the absence of the IMSID parameter when a batch job runs in DLI mode.

N

The IMSID parameter must be specified in the HFMIMSIN input of batch functions that run in DLI mode.

Y

The IMSID parameter in the HFMIMSIN input of batch functions that run in DLI mode is optional.

Prior to version 8, ZDT/IMS did not generate an IMSID parameter when the batch function ran in DLI mode.

Therefore, if you are migrating from a version of ZDT/IMS earlier than version 8, and you specify (or default to) COMPAT=N, some ZDT/IMS batch job steps that ran with the version you are migrating from will require modification before they can be run with the new version of ZDT/IMS.

In particular, you will need to add the IMSID parameter to ZDT/IMS job steps:

- That execute the IXB, ILB, IPR, IEB or IBB function and that specify REGNTYPE=DLI.
- That execute the DIB function.

If you specify COMPAT=Y, JCL generated by versions of ZDT/IMS earlier than version 8 should run without modification.

However, COMPAT=N is recommended.

This parameter can only be specified on the HFM1POPD macro statement.

This parameter is optional. The default is N. The user cannot override the value you specify for this parameter.

DBDLIB*n*

```
DBDLIB1= { IMS.DBDLIB | dsname }
DBDLIBn=dsname
```

DBDLIB*n*

Where *n* is a number in the range 1 to 6, specifies the names of the load libraries that contain the database definitions (DBDs) that ZDT/IMS and IMS™ are to use.

The specified data sets are used when the ACBs are managed by your installation (ACBMGMT=ACBLIB).

These parameters are optional. But if you specify a DBDLIB parameter, you must also specify the DBDLIB parameters that precede it. So, for example, if you specify the DBDLIB3 parameter, you must also specify the DBDLIB1 parameter and the DBDLIB2 parameter.

If no DBDLIB parameters are specified on the HFM1POPI macro statement, ZDT/IMS uses the DBDLIB values on the HFM1POPD macro statement (if specified), otherwise it uses IMS.DBDLIB.

If you do not want the user to override the values you specify for these parameters, set UDBDLIB=N.

Note: The DBDLIBn data sets are not required when IMS management of ACBs is enabled (ACBMGMT=CATALOG), as ZDT/IMS and IMS get the DBDs from the IMS™ catalog that the subsystem uses.

DBRC

```
DBRC={IMS | Y | N}
```

DBRC

Specifies whether or not ZDT/IMS functions use Database Recovery Control (DBRC) when running in DLI mode.

IMS

DBRC is used if the IMSCTRL macro statement specifies DBRC=YES or DBRC=FORCE. Otherwise, DBRC is not used.

Y

DBRC is used.

N

DBRC is not used, unless the IMSCTRL macro specifies DBRC=FORCE.

The DBRC parameter is used when the ACBs are managed by your installation (ACBMGMT=ACBLIB) and the database being accessed is not a HALDB or a HALDB partitioned secondary index.

This parameter is optional. If it is not specified on the HFM1POPI macro statement, ZDT/IMS uses either the value specified on the HFM1POPD macro statement (if specified on that statement), or IMS™ (otherwise).

If you do not want the user to override the value you specify for this parameter, set UDBRC=N.

Notes:

- When the database being accessed is a HALDB, DBRC is used unless the database is the IMS™ catalog and the REGCATLG parameter is set to N.
- When the database being accessed is a HALDB partitioned secondary index, DBRC is used unless the database is the IMS™ catalog's secondary index and the REGCATLG parameter is set to N.
- When IMS™ management of ACBs is enabled (ACBMGMT=CATALOG) DBRC is used unless the REGCATLG parameter is set to N.

DESC

```
DESC='IMS subsystem description'
```

DESC

Specifies a description of the IMS™ subsystem, to be displayed on the Subsystem Selection panel. The maximum length of the description is 30 characters. The description should be enclosed in quotation marks.

This parameter can only be specified on the HFM1POPI macro statement.

This parameter is optional. The default is no description. The user cannot override the value you specify for this parameter.

DFSDF

```
DFSDF= { 000 | xxx }
```

DFSDF

Specifies the 3-character suffix of the DFSDFxxx member of the IMS™ PROCLIB data set that contains the settings and attributes of the IMS™ catalog. This member is used by functions that run in DLI mode when IMS™ management of ACBs is enabled (ACBMGMT=CATALOG) and the IMS™ Catalog Definition exit routine (DFS3CDX0) is not used.

When the member is used, ensure that it has a CATALOG section in which the CATALOG parameter is set to YES, the ACBMGMT parameter is set to CATALOG and the ALIAS parameter is set to the alias for the IMS catalog. If the IMS catalog database and its secondary index are not registered with DBRC (REGCATLG=N), ensure that their names are specified on the UNREGCATLG parameter in the DATABASE section of the member.

This parameter is optional. If it is not specified on the HFM1POPI macro statement, ZDT/IMS uses either the value specified on the HFM1POPD macro statement (if specified on that statement), or 000 (otherwise).

The user cannot override the value you specify for this parameter.

DFSRRC00

```
DFSRRC00= { DFSRRC00 | pgmname }
```

DFSRRC00

Specifies the name of the program that ZDT/IMS attaches to invoke IMS™. This will usually be the IMS™ region controller program, DFSRRC00.

If a different program is to be used, it must be in one of the RESLIB data sets. For the RESLIB parameters see RESLIBn on page 488.

This parameter is optional. If it is not specified on the HFM1POPI macro statement, ZDT/IMS uses either the value specified on the HFM1POPD macro statement (if specified on that statement), or DFSRRC00 (otherwise).

The user cannot override the value you specify for this parameter.

DFSVSAMP

```
DFSVSAMP= { IMS.PROCLIB | proclib }
```

DFSVSAMP

Specifies the name of the data set that contains the parameters defining the IMS™ buffer sub pools for VSAM and OSAM data sets. When a Z Data Tools/IMS function is run in DLI mode, the specified data set is allocated to the DFSVSAMP DD.

If the specified data set is a PDS or PDSE, specify the required member name of this data set in the VSMPMEM parameter.

This parameter is optional. If it is not specified on the HFM1POPI macro statement, ZDT/IMS uses either the value on the HFM1POPD macro statement (if one is specified), or IMS.PROCLIB.

If you do not want the user to override the value you specify for this parameter, set UDFSVSMP=N.

DYNACB

```
DYNACB={ IMS.DOPT.ACBLIB | dsname }
```

DYNACB

Specifies the name of the ACBLIB data set into which the ACB maintenance utility generates the DOPT PSBs.

The data set is not required when the dynamic PSBs are generated by the IMS Data Definition utility. (See USEDDL on page 507).

The following applies when the dynamic PSBs are generated by the PSB Generation utility.

This data set is required by functions that use dynamic PSBs to access databases in BMP mode. If IMS™ management of ACBs is enabled (ACBMGMT=CATALOG), it is also required by functions that use dynamic PSBs to access databases in DLI mode.

When the ACBs are managed by your installation (ACBMGMT=ACBLIB), the specified data set must be concatenated with the primary ACBLIB data set in the IMS execution JCL (see Providing a DOPT ACBLIB data set on page 263).

When IMS management of ACBs is enabled (ACBMGMT=CATALOG), the specified data set does not have to be concatenated with the primary ACBLIB data set. Instead, the specified data set is used as input for the IMS Catalog Populate utility (DFS3PU00), which adds the DOPT PSBs to the IMS catalog.

If APAR PH17975 has been applied or XDOPTLB=Y has been specified, you must specify a data set that only ZDT/IMS uses.

This parameter is optional. If it is not specified on the HFM1POPI macro statement, ZDT/IMS uses either the value specified on the HFM1POPD macro statement (if specified on that statement), or IMS.DOPT.ACBLIB.

The user cannot override the value you specify for this parameter.

DYNALLOC

```
DYNALLOC={ N | Y }
```

DYNALLOC

Specifies whether or not ZDT/IMS enforces usage of the database data sets specified in the dynamic allocation modules, when accessing databases in DLI mode.

N

The user is not forced to use the database data sets specified in the dynamic allocation modules when accessing databases in DLI mode.

Y

The user is forced to use the database data sets specified in the dynamic allocation modules when accessing databases in DLI mode.

If you specify DYNALLOC=Y:

- The Database Data Set Display panel is displayed in the ZDT/IMS dialog when a DLI region type is selected on the Entry panel.
- The JCL generated by ZDT/IMS dialogs does not include DD statements for the database data sets.
- Batch functions running in DLI mode will fail if their JCL includes DD statements for the database data sets.
- If, for an Edit or Browse in DLI mode, the database data set DD names are allocated to your TSO session prior to the DL/I batch processing region being started, ZDT/IMS frees the data sets allocated to these DD names.
- The database data sets are dynamically allocated by IMS™.

If you specify DYNALLOC=N:

- The Database Data Set Specification panel is displayed in the ZDT/IMS dialog when a DLI region type and a non-HALDB database is specified on the Entry panel.
- The JCL generated by ZDT/IMS dialogs includes DD statements for non-HALDB database data sets.
- In an Edit or Browse, ZDT/IMS allocates the database data sets specified on the Database Data Set Specification panel.

This parameter is optional. If it is not specified on the HFM1POPI macro statement, ZDT/IMS uses either the value specified on the HFM1POPD macro statement (if specified on that statement), or N (otherwise).

The user cannot override the value you specify for this parameter.

To ensure that the database data sets specified in the dynamic allocation modules are used when the subsystem is selected, you should also set URESLIB=N and URECON=N.

DYNPRFN

DYNPRFN= { **50** | *nnn* }

DYNPRFN

Specifies the maximum number of dynamic PSBs required by concurrent ZDT/IMS users (when the PSBs are generated by the IMS Data Definition utility) or by concurrent ZDT/IMS BMP users (otherwise).

The number must be in the range 1 to 999.

This parameter is optional. If it is not specified on the HFM1POPI macro statement, ZDT/IMS uses either the value specified on the HFM1POPD macro statement (if specified on that statement), or 50 (otherwise).

The user cannot override the value you specify for this parameter.

Note: If you plan to use dynamic PSBs, some customization of IMS might be required. For information on how to customize IMS to support dynamic PSBs, see Customizing IMS to support the use of dynamic PSBs on page 262.

DYNPRFX

```
DYNPRFX= { HFM0 | XXXXX }
```

DYNPRFX

Specifies the first 1 - 5 characters of the dynamic PSB names used by ZDT/IMS functions.

This parameter is used by functions that run in BMP mode and also by functions that run in DLI mode.

If the function runs in BMP mode or the PSBs are generated by the IMS Data Definition utility, the dynamic PSB names are obtained by appending a 3-digit number to the value you specify in this parameter. For example, if you specify DYNPRFX=HFM and DYNPRFN=3, ZDT/IMS uses HFM001, HFM002, and HFM003 for the dynamic PSB names.

Otherwise, the dynamic PSB names are obtained by appending '000' to the value you specify in this parameter. For example, if you specify DYNPRFX=HFM, ZDT/IMS uses HFM000 for the dynamic PSB name.

Note: If you plan to use dynamic PSBs, some customization of IMS may be required. For information on how to customize IMS to support dynamic PSBs, see Customizing IMS to support the use of dynamic PSBs on page 262.

This parameter is optional. If it is not specified on the HFM1POPI macro statement, ZDT/IMS uses either the value specified on the HFM1POPD macro statement (if specified on that statement), or HFM0 (otherwise).

The user cannot override the value you specify for this parameter.

DYNPSB

```
DYNPSB= { HFM.DYN.PSBLIB | dsname }
```

DYNPSB

Specifies the name of the PDSE program library in which ZDT/IMS stores dynamic PSBs when PADS=Y is specified on the HFM1POPD macro statement and the PSBs are generated by the PSB Generation utility.

If the IMS™ subsystem supports the use of dynamic PSBs in BMP mode, use of this library should be restricted to this subsystem.

This parameter is optional. If it is not specified on the HFM1POPI macro statement, ZDT/IMS uses either the value specified on the HFM1POPD macro statement (if specified on that statement), or HFM.DYN.PSBLIB (otherwise).

The user cannot override the value you specify for this parameter.

Note:

- When PADS=N is specified on the HFM1POPD macro statement, dynamic PSBs that are generated by the PSBGEN utility are stored in temporary data sets with system-generated names.
- The DYNPSB library is not required when dynamic PSBs are generated by the the IMS Data Definition utility. See USEDDL on page 507.

DYNTPLT

```
DYNTPLT = { N | Y }
```

DYNTPLT

Specifies whether the dynamic generation of ZDT/IMS templates is enabled.

N

ZDT/IMS functions that have requested a new or an existing view or criteria set don't attempt to generate a template for the database that the function is accessing.

Y

ZDT/IMS functions that have requested a new or an existing view or criteria set attempt to generate a template for the database that the function is accessing.

Functions generate the template from field definitions in the DBDs. Therefore, it is a requirement that application-defined fields must have been added to your DBDs.

If application-defined fields have not been added to your DBDs, set DYNTPLT=N, otherwise attempts to generate a template for the database would fail.

When ACBs are managed by IMS™ (ACBMGMT=CATALOG), functions get the field definitions from the IMS™ catalog . When ACBs are managed by your installation (ACBMGMT=ACBLIB), functions get the field definitions from the DBD libraries for the subsystem.

This parameter is optional. If it is not specified on the HFM1POPI macro statement, ZDT/IMS uses either the value specified on the HFM1POPD macro statement (if specified on that statement), or N (otherwise).

Users cannot override the value you specify for this parameter.

EDITFREQ

```
EDITFREQ= { (1, 1) | (nn,mm ) | pp }
```

EDITFREQ

Specifies the frequency of the automatic save function when editing databases; that is, the Edit checkpoint frequency.

When AUTOSAVE is set on, ZDT/IMS:

- Increments a count by 1 each time an action key is pressed and data changes have been requested.
- Issues a checkpoint when the count is equal to the Edit checkpoint frequency.

The valid range is 1 to 99.

When you specify two values in parentheses, separated by a comma, the first value is used for BMP mode and the second value is used for DLI mode. When you specify a single value, this is used for both BMP and DLI modes.

This parameter is optional. If it is not specified on the HFM1POPI macro statement, ZDT/IMS uses either the value specified on the HFM1POPD macro statement (if specified on that statement), or "(1,1)" (otherwise).

If you do not want the user to override the value you specify for this parameter, set UAUTOSAV=N.

GSGNAME

```
GSGNAME={ gsgname | NONE }
```

GSGNAME

If the IMS™ subsystem is part of a Remote Site Recovery (RSR) complex and the GSGNAME has not been specified in the IMSCTRL macro statement, and you want any activity performed by ZDT/IMS functions running in DLI mode to be tracked by RSR, specify the Global Service Group (GSG) name for the RSR complex.

If the IMS™ subsystem is part of an RSR complex and the GSGNAME has been specified in the IMSCTRL macro statement and you do not want the activity of ZDT/IMS functions running in DLI mode to be tracked, specify NONE.

Otherwise, do not specify a value for GSGNAME.

This parameter is optional. If it is not specified on the HFM1POPI macro statement, ZDT/IMS uses either the value specified on the HFM1POPD macro statement (if specified on that statement), or a null value (otherwise). When a null value is passed to the IMS™ region controller, IMS™ uses:

- The value specified for the GSGNAME parameter on the IMSCTRL macro statement (if specified on that statement), or NONE (otherwise).

If you do not want the user to override the value you specify for this parameter, set URSR=N.

IEBFREQ

```
IEBFREQ={ (100, 100) | (nnnnn,mmmmm ) | ppppp }
```

IEBFREQ

Specifies the frequency of the automatic save function when running a Batch Edit job; that is, the Batch Edit checkpoint frequency.

ZDT/IMS:

- Increments a count by 1 each time a Batch Edit job inserts, deletes or replaces a segment.
- Issues a checkpoint when the count is equal to the Batch Edit checkpoint frequency.

The valid range is 1 to 99999.

When you specify two values in parentheses, separated by a comma, the first value is used for BMP mode and the second value is used for DLI mode. When you specify a single value, this is used for both BMP and DLI modes.

This parameter is optional. If it is not specified on the HFM1POPI macro statement, ZDT/IMS uses either the value specified on the HFM1POPD macro statement (if specified on that statement), or "(100,100)" (otherwise).

If you do not want the user to override the value you specify for this parameter, set UIEBFREQ=N.

IMSAUDLG

```
IMSAUDLG={N | Y | D}
```

IMSAUDLG

Specifies whether or not audit logging is enforced during Edit.

N

Audit logging is not enforced during Edit.

Y

Audit logging is enforced during Edit.

D

Audit logging is enforced during edit, and at the conclusion of the edit session an audit report job is submitted to report on the changes. This job can be customized by changing member HFM1FTAD from HFM.SHFMSLIB to specify the job card and the reporting options you require.



Note: ZDT/IMS only uses the value specified for this parameter, when the FMAUDIT parameter in the HFM1PARM member in the SYS1.PARMLIB (or its concatenations) specifies SAF_CTRL=NO. When the FMAUDIT parameter specifies SAF_CTRL=YES, ZDT/IMS audit logging is controlled by SAF and ZDT/IMS ignores this parameter.

This parameter is optional. If it is not specified on the HFM1POPI macro statement, ZDT/IMS uses either the value specified on the HFM1POPD macro statement (if specified on that statement), or N (otherwise). This is true if ZDT/IMS does not call an Audit trail exit.

If ZDT/IMS calls an Audit trail exit, the value returned by the exit overrides the value specified on the HFM1POPI and HFM1POPD macro statements.

IMSBKO

```
IMSBKO={Y | N}
```

IMSBKO

Specifies whether or not a dynamic backout is to be performed when an IMS™ pseudoabend occurs in a Z Data Tools/IMS function running in DLI mode.

Y

A dynamic backout is performed when an IMS™ pseudoabend occurs in a Z Data Tools/IMS function running in DLI mode.

If you specify IMSBKO=Y, users will be able to use the UNDO or CANCEL command when editing a database in DLI mode.

N

A dynamic backout is not performed when an IMS™ pseudoabend occurs in a Z Data Tools/IMS function running in DLI mode.

This parameter only takes effect when the ZDT/IMS function is running with an IMS™ log data set.

This parameter is optional. If it is not specified on the HFM1POPI macro statement, ZDT/IMS uses either the value specified on the HFM1POPD macro statement (if specified on that statement), or Y (otherwise).

If you do not want the user to override the value you specify for this parameter, set UIMSBKO=N.

IMSGROUP=identifier

IMSGROUP

If the IMS™ subsystem is a member of an IMSGROUP, specify the 1- to 4-character IMSGROUP identifier. Otherwise, do not specify any values for this parameters.

This parameter is supported only in BMP mode and can be specified only on the HFM1POPI macro statement. The IMSGROUP value specified onHFM1POPI must match the value defined for the corresponding IMSID in the DFSPBxxx member. You cannot override the value specified for this parameter.

Example

Defining an IMSGROUP in the ZDT/IMS installation options module

Suppose that three IMS subsystems—IFA1, IFA2, and IFA3—are members of IMSGROUP IMSA. To allow ZDT/IMS to connect to an active member of the group, define the HFM1POPI entries in the preferred connection order, as shown in the following example.

```
FMNPOPI SSID= 1,                                                ← First Preference
    IMSGROUP=IMSA,
    DESC='IMS V14',
    LIB1=IMS.IFA1.SDFSRESL,
    ...

FMN1POPI SSID=IFA2,                                             ← Second preference
    IMSGROUP=IMSA,
    DESC='IMS V15',
    LIB1=IMS.IFA2.SDFSRESL,
    ...
FMN1POPI SSID=IFA3,                                             ← Third preference
    IMSGROUP=IMSA,
    DESC='IMS V15',
    LIB1=IMS.IFA3.SDFSRESL,


When the user specifies IMSA as the IMS subsystem name in BMP mode,ZDT/IMS
scans the group members in the order they are defined in the installation options module and connects to the
 first active member:
IFA1 status        IFA2 status        IFA3 status           ZDT/IMS connects to
ACTIVE             ACTIVE             ACTIVE                IFA1 (first in sequence)
INACTIVE           ACTIVE             ACTIVE                IFA2 (first active in sequence)
INACTIVE           INACTIVE           ACTIVE                IFA3 (first active in sequence)
INACTIVE           INACTIVE           INACTIVE              Error - All members inactive
```

Note: If a member SSID has the same name as the IMSGROUP name, for example, `SSID=IFF1` and `IMSGROUP=IFF1`, define that entry first in the installation options module. This ensures that ZDT/IMS connects directly to the IMS subsystem when it is active. If the IMS subsystem is inactive, ZDT/IMS continues scanning the remaining HFM1POPI entries in sequence.

IMSNBA

IMSNBA= { **10** | *nnnn* }

IMSNBA

Specifies the number of Fast Path database buffers to be made available in the Common Service Area (CSA) when a Fast Path region is activated. The number specified is used by ZDT/IMS functions when they access Fast Path databases.

The number you specify must be in the range 1 to 9999.

This parameter is optional. If it is not specified on the HFM1POPI macro statement, ZDT/IMS uses either the value specified on the HFM1POPD macro statement (if specified on that statement), or 10 (otherwise).

If you do not want the user to override the value you specify for this parameter, set UIMSNBA=N.

IMSOBA

IMSOBA= { **6** | *nnnn* }

IMSOBA

Specifies the number of additional page-fixed buffers to be made available to a Fast Path region if the normal allotment (specified or defaulted in IMSNBA) is used. The number specified is used by ZDT/IMS functions when they access Fast Path databases.

The number you specify must be in the range 1 to 9999.

This parameter is optional. If it is not specified on the HFM1POPI macro statement, ZDT/IMS uses either the value specified on the HFM1POPD macro statement (if specified on that statement), or 6 (otherwise).

If you do not want the user to override the value you specify for this parameter, set UIMSNBA=N.

IMSPLEX

```
IMSPLEX=imsplex_identifier
```

IMSPLEX

If the IMS™ subsystem is a member of an IMSplex, specify the 1 to 5-character IMSplex identifier. Otherwise, do not specify a value for this parameter.

When ACBSHR=Y is specified, ZDT/IMS serializes the use of dynamic PSBs across the specified IMSplex.

The parameter can only be specified on the HFM1POPI macro statement.

The user cannot override the value you specify for this parameter.

IRLM

```
IRLM={IMS | Y | N}
```

IRLM

Specifies whether or not ZDT/IMS functions uses an Internal Resource Lock Manager (IRLM) when running in DLI mode.

IMS™

IRLM is used if either the IMSCTRL macro statement specifies IRLM=Y, or the IMSCTRL macro statement specifies an IRLMNM and does not specify IRLM=N.

IRLM is not used if either the IMSCTRL macro statement specifies IRLM=N or the IMSCTRL macro statement does not specify a value for IRLM and IRLMNM.

Y

IRLM is used.

N

IRLM is not used unless the IMSCTRL macro statement specifies IRLM=Y and an IRLMNM.

This parameter is optional. If it is not specified on the HFM1POPI macro statement, ZDT/IMS uses either the value specified on the HFM1POPD macro statement (if specified on that statement), or IMS™ (otherwise).

If you do not want the user to override the value you specify for this parameter, set UIRLM=N.

IRLMNAME

```
IRLMNAME=user_irlmname
```

IRLMNAME

Specifies the 4-byte z/OS® subsystem name assigned to the Internal Resource Lock Manager (IRLM). ZDT/IMS functions running in DLI mode pass the name to the IMS™ region controller when IRLM=Y or IRLM=IMS.

This parameter is optional. If it is not specified on the HFM1POPI macro statement and you specify IRLM=Y or IRLM=IMS, ZDT/IMS uses either the value specified on the HFM1POPD macro statement (if specified on that statement), or a null value (otherwise). When a null value is passed to the IMS™ region controller, IMS™ uses:

- The value specified for the IRLMNM parameter on the IMSCTRL macro statement (if specified on that statement), or IRLM (otherwise).

If you do not want the user to override the value you specify for this parameter, set UIRLM=N.

LIST

```
LIST = { ALL  | ACTIVE }
```

LIST

Controls which IMS™ subsystems are displayed in the ZDT/IMS subsystem selection list. Specify this parameter in the HFM1POPD macro. The parameter applies to all subsystems defined in HFM1POPI entries.

ALL

All IMS™ subsystems defined in the ZDT/IMS installation options module (HFM1POPT) are eligible for inclusion in the subsystem selection list, regardless of whether those subsystems are currently active. A subsystem defined in HFM1POPI will appear in the selection list even if it is not registered in the z/OS Subsystem Vector Table (SSCVT) at the time the Subsystem selection list is built.

ACTIVE

Only IMS™ subsystems that are both defined in the ZDT/IMS installation options module and currently active, that is, present in the z/OS Subsystem Vector Table (SSCVT), are included in the subsystem selection list. A subsystem defined in HFM1POPI will not appear in the selection list if it is not registered and active in z/OS at the time the Subsystem selection list is built.

Note: In DLI mode, the **LIST=ACTIVE** parameter is ignored. All defined subsystems are included in the selection list.

Default

All.

LKEYDATAC

```
LKEYDATAC=XXXXXXXX
```

LKEYDATAC

Specifies the SMS data class that ZDT/IMS uses when allocating the Extract logical key VSAM work file. The maximum length you can specify for LKEYDATAC is eight (8) bytes. LKEYDATAC is optional. If omitted the SMS data class specified in PDATAC is used, if supplied.

LKEYMGMTC

```
LKEYMGMTC=XXXXXXXX
```

LKEYMGMTC

Specifies the default SMS management class that ZDT/IMS uses when allocating the Extract logical key VSAM work file. The maximum length you can specify for LKEYMGMTC is eight (8) bytes. LKEYMGMTC is optional. If omitted the SMS management class specified in PMGMT is used, if supplied.

LKEYPQTY

```
LKEYPQTY= { 10 | nnnnnnnn }
```

LKEYPQTY

Specifies the amount of DASD space to be used for primary space allocation of the Extract logical key VSAM work file. The range depends on the space unit specified and the DASD device type.

LKEYSQTY

```
LKEYSQTY= { 10 | nnnnnnnn }
```

LKEYSQTY

Specifies the amount of DASD space to be used for secondary space allocation of the Extract logical key work file. The range depends on the space unit specified and the DASD device type.

LKEYSTORC

```
LKEYSTORC=XXXXXXXX
```

LKEYSTORC

Specifies the default SMS storage class that ZDT/IMS uses when allocating the Z Data Tools Extract logical key VSAM work file. The maximum length you can specify for LKEYSTORC is eight (8) bytes. LKEYSTORC is optional. If omitted the SMS storage class specified in PSTORC is used, if supplied.

LKEYSUNIT

```
LKEYSUNIT={CYL | REC | KB | MB | TRK}
```

LKEYSUNIT

Specifies the unit of primary and secondary space to be allocated to the Extract logical key VSAM work file.

REC

Record of average size.

KB

Kilobyte, a kilobyte is 1024 bytes.

MB

Megabyte, a megabyte is 1048576 bytes.

TRK

Track of a direct access storage device (DASD).

CYL

Cylinder of a DASD.

LKEYVOL*n*

```
LKEYVOLn=XXXXXX
```

LKEYVOL*n*

Where *n* is a number in the range 1 to 3, specifies the default serial number of the volume which is to contain the Extract logical key VSAM work file. This field should only be used for sites without SMS controlled data sets as SMS overrides this value.

LOADFREQ

```
LOADFREQ={(100, 100)|(nnnnn,mmmmm ) |ppppp}
```

LOADFREQ

Specifies the frequency of the automatic save function when running a Load job; that is, the Load checkpoint frequency.

ZDT/IMS:

- Increments a count by 1 each time a Load job inserts or replaces a segment.
- Issues a checkpoint when the count is equal to the Load checkpoint frequency.

The valid range is 1 to 99999.

When you specify two values in parentheses, separated by a comma, the first value is used for BMP mode and the second value is used for DLI mode. When you specify a single value, this is used for both BMP and DLI modes.

This parameter is optional. If it is not specified on the HFM1POPI macro statement, ZDT/IMS uses either the value specified on the HFM1POPD macro statement (if specified on that statement), or "(100,100)" (otherwise).

If you do not want the user to override the value you specify for this parameter, set ULOADFRQ=N.

LOCKMAX

```
LOCKMAX={ (,) | (nnnnn,mmmmm) | ppppp }
```

LOCKMAX

Specifies the maximum number of locks (in units of 1000) that a Z Data Tools/IMS function is allowed to hold at one time.

The valid range is 0 to 32767.

If you specify 0 for LOCKMAX there is no limit on the number of locks the function can hold at one time.

When you specify two values in parentheses, separated by a comma, the first value is used for BMP mode and the second value is used for DLI mode. When you specify a single value, this is used for both BMP and DLI modes.

This parameter is optional. If it is not specified on the HFM1POPI macro statement, ZDT/IMS uses either the value specified on the HFM1POPD macro statement (if specified on that statement), or a null value (otherwise). When a null value is passed to the IMS™ region controller, IMS™ uses:

- The value specified for the LOCKMAX parameter on the PSBGEN macro statement (if specified on that statement), or 0 (no limit) (otherwise).

If you do not want the user to override the value you specify for this parameter, set ULOCKMAX=N.

LOGDATAC

```
LOGDATAC=XXXXXXXX
```

LOGDATAC

Specifies the SMS data class that ZDT/IMS uses when allocating the IMS™ log. The maximum length you can specify for LOGDATAC is eight (8) bytes. LOGDATAC is optional. If omitted the SMS data class specified in PDATAC is used, if supplied.

LOGDSN

```
LOGDSN = { &&UID. . HFMLOG . &&YYMMDD. . &&HHMMSS. . &&DBNAME. | name-pattern }
```

LOGDSN

Specifies the name pattern that is used by ZDT/IMS functions to generate IMS™ log data set names.

The name pattern can consist of up to five qualifiers. For each qualifier you can specify either the required character string or a symbol. The data set name is generated by replacing the symbols in the name pattern with their runtime values.

For the remainder of this parameter description entry, "the function" refers to the ZDT/IMS function that is to use the IMS™ log.

The parameter supports these symbols:

&&DBNAME.

The name of the primary database that the function accesses.

&&PREFIX.

Either the user's TSO prefix (if the name is being generated in a TSO environment) or the value that is specified in the TSOPREFIX parameter (otherwise). The value might be null.

&&FUNCOD.

The code for the function. The following are the codes for the functions that might use an IMS™ log:

IE

Edit

IB

Browse

DIB

Initialize

IXB

Extract

ILB

Load

IPR

Print

IEB

Batch Edit

IBB

Batch Browse

&&HHMMSS.

Thhmmss where hhmmss is the time of day that the IMS™ log data set name is generated expressed in hours (HH), minutes (MM), and seconds (SS).

&&SSID.

The name of the IMS™ subsystem that the function is to access.

&&UID.

Either &&PREFIX. (if its value is not null) or the User ID (otherwise).

&&USER.

User ID

&&YYMMDD.

Dyymmdd where yymmdd is the date that the IMS™ log data set name is generated expressed as a 2-digit year (YY), month (MM), and day (DD).

This parameter is optional. If it is not specified on the HFM1POPI macro statement, ZDT/IMS uses either the value that is specified on the HFM1POPD macro statement (if specified on that statement), or &&UID..HFMLOG.&&YYMMDD..&&HHMMSS..&&DBNAME. (otherwise).

If you do not want users to override the value you specify for this parameter, set ULOGDSN=N.

LOGMGMTC

```
LOGMGMTC=XXXXXXXX
```

LOGMGMTC

Specifies the default SMS management class that ZDT/IMS uses when allocating the IMS™ Log. The maximum length you can specify for LOGMGMTC is eight (8) bytes. LOGMGMTC is optional. If omitted the SMS management class specified in PMGMT is used, if supplied.

LOGPQTY

```
LOGPQTY= { 20 | nnnnnnnn }
```

LOGPQTY

Specifies the amount of DASD space to be used for primary space allocation of the IMS™ Log. The range depends on the space unit specified and the DASD device type.

LOGSQTY

```
LOGSQTY= { 20 | nnnnnnnn }
```

LOGSQTY

Specifies the amount of DASD space to be used for secondary space allocation of the IMS™ Log. The range depends on the space unit specified and the DASD device type.

LOGSTORC

```
LOGSTORC=XXXXXXXX
```

LOGSTORC

Specifies the default SMS storage class that ZDT/IMS uses when allocating the IMS™ log. The maximum length is 8 bytes. LOGSTORC is optional. If omitted the SMS storage class specified in PSTORC is used, if supplied.

LOGSUNIT

```
LOGSUNIT={ TRK | REC | KB | MB | CYL }
```

LOGSUNIT

Specifies the unit of primary and secondary space to be allocated to the IMS™ Log.

REC

Record of average size.

KB

Kilobyte, a kilobyte is 1024 bytes.

MB

Megabyte, a megabyte is 1048576 bytes.

TRK

Track of a direct access storage device (DASD).

CYL

Cylinder of a DASD.

LOGUNIT

```
LOGUNIT=XXXXXXXX
```

LOGUNIT

Specifies the default permanent unit that ZDT/IMS when allocating the IMS™ Log. The maximum length you can specify for LOGUNIT is eight (8) bytes. LOGUNIT is optional. If omitted the SMS storage class specified in PUNIT is used.

LOGUSAGE

```
LOGUSAGE { KEEPUPD | KEEP | DELETE | NONE }
```

LOGUSAGE

Specifies whether ZDT/IMS functions which are using PSBs that have update intent use an IMS™ log when they are run in DLI mode and, if they do use a log, whether the log is kept when the function ends.

KEEP

Use an IMS™ log. Keep the log when the function ends.

KEEPUPD

Use an IMS™ log. Keep the log if the function updates databases or does not end normally.

DELETE

Use an IMS™ log. Keep the log only if the function does not end normally.

NONE

Do not use an IMS™ log.

This parameter is optional. If it is not specified on the HFM1POPI macro statement, ZDT/IMS uses either the value that is specified on the HFM1POPD macro statement (if specified on that statement), or KEEPUPD (otherwise).

If you do not want the user to override the value you specify for this parameter, set ULOGUSAG=N.

MACLIB

```
MACLIB={ IMS.SDFSMAC | maclib }
```

MACLIB

Specifies the name of the IMS™ macro library. ZDT/IMS uses the IMS™ macros in the specified library when dynamic PSBs are generated by the PSB Generation utility.

Note: The MACLIB library is not required when dynamic PSBs are generated by the the IMS Data Definition utility. See USEDDL on page 507.

This parameter is optional. If it is not specified on the HFM1POPI macro statement, ZDT/IMS uses either the value on the HFM1POPD macro statement (if one is specified), or IMS.SDFSMAC.

The user cannot override the value you specify for this parameter, when a function is run in BMP mode. If you do not want the user to override the value you specify for this parameter, when a function is run in DLI mode, set UMACLIB=N.

MAXGN

```
MAXGN={ (10000, 10000) | (nnnnnnnnn, mmmmmmmmmm ) | ppppppppp }
```

MAXGN

Specifies the maximum number of Get Next calls that a Find/Change command can issue for a database search.

When the MAXGN limit is reached, the Search Interrupt popup is displayed, giving the user the option of continuing or discontinuing the search.

The valid range is 0 to 2147483647.

If you specify 0 for MAXGN there is no limit on the number of Get Next calls that a Find/Change command can issue.

When you specify two values in parentheses, separated by a comma, the first value is used for BMP mode and the second value is used for DLI mode. When you specify a single value, this is used for both BMP and DLI modes.

This parameter is optional. If it is not specified on the HFM1POPI macro statement, ZDT/IMS uses either the value specified on the HFM1POPD macro statement (if specified on that statement), or "(10000,10000)" (otherwise).

The user cannot override the values you specify for this parameter.

PADS

```
PADS={N | Y}
```

PADS

Specifies whether Program Access to Data Sets (PADS) is used in the environments in which ZDT/IMS is run.

N

RACF® PADS is not used in the environments in which ZDT/IMS is run.

Y

RACF® PADS is used in the environments in which ZDT/IMS is run.

When dynamic PSBs are generated by the PSB Generation utility, ZDT/IMS stores them in temporary data sets with system-generated names when PADS=N is specified, and in the dynamic PSB library specified on the DYNPSB parameter when PADS=Y is specified.

This parameter can only be specified on the HFM1POPD macro statement.

This parameter is optional. The default is N.

The user cannot override the value you specify for this parameter.

PARDLI

```
PARDLI={0 | 1}
```

PARDLI

Specifies the parallel DL/I option used by ZDT/IMS functions running in BMP mode.

0

DL/I processing is performed in the BMP region.

1

All DL/I processing for the BMP is performed in the IMS™ control region. This prevents control region system 113 abends resulting from system X22 abends in the BMP region.

This parameter is optional. If it is not specified on the HFM1POPI macro statement, ZDT/IMS uses either the value specified on the HFM1POPD macro statement (if specified on that statement), or 0 (otherwise).

If you do not want the user to override the value you specify for this parameter, set UPARDLI=N.

Note: If you specify PARDLI=1, parallel DL/I is disabled. This can degrade performance.

For important information about the use of PARDLI and IMS™ control region abends, see PARDLI considerations on page 268.

PROCLIB

```
PROCLIB={ IMS.PROCLIB | proclib }
```

PROCLIB

Specifies the name of the IMS™ PROCLIB data set that contains the required DFSDFxxx member (see DFSDF on page 464).

The specified data set is allocated to the PROCLIB DD when a Z Data Tools/IMS function is run in DLI mode and IMS™ management of ACBs is enabled (ACBMGMT=CATALOG).

This parameter is optional. If it is not specified on the HFM1POPI macro statement, ZDT/IMS uses either the value on the HFM1POPD macro statement (if one is specified), or IMS.PROCLIB.

The user cannot override the value you specify for this parameter.

PROCOPTB

```
PROCOPTB={(G,G)|([{G |GO}],[{G |GO}]) |G |GO}
```

PROCOPTB

Specifies the PSB processing option (PROCOPT) that a dynamic PSB generated for a Browse uses to read the database.

G

The Browse reads with integrity

GO

The Browse reads without integrity

When you specify two values in parentheses, separated by a comma, the first value is used for BMP mode and the second value is used for DLI mode. When you specify a single value, this is used for both BMP and DLI modes.

This parameter is optional. If it is not specified on the HFM1POPI macro statement, ZDT/IMS uses either the value specified on the HFM1POPD macro statement (if specified on that statement), or "(G,G)" (otherwise).

If you do not want the user to override the value you specify for this parameter, set UPROCOPB=N.

PROCOPTP

```
PROCOPTP={(G,G)|([{G  | GO}],[{G  | GO}])  | G  | GO}
```

PROCOPTP

Specifies the PSB processing option (PROCOPT) that a dynamic PSB generated for a Print uses to read the database.

G

The Print reads with integrity

GO

The Print reads without integrity

When you specify two values in parentheses, separated by a comma, the first value is used for BMP mode and the second value is used for DLI mode. When you specify a single value, this is used for both BMP and DLI modes.

This parameter is optional. If it is not specified on the HFM1POPI macro statement, ZDT/IMS uses either the value specified on the HFM1POPD macro statement (if specified on that statement), or "(G,G)" (otherwise).

If you do not want the user to override the value you specify for this parameter, set UPROCOPP=N.

PROCOPTX

```
PROCOPTX={(G,G)|([{G  | GO}],[{G  | GO}])  | G  | GO}
```

PROCOPTX

Specifies the PSB processing option (PROCOPT) that a dynamic PSB generated for an Extract uses to read the database.

G

The Extract reads with integrity

GO

The Extract reads without integrity

When you specify two values in parentheses, separated by a comma, the first value is used for BMP mode and the second value is used for DLI mode. When you specify a single value, this is used for both BMP and DLI modes.

This parameter is optional. If it is not specified on the HFM1POPI macro statement, ZDT/IMS uses either the value specified on the HFM1POPD macro statement (if specified on that statement), or "(G,G)" (otherwise).

If you do not want the user to override the value you specify for this parameter, set UPROCOPX=N.

PROCOPTY

```
PROCOPTY={(G,G)|([{G | GO}],[{G | GO}]) | G | GO}
```

PROCOPTY

Specifies the PSB processing option (PROCOPT) that a dynamic PSB generated for a Batch Browse uses to read the database.

G

The Batch Browse reads with integrity.

GO

The Batch Browse reads without integrity.

When you specify two values in parentheses, separated by a comma, the first value is used for BMP mode and the second value is used for DLI mode. When you specify a single value, this is used for both BMP and DLI modes.

This parameter is optional. If it is not specified on the HFM1POPI macro statement, ZDT/IMS uses either the value specified on the HFM1POPD macro statement (if specified on that statement), or "(G,G)" (otherwise).

If you do not want the user to override the value you specify for this parameter, set UPROCOPY=N.

PSBLIB*n*

```
PSBLIB1={IMS.PSBLIB | dsname}
PSBLIBn=dsname
```

PSBLIB*n*

Where *n* is a number in the range 1 to 6, specifies the names of the load libraries that contain the program specification blocks (PSBs) that ZDT/IMS and IMS™ are to use.

The specified data sets are used when the ACBs are managed by your installation (ACBMGMT=ACBLIB).

These parameters are optional. But if you specify a PSBLIB parameter, you must also specify the PSBLIB parameters that precede it. So, for example, if you specify the PSBLIB3 parameter, you must also specify the PSBLIB1 parameter and the PSBLIB2 parameter.

If no PSBLIB parameters are specified on the HFM1POPI macro statement, ZDT/IMS uses either the PSBLIB values on the HFM1POPD macro statement (if specified), or IMS.PSBLIB.

If you do not want the user to override the values you specify for these parameters, set UPSBLIB=N.

Note: The PSBLIBn data sets are not required when IMS™ management of ACBs is enabled (ACBMGMT=CATALOG), as ZDT/IMS and IMS™ get the PSBs from the IMS™ catalog that the subsystem uses.

PSBTYPE

```
PSBTYPE={DYNAMIC | STATIC}
```

PSBTYPE

Specifies the type of PSB that the ZDT/IMS Extract, Load, Print, Batch Edit and Batch Browse functions use to access databases when no PSBTYPE= statement is specified in the HFMIMSIN input for the function.

DYNAMIC

The batch functions use a temporary PSB that they build when they are invoked.

STATIC

The batch functions use an existing PSB that the user specifies in the JCL.

This parameter can only be specified on the HFM1POPD macro statement.

This parameter is optional. The default is DYNAMIC.

If you do not want the user to override the value you specify for this parameter, set UPSBTYPE=N.

PSBTYPES

```
PSBTYPES={(BOTH,BOTH)|([{BOTH | STATIC | DYNAMIC}],[{BOTH | STATIC | DYNAMIC}]) | BOTH | STATIC | DYNAMIC}
```

PSBTYPES

Specifies the PSB types that can be used to access databases in the IMS™ subsystem.

BOTH

Both dynamic and static PSBs can be used.

STATIC

Only static PSBs can be used.

DYNAMIC

Only dynamic PSBs can be used.

When you specify two values in parentheses, separated by a comma, the first value is used for BMP mode and the second value is used for DLI mode. When you specify a single value, this is used for both BMP and DLI modes.

This parameter is optional. If it is not specified on the HFM1POPI macro statement, ZDT/IMS uses either the value specified on the HFM1POPD macro statement (if specified on that statement), or "(BOTH,BOTH)" (otherwise).

The user cannot override the value you specify for this parameter.

READONLY

```
READONLY={(N,N)|([{N | Y}],[{N | Y}]) | N | Y}
```

READONLY

Specifies whether or not the IMS™ subsystem is to be defined as Read Only.

When an IMS™ subsystem is defined read-only, only the ZDT/IMS non-update functions, that is, Browse, Batch Browse, Extract and Print, can be run against databases in the IMS™ subsystem; ZDT/IMS prevents the user running the update functions, that is, Edit, Batch Edit, Load, Initialize and Delete/Define.

N

The IMS™ subsystem is not to be defined as Read Only.

Y

The IMS™ subsystem is to be defined as Read Only.

When you specify two values in parentheses, separated by a comma, the first value is used for BMP mode and the second value is used for DLI mode. When you specify a single value, this is used for both BMP and DLI modes.

This parameter is optional. If it is not specified on the HFM1POPI macro statement, ZDT/IMS uses either the value specified on the HFM1POPD macro statement (if specified on that statement), or "(N,N)" (otherwise).

The user cannot override the value you specify for this parameter.

REGCATLG

```
REGCATLG={Y | N}
```

REGCATLG

Specifies whether the IMS™ catalog is registered with Database Recovery Control (DBRC).

Y

The IMS™ catalog is registered with DBRC. The IMS catalog partitions are defined with the DBRC utility (DSPURX00). The catalog partition definitions are stored in the RECON data sets.

N

The IMS™ catalog is not registered with DBRC. The IMS catalog partitions are defined with the IMS catalog partition definition utility (DFS3UCD00). The catalog partition definitions are stored in the catalog partition definition data set.

This parameter is used by ZDT/IMS functions run in DLI mode when IMS™ management of ACBs is enabled (ACBMGMT=CATALOG) or the database being accessed is the IMS catalog or its secondary index. When REGCATLG=Y, functions use DBRC and database data set names are obtained from the RECON data sets.

This parameter is optional. If it is not specified on the HFM1POPI macro statement, ZDT/IMS uses either the value on the HFM1POPD macro statement (if one is specified), or Y.

The user cannot override the value you specify for this parameter.

REGTYPES

```
REGTYPES={ BOTH  | BMP  | DLI }
```

REGTYPES

Specifies the modes in which databases in the IMS™ subsystem can be accessed.

BOTH

Databases in the IMS™ subsystem can be accessed in both BMP mode and DLI mode.

BMP

Databases in the IMS™ subsystem can be accessed in BMP mode only.

DLI

Databases in the IMS™ subsystem can be accessed in DLI mode only.

This parameter is optional. If it is not specified on the HFM1POPI macro statement, ZDT/IMS uses either the value specified on the HFM1POPD macro statement (if specified on that statement), or BOTH (otherwise).

The user cannot override the value you specify for this parameter.

RESLIB*n*

```
RESLIB1={ IMS.SDFSRESL | dsname }
RESLIBn=dsname
```

RESLIB*n*

Where *n* is a number in the range 1 to 6, specifies the names of the data sets containing the IMS™ SVC modules, the DFSMDA dynamic allocation modules and the IMS™ exit routines. When a Z Data Tools/IMS function accesses the subsystem, the specified data sets are allocated to a system generated DD name and those data sets that are APF-authorized are allocated to the DFSRESLB DD.

These parameters are optional. But if you specify a RESLIB parameter, you must also specify the RESLIB parameters that precede it. So, for example, if you specify the RESLIB3 parameter, you must also specify the RESLIB1 parameter and the RESLIB2 parameter.

If no RESLIB parameters are specified on the HFM1POPI macro statement, ZDT/IMS uses either the RESLIB values specified on the HFM1POPD macro statement (if specified on that statement), or IMS.SDFSRESL (otherwise).

The user cannot override the values you specify for these parameters when a function is run in BMP mode.

If you do not want the user to override the values you specify for these parameters, when a function is run in DLI mode, set URESLIB=N.

RKEYDATAC

```
RKEYDATAC=XXXXXXXX
```

RKEYDATAC

Specifies the SMS data class that ZDT/IMS uses when allocating the Extract root key VSAM work file. The maximum length you can specify for RKEYDATAC is eight (8) bytes. RKEYDATAC is optional. If omitted the SMS data class specified in PDATAC is used, if supplied.

RKEYMGMTC

```
RKEYMGMTC=XXXXXXXX
```

RKEYMGMTC

Specifies the default SMS management class that ZDT/IMS uses when allocating the Extract root key VSAM work file. The maximum length you can specify for RKEYMGMTC is eight (8) bytes. RKEYMGMTC is optional. If omitted the SMS management class specified in PMGMT is used, if supplied.

RKEYPQTY

```
RKEYPQTY= { 10 | nnnnnnnn }
```

RKEYPQTY

Specifies the amount of DASD space to be used for primary space allocation of the Extract root key VSAM work file. The range depends on the space unit specified and the DASD device type.

RKEYSQTY

```
RKEYSQTY= { 10 | nnnnnnnn }
```

RKEYSQTY

Specifies the amount of DASD space to be used for secondary space allocation of the Extract root key VSAM work file. The range depends on the space unit specified and the DASD device type.

RKEYSTORC

```
RKEYSTORC=XXXXXXXX
```

RKEYSTORC

Specifies the default SMS storage class that ZDT/IMS uses when allocating the Extract root key VSAM work file. The maximum length you can specify for RKEYSTORC is eight (8) bytes. RKEYSTORC is optional. If omitted the SMS storage class specified in PSTORC is used, if supplied.

RKEYSUNIT

```
RKEYSUNIT= [ { CYL | REC | KB | MB | TRK } ]
```

RKEYSUNIT

Specifies the unit of primary and secondary space to be allocated to the Extract root key VSAM work file.

REC

Record of average size.

KB

Kilobyte, a kilobyte is 1024 bytes.

MB

Megabyte, a megabyte is 1048576 bytes.

TRK

Track of a direct access storage device (DASD).

CYL

Cylinder of a DASD.

RKEYVOL*n*

```
RKEYVOLn=XXXXXX
```

RKEYVOL*n*

Where *n* is a number in the range 1 to 3, specifies the default serial number of the volume which is to contain the Extract root key VSAM work file. This field should only be used for sites without SMS controlled data sets as SMS overrides this value.

SDSNLOAD*n*

```
SDSNLOAD n = dsname
```

SDSNLOAD*n*

Where *n* is a number in the range 1 to 6, this parameter specifies the names of the data sets that contain the external subsystem load modules required for IMS BMP processing, such as Db2 ®attachment and related external subsystem support.

When a ZDT/IMS function accesses the IMS™ subsystem in BMP mode for Edit, the specified APF-authorized data sets are allocated to the DFSESL DD statement as a concatenation with the RESLIB*n* data sets.

The SDSNLOAD*n* parameters are optional. However, if you specify an SDSNLOAD parameter, you must also specify all preceding SDSNLOAD parameters.

Example

For example, If you specify the SDSNLOAD3 parameter, you must also specify SDSNLOAD1 and SDSNLOAD2.

Default

If no SDSNLOAD*n* parameters are specified on the HFM1POPI macro statement, ZDT/IMS uses the SDSNLOAD*n* values specified on the HFM1POPD macro statement, if present. Otherwise, no SDSNLOAD data sets are allocated to the DFSESL DD statement.

SKELLIB

```
SKELLIB = { HFM.SHFMSLIB | dsname }
```

SKELLIB

The name of the Z Data Tools target skeleton library.

This parameter is used in the HFM1FTEX job control skeleton. This parameter can only be specified on the HFM1POPD macro statement. The parameter is optional.

SSID

```
SSID=XXXX
```

SSID

Specifies the 1 to 4-character IMS™ subsystem identifier.

This parameter is required, and can only be specified on the HFM1POPI macro statement.

TIMEOUTI

```
TIMEOUTI= { 0000 | HHMM }
```

TIMEOUTI

Specifies the time interval that a user has to respond before an Edit/Browse BMP is timed out. The time interval is represented as zoned decimal digits of the form HHMM, where HH is hours and MM is minutes.

The specified time interval must not be greater than 24 hours and MM must be in the range 00 to 59.

If you specify 0000, then the BMP is not timed out.

This parameter is optional. If it is not specified on the HFM1POPI macro statement, ZDT/IMS uses either the value specified on the HFM1POPD macro statement (if specified on that statement), or "0000" otherwise.

The user cannot override the value you specify for this parameter.

TMINAME

```
TMINAME=tminame
```

TMINAME

If the IMS™ subsystem is part of a Remote Site Recovery (RSR) complex and the activity performed by ZDT/IMS functions running in DLI mode is to be tracked by RSR, specify the one to four-character Transport Manager Instance (TMI) name that the functions are to use.

This parameter is optional. If it is not specified on the HFM1POPI macro statement, ZDT/IMS uses either the value specified on the HFM1POPD macro statement (if specified on that statement), or a null value (otherwise).

When the activity is tracked by RSR and a null value is passed to the IMS™ region controller, IMS™ uses:

- The value specified for the TMINAME parameter on the IMSCTRL macro statement (if specified on that statement), or blanks (otherwise).

If you do not want the user to override the value you specify for this parameter, set URSR=N.

TPLLIB*n*

```
TPLLIB1={ HFM.TEMPLATE | dsname }
TPLLIBn=dsname
```

TPLLIB*n*

Where n is a number in the range 1 to 6, specifies the names of the data sets containing the templates that ZDT/IMS functions are to use.

These parameters are optional. But if you specify a TPLLIB parameter, you must also specify the TPLLIB parameters that precede it. So, for example, if you specify the TPLLIB3 parameter, you must also specify the TPLLIB1 parameter and the TPLLIB2 parameter.

If no TPLLIB parameters are specified on the HFM1POPI macro statement, ZDT/IMS uses either the TPLLIB values specified on the HFM1POPD macro statement (if specified on that statement), or HFM.TEMPLATE (otherwise).

If you do not want the user to override the values you specify for these parameters, set UTPLLIB=N.

UACBLIB

```
UACBLIB={ Y | N }
```

UACBLIB

Specifies whether or not the user can override the data set specified in the ACBLIB parameter.

Y

The user can override the ACBLIB parameter.

N

The user cannot override the ACBLIB parameter.

If you specify UACBLIB=N, the ACBLIB **Data set name** field on the DLI Mode Data Set 2 panel is protected.

This parameter is optional. If it is not specified on the HFM1POPI macro statement, ZDT/IMS uses either the value specified on the HFM1POPD macro statement (if specified on that statement), or Y (otherwise).

UAGNS

```
UAGNS={N | Y}
```

UAGNS

Specifies whether or not the IMS™ subsystem uses AGNs to secure dependent regions.

N

The IMS™ subsystem does not use AGNs to secure dependent regions.

Y

The IMS™ subsystem uses AGNs to secure dependent regions.

This parameter is optional. If it is not specified on the HFM1POPI macro statement, ZDT/IMS uses either the value specified on the HFM1POPD macro statement (if specified on that statement), or N (otherwise).

The user cannot override the value you specify for this parameter.

UAUTOSAV

```
UAUTOSAV={(N,Y)|([{Y | N}],[{Y | N}]) | Y | N}
```

UAUTOSAV

Specifies whether or not the user can override the values specified for the AUTOSAVE, CHGAFREQ and EDITFREQ parameters.

N

The user cannot override the parameters.

Y

The user can override the parameters.

When you specify two values in parentheses, separated by a comma, the first value is used for BMP mode and the second value is used for DLI mode. When you specify a single value, this is used for both BMP and DLI modes.

If you specify UAUTOSAV=N for BMP mode, the Autosave, Edit Checkpoint Frequency, and Change All/Repeat All Checkpoint Frequency fields on the BMP Mode Options panel are protected.

If you specify UAUTOSAV=N for DLI mode, the Autosave, Edit Checkpoint Frequency, and Change All/Repeat All Checkpoint Frequency fields on the DLI Mode Options panel are protected.

This parameter is optional. If it is not specified on the HFM1POPI macro statement, ZDT/IMS uses either the value specified on the HFM1POPD macro statement (if specified on that statement), or "(N,Y)" (otherwise).

UBUF

```
UBUF={Y | N}
```

UBUF

Specifies whether or not the user can specify the BUF parameter passed to the IMS™ region controller when ZDT/IMS functions run in DLI mode.

Y

The user can specify the BUF parameter.

N

The user cannot specify the BUF parameter.

If you specify UBUF=N:

- The **BUF** parameter field on the DLI Mode Parameters panel is protected.
- IMSBUF= statements in the HFMIMSIN input of batch jobs are ignored.
- No IMSBUF= statements are included in the JCL generated by ZDT/IMS dialogs.

This parameter is optional. If it is not specified on the HFM1POPI macro statement, ZDT/IMS uses either the value specified on the HFM1POPD macro statement (if specified on that statement), or Y (otherwise).

UDBDLIB

```
UDBDLIB={Y | N}
```

UDBDLIB

Specifies whether or not the user can override the data sets specified in the DBDLIBn parameters.

Y

The user can override the DBDLIBn data set parameters.

N

The user cannot override the DBDLIBn data set parameters.

If you specify UDBDLIB=N then:

- The DBDLIB Data set name fields on the PSB and DBD Data Sets panel are protected.
- DBDDSN=, DBDDSN2=, DBDDSN3=, DBDDSN4=, DBDDSN5= and DBDDSN6= statements in the HFMIMSIN input of batch jobs are ignored.
- No DBDDSN=, DBDDSN2=, DBDDSN3=, DBDDSN4=, DBDDSN5= and DBDDSN6= statements are included in the JCL generated by ZDT/IMS dialogs.

This parameter is optional. If it is not specified on the HFM1POPI macro statement, ZDT/IMS uses either the value specified on the HFM1POPD macro statement (if specified on that statement), or Y (otherwise).

UDBRC

```
UDBRC={Y | N}
```

UDBRC

Specifies whether or not the user can override the value specified for the DBRC parameter.

Y

The user can override the DBRC parameter.

N

The user cannot override the DBRC parameter.

If you specify UDBRC=N:

- The **DBRC** option field on the DLI Mode Parameters panel is protected.
- DBRC= statements in the HFMIMSIN input of batch jobs are ignored.
- No DBRC= statements are included in the JCL generated by ZDT/IMS dialogs.

This parameter is optional. If it is not specified on the HFM1POPI macro statement, ZDT/IMS uses either the value specified on the HFM1POPD macro statement (if specified on that statement), or Y (otherwise).

UDFSVSMP

```
UDFSVSMP={Y | N}
```

UDFSVSMP

Specifies whether or not the user can override the data set specified in the DFSVSAMP parameter and the member specified in the VSMPMEM parameter.

Y

The user can override the DFSVSAMP and VSMPMEM parameters.

N

The user cannot override the DFSVSAMP and VSMPMEM parameters.

If you specify UDFSVSMP=N:

- The DFSVSAMP **Data set name** and **Member** fields on the DLI Mode Data Sets 1 panel are protected.
- DFSVSAMP= and VSMPMEM= statements in the HFMIMSIN input of batch jobs are ignored.
- No DFSVSAMP= and VSMPMEM statements are included in the JCL generated by ZDT/IMS dialogs.

This parameter is optional. If it is not specified on the HFM1POPI macro statement, ZDT/IMS uses either the value specified on the HFM1POPD macro statement (if specified on that statement), or Y (otherwise).

UIEBFREQ

```
UIEBFREQ={(N,Y)|([{Y | N}],[{Y | N}]) | Y | N}
```

UIEBFREQ

Specifies whether or not the user can override the value specified for the IEBFREQ parameter.

N

The user cannot override the IEBFREQ parameter.

Y

The user can override the IEBFREQ parameter.

When you specify two values in parentheses, separated by a comma, the first value is used for BMP mode and the second value is used for DLI mode. When you specify a single value, this is used for both BMP and DLI modes.

If you specify UIEBFREQ=N for BMP mode:

- The **Batch Edit** checkpoint frequency field on the BMP Mode Options panel is protected.
- CHKPFREQ= statements in the HFMIMSIN input of Batch Edit jobs run in BMP mode are ignored.
- No CHKPFREQ= statements are included in the JCL generated by the Batch Edit dialog when the Batch Edit is to run in BMP mode.

If you specify UIEBFREQ=N for DLI mode:

- The **Batch Edit** checkpoint frequency field on the DLI Mode Options panel is protected.
- CHKPFREQ= statements in the HFMIMSIN input of Batch Edit jobs run in DLI mode are ignored.
- No CHKPFREQ= statements are included in the JCL generated by the Batch Edit dialog when the Batch Edit is to run in DLI mode.

This parameter is optional. If it is not specified on the HFM1POPI macro statement, ZDT/IMS uses either the value specified on the HFM1POPD macro statement (if specified on that statement), or "(N,Y)" (otherwise).

UIEFRDER

```
UIEFRDER{Y | N}
```

UIEFRDER

Specifies whether ZDT/IMS batch functions which are using PSBs that have update intent, use the IEFRDER DD specified in the JCL when they are run in DLI mode.

Y

Use the IEFRDER DD.

N

Do not use the IEFRDER DD.

If you specify UIEFRDER=N, then batch functions which are using PSBs that have update intent:

- Check whether the IEFRDER DD was allocated by the job step and, if it was, frees this allocation.
- Dynamically allocates an IEFRDER DD according to the LOG parameters specified in the HFMIMSIN input (when the parameters are specified and are not fixed) or the subsystem defaults for the LOG parameters (otherwise).

If you specify UIEFRDER=Y, then batch functions which are using PSBs that have update intent use the IEFRDER DD allocated by the job step (when the IEFRDER DD is allocated) or dynamically allocate an IEFRDER DD according to the LOG parameters specified in the HFMIMSIN input (when the parameters are specified and are not fixed) or the subsystem defaults for the LOG parameters (otherwise).

This parameter is optional. If it is not specified on the HFM1POPI macro statement, ZDT/IMS uses either the value that is specified on the HFM1POPD macro statement (if specified on that statement), or Y (otherwise).

UIMSBKO

```
UIMSBKO={Y | N}
```

UIMSBKO

Specifies whether or not the user can override the value specified for the IMSBKO parameter.

Y

The user can override the IMSBKO parameter.

N

The user cannot override the IMSBKO parameter.

If you specify UIMSBKO=N:

- The **Dynamic backout** option field on the DLI Mode Parameters panel is protected.
- IMSBKO= statements in the HFMIMSIN input of Load and Batch Edit jobs are ignored.
- No IMSBKO= statements are included in the JCL generated by the Load and Batch Edit dialogs.

This parameter is optional. If it is not specified on the HFM1POPI macro statement, ZDT/IMS uses either the value specified on the HFM1POPD macro statement (if specified on that statement), or Y (otherwise).

UIMSNBA

```
UIMSNBA={Y | N}
```

UIMSNBA

Specifies whether or not the user can override the values specified for the IMSNBA and IMSOBA parameters.

Y

The user can override the IMSNBA and IMSOBA parameters.

N

The user cannot override the IMSNBA and IMSOBA parameters.

If you specify UIMSNBA=N:

- The **NBA** and **OBA** Fast Path Buffer Allocation fields on the BMP Mode Parameters panel are protected.
- IMSNBA= and IMSOBA= statements in the HFMIMSIN input of batch jobs are ignored.
- No IMSNBA= and IMSOBA= statements are included in the JCL generated by ZDT/IMS dialogs.

This parameter is optional. If it is not specified on the HFM1POPI macro statement, ZDT/IMS uses either the value specified on the HFM1POPD macro statement (if specified on that statement), or Y (otherwise).

UIRLM

```
UIRLM={Y | N}
```

UIRLM

Specifies whether or not the user can override the value specified for the IRLM and IRLMNAME parameters.

Y

The user can override the IRLM and IRLMNAME parameters.

N

The user cannot override the IRLM and IRLMNAME parameters.

If you specify UIRLM=N:

- The **IRLM** option and the **IRLMNM** parameter fields on the DLI Mode Parameters panel are protected.
- IRLM= and IRLMNAME= statements in the HFMIMSIN input of batch jobs are ignored.
- No IRLM= and IRLMNAME= statements are included in the JCL generated by ZDT/IMS dialogs.

This parameter is optional. If it is not specified on the HFM1POPI macro statement, ZDT/IMS uses either the value specified on the HFM1POPD macro statement (if specified on that statement), or Y (otherwise).

ULOADFRQ

```
ULOADFRQ={(N,Y)|([{Y | N}],[{Y | N}]) | Y | N}
```

ULOADFRQ

Specifies whether or not the user can override the value specified for the LOADFREQ parameter.

N

The user cannot override the LOADFREQ parameter.

Y

The user can override the LOADFREQ parameter.

When you specify two values in parentheses, separated by a comma, the first value is used for BMP mode and the second value is used for DLI mode. When you specify a single value, this is used for both BMP and DLI modes.

If you specify ULOADFRQ=N for BMP mode:

- The **Load** checkpoint frequency field on the BMP Mode Options panel is protected.
- CHKPFREQ= statements in the HFMIMSIN input of Load jobs run in BMP mode are ignored.
- No CHKPFREQ= statements are included in the JCL generated by the Load dialog when the Load is to run in BMP mode.

If you specify ULOADFRQ=N for DLI mode:

- The **Load** checkpoint frequency field on the DLI Mode Options panel is protected.
- CHKPFREQ= statements in the HFMIMSIN input of Load jobs run in DLI mode are ignored.
- No CHKPFREQ= statements are included in the JCL generated by the Load dialog when the Load is to run in DLI mode.

This parameter is optional. If it is not specified on the HFM1POPI macro statement, ZDT/IMS uses either the value specified on the HFM1POPD macro statement (if specified on that statement), or "(N,Y)" (otherwise).

ULOCKMAX

```
ULOCKMAX={(N,Y)|([{Y | N}],[{Y | N}]) | Y | N}
```

ULOCKMAX

Specifies whether or not the user can override the value specified for the LOCKMAX parameter.

N

The user cannot override the LOCKMAX parameter.

Y

The user can override the LOCKMAX parameter.

When you specify two values in parentheses, separated by a comma, the first value is used for BMP mode and the second value is used for DLI mode. When you specify a single value, this is used for both BMP and DLI modes.

If you specify ULOCKMAX=N for BMP mode:

- The **LOCKMAX** parameter field on the BMP Mode Parameters panel is protected.
- LOCKMAX= statements in the HFMIMSIN input of batch jobs run in BMP mode are ignored.
- No LOCKMAX= statements are included in the JCL generated by ZDT/IMS dialogs when the job is to run in BMP mode.

If you specify ULOCKMAX=N for DLI mode:

- The **LOCKMAX** parameter field on the DLI Mode Parameters panel is protected.
- LOCKMAX= statements in the HFMIMSIN input of batch jobs run in DLI mode are ignored.
- No LOCKMAX= statements are included in the JCL generated by ZDT/IMS dialogs when the job is to run in DLI mode.

This parameter is optional. If it is not specified on the HFM1POPI macro statement, ZDT/IMS uses either the value specified on the HFM1POPD macro statement (if specified on that statement), or "(N,Y)" (otherwise).

This parameter is optional. If it is not specified on the HFM1POPI macro statement, ZDT/IMS uses either the value specified on the HFM1POPD macro statement (if specified on that statement), or "(N,Y)" (otherwise).

ULOGDSN

```
ULOGDSN { Y  | N }
```

ULOGDSN

Specifies whether the user can override the name pattern that is specified in the LOGDSN parameter.

Y

The user can override the LOGDSN parameter.

N

The user cannot override the LOGDSN parameter.

If you specify ULOGDSN=N:

- The **IMS log Data set name pattern** field on the **DLI Mode Data Sets 2** panel is protected.
- LOGDSN= statements in the HFMIMSIN input of batch jobs are ignored.
- No LOGDSN= statements are included in the JCL generated by ZDT/IMS dialogs.

This parameter is optional. If it is not specified on the HFM1POPI macro statement, ZDT/IMS uses either the value that is specified on the HFM1POPD macro statement (if specified on that statement), or Y (otherwise).

ULOGUSAG

```
ULOGUSAG { Y  | N }
```

ULOGUSAG

Specifies whether the user can override the value that is specified for the LOGUSAGE parameter.

Y

The user can override the LOGUSAGE parameter.

N

The user cannot override the LOGUSAGE parameter.

If you specify ULOGUSAG=N:

- The **IMS log option** field on the **DLI Mode Parameters** panel is protected.
- LOGUSAGE= statements in the HFMIMSIN input of batch jobs are ignored.
- No LOGUSAGE= statements are included in the JCL generated by ZDT/IMS dialogs.

This parameter is optional. If it is not specified on the HFM1POPI macro statement, ZDT/IMS uses either the value that is specified on the HFM1POPD macro statement (if specified on that statement), or Y (otherwise).

UMACLIB

```
UMACLIB={Y | N}
```

UMACLIB

Specifies whether or not the user can override the data set specified in the MACLIB parameter, when a function is run in DLI mode. (When a function is run in BMP mode, irrespective of the setting of this parameter, the user cannot override the data set specified in the MACLIB parameter.)

Y

The user can override the MACLIB parameter.

N

The user cannot override the MACLIB parameter.

If you specify UMACLIB=N:

- The IMS Macros **Data set name** field on the DLI Mode Data Sets 1 panel is protected.
- MACLIB= statements in the HFMIMSIN input of batch jobs are ignored.
- No MACLIB= statements are included in the JCL generated by ZDT/IMS dialogs.

This parameter is optional. If it is not specified on the HFM1POPI macro statement, ZDT/IMS uses either the value specified on the HFM1POPD macro statement (if specified on that statement), or Y (otherwise).

UPARDLI

```
UPARDLI={N | Y}
```

UPARDLI

Specifies whether or not the user can override the value specified in the PARDLI parameter.

N

The user cannot override the PARDLI parameter.

Y

The user can override the PARDLI parameter.

If you specify UPARDLI=N:

- The **PARDLI** option field on the BMP Mode Parameters panel is protected.
- PARDLI= statements in the HFMIMSIN input of batch jobs are ignored,
- No PARDLI= statements are included in the JCL generated by ZDT/IMS dialogs.

This parameter is optional. If it is not specified on the HFM1POPI macro statement, ZDT/IMS uses either the value specified on the HFM1POPD macro statement (if specified on that statement), or N (otherwise).

UPROCOPB

```
UPROCOPB={(N,Y)|({Y|N},{Y|N})|Y|N}
```

UPROCOPB

Specifies whether or not the user can override the value specified for the PROCOPTB parameter.

N

The user cannot override the PROCOPTB parameter.

Y

The user can override the PROCOPTB parameter.

When you specify two values in parentheses, separated by a comma, the first value is used for BMP mode and the second value is used for DLI mode. When you specify a single value, this is used for both BMP and DLI modes.

If you specify UPROCOPB=N for BMP mode, the **Browse** PSB Processing Option field on the BMP Mode Options panel is protected.

If you specify UPROCOPB=N for DLI mode, the **Browse** PSB Processing Option field on the DLI Mode Options panel is protected.

This parameter is optional. If it is not specified on the HFM1POPI macro statement, ZDT/IMS uses either the value specified on the HFM1POPD macro statement (if specified on that statement), or "(N,Y)" (otherwise).

UPROCOPP

```
UPROCOPP={(N,Y)|({Y|N},{Y|N})|Y|N}
```

UPROCOPP

Specifies whether or not the user can override the value specified for the PROCOPTP parameter.

N

The user cannot override the PROCOPTP parameter.

Y

The user can override the PROCOPTP parameter.

When you specify two values in parentheses, separated by a comma, the first value is used for BMP mode and the second value is used for DLI mode. When you specify a single value, this is used for both BMP and DLI modes.

If you specify UPROCOPP=N for BMP mode:

- The **Print** PSB Processing Option field on the BMP Mode Options panel is protected.
- PROCOPT= statements in the HFMIMSIN input of Print jobs run in BMP mode are ignored.
- No PROCOPT= statements are included in the JCL generated by the Print dialog when the Print is to run in BMP mode.

If you specify UPROCOPP=N for DLI mode:

- The **Print** PSB Processing Option field on the DLI Mode Options panel is protected.
- PROCOPT= statements in the HFMIMSIN input of Print jobs run in DLI mode are ignored.
- No PROCOPT= statements are included in the JCL generated by the Print dialog when the Print is to run in DLI mode.

This parameter is optional. If it is not specified on the HFM1POPI macro statement, ZDT/IMS uses either the value specified on the HFM1POPD macro statement (if specified on that statement), or "(N,Y)" (otherwise).

UPROCOPX

```
UPROCOPX={(N,Y)|({Y |N},{Y |N}) |Y |N}
```

UPROCOPX

Specifies whether or not the user can override the value specified for the PROCOPTX parameter.

N

The user cannot override the PROCOPTX parameter.

Y

The user can override the PROCOPTX parameter.

When you specify two values in parentheses, separated by a comma, the first value is used for BMP mode and the second value is used for DLI mode. When you specify a single value, this is used for both BMP and DLI modes.

If you specify UPROCOPX=N for BMP mode:

- The **Extract** PSB Processing Option field on the BMP Mode Options panel is protected.
- PROCOPT= statements in the HFMIMSIN input of Extract jobs run in BMP mode are ignored.
- No PROCOPT= statements are included in the JCL generated by the Extract dialog when the Extract is to run in BMP mode.

If you specify UPROCOPX=N for DLI mode:

- The **Extract** PSB Processing Option field on the DLI Mode Options panel is protected.
- PROCOPT= statements in the HFMIMSIN input of Extract jobs run in DLI mode are ignored.
- No PROCOPT= statements are included in the JCL generated by the Extract dialog when the Extract is to run in DLI mode.

This parameter is optional. If it is not specified on the HFM1POPI macro statement, ZDT/IMS uses either the value specified on the HFM1POPD macro statement (if specified on that statement), or "(N,Y)" (otherwise).

UPROCOPY

```
UPROCOPY={(N,Y)|({Y | N},{Y | N}) | Y | N}
```

UPROCOPY

Specifies whether or not the user can override the value specified for the PROCOPTY parameter.

N

The user cannot override the PROCOPTY parameter.

Y

The user can override the PROCOPTY parameter.

When you specify two values in parentheses, separated by a comma, the first value is used for BMP mode and the second value is used for DLI mode. When you specify a single value, this is used for both BMP and DLI modes.

If you specify UPROCOPY=N for BMP mode:

- The **Batch Browse** PSB Processing Option field on the BMP Mode Options panel is protected.
- PROCOPT= statements in the HFMIMSIN input of Batch Browse jobs run in BMP mode are ignored.
- No PROCOPT= statements are included in the JCL generated by the Batch Browse dialog when the Batch Browse is to run in BMP mode.

If you specify UPROCOPY=N for DLI mode:

- The **Batch Browse** PSB Processing Option field on the DLI Mode Options panel is protected.
- PROCOPT= statements in the HFMIMSIN input of Batch Browse jobs run in DLI mode are ignored.
- No PROCOPT= statements are included in the JCL generated by the Batch Browse dialog when the Batch Browse is to run in DLI mode.

This parameter is optional. If it is not specified on the HFM1POPI macro statement, ZDT/IMS uses either the value specified on the HFM1POPD macro statement (if specified on that statement), or "(N,Y)" (otherwise).

UPSBLIB

```
UPSBLIB={Y | N}
```

UPSBLIB

Specifies whether or not the user can override the data sets specified in the PSBLIBn parameters.

Y

The user can override the PSBLIBn data set parameters.

N

The user cannot override the PSBLIBn data set parameters.

If you specify UPSBLIB=N:

- The PSBLIB **Data set name** fields on the PSB and DBD Data Sets panel are protected.
- PSBDSN=, PSBDSN2=, PSBDSN3=, PSBDSN4=, PSBDSN5= and PSBDSN6= statements in the HFMIMSIN input of batch jobs are ignored.
- No PSBDSN=, PSBDSN2=, PSBDSN3=, PSBDSN4=, PSBDSN5= and PSBDSN6= statements are included in the JCL generated by ZDT/IMS dialogs.

This parameter is optional. If it is not specified on the HFM1POPI macro statement, ZDT/IMS uses either the value specified on the HFM1POPD macro statement (if specified on that statement), or Y (otherwise).

UPSBTYPE

```
UPSBTYPE={Y | N}
```

UPSBTYPE

Specifies whether or not the user can override the value specified in the PSBTYPE parameter.

Y

The user can override the PSBTYPE parameter.

N

The user cannot override the PSBTYPE parameter.

This parameter can only be specified on the HFM1POPD statement.

If you specify UPSBTYPE=N:

- PSBTYPE= statements in the HFMIMSIN input of batch jobs are ignored.
- No PSBTYPE= statements are included in the JCL generated by ZDT/IMS dialogs.

This parameter is optional. The default is Y.

URECON

```
URECON={Y | N}
```

URECON

Specifies whether or not the user can specify RECON data sets.

Y

The user can specify RECON data sets.

N

The user cannot specify RECON data sets.

If you specify URECON=N:

- The RECON **Primary data set**, **Secondary**, and **Spare** data set name fields on the DLI Mode Data Sets 2 panel are protected.
- RECON1=, RECON2= and RECON3= statements in the HFMIMSIN input of batch jobs are ignored.
- No RECON1=, RECON2= and RECON3= statements are included in the JCL generated by ZDT/IMS dialogs.

This parameter is optional. If it is not specified on the HFM1POPI macro statement, ZDT/IMS uses either the value specified on the HFM1POPD macro statement (if specified on that statement), or Y (otherwise).

URESLIB

```
URESLIB={Y | N}
```

URESLIB

Specifies whether or not the user can override the data sets specified in the RESLIBn parameters, when a function is run in DLI mode. (When a function is run in BMP mode, irrespective of the setting of this parameter, the user cannot override the data sets specified in the RESLIB*n* parameters.)

Y

The user can override the RESLIBn data set parameters.

N

The user cannot override the RESLIBn data set parameters.

If you specify URESLIB=N:

- The RESLIB Data set name fields on the DLI Mode Data Sets 1 panel are protected.
- RESLIB1=, RESLIB2=, RESLIB3=, RESLIB4=, RESLIB5= and RESLIB6= statements in the HFMIMSIN input of batch jobs are ignored.
- No RESLIB1=, RESLIB2=, RESLIB3=, RESLIB4=, RESLIB5= and RESLIB6= statements are included in the JCL generated by ZDT/IMS dialogs.

This parameter is optional. If it is not specified on the HFM1POPI macro statement, ZDT/IMS uses either the value specified on the HFM1POPD macro statement (if specified on that statement), or Y (otherwise).

URSR

```
URSR={Y | N}
```

URSR

Specifies whether or not the user can override the values specified in the GSGNAME and TMINAME parameters.

Y

The user can override the GSGNAME and TMINAME parameters.

N

The user cannot override the GSGNAME and TMINAME parameters.

If you specify URSR=N:

- The **GSGNAME** and **TMINAME** parameter fields on the DLI Mode Parameters panel are protected.
- GSGNAME= and TMINAME= statements in the HFMIMSIN input of batch jobs are ignored.
- No GSGNAME= and TMINAME= statements are included in the JCL generated by ZDT/IMS dialogs.

This parameter is optional. If it is not specified on the HFM1POPI macro statement, ZDT/IMS uses either the value specified on the HFM1POPD macro statement (if specified on that statement), or Y (otherwise).

USEDDL

```
USEDDL={Y | N}
```

USEDDL

Specifies whether or not ZDT/IMS uses the IMS Data Definition utility (DFS3ID00) to generate dynamic PSBs when IMS management of ACBs is enabled (ACBMGMT=CATALOG).

Y

Dynamic PSBs are generated by submitting Data Definition Language (DDL) SQL statements to the Data Definition utility.

N

ZDT/IMS does not use the Data Definition utility.

These utilities are used instead:

- The PSB Generation utility
- The IMS Catalog Library Builder utility (DFS3LU00)
- The ACB Maintenance utility
- The IMS Catalog Populate utility (DFS3PU00)

This method of generating dynamic PSBs has been deprecated, so using it is not recommended.

The default and recommended setting is USEDDL=Y.

Note:

- Support for running the Data Definition utility in a BMP region was delivered with APAR PH45367/PTF UI81894 and enabled by IMS 15.3 and support for running it in a DL/I batch region was delivered with APAR PH51761/PTF UI91330 and enabled by IMS 15.4. Ensure you are at or above these levels to use the Data Definition utility.
- The Data Definition utility can only be used to generate dynamic PSBs when IMS management of ACBs is enabled (ACBMGMT=CATALOG). When ACBs are managed by your installation (ACBMGMT=ACBLIB), the value specified for this parameter is ignored.

This parameter is optional. If it is not specified on the HFM1POPI macro statement, ZDT/IMS uses either the value specified on the HFM1POPD macro statement (if specified on that statement), or Y (otherwise).

UTPLLIB

```
UTPLLIB={Y | N}
```

UTPLLIB

Specifies whether or not the user can override the data sets specified in the TPLLIBn parameters.

Y

The user can override the TPLLIBn data set parameters.

N

The user cannot override the TPLLIBn data set parameters.

If you specify UTPLLIB=N:

- The Template data set name fields on the Template Data Sets panel are protected.
- The TCINDD DD statements in the Batch Edit and Batch Browse JCL is ignored.
- No TCINDD DD statements are included in the JCL generated by the Batch Edit and Batch Browse dialogs.

This parameter is optional. If it is not specified on the HFM1POPI macro statement, ZDT/IMS uses either the value specified on the HFM1POPD macro statement, (if specified on that statement), or Y (otherwise).

VCURUDT

```
VCURUDT = {C | R}
```

VCURUDT

Specifies the usage rules that apply to views and criteria sets when the function generates a template for the database.

C

A view or criteria set can be used with a database only if it provides one or more layouts for each segment in the database. If it does provide one or more layouts for each segment in the

database, the function uses a composite view or criteria set that is created by adding the view or criteria set's selection information to the template that was generated for the database.

R

A view or criteria set can be used with a database only if it has the same fields as the template that was generated for the database. If it does have the same fields as the generated template, the function uses the view or criteria set that is specified by the user.

VCURUDT=C is the default setting and the recommended setting.

This parameter is optional. If it is not specified on the HFM1POPI macro statement, ZDT/IMS uses either the value specified on the HFM1POPD macro statement (if specified on that statement), or C (otherwise).

The user cannot override the value you specify for this parameter.

VCURULE

```
VCURULE={T | C | R | S}
```

VCURULE

Specifies the view or criteria set usage rule.

C

A view or criteria set can be used with a database only if it provides one or more layouts for each segment in the database. If it does provide one or more layouts for each segment in the database, the function uses a composite view or criteria set that is created by adding the view or criteria set's selection information to the static template for the database.

R

A view or criteria set can be used with a database only if it has the same fields as the static template for the database. If it does have the same fields as the static template for the database, the function uses the view or criteria set that is specified by the user.

S

A view or criteria set can be used with a database only if it provides one or more layouts for each segment in the database. If it does provide one or more layouts for each segment in the database, the function uses the view or criteria set specified by the user.

The static template for the database is not used.

T

A view or criteria set can be used with a database only if these two conditions are true:

- The view or criteria set provides one or more layouts for each segment in the database.
- The view or criteria set was created for the database OR the view or criteria set and the static template for the database have the same fields.

If the view or criteria set satisfies these conditions, the function uses the view or criteria set that is specified by the user.

Note: VCURULE=T is the default setting, but VCURULE=C is the recommended setting.

This parameter is optional. If it is not specified on the HFM1POPI macro statement, ZDT/IMS uses either the value specified on the HFM1POPD macro statement (if specified on that statement), or T (otherwise).

The user cannot override the value you specify for this parameter.

VSMPMEM

```
VSMPMEM={ DFSVSM00 | member }
```

VSMPMEM

If the data set specified in the DFSVSAMP parameter is a PDS or PDSE, VSMPMEM specifies the member of that data set which contains the parameters defining the IMS™ buffer subpools for OSAM and VSAM data sets.

The specified member is allocated to the DFSVSAMP DD when ZDT/IMS functions are run in DLI mode.

This parameter is optional. If it is not specified on the HFM1POPI macro statement and the data set specified in the DFSVSAMP parameter is a PDS or a PDSE, ZDT/IMS uses either the value specified on the HFM1POPD macro statement (if specified on that statement), or DFSVSM00 (otherwise).

If you do not want the user to override the value you specify for this parameter, set UDFSVSMP=N.

XDOPTLB

```
XDOPTLB={ N | Y }
```

Note: APAR PH17975 makes setting the XDOPTLB parameter unnecessary. The following sections describe how XDOPTLB works when APAR PH17975 is not applied, and explain a check that you might need to perform when APAR PH17975 is applied.

XDOPTLB parameter when APAR PH17975 is not applied

This section describes the effect of the XDOPTLB parameter when APAR PH17975 has *not* been applied.

XDOPTLB

Specifies whether the ACB Maintenance utility deletes all PSBs and DBDs from the DOPT ACBLIB data set and makes all of their space available for reuse, before building blocks for the dynamic PSB.

N

ZDT/IMS deletes the PSB from the DOPT ACBLIB data set after the BMP terminates. No other modules are deleted from the DOPT ACBLIB data set and no space is made available for reuse.

Y

ZDT/IMS executes the ACB Maintenance utility with BUILD PSB=ALL specified on the SYSIN control statement. With this control statement specified, the utility deletes all PSBs and DBDs (and any other modules) from the DOPT ACBLIB data set and makes all of their space available for reuse, before building control blocks for the dynamic PSB.

If this option is selected, you must specify a DOPT ACBLIB that is only used by ZDT/IMS.

This parameter is optional. If it is not specified on the HFM1POPI macro statement, ZDT/IMS uses either the value on the HFM1POPD macro statement (if one is specified), or N.

The user cannot override the value you specify for this parameter.

XDOPTLB parameter when APAR PH17975 is applied

When dynamic PSBs are generated by the IMS Data Definition utility (see USEDDL on page 507), ZDT/IMS does not use the DOPT ACBLIB data set, so no action is required.

When APAR PH17975 has been applied, ZDT/IMS always follows the behavior for XDOPTLB=Y. Therefore, if XDOPTLB was previously set to Y, no action is required now.

If XDOPTLB was previously set to N, check whether the DOPT ACBLIB data set specified in the DYNACB parameter is used by any other product.

If the DOPT ACBLIB data set is used only by ZDT/IMS, no action is required now. If housekeeping procedures were in place for the DOPT ACBLIB they will no longer be required.

If a product other than ZDT/IMS also uses the DOPT ACBLIB data set, do the following steps:

1. Provide a new DOPT ACBLIB data set that only ZDT/IMS will use.
2. Change the DYNACB parameter on the HFM1POPI macro statement, to specify the name of the new DOPT ACBLIB data set.
3. Reassemble and link-edit the ZDT/IMS installation options module.
4. If the ACBs are managed by your installation (ACBMGMT=ACBLIB), concatenate the new DOPT ACBLIB data set with the ACBLIB DD in the IMS™ execution JCL (see Providing a DOPT ACBLIB data set on page 263).

XKEYDATAC

XKEYDATAC=*XXXXXXXX*

XKEYDATAC

Specifies the SMS data class that ZDT/IMS uses when allocating the Extract keys file. The maximum length you can specify for XKEYDATAC is eight (8) bytes. XKEYDATAC is optional. If omitted the SMS data class specified in PDATAC is used, if supplied. This parameter can only be specified on the HFM1POPD macro statement.

XKEYMGMTC

```
XKEYMGMTC=XXXXXXXX
```

XKEYMGMTC

Specifies the default SMS management class that ZDT/IMS uses when allocating the Extract keys file. The maximum length you can specify for XKEYMGMTC is eight (8) bytes. XKEYMGMTC is optional. If omitted the SMS management class specified in PMGMT is used, if supplied. This parameter can only be specified on the HFM1POPD macro statement.

XKEYPQTY

```
XKEYPQTY= { 1 | nnnnnnnn }
```

XKEYPQTY

Specifies the amount of DASD space to be used for primary space allocation of the Extract keys file. The range depends on the space unit specified and the DASD device type. This parameter can only be specified on the HFM1POPD macro statement.

XKEYSQTY

```
XKEYSQTY= { 20 | nnnnnnnn }
```

XKEYSQTY

Specifies the amount of DASD space to be used for secondary space allocation of the Extract keys file. The range depends on the space unit specified and the DASD device type. This parameter can only be specified on the HFM1POPD macro statement.

XKEYSTORC

```
XKEYSTORC=XXXXXXXX
```

XKEYSTORC

Specifies the default SMS storage class that ZDT/IMS uses when allocating the Extract keys file. The maximum length you can specify for XKEYSTORC is eight (8) bytes. XKEYSTORC is optional. If omitted the SMS storage class specified in PSTORC is used, if supplied. This parameter can only be specified on the HFM1POPD macro statement.

XKEYSUNIT

```
XKEYSUNIT= { TRK | REC | KB | MB | CYL }
```

XKEYSUNIT

Specifies the unit of primary and secondary space to be allocated to the Extract keys file. This parameter can only be specified on the HFM1POPD macro statement.

REC

Record of average size.

KB

Kilobyte, a kilobyte is 1024 bytes.

MB

Megabyte, a megabyte is 1048576 bytes.

TRK

Track of a direct access storage device (DASD).

CYL

Cylinder of a DASD.

XKEYUNIT

```
XKEYUNIT=XXXXXXXX
```

XKEYUNIT

Specifies the default permanent unit that ZDT/IMS uses when allocating the Extract keys file. The maximum length you can specify for XKEYUNIT is eight (8) bytes. XKEYUNIT is optional. If omitted the SMS storage class specified in PUNIT is used. This parameter can only be specified on the HFM1POPD macro statement.

HFM1AGNT macro

AGN

```
AGN=agn_id
```

AGN

Specifies the 1- to 8-character AGN (Application Group Name) that you want ZDT/IMS to use when it starts a BMP region.

This parameter is required.

DESC

```
DESC= [AGN description]
```

DESC

Specifies a description of the AGN, to be displayed on the AGN Selection panel. The maximum length of the description is 45 characters. The description should be enclosed in quotation marks.

The parameter is optional. The default is no description.

SSID

```
SSID=nnnn
```

SSID

Specifies the 1- to 4-character identifier of the subsystem that the AGN is for. You can only specify HFM1AGNT macros for subsystems that you have provided HFM1POPI macro statement for.

This parameter is required.

Appendix D. Z Data Tools options specified in PARMLIB members

This section describes the Z Data Tools options specified in PARMLIB members. You can modify these Z Data Tools options to suit your requirements.

Z Data Tools options specified in HFM0PARM

This section describes the Z Data Tools options specified in parmlib member HFM0PARM. You can modify these Z Data Tools options to suit your requirements.

The HFM0PARM member provides a secure location for sensitive Z Data Tools parameters used to control, for example, audit logging. Z Data Tools uses system macros to locate this member in the current logical parmlib concatenation, making it difficult to over-ride or bypass the contents, which can occur when using the HFM0POPT module. When used to specify SAF- rule controlled audit logging, particularly to SMF, a high level of integrity over the auditing process is possible. The HFM0PARM member can also be used to force the use of a particular version of the HFM0POPT module, by specifying a specific data set from which the module should be loaded. This can be used to prevent a knowledgeable user creating a private version of the HFM0POPT module and loading it from a data set allocated to their TSO session, thereby bypassing installation-specified parameters.

This section includes the description of the options that can be specified in the HFM0PARM member, followed by a description of the facilities that can be used to customize the HFM0PARM member. Customization is entirely optional, but might be necessary, for example, in a sysplex environment where not all z/OS® images have the same auditing requirements. It is also possible to specify different auditing options for different users using HFM0PARM statements; these changes can be made directly to the HFM0PARM member. When changes are made to the HFM0PARM member, no IPL is required and the changes become active the next time the user starts Z Data Tools.

FMAUDIT

SAF_CTRL

```
SAF_CTRL= { YES | NO }
```

SAF_CTRL

Specifies whether Z Data Tools should use SAF to control all aspects of Z Data Tools audit logging. See Preparing to customize Z Data Tools on page 23 and Customizing Z Data Tools to write audit records to SMF on page 86 for more information. SAF_CTRL is optional. The default is "YES", to use SAF.

Note: When SAF-rule controlled auditing is in effect (SAF_CTRL=YES), Z Data Tools ignores the values specified for the AUDITLOG and SMFNO parameters on the HFM0POPI macro statement.

SMF_NO

```
SMF_NO= { 0 | number }
```

SMF_NO

Specifies the SMF record type for audit logging, where *number* is record type. You can specify a value from 128 to 255, or 0. The default is 0. If you specify an invalid SMF record number then auditing to SMF is inoperative.

FMOPTMOD

DSNAME

```
DSNAME=data_set_name
```

DSNAME

Specifies the name of the library from which Z Data Tools is to load the Z Data Tools options module - usually HFM0POPT. Note that the member keyword can be used to specify an alternative name to HFM0POPT, if required.

If the DSNAME keyword is specified, Z Data Tools bypasses the usual library search when loading the Z Data Tools options module and only searches the specified library for the module.

When the DSNAME keyword is omitted, Z Data Tools searches load libraries allocated to the user's session in the normal order, for example, LIBDEF, ISPLLIB, STEPLIB, JOBLIB, link list.

One use of this option is in an environment with stringent security requirements, where it is necessary to ensure that all Z Data Tools users access a particular Z Data Tools options module, and that the use of this option module cannot be avoided.

The DSNAME keyword is optional.

MEMBER

```
MEMBER=member
```

MEMBER

Specifies the name of the member that contains the Z Data Tools options module - usually HFM0POPT. The MEMBER keyword is optional.

Facilities for customizing the HFM0PARM definitions

You can use the HFM0PARM definitions to provide different options for Z Data Tools users, based on the following environmental settings:

- The current z/OS® system ID
- The current version of Z Data Tools
- The user's logonid

You can also split the HFM0PARM definitions across two or more members using the % (include) specification, and include comments using the * specification.

All of the facilities described in this section are optional. You do not need to use any of them.

Tags

A tag is an entry in the HFM0PARM member used to specify different options based on the current settings of certain environment variables. The following table shows the tags that can be specified.

Table 74. Z Data Tools Parmlib member tag types

Tag Type	Priority	Description	Usage
Z	1	z/OS® system	In a sysplex environment with multiple z/OS® images.
V	2	ZDT Version	In a environment with multiple concurrent versions of Z Data Tools.
U	3	User's logonid	Different options for specific users, typically by exception.

Rules for specifying tags

- Each tag should be entered on a separate line.
- Every initial tag must have a corresponding end tag. The initial tag is called the *"start tag"*.
- A start tag begins with a '<' and ends with a '>' character.
- The first character of the line containing a start tag must be '<', followed by the tag type.
- An end tag is specified as '<Ex>' or '</x>, where x is the tag type.
- The first character of the line containing an end tag must be '<', followed immediately by the rest of the end tag.
- One or more values can be specified for each tag. Separate values using a space or comma. Do not specify an '=' after the tag type.
- Values specified for each tag can include the * wildcard character. When specified, the '*' matches 0, 1, or many characters. Example: F* matches F, FRED, FUNNY, but not AFTER. The ** specification and other wildcard characters are not supported.
- Tags should be specified in order of priority to ensure correct processing.

Example of tag usage

Tags can be used to specify different HFM0PARM options based on certain environment variables (the z/OS® system ID, the Z Data Tools version and the user's TSO logonid). The most usual usage of tags is to provide an exception to the normal processing, for some particular set of circumstances.

Example 1:

Suppose SAF-rule controlled auditing is turned off for all users in a single z/OS-image. The HFM0PARM member would be coded as:

```
FMAUDIT SAF_CTRL=NO
```

Now suppose that there is a requirement to implement SAF-rule controlled auditing, and that one particular logonid (TEST1) has been selected for testing purposes. Assuming all the relevant SAF rules have been written, SAF-rule controlled auditing can be turned on for logonid TEST1 using:

```
FMAUDIT SAF_CTRL=NO     (1)
<U TEST1>          (2)
FMAUDIT SAF_CTRL=YES      (3)
<EU>             (4)
```

Explanation:

When Z Data Tools parses the above, the line (1) is processed for all users, and turns SAF-rule controlled auditing off. When line (2) is processed, the user's TSO logonid is compared with the value TEST1. For all users other than TEST1, the test fails. When a tag fails to match a condition, all lines from the start tag to the matching end tag (inclusive) are ignored, so the net result is that, in the above example, only line (1) is processed. This ensures that the default of no SAF-rule controlled auditing applies to all users other than TEST1.

For user TEST1 however, the tag comparison on line (2) matches, so line (3) is included. Z Data Tools re-processes the FMAUDIT statement for line (3), this time with SAF_CTRL=YES, resulting in SAF-rule controlled auditing being turned on for user TEST1.



Note: This example demonstrates two important principles when processing the statements within an HFM0PARM member:

- Multiple statements for the same option are allowed.
- If multiple statements for the same option are encountered, the last statement processed is the one that determines the setting.

Example 2:

Suppose in a sysplex environment there are nine z/OS® images, with system IDs SYS1, SYS2, SYS3 … SYS9. Z Data Tools is available on all images, and SAF-rule controlled auditing is not required on SYS1, SYS2, … SYS7 inclusive, but is required on SYS8 and SYS9. Further, on z/OS® system SYS8, only TSO logonids that commence with DEV should be subject to SAF rule controlled audit; all other TSO logonids are exempt. On system SYS9 however, all TSO logonids should be subject to SAF-rule controlled audit, with the single exception of TSO logonid MAIN1.

This could be coded as follows:

```
FMAUDIT SAF_CTRL=NO     (1)
<Z SYS8>            (2)
<U DEV*>            (3)
FMAUDIT SAF_CTRL=YES      (4)
</U>              (5)
```

```
</Z>              (6)
<Z SYS9>            (7)
FMAUDIT SAF_CTRL=YES      (8)
<U MAIN1>         (9)
FMAUDIT SAF_CTRL=NO     (10)
</U>              (11)
</Z>              (12)
```

Explanation:

Line (1) sets the default, which is not to use SAF- rule controlled auditing.

For all users on z/OS® systems SYS1 … SYS7 inclusive, the Z tags for SYS8 on line (2) and SYS9 on line (7) will not match the current environment, resulting in lines 2-12 inclusive being ignored.

For users running on z/OS® system SYS8, line (2)-(6) inclusive are considered; lines (7)-(12) are ignored. Any TSO logonid that does not start with DEV (for example, PROD1) will not match the U tag (line 3), resulting in lines 3-5 inclusive being ignored. This leaves only line (1) to consider, which sets SAF-rule controlled auditing off. For a TSO logonid such as DEV76, the U tag on line (3) matches, so line (4) is included. Z Data Tools processes the FMAUDIT statement on line (4) and sets SAF-rule controlled auditing on.

For users running on z/OS® system SYS9, lines (2)-(6) inclusive are ignored; lines (7)-(12) inclusive are considered. Line (8) changes the default (for all users on system SYS9) to use SAF- rule controlled auditing. For all users other than TSO logonid MAIN1, lines (9)-(11) are ignored, resulting in the new default (line 8) being used. This turns SAF-rule controlled auditing on. For TSO logonid MAIN1 only, the U tag on line (9) matches and the FMAUDIT statement on line (10) is included. This turns SAF-rule controlled auditing off for user MAIN1.

Included members

You can divide the HFM0PARM across multiple members using the include specification. The format for an included member is as follows:

- The first character of the line must be a '%' character.
- Specify the member name to be included after the % character.
- Specify a single member name only after the % character.

Multiple levels of include nesting are supported, circular (recursive) definitions will result in an error and Z Data Tools will not initialize.

Empty included members are ignored.

Example 1:

Suppose in a sysplex environment there are 4 z/OS® images, with system IDs SYS1, SYS2, SYS3, and SYS4. Z Data Tools is available on all images, with different auditing requirements on each image.

Include members can be used to code this as follows:

Member HFM0PARM

```
<Z SYS1>
%HFMPSYS1
</Z>
<Z SYS2>
%HFMPSYS2
</Z>
<Z SYS3>
%HFMPSYS3
</Z>
<Z SYS4>
%HFMPSYS4
</Z>
```

Member HFMPSYS1

```
FMAUDIT SAF_CTRL=YES
```

Member HFMPSYS2

```
FMAUDIT SAF_CTRL=NO
```

Member HFMPSYS3

```
FMAUDIT SAF_CTRL=NO
```

Member HFMPSYS4

```
FMAUDIT SAF_CTRL=YES
```

This has the advantage of consolidating all the options for each z/OS® image in a specific member.

If you used Method 1 to add member HFM0PARM to the logical parmlib concatenation (see Defining the HFM0PARM member on page 92 for more information), be careful to avoid any conflicts between included member names and existing member names in the logical parmlib concatenation.

Comments

You can add comments to the HFM0PARM member as follows:

- The first character of a comment line must be a '*' character.
- The rest of the comment line is ignored.

Continuing specifications across multiple lines

You can specify the Z Data Tools options across multiple lines as follows:

- The last character of each line to be continued must be a comma.
- Continue the text on the next line starting at any position (1-71).

Example:

```
FMAUDIT SAF_CTRL=YES
FMOPTMOD  DSNAME=ZDTOOLS.OTHER.OPTIONS,
    MEMBER=HFM0POPX
```

ZDT/IMS options specified in HFM1PARM

This section describes the ZDT/IMS options specified in parmlib member HFM1PARM. You can modify these ZDT/IMS options to suit your requirements.

The HFM1PARM member provides a secure location for sensitive ZDT/IMS parameters used to control, for example, audit logging. ZDT/IMS uses system macros to locate this member in the current logical parmlib concatenation, making it difficult to over-ride or bypass the contents, which can occur when using the HFM1POPT module. When used to specify SAF-rule controlled audit logging, particularly to SMF, a high level of integrity over the auditing process is possible. The HFM1PARM member can also be used to force the use of a particular version of the HFM1POPT module, by specifying a specific data set from which the module should be loaded. This can be used to prevent a knowledgeable user creating a private version of the HFM1POPT module and loading it from a data set allocated to their TSO session, thereby bypassing installation-specified parameters.

This section includes the description of the options that can be specified in the HFM1PARM member, followed by a description of the facilities that can be used to customize the HFM1PARM member. Customization is entirely optional, but might be necessary, for example, in a sysplex environment where not all z/OS® images have the same auditing requirements. It is also possible to specify different auditing options for different users using HFM1PARM statements; these changes can be made directly to the HFM1PARM member. When changes are made to the HFM1PARM member, no IPL is required and the changes become active the next time the user starts ZDT/IMS.

FMAUDIT

SAF_CTRL

```
SAF_CTRL={ YES  | NO }
```

SAF_CTRL

Specifies whether ZDT/IMS should use SAF to control all aspects of ZDT/IMS audit logging. See Preparing to customize ZDT/IMS on page 244 and Customizing the Z Data Tools audit facility for IMS component on page 297 for more information. SAF_CTRL is optional. The default is "YES", to use SAF.

Note: When SAF-rule controlled auditing is in effect (SAF_CTRL=YES), ZDT/IMS ignores both the values specified for the IMSAUDLG parameter on the HFM1POPD and HFM1POPI macro statements and the value specified for the SMFNO parameter on the HFM0POPI macro statement.

SMF_NO

```
SMF_NO={ 0  | number }
```

SMF_NO

Specifies the SMF record type for audit logging, where *number* is record type. You can specify a value from 128 to 255, or 0. The default is 0. If you specify an invalid SMF record number then auditing to SMF is inoperative.

FMOPTMOD

DSNAME

```
DSNAME=data_set_name
```

DSNAME

Specifies the name of the library from which ZDT/IMS is to load the ZDT/IMS options module - usually HFM1POPT. Note that the member keyword can be used to specify an alternative name to HFM1POPT, if required.

If the DSNAME keyword is specified, ZDT/IMS bypasses the usual library search when loading the ZDT/IMS options module and only searches the specified library for the module.

When the DSNAME keyword is omitted, ZDT/IMS searches load libraries allocated to the user's session in the normal order, for example, LIBDEF, ISPLLIB, STEPLIB, JOBLIB, link list.

One use of this option is in an environment with stringent security requirements, where it is necessary to ensure that all ZDT/IMS users access a particular ZDT/IMS options module, and that the use of this option module cannot be avoided.

The DSNAME keyword is optional.

MEMBER

```
MEMBER=member
```

MEMBER

Specifies the name of the member that contains the ZDT/IMS options module - usually HFM1POPT. The MEMBER keyword is optional.

Facilities for customizing the HFM1PARM definitions

You can use the HFM1PARM definitions to provide different options for ZDT/IMS users, based on the following environmental settings:

- The current z/OS® system ID
- The current version of ZDT/IMS
- The user's logonid

You can also split the HFM1PARM definitions across two or more members using the % (include) specification, and include comments using the * specification.

All of the facilities described in this section are optional. You do not need to use any of them.

Tags

A tag is an entry in the HFM1PARM member used to specify different options based on the current settings of certain environment variables. The following table shows the tags that can be specified.

Table 75. Z Data Tools Parmlib member tag types

Tag Type	Priority	Description	Usage
Z	1	z/OS® system	In a sysplex environment with multiple z/OS® images.
V	2	ZDT Version	In a environment with multiple concurrent versions of Z Data Tools.
U	3	User's logonid	Different options for specific users, typically by exception.

Rules for specifying tags

- Each tag should be entered on a separate line.
- Every initial tag must have a corresponding end tag. The initial tag is called the *"start tag"*.
- A start tag begins with a '<' and ends with a '>' character.
- The first character of the line containing a start tag must be '<', followed by the tag type.
- An end tag is specified as '<Ex>' or '</x>, where x is the tag type.
- The first character of the line containing an end tag must be '<', followed immediately by the rest of the end tag.
- One or more values can be specified for each tag. Separate values using a space or comma. Do not specify an '=' after the tag type.
- Values specified for each tag can include the * wildcard character. When specified, the '*' matches 0, 1, or many characters. Example: F* matches F, FRED, FUNNY, but not AFTER. The ** specification and other wildcard characters are not supported.
- Tags should be specified in order of priority to ensure correct processing.

Example of tag usage

Tags can be used to specify different HFM1PARM options based on certain environment variables (the z/OS® system ID, the ZDT/IMS version and the user's TSO logonid). The most usual usage of tags is to provide an exception to the normal processing, for some particular set of circumstances.

Example 1:

Suppose SAF-rule controlled auditing is turned off for all users in a single z/OS-image. The HFM1PARM member would be coded as:

```
FMAUDIT SAF_CTRL=NO
```

Now suppose that there is a requirement to implement SAF-rule controlled auditing, and that one particular logonid (TEST1) has been selected for testing purposes. Assuming all the relevant SAF rules have been written, SAF-rule controlled auditing can be turned on for logonid TEST1 using:

```
FMAUDIT SAF_CTRL=NO     (1)
<U TEST1>          (2)
FMAUDIT SAF_CTRL=YES      (3)
<EU>             (4)
```

Explanation:

When ZDT/IMS parses the above, the line (1) is processed for all users, and turns SAF-rule controlled auditing off. When line (2) is processed, the user's TSO logonid is compared with the value TEST1. For all users other than TEST1, the test fails. When a tag fails to match a condition, all lines from the start tag to the matching end tag (inclusive) are ignored, so the net result is that, in the above example, only line (1) is processed. This ensures that the default of no SAF-rule controlled auditing applies to all users other than TEST1.

For user TEST1 however, the tag comparison on line (2) matches, so line (3) is included. ZDT/IMS re-processes the FMAUDIT statement for line (3), this time with SAF_CTRL=YES, resulting in SAF-rule controlled auditing being turned on for user TEST1.

Note: This example demonstrates two important principles when processing the statements within an HFM1PARM member:

- Multiple statements for the same option are allowed.
- If multiple statements for the same option are encountered, the last statement processed is the one that determines the setting.

Example 2:

Suppose in a sysplex environment there are nine z/OS® images, with system IDs SYS1, SYS2, SYS3 … SYS9. Z Data Tools is available on all images, and SAF-rule controlled auditing is not required on SYS1, SYS2, … SYS7 inclusive, but is required on SYS8 and SYS9. Further, on z/OS® system SYS8, only TSO logonids that commence with DEV should be subject to SAF rule controlled audit; all other TSO logonids are exempt. On system SYS9 however, all TSO logonids should be subject to SAF-rule controlled audit, with the single exception of TSO logonid MAIN1.

This could be coded as follows:

```
FMAUDIT SAF_CTRL=NO     (1)
<Z SYS8>            (2)
<U DEV*>            (3)
FMAUDIT SAF_CTRL=YES      (4)
</U>              (5)
</Z>              (6)
<Z SYS9>            (7)
FMAUDIT SAF_CTRL=YES      (8)
<U MAIN1>         (9)
FMAUDIT SAF_CTRL=NO     (10)
</U>              (11)
</Z>              (12)
```

Explanation:

Line (1) sets the default, which is not to use SAF- rule controlled auditing.

For all users on z/OS® systems SYS1 … SYS7 inclusive, the Z tags for SYS8 on line (2) and SYS9 on line (7) will not match the current environment, resulting in lines 2-12 inclusive being ignored.

For users running on z/OS® system SYS8, line (2)-(6) inclusive are considered; lines (7)-(12) are ignored. Any TSO logonid that does not start with DEV (for example, PROD1) will not match the U tag (line 3), resulting in lines 3-5 inclusive being ignored. This leaves only line (1) to consider, which sets SAF-rule controlled auditing off. For a TSO logonid such as DEV76, the U tag on line (3) matches, so line (4) is included. ZDT/IMS processes the FMAUDIT statement on line (4) and sets SAF-rule controlled auditing on.

For users running on z/OS® system SYS9, lines (2)-(6) inclusive are ignored; lines (7)-(12) inclusive are considered. Line (8) changes the default (for all users on system SYS9) to use SAF- rule controlled auditing. For all users other than TSO logonid MAIN1, lines (9)-(11) are ignored, resulting in the new default (line 8) being used. This turns SAF-rule controlled auditing on. For TSO logonid MAIN1 only, the U tag on line (9) matches and the FMAUDIT statement on line (10) is included. This turns SAF-rule controlled auditing off for user MAIN1.

Included members

You can divide the HFM1PARM across multiple members using the include specification. The format for an included member is as follows:

- The first character of the line must be a '%' character.
- Specify the member name to be included after the % character.
- Specify a single member name only after the % character.

Multiple levels of include nesting are supported, circular (recursive) definitions will result in an error and ZDT/IMS will not initialize.

Empty included members are ignored.

Example 1:

Suppose in a sysplex environment there are 4 z/OS® images, with system IDs SYS1, SYS2, SYS3, and SYS4. Z Data Tools is available on all images, with different auditing requirements on each image.

Include members can be used to code this as follows:

Member HFM1PARM

```
<Z SYS1>
%HFMPSYS1
</Z>
<Z SYS2>
%HFMPSYS2
</Z>
<Z SYS3>
%HFMPSYS3
</Z>
<Z SYS4>
```

```
%HFMPSYS4
</Z>
```

Member HFMPSYS1

```
FMAUDIT SAF_CTRL=YES
```

Member HFMPSYS2

```
FMAUDIT SAF_CTRL=NO
```

Member HFMPSYS3

```
FMAUDIT SAF_CTRL=NO
```

Member HFMPSYS4

```
FMAUDIT SAF_CTRL=YES
```

This has the advantage of consolidating all the options for each z/OS® image in a specific member.

If you used Method 1 to add member HFM1PARM to the logical parmlib concatenation (see Defining the HFM1PARM member on page 301 for more information), be careful to avoid any conflicts between included member names and existing member names in the logical parmlib concatenation.

Comments

You can add comments to the HFM1PARM member as follows:

- The first character of a comment line must be a '*' character.
- The rest of the comment line is ignored.

Continuing specifications across multiple lines

You can specify the ZDT/IMS options across multiple lines as follows:

- The last character of each line to be continued must be a comma.
- Continue the text on the next line starting at any position (1-71).

Example:

```
FMAUDIT SAF_CTRL=YES
FMOPTMOD  DSNAME=ZDTOOLS.OTHER.OPTIONS,
    MEMBER=HFM1POPX
```

ZDT/Db2 options specified in HFM2PARM

This section describes the ZDT/Db2 options specified in parmlib member HFM2PARM. You can modify these ZDT/Db2 options to suit your requirements.

The HFM2PARM member provides a secure location for sensitive ZDT/Db2 parameters used to control, for example, audit logging. ZDT/Db2 uses system macros to locate this member in the current logical parmlib concatenation, making it difficult

to over-ride or bypass the contents, which can occur when using the HFM2POPT module. When used to specify SAF- rule controlled audit logging, particularly to SMF, a high level of integrity over the auditing process is possible. The HFM2PARM member can also be used to force the use of a particular version of the HFM2POPT module, by specifying a specific data set from which the module should be loaded. This can be used to prevent a knowledgeable user creating a private version of the HFM2POPT module and loading it from a data set allocated to their TSO session, thereby bypassing installation-specified parameters.

This section includes the description of the options that can be specified in the HFM2PARM member, followed by a description of the facilities that can be used to customize the HFM2PARM member. Customization is entirely optional, but might be necessary, for example, in a sysplex environment where not all z/OS® images have the same auditing requirements. It is also possible to specify different auditing options for different users using HFM2PARM statements; these changes can be made directly to the HFM2PARM member. When changes are made to the HFM2PARM member, no IPL is required and the changes become active the next time the user starts ZDT/Db2.

FMAUDIT

SAF_CTRL

```
SAF_CTRL={ YES | NO }
```

SAF_CTRL

Specifies whether ZDT/Db2 should use SAF to control all aspects of ZDT/Db2 audit logging. See Preparing to customize ZDT/Db2 on page 153 and SAF-controlled auditing for Z Data Tools Db2 component on page 192 for more information. SAF_CTRL is optional. The default is "YES", to use SAF.

Note: When SAF-rule controlled auditing is in effect (SAF_CTRL=YES), ZDT/Db2 ignores both the values specified for the AUDIT parameter on the HFM2SSDM macro statements and the value specified for the SMFNO parameter on the HFM0POPI macro statement.

SMF_NO

```
SMF_NO={ 0 | number }
```

SMF_NO

Specifies the SMF record type for audit logging, where *number* is record type. You can specify a value from 128 to 255, or 0. The default is 0. If you specify an invalid SMF record number then auditing to SMF is inoperative.

FMOPTMOD

DSNAME

```
DSNAME=data_set_name
```

DSNAME

Specifies the name of the library from which ZDT/Db2 is to load the ZDT/Db2 options module - usually HFM2POPT. Note that the member keyword can be used to specify an alternative name to HFM2POPT, if required.

If the DSNAME keyword is specified, ZDT/Db2 bypasses the usual library search when loading the ZDT/Db2 options module and only searches the specified library for the module.

When the DSNAME keyword is omitted, ZDT/Db2 searches load libraries allocated to the user's session in the normal order, for example, LIBDEF, ISPLLIB, STEPLIB, JOBLIB, link list.

One use of this option is in an environment with stringent security requirements, where it is necessary to ensure that all ZDT/Db2 users access a particular ZDT/Db2 options module, and that the use of this option module cannot be avoided.

The DSNAME keyword is optional.

MEMBER

```
MEMBER=member
```

MEMBER

Specifies the name of the member that contains the ZDT/Db2 options module - usually HFM2POPT. The MEMBER keyword is optional.

Facilities for customizing the HFM2PARM definitions

You can use the HFM2PARM definitions to provide different options for ZDT/Db2 users, based on the following environmental settings:

- The current z/OS® system ID
- The current version of ZDT/Db2
- The user's logonid

You can also split the HFM2PARM definitions across two or more members using the % (include) specification, and include comments using the * specification.

All of the facilities described in this section are optional. You do not need to use any of them.

Tags

A tag is an entry in the HFM2PARM member used to specify different options based on the current settings of certain environment variables. The following table shows the tags that can be specified.

Table 76. Z Data Tools Parmlib member tag types

Tag Type	Priority	Description	Usage
Z	1	z/OS® system	In a sysplex environment with multiple z/OS® images.
V	2	ZDT Version	In a environment with multiple concurrent versions of Z Data Tools.
U	3	User's logonid	Different options for specific users, typically by exception.

Rules for specifying tags

- Each tag should be entered on a separate line.
- Every initial tag must have a corresponding end tag. The initial tag is called the *"start tag"*.
- A start tag begins with a '<' and ends with a '>' character.
- The first character of the line containing a start tag must be '<', followed by the tag type.
- An end tag is specified as '<Ex>' or '</x>, where x is the tag type.
- The first character of the line containing an end tag must be '<', followed immediately by the rest of the end tag.
- One or more values can be specified for each tag. Separate values using a space or comma. Do not specify an '=' after the tag type.
- Values specified for each tag can include the * wildcard character. When specified, the '*' matches 0, 1, or many characters. Example: F* matches F, FRED, FUNNY, but not AFTER. The ** specification and other wildcard characters are not supported.
- Tags should be specified in order of priority to ensure correct processing.

Example of tag usage

Tags can be used to specify different HFM2PARM options based on certain environment variables (the z/OS® system ID, the ZDT/Db2 version and the user's TSO logonid). The most usual usage of tags is to provide an exception to the normal processing, for some particular set of circumstances.

Example 1:

Suppose SAF-rule controlled auditing is turned off for all users in a single z/OS-image. The HFM2PARM member would be coded as:

```
FMAUDIT SAF_CTRL=NO
```

Now suppose that there is a requirement to implement SAF-rule controlled auditing, and that one particular logonid (TEST1) has been selected for testing purposes. Assuming all the relevant SAF rules have been written, SAF-rule controlled auditing can be turned on for logonid TEST1 using:

```
FMAUDIT SAF_CTRL=NO     (1)
<U TEST1>          (2)
FMAUDIT SAF_CTRL=YES      (3)
<EU>             (4)
```

Explanation:

When ZDT/Db2 parses the above, the line (1) is processed for all users, and turns SAF-rule controlled auditing off. When line (2) is processed, the user's TSO logonid is compared with the value TEST1. For all users other than TEST1, the test fails. When a tag fails to match a condition, all lines from the start tag to the matching end tag (inclusive) are ignored, so the net result is that, in the above example, only line (1) is processed. This ensures that the default of no SAF-rule controlled auditing applies to all users other than TEST1.

For user TEST1 however, the tag comparison on line (2) matches, so line (3) is included. ZDT/Db2 re-processes the FMAUDIT statement for line (3), this time with SAF_CTRL=YES, resulting in SAF-rule controlled auditing being turned on for user TEST1.

Note: This example demonstrates two important principles when processing the statements within an HFM2PARM member:

- Multiple statements for the same option are allowed.
- If multiple statements for the same option are encountered, the last statement processed is the one that determines the setting.

Example 2:

Suppose in a sysplex environment there are nine z/OS® images, with system IDs SYS1, SYS2, SYS3 … SYS9. Z Data Tools is available on all images, and SAF-rule controlled auditing is not required on SYS1, SYS2, … SYS7 inclusive, but is required on SYS8 and SYS9. Further, on z/OS® system SYS8, only TSO logonids that commence with DEV should be subject to SAF rule controlled audit; all other TSO logonids are exempt. On system SYS9 however, all TSO logonids should be subject to SAF-rule controlled audit, with the single exception of TSO logonid MAIN1.

This could be coded as follows:

```
FMAUDIT SAF_CTRL=NO     (1)
<Z SYS8>            (2)
<U DEV*>            (3)
FMAUDIT SAF_CTRL=YES      (4)
</U>              (5)
</Z>              (6)
<Z SYS9>            (7)
FMAUDIT SAF_CTRL=YES      (8)
<U MAIN1>         (9)
FMAUDIT SAF_CTRL=NO     (10)
</U>              (11)
</Z>              (12)
```

Explanation:

Line (1) sets the default, which is not to use SAF- rule controlled auditing.

For all users on z/OS® systems SYS1 … SYS7 inclusive, the Z tags for SYS8 on line (2) and SYS9 on line (7) will not match the current environment, resulting in lines 2-12 inclusive being ignored.

For users running on z/OS® system SYS8, line (2)-(6) inclusive are considered; lines (7)-(12) are ignored. Any TSO logonid that does not start with DEV (for example, PROD1) will not match the U tag (line 3), resulting in lines 3-5 inclusive being

ignored. This leaves only line (1) to consider, which sets SAF-rule controlled auditing off. For a TSO logonid such as DEV76, the U tag on line (3) matches, so line (4) is included. ZDT/Db2 processes the FMAUDIT statement on line (4) and sets SAF-rule controlled auditing on.

For users running on z/OS® system SYS9, lines (2)-(6) inclusive are ignored; lines (7)-(12) inclusive are considered. Line (8) changes the default (for all users on system SYS9) to use SAF- rule controlled auditing. For all users other than TSO logonid MAIN1, lines (9)-(11) are ignored, resulting in the new default (line 8) being used. This turns SAF-rule controlled auditing on. For TSO logonid MAIN1 only, the U tag on line (9) matches and the FMAUDIT statement on line (10) is included. This turns SAF-rule controlled auditing off for user MAIN1.

Included members

You can divide the HFM2PARM across multiple members using the include specification. The format for an included member is as follows:

- The first character of the line must be a '%' character.
- Specify the member name to be included after the % character.
- Specify a single member name only after the % character.

Multiple levels of include nesting are supported, circular (recursive) definitions will result in an error and Z Data Tools will not initialize.

Empty included members are ignored.

Example 1:

Suppose in a sysplex environment there are 4 z/OS® images, with system IDs SYS1, SYS2, SYS3, and SYS4. Z Data Tools is available on all images, with different auditing requirements on each image.

Include members can be used to code this as follows:

Member HFM2PARM

```
<Z SYS1>
%HFMPSYS1
</Z>
<Z SYS2>
%HFMPSYS2
</Z>
<Z SYS3>
%HFMPSYS3
</Z>
<Z SYS4>
%HFMPSYS4
</Z>
```

Member HFMPSYS1

```
FMAUDIT SAF_CTRL=YES
```

Member HFMPSYS2

```
FMAUDIT SAF_CTRL=NO
```

Member HFMPSYS3

```
FMAUDIT SAF_CTRL=NO
```

Member HFMPSYS4

```
FMAUDIT SAF_CTRL=YES
```

This has the advantage of consolidating all the options for each z/OS® image in a specific member.

If you used Method 1 to add member HFM2PARM to the logical parmlib concatenation (see Defining the HFM2PARM member on page 193 for more information), be careful to avoid any conflicts between included member names and existing member names in the logical parmlib concatenation.

Comments

You can add comments to the HFM2PARM member as follows:

- The first character of a comment line must be a '*' character.
- The rest of the comment line is ignored.

Continuing specifications across multiple lines

You can specify the ZDT/Db2 options across multiple lines as follows:

- The last character of each line to be continued must be a comma.
- Continue the text on the next line starting at any position (1-71).

Example:

```
FMAUDIT SAF_CTRL=YES
FMOPTMOD  DSNAME=ZDTOOLS.OTHER.OPTIONS,
    MEMBER=HFM2POPX
```

ZDT/CICS options specified in HFM3PARM

This section describes the ZDT/CICS options specified in parmlib member HFM3PARM. You can modify these ZDT/CICS options to suit your requirements.

The HFM3PARM member provides a secure location for sensitive ZDT/CICS parameters used to control, for example, audit logging. ZDT/CICS uses system macros to locate this member in the current logical parmlib concatenation, making it difficult to over-ride or bypass the contents, which can occur when using the HFM3POPT module. When used to specify SAF- rule controlled audit logging, particularly to SMF, a high level of integrity over the auditing process is possible. The HFM3PARM member can also be used to force the use of a particular version of the HFM3POPT module, by specifying a specific data set from which the module should be loaded. This can be used to prevent a knowledgeable user creating a private version of the HFM3POPT module and loading it from a data set allocated to their TSO session, thereby bypassing installation-specified parameters.

This section includes the description of the options that can be specified in the HFM3PARM member, followed by a description of the facilities that can be used to customize the HFM3PARM member. Customization is entirely optional, but might be necessary, for example, in a sysplex environment where not all z/OS® images have the same auditing requirements. It is also possible to specify different auditing options for different users using HFM3PARM statements; these changes can be made directly to the HFM3PARM member. When changes are made to the HFM3PARM member, no IPL is required and the changes become active the next time the user starts ZDT/CICS.

FMAUDIT

SAF_CTRL

```
SAF_CTRL={ YES | NO }
```

SAF_CTRL

Specifies whether ZDT/CICS should use SAF to control all aspects of ZDT/CICS audit logging. See Preparing to access CICS resources from Z Data Tools on page 325 and Customizing the Z Data Tools audit facility for CICS component on page 352 for more information. SAF_CTRL is optional. The default is "YES", to use SAF.

Note: When SAF-rule controlled auditing is in effect (SAF_CTRL=YES), ZDT/CICS ignores the values specified for the AUDITLOG and SMFNO parameters on the HFM0POPI macro statement.

SMF_NO

```
SMF_NO={ 0 | number }
```

SMF_NO

Specifies the SMF record type for audit logging, where *number* is record type. You can specify a value from 128 to 255, or 0. The default is 0. If you specify an invalid SMF record number then auditing to SMF is inoperative.

FMOPTMOD

DSNAME

```
DSNAME=data_set_name
```

DSNAME

Specifies the name of the library from which ZDT/CICS is to load the ZDT/CICS options module - usually HFM3POPT. Note that the member keyword can be used to specify an alternative name to HFM3POPT, if required.

If the DSNAME keyword is specified, ZDT/CICS bypasses the usual library search when loading the ZDT/CICS options module and only searches the specified library for the module.

When the DSNAME keyword is omitted, ZDT/CICS searches load libraries allocated to the user's session in the normal order, for example, LIBDEF, ISPLLIB, STEPLIB, JOBLIB, link list.

One use of this option is in an environment with stringent security requirements, where it is necessary to ensure that all ZDT/CICS users access a particular ZDT/CICS options module, and that the use of this option module cannot be avoided.

The DSNAME keyword is optional.

MEMBER

```
MEMBER=member
```

MEMBER

Specifies the name of the member that contains the ZDT/CICS options module - usually HFM3POPT. The MEMBER keyword is optional.

Facilities for customizing the HFM3PARM definitions

You can use the HFM3PARM definitions to provide different options for ZDT/CICS users, based on the following environmental settings:

- The current z/OS® system ID
- The current version of ZDT/CICS
- The user's logonid

You can also split the HFM3PARM definitions across two or more members using the % (include) specification, and include comments using the * specification.

All of the facilities described in this section are optional. You do not need to use any of them.

Tags

A tag is an entry in the HFM3PARM member used to specify different options based on the current settings of certain environment variables. The following table shows the tags that can be specified.

Table 77. Z Data Tools Parmlib member tag types

Tag Type	Priority	Description	Usage
Z	1	z/OS® system	In a sysplex environment with multiple z/OS® images.
V	2	ZDT Version	In a environment with multiple concurrent versions of Z Data Tools.
U	3	User's logonid	Different options for specific users, typically by exception.

Rules for specifying tags

- Each tag should be entered on a separate line.
- Every initial tag must have a corresponding end tag. The initial tag is called the *"start tag"*.
- A start tag begins with a '<' and ends with a '>' character.
- The first character of the line containing a start tag must be '<', followed by the tag type.
- An end tag is specified as '<Ex>' or '</x>, where x is the tag type.
- The first character of the line containing an end tag must be '<', followed immediately by the rest of the end tag.
- One or more values can be specified for each tag. Separate values using a space or comma. Do not specify an '=' after the tag type.
- Values specified for each tag can include the * wildcard character. When specified, the '*' matches 0, 1, or many characters. Example: F* matches F, FRED, FUNNY, but not AFTER. The ** specification and other wildcard characters are not supported.
- Tags should be specified in order of priority to ensure correct processing.

Example of tag usage

Tags can be used to specify different HFM3PARM options based on certain environment variables (the z/OS® system ID, the ZDT/CICS version and the user's TSO logonid). The most usual usage of tags is to provide an exception to the normal processing, for some particular set of circumstances.

Example 1:

Suppose SAF-rule controlled auditing is turned off for all users in a single z/OS-image. The HFM3PARM member would be coded as:

```
FMAUDIT SAF_CTRL=NO
```

Now suppose that there is a requirement to implement SAF-rule controlled auditing, and that one particular logonid (TEST1) has been selected for testing purposes. Assuming all the relevant SAF rules have been written, SAF-rule controlled auditing can be turned on for logonid TEST1 using:

```
FMAUDIT SAF_CTRL=NO     (1)
<U TEST1>          (2)
FMAUDIT SAF_CTRL=YES      (3)
<EU>             (4)
```

Explanation:

When ZDT/CICS parses the above, the line (1) is processed for all users, and turns SAF-rule controlled auditing off. When line (2) is processed, the user's TSO logonid is compared with the value TEST1. For all users other than TEST1, the test fails. When a tag fails to match a condition, all lines from the start tag to the matching end tag (inclusive) are ignored, so the net result is that, in the above example, only line (1) is processed. This ensures that the default of no SAF-rule controlled auditing applies to all users other than TEST1.

For user TEST1 however, the tag comparison on line (2) matches, so line (3) is included. ZDT/CICS re-processes the FMAUDIT statement for line (3), this time with SAF_CTRL=YES, resulting in SAF-rule controlled auditing being turned on for user TEST1.

Note: This example demonstrates two important principles when processing the statements within an HFM3PARM member:

- Multiple statements for the same option are allowed.
- If multiple statements for the same option are encountered, the last statement processed is the one that determines the setting.

Example 2:

Suppose in a sysplex environment there are nine z/OS® images, with system IDs SYS1, SYS2, SYS3 … SYS9. Z Data Tools is available on all images, and SAF-rule controlled auditing is not required on SYS1, SYS2, … SYS7 inclusive, but is required on SYS8 and SYS9. Further, on z/OS® system SYS8, only TSO logonids that commence with DEV should be subject to SAF rule controlled audit; all other TSO logonids are exempt. On system SYS9 however, all TSO logonids should be subject to SAF-rule controlled audit, with the single exception of TSO logonid MAIN1.

This could be coded as follows:

```
FMAUDIT SAF_CTRL=NO     (1)
<Z SYS8>            (2)
<U DEV*>            (3)
FMAUDIT SAF_CTRL=YES      (4)
</U>              (5)
</Z>              (6)
<Z SYS9>            (7)
FMAUDIT SAF_CTRL=YES      (8)
<U MAIN1>         (9)
FMAUDIT SAF_CTRL=NO     (10)
</U>              (11)
</Z>              (12)
```

Explanation:

Line (1) sets the default, which is not to use SAF- rule controlled auditing.

For all users on z/OS® systems SYS1 … SYS7 inclusive, the Z tags for SYS8 on line (2) and SYS9 on line (7) will not match the current environment, resulting in lines 2-12 inclusive being ignored.

For users running on z/OS® system SYS8, line (2)-(6) inclusive are considered; lines (7)-(12) are ignored. Any TSO logonid that does not start with DEV (for example, PROD1) will not match the U tag (line 3), resulting in lines 3-5 inclusive being ignored. This leaves only line (1) to consider, which sets SAF-rule controlled auditing off. For a TSO logonid such as DEV76, the U tag on line (3) matches, so line (4) is included. ZDT/CICS processes the FMAUDIT statement on line (4) and sets SAF-rule controlled auditing on.

For users running on z/OS® system SYS9, lines (2)-(6) inclusive are ignored; lines (7)-(12) inclusive are considered. Line (8) changes the default (for all users on system SYS9) to use SAF- rule controlled auditing. For all users other than TSO logonid MAIN1, lines (9)-(11) are ignored, resulting in the new default (line 8) being used. This turns SAF-rule controlled auditing on. For TSO logonid MAIN1 only, the U tag on line (9) matches and the FMAUDIT statement on line (10) is included. This turns SAF-rule controlled auditing off for user MAIN1.

Included members

You can divide the HFM3PARM across multiple members using the include specification. The format for an included member is as follows:

- The first character of the line must be a '%' character.
- Specify the member name to be included after the % character.
- Specify a single member name only after the % character.

Multiple levels of include nesting are supported, circular (recursive) definitions will result in an error and ZDT/CICS will not initialize.

Empty included members are ignored.

Example 1:

Suppose in a sysplex environment there are 4 z/OS® images, with system IDs SYS1, SYS2, SYS3, and SYS4. Z Data Tools is available on all images, with different auditing requirements on each image.

Include members can be used to code this as follows:

Member HFM3PARM

```
<Z SYS1>
%HFMPSYS1
</Z>
<Z SYS2>
%HFMPSYS2
</Z>
<Z SYS3>
%HFMPSYS3
</Z>
<Z SYS4>
%HFMPSYS4
</Z>
```

Member HFMPSYS1

```
FMAUDIT SAF_CTRL=YES
```

Member HFMPSYS2

```
FMAUDIT SAF_CTRL=NO
```

Member HFMPSYS3

```
FMAUDIT SAF_CTRL=NO
```

Member HFMPSYS4

```
FMAUDIT SAF_CTRL=YES
```

This has the advantage of consolidating all the options for each z/OS® image in a specific member.

If you used Method 1 to add member HFM3PARM to the logical parmlib concatenation (see Defining the HFM3PARM member on page 358 for more information), be careful to avoid any conflicts between included member names and existing member names in the logical parmlib concatenation.

Comments

You can add comments to the HFM3PARM member as follows:

- The first character of a comment line must be a '*' character.
- The rest of the comment line is ignored.

Continuing specifications across multiple lines

You can specify the ZDT/CICS options across multiple lines as follows:

- The last character of each line to be continued must be a comma.
- Continue the text on the next line starting at any position (1-71).

Example:

```
FMAUDIT SAF_CTRL=YES
FMOPTMOD  DSNAME=ZDTOOLS.OTHER.OPTIONS,
    MEMBER=HFM3POPX
```

Appendix E. The library management system exit

This section describes the code and functions of the sample COBOL exit, HFMCRAEX, and also provides information on supporting multiple LMSs, and considerations when writing the exit in High Level Assembler.

Coding the exit: the basics

The main program and first argument: the operation code

The arguments to the exit follow standard 390 level two coding conventions: a pointer is provided to each argument.

The sample COBOL exit takes advantage of the fact that COBOL expects each argument to be a pointer. COBOL internally sets the address of the associated 01 structure in the LINKAGE SECTION.

The first argument in every call to the exit is a pointer to an operation code:

```
01 RAM-OPCODE PIC S9(9) BINARY.
01 RAM-OPCODE-VAL REDEFINES RAM-OPCODE PIC X(4).
88 OPCODE-INIT-RAM           VALUE X"09010000".
88 OPCODE-TERM-RAM           VALUE X"09010001".
88 OPCODE-VALIDATE-REPOS     VALUE X"09010004".
88 OPCODE-GET-1ST-MEM-REC    VALUE X"09010014".
88 OPCODE-GET-NEXT-MEM-REC   VALUE X"09010015".
88 OPCODE-GET-1ST-MEM-INFO   VALUE X"09010010".
88 OPCODE-GET-NEXT-MEM-INFO  VALUE X"09010011".
88 OPCODE-GET-DISPLAY-INFO   VALUE X"09010008".
```

The calls defined as level 88s in the COBOL code are the services which HFMCRAEX must provide.

Here is the main program showing how the OpCode is received by the HFMCRAEX program:

```
PROCEDURE DIVISION USING RAM-OPCODE,
                         RAM-WORK-AREA-PTR,
                         RAM-PARM2,
                         RAM-PARM3,
                         RAM-PARM4,
                         RAM-PARM5,
                         RAM-PARM6.
MAIN SECTION.

STARTUP-RAM.
    IF TRACELVL >= TRACEBAS THEN PERFORM TRACE-STARTUP.
    MOVE 0 TO RAM-RC.

SELECT-RAM-FUNCTION.
    EVALUATE TRUE
       WHEN OPCODE-INIT-RAM
          PERFORM INIT-RAM
       WHEN OPCODE-TERM-RAM
          PERFORM TERM-RAM
       WHEN OPCODE-VALIDATE-REPOS
          PERFORM VALIDATE-REPOS
       WHEN OPCODE-GET-1ST-MEM-REC
```

```
          PERFORM GET-1ST-MEM-REC
       WHEN OPCODE-GET-NEXT-MEM-REC
          PERFORM GET-NEXT-MEM-REC
       WHEN OPCODE-GET-1ST-MEM-INFO
          PERFORM GET-1ST-MEM-INFO
       WHEN OPCODE-GET-NEXT-MEM-INFO
          PERFORM GET-NEXT-MEM-INFO
       WHEN OPCODE-GET-DISPLAY-INFO
          PERFORM GET-DISPLAY-INFO
       WHEN OTHER
          SET RAM-RC-UNKNOWN-OPCODE TO TRUE
    END-EVALUATE.

RETURN-FROM-RAM.
    MOVE RAM-RC TO RETURN-CODE.
    GOBACK.
```

The return code

The return code from the sample COBOL exit is first set into a WORKING STORAGE variable RAM-RC, and at the end of the exit is placed into the special register RETURN-CODE.

Here are the possible return codes from the exit:

```
WORKING-STORAGE SECTION.

77 RAM-RC PIC S9(9) COMP.
   88 RAM-RC-OK VALUE 0.
   88 RAM-RC-UNKNOWN-OPCODE          VALUE 999.
   88 RAM-RC-REP-NOT-FOUND           VALUE 116.
   88 RAM-RC-INTERNAL-ERROR          VALUE 100.
   88 RAM-RC-REP-NOT-SUPPORTED       VALUE 120.
   88 RAM-RC-REP-IN-USE              VALUE 124.
   88 RAM-RC-MEM-ID-SPEC-BAD         VALUE 156.
   88 RAM-RC-REP-UNAVAIL-UNKNOWN     VALUE 132.
   88 RAM-RC-EOF                     VALUE 180.
   88 RAM-RC-REP-ERROR               VALUE 136.
   88 RAM-RC-BAD-FILTER              VALUE 152.
```

The return codes have the following meanings:

RAM-RC-UNKNOWN-OPCODE

Indicates that the requested function is not supported by the exit.

RAM-RC-REP-NOT-FOUND

Indicates that the library was not found. If the library name was a data set name, this means the data set was not found.

RAM-RC-INTERNAL-ERROR

Indicates some internal error occurred in the exit itself.

RAM-RC-REP-NOT-SUPPORTED

The library is not supported by this exit. Z Data Tools should try to access the library using other access methods such as CA-Panvalet or PDS access.

RAM-RC-REP-IN-USE

The library is in exclusive use.

RAM-RC-MEM-ID-SPEC-BAD

The Member Identifier Specification (the member name) parameter was not useable.

RAM-RC-REP-UNAVAIL-UNKNOWN

The library is unavailable for unknown reasons.

RAM-RC-EOF

Normal End Of File was received during the process of extracting member records or getting member information or metadata (for a member selection list).

RAM-RC-REP-ERROR

Some error occurred in the library.

RAM-RC-BAD-FILTER

The "filter" argument to the GetMemberInfo calls, which is a member name with optional wildcard characters, was bad.

Initializing the exit

Initializing the exit is trivial for a High Level Language (HLL) like COBOL or PL/I. Language Environment® (LE) obtains storage for HLLs. In the case of the sample exit, before COBOL starts, LE obtains the required COBOL working storage. Whenever the COBOL program is called, LE ensures it has access to its working storage. LE also maintains the data in working storage across calls.

If the user exit is written in HLASM, however, initializing the exit is more complex. You can use the INIT-RAM and TERM-RAM calls to get and release storage needed by the exit. The exit can set the second argument in every call to the exit, the RAM-WORK-AREA-PTR, at any time. Z Data Tools preserves and passes this value with every call to the exit. This provides a way for the exit to gain addressability to storage on every call. The exit should release all storage when it receives a call to TERM-RAM.

The sample COBOL exit code does not use the RAM-WORK-AREA-PTR, because, as mentioned above, LE provides an alternative to this mechanism for HLLs.

The code in INIT-RAM in the sample exit simply sets the tracing level from the caller and sets up the callback logging function address.

The arguments are:

```
Parm0 = OpCode = InitializeRAM
Parm1 = RAM-WORK-AREA-PTR
```

```
Parm2 = (Reserved)
Parm3 = TRACELVL  (integer value from 0 to 3)
Parm4 = Trace callback function address
```

The sample exit has two statements, **PERFORM CLOSE-AND-FREE-DIRFILE** and **PERFORM CLOSE-AND-FREE-MEMFILE**, that you may need to change to free your own resources. This is done to ensure that even if a previous instance of the exit abended prior to freeing resources, the resources will be freed.

Tracing

You can trace your exit while it is running under Z Data Tools. To do this, the exit writes to a log file. To write to a log file, define a DD with the name CRALOG. It should be a sequential file with RECFM FB and LRECL 132.

HFMCRAEX uses a logging callback function so that different logging functions can be provided depending on the caller of HFMCRAEX. In the sample TEST program, the log output is simply DISPLAYed. When Z Data Tools calls HFMCRAEX, as described above, it writes the log information to the predefined DD.

The callback mechanism means that logging calls should be written in the following style:

```
IF TRACELVL >= TRACEBAS THEN
   MOVE SPACES TO LOG-TEXT
   MOVE TRACELVL TO INT-TO-CHAR
   STRING "Initializing RAM, TRACELVL = " DELIMITED BY SIZE
          INT-TO-CHAR                     DELIMITED BY SIZE
     INTO LOG-TEXT
  CALL LOGFUNC USING LOG-HOST, LOG-MODULE, LOG-TEXT
END-IF.
```

The levels provided for tracing are:

TRACEOFF

No tracing

TRACEERR

Trace only error messages

TRACEBAS

Trace basic exit function

TRACEFUL

Trace everything

Terminating the exit

For an HLL, there is little work to do. The code should ensure that all data sets are closed, all DD names are freed, and in general that all resources are released.

For an exit written in HLASM, free any additional storage that has been allocated.

The sample exit has two statements, **PERFORM CLOSE-AND-FREE-DIRFILE** and **PERFORM CLOSE-AND-FREE-MEMFILE**, that you may need to change to free your own resources.

Note: See also the discussion about initializing the exit, in Initializing the exit on page 541.

Coding the exit: required services

The services that are required from the exit are:

1. Validate the library.
2. Extract member contents.
3. Get Member Information (also called metadata).
4. Get Display Information (ISPF header data).

Common argument and the name of the library

The argument RAM-ARGUMENTS is passed to all of the functions providing required services. It contains the name of the library as well as additional information.

Here are the data declarations and code in the sample exit to access the library name in RAM-ARGUMENTS:

```
 01 RAM-ARGUMENTS.
    05 SOFTWARE-VERSION PIC S9(4) BINARY.
    05 REP-EXEC-SPEC-PTR          POINTER.
    05 REP-ID-SPEC-PTR            POINTER.
    05 REP-PARMS-SPEC-PTR         POINTER.

 01 REP-ID-SPEC.
    05 SOFTWARE-VERSION PIC S9(4) BINARY.
    05 RESOURCE-ID-TYPE PIC S9(4) BINARY.
    05 REP-NAME-STRING-PTR        POINTER.

 01 REP-NAME-STRING.
    05 REP-NAME-LEN PIC  9(4) BINARY.
    05 REP-NAME-GROUP.
       10 REP-NAME  PIC  X
                    OCCURS 1 TO 50 TIMES
                    DEPENDING ON REP-NAME-LEN.

GET-RAM-PARM2-RAM-ARGS.
     SET ADDRESS OF RAM-ARGUMENTS TO RAM-PARM2.
     SET ADDRESS OF REP-ID-SPEC TO REP-ID-SPEC-PTR.
     SET ADDRESS OF REP-NAME-STRING TO REP-NAME-STRING-PTR
     IF TRACELVL >= TRACEBAS THEN
       MOVE SPACES TO LOG-TEXT
       STRING "Repository name = " DELIMITED BY SIZE
              REP-NAME-GROUP DELIMITED BY SIZE
              INTO LOG-TEXT
       CALL LOGFUNC USING LOG-HOST, LOG-MODULE, LOG-TEXT
     END-IF.
```

Validate the library

The goal of the function "Validate library" is to return a return code indicating whether the library can be supported by the exit.

The key return codes are:

0

Library supported by this exit.

120

Library not supported. Z Data Tools should try other access methods to try to read the data in the library.

(other)

Some other error occurred.

The sample COBOL exit validates a PDS by checking to see if it contains a member named $$HFM$$. If $$HFM$$ exists, the exit returns rc=0. If not, it returns rc=120.

There is a performance enhancement included in the sample code: if the library has already been validated, the validation is not repeated. This can only safely be done if the previous return code was 0 or 120. If it was some other value, the validation must be repeated. For example, if the previous validation found that the data set was in use, the validation must be repeated.

To support a customer LMS, you can replace the USER-REPOS-VALIDATE paragraph and leave the remaining code unchanged.

Get member records

In order to extract all records in a member, Z Data Tools first makes a GetFirstMemberRecord call, followed by multiple GetNextMemberRecord calls. If there is a record, then the rc=0. At any time, the process can be normally terminated with an End Of File (RAM-RC-EOF) return code.

The GetFirstMemberRecord call requires a member name as a parameter. It is passed as follows:

```
01 MEMBER-ID-SPEC.
    05 SOFTWARE-VERSION PIC 9(4) BINARY.
    05 MEM-ID-SPEC-COLL-PTR POINTER.
01 MEM-ID-SPEC-COLL.
    05 SOFTWARE-VERSION PIC 9(4) BINARY.
    05 MEM-ID-COMPONENT-COUNT PIC 9(4) BINARY.
    05 MEM-ID-COMPONENT-PTR POINTER
           OCCURS 1 TO 10 TIMES
           DEPENDING ON MEM-ID-COMPONENT-COUNT.
01 MEM-ID-COMPONENT.
    05 MEM-ID-COMP-TYPE PIC 9(4) BINARY.
    05 MEM-ID-COMP-NAME-STRING.
       10 MEM-ID-COMP-NAME-LEN PIC 9(4) BINARY.
       10 MEM-ID-COMP-NAME-GROUP.
          15 MEM-ID-COMP-NAME PIC X
                OCCURS 1 TO 10 TIMES
                DEPENDING ON MEM-ID-COMP-NAME-LEN.
01 MEMBER-NAME-STRING.
    05 MEMBER-NAME-LEN PIC 9(4) BINARY.
```

```
    05 MEMBER-NAME-GROUP.
       10 MEMBER-NAME PIC X
                      OCCURS 1 TO 10 TIMES
                      DEPENDING ON MEMBER-NAME-LEN.
GET-RAM-PARM3-MEM-ID.
     SET ADDRESS OF MEMBER-ID-SPEC TO RAM-PARM3
     SET ADDRESS OF MEM-ID-SPEC-COLL
                 TO MEM-ID-SPEC-COLL-PTR
     IF MEM-ID-COMPONENT-COUNT NOT = 1 THEN
        SET RAM-RC-MEM-ID-SPEC-BAD TO TRUE
     ELSE
        SET ADDRESS OF MEM-ID-COMPONENT
            TO MEM-ID-COMPONENT-PTR(1)
        IF MEM-ID-COMP-TYPE NOT = 0 THEN
           SET RAM-RC-MEM-ID-SPEC-BAD TO TRUE
        END-IF
     END-IF
     IF RAM-RC-OK THEN
        SET ADDRESS OF MEMBER-NAME-STRING
           TO ADDRESS OF MEM-ID-COMP-NAME-STRING
        IF TRACELVL >= TRACEBAS THEN
          MOVE SPACES TO LOG-TEXT
          STRING "Member = " DELIMITED BY SIZE
                 MEMBER-NAME-GROUP DELIMITED BY SIZE
                 INTO LOG-TEXT
          CALL LOGFUNC USING LOG-HOST, LOG-MODULE, LOG-TEXT
        END-IF
     END-IF.
```

If you are writing a user exit in COBOL, it is unlikely you will need to change this code which gets the member name argument. This code describes the data structures and is provided for users who want to rewrite the exit into PL/I or HLASM.

Here is the code for getting the first member record:

```
GET-1ST-MEM-REC.
     PERFORM GET-RAM-PARM2-RAM-ARGS
     PERFORM VAL-REPOS-VS-RAM-ARGS
     IF RAM-RC-OK THEN
        PERFORM GET-RAM-PARM3-MEM-ID
        PERFORM USER-GET-1ST-MEM-REC
     END-IF.
```

It uses the VAL-REPOS-VS-RAM-ARGS routine to determine if the library has already been validated. If so, it does not need to be redone. Then it gets the member name, and calls the USER-GET-1ST-MEM-REC paragraph. This is where you should insert your calls to the user library access methods.

The member record results are returned in a data area which looks like this:

```
01 REC-DATA-RETURN-AREA.
    05 REC-DATA-RETURN-ATTR1 PIC 9(4) BINARY VALUE 1.
    05 REC-DATA-RETURN-LEN1  PIC 9(4) BINARY VALUE 80.
    05 REC-DATA-RETURN-STR1  PIC X(80).
    05 REC-DATA-RETURN-END   PIC X(2)        VALUE X"FFFF".
```

The member record text is placed in REC-DATA-RETURN-STR1 and a return code of 0 returned.

If the member is empty, a RAM-RC-EOF (180) return code is returned.

Note: If you are providing support for an LMS which provides the capability to place proprietary INCLUDE statements in the member, the proprietary INCLUDEs should be **fully expanded**.

When getting the second and subsequent records in a member, the member name parameter is not passed to the exit.

Get member information (metadata)

The two calls associated with getting member metadata are GetFirstMemberInfo and GetNextMemberInfo. As in the case of getting member records, the normal return codes are 0 when member metadata is being returned, and 180 at end of file.

The argument required is not exactly the same as a member name. Instead, a filter is expected. The filter may be a member name, or it may contain characters and wildcard characters.

The wildcard characters supported by Z Data Tools are ***** matching zero or more characters, and **%** matching exactly one character.

Here is how a filter argument is passed:

```
01 SIMPLE-FILTER-SPEC.
    05 SOFTWARE-VERSION PIC 9(4) BINARY.
    05 SIMPLE-FILTER-FIELD-ID PIC 9(4) BINARY.
       88 SIMPLE-FILTER-IS-MEMBER-NAME VALUE 210.
    05 SIMPLE-FILTER-LENGTH PIC 9(4) BINARY.
    05 SIMPLE-FILTER-MASK PIC X(64). GET-RAM-PARM3-FILTER.
     IF RAM-PARM3 = NULL THEN
        MOVE 1 TO PATTERN-LENGTH
        MOVE "*" TO PATTERN-CHARS
     ELSE
        SET ADDRESS OF SIMPLE-FILTER-SPEC TO RAM-PARM3
        IF SIMPLE-FILTER-LENGTH < 0
              OR SIMPLE-FILTER-LENGTH > 10 THEN
           SET RAM-RC-BAD-FILTER TO TRUE
        ELSE
           MOVE SIMPLE-FILTER-LENGTH TO PATTERN-LENGTH
           MOVE SIMPLE-FILTER-MASK(1:PATTERN-LENGTH)
             TO PATTERN-CHARS
        END-IF
     END-IF.
```

A COBOL nested program called MATCH is provided in the sample COBOL exit to provide function to match a filter (pattern) to a member name.

Note: Z Data Tools strips any leading blanks prior to passing the filter (pattern), so you do not need to remove leading blanks from the filter yourself.

Note: Blanks are treated like any other character in the MATCH program, and if found, must be matched. Therefore, if the LMS returns leading blanks in member names you should strip them. If the member name includes trailing blanks, eliminate them from comparison by setting the length of the member name appropriately.

The metadata returned to Z Data Tools has the following structure. Note that the sample COBOL exit has given some structure to the RHS, which you may want to change:

```
01 DIR-DATA-RETURN-AREA.
    05 DIR-DATA-RETURN-ATTR1 PIC 9(4) COMP-5 VALUE 20000.
    05 DIR-DATA-RETURN-LEN1  PIC 9(4) BINARY VALUE 10.
    05 DIR-DATA-RETURN-LHS   PIC X(10) VALUE " ".
    05 DIR-DATA-RETURN-ATTR2 PIC 9(4) COMP-5 VALUE 20001.
    05 DIR-DATA-RETURN-LEN2  PIC 9(4) BINARY VALUE 51.
    05 DIR-DATA-RETURN-RHS.       10 RHS-USERID PIC X(7).
       10 FILLER PIC XX VALUE " ".
       10 RHS-MOD-DATE PIC X(10).
       10 FILLER PIC X VALUE " ".
       10 RHS-MOD-TIME PIC X(8).
       10 FILLER PIC XX VALUE " ".
       10 RHS-NLINES PIC ZZZZ9 DISPLAY.
       10 FILLER PIC XX VALUE " ".
       10 RHS-CRE8-DATE PIC X(10).
       10 FILLER PIC X(4).
    05 DIR-DATA-RETURN-END   PIC X(2)        VALUE X"FFFF".
```

The member name should go into DISP-INFO-RETURN-LHS, padded with trailing blanks. The attribute information should go into DISP-INFO-RETURN-RHS. There is space for 8 to 10 characters for the member name (LHS) and the total line is limited to 80 characters. The RHS can contain the remaining characters. The size of the prompt field is variable. See Get display information on page 547 for more information on the layout.

Get display information

Get Display returns the column headings for the member selection list panel when no member name or member name pattern is specified. The panel looks like this:

```
(Sel)  (LHS)    (Prompt) (RHS)

 Sel   Name      Prompt   Date Created    …
 _   MEMBER01   *Browsed 08/02/2003 14:53 …
 _   MEMBER02            12/21/2002 08:22 …
```

The names in parentheses at the top of the illustration are used in this example to describe the columns. There is a selection prefix area called **Sel**. There is a member name field called LHS (meaning the left hand side of the prompt). There is a prompt field, sometimes blank, and sometimes filled with information from Z Data Tools about past actions. The RHS contains any attribute information desired.

The GetDisplayInfo call returns the following static (unchanging) data structure with the heading:

```
01 DISP-INFO-RETURN-AREA.
    05 DISP-INFO-RETURN-ATTR1 PIC 9(4) COMP-5 VALUE 20000.
    05 DISP-INFO-RETURN-L11   PIC 9(4) BINARY VALUE 10.
    05 DISP-INFO-RETURN-L12   PIC 9(4) BINARY VALUE 10.
    05 DISP-INFO-RETURN-LHS   PIC X(8) VALUE " Name  ".
    05 DISP-INFO-RETURN-ATTR2 PIC 9(4) COMP-5 VALUE 20001.
    05 DISP-INFO-RETURN-L21   PIC 9(4) BINARY VALUE 51.
    05 DISP-INFO-RETURN-L22   PIC 9(4) BINARY VALUE 51.
    05 DISP-INFO-RETURN-RHS   PIC X(51)
        VALUE "  Id           Changed         Recs   Created".
    05 DISP-INFO-RETURN-ATTR3 PIC 9(4) COMP-5 VALUE 20002.
    05 DISP-INFO-RETURN-L31   PIC 9(4) BINARY VALUE 8.
    05 DISP-INFO-RETURN-L32   PIC 9(4) BINARY VALUE 8.
    05 DISP-INFO-RETURN-PROMP PIC X(8) VALUE " Prompt".
    05 DISP-INFO-RETURN-ATTR4 PIC 9(4) COMP-5 VALUE 20003.
    05 DISP-INFO-RETURN-L41   PIC 9(4) BINARY VALUE 4.
    05 DISP-INFO-RETURN-L42   PIC 9(4) BINARY VALUE 4.
    05 DISP-INFO-RETURN-PROMP PIC X(4) VALUE "Sel ".
    05 DISP-INFO-RETURN-END   PIC X(2)        VALUE X"FFFF".
```

Supporting multiple library management systems

Considerations when supporting multiple library management systems:

- Z Data Tools only calls one exit named HFMCRAEX.
- It is suggested that you write a separate exit for each LMS you need to support. Then, write a single HFMCRAEX that calls these LMS exits. Use the "Validate library" function (see Validate the library on page 544) to determine which LMS exit to use for a given library. This has many advantages, but the main one is that it is easy to add or remove supported LMSs.

Writing the exit in HLASM

The following considerations apply to writing the exit in HLASM:

- There is currently no sample exit written in HLASM. You will have to write this yourself.
- Use the HLASM register save, call and return macros provided by Language Environment®. The caller code in Z Data Tools is LE-enabled and assumes the callee is LE enabled.
- Use argument 1, referred to in the sample COBOL program as RAM-WORK-AREA-PTR, in all the calls to the exit so that Z Data Tools remembers the address of your root block of storage. You can set this to any value and Z Data Tools will remember it and provide it back to your exit on every call.
- Get memory during the INIT-RAM call and free all memory during the TERM-RAM call.
- Use the TEST sample COBOL program to call your HLASM exit routine and test it outside of Z Data Tools.

Appendix F. Z Data Tools audit records

This section describes the structure of, and information recorded in, Z Data Tools audit records.

An audit log record may occupy one or more physical records in the audit file. When an audit log record requires two or more physical records, the second and subsequent log records are continuation records.

Each Z Data Tools audit record consists of three parts:

- Record header
- Data item reference section
- Data section

Part 1 - record header

Table 78: Audit record: Header on page 549 shows the information contained in the record header. The record header is at the start of each physical record that makes up the audit log record.

Table 78. Audit record: Header

Offsets		Type	Len	Name (Dim)	Description
Dec	Hex				
0	(0)	BIT(8)	1	LOG_FLAGS	Log flags
		`1... ....`		LOG_CONT	Continuation indicator
		`.1.. ....`		LOG_IMS	Log record for ZDT/IMS
		`..1. ....`		LOG_DB2	Log record for ZDT/Db2
		`...1 ....`		LOG_CICS	Log record for ZDT/CICS
		`.... 111.`		*	
		`.... ...1`		LOG_V2IND	Log record Version 2 or later
1	(1)	UNSIGNED BI-NARY	1	LOG_VERSION	Version number
2	(2)	UNSIGNED BI-NARY	4	LOG_RECNUM	Record count within the current audit session
6	(6)	CHARACTER	5	LOG_ID	Log record constant 'FMIMS' (IMS™ eyecatcher) 'FMDB2' (Db2® eyecatcher)

Table 78. Audit record: Header

(continued)

Offsets		Type	Len	Name (Dim)	Description
Dec	Hex				
					'FMCIC' (CICS® eyecatcher) 'FMLOG' (base eyecatcher)
11	(B)	CHARACTER	3	*	
14	(E)	UNSIGNED PACKED DECIMAL	4	LOG_DATE	Date - yyyymmdd
18	(12)	UNSIGNED PACKED DECIMAL	4	LOG_TIME	Time - hhmmssth
22	(16)	CHARACTER	8	LOG_SYSID	MVS™ system ID (CVTSNAME)
30	(1E)	CHARACTER	8	LOG_USERID	Userid
38	(26)	CHARACTER	8	LOG_SSID	ZDT/Db2 System ID, IMS Subsystem name, or CICS® Applid
46	(2E)	CHARACTER	8	LOG_JOBNAME	Job Name
54	(36)	CHARACTER	8	LOG_JOBID	Job ID
62	(3E)	CHARACTER	1	LOG_SESSID	Session ID
63	(3F)	CHARACTER	8	LOG_FUNCOD	Z Data Tools function code
71	(47)	CHARACTER	8	LOG_DBNAME	IMS™ Database name
79	(4F)	CHARACTER	23	*	

Part 2 - data item reference section

The data item reference section is in the first physical record of the audit record, immediately after the audit record header. It consists of 2 parts: the data item reference header, followed by the data item reference array. Table 79: Audit record: Data item reference header on page 551 shows the information contained in the data item reference header.

Table 79. Audit record: Data item reference header

Offsets		Type	Len	Name (Dim)	Description
Dec	Hex				
102	(66)	UNSIGNED BINARY	2	LOG_ITEM_CNT	Number of data items in the data section of the audit record.
104	(68)	UNSIGNED BINARY	2	LOG_REC_COUNT	Number of physical records required to write the audit log record. A value greater than 1 implies there are continuation records.

The data item reference array is an array of entries. Each entry describes a single data item. The LOG_ITEM_CNT value in the data item reference header (see Table 79: Audit record: Data item reference header on page 551) specifies the number of elements in the array.

Table 80: Audit record: Data item reference section on page 551 shows the information contained in a data item reference array element.

Table 80. Audit record: Data item reference section

Offsets		Type	Len	Name (Dim)	Description
Dec	Hex				
0	(0)	STRUCTURE	8	LOG_DATA_ITEM(*)	
0	(0)	UNSIGNED BINARY	2	LOG_ITEM_CODE	Item data code. See Table 83: Data items common to all audit records on page 553.
2	(2)	CHARACTER	1	LOG_DATATYPE	Data type code. See Table 84: Data item types common to all audit records on page 553.
3	(3)	CHARACTER	1	*	Reserved/filler
4	(4)	SIGNED BINARY	4	LOG_ITEM_DATLEN	Fullword length

Part 3 - data section

The data section contains all of the data for the various data items defined in the data item reference section of the record. Each data item is stored in sequence and occupies exactly the length given in the data item reference section. There are no space bytes between data items.

The data items depend on the Z Data Tools component and the record type. In general terms, each record type is associated with some data items that are always written, followed by some data items that are specific to the record type and the component that is in use.

Table 81: Audit record: Data section on page 552 shows the information contained in the data section:

Table 81. Audit record: Data section

Offsets		Type	Len	Name (Dim)	Description
Dec	Hex				
0	(0)	STRUCTURE	*	LOG_DATAAREA	
0	(0)	UNSIGNED BINARY	2	LOG_DATALEN	Length of LOG_DATA
2	(2)	STRUCTURE	*	LOG_DATA	logging data

Note: For continuation records, the data for the incomplete data item continues from the previous record, immediately after the continuation record header, as shown in Table 82: Audit record: continued record on page 552.

Table 82. Audit record: continued record

Offsets		Type	Len	Name (Dim)	Description
Dec	Hex				
0	(0)	CHARACTER	102	LOG_HEADER_FIXED	The Audit record header (see Table 78: Audit record: Header on page 549)
102	(66)	UNSIGNED BINARY	2	LOG_CONT_LEN	Length of continued data
104	(68)	CHARACTER	*	LOG_CONT_DATA	The data for the incomplete data item (and possibly other data items)

Table 83. Data items common to all audit records

Data item	Usage	Usual length
4	Audit record type	1
3	Z Data Tools component that wrote the record. A fixed field with the possible values: **0** Z Data Tools Base component **1** ZDT/IMS component **2** ZDT/Db2 component **3** ZDT/CICS component	1
91[85]	Z Data Tools function code	8
92	Z Data Tools internal function code. This will usually be the same as the Z Data Tools function code.	8

Table 84. Data item types common to all audit records

Data item type code	Data item type
B	Binary
C	Character (default)
U	Unformatted

Z Data Tools (Base) audit records

This section describes the Z Data Tools Base component audit records.

Z Data Tools (Base) audit data items

The following table shows the data items that appear in Z Data Tools Base component audit log records.

85. This data item is not included in any ZDT/IMS audit records except for the Template record (Audit record type T).

Table 85. Data items that appear in Z Data Tools Base component audit log records

Data item identifier	Description	Usual length
15	Data set name	54
21	Template data set member name	54
22	Template type	1
31	Type of record	1
32	Record	Variable
41	Copy utility used	8
42	Copied count	4
43	Replaced count	4
44	Not copied count	4
45	IO error count	4
51	Output data set name	54
52	KSDS key location	4
53	KSDS key length	4
101	Template fd99 segment	36
102	Template	Variable

Z Data Tools (Base) audit record types

The following table shows the Z Data Tools Base component record types - each record type has a single character code.

Table 86. Z Data Tools Base component record types

Audit Record Type Code	Usage
A	New record.
D	Deleted record.
E	Record counts.
O	Original record. An 'O' record is written for an update (change) operation, and is followed by an 'R' record.
R	Replacement record. An 'R' record is written for an update (change) operation, and is preceded by an 'O' record.
S	Start of session record. Each 'S' record should have a corresponding 'Z' record.

Table 86. Z Data Tools Base component record types (continued)

Audit Record Type Code	Usage
T	Template record. The template is written in internal format.
V	Read record.
W	Written record. The record was written using the Z Data Tools REXX WRITE function.
Z	End of session record.

Data items specific to Z Data Tools (Base) audit records

The following table shows the data items specific to Z Data Tools Base component audit records.

Table 87. Data items specific to Z Data Tools Base component audit records

Audit Record Type	Data items
A	31, 32
D	21, 32
E	41, 42, 43, 44, 45
O	31, 32
R	31, 32
S	15, 51, 52, 53
T	21, 22, 101, 102
V	31, 32
W	31, 51, 32
Z	See Table 83: Data items common to all audit records on page 553

ZDT/Db2 audit records

This section describes the ZDT/Db2 audit records.

ZDT/Db2 audit data items

Table 88: Data items that appear in ZDT/Db2 audit log records on page 555 shows the data items that appear in ZDT/Db2 audit log records.

Table 88. Data items that appear in ZDT/Db2 audit log records

Data item identifier	Description	Usual length
14	Input/Output indicator	1

Table 88. Data items that appear in ZDT/Db2 audit log records (continued)

Data item identifier	Description	Usual length
21	Template data set member name	10
22	Template type (input or output)	1
101	Template fd99 segment	36
102	Template	Variable
2003	Current server	16
2004	Current SQLID	8
2005	Connection type	5
2011	Object location	16
2012	Object database	8
2013	Object table space	8
2014	Object owner	8
2015	Object name	128
2031	Type of Db2® row	1
2032	Db2® row (data)	Variable
2041	Row count (in)	4
2042	Row count editor	4
2043	Row count (out)	4
2051	SQL Stmt type	3
2052	SQL Reason code	4
2053	Affected row count	4
2071	Db2® Object type (for DDL statements)	3
2072	Db2® Privilege type (for Grant/Revoke statements)	3
2501	SQL statement	Varies
2502	Db2® command	Varies

ZDT/Db2 audit record types

Table 89: ZDT/Db2 audit record types on page 557 shows the ZDT/Db2 audit record types. Each record type has a single character code.

Table 89. ZDT/Db2 audit record types

Audit Record Type Code	Usage
A	New record or row.
D	Deleted record or row.
E	Enumeration record: reports the number of records or rows read or copied.
I	Identification record: contains information about the data set or Db2® object that is being processed.
J	Identification record: specific to Db2®. Contains the SQL SELECT statement used to access Db2® objects.
O	Original record or row. An 'O' record is typically written for an update (change) operation, and is followed by an 'R' record.
Q	Contains the text for an SQL statement. This is specific to ZDT/Db2.
R	Replacement record or row. An 'R' record is typically written for an update (change) operation, and is preceded by an 'O' record.
S	Start of session record. An 'S' record is written at the start of auditing for some ZDT/Db2 function. Each 'S' record should have a corresponding 'Z' record.
T	Template record. Contains a Z Data Tools/Db2 template used to process a data set or Db2® object. The template is written in internal format.
Z	End of session record. A 'Z' record is written at the end of auditing for some ZDT/Db2 function. Each 'Z' record should have a corresponding 'S' record.
2	Contains the text of a Db2® command. This is specific to ZDT/Db2.

Data items specific to ZDT/Db2 audit records

Table 90. Data items specific to ZDT/Db2 audit records

Audit Record Type	Data items
A	2031, 2032
D	2031, 2032
E	2041, 2042, 2043
I	14, 2011, 2012, 2013, 2014, 2015
J	2501
O	2031, 2032
Q	2051, 2052, 2053, 2071, 2072, 2501

Table 90. Data items specific to ZDT/Db2 audit records (continued)

Audit Record Type	Data items
R	2031, 2032
S	2003, 2004, 2005
T	21, 22, 101, 102
Z	See Table 83: Data items common to all audit records on page 553
2	2502

ZDT/IMS audit records

This section describes the ZDT/IMS audit records.

ZDT/IMS audit data items

The following table shows the data items that appear in ZDT/IMS audit log records.

Table 91. Data items that appear in ZDT/IMS audit log records

Data item identifier	Description	Type	Length
21	Template/view/criteria set data set and member name	CHARACTER	<=54
22	Template type: **C** Criteria set **T** Template **V** View	CHARACTER	1
101	Template FD99 segment		36
102	Template data		Variable
1001	Database name	CHARACTER	8
1002	Segment name	CHARACTER	8
1003	Segment level	UNSIGNED BINARY	1
1004	Segment description (as specified in template)	CHARACTER	0 or 15
1005	Identifier of template layout used to format data	SIGNED BINARY	4

Table 91. Data items that appear in ZDT/IMS audit log records (continued)

Data item identifier	Description	Type	Length

1006	Segment update flags:
	Segment occurrence is a child of the deleted segment
	Dependents of deleted segment have not been logged
	Not used
		BIT(8)
		`1... ....`
		`.1.. ....`
		`..11 1111`
			1

1007	Concatenated key value		Variable
1008	Segment data		Variable
1009	Checkpoint ID	CHARACTER	8

1010	Session flags:
	Dynamic PSB?
	BMP region?
	Using view/criteria set?
	Using secondary index?
	Using key values?
	Initial load?
	Load replaces existing segments?
	REXX procedure in-stream?
	Last 8 bits not used.
		BIT(16)
		`1... ....`
		`.1.. ....`
		`..1. ....`
		`...1 ....`
		`.... 1...`
		`.... .1..`
		`.... ..1.`
		`.... ...1`
			2

1011	Primary database name	CHARACTER	8
1012	PSB data set name	CHARACTER	0 or 44
1013	PSB name	CHARACTER	0 or 8
1014	Position of selected DB PCB in PSB's PCB list	SIGNED BINARY	2
1015	Name of secondary index	CHARACTER	0 or 8
1016	Application group name (AGN)	CHARACTER	0 or 8
1017	REXX procedure data set name	CHARACTER	0 or 44
1018	REXX procedure member name	CHARACTER	0 or 8
1019	Key values data set and member name	CHARACTER	0 or 54

ZDT/IMS audit record types

The following table shows the ZDT/IMS audit record types - each record type has a single character code.

Table 92. ZDT/IMS audit record types

Audit Record Type Code	Usage
A	Inserted segment record.
B	Rollback record.
C	Checkpoint record.
D	Deleted segment record. ZDT/IMS generates one record for the segment occurrence that is deleted and one record for each of its dependent segments.
F	Delete failure record. When issued it indicates that a Delete failed, so the set of D records that were generated for this Delete should be ignored.
O	Segment replace record - before image.
R	Segment replace record - after image.
S	Start of session record.
T	Template record.
Z	End of session record.

Data items specific to ZDT/IMS audit records

The following table shows the data items specific to ZDT/IMS audit records.

Note: Data items that are common to all ZDT/IMS audit records (4, 3, 92) are not listed.

Table 93. Data items specific to ZDT/IMS audit records

Audit Record Type	Data items
A	1001, 1002, 1003, 1004, 1005, 1006, 1007, 1008
B	1009
C	1009
D	1001, 1002, 1003, 1004, 1005, 1006, 1007, 1008
F	No data items except the ones that are common to all ZDT/IMS record types.

Table 93. Data items specific to ZDT/IMS audit records

(continued)

Audit Record Type	Data items
O	1001, 1002, 1003, 1004, 1005, 1006, 1007, 1008
R	1001, 1002, 1003, 1004, 1005, 1006, 1007, 1008
S	1010, 1011, 1012, 1013, 1014, 1015, 1016, 1017, 1018, 1019
T	21, 22, 101, 102
Z	No data items except the ones that are common to all ZDT/IMS record types.

ZDT/CICS audit records

This section describes the ZDT/CICS audit records.

ZDT/CICS audit data items

The following table shows the data items that appear in ZDT/CICS audit log records.

Table 94. Data items that appear in ZDT/CICS audit log records

Data item identifier	Description	Usual length
21	Template data set member name	54
22	Template type	1
31	Type of record	1
32	Record	Variable
101	Template fd99 segment	36
102	Template	Variable
3001	Resource type. The meaning of the bits from high-order to low-order is: • File • Intrapartition transient data queue • Extraparition transient data queue • Temporary storage queue • 4 bits reserved	1
3002	Resource name	16

Table 94. Data items that appear in ZDT/CICS audit log records (continued)

Data item identifier	Description	Usual length
3003	CICS® applid of resource	8
3004	Resource pool name	8
3005	DDname	8
3006	Data set name	44

ZDT/CICS audit record types

The following table shows the ZDT/CICS audit record types. Each record type has a single character code.

Table 95. ZDT/CICS audit record types

Audit Record Type Code	Usage
A	New record.
D	Deleted record.
O	Original record. An 'O' record is written for an update (change) operation, and is followed by an 'R' record.
R	Replacement record. An 'R' record is written for an update (change) operation, and is preceded by an 'O' record.
S	Start of session record. Each 'S' record should have a corresponding 'Z' record.
T	Template record. The template is written in internal format.
V	Read record.
Z	End of session record.

Data items specific to ZDT/CICS audit records

The following table shows the data items specific to ZDT/CICS audit records.

Table 96. Data items specific to ZDT/CICS audit records

Audit Record Type	Data items
A	31, 32
D	21, 32
O	31, 32
R	31, 32
S	3001, 3002, 3003, 3004, 3005, 3006

Table 96. Data items specific to ZDT/CICS audit records (continued)

Audit Record Type	Data items
T	21, 22, 101, 102
V	31, 32
Z	See Table 83: Data items common to all audit records on page 553

Notices

HCL Z Data Tools Licensed Materials - Property of HCL Technologies Ltd. @ Copyright IBM Corp. 2000, 2016. © Copyright HCL Technologies Limited 2017, 2026.

This information was developed for products and services offered in the U.S.A.

HCL may not offer the products, services, or features discussed in this document in other countries. Consult your local HCL representative for information on the products and services currently available in your area. Any reference to an HCL product, program, or service is not intended to state or imply that only that HCL product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any HCL intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-HCL product, program, or service.

For license inquiries regarding double-byte (DBCS) information, contact the HCL Intellectual Property Department in your country or send inquiries, in writing, to:

HCL
330 Potrero Ave.
Sunnyvale, CA 94085
USA
Attention: Office of the General Counsel

HCL TECHNOLOGIES LTD. PROVIDES THIS PUBLICATION “AS IS” WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some jurisdictions do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. HCL may make improvements and/or changes in the product(s) and/or the program(s) described in this document at any time without notice.

Any references in this information to non-HCL websites are provided for convenience only and do not in any manner serve as an endorsement of those websites. The materials at those websites are not part of the materials for this HCL product and use of those websites is at your own risk.

HCL may use or distribute any of the information you provide in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who want to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

HCL
330 Potrero Ave.
Sunnyvale, CA 94085

USA
Attention: Office of the General Counsel

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this information and all licensed material available for it are provided by HCL under terms of the HCL Customer Agreement, HCL International Program License Agreement, or any equivalent agreement between us.

The performance data discussed herein is presented as derived under specific operating conditions. Actual results may vary.

Information concerning non-HCL products was obtained from the suppliers of those products, their published announcements or other publicly available sources. HCL has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-HCL products. Questions on the capabilities of non-HCL products should be addressed to the suppliers of those products.

Statements regarding future direction or intent of HCL Z Data Tools are subject to change or withdrawal without notice, and represent goals and objectives only.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrates programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to HCL, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. HCL, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. The sample programs are provided "AS IS", without warranty of any kind. HCL shall not be liable for any damages arising out of your use of the sample programs.

Each copy or any portion of these sample programs or any derivative work must include a copyright notice as follows:

© (your company name) (year).
Portions of this code are derived from IBM Corp. and/or
HCL Ltd. sample programs.
© Copyright IBM Corp. 2000, 2016. © Copyright HCL Ltd. 2017, 2026.

Programming interface information

This documentation describes intended Programming Interfaces that allow the customer to write programs to obtain the services of Z Data Tools.

Trademarks

HCL, the HCL logo, and hcl.com® are trademarks or registered trademarks of HCL Technologies Ltd., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM® or other companies.

Programming interface information

This Customization Guide documents information that is **not** intended to be used as Programming Interfaces of Z Data Tools.

Index

A

ABENDCC processing option 384
ACBLIB option 458
ACBMGMT option 458
ACBs
  managed through IMS 253
ACBSHR option 459
access
  denying 287
  granting 287
access levels
  for SAF rules 95, 196, 360
additional security considerations 381
address space timeout 382
AFP-authorization
  authorizing the COBOL compiler library 44, 45
AGN option 513
aliases for DBRMs 172
ALTCOB processing option 384
alternate libraries for ISPF 27
ALTLIB 155
ALTPLI processing option 385
APF-authorization
  authorizing the COBOL compiler library 53
  authorizing the PL/I compiler library 53
  diagnosing problems 53
  planning to run Z Data Tools APF-authorized 27
  provided for ZDT/Db2 156
  running Z Data Tools APF-authorized 53, 53
ASCII processing option 385
ASCII translation tables, changing 63
ATTACH option 426
AUDDATAC processing option 386
audit
  controlling with SAF rules 95, 196, 360
  data set configuration 89
audit data set
  configuration 191
audit facility
  customizing for IMS component 297
  customizing for Z Data Tools 94
  customizing for Z Data Tools Base function 88
  customizing for ZDT/CICS 360
  customizing for ZDT/Db2 189, 195
  ZDT/Db2 157
audit log records
  determining if they should be written 96, 197, 361
audit logging
  under CICS logging 353
  under ZDT/CICS 353
  using SMF 86
AUDIT option 427
audit records
  customizing Z Data Tools to write to SMF 86
  information held by 549
  structure 549
  Z Data Tools Base component 553
  ZDT/CICS 561
  ZDT/Db2 555
  ZDT/IMS 558
audit trail
  determining if to be produced 190
  recording to dataset (IMS) 355
  recording ZDT/IMS to data set 299
audit trail (type A) security exit 289
AUDITBROWSE option 428
AUDITHLQ processing option 386
auditing
  controlling using SAF 300, 357
  HFM0POPT-controlled 89
  HFM3POPT-controlled 355
  SAF-controlled using SYS1.PARMLIB 357
  SAF-controlled without SYS1.PARMLIB 359
AUDITLOG processing option 388
AUDMGMTC processing option 389
AUDPQTY processing option 389
AUDSQTY processing option 389
AUDSTORC processing option 389
AUDSUNIT processing option 390
AUDUNIT processing option 390
AUTH_ACCESS option 428
AUTO_COMMIT option 429
AUTOSAVE option 459
AUXDATAC processing option 390
AUXDSN processing option 390
AUXHLQ processing option 391
AUXMDSN processing option 392
AUXMGMTC processing option 392
AUXSTORC processing option 392

B

batch JCL
  enabling edit models 39
batch JCL skeleton
  changing for Z Data Tools Base function 57
  changing for ZDT/Db2 181
  changing for ZDT/IMS 279
  customizing for the COBOL compiler library 58
batch return codes
  customizing for Z Data Tools 60
BDY processing option 392
binding Db2 for OAM functions 52
binding Db2 for ZDT/Db2 172
  possible SQL errors 173
BLP, controlling access 73
BMP mode 246
BSDSHLQ option 460

C

CATALIAS option 460
CATOWNER option 448
CATOWNERCDRM option 446
CCSID processing option 393
CDRM
  current DECFLOAT rounding mode 446
  important information 447
CERTRUST keyword 383
changing the ASCII translation tables 63
checklists
  Z Data Tools customization 23
  ZDT/CICS customization 326
  ZDT/Db2 customization 153
  ZDT/IMS customization 244
CHGAFREQ option 460
CHKPINTVL option 461
CICS
  external interface support 400
  setting up access for 102, 333
  supported versions 39
COBDBCS processing option 393
COBDPC processing option 393
COBEXTND processing option 393
COBMAXRTN processing option 394
COBMCASE processing option 394
COBOL
  ABENDS306 from a batch job 53
  adding the COBOL compiler to the LINKLIST 45
  authorizing the COBOL compiler library 53
  customizing for using COBOL copybooks 42
  options that must not be fixed in IGYCDOPT 46
  supported versions 39
  suppressing COBOL compiler warning messages 44, 45
  using the HFMCOB DD statement 45
  using the HFMTERM DD statement 44, 45
COBOL compiler
  controlling use of 76
COBOL compiler library in the batch JCL skeleton 58
COBOL compiler library in the ZDT/IMS batch JCL skeleton 279
COMMADEC processing option 394
commands
  SHOWPLI 46
comments
  in the HFM0PARM member 520
  in the HFM1PARM member 526
  in the HFM2PARM member 532
  in the HFM3PARM member 538
COMPAT option 461
COMPLANG processing option 395
composite view or criteria set 256
compression exit 123
concatenating libraries to the LINKLIST
  ZDT/Db2 155
  ZDT/IMS 258
CONNECT option 449
controller parameters, IMS regions 248
CPYCPYN option 430
CREPLACEn processing option 395
criteria sets
  configuring usage rules 256, 256, 508, 509
  temporary, and templates 255
  usage rules 256
CSYSLIBnn processing option 396
customization
  verifying 315
customization checklists
  Z Data Tools Base function 23
  ZDT/CICS 326
  ZDT/Db2 153
  ZDT/IMS 244
customization of Japanese component
  verifying 150
customization of Z Data Tools Base function
  verifying 148
customization of ZDT/CICS
  verifying 373
customization of ZDT/Db2
  verifying 218
customizing for national languages
  checklist for ZDT/CICS 368
  checklist for ZDT/Db2 211
  checklist for ZDT/IMS 310

customizing to use LMS libraries
checklist for Z Data Tools Base function 120
CYLHD processing option 396

D

DASD, controlling fullpack access 70
data set information, reporting 397
data sharing groups, Db2 174, 175
Database Access Control facility, ZDT/IMS 282
database data set allocation exit 289
DATAHDR processing option 396
DB2
supported versions 39
Db2 Admin Launchpad 165
Db2 catalog, granting access 161
Db2 data sharing groups, specifying for ZDT/Db2 174, 175
Db2 objects read access
controlling auditing of 205
Db2 objects update access
controlling auditing of 204
Db2 subsystems, specifying for ZDT/Db2 174, 175
Db2, binding ZDT/Db2 172
DB2ALIB option 430
DB2CLIB option 430
DB2ELIB option 430
DB2LLIB option 431
DB2MLIB option 431
DB2PLIB option 432
DB2PROC option 432
DB2RLIB option 432
DB2SLIB option 433
DB2TLIB option 433
DBCS-capable terminal
ZDT/CICS 369
DBDLIB1 option 462
DBDLIBn option 462
DBRC option 463
DBRMs 172
DEDB randomizing modules
customizing for ZDT/IMS 280
default options, changing
for Z Data Tools Base function 55
for ZDT/Db2 180
for ZDT/IMS 264
HFM0POPT 55
HFM1AGNT 264
HFM1POPI 264
HFM1POPT 264
HFM2POPT 180
DESC option 433, 463, 513
DFSDF option 464
DFSORT libraries
adding to the batch JCL skeleton 59
DFSORT, improving Z Data Tools performance 49
DFSORT, using to improve Z Data Tools performance
DFSORT as primary sort product 49
DFSORT is not the primary sort product 50
DFSORT SVC 51
DFSRRC00 option 464
DFSVSAMP option 464
DISPLAY option 433
display translation tables
changing for languages other than English 104, 211, 310, 368
changing for Z Data Tools Base function 62
PRTTRANS option 62
TR* translate tables 62
distribution library names 24
DLI mode 246
DSINFO processing option 397, 397
DSNAME option 516, 522, 527, 533
DSPINC processing option 397
DSPMAX processing option 397
DSPMIN processing option 398
DSPNUM processing option 398
DUMP processing option 398
DYNACB option 465
DYNALLOC option 465
dynamic PSBs 247
dynamic templates, IMS 255, 508
DYNPRFN option 466
DYNPRFX option 467
DYNPSB option 467
DYNTPLT option 468
DYNTPLT parameter 255

E

edit models
enabling 39
EDIT_MAX_ROWS option 434
EDITCAPS option 398, 450
EDITFREQ option 468
EDITOR_TIMEOUT option 435
EDMAXVIRT processing option 398
encryption exit 123
EOD processing option 399
example of HFM1AGNT macro 273
example of HFM1POPD macro 273
example of HFM1POPI macro 273
examples of HFM2POPI macros 178
examples of HFM2SSDM macro 175
EXCI
setting up access for 333
EXCITRAN processing option 400
exits
I/O exit 123
LMS exit 120, 539
scrambling exit 138
ZDT/IMS security exit 282
External CICS interface
setting up access for 333
external CICS interface, support 400

F

fast path for experienced installers xix
FMAUDIT macro 515, 521, 527, 533
FMEDITOR processing option 400
FMOPTMOD macro 516, 522, 527, 533
FORCE_WITH_UR option 436
fullpack access to DASD volumes 70

G

granting access to Db2 catalog 161
GSGNAME processing option 469

H

HEADERPG processing option 400
HFISRV1 381
HFM0MENU, multicultural version of 106
HFM0PARM
options specified in 515
HFM0PARM definitions
customizing 516
HFM0PARM member
tags 517
HFM0POPT 55
HFM0POPT-controlled auditing 89
HFM0RETC 60
HFM1AGNT 264
HFM1AGNT macro 513
HFM1AGNT macro, example 273
HFM1FTEX JCL skeleton 279
HFM1MENU, modifying 311
HFM1PARM
options specified in 521
HFM1PARM definitions
customizing 522
HFM1PARM member
tags 523
HFM1POPD macro, example 273
HFM1POPI 264
HFM1POPI macro, example 273
HFM1POPT 264
controlling audit logging 298
HFM1RNDM module 280
HFM1SXT 289
HFM1UMDM 311, 312
HFM1UMDN 314
HFM1UMDP 264
HFM1UMDS 295
HFM2DENU, modifying 215
HFM2FTEX JCL skeleton 182
HFM2FTSL JCL skeleton 181
HFM2GEN, default ZDT/Db2 reverse engineering plan 173
HFM2MENU, modifying 212
HFM2PARM
options specified in 526
HFM2PARM definitions
customizing 528
HFM2PARM member
defining 193
tags 528
HFM2PLAN 173
HFM2POPI macro 446
HFM2POPI macros
usage tips 180
HFM2POPT 180
HFM2RESS 155
HFM2SSDM macro 426
HFM2SSDM macro, examples 175
HFM2SSDM macros
usage tips 180
HFM2UMDB 182
HFM2UMDE 182
HFM2UMDM 212
HFM2UMDN 217
HFM2UMDP 181
HFM3MENU, modifying 370
HFM3PARM
options specified in 532
HFM3PARM definitions
customizing 534
HFM3PARM member
tags 534
HFM3UMDM 370
HFM3UMDN 372
HFM4POPT 55
HFMAUTH DD usage 379
HFMBDIRS 52
HFMCOB DD statement 45
HFMFTEXC JCL skeleton 57
HFMINIT 25
HFMIOXCx sample copybooks 124
HFMIOXxx sample I/O exits 124
HFMSCXCx sample copybooks 138
HFMSCXxx sample scrambling exits 138
HFMTERM DD statement 44, 45

HFMTRJPN 109
HFMTRTBS 62
HFMUMDD 215
HFMUMODA 64
HFMUMODB 59
HFMUMODH 137
HFMUMODJ 109
HFMUMODM 106
HFMUMODN 109
HFMUMODP 55
HFMUMODR 60
HFMUMODT 63
HFMUMODU 137
HFMUMODX 104
High Level Assembler copybooks 49
HLASM
  customizing for using HLASM copybooks 49
  LANG=HLASM 49
HLASM copybooks 49
HLDBCS processing option 401
HLMAXRTN processing option 401
HLNOALIGN processing option 401
HSYSLIBnn processing option 402

I

I/O exit 123, 123
ICEDFSRB alias 49
ICEDFSRT alias 49
IEBFREQ option 469
IGYCDOPT 46
IMS
  managing ACBs 253
  supported versions 39
IMS databases supported 245
IMS initialization security exit 289
IMS processing modes
  BMP mode 246
  DLI mode 246
IMS region controller parameters 248
IMS subsystem security 249
IMS subsystems
  controlling access by read-only functions 286
  controlling access by update functions 286
  controlling access to 284
  controlling access to individual subsystems by individual functions 287
IMS subsystems, specifying for ZDT/IMS 273
IMS templates 254
IMS termination security exit 289
IMSAUDLG option 470
IMSBKO option 471
IMSGROUP option 471
IMSNBA option 472
IMSOBA option 472
IMSPLEX option 473
installation
  experienced installers xix
  verifying 315
installation of Japanese component
  verifying 150
installation of Z Data Tools Base function
  verifying 148
installation of ZDT/CICS
  verifying 373
installation of ZDT/Db2
  verifying 218
IRLM option 473
IRLMNAME option 474
ISPF alternate libraries 27
ISPF command table
  defining Z Data Tools Base function 37
  defining ZDT/Db2 159
  defining ZDT/IMS 261
ISPF messages
  translating for Z Data Tools Base function 107
  translating for ZDT/Db2 213
  translating for ZDT/IMS 312
ISPF Primary Option Menu
  adding Z Data Tools Base function 36
  adding ZDT/Db2 159
  adding ZDT/IMS 261
  invoking Z Data Tools using LIBDEFs 32
  preparing Z Data Tools to run with LIBDEFs 31
ISPFGLBL processing option 402
ISPFPACK processing option 402

J

Japanese component
  verifying customization 150
  verifying installation 150
Japanese national language
  3277KN or 3278KN terminals 109
  HFMTRJPN 109
  Z Data Tools Base function 109
  ZDT/CICS 372
  ZDT/Db2 217
  ZDT/IMS 314
Java
  supported versions 39
JCL processing option 403
JCL skeletons
  adding the COBOL compiler 45
  Z Data Tools Base function 57
  ZDT/Db2 181
  ZDT/IMS 279

K

keyword
  CERTRUST 383
  REPOS 383

L

language environment
  considerations 54
Language Environment
  runtime library 46
LANGUAGE option keywords 108, 214, 313, 371
LANGUAGE processing option 403
LDFDDLN option 437
LIBDEFs
  invoking Z Data Tools 32
  preparing Z Data Tools 31
libraries
  concatenating to LINKLIST 26
Library Management System libraries (LMSs)
  adding to the batch JCL skeleton 59
  customizing Z Data Tools Base function 120
  See also LMS sample exit
library names 24
LINKLIST
  concatenating Z Data Tools libraries 26
LINKLIST, add the COBOL compiler library 45
LINKLIST, concatenating ZDT/Db2 libraries 155
LINKLIST, concatenating ZDT/IMS libraries 258
LIST option 452, 474
LKEYDATAC processing option 475
LKEYMGMTC processing option 475
LKEYPQTY processing option 475
LKEYSQTY processing option 475
LKEYSTORC processing option 475
LKEYSUNIT processing option 476
LKEYVOLn processing option 476
LMS processing option 404
LMS sample exit 539
  common argument 543
  getting display information 547
  getting member information 546
  getting member records 544
  HLASM considerations 548
  initializing the exit 541
  library name 543
  library validation 544
  main program and first argument 539
  metadata 546
  return codes 540
  supporting multiple LMSs 548
  terminating 542
  tracing 542
LMSUBSYS processing option 404
LOADFREQ option 476
LOADLIB processing option 405
LOCATION option 437
LOCATION_NICKNAME option 437
LOCKMAX option 477
LODINDN option 438
LOGDATAC processing option 477
LOGDSN processing option 478
LOGMGMTC processing option 479
logon messages
  translating for ZDT/CICS 369
logon procedure, TSO 27, 155, 258
LOGPQTY processing option 479
LOGSQTY processing option 479
LOGSTORC processing option 480
LOGSUNIT processing option 480
LOGUNIT processing option 480
LOGUSAGE processing option 480

M

MACLIB option 481
MAXGN option 481
MEMBER option 516, 522, 528, 534
members
  included 519, 525, 531, 537
messages
  translating for Z Data Tools Base function 106
  translating for ZDT/Db2 212
  translating for ZDT/IMS 311
messages, using translated 108, 214, 313, 371
MQ, controlling access to 81
MQREPHLQ processing option 405
MQSeries
  support requirements 39
MSGUPPER processing option 406

N

national language default
  for Z Data Tools Base function 55
  for ZDT/Db2 181
  for ZDT/IMS 273
national languages
  customizing Z Data Tools Base function 104
  customizing ZDT/CICS 368
  customizing ZDT/Db2 211
  customizing ZDT/IMS 310

NOTRUNC processing option 406

O

OAM functions, binding Db2 for 52
OPSCRAM processing option 406
OPTEVT1 option 438
OPTEVT2 option 438
OPTEVT3 option 438
OPTEVT4 option 439
Optim Data Privacy Provider API 122
options
  ABENDCC 384
  ALTCOB 384
  ALTPLI 385
  ASCII 385
  AUDDATAC 386
  AUDITHLQ 386
  AUDITLOG 388
  AUDMGMTC 389
  AUDPQTY 389
  AUDSQTY 389
  AUDSTORC 389
  AUDSUNIT 390
  AUDUNIT 390
  AUXDATAC 390
  AUXDSN 390
  AUXHLQ 391
  AUXMDSN 392
  AUXMGMTC 392
  AUXSTORC 392
  BDY 392
  CCSID 393
  COBDBCS 393
  COBDPC 393
  COBEXTND 393
  COBMAXRTN 394
  COBMCASE 394
  COMMADEC 394
  COMPLANG 395
  CREPLACEn 395
  CSYSLIBnn 396
  CYLHD 396
  DATAHDR 396
  DSINFO 397
  DSPINC 397
  DSPMAX 397
  DSPMIN 398
  DSPNUM 398
  DUMP 398
  EDITCAPS 398
  EDMAXVIRT 398
  EOD 399
  EXCITRAN 400
  FMEDITOR 400
  HEADERPG 400
  HLDBCS 401
  HLMAXRTN 401
  HLNOALIGN 401
  HSYSLIBnn 402
  ISPFGLBL 402
  ISPFPACK 402
  JCL 403
  LANGUAGE 403
  LKEYDATAC 475
  LKEYMGMTC 475
  LKEYPQTY 475
  LKEYSQTY 475
  LKEYSTORC 475
  LKEYSUNIT 476
  LKEYVOLn 476
  LMS 404
  LMSUBSYS 404
  LOADLIB 405
  LOGDATAC 477
  LOGDSN 478
  LOGMGMTC 479
  LOGPQTY 479
  LOGSQTY 479
  LOGSTORC 480
  LOGSUNIT 480
  LOGUNIT 480
  LOGUSAGE 480
  MQREPHLQ 405
  MSGUPPER 406
  NOTRUNC 406
  OPSCRAM 406
  PAD 407
  PAGESIZE 407
  PAGESKIP 407
  PDATAC 408
  PLI31DIGIT 408
  PLI63BIT 408
  PLIGRAPHIC 408
  PLIMAXRTN 409
  PLIUNALIGN 409
  PMGMTC 409
  PRINTDSN 409
  PRINTLEN 410
  PRINTOUT 410
  PRTCLASS 411
  PRTDATAC 411
  PRTDISP 411
  PRTMGMTC 411
  PRTPQTY 412
  PRTSQTY 412
  PRTSTORC 412
  PRTSUNIT 412
  PRTTRANS 413
  PRTUNIT 413
  PSTORC 413
  PSYSLIBnn 413
  PUNIT 414
  RECLIMIT 414
  RKEYDATAC 489
  RKEYMGMTC 489
  RKEYPQTY 489
  RKEYSQTY 489
  RKEYSTORC 489
  RKEYSUNIT 489
  RKEYVOLn 490
  RLS 414
  SEC 415
  SHOWCOPY 415
  SHOWDATAC 454
  SHOWMGMTC 454
  SHOWPQTY 454
  SHOWSQTY 454
  SHOWSTORC 454
  SHOWSUNIT 455
  SHOWUNIT 455
  SMFNO 416
  TAPELBL 416
  TDATAC 417
  TEMPHLQ 417
  TERMTYPE 418
  TMGMTC 418
  TRACECLS 418
  TRACEDSN 419
  TRACELIM 419
  TRACEOUT 419
  TRCDATAC 419
  TRCMGMTC 420
  TRCPQTY 420
  TRCSQTY 420
  TRCSTORC 420
  TRCSUNIT 420
  TRCUNIT 421
  TSTORC 421
  TUNIT 421
  UIEFRDER 496
  ULOGDSN 500
  ULOGUSAG 500
  USEIOX 422
  VIEWAUX 422
  VSAUTO 422
  VSCHGAUTO 423
  VSCHGFRQ 423
  VSSAVE 423
  WBLKSIZE 424
  WIDEPRT 424
  WLRECL 424
  XKEYDATAC 511
  XKEYMGMTC 512
  XKEYPQTY 512
  XKEYSQTY 512
  XKEYSTORC 512
  XKEYSUNIT 512
  XKEYUNIT 513
options, changingcustomizing
  Customizing miscellaneous options in HFM4POPT 383
  for Customizing miscellaneous options in HFM4POPT 383
overview xvii

P

PAD processing option 407
PADS option 482
PAGESIZE processing option 407
PAGESKIP processing option 407
panels
  translating for Z Data Tools Base function 107
  translating for ZDT/CICS 371
  translating for ZDT/Db2 214
  translating for ZDT/IMS 313
panels, using translated 108, 214, 313, 371
PARDLI option 482
PARDLI parameter 268
PARMLIB
  options specified in 515
PDATAC processing option 408
performance
  improving, ZDT/Db2 156
  using DFSORT to improve 49
PL/I
  customizing for using PL/I include books 46
  Language Environment runtime library 46
  show level 46
  supported versions 39
PL/I include books 46
PLAN option 439
PLAN2 option 439
PLI31DIGIT processing option 408
PLI63BIT processing option 408
PLIGRAPHIC processing option 408
PLIMAXRTN processing option 409
PLIUNALIGN processing option 409
PMGMTC processing option 409
Primary Option Menu panel
  customizing ZDT/Db2 160
print translation tables

changing for languages other than English 104, 211, 310, 368
changing for Z Data Tools Base function 62
PRTTRANS option 62
TR* translate tables 62
PRINTDSN processing option 409
PRINTLEN processing option 410
PRINTOUT processing option 410
problems with APF-authorizing, diagnosing 53
processing modes
BMP mode 246
DLI mode 246
processing options
DSINFO 397
PROCLIB option 483
PROCOPTB option 483
PROCOPTP option 484
PROCOPTX option 484
PROCOPTY option 485
PROD_EDIT option 439
profile names for protected functions 76
programming interface information dlxvi
providing a multicultural version of HFM2DENU
ZDT/Db2 215
PRTCLASS processing option 411
PRTDATAC processing option 411
PRTDISP processing option 411
PRTMGMTC processing option 411
PRTPQTY processing option 412
PRTSQTY processing option 412
PRTSTORC processing option 412
PRTSUNIT processing option 412
PRTTRANS processing option 413
PRTUNIT processing option 413
PSB processing
dynamic PSBs 247
static PSBs 247
PSBLIB1 option 485
PSBLIBn option 485
PSBTYPE option 486
PSBTYPES option 486
PSTORC processing option 413
PSYSLIBnn processing option 413
PUNIT processing option 414

R

RACF 65
RACF, using to control access 65
randomizing modules that cause abends
ZDT/IMS 280
RBXWRKN option 440
READONLY option 487
RECLIMIT processing option 414
REGCATLG option 487
REGTYPES option 488
release level
display using VER command 148
Remote Services
preparing for 379
REPOS keyword 383
RESLIB1 option 488
RESLIBn option 488
return codes
customizing for batch utilities 60
reverse engineering plan, ZDT/Db2 173
RKEYDATAC processing option 489
RKEYMGMTC processing option 489
RKEYPQTY processing option 489
RKEYSQTY processing option 489
RKEYSTORC processing option 489
RKEYSUNIT processing option 489
RKEYVOLn processing option 490
RLS processing option 414
ROGUNLN option 441

S

SAF 68
auditing (ZDT/Db2) 192, 195
auditing (ZDT/Db2) using SYS1.PARMLIM 193
controlling auditing 91
controlling auditing using SYS1.PARMLIB 91
controlling auditing without SYS1.PARMLIB 93
used to control ZDT/CICS auditing 300, 357
SAF rules
access levels 95, 196, 360
how they control auditing 95, 196, 360
SAF_CTRL option 515, 521, 527, 533
SAF-controlled auditing
Z Data Tools IMS component 300
Z Data Tools IMS component without SYS1.PARMLIB 303
SAF-rule controlled auditing
testing 206
SAF-rule tables 101, 207, 366
sample jobs
HFM1AXIT 295
HFM1CXIT 295
HFM1IVP 315
HFM1XITA 295
HFM1XITC 295
HFM2BIND 172
HFMBDIRS 52
HFMIOXxx 124
HFMSCXxx 138
HFMVERIF 148
SCEERUN, LE runtime library 46
SCEERUN, LE runtime library, in LINKLIST 26
scrambling exit 138
SEC processing option 415
security
controlling access to BLP 73
controlling access to MQ 81
controlling access using RACF 65
controlling access using SAF 68
controlling fullpack access to DASD volumes 70
HFM1SXT 289
IMS subsystem 249
RACF 65
ZDT/IMS function 249
security considerations
additional 381
security environment
customizing for Z Data Tools 65
customizing for ZDT/IMS 282
security products
non-IBM 70, 289
service provider
installing 112
maintenance 116
overview 111
prerequisites 111
uninstalling 118
setting the LANGUAGE option
ZDT/CICS 368
SHOWCOPY processing option 415
SHOWDATAC processing option 454
SHOWMGMTC processing option 454
SHOWPLI command 46
SHOWPQTY processing option 454
SHOWSORT command
using to determine if DFSORT is available 51
SHOWSQTY processing option 454
SHOWSTORC processing option 454
SHOWSUNIT processing option 455
SHOWUNIT processing option 455
SICELINK, used by Z Data Tools 49
SKELLIB option 491
SLDJCL1 option 441
SLDJCL2 option 441
SLDJCL3 option 441
SLDJCL4 option 441
SMF
audit logging 86
writing audit records to 86
SMF_NO option 516, 521, 527, 533
SMFNO processing option 416
SORTLPA, used by Z Data Tools 49
space timeouts
address 382
specifications
continuing across multiple lines 520, 526, 532, 538
SQL errors from binding Db2 173
SSID option 442, 491, 514
SSIDCMD1 option 455
SSIDCMD2 option 456
static PSBs 247
static templates, IMS 255
STMJCL1 option 442
STMJCL2 option 442
STMJCL3 option 443
STMJCL4 option 443
subsystems, Db2 174, 175
subsystems, IMS 273
support software 39
supported IMS databases 245
SVC, DFSORT 51
syntax notation used in this book xx
SYS1.PARMLIB
SAF-controlled auditing 91
system requirements 39

T

TABLE_LOCKING option 443
tags
as entries in HFM0PARM member 517
as entries in HFM1PARM member 523
as entries in HFM2PARM member 528
as entries in HFM3PARM member 534
example of usage 517, 523, 529, 535
rules for specifying 517, 523, 529, 535
TAPELBL processing option 416
target library names 24
TDATAC processing option 417
TEMPHLQ processing option 417
template repository, setting up 55
templates
IMS 254
temporary, views and criteria sets 255
temporary view or criteria set
and templates 255
TERMTYPE processing option 418
Thai language, sample translation table 104
TIMEOUTI option 491
TMGMTC processing option 418
TMINAME option 491
TMPDDLN option 444

TPLLIB1 option 492
TPLLIBn option 492
TPLLIBn parameters 255
TPREPOS ddname 55
TRACECLS processing option 418
TRACEDSN processing option 419
TRACELIM processing option 419
TRACEOUT processing option 419
translating logon messages
  ZDT/CICS 369
translating message text
  ISPF messages 107, 213, 312
  modifying HFM0MENU 106
  modifying HFM1MENU 311
  modifying HFM2MENU 212
  modifying HFM3MENU 370
  Z Data Tools Base function 106
  ZDT/Db2 212
  ZDT/IMS 311
translating object list utility statements
  modifying HFM2DENU 215
translating panel text
  Z Data Tools Base function 107
  ZDT/CICS 371
  ZDT/Db2 214
  ZDT/IMS 313
TRCDATAC processing option 419
TRCMGMTC processing option 420
TRCPQTY processing option 420
TRCSQTY processing option 420
TRCSTORC processing option 420
TRCSUNIT processing option 420
TRCUNIT processing option 421
TSO ALTLIB 155
TSO logon procedure, modifying
  for Z Data Tools Base function 27
  for ZDT/Db2 155
  for ZDT/IMS 258
  using a CLIST or REXX exec 155
TSTORC processing option 421
TUNIT processing option 421
type A exit parameters 291
type D exit parameters 293
type I exit parameters 293
TYPE option 444

U

UACBLIB option 492
UAGNS option 493
UAUTOSAV option 493
UBUF option 494
UDBDLIB option 494
UDBRC option 495
UDFSVSMP option 495
UIEBFREQ option 496
UIEFRDER processing option 496
UIMSBKO option 497
UIMSNBA option 497
UIRLM option 498
ULOADFRQ option 498
ULOCKMAX option 499
ULOGDSN processing option 500
ULOGUSAG processing option 500
UMACLIB option 501
UNLPUNN option 444
UNLUNLN option 444
unprotected functions 76
UPARDLI option 501
UPROCOPB option 502
UPROCOPP option 502
UPROCOPX option 503
UPROCOPY option 504
UPSBLIB option 504
UPSBTYPE option 505
URECON option 505
URESLIB option 506
URSR option 506
usage rules for views and criteria sets, configuring 256, 256, 508, 509
usage rules, for views and criteria sets 256
USEDDL option 507
USEIOX option
  indicating a user I/O exit 123
  providing a user I/O exit 137
USEIOX processing option 422
user I/O exit 123
  copybooks 124
  sample programs 124
user scrambling exit 138
  copybooks 138
  sample programs 138
USER_SELECT_EDIT option 445
usermods
  HFM1UMDM 311, 312
  HFM1UMDN 314
  HFM1UMDP 264
  HFM1UMDS 295
  HFM2UMDB 182
  HFM2UMDE 182
  HFM2UMDM 212
  HFM2UMDN 217
  HFM2UMDP 181
  HFM3UMDM 370
  HFM3UMDN 372
  HFMUMDD 215
  HFMUMODA 64
  HFMUMODB 59
  HFMUMODH 137
  HFMUMODJ 109
  HFMUMODM 106
  HFMUMODN 109
  HFMUMODP 55
  HFMUMODR 60
  HFMUMODT 63
  HFMUMODU 137
  HFMUMODX 104
users who should use this book xix
using the translated messages and panels
  Z Data Tools Base function 108
  ZDT/CICS 371
  ZDT/Db2 214
  ZDT/IMS 313
UTPLLIB option 508

V

VCURUDT option 508
VCURUDT parameter 256
VCURULE option 509
VCURULE parameter 256
VER command 148
  using to determine APF-authorization 54
VIEWAUX processing option 422
views
  configuring usage rules 256, 256, 508, 509
  temporary, and templates 255
  usage rules 256
VS Code extension 377
VSAM default editor and browser 38
VSAUTO processing option 422
VSCHGAUTO processing option 423
VSCHGFRQ processing option 423
VSMPMEM option 510
VSSAVE processing option 423

W

WBLKSIZE processing option 424
WebSphere MQ
  customizing to access 41
WIDEPRT processing option 424
WLRECL processing option 424

X

XDOPTLB option 510
XKEYDATAC processing option 511
XKEYMGMTC processing option 512
XKEYPQTY processing option 512
XKEYSQTY processing option 512
XKEYSTORC processing option 512
XKEYSUNIT processing option 512
XKEYUNIT processing option 513

Z

Z Common Components (ZCC)
  server configuration 381, 381
Z Data Tools
  customizing to write audit records to SMF 86
Z Data Tools functions
  controlling auditing of 100
Z Data Tools options 383
Z Data Tools Service Provider 111
Z Data Tools subsystem options
  DSNAME 516
  MEMBER 516
  SAF_CTRL 515
  SMF_NO 516
Z Data Tools supported software 39
Z Data Tools user I/O exit 123
  close call 135
  exit control block description 126
  exit protocol 124
  initialization call 134
  installing the exit 136
  open call 135
  read and write calls 136
  termination call 135
  using the I/O exit control block 133
  writing the exit 124
Z Data Tools user scrambling exit 138
  exit control block description 139
  exit protocol 138
  installing the exit 146
  using the scrambling exit control block 145
  writing the exit 138
z/OS
  supported versions 39
z/OS Connect API
  service provider 111
ZCC server
  HFISRV1 381
ZDT commands and applications
  avoiding conflicts 38
ZDT/CICS
  customizing audit facility 352
ZDT/CICS functions
  controlling auditing of 366
ZDT/CICS subsystem options
  DSNAME 533
  MEMBER 534
  SAF_CTRL 533
  SMF_NO 533
ZDT/Db2
  adding to the Db2 Admin Launchpad 165
  binding Db2, required customization 172

customizing audit facility 189
Db2 subsystem selection panel, required customization 174
DBRMs 172
granting access to Db2 catalog 161
re-installing after migration of Db2 system 242
running multiple versions 173
running with APF-authorization 156
starting from an external application 167
starting from Db2 Admin Tool 166
ZDT/Db2 commands
customizing 178
ZDT/Db2 customization, required 153
ZDT/Db2 options
CATOWNER 448
CONNECT 449
EDITCAPS 450
HFM2POPI macro 446
LIST 452
SSIDCMD1 455
SSIDCMD2 456
ZDT/Db2 Options 426
ZDT/Db2 subsystem options
ATTACH 426
AUDIT 427
AUDITBROWSE 428
AUTH_ACCESS 428
AUTO_COMMIT 429
CATOWNERCDRM 446
CPYCPYN 430
DB2CLIB 430
DB2ELIB 430
DB2LLIB 430, 431
DB2MLIB 431
DB2PLIB 432
DB2PROC 432
DB2RLIB 432
DB2SLIB 433
DB2TLIB 433
DESC 433
DISPLAY 433
DSNAME 527
EDIT_MAX_ROWS 434
EDITOR_TIMEOUT 435
FORCE_WITH_UR 436
HFM2SSDM macro 426
LDFDDLN 437
LOCATION 437
LOCATION_NICKNAME 437
LODINDN 438
MEMBER 528
OPTEVT1 438
OPTEVT2 438
OPTEVT3 438
OPTEVT4 439
PLAN 439
PLAN2 439
PROD_EDIT 439
RBXWRKN 440
ROGUNLN 441
SAF_CTRL 527
SLDJCL1 441
SLDJCL2 441
SLDJCL3 441
SLDJCL4 441
SMF_NO 527
SSID 442
STMJCL1 442
STMJCL2 442
STMJCL3 443
STMJCL4 443
TABLE_LOCKING 443
TMPDDLN 444
TYPE 444
UNLPUNN 444
UNLUNLN 444
USER_SELECT_EDIT 445
ZDT/IMS
customizing the security environment 282
ZDT/IMS Batch JCL skeleton
customizing for the COBOL compiler library 279
ZDT/IMS function security 249
ZDT/IMS functions
controlling access to 284
ZDT/IMS options 457
ACBLIB 458
ACBMGMT 458
ACBSHR 459
AUTOSAVE 459
BSDSHLQ 460
CATALIAS 460
CHGAFREQ 460
CHKPINTVL 461
COMPAT 461
DBDLIB1 462
DBDLIBn 462
DBRC 463
DESC 463
DFSDF 464
DFSRRC00 464
DFSVSAMP 464
DYNACB 465
DYNALLOC 465
DYNPRFN 466
DYNPRFX 467
DYNPSB 467
DYNTPLT 468
EDITFREQ 468
GSGNAME 469
IEBFREQ 469
IMSAUDLG 470
IMSBKO 471
IMSGROUP 471
IMSNBA 472
IMSOBA 472
IMSPLEX 473
IRLM 473
IRLMNAME 474
LIST 474
LOADFREQ 476
LOCKMAX 477
MACLIB 481
MAXGN 481
PADS 482
PARDLI 482
PROCLIB 483
PROCOPTB 483
PROCOPTP 484
PROCOPTX 484
PROCOPTY 485
PSBLIB1 485
PSBLIBn 485
PSBTYPE 486
PSBTYPES 486
READONLY 487
REGCATLG 487
REGTYPES 488
RESLIB1 488
RESLIBn 488
SKELLIB 491
SSID 491
TIMEOUTI 491
TMINAME 491
TPLLIB1 492
TPLLIBn 492
UACBLIB 492
UAGNS 493
UAUTOSAV 493
UBUF 494
UDBDLIB 494
UDBRC 495
UDFSVSMP 495
UIEBFREQ 496
UIMSBKO 497
UIMSNBA 497
UIRLM 498
ULOADFRQ 498
ULOCKMAX 499
UMACLIB 501
UPARDLI 501
UPROCOPB 502
UPROCOPP 502
UPROCOPX 503
UPROCOPY 504
UPSBLIB 504
UPSBTYPE 505
URECON 505
URESLIB 506
URSR 506
USEDDL 507
UTPLLIB 508
VCURUDT 508
VCURULE 509
VSMPMEM 510
XDOPTLB 510
ZDT/IMS Options
AGN 513
DESC 513
HFM1AGNT macro 513
SSID 514
ZDT/IMS security
AGN security 249
common exit parameters 290
controlling access 284
controlling access to individual functions 286
controlling access to read-only functions 285
controlling access to update functions 285
Controlling DB access by ZDT/IMS functions 248
Database Access Control facility 282
exit type A parameters 291
exit type D parameters 293
exit type I parameters 293
HFM1SXT 248, 289
invoking HFM1SXT 290
invoking the security exit 290
RACF PADS considerations 250
sample programs 295
security exit 248
security exit parameters 291
types of security exit 289
ZDT/IMS security exit sample programs
HFM1AXIT 295
HFM1CXIT 295
HFM1SECC 295
HFM1UMDS 295
HFM1XITA 295
HFM1XITC 295
ZDT/IMS subsystem options

DSNAME 522
MEMBER 522
SAF_CTRL 521
SMF_NO 521
ZDT/IMS, customizing IMS 262
ZDT/IMS, declaring dynamic PSBs 262
ZDT/IMS, providing a DOPT ACBLIB data set 263