

**HCL Informix 14.10 - High-
Performance Loader User's Guide**



Contents

Chapter 1. High-Performance Loader User's Guide.....	4
High-Performance Loader overview.....	4
Overview of HPL features.....	5
The HPL data-load process.....	5
The HPL data-unload process.....	6
HPL loading modes.....	7
HPL components.....	8
Environment variables needed for the HPL.....	10
Architecture of the onpload utility.....	13
Examples of loading and unloading jobs using the ipload utility.....	18
Prepare to use the ipload utility.....	19
Create a file of data.....	19
Create a database.....	19
The ipload utility.....	20
Load Job windows.....	22
Device-Array windows.....	24
Format windows.....	26
Filter, Discard Records, and Logfile text boxes.....	30
Map Views window.....	31
Completing the Load Record Maps window.....	33
Map-Definition window.....	34
Load Options window.....	37
Running the ipload job.....	38
Active Job window.....	39
The ipload utility Generate options.....	40
The ipload utility windows.....	45
The ipload utility GUI or the onpladm command- line interface.....	45
The ipload utility GUI.....	45
The HPL ipload utility buttons.....	57
The HPL online help.....	66
The UNIX™ keyboard commands to move the cursor.....	66
Define HPL projects.....	67
HPL projects.....	67
Project organization.....	67
Select or create a project with the Projects window.....	68
Configure the High-Performance Loader.....	70
Configure the ipload utility.....	70
Select a database server.....	71
Modify the onpload default values.....	72
Selecting a driver.....	75
Modify the machine description.....	76
Define device arrays.....	79
Device arrays.....	79
Multiple devices in a device array.....	80
The Device-Array Selection window.....	80
The Device-Array Definition window.....	81
Define formats.....	84
Formats.....	85
Formats of supported datafile records.....	85
Fixed-length records.....	86
Delimited records.....	97
COBOL records.....	100
Other formats.....	103
Format options.....	103
Format Views window.....	107
Define queries.....	108
HPL queries.....	108
Creating a query.....	109
Editing a query.....	115
Exporting and importing queries.....	116
The Database Views window.....	118
Define maps.....	119
Load and unload maps.....	119
Unload maps.....	124
Mapping options.....	127
Editing options.....	130
Map Views window.....	132
Define filters.....	135
Filters.....	135
Example of using filter.....	135
Creating a filter.....	136
Modifying a filter.....	139
Filter views.....	141
Filters with code-set conversion (GLS).....	142
Unload data from a database.....	142
Unload jobs.....	142
Components of the unload job.....	142
The Unload Job windows.....	143
Generate options for an unload job.....	149
Load data to a database table.....	150
Load jobs.....	150
Components of the load job.....	150
The Load Job windows.....	153
Generate options for a load job.....	160
The Generate options of the ipload utility.....	160
Overview of the ipload Generate options.....	161
Tasks that generate load or unload components.....	161

Generate from the Load Job window.....	161
Generate from the Unload Job window.....	162
Generate from the Components menu.....	165
The HPL browsing options.....	170
Browsing options.....	170
Manage the High-Performance Loader.....	177
Manage modes, errors, and performance.....	177
HPL modes.....	177
HPL load and unload errors.....	181
HPL performance.....	182
Limitation when using the Excalibur Text DataBlade® Module indexes.....	189
The onpload utility.....	189
Overview of the onpload utility.....	190
The onpload file name size limitations on UNIX™.....	190
Start the onpload utility.....	190
Using the onpload utility.....	191
The onpload utility syntax.....	191
Set the onpload run mode with the -f option.....	193
Modify the size of onpload database parameters.....	195
Override the onpload database values.....	198
Load data into collection data type columns.....	198
The onpladm utility.....	199
Overview of the onpladm utility.....	199
Define onpladm utility jobs.....	202
Define device arrays.....	214
Define maps.....	217
Define formats.....	223
Define queries.....	228
Define filters.....	230
Define projects.....	233
Define machine types.....	235
Define database operations.....	237
Index.....	241

Chapter 1. High-Performance Loader User's Guide

These topics describe how to use the HCL Informix® High-Performance Loader (HPL) to load and unload large quantities of data efficiently to or from Informix® databases.

These topics describe:

- The architecture of HPL
- The onpload utility that allows you to control HPL from a script.
- The onpload database that maintains information about load and unload jobs that you have prepared.
- The ipload graphical user interface (GUI).
- The onpladm utility.

The ipload utility is a UNIX™ application that helps users prepare load and unload jobs for both UNIX™ and Windows™. The onpladm utility is a command-line version of the ipload utility that operates on both UNIX™ and Windows™.

The first section introduces the HPL, provides a general overview of the tasks that the HPL performs, describes the architecture of the HPL, and includes two tutorial examples that take you through the process of loading and unloading data.

The second section introduces the ipload utility, a graphical user interface that you can use to set the parameters for the HPL.

Subsequent sections give details about developing the onpload database by using the individual components of the ipload, onpload, and onpladm utilities.

These topics are written for the following users:

- Database administrators
- Database server administrators

These topics assume that you have the following background:

- A working knowledge of your computer, your operating system, and the utilities that your operating system provides.
- Some experience working with relational databases or exposure to database concepts.
- Some experience with computer programming.
- Some experience with database server administration, operating-system administration, or network administration.

For information about software compatibility, see the HCL Informix® release notes.

These topics are taken from *IBM® Informix® High-Performance Loader User's Guide*.

High-Performance Loader overview

Overview of HPL features

The High-Performance Loader (HPL) is a database server tool that you can use to load and unload large quantities of data efficiently to or from a database.

The HPL lets you exchange data with tapes, data files, and programs, and converts data from these sources into a format compatible with HCL Informix® databases. The HPL also allows you to manipulate and filter the data as you perform load and unload operations.

The HPL includes the following components:

- The `ipload` utility, which is a graphical user interface (GUI). This UNIX™ application helps you prepare load and unload jobs for both UNIX™ and Windows™ and includes the following functionality:
 - The `ipload` utility provides a **Generate** option that lets you automatically generate the HPL components that are required for a load or unload job.
 - The database-load feature lets you update your databases with data from any of the supported file types, while allowing you to control the format and selection of records from the input files. Data can be loaded from or unloaded to files, tapes, or application pipes (for UNIX™), or to any combination of these three device types.
 - The load and unload operations run in the background of your multitasking operating system. Once the operation begins, you can continue to use `ipload` to perform other functions.
 - The `ipload` utility provides context-sensitive online help. The online help also includes a glossary.
- The `onpladm` utility, which is a command-line version of the `ipload` utility.
- The `onpload` utility, which allows you to control HPL from a script.
- The onpload database, which maintains information about load and unload jobs that you have prepared.

You use the `ipload` utility to manage the onpload database on any database server on your network.

Any database server on your network can use the onpload database, which contains parameters and controls that the HPL uses. This accessibility allows centralized management of your load and unload controls. These parameters and controls include the HPL components such as formats, maps, and projects.

The HPL:

- Supports COBOL, ASCII, multibyte, delimited, or binary data.
- Can load and unload data of a different GLS locale than that of the database server.
- Provides synonym support for tables that are valid for the local database server. You can use synonyms for both the load and unload operations.
- Provides support for unloading data with a query that accesses a view in its SELECT statement.
- Supports loading of raw tables in express mode.

The HPL data-load process

The data-load process reads a source data file, converts the data to a different format, and inserts the converted data into a database table.

The source data can come from one or more of the following sources:

- Files
- Tapes
- Pipes (application-generated data) (UNIX™)

The High-Performance Loader (HPL) supports load and unload data greater than 2 GB to files and tapes.

During conversion, the source data is often manipulated so that the converted data displays different characteristics. Examples of this manipulation include:

- Changing lowercase letters to uppercase letters
- Loading default values, loading certain table columns, or replacing nulls
- Masking the data to include only part of a value
- Converting from one data type to another, such as conversion of a numeric string to a float
- Converting from the code set of one locale to the code set of another locale

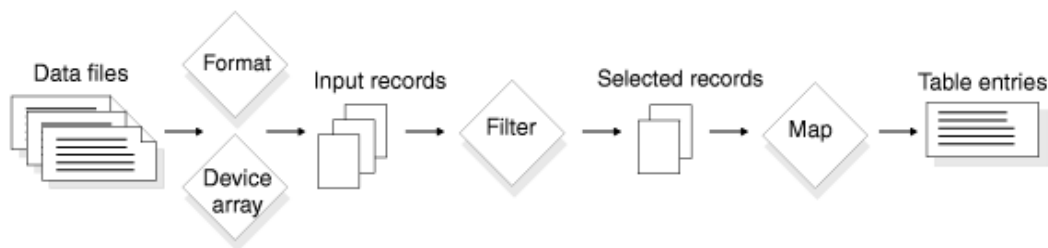
When you prepare to run a data load with the HPL, you describe the actions that the HPL must take by defining a set of metadata *components*. The components describe different aspects of the load process. The HPL uses information from:

- The device array to find the set of the source-data files
- The format to define the data types and layout of the source data
- The filter to select the records from the source data that should be written to the database table
- The map to modify and reorganize the data

The `ipload` utility helps you prepare the components. [Load data to a database table on page 150](#), addresses the process of loading a file to a database.

The following figure shows the data load process. The figure summarizes the data-load process by showing how data moves from data files to table entries.

Figure 1. The data-load process



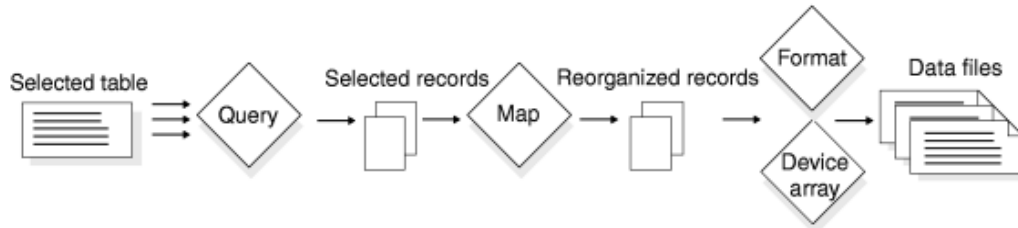
The HPL data-unload process

The data-unload process is essentially the same as the load process, but in reverse. The data-unload process extracts the source data from one or more database tables; converts the data to a new format; and writes the converted data to a file, tape, or on UNIX™ to a pipe (application). As in a load, you can manipulate the data from a database table so that the converted data displays different characteristics.

The following figure shows how the components of the High-Performance Loader (HPL) affect the data as it moves from a database to data files during the unload process. The HPL uses:

- The query to select records from the database
- The map to reorganize or modify the selected records
- The format to prepare the records for writing out to the data files
- The device array to find the location of the data files

Figure 2. The data-unload process



The HPL uses the same components for an unload job as for a load job, with one exception. For an unload job, ipload creates a Structured Query Language (SQL) query that extracts selected data from the table. As with a load, unload components are grouped together into an unload job. Unload jobs can be saved, retrieved, and rerun as many times as necessary. Unload jobs can be grouped together with load jobs in the same project.

Related reference

[Unload data from a database on page 142](#)

HPL loading modes

The High-Performance Loader (HPL) offers two load modes, deluxe mode and express mode. The express mode is faster, and the deluxe mode is more flexible.

The HPL express mode

The HPL express-mode loads are faster than deluxe-mode loads, but less flexible. In express mode, you cannot update the table or read the new data entries until the load is complete. The express mode disables indexes, constraints, and triggers during the load. After the load, HPL rebuilds and re-enables indexes, evaluates and re-enables constraints, if possible, and re-enables triggers. (HPL does not evaluate triggers with respect to the loaded data.)

You must perform a level-0 backup after an express-mode load.

The HPL deluxe mode

The HPL deluxe-mode loads are not as fast as express-mode loads, but are more flexible. In deluxe mode, you can access and update the table that is being loaded. The deluxe mode updates indexes, performs constraint checking, and evaluates triggers as data is inserted into the table.

Related reference

[Manage the High-Performance Loader on page 177](#)

HPL components

The HPL consists of the onpload utility, the ipload utility, the onpladm command-line utility, and the onpload database.

The onpload utility

The onpload utility has the following features:

- Converts, filters, and moves data between a database and a storage device
- Uses information from the onpload database to run the load and unload jobs and to convert the data
- Records information during a load about data records that do not meet the load criteria

The onpload utility can load or unload data from files that are larger than 2 GB and can generate `.log`, `.rej` and `.flt` files that are larger than 2 GB.

The onpload database must be accessible from the machine on which the onpload utility is run. It does not need to be on the same machine.

The database that the onpload loads or unloads must be in a database server instance on the machine that is running the onpload (or ipload or onpladm) utility.

You can start onpload by using ipload or from the command line.

When you start onpload from ipload, the onpload and ipload utilities communicate by using a socket connection. The onpload utility runs independently from ipload (see the information to the right of the vertical line in [Figure 3: Relationships among the parts of the HPL on page 10](#)).

Related reference

[The onpload utility on page 189](#)

Related information

[Architecture of the onpload utility on page 13](#)

The ipload utility

The ipload utility is a UNIX-based graphical user interface that has the following features:

- Creates and manages the onpload database
- Creates and stores information for onpload
- Lets you create, edit, and group the components of the load and unload jobs
- Stores information about the load components in the database

You can use `ipload` to manage onpload databases on both UNIX™ and Windows™. The `ipload` utility is not available on Mac OS X.

Related reference

[The `ipload` utility on page 20](#)

[Configure the `ipload` utility on page 70](#)

Run load and unload jobs on a Windows computer

Related information

[Start the `ipload` utility on page 45](#)

The `onpladm` utility

The `onpladm` utility is a command-line utility for both UNIX™ and Windows™ with the same functionality as `ipload`.

Related reference

[The `onpladm` utility on page 199](#)

The onpload database

The onpload database has the following features:

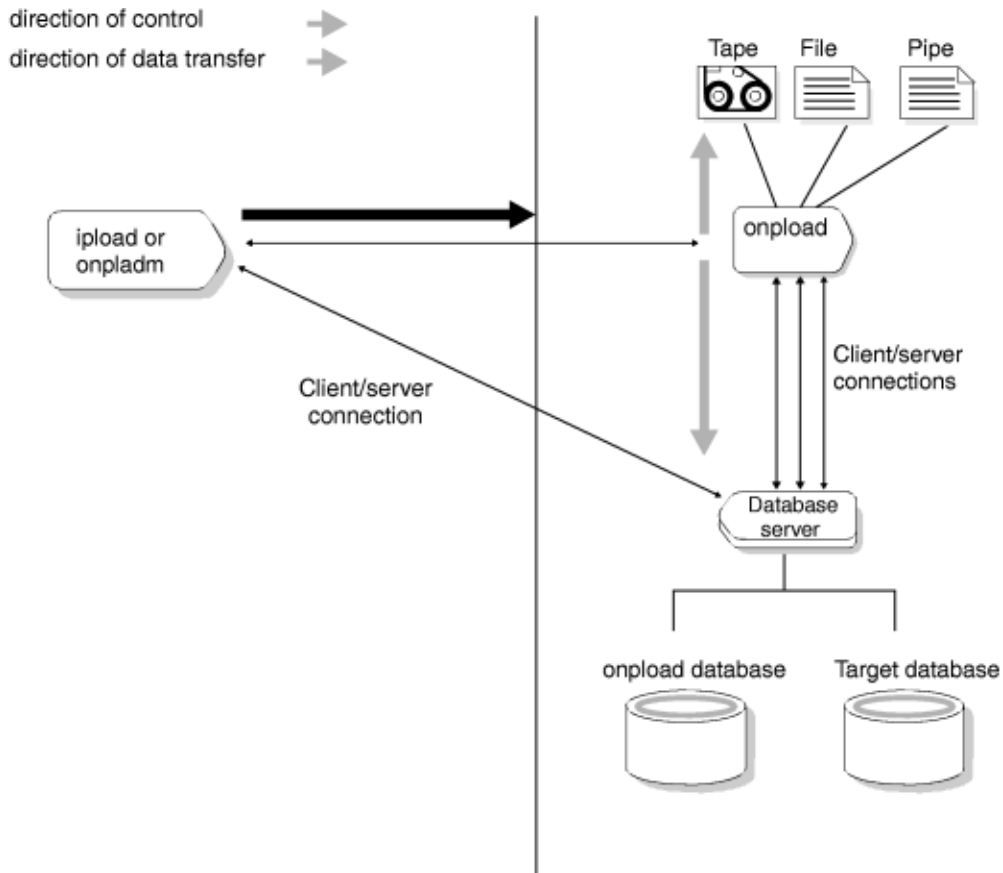
- Contains information that the onpload utility requires to perform data loads and unloads
- Must be accessible from the machine where the onpload utility is run. It does not need to be on the same machine.

However, the database that the utility loads or unloads must be in a database server instance on the machine where the onpload utility (and hence the `ipload` or `onpladm` utility) is being run.

Relationships among the parts of the HPL

The following figure shows the relationships among the parts of the High-Performance Loader (HPL). The `ipload` utility or `onpladm` command-line utility connects to the database server to populate the onpload database. The `ipload` utility or the `onpladm` utility controls the onpload utility, which uses multithreaded architecture to make multiple connections to the database server and multiple I/O connections to tapes, files, or pipes.

Figure 3. Relationships among the parts of the HPL



The information to the right of the vertical line in [Figure 3: Relationships among the parts of the HPL on page 10](#) shows onpload loading or unloading data with no interaction from iupload.

Environment variables needed for the HPL

The High-Performance Loader (HPL) is part of the database server, so you must start the database server before you use the HPL. Before you start the database server and use HPL, you must set environment variables.

The following environment variable must be set:

- **INFORMIXDIR**
- **ONCONFIG**
- **INFORMIXSERVER**
- **LD_LIBRARY_PATH**

Some computers use the **LD_LIBRARY_PATH** environment variable for shared libraries. The name of this environment variable is platform-dependent. See your operating system documentation for the name of the environment variable that specifies the search path for shared libraries. Then see the machine notes for information about **LD_LIBRARY_PATH**.

You can use the **IFX_ONPLOAD_AUTO_UPGRADE** environment variable with the iupload or onploadm utilities to automatically upgrade the onpload database the first time you run an HPL utility by using the iupload or onploadm command after you

migrate to a new database server version. You cannot use the **IFX_ONPLOAD_AUTO_UPGRADE** environment variable with the onpload utility.

In addition to the environment variables listed above, the following environment variables pertain to the HPL:

- **DBONPLOAD**
- **PLCONFIG**
- **PLOAD_SHMBASE**
- **PLOAD_LO_PATH**
- **PLOAD_SHMAT**



Tip: To maximize available memory and scan resources, HPL automatically sets the **PDQPRIORITY** environment variable to `100`, if it is not already set. If the **PDQPRIORITY** environment variable is set, HPL uses that value. If the **PDQPRIORITY** environment variable is set to `0`, then HPL cannot unload multiple devices.

Related information

[INFORMIXDIR environment variable on page](#)

[INFORMIXSERVER environment variable on page](#)

[ONCONFIG environment variable on page](#)

[PDQPRIORITY environment variable on page](#)

Preparing multiple onpload databases

Each database server can have only one active onpload database. In most cases, you can use the default onpload database, which is named onpload. If you choose to prepare multiple databases to hold different types of control information, you must set the **DBONPLOAD** environment variable before you can run a load or unload.

To prepare multiple onpload databases with ipload:

1. Set the **DBONPLOAD** environment variable to the name of the alternative onpload database.
2. Restart ipload.
3. Use ipload to prepare the alternative database.

Results

During load and unload jobs, the ipload, onpload, and onpladm utilities use the value of the **DBONPLOAD** environment variable that is set in the client environment, regardless of whether the **DBONPLOAD** environment variable is set on the server. The ipload, onpload, and onpladm utilities use the value of the **DBONPLOAD** environment variable that is set on the server only if the **DBONPLOAD** environment variable is not set in the client environment.

Related information[DBONPLOAD environment variable on page](#)[Create the onpload database on page 72](#)

The PLCONFIG environment variable

The default configuration file for the onpload utility is `plconfig.std`. The configuration file is always in the `$INFORMIXDIR/etc` directory. To use an alternative configuration file, you must set the **PLCONFIG** environment variable to the name of the alternative onpload configuration file. If you use `plconfig.std`, you do not need to set **PLCONFIG**.

Related reference

High-Performance Loader configuration file

Related information[PLCONFIG environment variable on page](#)

The PLOAD_SHMBASE environment variable

The **PLOAD_SHMBASE** environment variable allows you to specify shared-memory address attachments specifically for onpload processes. You can set the **PLOAD_SHMBASE** environment variable to avoid shared-memory collisions between onpload and the database server or allow the HPL to specify the attachments.



Tip: To use the **PLOAD_SHMBASE** environment variable, you must start onpload from the command line, not from within `ipload`.

Related information[PLOAD_SHMBASE environment variable on page](#)

Avoid shared-memory collision

Both the database server and onpload allocate shared memory. When onpload starts up, the database server allocates shared memory for buffers for onpload processes. The onpload utility also allocates shared memory for its internal use. Because of the dynamic nature of shared-memory allocations, shared-memory collisions can occur between onpload and the database server. If this collision occurs, the onpload job can also fail and error messages are sent to the onpload log file or the database server log file.

To verify if a collision has occurred, use the `onstat -g seg` option. Check for overlap between the shared-memory segments that the database server is using and the SHMBASE reported in the onpload log file.

Related information

[onstat -g seg command: Print shared memory segment statistics on page](#)

Set the PLOAD_SHMBASE environment variable

To override the initial shared-memory allocation in a shared-memory collision between onpload and the database server, set the **PLOAD_SHMBASE** environment variable to a value much higher or much lower than the value for the shared memory that the database server uses.

The PLOAD_LO_PATH environment variable

The **PLOAD_LO_PATH** environment variable allows you to specify the location of smart large object (CLOB or BLOB) output files. If this environment variable is not set, the database server puts the output files in `/tmp`.

Related information

[PLOAD_LO_PATH environment variable on page](#)

The PLOAD_SHMAT environment variable

If the pload converter thread cannot attach to the passed address, you receive a message that asks you to set the **PLOAD_SHMAT** environment variable.

When you set the **PLOAD_SHMAT** environment variable, the pload converter calculates the address by using a global attached segment list that is maintained across pload virtual processors. The pload converter attaches at the next available address after the highest address on the list, ensuring that the converter always attaches to an unused shared memory segment.

Architecture of the onpload utility

The ipload utility is the interface that allows you to prepare the parameters that the onpload utility uses. The onpload utility, which is a client application that attaches to the database server, actually loads and unloads the data.

The onpload utility can take advantage of parallel processing to perform both I/O and data conversion as efficiently as possible because it uses the same multithreading architecture that the database server uses.

The following sections describe how onpload uses multithreading for deluxe loads, express loads, and unloads.

Related information

[The onpload utility on page 8](#)

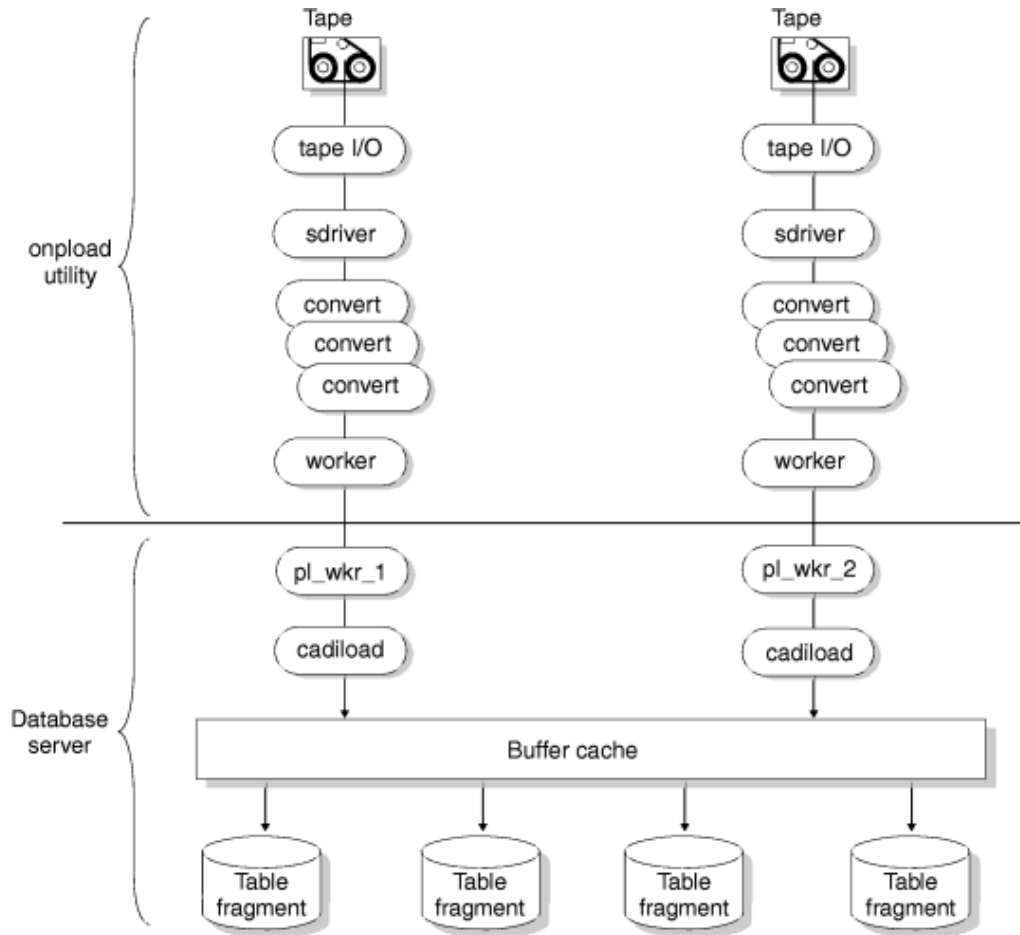
[Save memory and resources on page](#)

The onpload utility deluxe-mode process

The onpload utility uses specific threads during the deluxe-mode process.

The following figure shows the threads that onpload uses in a deluxe-mode load process. In deluxe mode, data is subject to the same constraints as if it were loaded by using SQL INSERT statements.

Figure 4. A deluxe-mode load



Related information

[The HPL deluxe mode on page 177](#)

Threads that the onpload utility uses

The onpload utility starts the following threads:

Table 1. Threads that the onpload utility starts

Thread	Description
tape I/O	The onpload utility starts one tape I/O thread for each tape device. It reads data from the tape device asynchronously.

Table 1. Threads that the onpload utility starts (continued)

Thread	Description
thre ads	The onpload utility starts a similar thread for piped output. (UNIX™ only) If the disk file inputs, onpload uses the multithreading AIO subsystem instead of a dedicated I/O thread.
sdriver thre ads	The worker threads control I/O from input files. They handle device abstraction for the different device types handled. The sdriver threads also are responsible for passing out records from the input and passing records to the converters.
conv ert thre ads	The onpload utility starts one or more convert threads for each device. These threads perform conversions on the input data such as uppercase to lowercase conversion or code-set conversion.
worker thre ads	The onpload utility starts one worker thread for each input device. These threads communicate with the database server. The main responsibility of the worker thread is to pass data to the database server.

To see the status of the onpload threads, you must use the **-j** option of the onstat utility.

Related reference

The onstat -j option

Threads that the database server uses

The database server uses the following threads to insert the data into the database:

Table 2. Threads that insert data into the database

Thread	Description
pl_wkr threads	Each worker thread of the onpload utility is paired with a pl_wkr thread in the database server. These threads receive the data from onpload. In a utility that shows the database server status, the pl_wkr threads are named pl_wkr_1 , pl_wkr_2 , pl_wkr_3 , and so on.
cadil oad threads	The cadiload threads are the insert threads. The insert threads perform a normal insert into the database, like during an INSERT statement. The SQL optimizer governs the method that is used for inserting the data.

The onpload utility express-mode load process

The onpload utility behaves the same way during express-mode deluxe-mode loads. However, the database server behaves differently during an express load.

During an express-mode load, the **pl_wkr** threads pass the data to **stream** threads (also called **fragmenter** threads) that decide where the data is to be stored. The **fragmenter** threads pass the data to an exchange that distributes the data to **setrw** threads. The **setrw** threads write table rows to disk a page at a time, bypassing the buffer cache.

The number of input devices can be different from the number of table fragments. The exchange operator handles multiplexing of data. The data is processed in parallel with respect to the data read from the device array and also with respect to the data written out to table fragments on separate disks. Pipeline parallelism is also present in the data flow from input devices to table fragments on disk. Parallelism is the main mechanism for achieving high performance.

During express-mode load, the database server writes the data to new extents on disk, but those extents are not yet part of the table. At the end of an express-mode load, the database server adds the new extents to the table.

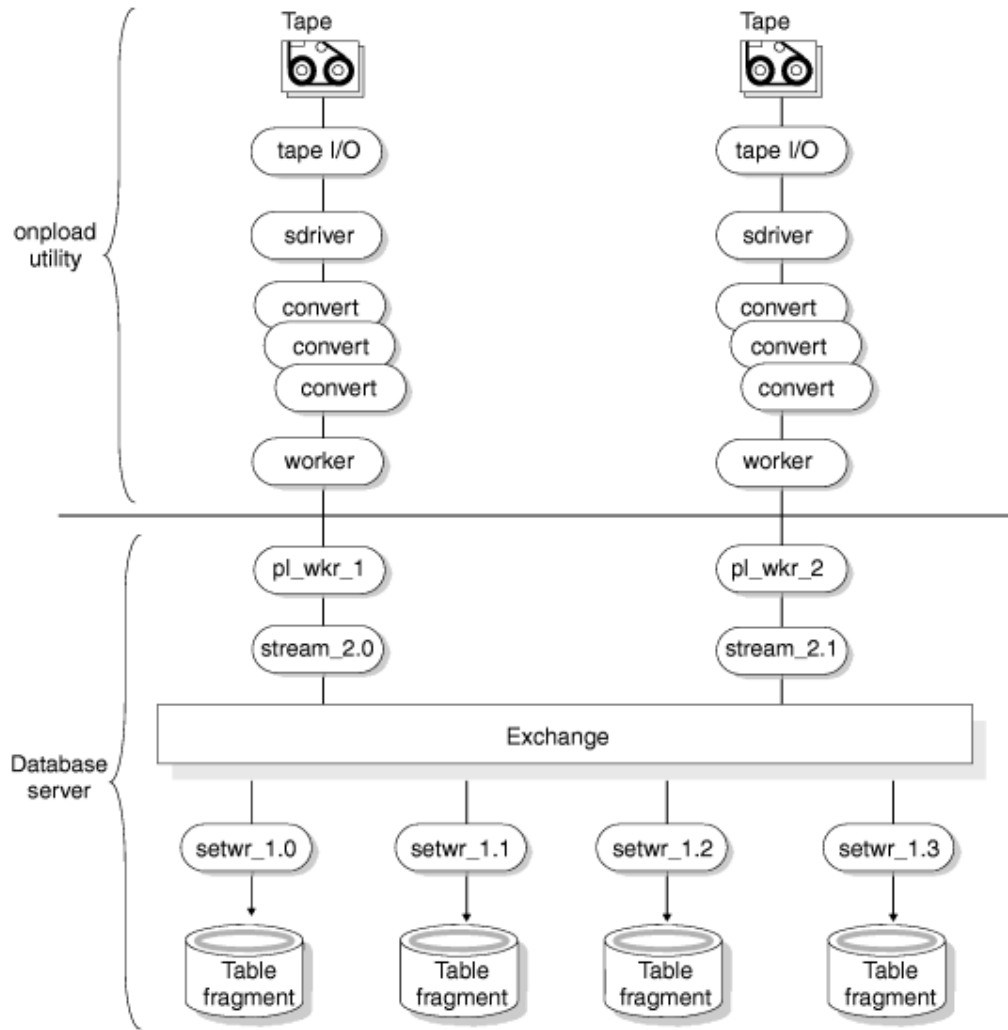
Important: After the express-mode load, you must perform a level-0 backup before you can write to the target database. If you try to write to the table before you perform a level-0 backup, the database server issues ISAM error -197, as follows:

```
Partition recently appended to; can't open for write or logging.
```

If your database is ANSI-compliant, all access (both read and write) is denied until you perform a level-0 backup. Because data is not logged in express mode, the level-0 backup is necessary to allow for recovery in case of media failure.

The following figure shows a single express-mode load process. In express mode, the data is inserted directly into an extent without any evaluation of objects such as constraints, indexes, or triggers.

Figure 5. An express-mode load

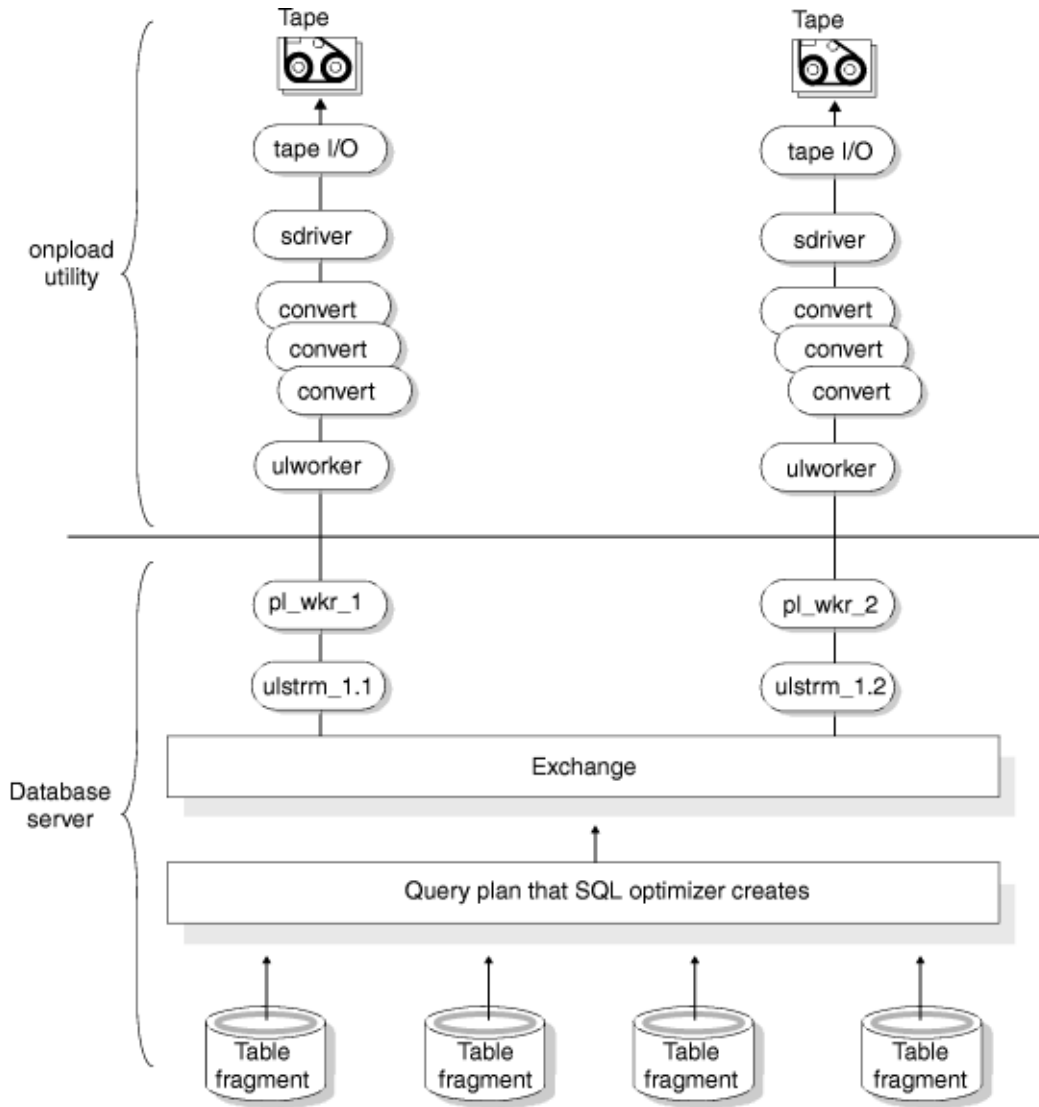
**Related information**

[The HPL express mode on page 178](#)

The onpload utility unload process

The following figure shows the onpload unload process. In the unload process, the behavior of onpload parallels the behavior described in [Threads that the onpload utility uses on page 14](#) and [Threads that the database server uses on page 15](#), except that the threads are unloading the data instead of loading it.

Figure 6. The unload procedure



The **ulstrm** (unload-stream) thread packages data for output to the onpload client from the query plan. The SQL optimizer creates the query plan. The query plan behaves like a query plan running from any other client, such as DB-Acess. The exchange operator distributes the resulting data to the **ulworker** threads in a round-robin fashion, and onpload unloads the data onto tapes or files.

Parallelism with respect to the output device, the source table fragments, and the flow of the data is evident in the preceding figure.

Examples of loading and unloading jobs using the ipload utility

This section shows how the components of the High-Performance Loader (HPL) fit together. The section includes a step-by-step tutorial with two examples (for a load and an unload job) that use the ipload graphical user interface (GUI).

The illustrations in the first example use a database with only one table. The table contains three columns. The data to be loaded into the database is in a file that has only four records. In a real production environment, you would probably use the INSERT statement, the dbimport utility, or the LOAD statement for such a simple operation. However, by using a simple example, the illustrations can show what happens at each step.

The onpladm command-line interface is equivalent to the ipload utility.

Related reference

[The onpladm utility on page 199](#)

Prepare to use the ipload utility

The ipload utility is a part of UNIX™ database servers. Before you can use ipload, you must have installed the database server. If the database that you want to load or unload is on the computer where you plan to run ipload, you must also start the database server.

You can also use ipload to manage load and unload operations for a database server on a different computer. For example, you can use ipload on a UNIX™ computer to manage the onpload database on a Windows™ computer.

Related reference

[Run load and unload jobs on a Windows computer](#)

Create a file of data

The illustrations in this section assume that the data to be loaded is in a file named `/work/mydata`. Create a file that contains the following data:

```
1|a|50
2|b|25
3|c|10
4|d|5
```

Create a database

The High-Performance Loader (HPL) loads data into an existing table in an existing database. The examples in this section load the information from the file `/work/mydata` into a three-column table named **tab1** in a database named **testdb**. You can use DB-Acess to prepare the database and table, as follows:

```
CREATE DATABASE testdb;
CREATE TABLE tab1
(
  col1 INTEGER,
  col2 CHAR(1),
  col3 INTEGER
);
GRANT ALL ON tab1 TO PUBLIC;
GRANT CONNECT TO PUBLIC;
```

After you finish preparing the database for the examples, exit from DB-Acess.

The ipload utility

The High-Performance Loader (HPL) uses information from the onpload database to control loading and unloading of data. Theoretically, you could create the onpload database and use DB-Acess or some other database tool to populate it. However, it is recommended that you always use ipload to manage the onpload database.

Related information

[The ipload utility on page 8](#)

Start the ipload utility

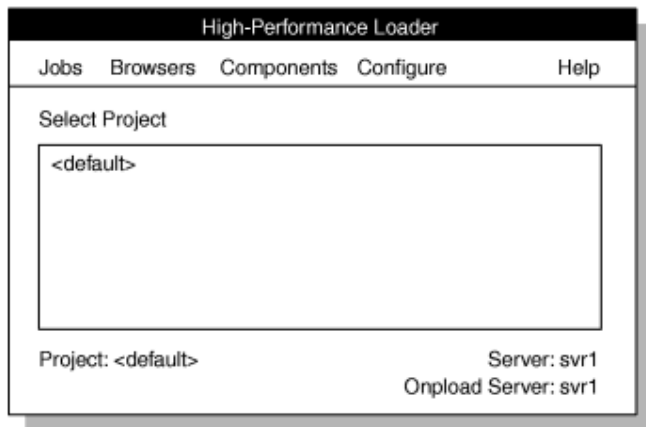
To start ipload, type the following at the system prompt `ipload`.

A decorative splash screen appears and stays on the display while ipload finishes loading. If you do not want to see the splash screen, use the `-n` flag, `ipload -n`.

The first time you start ipload, it automatically creates the onpload database. The ipload utility also puts certain default values into the database.

When ipload starts, the High-Performance Loader (HPL) main window appears, as the following figure shows.

Figure 7. The HPL main window



Tip: To exit from ipload, choose **Exit** from the **Jobs** menu.

The onpladm command-line utility provides the same functionality as the ipload graphical user interface.

Related reference

[The onpladm utility on page 199](#)

The onpload database

Choose a project

You use the High-Performance Loader (HPL) by preparing load jobs that import data or unload jobs that export data. You can assign the load and unload jobs to various projects to organize the jobs into functional groups.

The ipload utility automatically creates a project named **<default>**. If you choose not to organize your work into projects, you can put all of the load and unload jobs in the default project.

For this example, you can use the default project. Click **<default>** on the HPL main window to choose the default project.

Related information

[Define HPL projects on page 67](#)

Check the ipload utility default values

The default values that ipload selects when it is first started specify machine type, character code set for character-type data, and other operating characteristics. In most cases, the only default that you might need to change is the machine type.

Related reference

[Configure the High-Performance Loader on page 70](#)

Look at the Defaults window

Choose **Configure > Defaults** to see the current default values. After you check the defaults, you can click **Cancel** to exit from the **Defaults** window.

Related information

[Modify the onpload default values on page 72](#)

Look at the Machines window

Choose **Configure > Machines** to see the current default values. Make sure that the machine type displayed in the **Machines** window matches the current machine type. Click the **Machine > Type** down arrow to select a machine type. Click **Cancel** to exit from the **Machines** window.

Related information

[Modify the machine description on page 76](#)

Load Job windows

A load job is a collection of the specific pieces of information that you require to move data from a data file into a database. The **Load Job** window shows a flowchart that includes all of the components of a load job.

The **Load Job** window is illustrated in [Figure 9: The Load Job window on page 23](#). After you become familiar with `ipload`, you can create or modify the individual components of a load or unload job with direct menu choices from the HPL main window.

Load Job Select window

The data-load example takes records from `/work/mydata` and loads them into **tab1** of the **testdb** database. To perform the load, you need to create a load job.

About this task

To create a load job:

1. Choose **Jobs > Load** from the HPL window.

The **Load Job Select** window appears, as the following figure shows.

Figure 8. The Load Job Select window

Load Job Select

Delete Notes Connect

Selection Type

Open Create Job Name:

Command Line:

Write/read to/from tape until end of device.

Job Information

Job	Type	Status	Server	Map	Datasource
-----	------	--------	--------	-----	------------

Notes

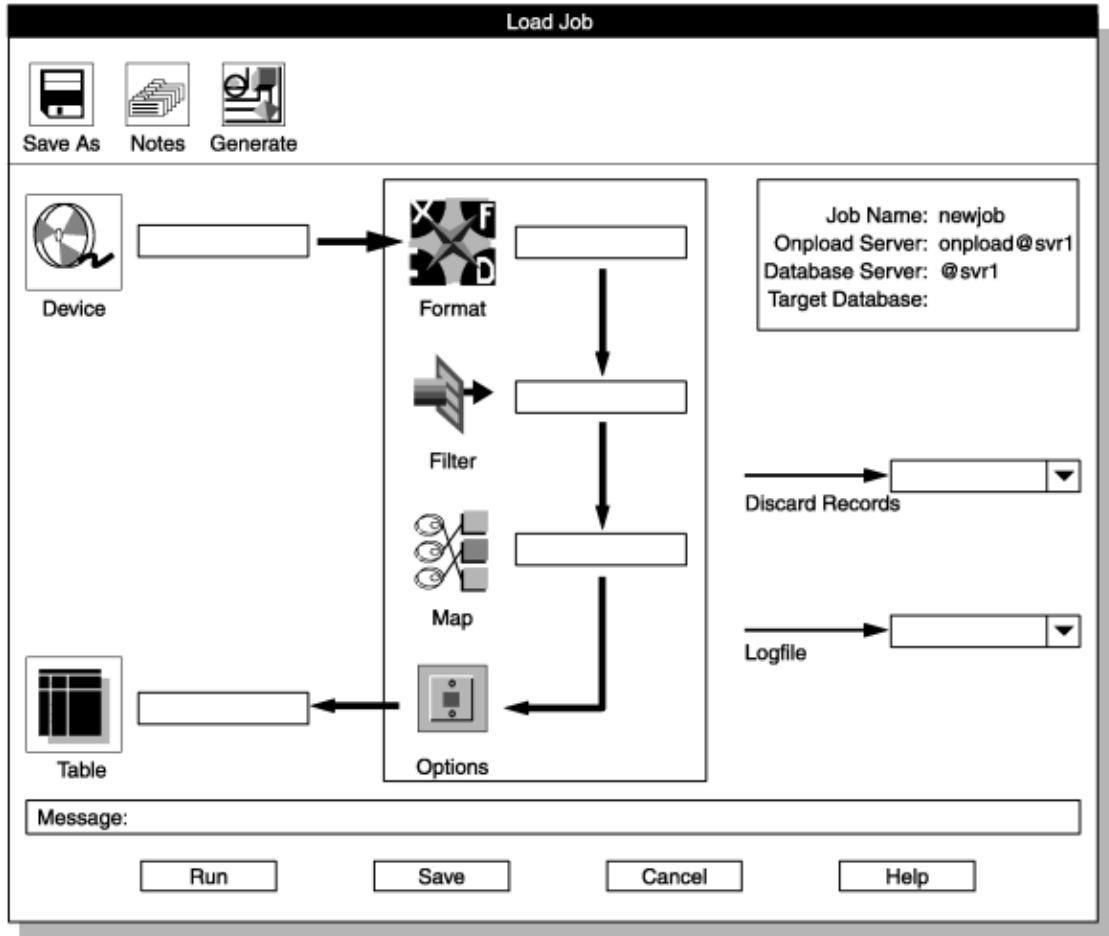
Message: Enter a job name to create

OK Cancel Help

2. Click **Create** in the **Selection Type** group.
3. Choose a name for the load job and type it in the **Job Name** text box.
This example uses **newjob**.
4. Click **OK**.

The **Load Job** window appears, as the following figure shows.

Figure 9. The Load Job window



Load Job window

The thick arrows on the **Load Job** window indicate the steps that you take as you build a load job. The icons indicate the task at each step. The thin arrows indicate file names for error recording. To create a load job, you need to complete the following tasks:

Task	Click
Specify the source of the data	Device
Describe the data	Format
Tell ipload which data you want to discard (optional)	Filter

Task	Click
Specify association between input fields and load-table columns	Map
Specify options for the load job	Options
Specify the database table to load	Table
Record rejected data records (optional)	Discard Records
Record information about the job (optional)	Logfile

Device-Array windows

A device array is a collection of files, tape devices, and pipes that onpload uses for input and output. Pipes are only supported on UNIX™. You can create a device array and specify the location of the input data from the **Device Array Selection** and **Device-Array Definition** windows.

In this example, the input data is the `/work/mydata` file that you created in [Create a file of data on page 19](#).

Device Array Selection window

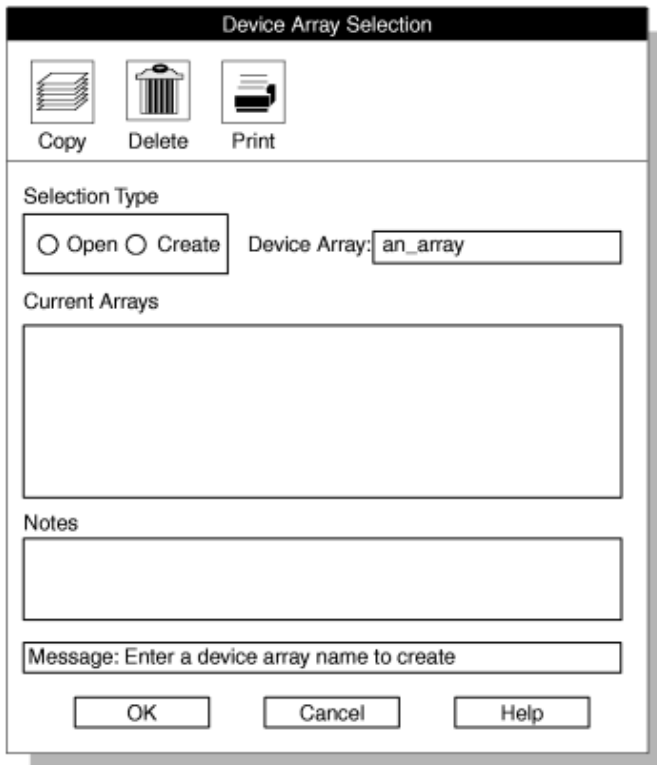
About this task

To create a device array:

1. Click **Device** in the **Load Job** window.

The **Device Array Selection** window appears, as the following figure shows.

Figure 10. The Device Array Selection window



2. Click **Create** in the **Selection Type** group.
3. Select a name for the device array and type it in the **Device Array** text box.
This example uses **an_array**.
4. Click **OK**.

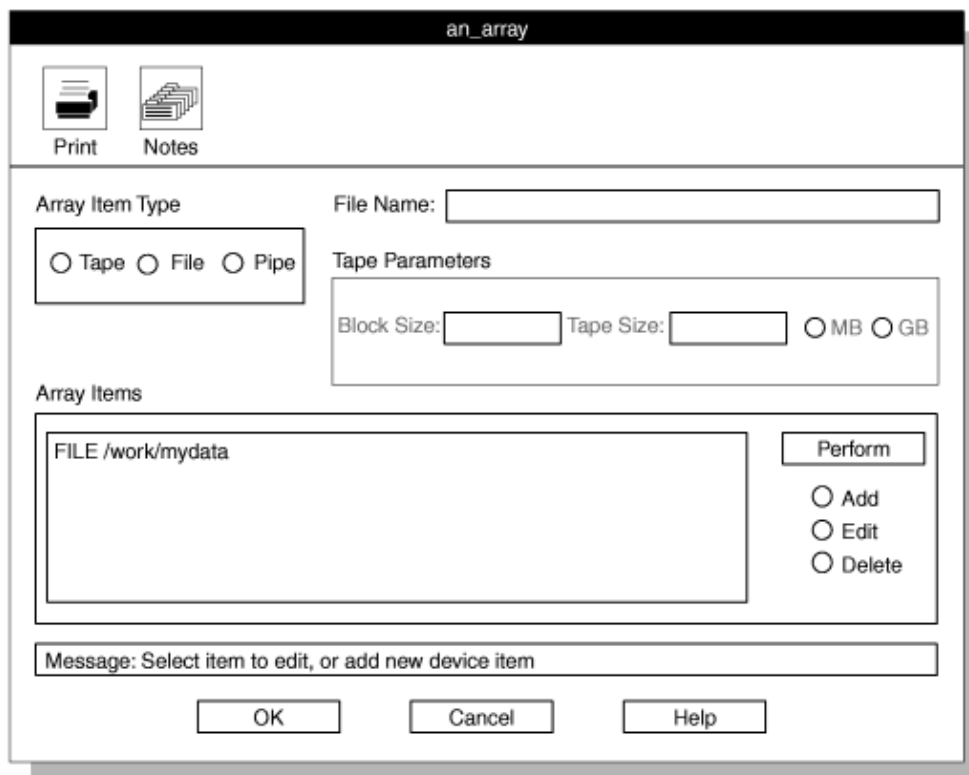
The **Device-Array Definition** window appears, as [Figure 11: The Device-Array Definition window with one array item on page 26](#) shows.

Device-Array Definition window

About this task

The title bar of the **Device-Array Definition** window shows the array name that you typed in the **Device Array** text box.

Figure 11. The Device-Array Definition window with one array item



To add a device to the array:

1. Click **Add** in the **Perform** group (lower right).
2. Click **File** in the **Array Item Type** group (upper left).
3. Type the full path name of a device in the **File Name** text box.
In this example, the device is `/work/mydata`.
4. Click **Perform** to add the `/work/mydata` file to the device array.

The `ipload` utility lists each item of the device array in the **Array Items** list box. [Figure 11: The Device-Array Definition window with one array item on page 26](#) shows the window after you add `/work/mydata` to the array.

5. Click **OK**.

The display returns to the Load Job window. The **Device** list box now displays the device array name that you chose.

Format windows

The format specifies the organization of the input data.

In this example, the input data is in the file `/work/mydata`, which you created in [Create a file of data on page 19](#). Each record in the file has three fields.

Format Views window

The **Format Views** window displays existing formats, so that you can choose the format to use in the load job.

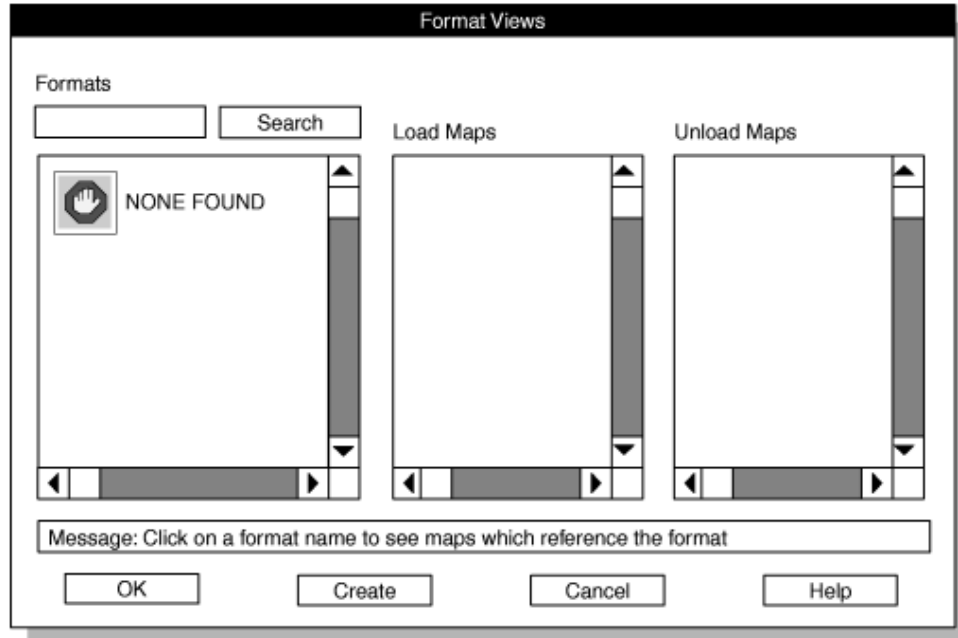
About this task

To open the **Format Views** window:

1. Click the **Format** in the **Load Job** window.

The **Format Views** window appears, as the following figure shows. When you first start ipload, no formats are defined, as the NONE FOUND icon illustrates.

Figure 12. The Format Views window



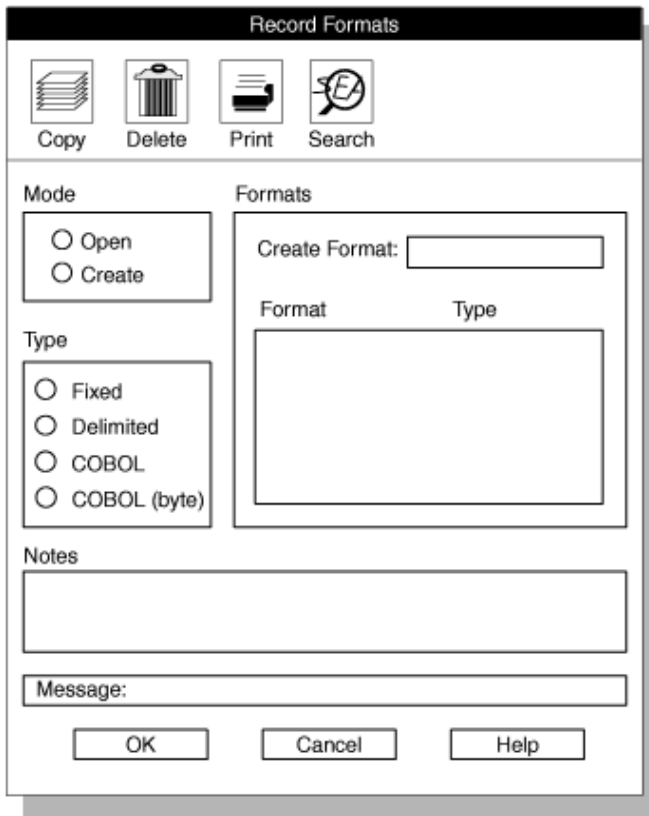
2. Click **Create** to open the **Record Formats** window.

Record Formats window

About this task

The following figure shows the **Record Formats** window. You can create a format or open an existing format from the **Record Formats** window.

Figure 13. The Record Formats window



To create a format:

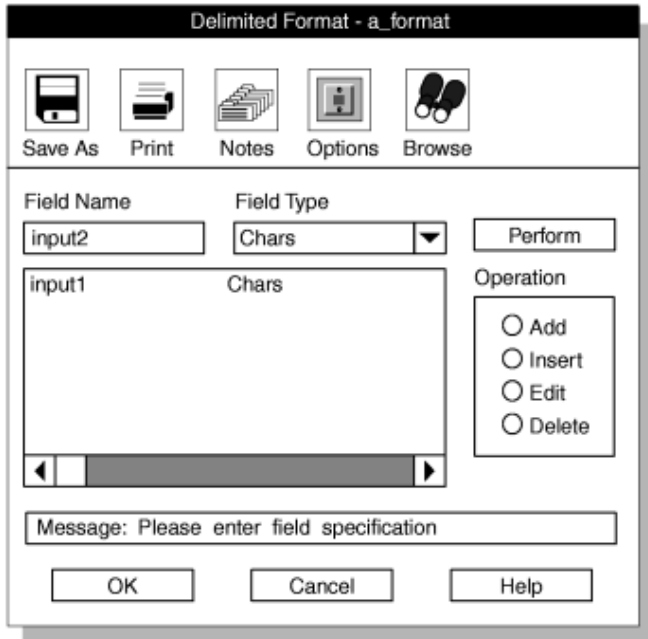
1. Click **Create** in the **Mode** group.
2. Click **Delimited** in the **Type** group.

The input data file, `/work/mydata`, is in delimited format. Other formats are described in [Define formats on page 84](#).

3. Choose a name for the format and type it in the **Create Format** text box.
This example uses the name **a_format**.
4. Click **OK**.

The **Format-Definition** window appears. The following figure shows a partially completed **Format-Definition** window. The title bar of the **Format-Definition** window shows the name that you chose for the new format.

Figure 14. The Format-Definition window



Format-Definition window

In the **Format-Definition** window, you must make an entry for each field of the data records in the input file. The input file for this example (`/work/mydata`) has three fields of data in each record, so you must enter format information for three pieces of data.

About this task

To enter a format definition:

1. Click **Add** in the **Operation** group.
2. In the **Field Name** text box, type a descriptive name for the first field of the data record.

You can choose any descriptive name. This example uses **input1**, **input2**, and **input3** for the three fields of `/work/mydata`.

3. In the **Field Type** text box, type the data type or click the down arrow for a list of selections.

Because the data in `/work/mydata` is simple ASCII data, the type is **Chars**. Other data types are discussed in [Define formats on page 84](#).

4. Click **Perform**.

[Figure 14: The Format-Definition window on page 29](#) shows the partially completed **Format-Definition** window.

The entry for the first item is complete. The **Field Name** and **Field Type** for the second item are present and ready for you to click **Perform**.

5. Repeat steps 2 on page 29 through 4 on page 29 for each of the three input fields.
6. Click **OK** after you complete all of the input fields.

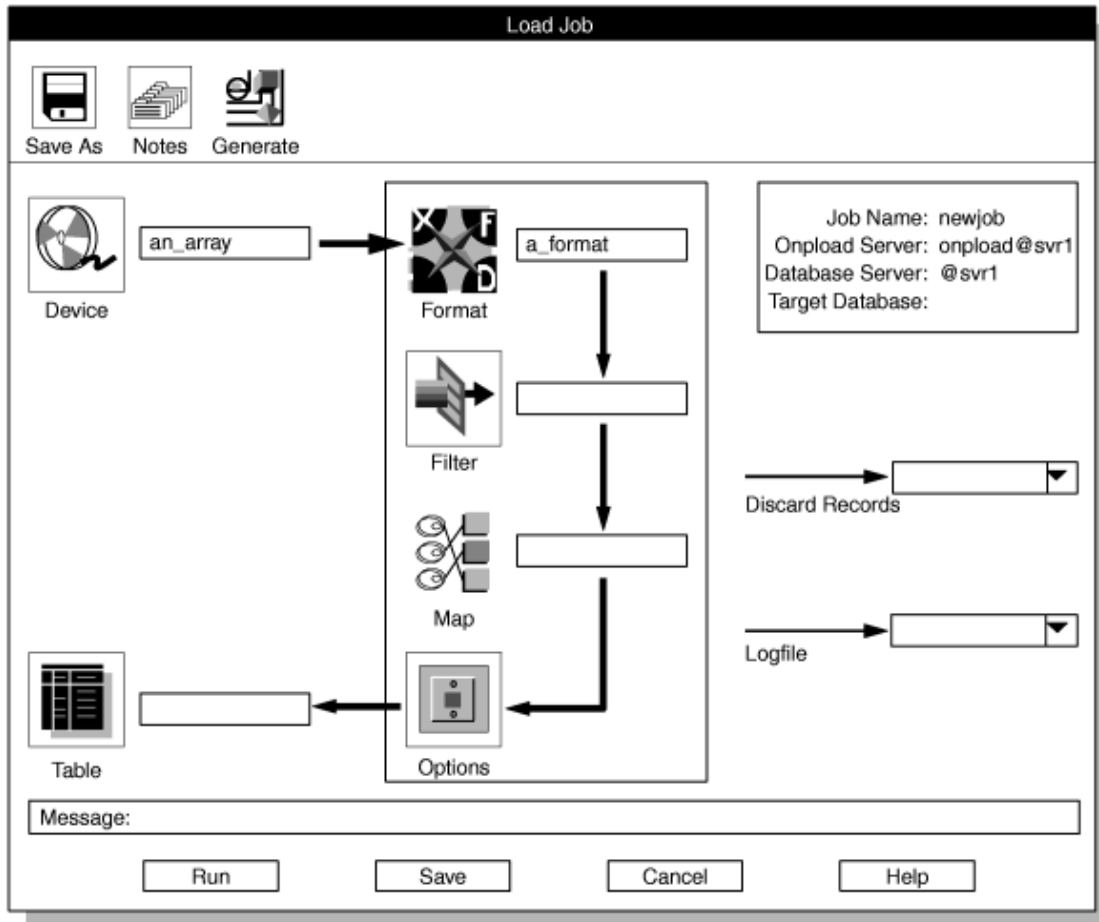
The display returns to the **Format-Views** window, which displays the new format in the **Formats** list box.

7. Click **Cancel** to return to the **Load Job** window.

Results

The **Load Job** window now displays the name of the device and the name of the format, as the following figure shows.

Figure 15. Partially completed Load Job window



Filter, Discard Records, and Logfile text boxes

The **Load Job** window now has entries for a device and format. The next incomplete items in the Load Job window are the **Filter** text box, the **Discard Records** text box, and the **Logfile** text box.

Filter text box

Use a filter to choose the records from the data file that should be inserted into the table. In this example, all of the records from the `/work/mydata` data file are inserted into the database table. Therefore, you do not need to create a filter. For this example, you can leave the **Filter** text box blank.

[Define filters on page 135](#), describes how to create and use a filter.

Discard Records text box

The **Discard Records** text box specifies a file that keeps information about records that were rejected because of incorrect format or invalid data. For this example, you can leave the **Discard Records** text box blank.

[Reviewing records that the conversion rejected on page 173](#) describes how to view rejected records.

Logfile text box

The **Logfile** text box specifies a file where a record of the load or unload job is kept. For this example, you can leave the **Logfile** text box blank.

[View the status of a load job or unload job on page 175](#) describes how to view the log file.

Map Views window

From the **Map Views** window, you can create a map that specifies which data field from the input file (`/work/mydata`) is entered into which columns in the database table.

Creating a map by using Map Views window

Use the **Map Views** window to create a map or edit an existing map. You need to create a map that shows how the input data that the record format **a_format** describes is to be loaded into the table **tab1**.

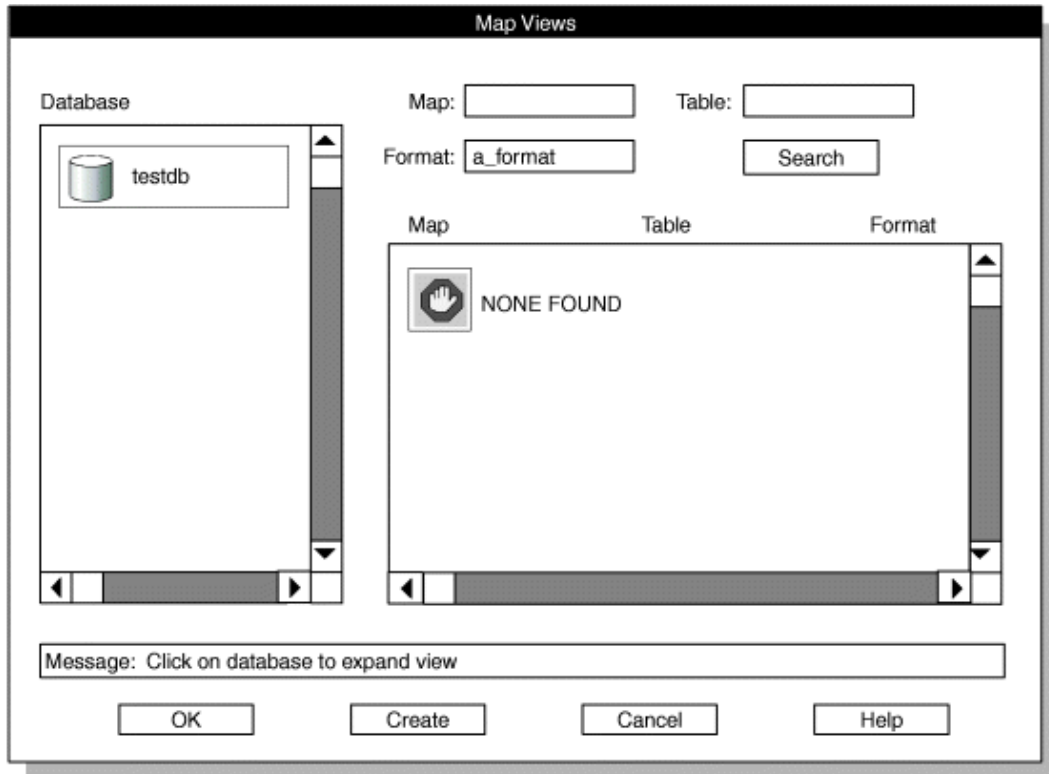
About this task

To create a map:

1. Click the **Map** button in the **Load Job** window.

The **Map Views** window appears, as the following figure shows.

Figure 16. The Map Views window

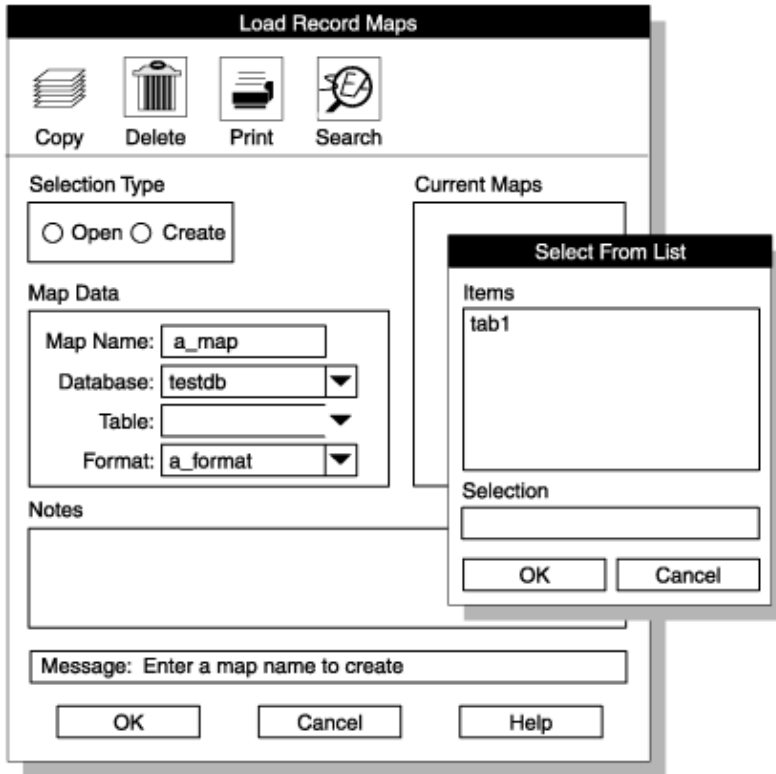


The NONE FOUND icon in the Map column is appropriate. The **Map Views** window does not contain any information because you have not yet specified any relationships.

2. Click **Create**.

The **Load Record Maps** window appears, as the following figure shows.

Figure 17. A partially completed Load Record Maps window with an open selection list



Completing the Load Record Maps window

The **Load Record Maps** window specifies the device array that holds the input data, the format that describes the input data, and the database and table where the input data will be stored.

About this task

To complete the **Load Record Maps** window:

1. Click **Create** in the **Selection Type** group.
2. Select a name for the map and type it in the **Map Name** text box.
This example uses **a_map**.
3. Type the name of your database (`testdb`) in the **Database** text box. Or, click the down arrow to select a database from the selection list.
4. Click the down arrow beside the **Table** text box to see a list of tables in the selected database.

Figure 17: A partially completed Load Record Maps window with an open selection list on page 33 shows the **Load Record Maps** window and the selection list.

5. Select a table from the list and click **OK**.

Because you already filled in the **Format** text box on the **Load Jobs** window, the **Format** text box is already complete.

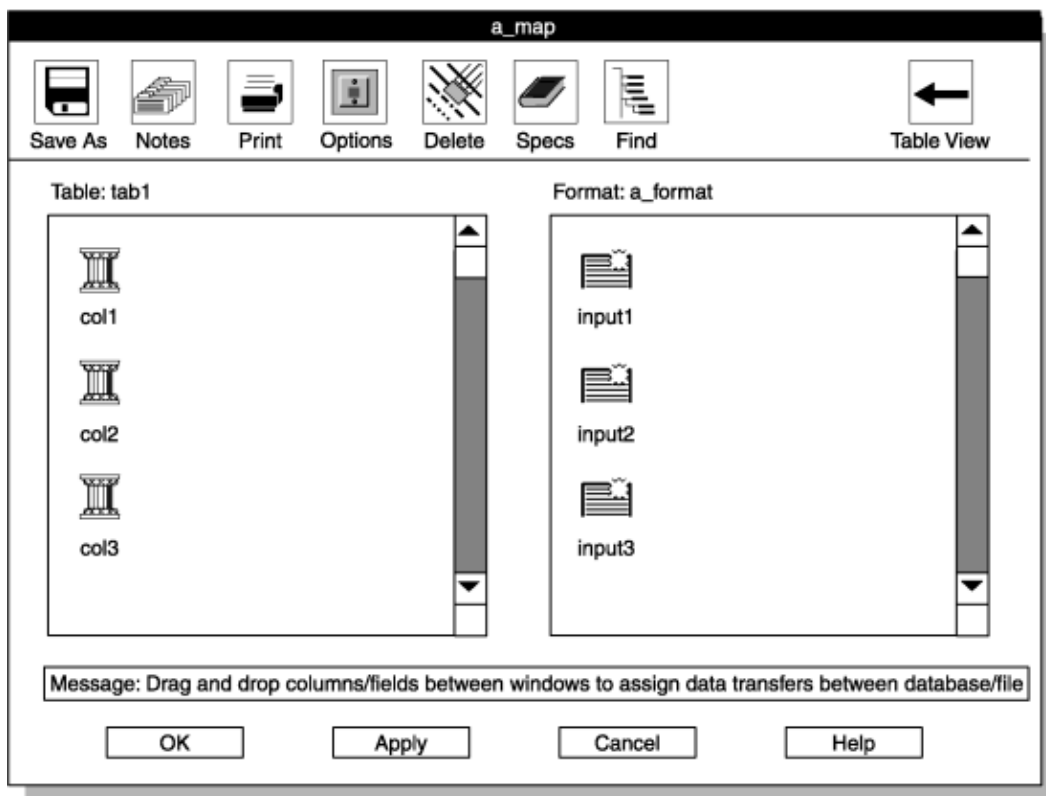
6. Click **OK** to open the **Map-Definition** window.

Map-Definition window

You can associate an input item with a table column in the **Map-Definition** window. This example stores the data from / work/mydata as follows, by using the field names assigned in step 1 on page 28.

Data from input field	Goes into table column
input1	col3
input2	col2
input3	col1

The following figure shows the **Map-Definition** window. The title bar of this window shows the map name that you chose. Figure 18. The Map-Definition window



Associating each input item with a column of the database table

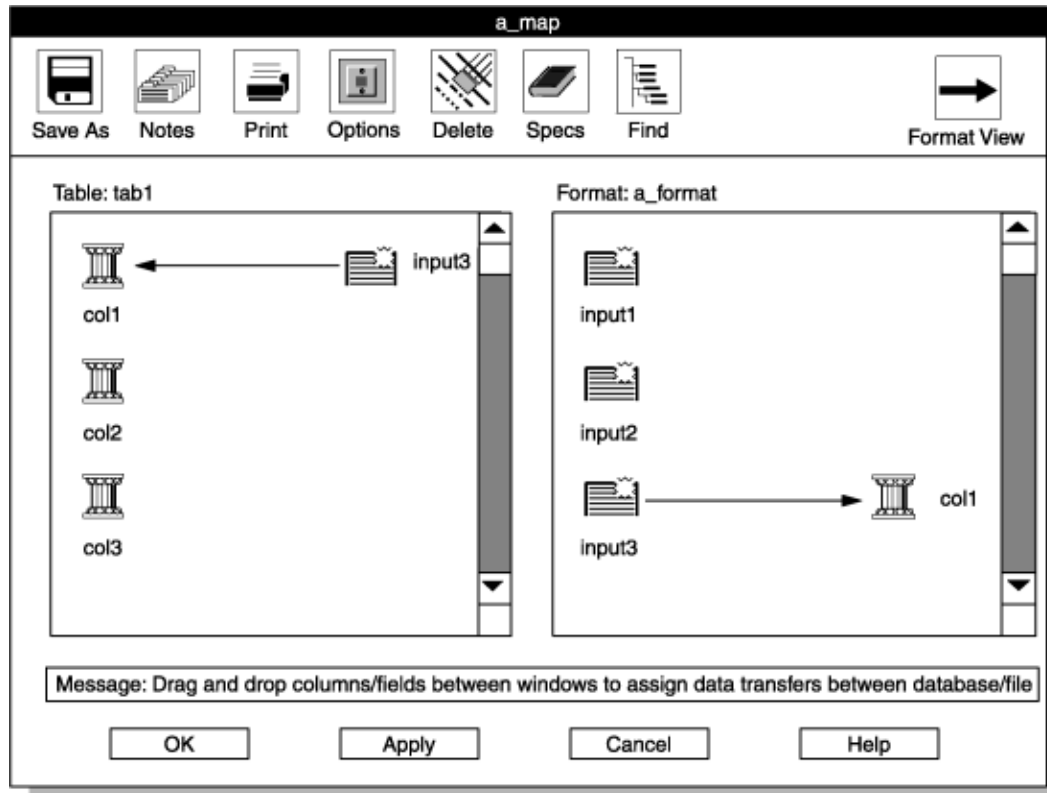
About this task

To associate each input item with a column of the database table:

1. Click the **col1** icon and hold the mouse button down.
A box appears around the icon and its name.
2. Drag the boxed icon to the **input3** icon in the right pane.
3. Release the mouse button.

The associated items appear in the second column of each pane. The following figure shows the **Map-Definition** window with this step completed.

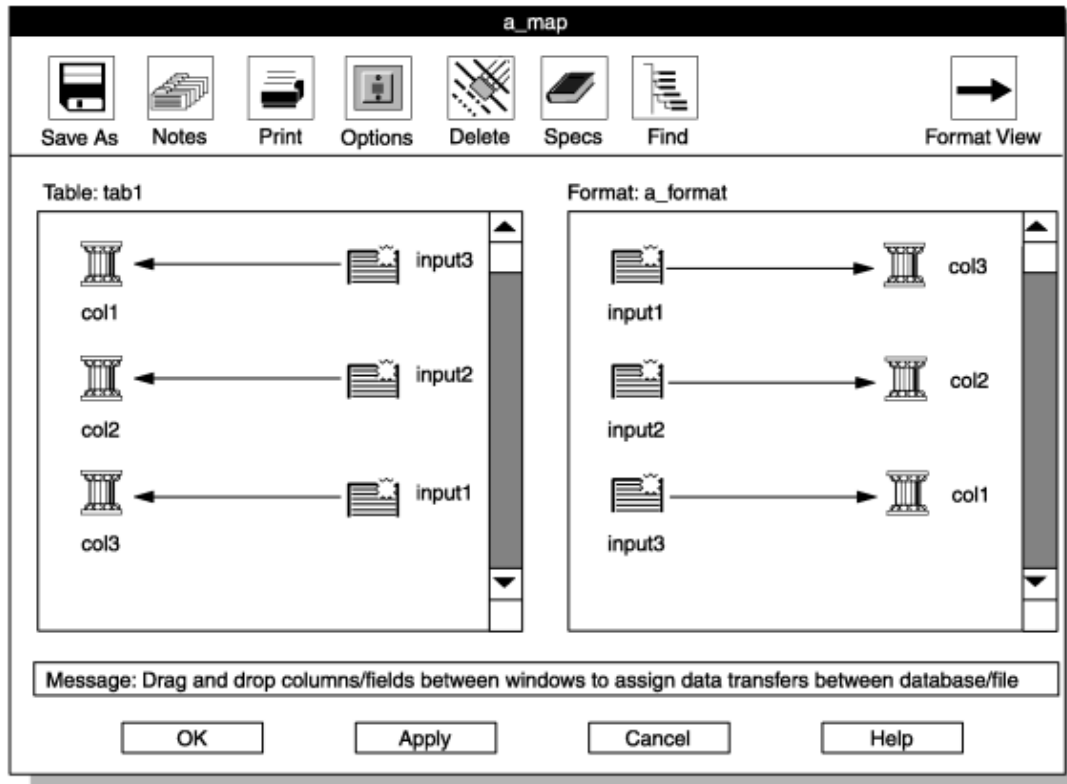
Figure 19. The Map-Definition window with one association completed



4. Connect **col2** to **input2**.
5. Connect **col3** to **input1**.

The following figure shows the window with all three connections completed.

Figure 20. The Map-Definition window with all associations completed

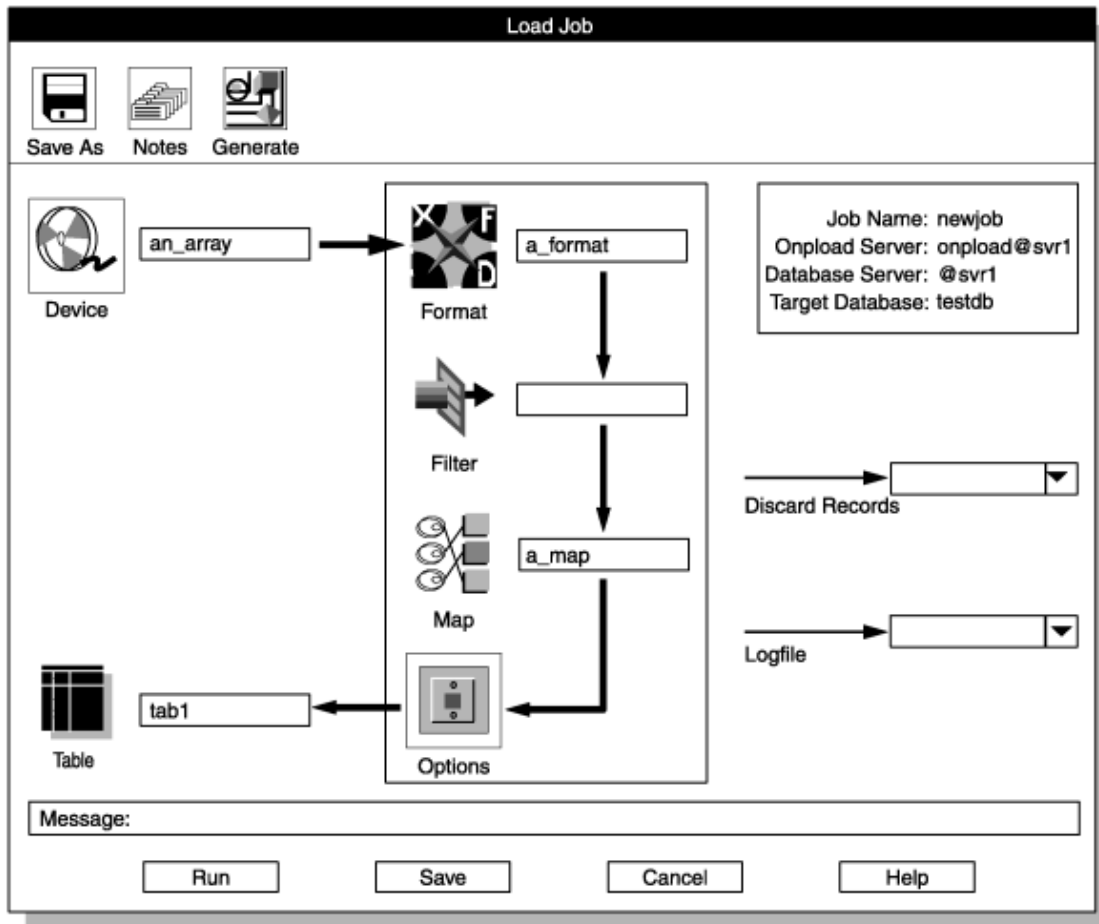


6. Click **OK** to return to the **Map Views** window.
7. Click **Cancel** to return to the **Load Job** window.

Results

The **Load Job** window now has entries in all of the required areas, as the following figure shows. The **ipload** utility was able to enter the table name and the target database name (upper right area) because you specified the database and table as you built the map.

Figure 21. The Load Job window with all required component boxes completed



You have finished all of the required parts of the **Load Job** window, but you might want to modify the options, as discussed in the next section.

Load Options window

About this task

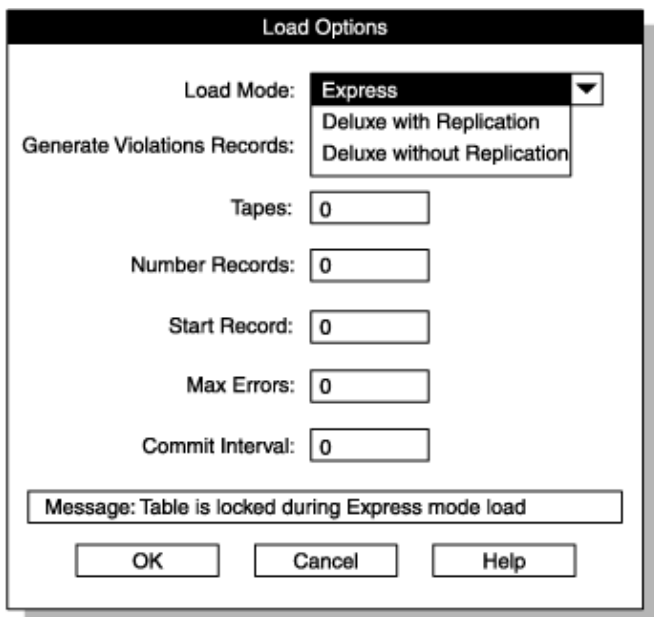
The High-Performance Loader (HPL) has three modes of operation: express, deluxe with replication, and deluxe without replication. The express mode is optimized for speed and the deluxe mode provides the full functionality of SQL inserts as data is loaded. For a detailed comparison of these modes, see [Manage the High-Performance Loader on page 177](#). This example uses the express mode.

To set the load-job options:

1. Click **Options** in the **Load Job** window.

The **Load Options** window appears, as the following figure shows.

Figure 22. The Load Options window



2. Select **Express** from the **Load Mode** list box.
3. Select **Yes** from the **Generate Violations Records** list box.
4. Make sure all of the other entries are 0.
5. Click **OK** to return to the **Load Job** window.

Running the iupload job

You are now ready to perform the load.

About this task

To finish this example:

1. Click **Save** to save the load job.

After you save the load job, the message line displays the following message:

```
Saved job successfully
```

You can now run the job, or you can return at a later time and run the job.

2. Click **Run** to run the load job.

The Active Job window appears, as [Figure 23: The Active Job window on page 39](#) shows.

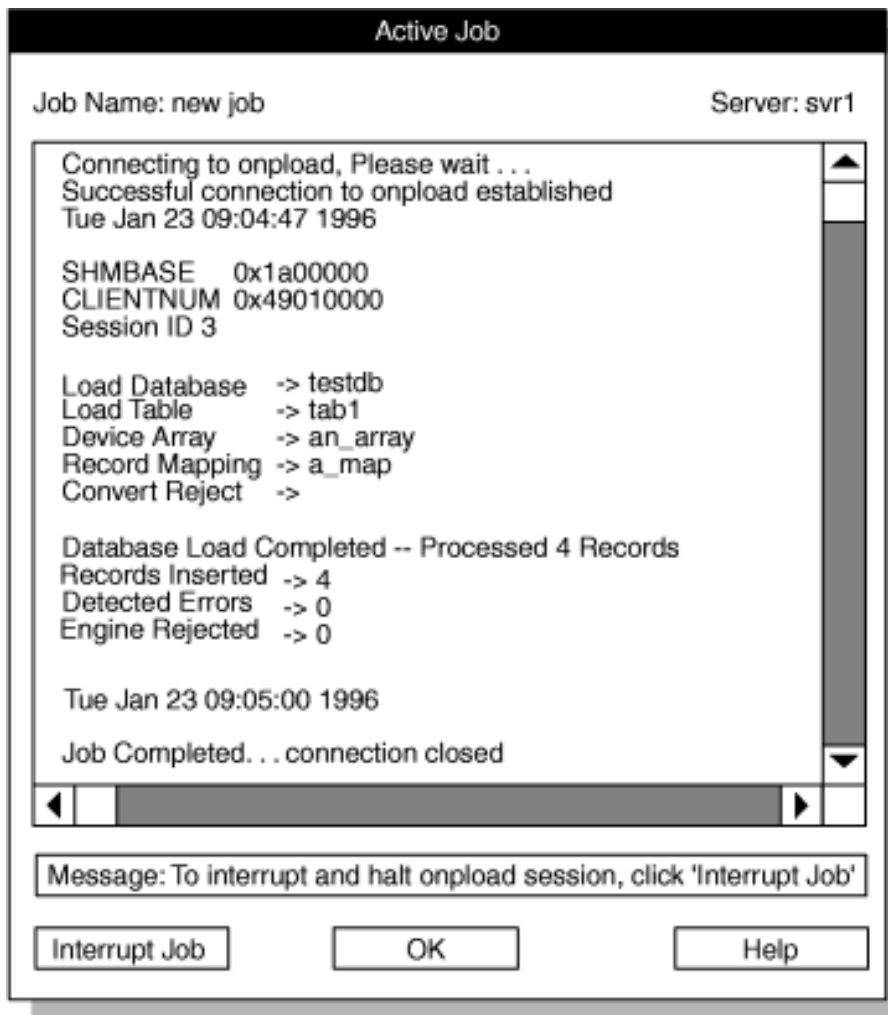
Results

i Tip: The `ipload` utility generates an `onpload` command and then runs the command to run your job. To see the command that `ipload` generates, look at the **Command Line** text box on the **Load Job Select** window. For more information, see [The command-line information on page 157](#).

Active Job window

The **Active Job** window reports the progress of your job. The following figure shows the **Active Job** window after the load is complete.

Figure 23. The Active Job window



When the **Active Job** window reports that the load job is complete, click **OK** to return to the **Load Job Select** window.

Verify the transfer of data

You can use DB-Acess to verify that the data from `/work/mydata` was transferred into your database.

Perform a level-0 backup

The onpload utility does not log the data that it writes to a table during an express-mode load. For safety, onpload flags the dbspaces that are associated with the table as read-only. To allow for data recovery in case of disk corruption, you must make a level-0 backup. A level-0 backup on the dbspaces affected by the express-mode load saves the data and unsets the read-only flags.

If you do not care about data recovery, you can make a level-0 backup by using `/dev/null` as the backup device. This action unsets the read-only flag without backing up data to any real device.

The ipload utility Generate options

The ipload utility has Generate options that you can use to automatically create a format, map, query, and device. After the components are generated, you can modify the components to meet your needs.

This example uses the **Generate** button in the **Unload Job** window to create the components that are required for an unload job. After you create the components, you can use the **Run** option to run the unload job.

Related reference

[The Generate options of the ipload utility on page 160](#)

Use the information you created with the ipload example

If you completed the first example, your database server and ipload are ready for you to use. If you did not complete the example, you need to complete the following tasks as the first example describes:

- Start your database server ([Prepare to use the ipload utility on page 19](#))
- Start ipload ([Start the ipload utility on page 20](#))
- Check the defaults ([Check the ipload utility default values on page 21](#))

Preparing the Unload Job window

An unload job is a collection of the specific pieces of information that are required to move data from a database into a data file. The **Unload Job** window shows a flowchart that includes all of the components of an unload job. You can use the **Generate** option to create the components of the unload job and to complete the items on the **Unload Job** window.

About this task

The generate example uses the **Generate** option to unload the contents of the **items** table of the **stores_demo** database into a file named `/work/items_out`. For instructions on how to create the **stores_demo** database and other demonstration databases, see the *HCL® Informix® DB-Access User's Guide*.

To generate the unload job:

1. Choose **Jobs > Unload** from the HPL window.

The **Unload Job Select** window appears, as the following figure shows.

Figure 24. The Unload Job Select window

Unload Job Select

Delete Notes Connect

Selection Type

Open Create

Job Name: unld

Command Line:

Write/read to/from tape until end of device.

Job Information

Job	Type	Status	Server	Map	Datasource

Notes

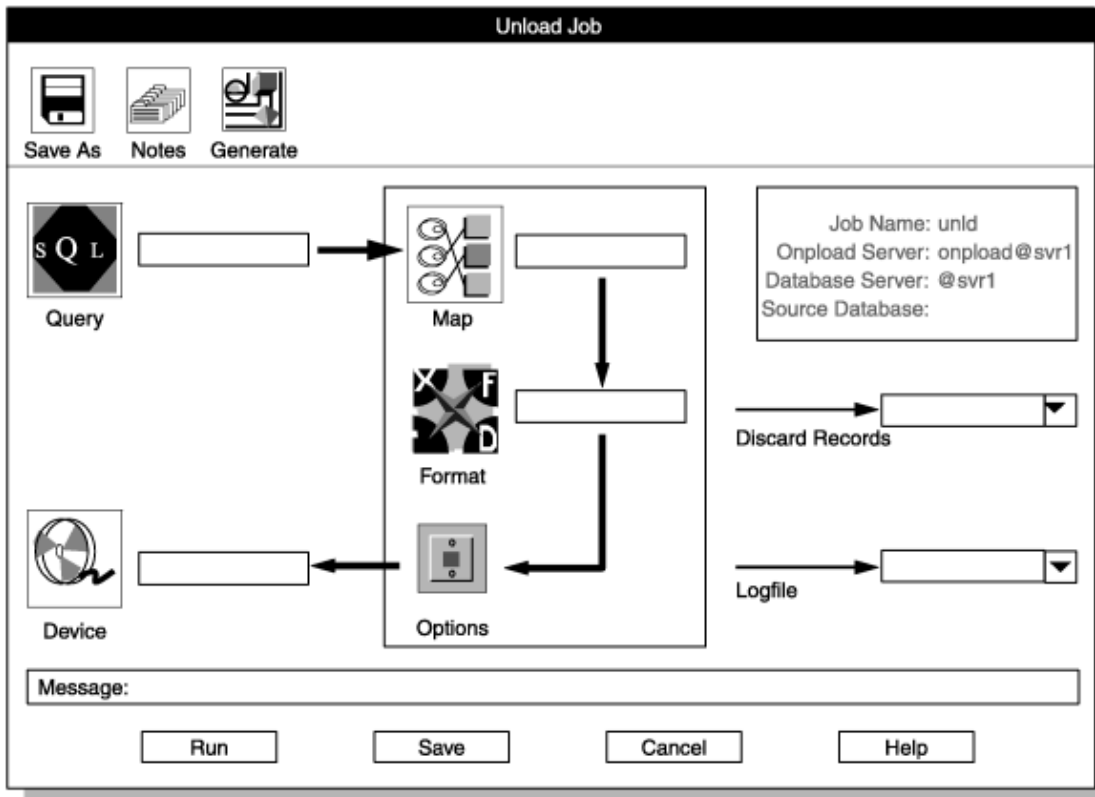
Message: Enter a job name to create

OK Cancel Help

2. Click **Create** in the **Selection Type** group.
3. Choose a name for the unload job and type it in the **Job Name** text box.
This example uses the name **unld**.
4. Click **OK**.

The **Unload Job** window appears, as the following figure shows. The information box in the upper right part of the display shows the name of the unload job, the name of the database server where the onpload database is stored, and the name of the database server where ipload is running.

Figure 25. The Unload Job window



5. Click the **Generate** button.

The **Autogenerate Unload Components** window appears. [Figure 26: The Autogenerate Unload Components window on page 43](#) shows the completed window.

6. Click **Table**.

You can unload an entire database table or only selected records from the table. **Table** indicates that you want to unload the entire table. **Query** indicates that you want to unload selected records.

7. Type `stores_demo` in the **Database** text box.

For this step and steps [8 on page 42](#) and [10 on page 43](#), you can click the down arrow to the right of the text box and select the entry from a selection list. [Figure 17: A partially completed Load Record Maps window with an open selection list on page 33](#) shows an example of a selection list.

8. Type `items` in the **Table** text box.

Figure 26. The Autogenerate Unload Components window

9. Click **File**.

File indicates that you want to type the name of a file. If you choose **Device Array**, you must type the name of an existing device array.

10. Type the full path name of the file that will store the unloaded data. This file can be in any directory to which you have write access.
11. Click **OK**.

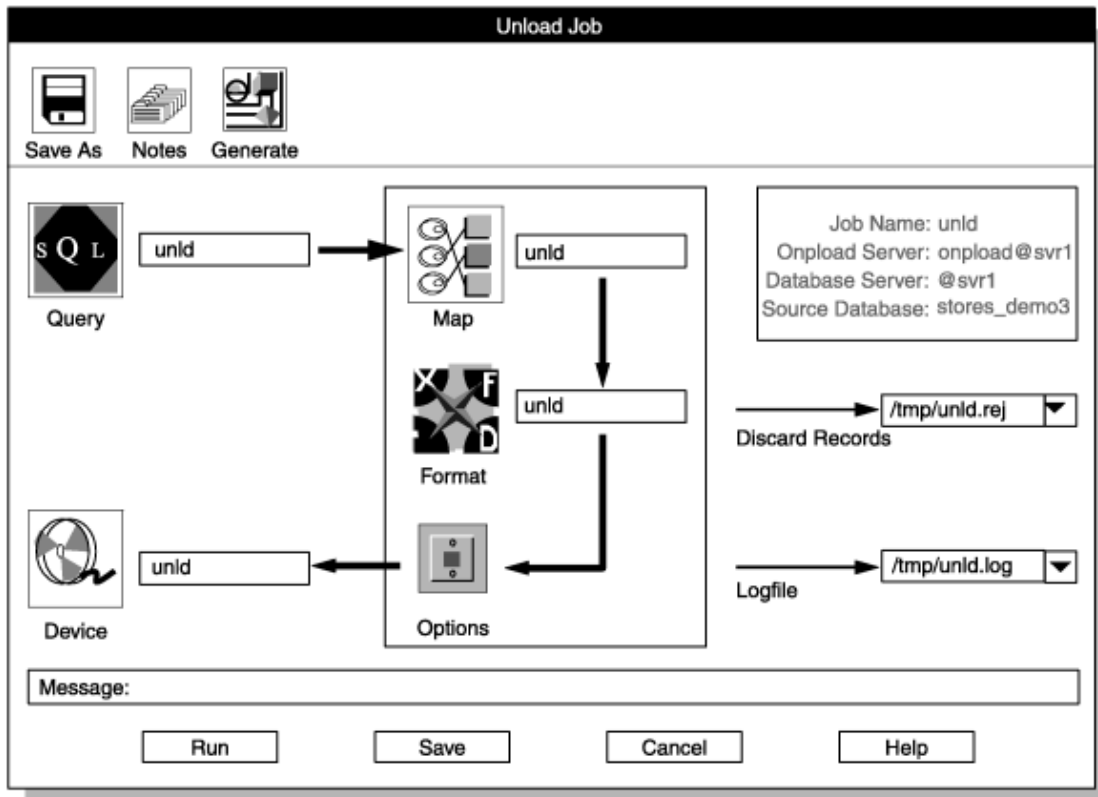
Results

The **Generate** option creates the Query, Format, and Map components for the unload job and completes the **Unload Job** window. These components are all named **unld**. The **Generate** option also creates a device array named **unld** and puts the file that you specified (`/work/items_out`) into that array.

Tip: After you finish this exercise, you can choose **Components > Devices** from the HPL window and examine the **unld** device array.

The following figure shows the **Unload Job** window as completed by the **Generate** option.

Figure 27. The Unload Job window



In addition to completing the main flow of the Unload Job window, the **Generate** option also enters the Source Database information in the upper right corner and creates path names for the Discard Records file and the Logfile. [The HPL browsing options on page 170](#), describes the rejected records file and the log file.

Performing the unload job

You are now ready to perform the unload job.

About this task

i Tip: You might want to preview the records that the query will select. See [Preview data-file records on page 170](#).

To finish this example:

1. Click **Save** to save the unload job.

After you save the unload job, the Message line displays the following message:

Saved job successfully

You can now run the job, or you can return at a later time and run the job.

2. Click **Run** to run the unload job.

The **Active Job** window appears. This window reports the progress of your unload job. The **Active Job** window for an unload job is similar to the **Active Job** window for a load job, which [Figure 23: The Active Job window on page 39](#) shows.

3. When the **Active Job** window reports that the unload job is finished, click **OK** to return to the **Unload Job** window.
4. Click **Cancel** to return to the HPL main window.
5. You can create another job or choose **Jobs > Exit** to exit iupload.

The iupload utility windows

This section contains information about using the iupload utility.

The iupload utility GUI or the onpladm command-line interface

The iupload utility is a graphical user interface (GUI) that contains windows, buttons, online help, and keyboard commands. The onpladm command-line interface is equivalent to the iupload utility.

Related reference

[The onpladm utility on page 199](#)

Start the iupload utility

To start iupload, type `iupload` at the system prompt.

A splash screen appears and stays on the display while iupload finishes loading. If you do not want to see the splash screen, use the `-n` flag with your command, `iupload -n`.

Related information

[The iupload utility on page 8](#)

The iupload utility GUI

The iupload utility has the following types of displays:

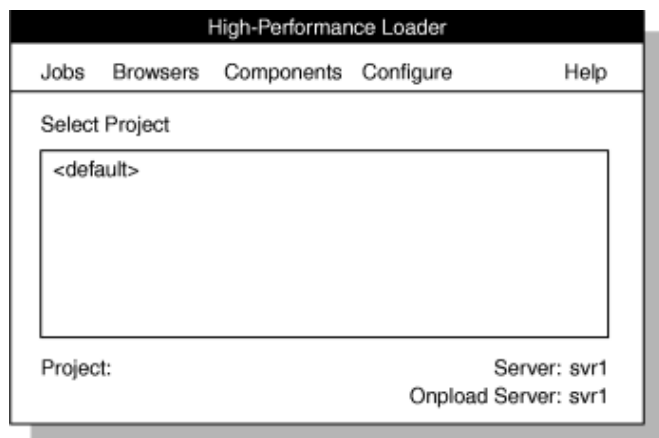
- HPL main window
- Component-Selection windows
- Component-Definition windows
- Load Job and Unload Job windows
- Views windows
- Selection-List windows
- Message windows

! **Important:** While the onpload and onpladm utilities include support for object names that contain up to 128 characters, the ipload utility does not. If you use long database, table or column names and create jobs by using onpladm, you cannot run these jobs by using ipload. For ipload, database, table, and column names cannot exceed 18 characters.

The HPL main window

When you start ipload, the High-Performance Loader (HPL) main window appears. The HPL main window is the focus of the user interface. You return to the main window after each task and choose a new option.

Figure 28. The HPL main window



Initial options on the HPL main window

When you first enter ipload, you can:

- Select the default project in the Select Project list.
- Choose **Configure > Project** to create a project.
- Choose **Configure > Server** to select a database server and an onpload database server.
- Choose **Help** to look at the online help.
- Choose **Jobs > Exit** to exit from ipload.

Related information

[Defining a new project on page 69](#)

[Select a database server on page 71](#)

Options of the HPL main window

After you select a project, you can choose options from any of the menus on the HPL main window.

The following table shows the options of the main menu:

Table 3. HPL main menu and submenu options

Main menu option	Submenu option	Purpose	See
Jobs	Load	Create a load job and use the Load Job window to load data into a database.	Load data to a database table on page 150
Jobs	Unload	Create an unload job and use the Unload Job window to unload data from a database to a file.	Creating an unload job on page 144
Jobs	Exit	Exit from the user interface.	The ipload utility on page 20
Browsers	Record	Review records in a specified format, search the list of available formats, or edit a format.	Reviewing records that the conversion rejected on page 173
Browsers	Violations	View records that passed the filter and conversion but were rejected by the database.	Viewing the violations table on page 173
Browsers	Logfile	View load status and see where any errors occurred.	View the status of a load job or unload job on page 175
Components	Formats	Create or modify data-file formats.	Define formats on page 84
Components	Maps	Create or modify maps that show the relationship between data-file fields and database columns.	Define maps on page 119
Components	Query	Build, modify, or retrieve SQL-based queries.	HPL queries on page 108
Components	Filter	Create or modify filters that determine source data-file records for conversion and load.	Define filters on page 135
Components	Devices	Specify a set of files, tapes, or pipes (UNIX™ only) that will be read simultaneously for loading or unloading the database.	Define device arrays on page 79

Table 3. HPL main menu and submenu options

(continued)

Main menu option	Submenu option	Purpose	See
Components	Generate Job	Automatically generate the components for load and unload jobs.	The Generate options of the ipload utility on page 160
Configure	Server	Select the database servers that hold the onpload database and the target database.	Configure the High-Performance Loader on page 70
Configure	Project	Create a project under which formats, filters, queries, maps, and load and unload jobs are stored.	Define HPL projects on page 67
Configure	Defaults	Specify the default character sets for the data file and databases.	Modify the onpload default values on page 72
Configure	Machines	Specify the machine parameters that are used to convert binary data.	Modify the machine description on page 76
Help	Glossary	View definitions of terms that pertain to the HPL.	The HPL online help on page 66
Help	Contents	View the main contents page that directs you to discussions of various HPL topics.	The HPL online help on page 66

Component-Selection windows

The windows for creating or modifying components often (but not always) come in pairs. In the first window, the **Component-Selection** window, you can create a component or select an existing component to modify. You can also view notes and copy, delete, or print information about a component. In the second window, the **Component-Definition** window, you can make changes.

The details of a selection window vary depending on the operation that you are performing.

The following figure shows the **Device Array Selection** window to illustrate the standard features of **Component-Selection** windows.

Figure 29. The Device Array Selection window

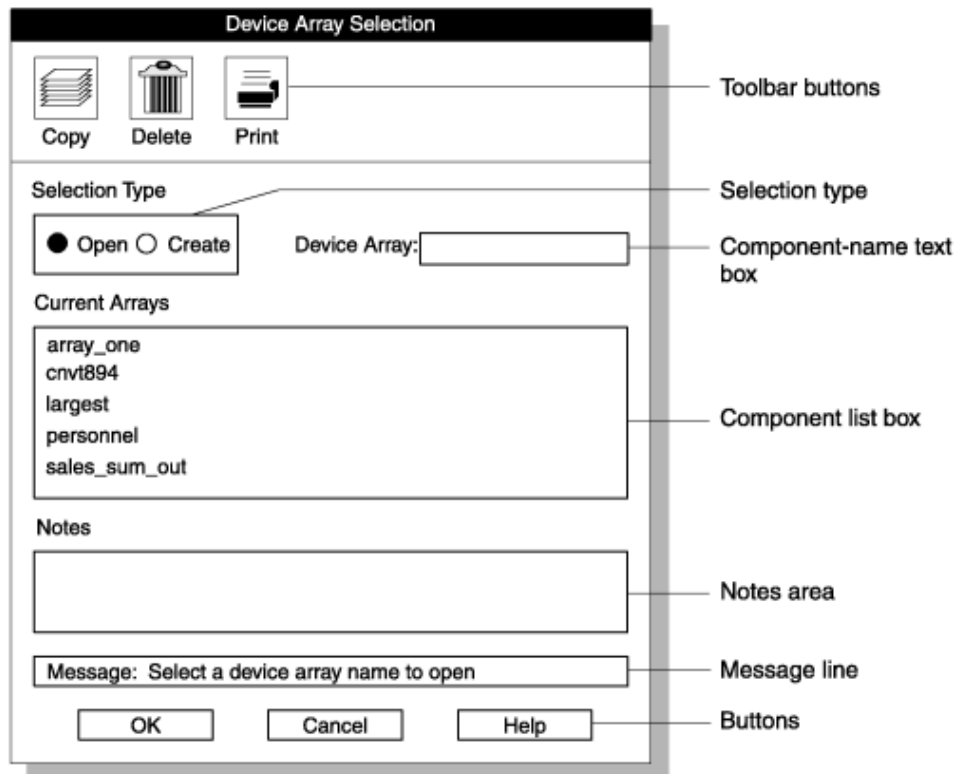


Table 4. The Device Array Selection window options

Display option	Description
Toolbar buttons	The buttons across the top of the display represent actions that you can take after you select a component from the component list. For example, in Figure 29: The Device Array Selection window on page 49 , the toolbar buttons indicate that you can copy, delete, or print an array. The HPL iupload utility buttons on page 57 explains how to use these buttons.
Select ion type	The selection type allows you to specify the action that you want to take. In most of the displays, you can either open an existing component or create a component.
Component-N ame text box	If you click Create , you must type a name for the new component in the Device-Array text box. (In Figure 29: The Device Array Selection window on page 49 , you must give a name for the new device array.)
Device Array text box	Before you can type a name in the Device Array text box, you must click inside that text box to activate it. When the text box is active, it has a narrow black border. If you type a character that is not valid, the interface beeps at you, displays a message on the message line, and refuses to display the invalid character.

Table 4. The Device Array Selection window options (continued)

Display option	Description
Component list box	The component list box lists the components that currently exist in this project. If you click Open in the selection group, you must select a component from this list.
Notes area	The notes area displays stored comments about the selected component. This area is not an active area. To store a comment about a component, you must select a component and use the Notes button. For more information about notes, see The Notes button on page 60 .
Message Line	The message line primarily gives instructions for the next logical action. The message line also gives an error message when an action fails or a completion message when a process is finished.
Buttons	The buttons across the bottom of the display let you indicate your next action. For a more complete discussion, see The HPL iupload utility buttons on page 57 .

Component-Definition windows

After you select a component to create or modify and click **OK** in the **Component-Selection** window, you typically see the **Component-Definition** window. From the **Component-Definition** window, you can enter, edit, or delete values or items that describe the component.

The following figure shows an example of a **Component-Definition** window: the **Device-Array Definition** window.

Figure 30. The Device-Array Definition window

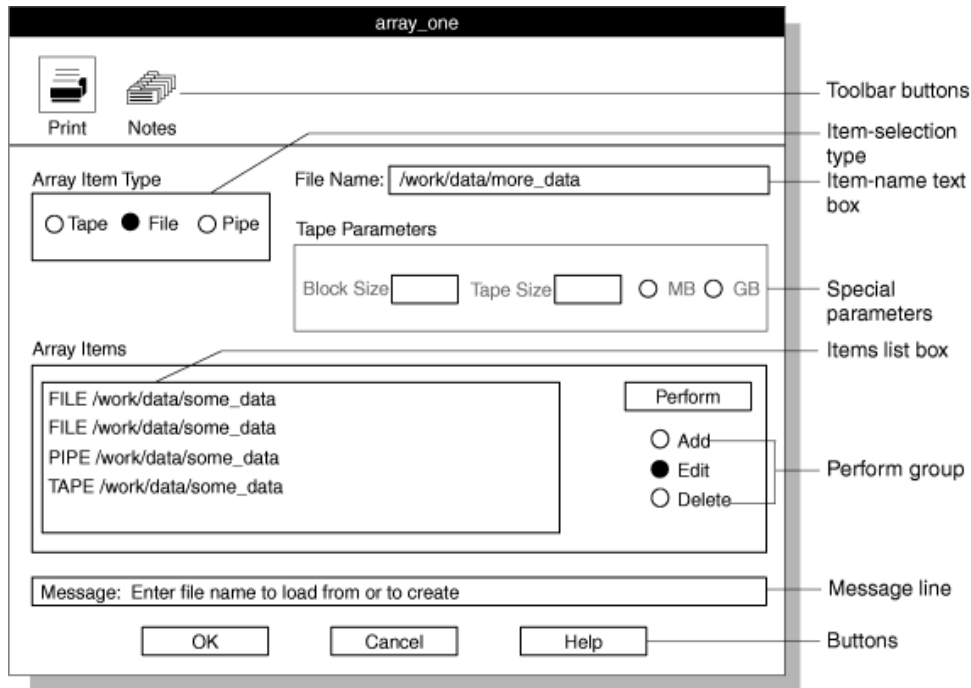


Table 5. The Device-Array Definition window options


Dis play option	Description
Tool bar buttons	The buttons across the top of the display represent actions that you can take after you select a component from the component list box. For example, in Figure 30: The Device-Array Definition window on page 50 (the Device-Array Definition window), the toolbar buttons indicate that you can print or make a note about an item. The HPL ipload utility buttons on page 57 explains how to use these buttons.
Item-selection group	The item-selection group lets you specify the type of item that you want to edit or the type of action that you want to take. After you specify a choice in the item-selection group, other options become active. In the Device-Array Definition window, the item-selection group is labeled Array Item Type . After you select Tape , File , or Pipe (only on UNIX™), other options become active.
Item-name text box	The item-name text box lets you specify the name or description of one of the items that makes up the component. For example, in the Device-Array Definition window, you type the full path name of a device in the item-name text box. In the Device-Array Definition window, the item-name text box is labeled Tape Name , File Name , or Pipe Name (only on UNIX™), depending on the type of component that you select from the Array Item Type group.
Special-parameters group	When a Component-Definition window first appears, some of the choices are inactive (shown in gray letters). In general, the inactive choices are not meaningful until you specify some other characteristic of the component that you are editing. The special-parameters group in the Device-Array Definition window is the Tape Parameters group. The items in the special-parameters group are meaningful only for tapes. The choices in the Tape Parameters group become active only if you select Tape from the Array Item Type group. The choices in Figure 30: The Device-Array Definition window on page 50 are gray because File is selected in the Array Item Type group.
Item list box	The item list box shows items that you already created to define the component. In the Device-Array Definition window, this list is labeled Array Items and shows the tapes, files, and pipes (UNIX™ only) that are already part of the current device array.
Perform group	The Perform group lets you specify the action that you want to take. After you select an item and an action, you must click Perform to complete the action. For example, to add a new device in the Device-Array Definition window, you must specify the name or description of the device and then click Perform to add it to the Array List .
	Important: Remember to click Perform to complete the action that you designated in the Perform group.
Message line	The message line primarily gives instructions for the next logical action. The message line also returns an error message when an action fails or a completion message when a process is finished.

Table 5. The Device-Array Definition window options (continued)

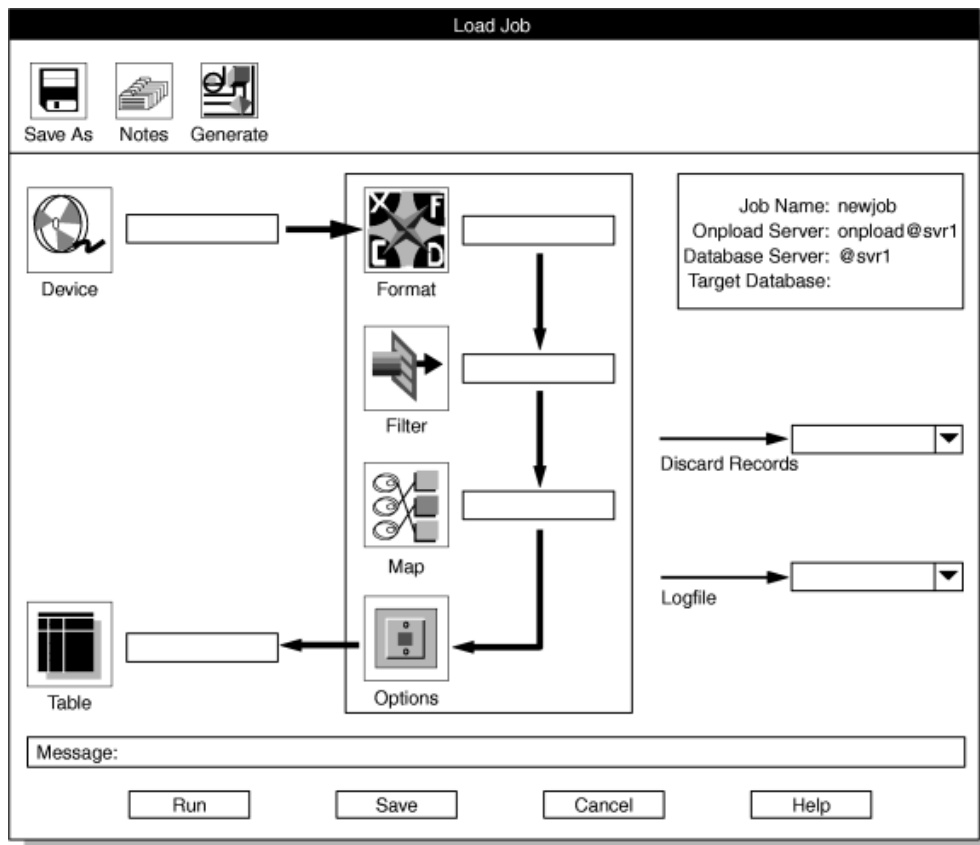
Dis play option	Description
Buttons	The buttons across the bottom of the display let you indicate your next action. For a more complete discussion, see The HPL ipload utility buttons on page 57 .

Load Job and Unload Job windows

The **Load Job** and **Unload Job** windows provide a visual presentation of the basic components that you choose for each job.

The following figure shows the **Load Job** window.

Figure 31. The Load Job window



Related reference

[Unload data from a database on page 142](#)

[Load data to a database table on page 150](#)

Views windows

A **Views** window shows a graphic representation of the relationships among various iupload components. From a **Views** window, you can search for specific components, select an existing component for editing, or create a component.

A **Views** window does not allow you to change any values. To change the values of a component, you must display a **Component-Definition** window.

Access views windows

The following table lists the four types of views windows and gives instructions for how to access each view.

Window name	Purpose	How to access	See
Format Views	Shows the load and unload maps that are associated with a particular format	<ul style="list-style-type: none"> • Click Search in the Record Formats window • Click Format in the Load Job window* • Click Format in the Unload Job window* 	Figure 13: The Record Formats window on page 28
Map Views	Shows the databases, tables, queries, and formats that are associated with a map	<ul style="list-style-type: none"> • Click Search in the Load Record Maps window • Click Search in the Unload Record Maps window • Click Map in the Load Job window* • Click Map in the Unload Job window* 	Figure 82: The Load Record Maps window on page 123
Database Views	Shows the tables in the database or the queries that are associated with the database	<ul style="list-style-type: none"> • Click Search in the Query window • Click Table in the Load Job window* • Click Query in the Unload Job window* 	Figure 71: The Query window on page 110
Filter Views	Shows the formats that are associated with a particular filter	<ul style="list-style-type: none"> • Click Search in the Filter window • Click Filter in the Load Job window* 	Filter views on page 141

* These options display the **Views** window only if the corresponding text box is empty. If the text box includes the name of a component, the **Component-Definition** window is displayed.

Available options in a Views window

The four types of **Views** windows operate in a similar manner. When a **Views** window appears, you have the following options:

- Type in a component name and search for the component.
- Click a label associated with an icon to expand the view and see related components.
- Click an icon to open the **Component-Definition** window that allows you to edit the component values.
- Click **Create** to display the **Component-Selection** window that allows you to create a component.

Search for a component in a Views window

You can use the **Search** button in a **Views** window to locate a specific component.

Type the component name in the search text box and then click **Search**. The view displays only the component names that match the text string.

You can use the following wildcard search characters in the search text string.

Table 6. Wildcard search characters

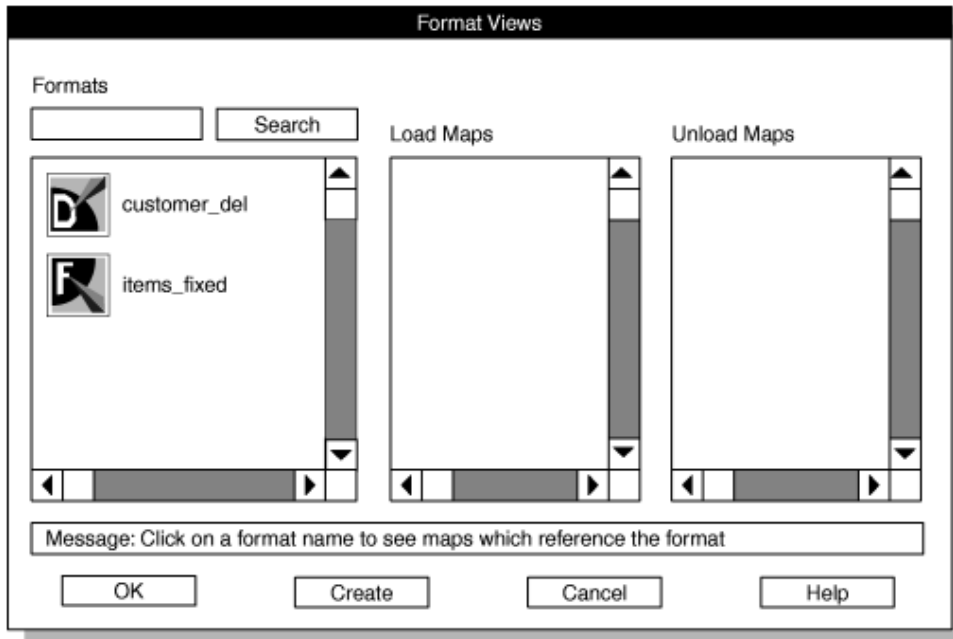
Wildcard symbol	Effect
?	Matches any single character
*	Matches any string of characters

Expand the view in a Views window

Three of the **Views** windows expand their views to show related components.

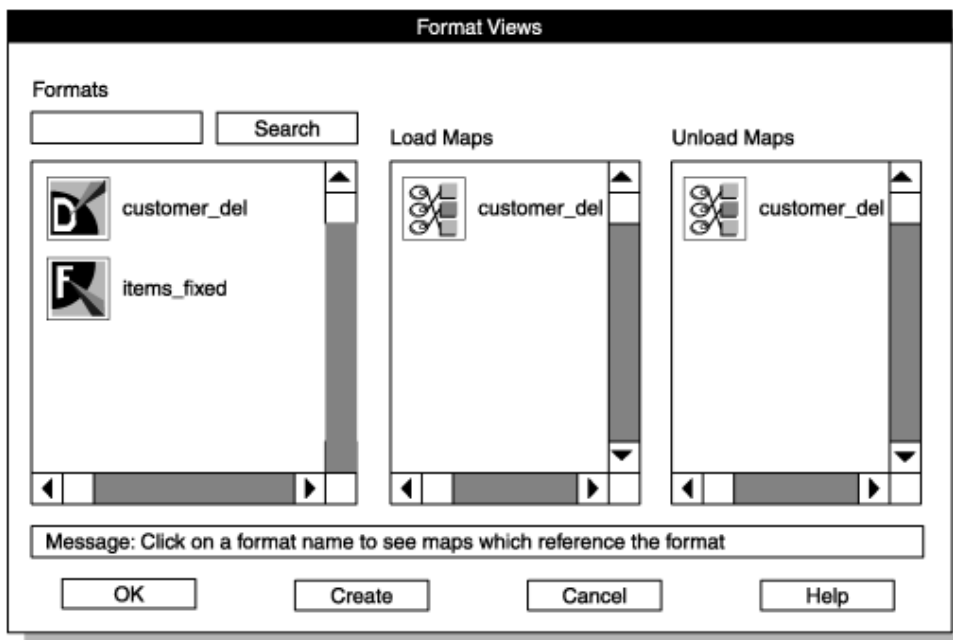
To expand the view, click an icon label (for example, **customer_del**) in the first pane. In the **Database Views** window, click an icon label in the second pane to expand the view further. The following figure shows the **Format Views** window.

Figure 32. The Format Views window



When you click an icon label in the **Formats** pane, the view expands to show maps that are related to your choice. The following figure shows the expanded view.

Figure 33. Expanded view of a format



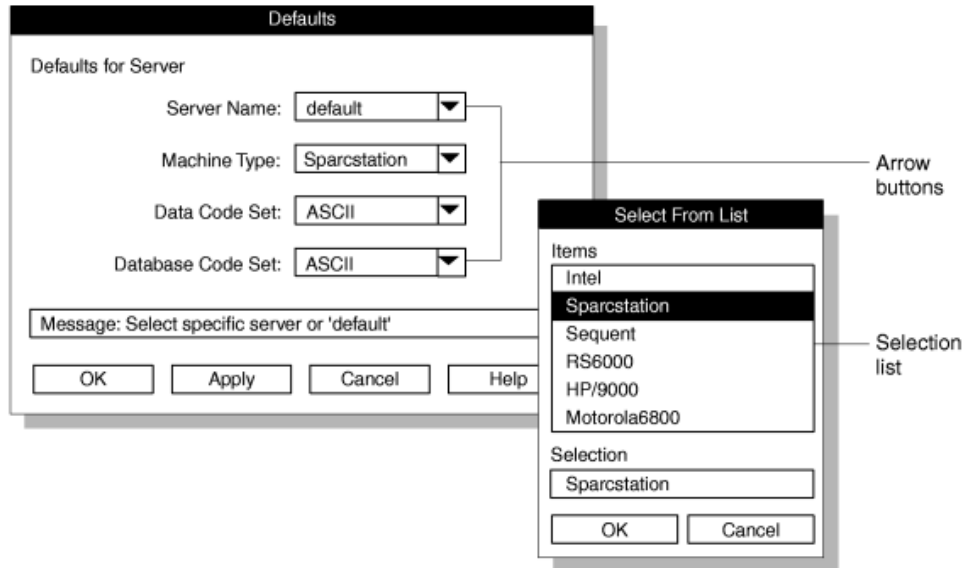
You can click the icon that you want to display a definition window for any format or map that is shown.

Selection-List windows

A **Selection-List** window lists the possible values for a text box. A down arrow that follows a text box indicates that you can use a selection list to see and select possible values for the text box. When you click the down arrow, the corresponding **Selection-List** window appears.

The following figure shows the selection list that is available for the **Machine Type** text box in the **Defaults** window. After you select an item in the list box, click **OK**, and the item appears in the text box on the original window.

Figure 34. The Defaults window and a selection list



i **Tip:** If your entry is rejected, look at the selection list. Your entry is invalid if it is not available in the selection list.

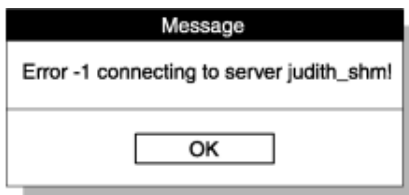
Selection-List windows are available for many text boxes throughout the HPL user interface. These windows have various names, but these topics refer to them as selection lists.

Message windows

A message window typically contains either a warning or an information update. A warning lets you verify or cancel the action that you have chosen. An update informs you about the successful completion of an operation or explains why an operation failed.

The following figure shows a typical error message.

Figure 35. The Message window



The HPL ipload utility buttons

After you move beyond the main window, every window has at least one button to help you move through the interface.











In general, buttons appear in three locations:





- Toolbar buttons appear across the top of the display.
- Icon buttons appear in the middle section of the display.
- Buttons appear across the bottom of the display.

The HPL ipload utility toolbar buttons

Toolbar buttons appear at the top of many windows. The function of the window determines which buttons appear.

The following sections describe the toolbar buttons. Buttons that appear in only one window are described with the specific window.

Button	Button name	Purpose	See
	Browse	Displays the Browse window	The HPL browsing options on page 170
	Copy	Copies the selected component (format, map, query, filter, device, or project) to a new item	The Copy button on page 58
	Connect	Lets you reattach to an active unload (or load) job from the Unload (or Load) Job Select window	Figure 103: The Load Job Select window on page 154
	Delete (trash can)	Deletes the selected component (format, map, and so on)	The Delete button on page 59
	Delete (eraser)	Breaks the association between a database column and a data-file field	Using the Delete button on page 130
	File	Displays the Import/Export File Selection window	Exporting and importing queries on page 116
	Find	Allows you to quickly locate a particular field or column in a Map window	Using the Find button on page 130
	Generate	Lets you generate jobs automatically	The Generate options of the ipload utility on page 160
	Notes	Allows you to type descriptive text for an item	The Notes button on page 60
	Options	Displays an options window where you can change default values or supply additional parameters	Changing the unload options on page 148, Changing the load options on page 158

Button	Button name	Purpose	See
	Print	Prints the parameters for the selected item	The Print button on page 62
	Save As	Saves a copy of the currently selected item (behaves in the same way as the Copy button)	The Copy button on page 58
	Search	Displays a Views window where you can see the relationships among components	Views windows on page 53
	Specs	Displays the Specifications window, where you can view the attributes for selected columns or fields	Using the Specs button on page 131

The Browse button

With the **Browse** button, you can look through the files that show information about the load or unload jobs and any problems that the onpload utility found.

Related information

[Browsing options on page 170](#)

The Copy button

You can copy a selected component with the **Copy** button. This feature can save you time when you are creating a component. You can copy an existing component and then modify the copy with your changes.

You can copy one component at a time, or you can select and copy multiple components at the same time. You can copy components that are grouped under a project (filters, formats, maps, and queries) within the same project, or to a different project.

If you copy a component within a project, you must give the copy a different name. If you copy a component to a different project, you can retain the name for the copy or give the copy a different name. If you copy multiple components, you must copy them to a different project. When you copy multiple components, the components retain their names.



Important: Devices are not project specific. When you copy a device, you must give the copy a new name.

Copying an existing format into a new format

About this task

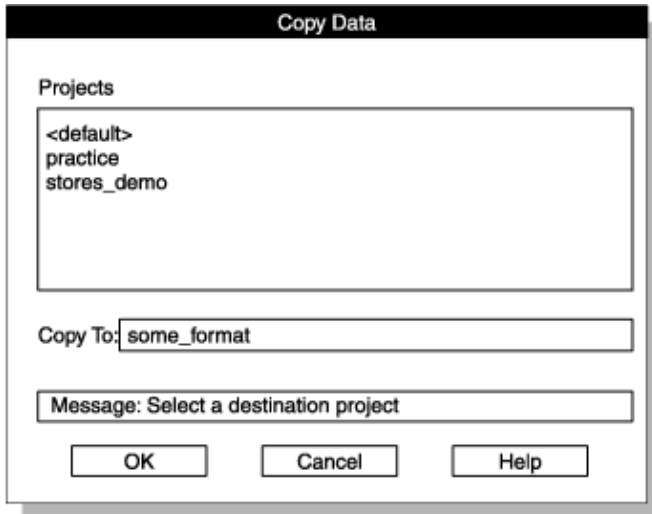
To copy an existing format to a new format:

1. In the HPL main window, select the project that includes the format that you want to copy.
2. Choose **Components > Formats** to access the **Record Formats** window.
For an example, see [Creating a fixed format on page 86](#).

3. Select the format that you want to copy.
This example assumes that the format to copy is **some_format**.
4. Click the **Copy** button.

The **Copy Data** window appears, as the following figure shows. The **Copy Data** window displays a list of existing projects. The **Copy To** text box shows the name of the format that you are copying.

Figure 36. The Copy Data window



5. Select the project to which you want to copy the format.
6. Type the name that you want to give to the copied format in the **Copy To** text box.

If you are copying the format to another project, you can keep the same name. You must change the name, however, if you are copying the format to the same project.

7. Click **OK**.
The display returns to the **Record Formats** window.
8. Click **Cancel** to return to the HPL main window.

The Delete button

The **Delete** button lets you delete one or more selected components.

Deleting an existing format

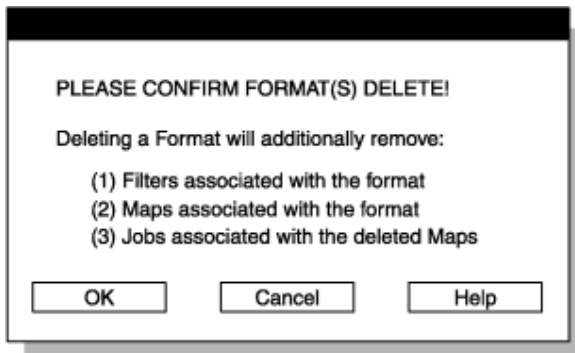
About this task

To delete an existing format:

1. In the HPL main window, select the project that includes the format that you want to delete.
2. Choose **Components > Formats** to access the **Record Formats** window.
For an example, see [Creating a fixed format on page 86](#).
3. Select the format that you want to delete.
4. Click the **Delete** button.

The **Confirm Delete** window appears, as the following figure shows. The **Confirm Delete** window describes the impact of deleting this format. The text in this window is different for each of the component types.

Figure 37. The Confirm Delete window



5. Click **OK** to confirm the deletion, or click **Cancel** to cancel it.

If you click **OK**, the format is deleted and any associated maps, filters, and jobs.

6. Click **Cancel** to return to the HPL main window.

The Notes® button

You can type descriptive text for an item with the **Notes** button. The text of the note is displayed in the **Notes** area in a window when you select the item. The **Notes** button is a useful tool for identifying ipload components, load jobs, unload jobs, and projects.

Related reference

The note table in the onpload database

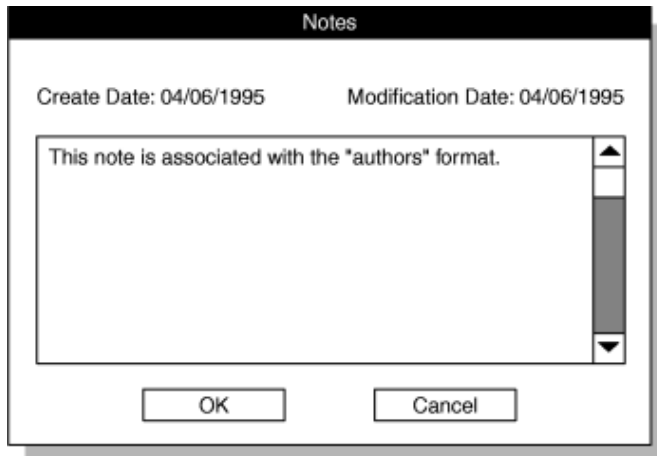
Creating a note

About this task

To create a note:

1. Click the **Notes** button in a **Component-Definition** window.

The **Notes** window appears, as the following figure shows.

Figure 38. The **Notes** window

2. Type the descriptive text in the **Notes** text box.
3. Click **OK** to store the note and return to the **Component-Definition** window.

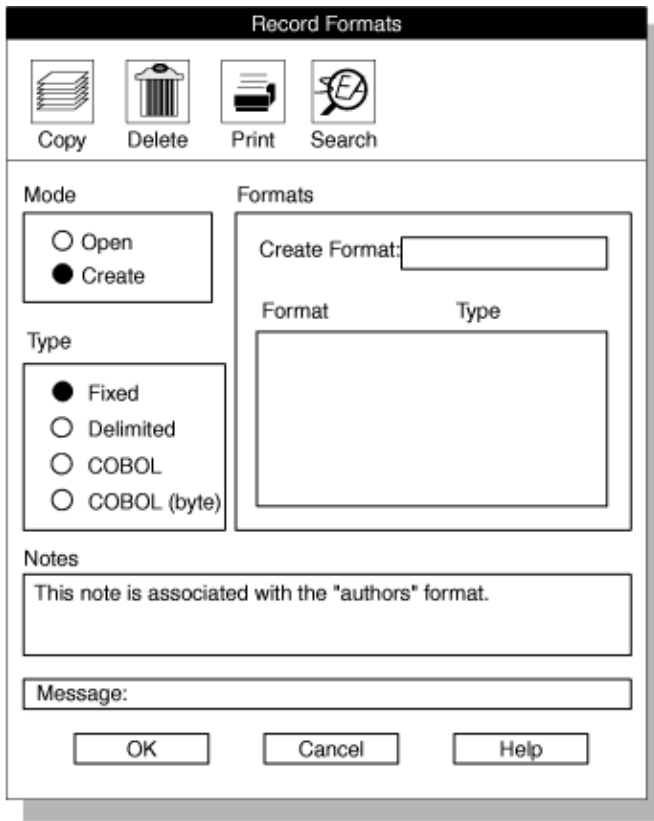
When you select the component, the note text is displayed in the **Notes** area.

Results

If you do not change a note, click **Cancel** instead of **OK**.

For example, the note created in the **Notes** window is associated with the **authors** format. The next time you go to the **Record Formats** window and select **authors**, ipload displays the note text, as the following figure shows.

Figure 39. The Record Formats window with notes text



The ipload utility stores the information that you type in the **note** table window of the onpload database. For a description of the **note** table, see The note table in the onpload database.

The Print button

You can print information that is associated with a component with the **Print** button. Before you start ipload, you must set your workstation so that it can find a printer.

For information about setting up a printer, see your operating-system publications.

If you click the **Print** button in the **Map-Definition** window in [Figure 20: The Map-Definition window with all associations completed on page 36](#), the following printout results:

```
-----
LOAD MAP REPORT
-----
```

```
Project : <default>
Name: a_map
```

```
OPTIONS
```

```
Database Table Format
```

```
-----
testdbtab1a_format
```




```
RECORD FORMAT MAP VIEW
```




```
Format FieldTable Column Option Data
-----
input1 col3
input2 col2
input3 col1
```


The HPL ipload utility icon buttons

Icon buttons appear in the middle sections of the **Load Job**, *Unload Job*, and **Views** windows. The icon buttons represent various components. When you click it, each button opens another display.

The following table shows and describes the icon buttons that are used in these windows.

Component	Description	Window	Action
 Device	The device or device array where the source files are located	Load Job, Unload Job	<ul style="list-style-type: none"> If the text box is empty, click the Device button to display the Device Array Selection window, where you can create or open a device type. If the text box has an entry, click the Device button to display the Device Array Selection window for that specific device or type the name of a different device in the text box.
 Filter	The filter that controls which records are selected from the data file for a database update (The use of a filter is optional.)	Load Job, Filter Views	<ul style="list-style-type: none"> If the text box is empty, click the Filter button to display the Filter Views window, where you select a filter and associated format. You can also create a filter from this window. If the text box has an entry, click the Filter button to display the Filter-Definition window for that specific filter or type the name of a different filter in the text box. In the Filter Views window, click the Filter button to display the Filter-Definition window for a specific filter.
 Format	The format of the source data used for this load or unload	Load Job, Unload Job	<ul style="list-style-type: none"> If the text box is empty, click the Format button to display the Format Views window, where you can select a format and associated map. You can also create a format from this window. If the text box has an entry, click the Format button to display the Format-Definition window for that

Component	Description	Window	Action
			<p>specific format or type the name of a different format in the text box.</p> <ul style="list-style-type: none"> • In all Views windows, click the Format button to display the format definition for a specific format. In these windows, the button shows only one of the three symbols (F, D, C) to indicate whether the type of format is fixed, delimited, or COBOL.
 <p>Map</p>	<p>The map that correlates fields of the data source to database columns</p>	<p>Load Job, Unload Job, Map Views, Format Views, Database Views</p>	<ul style="list-style-type: none"> • If the text box is empty, click the Map button to display the Map Views window, where you can select a map and associated table and format. You also can create a map from this window. • If the text box has an entry, click the Map button to display the Map-Definition window for that specific map or type the name of a different map in the text box. • In a Views window, click the Map button to display the Map-Definition window for a specific map.
 <p>Options</p>	<p>The options that let you specify characteristics of the load or unload</p>	<p>Load Job, Unload Job</p>	<ul style="list-style-type: none"> • Click the Options button to display the Load Options window.
 <p>Query</p>	<p>The query that selects data from the database table</p>	<p>Unload Job, Database Views, Map Views</p>	<ul style="list-style-type: none"> • If the text box is empty, click the Query button to display the Database Views window from which you can select the table and associated map and format. • If the text box has an entry, click the Query button to display the Query-Definition window for that specific query or type the name of a different query in the text box. • In a Views window, click the Query button to display the Query-Definition window for a specific query.

Component	Description	Window	Action
 Table	The database table into which the converted data will be loaded	Load Job, Database Views, Query Definition	<ul style="list-style-type: none"> Click the Table button to display the Database Views window from which you can select the table and associated map and format. If an association is not apparent, click Create to create one. Click the Table button in the Query-Definition window to choose a table and columns for the Select entry.

Related information

[Changing the load options on page 158](#)

Buttons at the bottom of the HPL ipload utility display

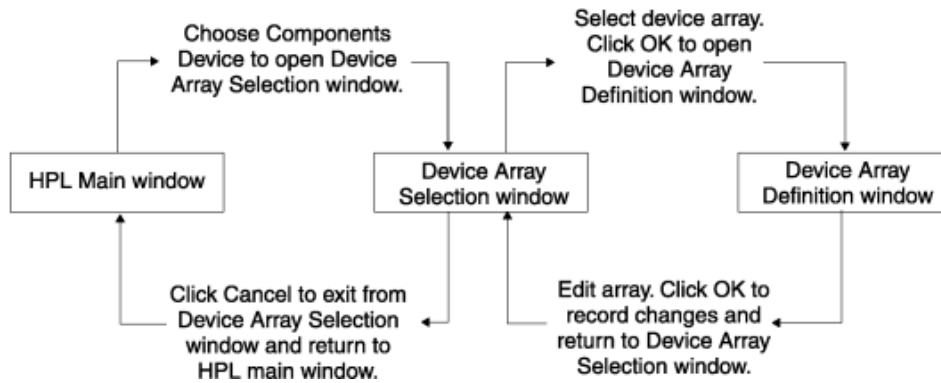
The buttons across the bottom of the display let you indicate the next action. Most windows have one or more of these buttons.

Table 7. Buttons that indicate the next action

Button name	Action
Apply	Saves changes but does not exit.
Cancel	Does not save any changes. Exit to the previous display.
Create	Displays the Component-Selection window.
Help	Displays context-sensitive help in a separate window. For information about the online help, see The HPL online help on page 66 .
OK	Save changes and exit to the previous display.

Use **OK** only when you have changed the display. If you are exiting from a series of displays, use **Cancel** to exit from the display. The following figure shows the use of **OK** and **Cancel**.

Figure 40. Using OK and Cancel from the HPL main window



The HPL online help

The **Help** menu on the High-Performance Loader (HPL) main window has Glossary and Contents choices.

The **Glossary** option opens a scrolling list of items. Select an item to see its definition. The **Contents** option takes you to the main contents page. This page directs you to discussions of various HPL topics.

If you click **Help** in any window other than the HPL main window, **Help** displays information that is related to the current window. After the **Help** window opens, you can click its **Help** button for more information about using the **Help** window.

The UNIX™ keyboard commands to move the cursor

Instead of using the mouse to move from area to area in the High-Performance Loader (HPL) user interface, you can use UNIX™ keyboard commands to move the cursor. As you move around, the currently selected item is highlighted with a box.

The following table lists the cursor-moving keystrokes.

Table 8. Cursor moving keystrokes

Keystroke	Result
TAB	Move from area to area. Sometimes used to move from tab stop to tab stop.
SHIFT-TAB	Back up; that is, move from area to area in reverse order.
CONTROL-TAB	Move from area to area when TAB is reserved to move from tab stop to tab stop.
Cursor keys	Move from item to item within a functional area.
SPACEBAR	Select the current item or action.

Most displays in the HPL user interface are divided into functional areas, such as toolbar buttons, selection group, component-name text box, component list box, and so on. Depending on the nature of the specific display, sometimes **TAB** moves from item to item (or even from tab stop to tab stop) within a major area. On other displays, **TAB** moves only between functional areas, and you must use **SPACEBAR** to move around within the functional area.

Define HPL projects

This section explains how to create a project and how projects are related. The individual components that you store in projects are described in later sections.

Related reference

[Choose a project on page 21](#)

HPL projects

You can organize your work by specifying *projects* with the High-Performance Loader (HPL). A project is a collection of individual pieces that you use to load and unload data. A project can include load and unload jobs and the maps, formats, filters, and queries that you use to build the load and unload jobs.

Project organization

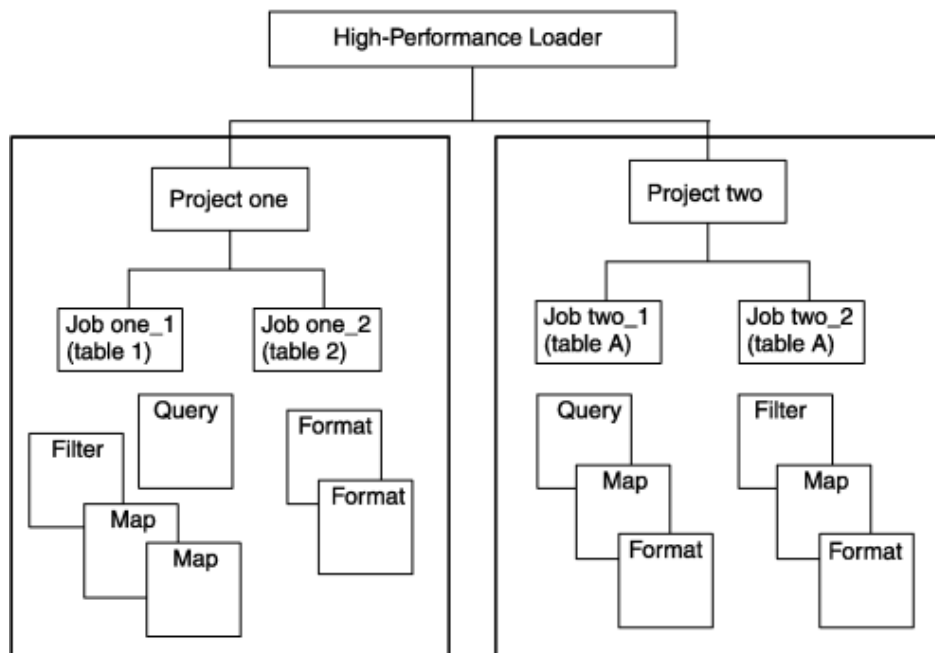
The High-Performance Loader (HPL) uses only one database, onpload, to track the preparation that you do for loading and unloading data. By using projects, you can organize your work into functional areas.

For example, you might regularly transfer data to or from several unrelated databases. You could put all of the preparation for each database into a separate project.

When you first start ipload, ipload creates a project named **<default>**. If you prefer, you can select the **<default>** project and assign all of your work to that project. The HPL does not require that you create any additional projects. However, creating projects and putting separate tasks into distinct projects makes your work easier to maintain.

The following figure shows the relationships among projects, jobs, and components.

Figure 41. Illustration of project hierarchy



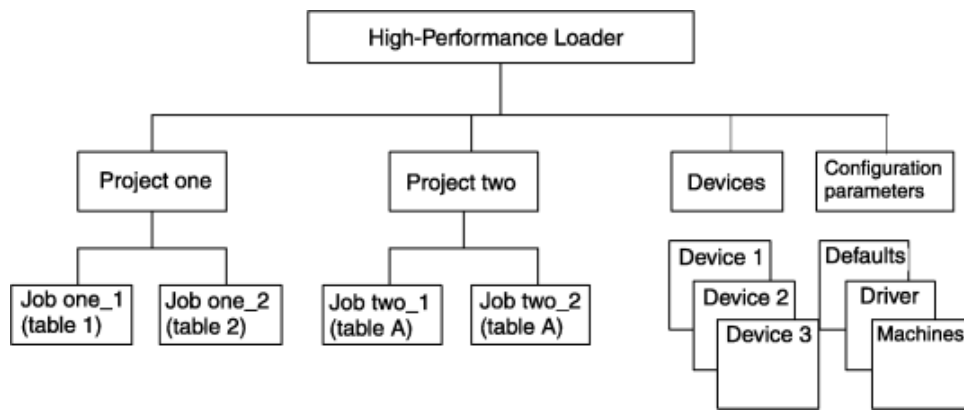
The preceding figure shows that jobs are linked directly to the projects. The format, map, filter, and query components belong to a project but are not directly linked to a job, as illustrated with Project One. In general, you create a format, map, and filter or query for each job, as shown with Project Two. However, in some cases, you might use the same component for more than one job within a project.

For example, for reports about a medical study, you might want to create three reports: one about subjects under 50 years of age, one about subjects over 50, and one about all subjects. In that case, the description of how to find the information (the format and map) is the same for all three reports, but the selection of information (the query) is different for each report. (Formats, maps, and queries are described in detail in later chapters.)

All components (maps, formats, queries, filters, and load and unload jobs) that you create in a project are associated with that project in the onpload database. Components that are associated with a project are visible (usable) only when the project is selected. When you select a different project, a different set of components becomes available.

Device-array definitions and configuration parameters are not included in project definitions. The following figure shows the components that the HPL uses. Each project is distinct, but the devices and configuration parameters apply to all projects.

Figure 42. Relationship of projects, devices, and configuration parameters



Related reference

[Define projects on page 233](#)

The project table in the onpload database

Select or create a project with the Projects window

You can select or create a project from the **Projects** window. After you select a project, you can copy the project, delete it, print the project parameters, or make a note that describes the project.

The ipload utility stores project information in the **project** table of the onpload database.

Figure 43. The Projects window



Related reference

The project table in the onpload database

Defining a new project

About this task

To define a new project:

1. Choose **Configure > Project** from the HPL main window.

The Projects window appears, as [Figure 43: The Projects window on page 69](#) shows.

2. Choose a name for the project and type it in the **Create Project** text box.
3. Click **Apply**.

The ipload utility creates the project but does not exit from the **Projects** window. You can create another project, or you can use the toolbar buttons to manipulate the project.

4. Click **Cancel** to return to the HPL main window.

Results

If you want to create one project and then exit, click **OK** instead of **Apply**.

Related information

[Initial options on the HPL main window on page 46](#)

Selecting a project

The HPL provides two methods for selecting a project.

About this task

Selecting a project for a load or unload job

About this task

To select a project for a load or unload job or to edit components:

1. Select the project name from the **Select Project** list box in the HPL main window.
2. Choose the action that you want to take from one of the menus on the HPL main window.

Selecting a project to edit

About this task

To select a project to edit:

1. Choose **Configure > Project** from the HPL main window.

The **Projects** window appears, as [Figure 43: The Projects window on page 69](#) shows.

2. Select the project that you want to edit from the **Projects** list box.
3. Perform the edit actions (copy, delete, print, or describe with a note) you want.
4. Click **Cancel** to return to the HPL main window.

Configure the High-Performance Loader

This section describes the process of configure the High-Performance Loader (HPL).

Related reference

[Check the ipload utility default values on page 21](#)

Configure the ipload utility

When you configure the ipload utility, you describe the type of computer, code sets, and other aspects of your database server environment. Configuration information is stored in the onpload database.

The configuration tasks include:

- Selecting a database server
- Modifying the onpload defaults
- Selecting or creating a driver
- Modifying the machine description

Related reference

[HPL performance on page 182](#)

Related information

[The ipload utility on page 8](#)

Select a database server

The High-Performance Loader (HPL) needs to know the location of two databases: the onpload database and the *target database*.

The target database is the HCL Informix® database into which you load data or from which you unload data. When you start ipload, ipload assumes that both the onpload database and the target database are on the database server that the **INFORMIXSERVER** environment variable specifies. You can use the **Connect Server** window ([Figure 44: The Connect Server window on page 72](#)) to specify different database servers.

The `sqlhosts` file or registry controls connectivity to database servers. The ipload utility scans the `sqlhosts` information to derive the lists of available database servers that the **Connect Server** window displays.



Restriction: You cannot use the alias **loghost** as a machine name in the `sqlhosts` file. The **loghost** alias is present on all computers; consequently, onpload cannot correctly identify computers based on this name.

Related information

[Initial options on the HPL main window on page 46](#)

Selecting a database server

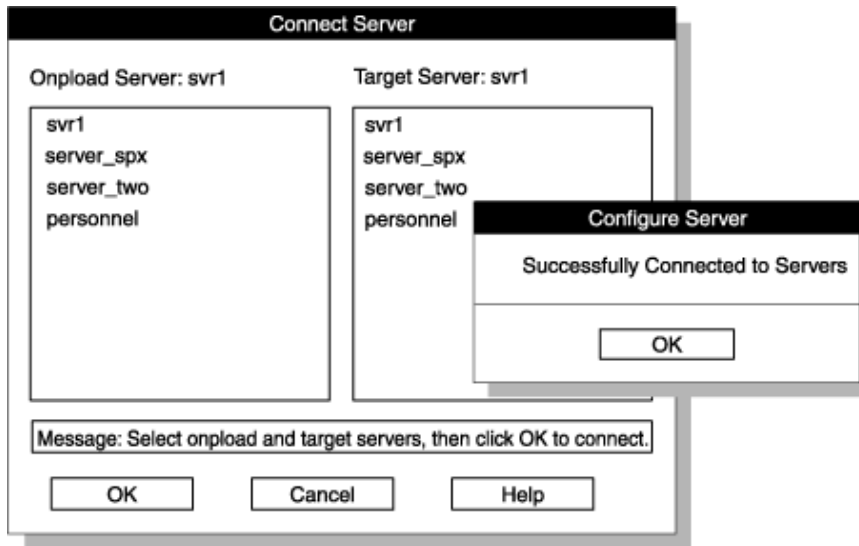
About this task

To select a database server:

1. Choose **Configure > Server** from the HPL main window.

The **Connect Server** window appears, as the following figure shows.

Figure 44. The Connect Server window



2. Select the database server where the onpload database is going from the **Onpload Server** list box.
3. Select the database server that includes the database that you will load or unload from the **Target Server** list box.
4. Click **OK**.
5. Click **OK** in the **Configure Server** window to return to the HPL main window.

Create the onpload database

When you first start ipload, ipload creates an onpload database. If you use the **Connect Server** window to choose a different onpload database server, ipload creates another onpload database on that database server.

The default name of the database that the High-Performance Loader (HPL) uses is onpload. To give some other name to the HPL database, set the **DBONPLOAD** environment variable.

Related information

[Preparing multiple onpload databases on page 11](#)

Modify the onpload default values

You must describe the computer environments of your database servers. This information applies to database servers. If you change the description of a database server, the changes apply to all jobs that you run on that database server. You can prepare a default computing environment that applies to all database servers that are not explicitly described.

Related reference

[Look at the Defaults window on page 21](#)

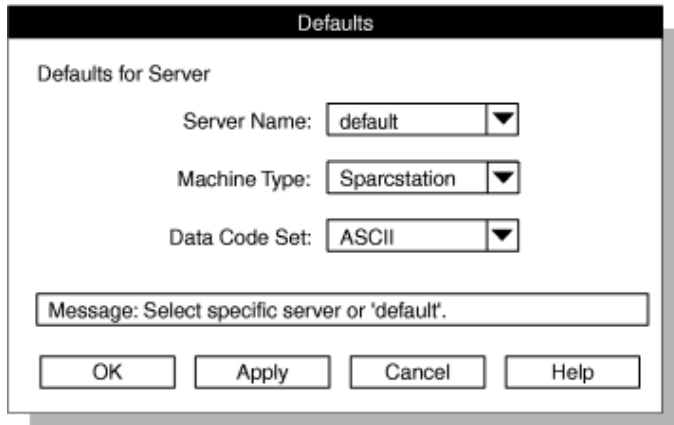
The defaults table in the onpload database

The Defaults window

You can change the server name, machine type, and data code set from the **Defaults** window.

The following figure shows the **Defaults** window.

Figure 45. The Defaults window



The ipload utility saves the information from the **Defaults** window in the **defaults** table of the onpload database. For more information about the **defaults** table, see The defaults table in the onpload database.

i **Tip:** You can use DB-Acess to examine the default values. The following tables in the onpload database contain default values: **defaults**, **delimiters**, **driver**, and **machines**.

Table 9. The Defaults window options

Di
sp
lay
o
pt
ion

Header

Se The **Server Name** text box specifies the database server with which the settings are associated. The information provided for the special server name **default** applies to all database servers for which no explicit information is provided. For example, if most the database servers on your network (that will be using the HPL) are BrandX computers, **default** should describe the BrandX computers. To describe the computing environment of the other database servers on the network, specify the database server name.

ame The selection list that is associated with the **Server Name** text box lists the database servers that are identified in the sqlhosts information.

M The **Machine Type** text box describes fixed-length, binary-format records. It defines the sizes and byte order of data in data files that the specified database server produces. The selection list that is associated with the **Machine Type** h

Table 9. The Defaults window options (continued)

<p>Display options</p> <p>The Display options text box provides descriptions of several computers. You can use the Machines option on the Configure menu to add descriptions of other computers to this list (see Modify the machine description on page 76).</p> <p>The Data Code Set text box specifies the character set of the data file. When you load data into a database, you can convert the character set of the data file into the character set of the database. For example, you can convert EBCDIC to ASCII, or any other character set that your system supports. Conversely, you can convert the data from a database into a selected character set when you unload data.</p> <p>You can select the GLS code set you want from this selection list. The character set of the database is determined by the DB_LOCALE environment variable. For information about locales and code sets, see the <i>HCL® Informix® GLS User's Guide</i>.</p>	<p style="text-align: center;">Header</p>
---	--

Specifying the onpload defaults

About this task

To specify defaults for onpload:

1. Choose **Configure > Defaults** from the HPL main window.

The **Defaults** window appears, as [Figure 45: The Defaults window on page 73](#) shows.

2. Update the values in each of the text boxes.
Click the down arrows to display selection lists that show possible values for each text box.
3. Click **Apply** to save the values and prepare the defaults for another database server.
4. Click **Cancel** to return to the HPL main window.

Results

If you want to prepare the defaults for only one database server, click **OK** instead of **Apply**.

Related information

[Modify the machine description on page 76](#)

Selecting a driver

About this task

In a UNIX™ environment, you can use pipe-device arrays to connect custom programs to the input or output of the onpload utility. Although pipe-device arrays provide the simplest way to customize the input or output of onpload, connecting the standard I/O of programs is less efficient than directly incorporating the functionality into onpload.

The ipload utility identifies custom software by the driver name that you assign to the record-format definitions.

To incorporate custom file-handling software directly into the onpload program:

1. Use the API described in Available API support functions to write the driver.
2. Add the driver name to the onpload database.
For details, see [Adding a custom-driver name on page 76](#).
3. Select the driver from the **Driver** selection list in the **Format Options** window.
For details, see [Format options on page 103](#).

The Drivers window

You can add information about custom drivers from the **Drivers** window. After you add a custom driver name, you can assign the driver to the record-format definitions.

The following figure shows the **Drivers** window. The **Driver Name** list box displays the currently available drivers, their class, and type.

Figure 46. The Drivers window



Table 10. The Drivers window options

Dis play option	Description
Driver Name	The Driver Name text box specifies the name of the custom-driver program. Before you can use the custom driver, you must add the program to your onpload shared library. For more information, see Rebuilding the shared-library file.
Driver Class	The Driver Class text box specifies the format type that the custom driver supports. The custom driver must produce data in a format that onpload supports. The onpload utility supports the following driver classes: <ul style="list-style-type: none"> • Fixed • Delimited • COBOL

Related reference

The driver table in the onpload database

Adding a custom-driver name

About this task

To add a custom-driver name:

1. Choose **Configure > Driver** from the HPL main window.

The **Drivers** window appears, as [Figure 46: The Drivers window on page 75](#) shows.

2. Type the name of the driver that you are adding in the **Driver Name** text box.
3. Select the driver class.
4. Click **Apply** to save this driver and add another driver.

If you want to add only one driver, click **OK** instead of **Apply**.

5. When you finish, click **OK** to return to the HPL main window.

Modify the machine description

The information that the **Machines** option of the **Configure** menu stores describes the characteristics of a specific computer. The High-Performance Loader (HPL) uses these characteristics to control the conversion of binary data formats. Unless you are converting binary data, you do not need to be concerned about the machine description.

When you first start ipload, ipload stores the characteristics of several computers. You can select one of the existing computer types, modify an existing description, or create a machine description.

You use the information from the **Machines** window when you prepare the defaults for the database servers on your network. The information from the Machines window is stored in the **machines** table of the onpload database. The default information for the HPL includes descriptions of the binary data sizes for several computers. If the default data does not include the computer from which you are reading data, you can create a description for that computer.

Related reference

[Look at the Machines window on page 21](#)

The machines table in the onpload database

Related information

[Specifying the onpload defaults on page 74](#)

The Machines window

The following figure shows the **Machines** window. If you select SPARCstation from the **Machine Type** selection list, the following values appear in the **Machines** window.

Figure 47. The Machines window

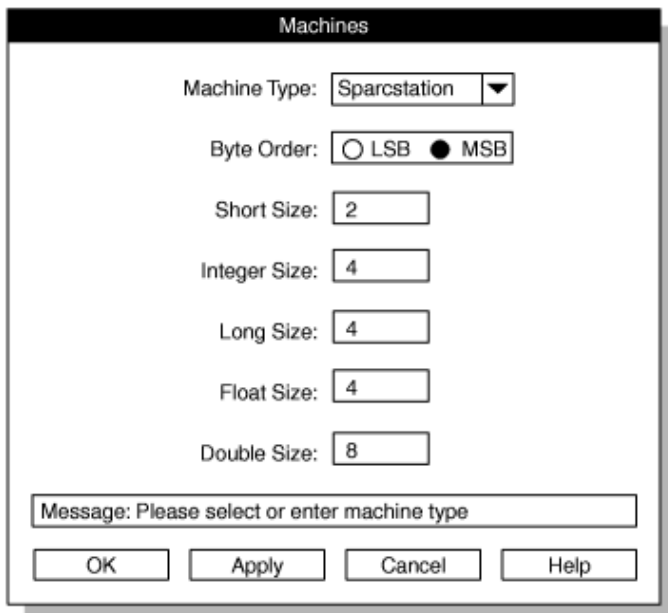


Table 11. The Machines window options

Option	Description
Machine Type	Name for the computer type that this entry describes.

Table 11. The Machines window options (continued)

Option	Description
Display option	
Byte Order	Bit ordering of binary information for this computer. The two possible formats are LSB and MSB. In the LSB format, the least-significant bits of a value are at lower memory addresses. In the MSB format, the most-significant bits of a value are at lower memory addresses.
Short Size	Number of bytes required to hold a short integer value.
Integer Size	Number of bytes required to hold an integer.
Long Size	Number of bytes required to hold a long-integer value.
Float Size	Number of bytes required to hold a floating-point value.
Double Size	Number of bytes required to hold a double-sized floating-point value.

Related reference

The machines table in the onpload database

Editing the description of a computer

About this task

To edit the description of a computer:

1. Choose **Configure > Machines** from the HPL main window.

The **Machines** window appears, as [Figure 47: The Machines window on page 77](#) shows.

2. Click the down arrow and select a machine type from the selection list.
3. Modify the values as appropriate.
4. Click **Apply** to save the values and modify another machine description.
5. When you finish, click **Cancel** to return to the HPL main window.

Results

If you want to modify the description of only one computer, click **OK** instead of **Apply**.

Adding a computer type to the Machines list

About this task

To add a computer type to the Machines list:

1. Choose **Configure > Machines** from the HPL main window.

The **Machines** window appears, as [Figure 47: The Machines window on page 77](#) shows.

2. Type the new name in the **Machine Type** text box.
3. Type an appropriate value in each text box.
4. Click **Apply** to save the values and add another computer type.
5. When you finish, click **Cancel** to return to the HPL main window.

Results

If you want to add only one computer type, click **OK** instead of **Apply**.

Define device arrays

This section describes how to define and use device arrays with the High-Performance Loader (HPL).

Related information

[Components of the unload job on page 142](#)

Device arrays

A device array groups several I/O devices so that the High-Performance Loader (HPL) can perform parallel processing of the input and output. When you specify multiple devices in a device array, onpload sets up separate, parallel streams of input or output, when it performs a database load or unload.

Device arrays set up simultaneous access to one or more tape devices, files, or pipes (UNIX™ only) so that the onpload utility can take advantage of parallel processing.

Device arrays are not project specific. You can use the same device array for a load or unload job on any of the projects that you define.

Related reference

[Define device arrays on page 214](#)

Multiple devices in a device array

You can include files, pipes (UNIX™ only), and tape devices in a single array. [Devices for the device array on page 184](#) discusses factors that you should take into account when you decide what devices to assign to an array. If your array includes pipe commands, onpload starts the pipe when it begins execution.

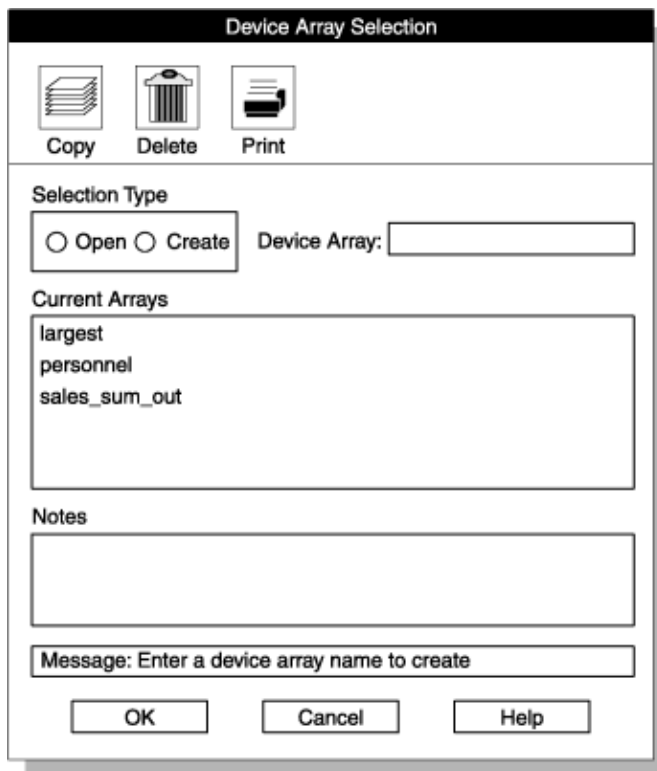
When the High-Performance Loader (HPL) unloads data, it assigns records to the devices of a device array in a round-robin fashion.

The Device-Array Selection window

You can create a device array or select an existing array from the **Device-Array Selection** window. If you select an existing array, you can edit that array or use one of the toolbar buttons to copy, delete, or print the array.

The following figure shows the **Device-Array Selection** window.

Figure 48. The Device-Array Selection window



Creating a device array

About this task

To create a device array:

1. Choose **Components > Devices** from the HPL main window.

The Device-Array Selection window appears, as [Figure 48: The Device-Array Selection window on page 80](#) shows.

2. Click **Create** in the **Selection Type** group.
3. Select a name for the new device and type it in the **Device Array** text box.
4. Click **OK**.

The **Device-Array Definition** window appears, as [Figure 49: A partially completed Device-Array Definition window on page 82](#) shows.

Opening an existing device array

About this task

To open an existing device array:

1. Choose **Components > Devices** from the HPL main window.

The **Device-Array Selection** window appears, as [Figure 48: The Device-Array Selection window on page 80](#) shows.

2. Click **Open** in the **Selection Type** group.
3. Select a device from the **Current Arrays** list box.
4. Click **OK**.

The **Device-Array Definition** window appears, as [Figure 49: A partially completed Device-Array Definition window on page 82](#) shows.

The Device-Array Definition window

You can add, edit, or delete devices from an array from the **Device-Array Definition** window.

Figure 49. A partially completed Device-Array Definition window

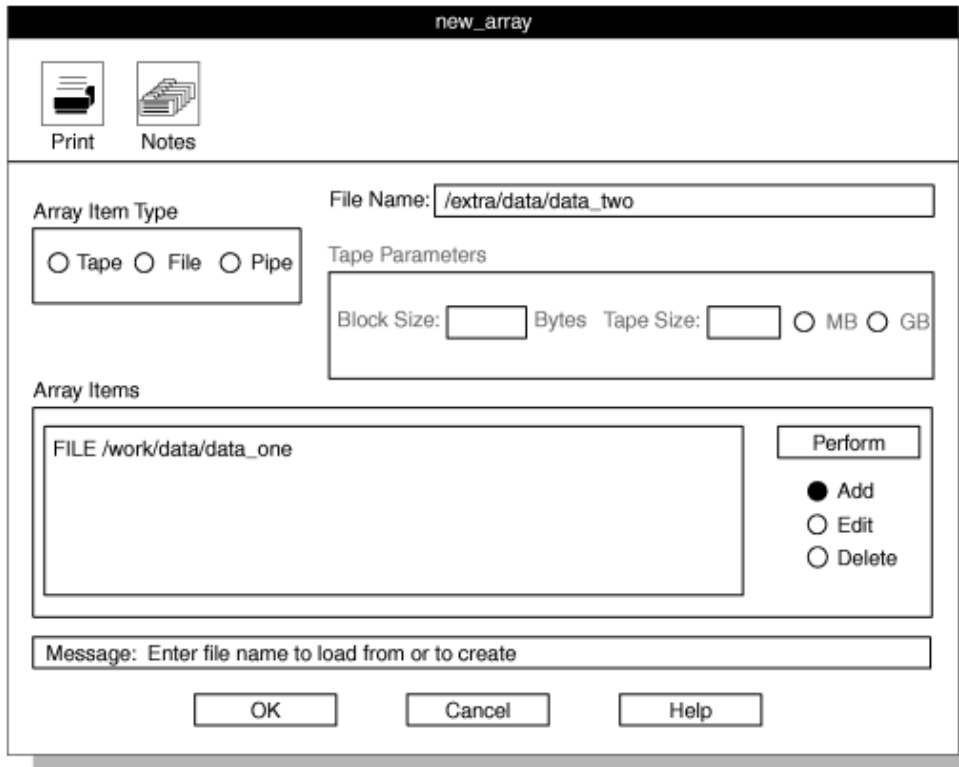


Table 12. The Device-Array Definition window options

	Display option	
Array Item Type group		The Array Item Type types of devices in a
Device text box		Depending on the arr you type a device nar follows.
		Device type
		What to type
		Tape Device
		The full path n
		File Name
		The full path n
		Pipe Command
		The full path n
Tape Parameters group		When you select Tap figure shows. You m

Table 12. The Device-Array Definition window options (continued)**Display option**

the following figure is
not support remote t
Figure 50. The tape p

Related reference

The device table in the onpload database

Adding, editing, and removing devices

You can add devices to the device array, you can edit the devices, and you can remove them.

About this task

Adding devices to the device array

About this task

To add devices to the device array:

1. Click **Add** in the **Device-Array Definition** window.
2. Click the device type in the **Array Item Type** group.
3. Type the full path name of the device in the **Device** text box.

If you specified a tape device, the **Tape Parameters** group becomes active, as [Figure 50: The tape parameters group on page 83](#) shows.

- a. Type the block size in kilobytes.
 - b. Click **MB** (megabytes) or **GB** (gigabytes) to specify the units to use for the tape size.
 - c. Specify the tape size.
4. When you have included all of the information for the device, click **Perform**.
The device that you added appears in the **Array Items** list box.

5. Repeat steps [2 on page 83](#) through [4 on page 83](#) to add other items to the device array.
6. When you have added all of the devices to the array, click **OK** to return to the **Device-Array Selection** window.
Your new array appears in the **Current Arrays** list box.
7. Click **Cancel** to return to the HPL main window.

Editing a device in the device array

About this task

To edit a device in the device array:

1. Click **Edit** in the **Device-Array Definition** window.
2. Select a device from the **Array Items** list box.
The selected item appears in the **Device** text box.
3. Edit the path name and tape parameters, as appropriate.
4. Click **Perform**.
5. Click **OK** to return to the **Device-Array Selection** window.
6. Click **Cancel** to return to the HPL main window.

Results



Tip: When you edit a device, you can change the path name and the tape parameters, but you cannot change the array-item type (tape, file, pipe). If you need to change the device type, you must delete the item and then add a new item.

Deleting a device from the device array

About this task

To delete a device from the device array:

1. Click **Delete** in the **Device-Array Definition** window.
2. Select a device from the **Array Items** list box.
3. Click **Perform**.
4. Click **OK** to return to the **Device-Array Selection** window.
5. Click **Cancel** to return to the HPL main window.

Define formats

This section describes the formats that the High-Performance Loader (HPL) provides and shows how to prepare and edit the format component.

Related information

[Components of the unload job on page 142](#)

Formats

A format describes the structure of the data in a data file.

Before you can import records from a data file into HCL Informix® databases or export records from a database to a data file, you must define a format that describes the data file. You do not need to define a format for the database because `ipload` already knows the schema (the organization) of the database table.

These topics use the term *format* in two ways:

- To refer to the arrangement of data fields in a record of a data file
- To refer to the HPL component that documents the arrangement of the data fields

After you familiarize yourself with the concepts in this section, you might save yourself some work by using one of the Generate options to create formats automatically.

Related reference

[The Generate options of the `ipload` utility on page 160](#)

Formats of supported datafile records

Data files can be structured in various ways. The High-Performance Loader (HPL) supports data-file records of the following formats:

- Fixed-length
- Delimited
- COBOL
- Other formats

You can define new format components at any time. Also, you can test your format before you actually load or unload data.

The `ipload` utility includes options that let you modify the data before it is inserted into the database.

The `ipload` utility stores information about formats in the **formatitem** and **format** tables of the onpload database.



Important: To prepare the format component, you must know the format of the records in the data file. If you do not know the data-file format, you must get it from the person who provided the data file.

Related reference

[Format options on page 103](#)

The formatitem table in the onpload database

The formats table in the onpload database

[Define formats on page 223](#)

Related information

[Preview data-file records on page 170](#)

Fixed-length records

In fixed-length or fixed-format records, each field starts and ends at the same place in every record. A data file that contains data records of equal and constant length might be organized as follows.

Figure 51. Sample file with fixed-length records

```
aaabbbbccdddggghhhh
```

The data file illustrated previously has three records. Each record has a field of three characters followed by a field of four characters, so the total record length is seven characters. The file does not contain any separation between records; delimiters are unnecessary because all fields have the same length. The VARCHAR data types are therefore always the fixed maximum size of the field.

When you define a fixed-length format, you specify the length of each field. The ipload utility calculates the offset for each field and the total length of the record from the field lengths that you supply.

Creating a fixed format

You can create and define formats for fixed-length records from the **Record Formats** and the **Fixed Format Definition** windows.

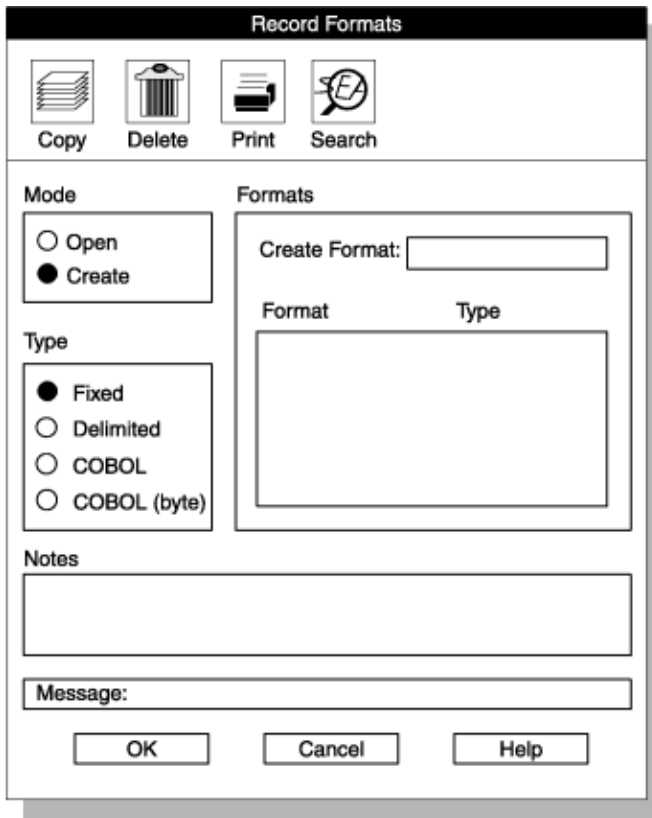
About this task

To create a format for fixed-length records:

1. Choose **Components > Formats** from the HPL main window.

The **Record Formats** window appears, as the following figure shows.

Figure 52. The Record Formats window



2. Click **Create** in the **Mode** group.
3. Click **Fixed** in the **Type** group.
4. Choose a name for the format and type it in the **Create Format** text box.
5. Click **OK**.

The **Fixed Format** definition window appears. The title bar includes the name that you chose for the format. The following figure shows the **Fixed Format Definition** window as it might appear after you prepare the format for the file that [Figure 51: Sample file with fixed-length records on page 86](#) shows.

Figure 53. A completed Fixed-Format Definition window with an open selection list

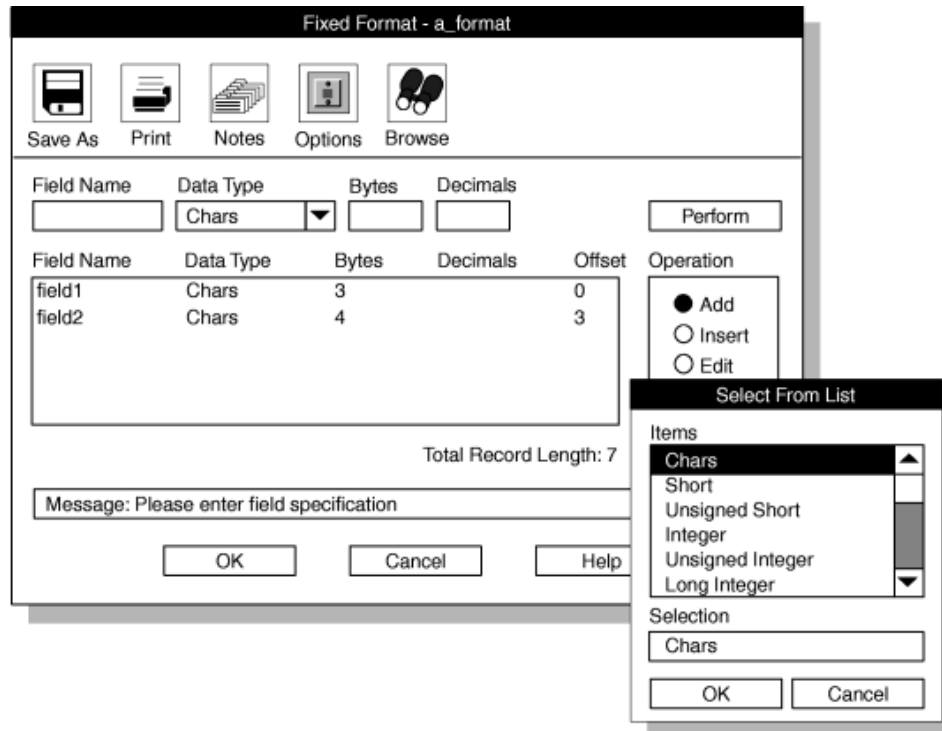


Table 13. The Fixed-Format Definition window options

Di
sp
lay
o
pt
ion

Description

By The **Bytes** text box specifies the number of characters that the field occupies in the record. In the **Bytes** text box, you must set the number of bytes for your data types. Although ipload uses default information to calculate an offset if you create a format that has a new length, it does not readjust the lengths for existing formats. To change the default information, see [The Machines window on page 77](#). The ipload utility automatically calculates the total length of the data file as you add each field description.

De The **Decimals** text box specifies the number of decimal places that are displayed when you convert floating-point types to ASCII. You can set the number of decimals only for the **Float** and **Double** data types.

6. Click **Add** in the **Operation** group.
7. Choose a name for the field and type the name in the **Field Name** text box.
8. Type the data type in the **Data Type** text box.

For columns that contain user-defined types (UDTs) columns, you must choose an Ext Type format. For information about user-defined types, see *HCL® Informix® Guide to SQL: Reference*.

The down arrow next to the **Data Type** text box displays the selection list that appears at the right in [Figure 53: A completed Fixed-Format Definition window with an open selection list on page 88](#). [Data types allowed in a fixed format on page 89](#) describes the data types that appear in the selection list.

9. Type the appropriate value in the **Bytes** text box (or in the **Decimals** text box, if appropriate).
If you chose the Ext Type String in the Data Type box, you must specify a size, or you get an error. Field size depends on the type of data and its representation in the data file.
10. Click **Perform**.

After you click **Perform**, ipload calculates the proper offset for this field in the record and displays the value under the Offset heading, as [Figure 53: A completed Fixed-Format Definition window with an open selection list on page 88](#) shows.
11. Repeat steps [7 on page 88](#) through [10 on page 89](#) for each field in your data file.
12. Click **OK** to save the format and return to the **Record Formats** window.
13. Click **Cancel** to return to the HPL main window.

Results



Tip: Use the field name to map the data file to the database. You can type any name that you choose. You might find it easier to remember the names if you use the same name as the corresponding column of the database.

Related reference

The formats table in the onpload database

Data types allowed in a fixed format

You can use the following data types when you are preparing a fixed format.

Data type	Description
Chars	ASCII format data
Short Unsigned Short Integer Unsigned Integer Long Integer Unsigned Long Float Double	The Machines description specifies the number of bytes required in fixed format for integers and floating-point values (see The Machines window on page 77 .) When you select one of these data types, iploadsets the number of bytes.
Date	Date string
UNIX™ Date	A long integer interpreted as the system date from a UNIX™ system

Data type	Description
Blob Length	The number of bytes of BYTE and TEXT (binary large object) that follow this record
Blob File	A file that contains BYTE and TEXT data
BLOB and CLOB	A file that contains smart large object (BLOB or CLOB) data
Simple LO Length	The number of bytes of simple large object data (BYTE and TEXT data) that follow this record
Simple LO File	The name of a file that contains a Simple LO data (BYTE and TEXT data)
INT8	An eight-byte integer type
SERIAL8	An eight-byte serial column
BIGINT	An ASCII character string or an eight-byte integer type
BIGSERIAL	An ASCII character string or an eight-byte integer type
Ext Type String	The ASCII representation of an extended data type (Ext Type) value
Ext Type String Length	The length of an ASCII Ext Type value. The Ext Type value follows at the end of the input/output record. Use Ext Type String Length data type if you have data that contains null UDT values.
Ext Type Binary	The binary representation of an Ext Type value
Ext Type Binary Length	The length of the binary representation of an Ext Type value. The binary value follows at the end of the input/output record. Use Ext Type Binary Length data type if you have data that contains null UDT values.

The HPL supports several data types under the Ext Type mechanism. As a result, the specific names of these data types do not appear in the data-type selection list. For the following data types, choose the appropriate Ext Type data type:

- INT8
- LVARCHAR
- SERIAL8
- BLOB
- BOOLEAN
- CLOB
- Collection data types
- Distinct data type
- Opaque data types
- Row types

Editing a format

After you create and save a format, you might need to add a new field, insert a new field, edit a field, or delete a field. The process for editing an existing format is essentially the same, regardless of the file type.

About this task

The following example uses a fixed-format file, but the same procedure applies to COBOL and delimited files also.

Related information

[Preview data-file records on page 170](#)

[Editing a format on page 172](#)

Adding a new field description to the format

About this task

To add a new field description to the format:

1. Open the **Fixed Format Definition** window.
For more information, see [Creating a fixed format on page 86](#).
2. Click **Add** in the **Operation** group.
3. Type the field specifications in the text boxes at the top of the window.
4. Click **Perform** to add the new field at the end of the list.
5. Click **OK** to save your changes and return to the **Record Formats** window.
6. Click **Cancel** to return to the HPL main window.

Inserting a new field into the format

About this task

To insert a new field into the format:

1. Open the **Fixed Format Definition** window.
For more information, see [Creating a fixed format on page 86](#).
2. Click **Insert** in the **Operation** group.
3. Select the field before which you want to insert the new field.
4. Type the field specifications in the text boxes at the top of the window.
5. Click **Perform** to insert the new field before the selected field.
6. Click **OK** to save your changes and return to the **Record Formats** window.
7. Click **Cancel** to return to the HPL main window.

Editing the description of a field

About this task

To edit the description of a field:

1. Open the **Fixed Format Definition** window.
For more information, see [Creating a fixed format on page 86](#).
2. Click **Edit** in the **Operation** group.
3. Select the field from the list of fields.
4. Change the information that you want.
5. Click **Perform**.
6. Click **OK** to save your changes and return to the **Record Formats** window.
7. Click **Cancel** to return to the HPL main window.

Deleting a field description from the format

About this task

To delete a field description from the format:

1. Open the **Fixed Format Definition** window.
For more information, see [Creating a fixed format on page 86](#).
2. Click **Delete** in the **Operation** group.
3. Select the field that you want to delete.
4. Click **Perform** to delete the field.
5. Click **OK** to save your changes and return to the **Record Formats** window.
6. Click **Cancel** to return to the HPL main window.

Create a fixed format that uses carriage returns

A fixed-format data file often includes a carriage return (new line) at the end of each record, such as the data file in the following example file.

20 chars	20 chars	15 chars2	chars
John Brown	100 Main St.	Citadel	LA
Mary Smith	3141Temple	WayChesapeake	AZ
Larry Little	44 Elm Rd. #6	Boston	MA

When you prepare the format for this data file, you must include a dummy field for the carriage return. When you create the load map for this format, do not link the dummy field to a database column. The following figure shows the format for the data file illustrated previously.

Figure 54. Fixed format with dummy entry for carriage return

Fixed Format - a_format

Save As Print Notes Options Browse

Field Name	Data Type	Bytes	Decimals
	Chars		

Perform

Field Name	Data Type	Bytes	Decimals	Offset	Operation
name	Chars	20		0	<input checked="" type="radio"/> Add
street	Chars	20		20	<input type="radio"/> Insert
city	Chars	15		40	<input type="radio"/> Edit
state	Chars	2		55	<input type="radio"/> Delete
dummyCR	Chars	1		57	

Total Record Length: 58

Message: Please enter field specification

OK Cancel Help

Related information

[Creating a load map on page 122](#)

Create a fixed format that includes BYTE or TEXT data

You can organize the byte or text data in a fixed-format data file with inline data or data in a separate file.

Inline data

BYTE or TEXT data that is included as part of a fixed-format data file is called inline data. When byte or text data is inline, the data-file record has two parts: a fixed-length part and a variable-length part.

For example, a record with two fields and byte or text data might be organized as follows.

Figure 55. Organization of a record that includes inline TEXT data

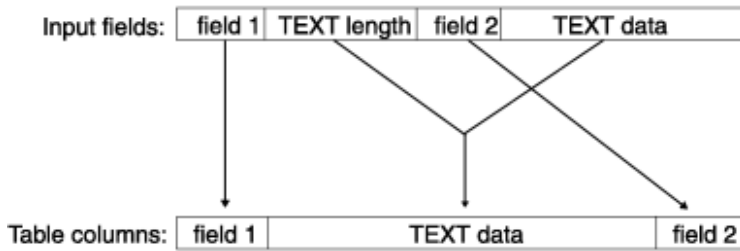
```
field1textlengthfield2textdata
```

The length of the TEXT data is included in the fixed-length part of the record. The actual TEXT data is inserted at the end of the fixed-length part of the record. The High-Performance Loader (HPL) reads the TEXT length from the fixed-length part of the record and uses that length to read the actual TEXT data. The HPL also uses the TEXT length to calculate the offset to the beginning of the next record.

The following figure shows the format definition of a record with inline BYTE and TEXT data. The arrows show how the HPL puts the record into the database. The arrows from **field 1** and **field 2** indicate entries in fixed-length format. The split arrow

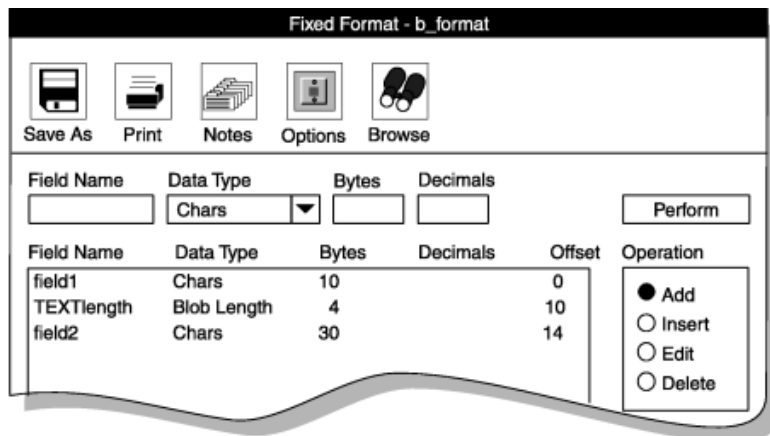
shows that the HPL uses the **TEXT length** information to find the **TEXT data** and insert it into the table. The HPL does not insert the **TEXT length** into the database.

Figure 56. Inline TEXT Data



When you define the format in the **Format-Definition** window, select **Blob Length** as the data type for the **textlength** field. The following figure shows the format for the example in [Figure 55: Organization of a record that includes inline TEXT data on page 93](#). The format does not include an entry for TEXT data.

Figure 57. Fixed format that includes TEXT data



When you create a map to link the input fields that are defined by the format to the columns of a database table, connect the **textlength** input field to the table column that contains the TEXT data.

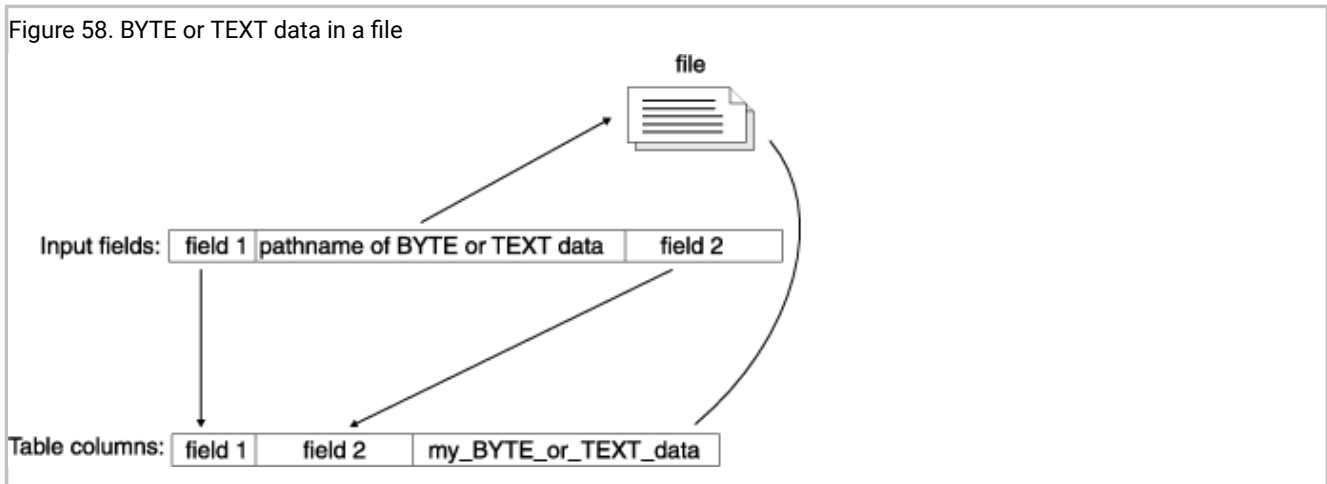
Related reference

[Define maps on page 119](#)

Data in a separate file

You can also store BYTE and TEXT data in separate files.

During a load, BYTE and TEXT data files are read and inserted into the database. During an unload job, the file is created, and BYTE and TEXT data is written to the file. When the fixed-format input contains the path name of a data file, the HPL uses that path name to insert data into a column of the database table, as the following figure shows. When you prepare the format, select **Blob File** for the data type.



When you create a map to link the fields of the input record to the columns of a database table, link the name of the BYTE or TEXT file with the BYTE or TEXT column. The arrows in [Figure 58: BYTE or TEXT data in a file on page 95](#) illustrate how the HPL inserts the BYTE or TEXT data into the column.

Related reference

[Define maps on page 119](#)

Create a fixed format that includes Ext type or Simple LO data

You can organize the Ext Type or Simple LO data in a fixed-format data file with fixed-length data or inline data.

Fixed-length data

When you designate a field as an Ext Type String or Ext Type Binary data type, you specify that the data is going occupy a fixed amount of space, similar to the behavior of a fixed-length Chars data type. With fixed-length data format, you must specify the number of bytes that the field occupies in the record.

Inline data

Simple LO data or varying-sized Ext Type data that is included as part of a fixed-format data file is called inline data. When Ext Type or Simple LO data is inline, the data-file record has two parts: a fixed-length part and a variable-length part.

For example, a record with two fields and a Simple LO might be organized as follows.

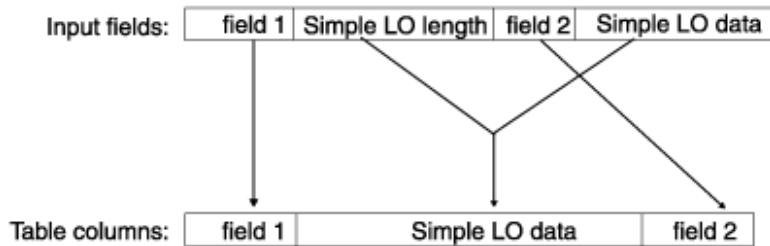
```
field1Simple LO lengthfield1Simple LO data
```

The data-type length of the Ext Type or Simple LO data is included in the fixed-length part of the record. The actual TEXT data is inserted at the end of the fixed-length part of the record. The HPL reads the Ext Type or Simple LO length from the fixed-length part of the record and uses that length to read the actual Ext Type or Simple LO data. The HPL also uses the Ext Type or Simple LO length to calculate the offset to the beginning of the next record.

The following figure shows the format definition of a record with inline Ext Type or Simple LO data. The arrows show how the HPL puts the record into the database. The arrows from **field 1** and **field 2** indicate entries in fixed-length format. The split

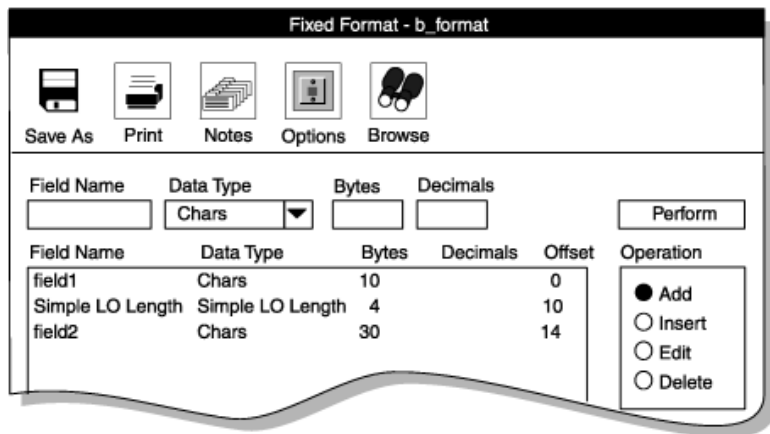
arrow shows that the HPL uses the **Simple LO length** information to find the Simple LO data and insert it into the table. The HPL does not insert the Simple LO length into the database.

Figure 59. Inline TEXT data



When you define the format in the format-definition window, select the appropriate data-type length data type (**Ext Type String Length**, **Ext Type Binary Length**, or **Simple LO Length**) for the data-type length field. The following figure shows the format for the example previously illustrated. The format does not include an entry for Simple LO data.

Figure 60. Fixed format that includes a Simple LO



! Important: Ext Type binary-length format is not supported for complex types.

When you create a map to link the input fields that are defined by the format to the columns of a database table, connect the data type length input field to the table column that contains that particular data. In this case, connect the **Simple LO length** input field to the table column that contains the Simple LO data.

Related reference

[Define maps on page 119](#)

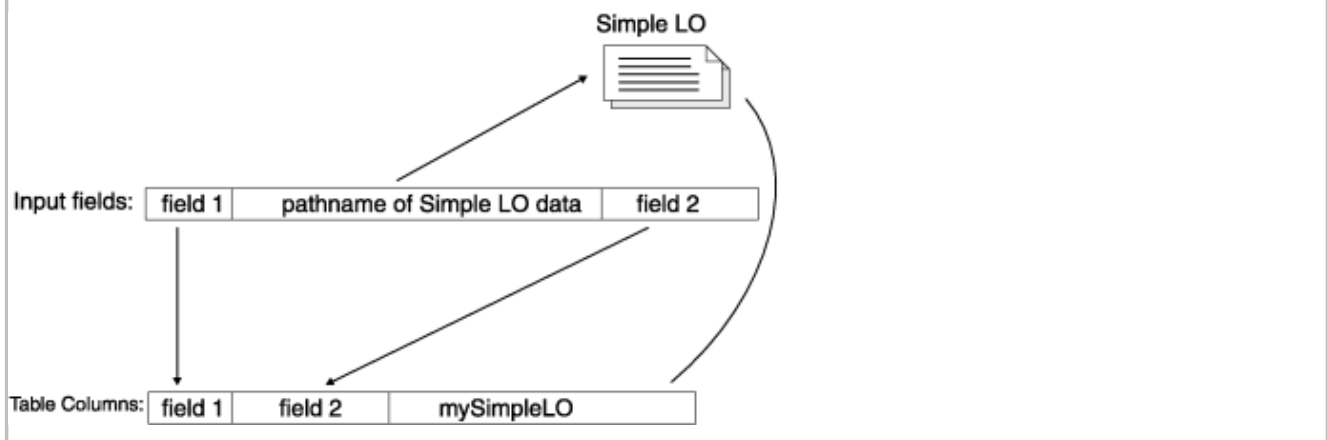
Simple LO data in a separate file

You can also store Simple LO data in separate files.

During a load, Simple LO data files are read and inserted into the database. During an unload job, the file is created and BYTE and TEXT data is written to the file. When the fixed-format input contains the path name of a data file, the High-Performance

Loader (HPL) uses that path name to insert data into a column of the database table, as the following figure shows. When you prepare the format, select **Blob File** for the data type.

Figure 61. Simple LO data in a file



When you create a map to link the fields of the input record to the columns of a database table, link the name of the Simple LO file with the Simple LO column. The arrows in [Figure 61: Simple LO data in a file on page 97](#) illustrate how the HPL inserts the Simple LO data into the column.

Related reference

[Define maps on page 119](#)

Delimited records

Delimited records are records whose fields can vary in length. In a data file that contains delimited records, the records and fields are separated by a delimiter.

The following data file uses a vertical bar (|) as the field delimiter and a carriage return as the record delimiter:

```
John Brown|100 Main St.|Citadel|LA|215/887-1931
Mary Smith|3141 Temple Way|Chesapeake|AZ|415/812-9919
Larry Little|44 Elm Rd. # 6|Boston|MA|617/184-1231
```

The ipload utility uses the vertical bar and carriage return as the default field and record delimiters.

Related information

[Modifying delimited-format options on page 104](#)

Create a delimited format

To create a format for delimited records, follow the same steps as in [Creating a fixed format on page 86](#), with the following modifications:

- In step 3 on page 87, click **Delimited** in the **Type** group.
- Omit step 9 on page 89. When you use delimited records, ipload does not need byte or decimal information.

Data types allowed in a delimited format

You can use the following data types when you prepare a delimited format.

Table 14. Data types allowed in delimited format

Data type	Description
Chars	Normal ASCII data
BYTE or TEXT File	File that contains BYTE or TEXT data
TEXT Data	TEXT data is formatted as ASCII text. If the text includes carriage returns (new lines) or delimiters, a backslash (\) must precede those characters.
BYTE or TEXT HexASCII	BYTE or TEXT data that is formatted in ASCII hexadecimal. The onpload utility translates the data into binary format before it loads the data into the database.
BLOB or CLOB	File that contains smart large object (BLOB or CLOB) data
Simple LO File	The name of a file that contains Simple LO data (BYTE and TEXT data)
Simple LO Text	Simple LO data that is formatted as ASCII text. If the text includes carriage returns, newline characters, or delimiters, a backslash (\) must precede those characters
Simple LO HexASCII	Simple LO data that is formatted in ASCII hexadecimal. The onpload utility translates the data into binary format before it loads the data into the database.
Ext Type String	The ASCII representation of a Data Type (Ext Type) value
Ext Type HexASCII	The HexASCII representation of an Ext Type value

Create a delimited format that includes BYTE or TEXT data

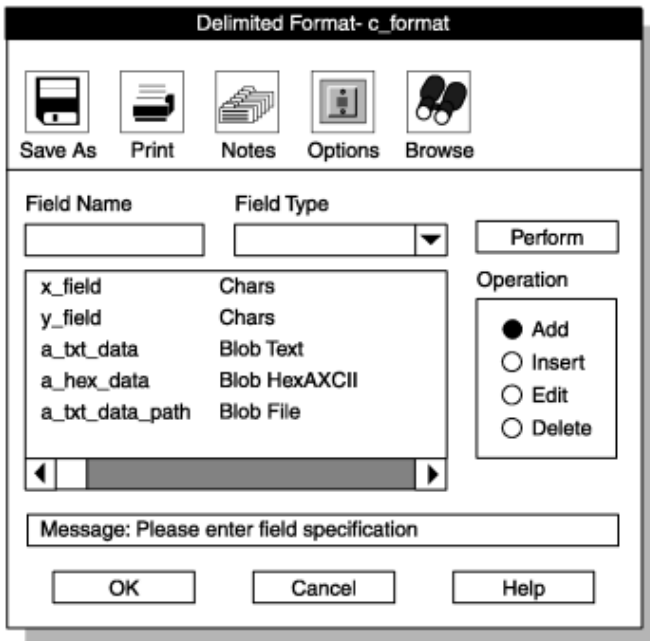
In a delimited format, BYTE or TEXT data can be characters, hexadecimal data, or a separate file.

The following sample data-file record shows a data record that has two fields of character data followed by a field of character BYTE or TEXT data, a field of hexadecimal byte or text data, and the path name of a file that contains BYTE or TEXT data.

```
field1|field2|TEXT data|BEEEF6699|/bbs/kaths/data2jn95
```

The following figure shows a format for the sample data-file record previously illustrated.

Figure 62. Delimited format with BYTE or TEXT entries



Create a delimited format with extended data types

Extended data types include the following data types:

- User defined (distinct and opaque)
- Collection
- CLOB and BLOB
- Row

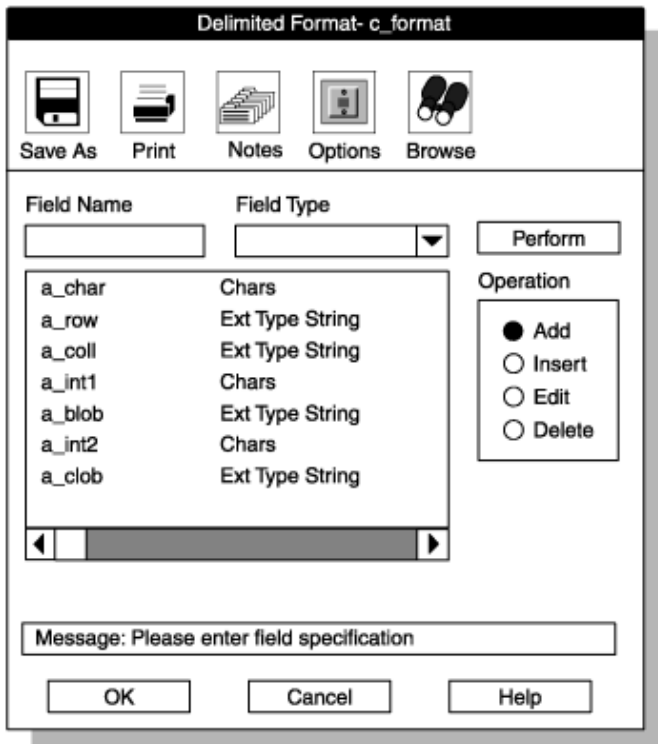
In a delimited format, CLOB and BLOB data is always written to a file. CLOB data can be ASCII or hexadecimal data. BLOB data can be binary data. The path name of the file, the file size, and the offset are embedded in the data file during unload. However, when you perform a load, you only need to specify the path name.

The following sample data-file record shows a data record that has a field of character data (field1), a row-type field (ROW('abcd', NULL)), a collection-type field (SET{1}), an integer field (10), a BLOB-type field (/work/data/photo.jpg), an integer field (20), and a CLOB-type field (work/data/text.txt).

```
field1|ROW('abcd', NULL|SET{1}|10|/work/photo.jpg|20|
/work/text.txt
```

The following figure shows a sample format for the sample data-file record that was previously illustrated.

Figure 63. Delimited format with extended data type entries



If you unload by using a format that is similar to [Figure 63: Delimited format with extended data type entries on page 100](#), the unloaded data might resemble the following sample data-file record. You can use this data to load the same file again, instead of using the path name.

Figure 64. Sample data-file record that includes extended data types

```
field1|ROW('abcd', NULL|SET{1 }|10|0,51f4,blob67e9.8ac|20|
0,c 692,clob67e9.9ad
```

By default, the clob67e9.8ad and blob67e9.8ac files in [Figure 64: Sample data-file record that includes extended data types on page 100](#) are written to /tmp. To change the default, modify the path in the **PLOAD_LO_PATH** environment variable.

COBOL records

The HPL supports COBOL sequential data files that do not contain internal indexing. The following figure shows the **COBOL-Format Definition** window for preparing a COBOL format.

Figure 65. Fixed-format definition window for a COBOL format

COBOL Format - see_cobol

Save As Print Notes Options Browse

Field Name Picture Usage

Perform

Name Picture Usage Operation

Add
 Insert
 Edit
 Delete

Record Length: 0

Message: Please enter field specification

OK Cancel Help

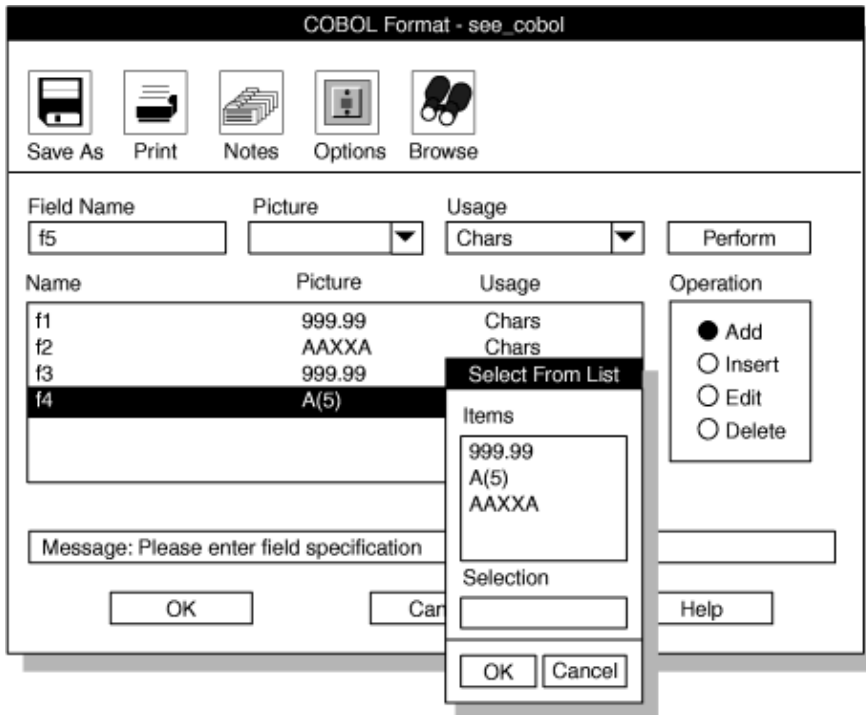
Create a COBOL format

To create a format for COBOL records, follow the same steps as in [Creating a fixed format on page 86](#), with the following modifications:

- In step [3 on page 87](#), click **COBOL** in the **Type** group in the Record Formats window.
- In step [8 on page 88](#), type the COBOL picture description in the **Picture** text box.
- In step [9 on page 89](#), type the data type in the **Usage** text box.

- The arrow displays the selection list of available data types for the **Usage** text box.

Figure 66. Fixed-format definition window for a COBOL format



! **Important:** Ext types are not supported in COBOL format.

Picture and usage descriptions

The picture and usage description must conform to ANSI-COBOL 85 specifications.

For information about COBOL picture strings, see the documentation for the COBOL compiler.

Picture description

The picture description must match the record file descriptor (FD) from the COBOL program that generates or use the data. For information about COBOL formats, see your COBOL programmer's publication.

Usage description

The usage description must match the data-field type described in the FD descriptor of the COBOL program. If the COBOL program does not include a usage clause, select the Chars (character) option for the usage.

Packed-decimal conversions

When values are converted to packed-decimal formats, supply a picture clause that matches the picture clauses in the COBOL programs that use the data. Otherwise, the COBOL interpretation of the values is wrong.

The following table lists some examples of appropriate picture clauses.

Picture	Input data	Output data	COBOL value =
9999999	123	0000123C	123
9999V99	123	0000123C	1.23
9999.99	123	0000123C	1.23
9999V99	-123.22	0012322D	-123.22

Other formats

In addition to delimited, fixed, and COBOL formats, the High-Performance Loader (HPL) provides two other formats for loading and unloading data: fast format and fast job. These formats are not included on the **Record Formats** window because the format specifications are predefined; you do not need to make any choices.

Fast format and fast job are the most efficient ways to load and unload data because their formats are predefined.

Fast format

Fast format loads or unloads data in which each individual column uses HCL Informix® internal format. You can reorder, add, or delete columns, but you cannot conduct conversion on the column itself.

Select **Fixed internal** in the **Generate** window to get this type of load. For information about the **Generate** window, see [The Generate options of the ipload utility on page 160](#).

Fast job

A fast job loads or unloads an entire row of the HCL Informix® database table in Informix® internal format. A fast job is also called a raw load or a no conversion job. For more information, see [Format type group on page 166](#).

The **-fn** flag of the onpload command-line utility specifies a fast job. For information about the onpload utility, see [The onpload utility on page 189](#).

Format options

The format options let you change the default driver, the character set, the default computer type, and the delimiters. Information about the format options is stored in the **formats** table of the onpload database.

Related reference

The formats table in the onpload database

[Format type group on page 166](#)

Related information

[Formats of supported datafile records on page 85](#)

Modifying fixed and COBOL formats

About this task

You can modify the following options for fixed and COBOL formats.

Character set

The code set that is used to translate the data in the data table

Driver

The driver that is used with the delimited format. For more information, see [Selecting a driver on page 75](#).

Machine

The machine type that produced the data files. For more information, see [Modify the machine description on page 76](#).

For a fixed format, you can select the GLS code set you want from the **Character Set** selection list. For information about locales and code sets, see the *HCL® Informix® GLS User's Guide*.

To modify the options for fixed and COBOL formats:

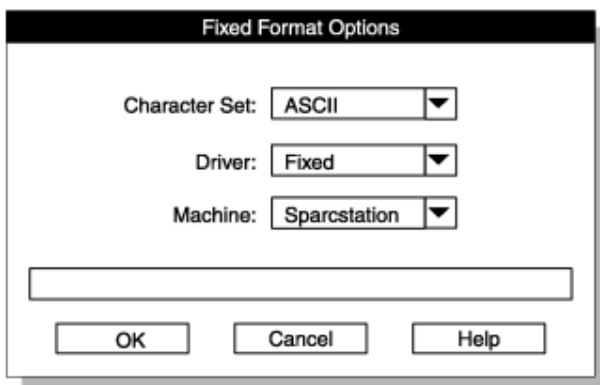
1. Display the **Format-Definition** window for the format that you want.

To display the window, follow the steps in [Creating a fixed format on page 86](#).

2. Click **Options**.

The **Options** window (in this example, the **Fixed Format Options** window) appears, as following figure shows.

Figure 67. The Fixed Format Options window



3. Modify the options as appropriate.
4. Click **OK** to save your options and return to the **Format-Definition** window.

Modifying delimited-format options

About this task

You can modify the following options of the delimited format from the **Delimiter Options** window.

Character set

The code set used to translate the data in the data table.

Driver

The driver that is used with the delimited format. For more information, see [Selecting a driver on page 75](#).

Delimiting characters

The delimiting characters, which are sometimes called record separators and field separators, indicate the beginning and end of records and fields.

You can specify the delimiting characters in ASCII, HEX, OCTAL, or DECIMAL format.

You can select the GLS code set you want from the **Character Set** selection list. For information about locales and code sets, see the *HCL® Informix® GLS User's Guide*.

To modify the options for delimited formats:

1. Display the **Delimited Format Definition** window.

To display the window, complete the steps in [Creating a fixed format on page 86](#) with the following modification: in step [3 on page 87](#), click **Delimited**.

2. Click the **Options** button in the **Delimited Format Definition** window.

The **Delimiter Options** window appears, as the following figure shows.

Figure 68. The Delimiter Options window

HEX	OCTAL	CONTROL	ASCII
00	000	Null	Null
01	001	Ctrl-A	sch
02	002	Ctrl-B	stx

3. Modify the options that you want to change, or add information.

For example, add **Field Start** and **Field End** information to mark the beginning and end of a field (column).

The default escape character is \ (the backslash).


If the **Field Start** and **Field End** values contain both single or double quotation marks and specified characters are not present in a field, an error occurs. The error occurs because the input does not match the expected layout. If one of these characters appears in a field, you must mark the character with the escape key (to escape the character).

For comma-separated values (CSV), enter , (a comma) in the **Field Separator** field. For more information, see [Testing the import of a CSV file on page 106](#).

An error can also occur if the **Field Separator** value appears inside a quoted field or any special character (such as the end field, end record, or escape character) appears in a field. In this case, you must also escape the character.

4. Click **OK** to save your changes and return to the **Delimited Format Definition** window.

Results

 **Tip:** You can use the **DBDELIMITER** environment variable to set the field delimiter for the dbexport utility and the LOAD and UNLOAD statements. However, do not use **DBDELIMITER** with the HPL because the onpload utility does not use this environment variable.

For more information about environment variables, see the *HCL® Informix® Guide to SQL: Reference*.

Related reference

The delimiters table in the onpload database

Related information

[Delimited records on page 97](#)

Testing the import of a CSV file

This procedure shows how to test importing sample data from a spreadsheet into HCL Informix®.

About this task

Files containing comma-separated values (CSV) are a commonly used for transferring simple text data between programs. The CSV format uses a comma as a field separator and a new line as a record separator. Double quotation marks are used to embed commas, new lines, or double quotation marks within strings. Some applications (for example, some spreadsheet applications) provide the option of exporting text data in the CSV format.

To test importing a CSV file:

1. Create a test database and simple table in Informix®.
2. In your spreadsheet application:
 - a. Create a spreadsheet with columns that correspond to the columns in your Informix® table.
 - b. Populate the columns with some test data.
 - c. Export the data by saving it in .csv format.

3. Copy the resulting `.csv` file and paste it in a location that you can access from ipload.
4. Start ipload and choose **Components > Generate Job** to create a job.
5. In the **Generate Job** window:
 - a. Select **Load/Unload Job** in the **Generate** group.
 - b. Select **Delimited** in the **Format Type** group.
 - c. Specify the `.csv` file as the source for input to the new job you are creating. Specify a name in the **Generate Name** field, select your database in the **Database** field, select your table in the **Table** field, and then insert the full path to the `.csv` file in the **Device** field.
 - d. Click **OK**.
6. Load the job by choosing **Jobs > Load**.
7. While you can do many things (such as filtering and mapping) on the **Load Job** window, for this test, select the **Format** button.

The **Delimited Format** window appears. You use this window to define the format of the input file.

8. In the **Delimited Format** window, click the **Options** button to display the **Delimited Options** window.
9. In the **Delimiter Options** window, change the value in the **Field Separator** field to `,` (a comma) and click **OK**.

The ipload utility returns you to the **Delimited Format** window.

10. In the **Delimited Format** window, click **OK**.

The ipload utility returns you to the **Load Job** window.

11. Click **Run** to run the job.

The ipload utility displays status information while the job runs, and then it displays the results of the job.

12. Click **OK** twice to return to the main ipload window.

Results

As an evaluation, you can use DBAccess to verify that the data was successfully loaded.

For more information, see [Generating load and unload components on page 167](#).

Format Views window

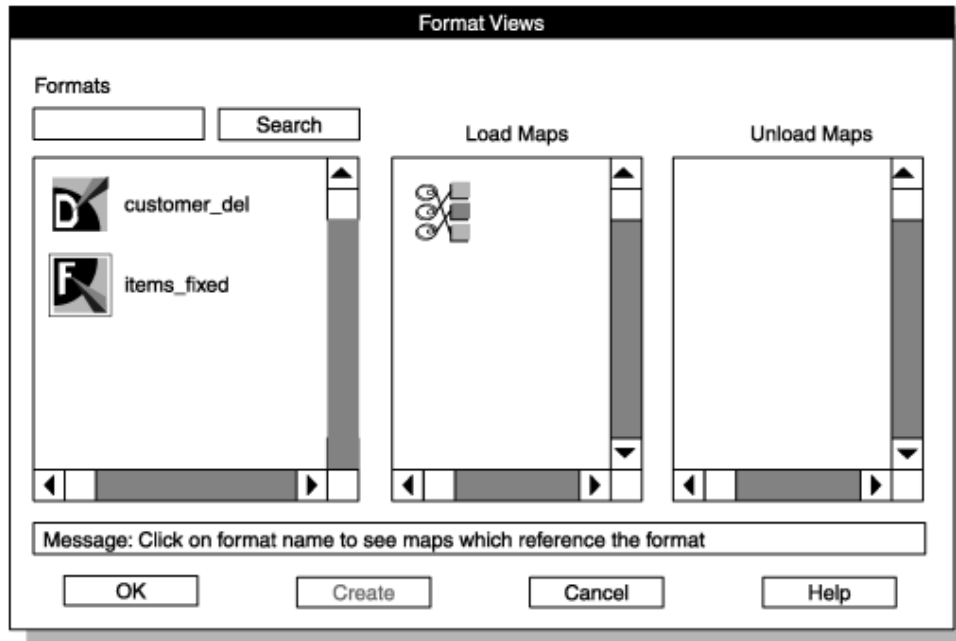
You can display a list of the formats and load and unload maps that are associated with a project from the **Format Views** window. You can also create or edit a format.

The **Format Views** window appears in the following situations:

- When you click **Format** in the **Unload Job** window and no format name is in the **Formats** text box
- When you click **Search** in the **Query** window

The following figure shows a **Format Views** window. [Views windows on page 53](#) discusses how to use **Views** windows.

Figure 69. The Format-Views window



Define queries

This section describe how to define queries including how to create, edit, and export and import queries.

Related information

[Components of the unload job on page 142](#)

HPL queries

You can build an SQL statement with the ipload query component.

The ipload utility stores query information in the **query** table of the onpload database. The SQL statement is stored as TEXT data.

During the unload process, the High-Performance Loader (HPL) uses a query to select data from a database table (or tables), as the following figure shows. The HPL can process any valid SQL statement.

Figure 70. Extracting data from a database table



These topics use the term *query* in two ways:

- To refer to the SQL statement that selects information from the database
- To refer to the HPL component that lets you build and store the SQL statement

Related reference

The query table in the onpload database

Creating a query

Use the **Query** window to create a query.

About this task

To create a query:

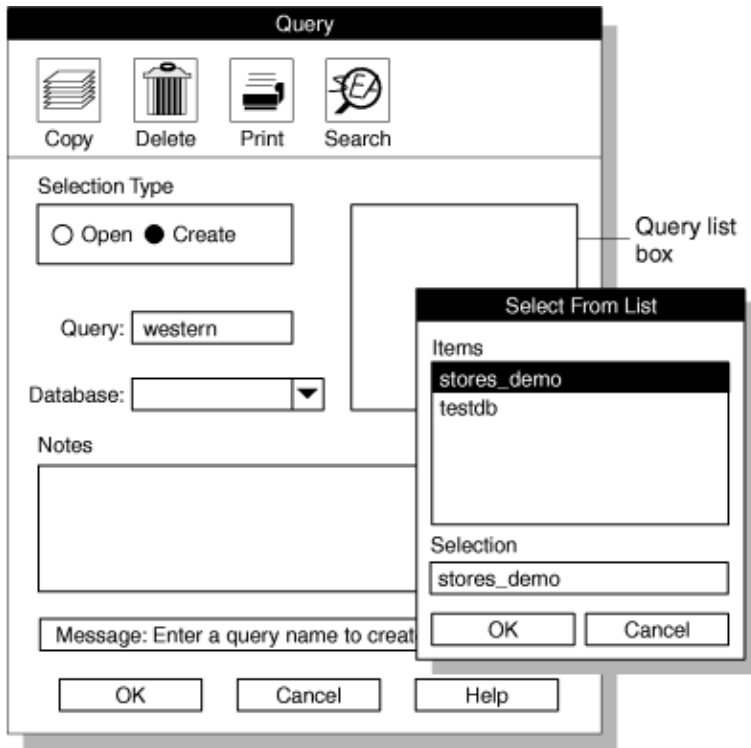
1. Choose **Components > Query** from the HPL main window.

The **Query** window appears, as [Figure 71: The Query window on page 110](#) shows.

2. Click **Create** in the **Selection Type** group.
3. Choose a name for your query and type it in the **Query** text box.
4. In the **Database** text box, type the name of the database that contains the tables from which you want to extract data.
Or, click the down arrow to select from a database selection list.

The following figure shows the **Query** window with the **Query** text box completed and **stores_demo** selected from the selection list.

Figure 71. The Query window



5. Click **OK**.

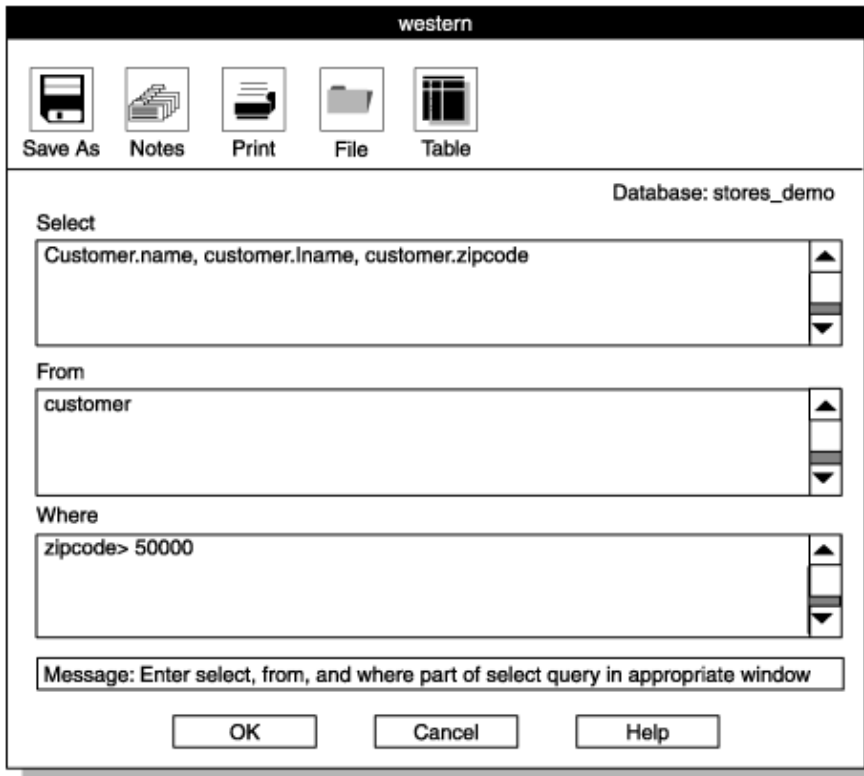
The **Query-Definition** window appears, as [Figure 72: The Query-Definition window on page 111](#) shows. The name that you chose for your query appears in the title bar.

6. Type your query in the **Select, From,** and **Where** text boxes.

The following figure shows the following simple query of the **customer** table of the **stores_demo** database:

```
SELECT customer.fname, customer.lname,
customer.zipcode
FROM customer
WHERE zipcode > 50000
```

Figure 72. The Query-Definition window



If you prefer, you can type the entire query into the **Select** text box. If you later edit the query, ipload divides the query into SELECT, FROM, and WHERE clauses.

7. Click **OK** to save the query and return to the **Query** window.

The query that you just created now appears in the **Query** list box at the right side of the **Query** window.

8. Click **Cancel** to return to the HPL main window.

Related reference

The query table in the onpload database

Using the Column Selection window

The **Table** button displays the **Column Selection** window. You can use the **Column Selection** window to build queries by selecting tables and columns. The ipload utility inserts the selected columns and tables into the appropriate text boxes of the **Query-Definition** window.

About this task

To use the **Column Selection** window:

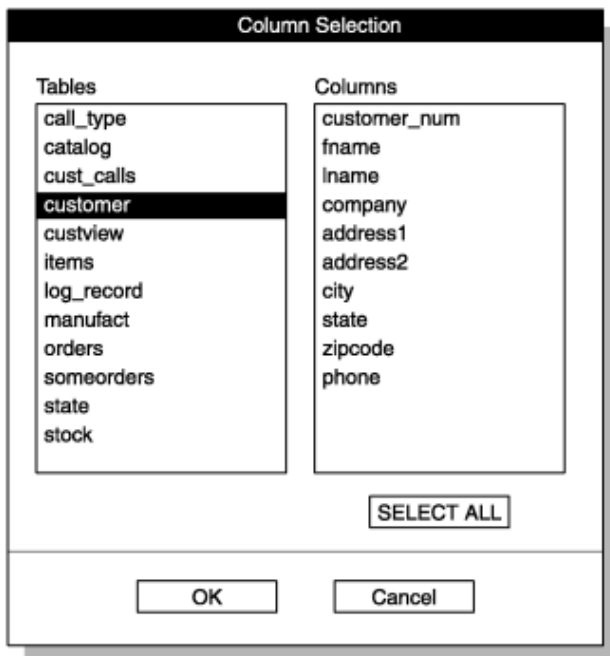
1. Follow the steps in [Creating a query on page 109](#) to display the **Query-Definition** window.
2. Click **Table**.

The **Column Selection** window appears, as [Figure 73: The Column Selection window after selecting a table on page 112](#) shows. The **Tables** list box includes synonyms and views that are valid for the local database server.

3. Select a table.

After you select a table, the right pane displays a list of the columns in that table. The following query shows the **Column Selection** window with the **customer** table selected.

Figure 73. The Column Selection window after selecting a table



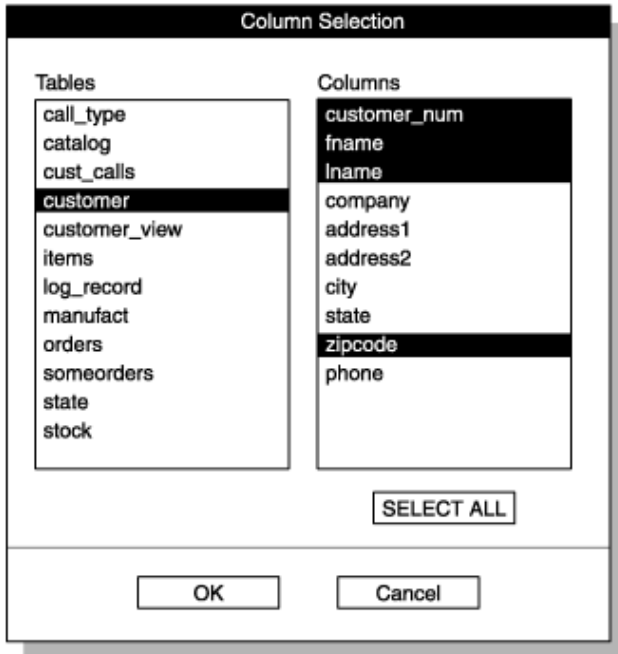
4. Select one or more columns to use in the query.

Choose from:

- To select a single column, select that column.
- To select all columns, click **Select All**.
- To select consecutive columns, select the first column. Move to the final column and hold down **SHIFT** while you select that column.
- To select nonconsecutive columns, select a column. Hold down **CONTROL** while you select additional items.

The following figure shows the **Column Selection** window with several columns selected.

Figure 74. Columns selected from a table



- When you finish selecting columns, click **OK** to return to the **Query-Definition** window.

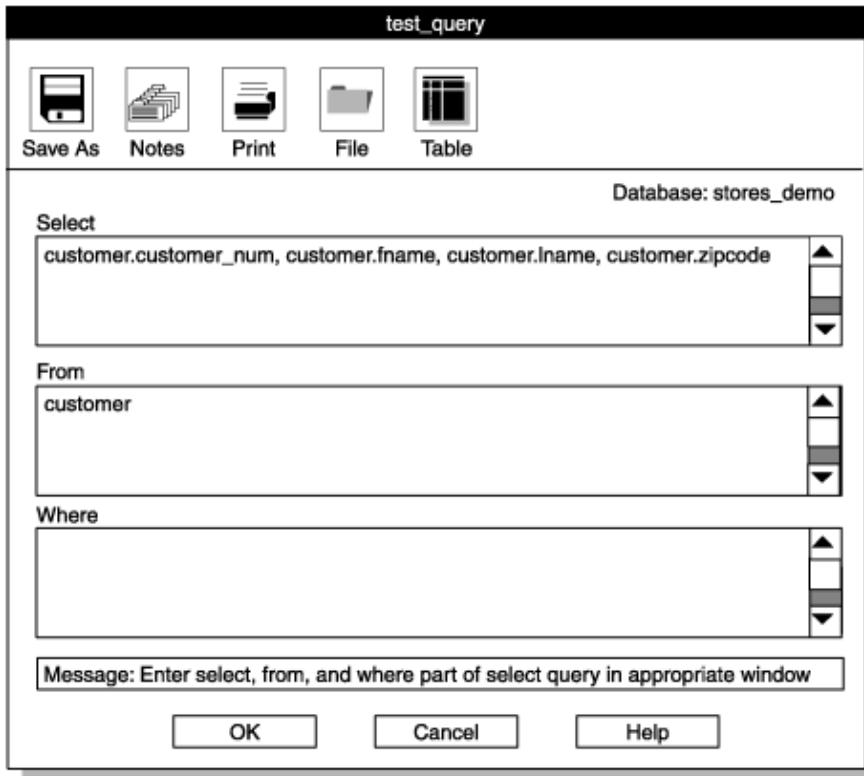
When the **Query-Definition** window reappears, the mouse cursor changes to a pointing hand and the message line reads:

Position Cursor Where Column Data to be Inserted

- Select the **Select** text box or the **Where** text box.

The following figure shows columns inserted into the **Select** text box. The iupload utility also inserts the table name into the **From** text box.

Figure 75. The Query-Definition window after using the table button



7. Repeat steps 2 on page 112 through 6 on page 113 to add columns from other tables.
8. Modify the text in the **Where** text box so that it is a valid WHERE clause.
See [Editing the WHERE clause on page 114](#).
9. Click **OK** to save the query and return to the **Query** window.
10. Click **Cancel** to return to the HPL window.

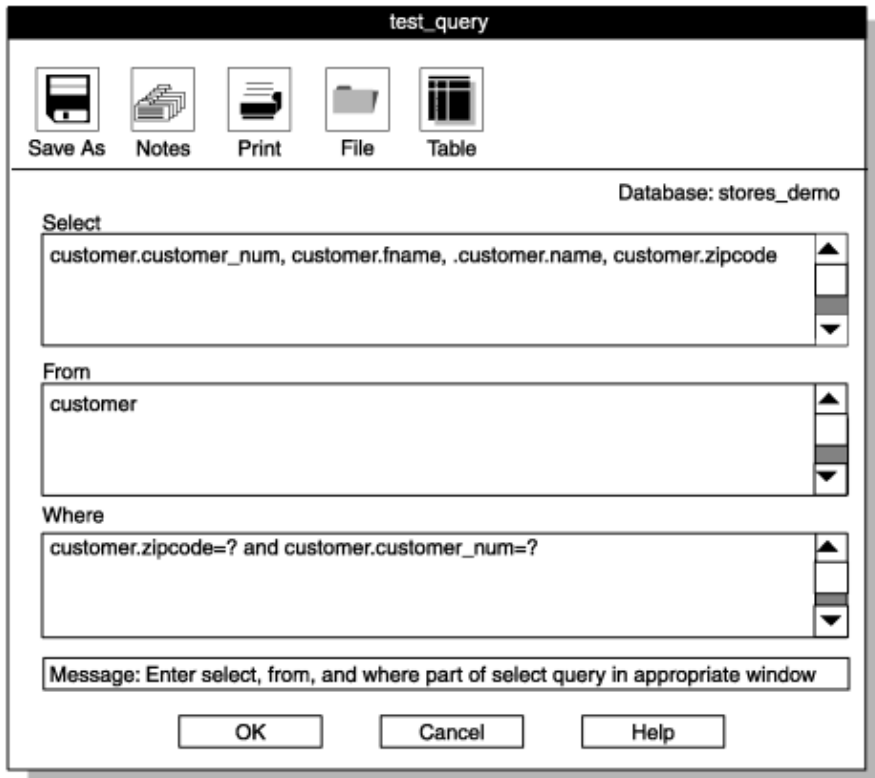
Editing the WHERE clause

When you use the **Column Selection** window to select a column or columns for the WHERE clause, the selected columns appear in the **Where** text box.

About this task

In the **Where** text box, as shown in the figure below, the `=?` symbols indicate where you must provide match conditions. The following figure shows the result when you choose **zipcode** and **customer_num** from the **customer** table.

Figure 76. The Where text box entry after you use the Table button



To edit the WHERE clause:

1. Select `=?` and change it to the match condition that you want.

For example, the **Where** text box in [Figure 76: The Where text box entry after you use the Table button on page 115](#) contains the following text:

```
customer.zipcode =? and customer.customer_num =?
```

You must change both occurrences of `=?` to valid match conditions. You might change the text as follows:

```
customer.zipcode > 50000 and
customer.customer_num > 150
```

For a full description of match conditions, see [Match condition operators and characters](#).

2. Check the comparison operators.

When you select multiple columns from the **Column Selection** window, ipload inserts `and` into the expression between each column. You might need to change `and` to `or`.

Editing a query

To edit a query, follow the same steps as for creating a query, but open an existing query in the **Query** window.

About this task

To edit a query:

1. Choose **Components > Query** from the HPL main window.

The Query window appears, as [Figure 71: The Query window on page 110](#) shows.

2. Click **Open**.
3. Select a query from the list of queries.
4. Click **OK**.

The **Query-Definition** window appears, as [Figure 72: The Query-Definition window on page 111](#) shows. The name of the query that you are editing appears in the title bar.

5. Modify the **Select**, **From**, and **Where** text boxes.
6. Click **OK** to save the modified query.
7. Click **Cancel** to return to the HPL main window.

Exporting and importing queries

You can use the **File** button on the **Query-Definition** window to export a query to a file or to import a query that you prepared with some other tool.

About this task

For example, you might use DB-Acess to prepare and test a query and to save the query to a file. You can then import that query into the High-Performance Loader (HPL).

Importing a query

You can use the **Import/Export File Selection** window to import a query that you prepared outside of the High-Performance Loader (HPL).

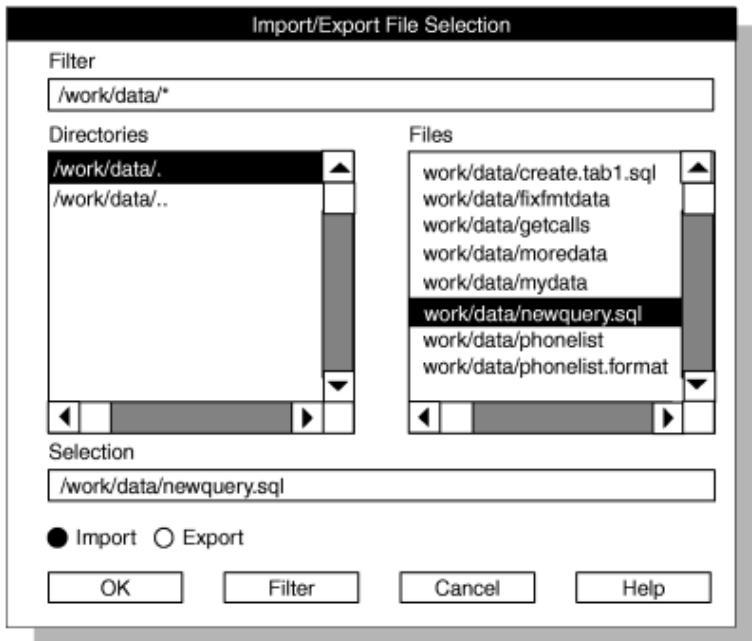
About this task

To import a query:

1. Display the **Query-Definition** window by following the steps in [Creating a query on page 109](#).
2. Click **File**.

The **Import/Export File Selection** window appears, as the following figure shows.

Figure 77. The Import/Export File Selection window



3. Click **Import**.
4. Specify the file that you want to import.
 - a. Type a path name and appropriate wildcards in the **Filter** text box and click **Filter**. Use an asterisk (*) to list all of the files in the directory. Then select a file and click **OK** or double-click a file name.
 - b. Type the full path name in the **Selection** text box and then click **OK**.

The text from the imported file appears in the **Query-Definition** window.

If ipload can interpret the SQL statement, the SQL statement is inserted into the appropriate **Select**, **From**, and **Where** text boxes.

If ipload cannot interpret the SQL statement, the entire content of the imported file appears in the **Select** text box.

5. Edit the query so that it meets your needs.
6. Click **OK**.

If the query is a valid SQL query, the display returns to the **Query** window.

If the query is not a valid SQL query, ipload highlights the portion of the query that it cannot interpret and provides an error message.

7. From the **Query** window, click **Cancel** to return to the HPL main window.

Exporting a query

The **File** button also allows you to export the query as an SQL statement.

About this task

You can prepare a query for export in the following ways:

- Create a query. (See [Creating a query on page 109.](#))
- Open an existing query.
- Import an already prepared query and modify it. (See [Importing a query on page 116.](#))

To export a file:

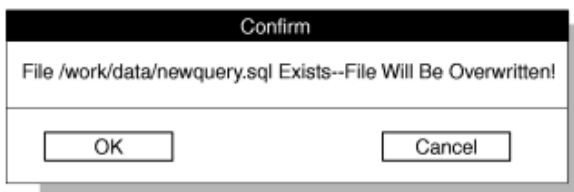
1. Follow the steps in [Creating a query on page 109](#) to prepare a query in the **Query-Definition** window (see [Figure 72: The Query-Definition window on page 111](#)).
2. Click **File**.

The Import/Export File Selection window appears (see [Figure 77: The Import/Export File Selection window on page 117](#)).

3. Click **Export**.
4. Select the directory and file where the query is to be stored.
 - a. Add the name of a new file to a path name in the **Selection** text box and click **OK**.
 - b. Type a path name and appropriate wildcard in the **Filter** text box and click **Filter**. Then select a file name.
5. Click **OK**.

If the file that you specified exists, ipload asks if you want to overwrite the existing file, as the following figure shows.

Figure 78. The Confirm File Overwrite window



6. You now have two choices:

Choose from:

- Click **OK** to overwrite. The display returns to the **Query** window.
- Click **Cancel** to choose a different file name.

The ipload utility writes the text from the **Select**, **From**, and **Where** text boxes into the specified file as a single SQL statement.

7. Click **OK**.
The display returns to the **Query** window.

The Database Views window

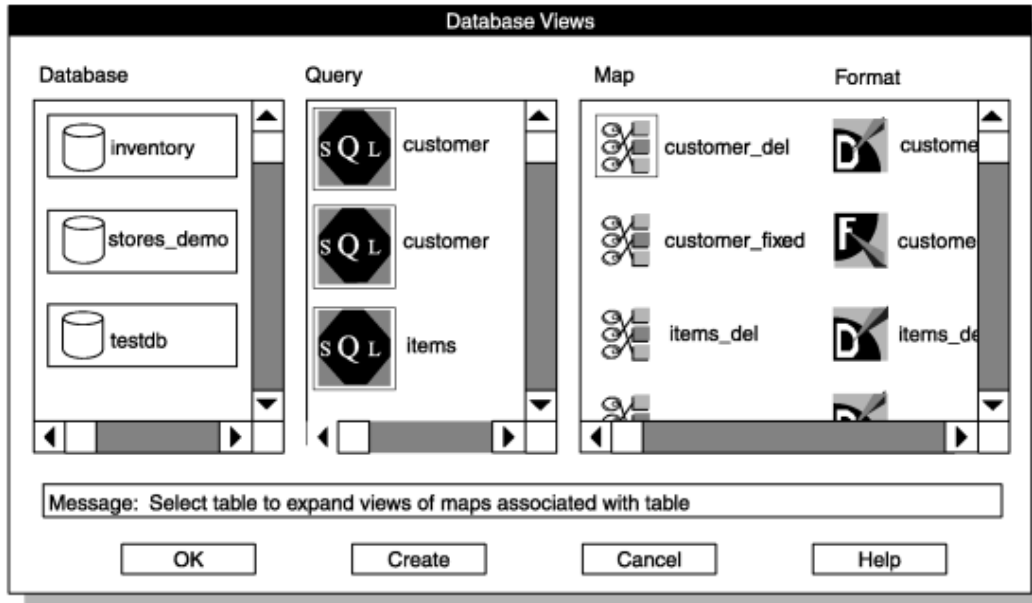
You can display a list of the queries, maps, and formats that are associated with a project from the Database Views window. You can also create or edit a query.

The Database Views window appears in the following situations:

- When you click **Query** in the **Unload Job** window and no query name is in the **Query** text box
- When you click **Search** in the **Query** window

The following figure shows the **Database Views** window. [Views windows on page 53](#) discusses how to use **Views** windows.

Figure 79. The Database Views window



Define maps

This section describes how to define maps with the High-Performance Loader (HPL). It also describes the options that are available for defining maps.

Related reference

[Inline data on page 95](#)

Related information

[Inline data on page 93](#)

[Data in a separate file on page 94](#)

[Simple LO data in a separate file on page 96](#)

[Components of the unload job on page 142](#)

Load and unload maps

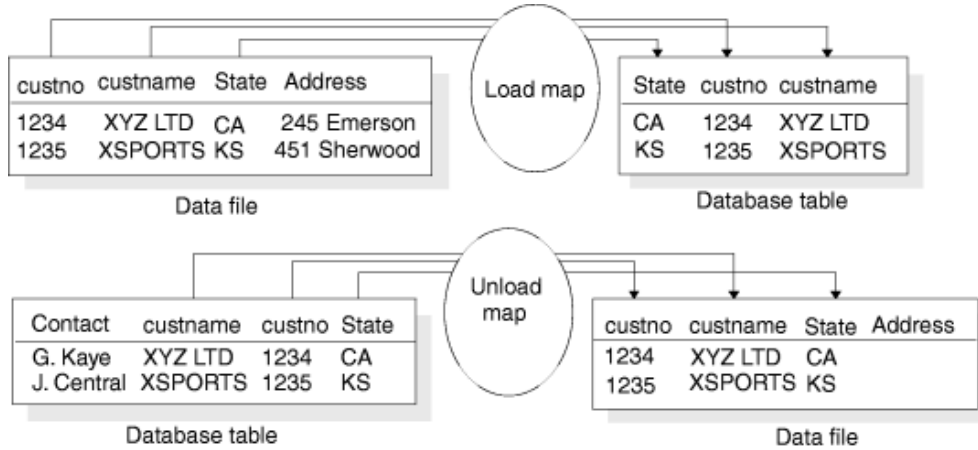
You can use the `ipload` utility to build a *map*. A map specifies the relationship between the fields of a data file and the columns of a database.

To load data into a database, you define a load map, which associates fields from records in a data file to columns in a database table.

To unload data, you define an unload map. The unload map associates the columns that a query retrieves from one or more tables to the fields in a data file.

The following figure shows the relationship between load and unload maps.

Figure 80. Using a map



The ipload utility stores information about maps in the **maps**, **mapitem**, **mapoption**, and **mapreplace** tables of the onpload database.

You can define a map at any time. After you define a map, you use it with the **Load Job** window or the onpload utility.

Related reference

The onpload database

[Define maps on page 217](#)

The mapitem table in the onpload database

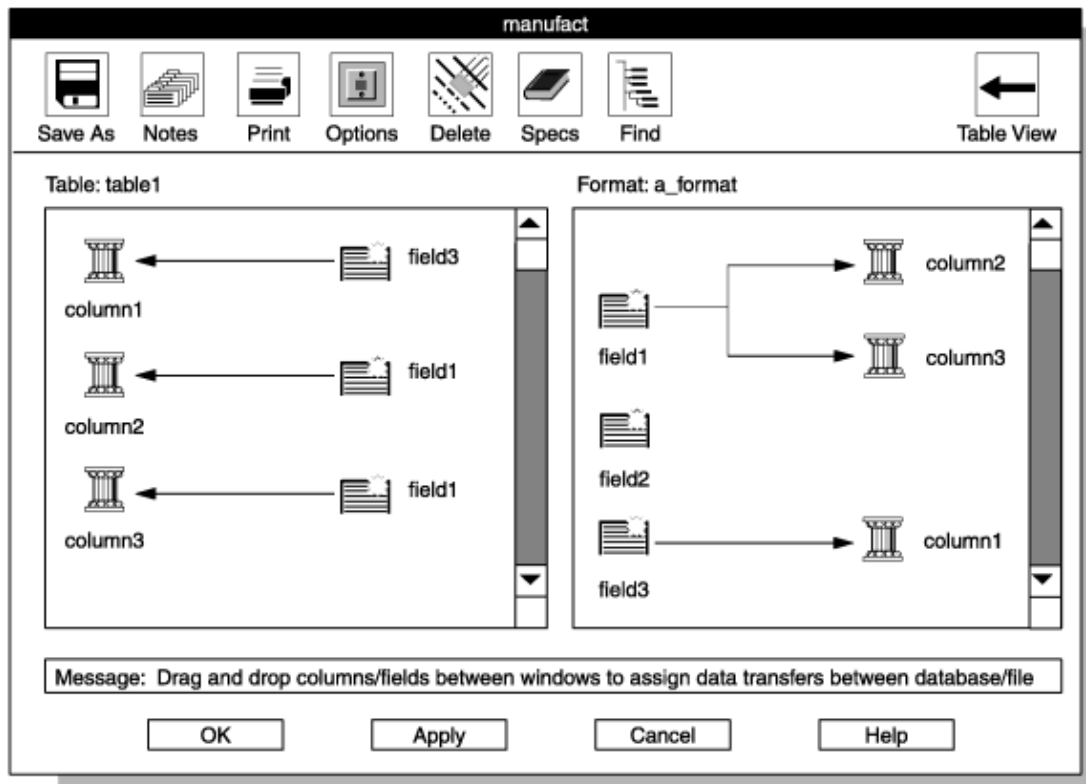
The maps table in the onpload database

The Map-Definition window

You can associate an input item with a table column from the **Map-Definition** window.

The following figure shows a **Map-Definition** window for a load map.

Figure 81. The Map-Definition window



The map specifies which fields of the data file are loaded into database columns. The data moves from the fields of a data file into the columns of a database.

The Table and Format panes

The **Map-Definition** window contains two panes: the **Table** pane and the **Format** pane.

The window has two panes so that you can take the following actions:

- Scroll the panes to see all of the columns or fields of a long data file or database table.
- Connect an input field to more than one column.

The left column of icons in each pane represents the active elements of the display. These left columns do not change. In a load map, the columns in the **Table** pane receive the input. In the **Format** pane, data from the fields moves into the columns of the database table.

The right column of icons in each pane represents the associations that you make. These columns change as you build the map. A field might be listed more than once in the right column of the **Table** pane because you can store a field from the data file in more than one database column. This field is mapped (with a split arrow) to two columns in the **Format** pane. A column never appears more than once in the list to the right of the **Format** pane because a column can only receive input from one database field.

By scanning the left pane, you can easily see which columns are receiving data from the data file. By scanning the right pane, you can see which fields of the data file are providing data and which fields are not being used.

Unassigned or multiple-assigned fields and columns

The High-Performance Loader (HPL) does not require a one-to-one connection between the fields and columns. You can map a field to multiple columns. [Figure 85: The Map-Definition window on page 126](#) shows a map where the data from one field is placed into two columns.

You can also have a column that has no mapping association. Field 1 in the **Format** pane in [Figure 83: Map-Definition window, one association completed on page 124](#) does not have an association. If a column does not receive input, onpload sets the column to null.

Identical field names and column names

When you create a format, you can assign arbitrary names to the fields of the data file. You might find it convenient to assign names that correspond to the names of the columns in the database. When you create a map, ipload automatically links columns and fields that have the same name and type, thus saving you work.

Creating a load map

You can create a load map from the **Load Job** window or from the **Components** menu of the High-Performance Loader (HPL) main window.

Before you begin

Before you can create a load map, you must create a format that describes the data file that you plan to load. For information about how to create a format, see [Define formats on page 84](#).

About this task



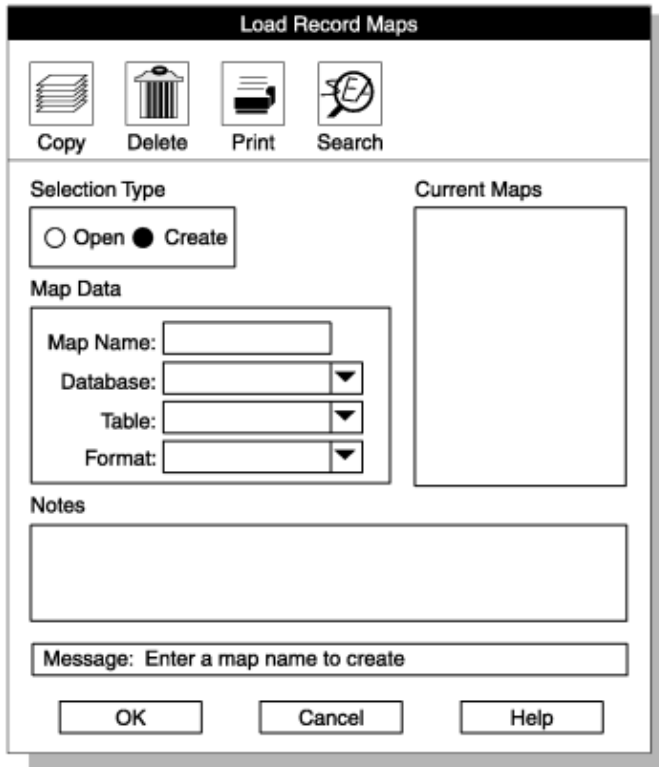
Important: The HPL does not support conversion from extended type data and smart-large-object data (Ext Type data types) to non-Ext Type data types. A field that is defined as an Ext Type data type can be mapped only to an Ext Type column. For more information about Ext Type data types, see [Data types allowed in a fixed format on page 89](#) or [Data types allowed in a delimited format on page 98](#)

To create a load map:

1. Choose **Components > Maps > Load Map** from the HPL main window.

The **Record Maps** window appears, as the following figure shows.

Figure 82. The Load Record Maps window



2. Click **Create** in the **Selection Type** group.
3. Choose a name for the map and type it in the Map Name text box.
4. Type the names of the database and table where the data will be loaded in their corresponding text boxes.

You can also click the down arrow to choose the names from a selection list. The Tables selection list includes synonyms that are valid for the local database server.

5. Type the format that describes the data file in the **Format** text box.

You can also click the down arrow to choose the format from a selection list.

6. Click **OK** to open the **Map-Definition** window.

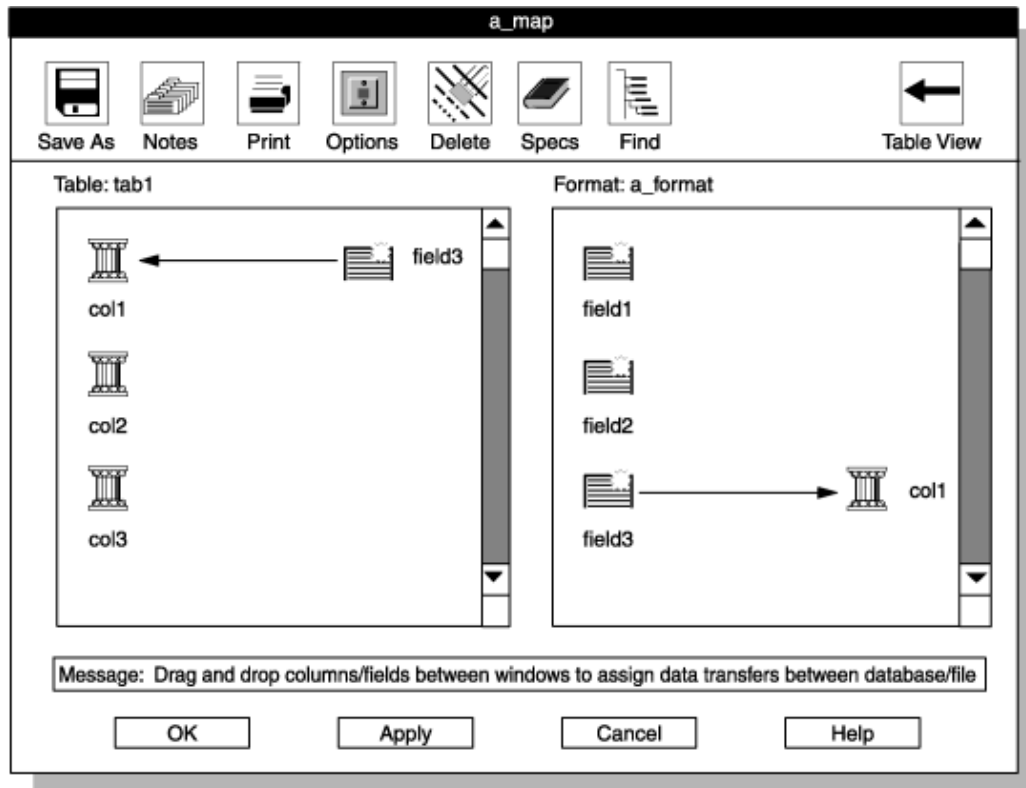
A **Map-Definition** window similar to [Figure 83: Map-Definition window, one association completed on page 124](#) appears.

7. Click a column icon in the left column in the **Table** pane and hold the mouse button down.
A box appears around the icon and its name.
8. Drag the box to a field icon in the **Format** pane.

When you connect columns to fields, it does not matter whether you drag a column to a field or drag a field to a column, but you must always connect items from the left column of each pane.

The following figure shows a **Map-Definition** window with this step completed.

Figure 83. Map-Definition window, one association completed



9. Repeat steps 7 on page 123 and 8 on page 123 for each field that you want to transfer into the database.
10. Add the options that you want, if any.
For instructions, see [Defining the mapping options on page 127](#).
11. Click **OK** to return to the **Load Record Maps** window.

Related reference

[Create a fixed format that uses carriage returns on page 92](#)

Unload maps

An unload map associates columns extracted from a database by a query with the fields in a data-file record. You can create an unload map from the **Load Job** window or from the **Components** menu of the HPL main window. After you define an unload map, you use it with the **Unload Job** window or the onpload utility.

Creating an unload map

Before you begin

Before you can create an unload map, you must define a query on the table that will be unloaded. For instructions on how to define a query, see [HPL queries on page 108](#).

About this task

! **Important:** The HPL does not support conversion from extended type data and smart-large-object data (Ext Type data types) to non-Ext Type data types. An Ext Type column can be mapped only to an Ext Type field. For more information about Ext Type data types, see [Data types allowed in a fixed format on page 89](#) or [Data types allowed in a delimited format on page 98](#).

To create an unload map:

1. Choose **Components > Maps > Unload Map** from the HPL main window.

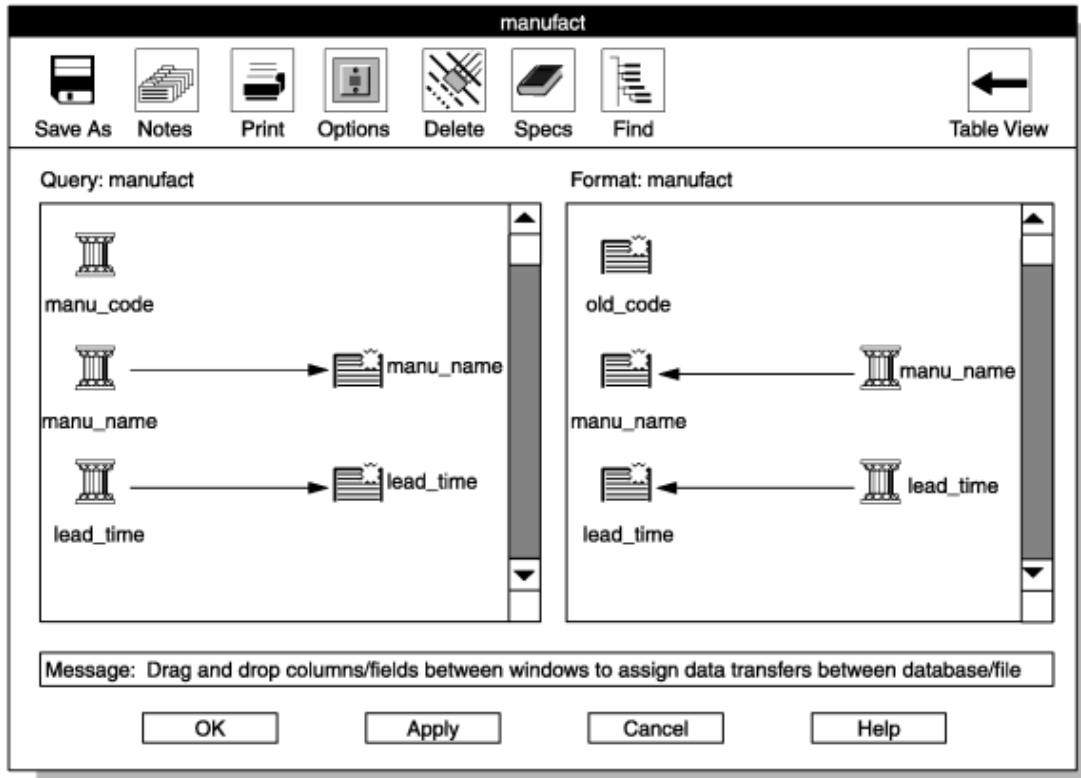
The **Unload Record Maps** window appears, as the following figure shows.

Figure 84. The Unload Record Maps window

2. Click **Create** in the **Selection Type** group.
3. Choose a name for the map and type the name in the **Map Name** text box.
4. Type the name of the database in the **Database** text box.
You can click the down arrow to choose a database from a selection list of databases.
5. Type the name of a query in the **Query** text box.
You can click the down arrow to choose a query from a selection list of queries.
6. Type the format name in the **Format** text box.
You can click the down arrow to choose a format from a selection list of formats.
7. Click **OK**.

A **Map-Definition** window is similar to the following figure. In this figure, some of the field names match column names. The ipload utility automatically maps columns to fields of the same name. The direction of the arrows indicates the flow of data, as shown.

Figure 85. The Map-Definition window



8. To map a database column to a data-file field, click the database-column icon. Drag the column to the data-file field icon that you want.

An arrow links the column icon to the field icon.

9. Repeat step 8 on page 126 until you have mapped all the columns you want to fields.

10. Define any mapping options as appropriate.

For information about mapping options, see [Mapping options on page 127](#).

11. Click **OK** to save the map and return to the **Unload Record Maps** window.

12. Click **Cancel** to return to the HPL main window.

Unload data by using functions

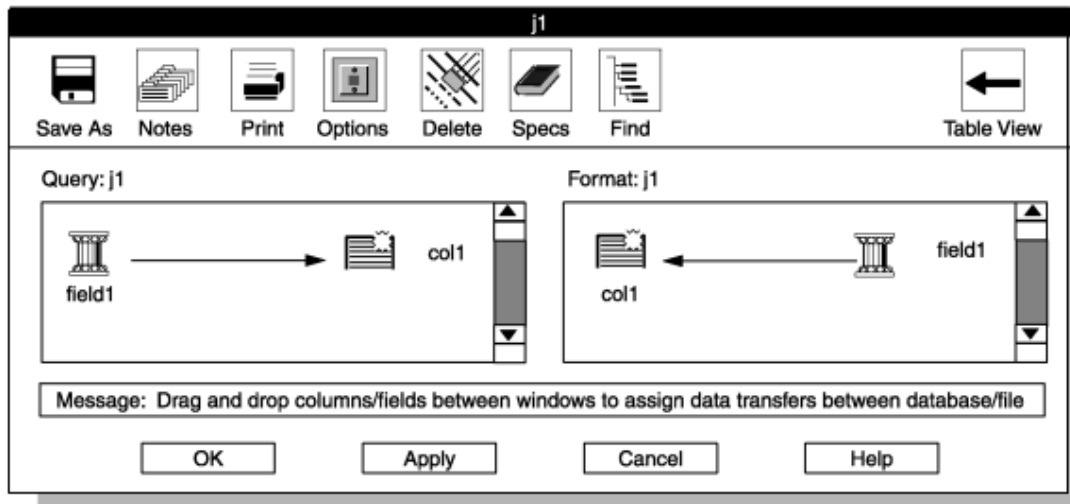
If you use a function in a query to unload data, you must associate a name with the result of that function.

In the following example, the returned value of the function TRIM is assigned the name `field1`.

```
SELECT TRIM(col1) field1 FROM tab1
```

After submitting the query, you must attach `field1` to `col1` of the unload file manually, as the following figure shows.

Figure 86. The Map-Definition window



Mapping options

The mapping options define conversions that onpload applies to the data before it inserts the data into the database (for a load job) or into the data file (for an unload job). These conversions can include case conversion, text justification, data masking through picture strings, default values, and fill characters. The mapping options also allow you to replace imported data with data from other database tables.

The information from the **Mapping Options** window is stored in the **mapoption** table of the onpload database.

Related reference

The onpload database

Defining the mapping options

This procedure describes how to specify mapping options.

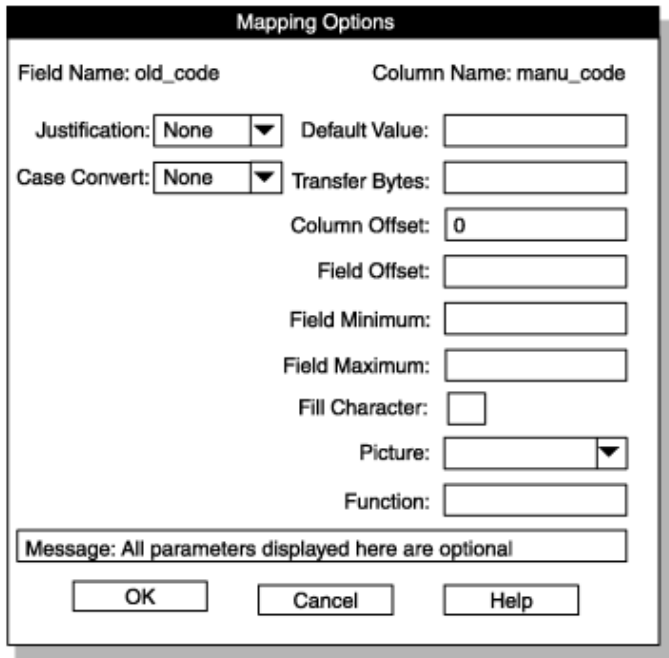
About this task

To define mapping options:

1. Display the **Map-Definition** window by following the steps for [Creating a load map on page 122](#), or [Creating an unload map on page 124](#).
2. Select the field or column (in the right column of a pane) that you want to modify.
3. Click **Options**.

The **Mapping Options** window appears, as the following figure shows.

Figure 87. The Mapping Options window

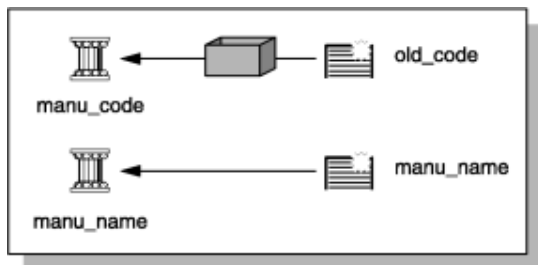


4. Change the options that you want.
5. When you have set all the options that you want, click **OK** to return to the **Map-Definition** window.

Results

When you return to the map window, an options symbol (a small box) appears between the field and the column, as the following figure shows. The options symbol indicates that mapping options are in effect.

Figure 88. Fragment of the Map-Definition window showing an options symbol



Related reference

The mapoption table in the onpload database

Set the mapping options

You can set as many of the choices on the **Mapping Options** window as you need.

Justification option

The **Justification** option positions text within a record. You can justify the text to the left or right, or you can center it.

Case Convert option

The **Case Convert** option converts the case of the data to the selected case. The High-Performance Loader (HPL) supports upper, lower, and proper-name conversions. For example, you can make the following conversions.

Input	Conversion type	Result
JOHN LEE SMITH	Proper Name	John Lee Smith
john lee smith	Proper Name	John Lee Smith
john lee smith	Upper	JOHN LEE SMITH
JOHN LEE SMITH	Lower	john lee smith

Default Value option

The **Default Value** option specifies the value that is inserted into the column when no field is mapped into that column.

Transfer Bytes option

The **Transfer Bytes** option specifies the number of bytes in the record field to transfer to the database column.

For variable-length format records, this number reflects the maximum size of the field. The actual number of bytes to transfer is determined by the record or field delimiters.

Column Offset option

The **Column Offset** option specifies the offset from the beginning of a column field at which to start transferring the data from the field of the data record. Offsets are zero based.

Field Offset option

The **Field Offset** option specifies the offset from the beginning of a record field at which to start transferring data to the column. Offsets are zero based.

Field Minimum and Field Maximum options

The **Field Minimum** and **Field Maximum** options specify the smallest and largest acceptable values for a numeric column. If the data in the field is outside that range, the HPL rejects the record. This option is available only for fields with numeric formats, such as integer, short, or float.

Fill Character option

The **Fill Character** option lets you specify a character that you use to pad the contents of a field. The fill character can be any character that you can type on the keyboard. You can specify a fill character for fixed ASCII and COBOL loads or unloads. The fill character is filled in as a trailing character.

Picture option

The **Picture** option lets you reformat and mask data from the field of a record before the data is transferred to the database. Picture strings, explains picture strings.

Function option

The **Function** option specifies a user-defined function that is called for every record that is processed. You must add the function to the dynamically linked library. For information about using custom functions, see the API interface documentation in Custom-conversion functions.

Editing options

This section discusses specialized options in the **Map-Definition** window.

Using the Delete button

You can break the association between a column and a field with the **Delete** button in the **Map-Definition** window.

About this task

To use the **Delete** button:

1. Click an icon in the right column of either of the panes in the **Map-Definition** window.
2. Click **Delete** to remove the arrow that connects the item to another item.

Using the Find button

You can find a column or field in a pane with the **Find** button in the **Map-Definition** window. The `ipload` utility scrolls the selected item into view and puts a box around it. This option is useful when the list of columns or fields is so long that the pane cannot display all of the items.

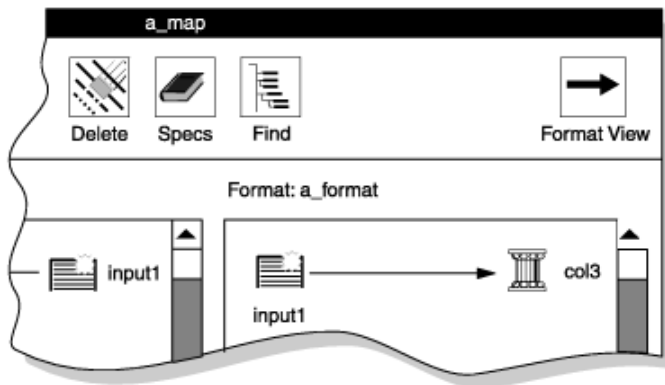
About this task

To use the **Find** button:

1. In the **Map-Definition** window, select either the **Table** pane or the **Format** pane.

When you select a pane, the view indicator in the upper right corner of the window changes to show which pane you selected. The following figure shows the upper portion of the **Map-Definition** window after you select the **Format** pane.

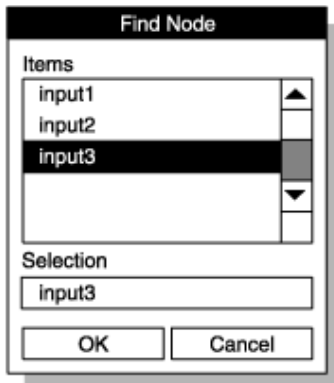
Figure 89. The view indicator



2. Click **Find**.

The **Find Node** window appears, as the following figure shows.

Figure 90. The Find Node window



Because the view indicator shows Format View, the **Find Node** window lists the fields of the data file. To see the columns of the database table, make sure that the view indicator shows Table View.

3. To select the item to find, you can use either of these methods:

Choose from:

- Scroll through the list box to locate the item that you want to find and then select the item.
- Type the name of the item that you want to find in the **Selection** text box.

4. Click **OK**.

The **Map-Definition** window appears again. The selected field or column is highlighted with a box.

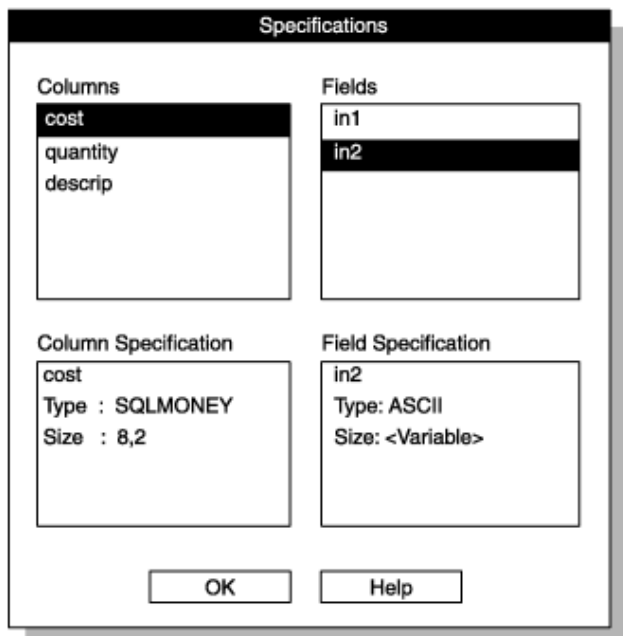
Using the Specs button

With the **Specs** button, you can display the **Specifications** window, where you can examine the characteristics of the columns and fields in your map.

About this task

The following figure shows a sample **Specifications** window.

Figure 91. The Specifications window



To use the **Specifications** window:

1. Click **Specs** in the **Map-Definition** window to display the **Specifications** window.
2. Select a column from the **Columns** list box or a field from the **Fields** list box or both.

The specification boxes in the lower part of the screen display the characteristics of the selected items.

3. When you finish examining the specifications, click **OK** to return to the **Map-Definition** window.

Results

The **Specifications** window displays the attributes of columns and fields. The **Specifications** window does not allow you to edit the attributes it displays. To change the attributes of a field, you must modify the format of the data file. (See [Format options on page 103](#).) To change the attributes of a column, you must use appropriate SQL statements to modify the database table.

Map Views window

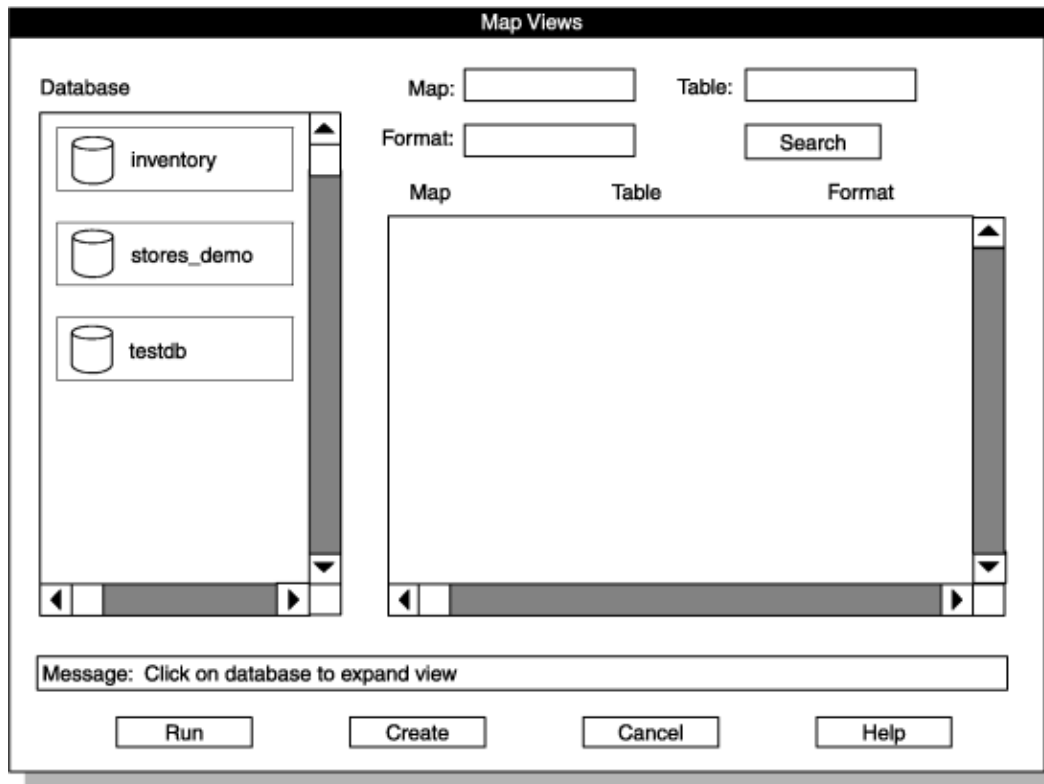
You can display a list of the components that are associated with a database in a specific project from the **Map Views** window. You can also create or edit a map.

The **Map Views** window appears in the following situations:

- If you click **Map** in the **Load Job** or **Unload Job** window when no map name is in the **Map** text box
- If you click **Search** in the **Load Record Maps** or **Unload Record Maps** window

The following figure shows the **Map Views** window for a load map.

Figure 92. The Map Views window for a load map



Seeing the load maps of a database

About this task

To see the load maps of a database:

1. Select a project in the HPL main window.
2. Choose **Components > Maps** from the HPL main window.
3. Choose **Load > Map** or **Maps > Unload Map**.
4. After the **Load Record Maps** or **Unload Record Maps** window appears, click **Search**.

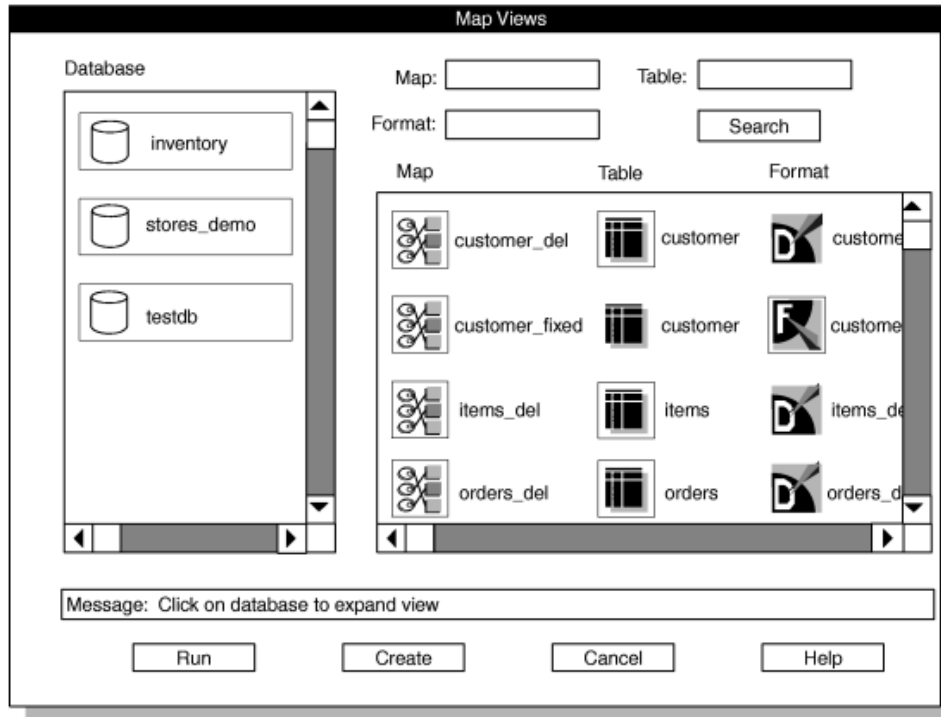
The **Map Views** window appears, as [Figure 92: The Map Views window for a load map on page 133](#) shows.

5. Select a database.

The `ipload` utility displays a list of the maps associated with that database, as the following figure shows. The **Table** and **Format** columns show the database column and the format associated with each map.

If you want to edit a specific map or format, click its button and the corresponding definition window appears.

Figure 93. The Map Views window with the view expanded



Seeing selected load maps

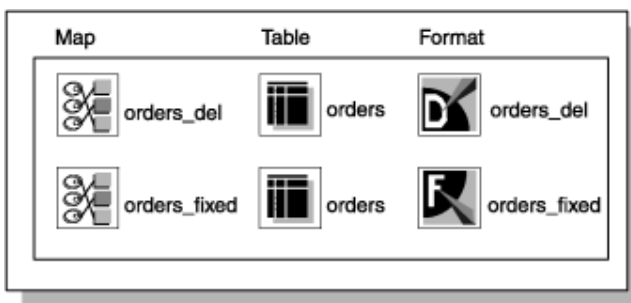
About this task

To see selected load maps:

1. Open the **Map Views** window.
2. Select a database.
3. Type the name or partial name of a map, table, or format in the **Map**, **Table**, or **Format** text box.
You can use wildcards in the name.
4. Click **Search**.

The following figure shows the maps that you find when you search for any table that includes **orders** in its name.

Figure 94. The maps that a search found



Define filters

This section describes how to create, edit, and delete filters.

Filters

Filters are similar to queries. However, queries select data from database tables, whereas filters select data from a data file. During the load process, ipload loads all of the records from a data file into a database table unless you use a filter to exclude some of the records.

A filter is a mechanism for prescreening data-file records for eligibility as database table entries. You can use the filter to include or exclude records explicitly during the load process. You define *match conditions* to filter the records. Match conditions are selection criteria that test one or more data-file fields for certain values or text.

You can define filters at any time. After you define a filter, you can specify it in the **Load Job** window. The **Load Job** window is illustrated in [Figure 104: The Load Job window on page 155](#).

Filters have the following restrictions:

- You cannot use filters with Ext Type data types.
- The DATE and DATETIME data filters can only be applied to the fixed ASCII and delimited format types.

Example of using filter

Suppose that you have a worldwide telemarketing data file that contains the name, country, yearly salary, and age of potential contacts, as the following example shows:

```
John BrownUS 125,00057
Mary SmithArgentina83,00043
Larry Little US 118,00042
Ann SouthCanada 220,00053
David PetersonFrance 175,00072
Richard NorthSpain350,00039
Nancy RichardsJapan150,00054
William ParkerEgypt200,00064
```

To create a database that includes people who earn over \$100,000 a year, are over the age of 50, and live outside the United States:

1. Use the match condition **discard salary < 100,000** to exclude people who earn less than \$100,000 a year. The selected records are as follows:

```
John BrownUS125,00057
Larry Little US118,00042
Ann SouthCanada220,00053
David PetersonFrance175,00072
Richard NorthSpain350,00039
Nancy RichardsJapan150,00054
William ParkerEgypt200,00064
```

2. Use the match condition **keep age > 50** to include people over the age of 50. The remaining records are as follows:

```
John BrownUS125,00057
Ann SouthCanada220,00053
David PetersonFrance175,00072
Nancy RichardsJapan150,00054
William ParkerEgypt200,00064
```

3. Use the match condition **discard country = US** to exclude people living in the United States. The remaining records are the records that match all of the restrictions:

```
Ann SouthCanada220,00053
David PetersonFrance175,00072
Nancy RichardsJapan150,00054
William ParkerEgypt200,00064
```

If you want to use the same data file to create a database of only those people who live in the United States, or only those people under the age of 30, simply define another filter. There is no limit to the number of filters that you can define for a data file.

Related reference

[Define filters on page 230](#)

Creating a filter

Before you begin

Before you can create a filter, you must create a format that describes the data file. For information about how to create a format, see [Define formats on page 84](#).

About this task

The ipload utility stores the filter information in the **filters** table of the onpload database. For more information about the **filters** table, see [The filters table in the onpload database](#).

To create a filter:

1. Choose **Components > Filter** from the HPL main window.

The **Filters** window appears, as the following figure shows.

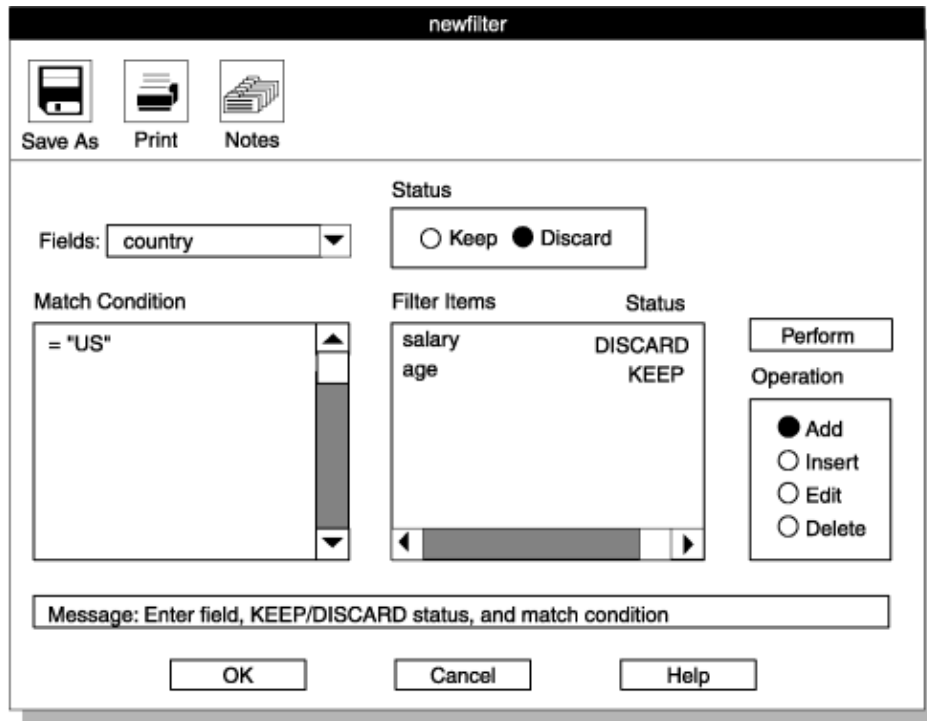
Figure 95. The Filters window



2. Click **Create** in the **Mode** group.
3. Choose a name for the filter and type the name in the **Filter** text box.
4. Type the name of an existing format in the **Format** text box, or click the down arrow and choose a format from the selection list.
5. Click **OK**.

The **Filter-Definition** window appears. The following figure shows a partially completed **Filter-Definition** window.

Figure 96. The Filter-Definition window



From the **Filter-Definition** window, you can prepare a filter that specifies which data from the input file is be loaded into the database table.

The **Filter-Definition** window has the following parts.

Table 15. Parts of the Filter-Definition window

Section	Description
Fields	Specifies the data-file field used in a match condition
Status	Indicates whether you want to keep or discard records that meet the match condition
Match Condition	Specifies the criteria for keeping or discarding a record
Filter Items/Status	Lists existing filter items and their status As you add match conditions, the conditions are added to this list.

Related reference

The filteritem table in the onpload database

The filters table in the onpload database

Preparing the filter definition

About this task

To prepare the filter definition:

1. Click **Add** in the **Operation** group to specify that you want to add a new match condition.
2. Type the name of the record field that you want to match in the **Fields** text box.
You can also click the down arrow to see a selection list.
3. Click **Keep** or **Discard** in the **Status** group.
This selection indicates whether the matching record should be entered into the database or discarded.
4. Type the match condition in the **Match Condition** text box by using the appropriate logical operators and match characters.

See Match condition operators and characters for a list of the logical operators and match characters.

5. Click **Perform**.
6. Repeat steps 2 on page 139 through 5 on page 139 for each additional filter item.
7. Click **OK** to save the filter and return to the **Filters** window.
8. Click **Cancel** to return to the HPL main window.

Modifying a filter

After you create a filter, you might need to modify that filter.

About this task

1. Choose **Components > Filter** from the HPL main window to display the **Filters** window.
2. Click **Open** in the **Mode** group.
3. Select the filter that you want to modify.
4. Click **OK** to display the **Filter-Definition** window.
5. Click **Add** in the **Operation** group.
6. Type the name of the record field in the **Fields** text box.
7. Type the match condition in the **Match Condition** text box.
8. Click **Keep** or **Discard** in the **Status** group to indicate the filter status.
9. Click **Perform**.
10. Click **OK** to save your changes and return to the **Filters** window.
11. Click **Cancel** to return to the HPL main window.

Results

Editing an existing filter

About this task

To edit an existing filter:

1. Choose **Components > Filter** from the HPL main window to display the **Filters** window.
2. Click **Open** in the **Mode** group.
3. Select the filter that you want to modify.

4. Click **OK** to display the **Filter-Definition** window.
5. Click **Edit** in the **Operation** group.
6. Select the filter item you want from the list of items.
The field, status, and match conditions appear in their respective areas on the screen.
7. Change the information that you want.
8. Click **Perform**.
9. Click **OK** to save your changes and return to the **Filters** window.
10. Click **Cancel** to return to the HPL main window.

Adding an item to the filter

About this task

To add an item to the filter:

1. Choose **Components > Filter** from the HPL main window to display the **Filters** window.
2. Click **Open** in the **Mode** group.
3. Select the filter that you want to modify.
4. Click **OK** to display the **Filter-Definition** window.
5. Click **Add** in the **Operation** group.
6. Type the name of the record field in the **Fields** text box.
7. Type the match condition in the **Match Condition** text box.
8. Click **Keep** or **Discard** in the **Status** group to indicate the filter status.
9. Click **Perform**.
10. Click **OK** to save your changes and return to the **Filters** window.
11. Click **Cancel** to return to the HPL main window.

Inserting an item into the filter sequence

About this task

To insert an item in the filter sequence:

1. Choose **Components > Filter** from the HPL main window to display the **Filters** window.
2. Click **Open** in the **Mode** group.
3. Select the filter that you want to modify.
4. Click **OK** to display the **Filter-Definition** window.
5. Click **Insert** in the **Operation** group.
6. From the list of items, select the filter item before which you want to insert the new item.
7. Type the name of the record field in the **Fields** text box.
8. Type the match condition in the **Match Condition** text box.
9. Click **Keep** or **Discard** in the **Status** group to indicate the filter status.
10. Click **Perform** to insert the new item before the selected filter item in the **Filter Items** list box.

11. Click **OK** to save your changes and return to the **Filters** window.
12. Click **Cancel** to return to the HPL main window.

Deleting a filter

About this task

To delete a filter:

1. Choose **Components > Filter** from the HPL main window to display the **Filters** window.
2. Click **Open** in the **Mode** group.
3. Select the filter that you want to edit.
4. Click **OK** to display the **Filter-Definition** window.
5. Click **Delete** in the **Operation** group.
6. Select the item that you want to delete from the list of filter items.
7. Click **Perform**.
8. Click **OK** to save your changes and return to the **Filters** window.
9. Click **Cancel** to return to the HPL main window.

Filter views

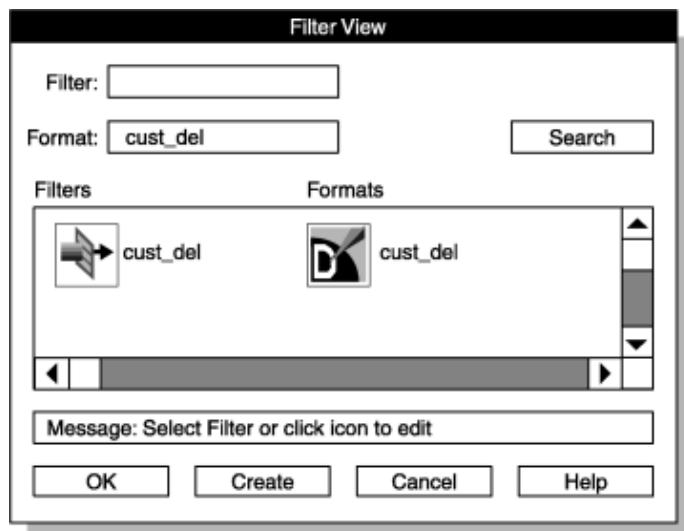
You can display a list of the filters and formats that are associated with a project from the **Filter Views** window. You can also create or edit a filter.

The **Filter Views** window appears in the following situations:

- If you click **Filter** in the **Load Job** window when no filter name is in the **Filter** text box
- If you click **Search** in the **Filters** window

The following figure shows the **Filter Views** window. [Views windows on page 53](#) discusses the use of **Views** windows.

Figure 97. The Filter Views window



Filters with code-set conversion (GLS)

When you use a filter to select or discard data during the load, the High-Performance Loader (HPL) interprets the filter specification in the code set of the database server. The filtering process on data that undergoes code-set conversion occurs in the following order:

1. The onpload utility converts the input data to the code set of the database server.
2. The onpload utility performs the filtering operation.

If the code-set conversion process creates lossy errors, then the output of the filter operation can be unexpected. For information about lossy errors and how to define or evaluate a code-set conversion specification, see the *HCL® Informix® GLS User's Guide*.

Unload data from a database

This section describes the **Unload Job** window.

Related information

[The HPL data-unload process on page 6](#)

[Load Job and Unload Job windows on page 52](#)

Unload jobs

An unload job converts HCL Informix® database records to a specified format and then unloads those records to a file, tape, pipe (UNIX™ only), or device array.

You can run an unload job from the **Unload Job** window of ipload, or you can run the onpload utility from the command line.

Related reference

[The onpload utility on page 189](#)

Components of the unload job

Before you can unload data, you must first define the following components of the unload job:

- The device array that receives the unloaded data.
- The format that describes the organization of the data file into which you are unloading data.
- The query that extracts the records that you want from the database.
- The unload map that describes the relationship between the columns of a database table and the fields of the data-file record.

The map also specifies any necessary data translations, such as case conversion and justification.

You can define these components in the following ways:

- Define each component from the **Unload Job** window.
- Define each component individually from the **Components** menu.
- Use the **Generate Job** option from the **Components** menu.
- Use the **Generate** button in the Unload Job window.

Related reference

[Define formats on page 84](#)

[Define queries on page 108](#)

[Define maps on page 119](#)

[The Generate options of the ipload utility on page 160](#)

Related information

[Define device arrays on page 79](#)

Choose the database server

You must run the unload job on the *target server*. The target server is the database server that contains the database from which you unload the data. The database must be on the same database server as onpload that extracts data from it. You can run ipload on any database server on your network.

Run multiple jobs

You can run multiple unload jobs concurrently.

However, because the High-Performance Loader (HPL) is designed to use as many system resources as possible, running concurrent jobs might overload the system. If you are using a UNIX™ **cron** job to run the load and unload jobs, let one job finish before you start the next.

The **Unload Job** window displays the target and onpload database servers in the upper right corner of the display.

The Unload Job windows

You can create an unload job or select an existing job for editing from the **Unload Job Select** window.

From the **Unload Job** window, you can also create or modify the components of an unload job and run the unload job. You can change unload options before you run the unload job. The unload options include the isolation level and the maximum number of errors to permit before onpload stops the unload job.

The ipload utility stores the information about the unload job in the **session** table of the onpload database. The **session** table draws information from other onpload tables, such as **maps**, **formats**, and so on.

Related reference

The onpload database

Creating an unload job

Use the **Unload Job Select** and **Unload Job** windows to create an unload job.

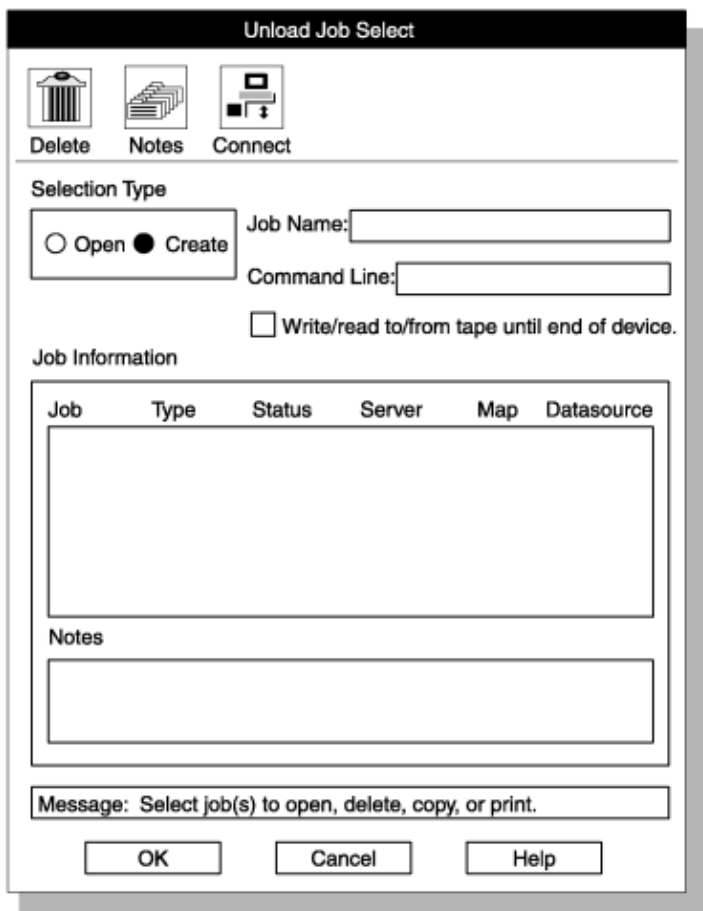
About this task

To create an unload job:

1. Choose **Jobs > Unload** from the HPL main window.

The **Unload Job Select** window appears, as the following figure shows.

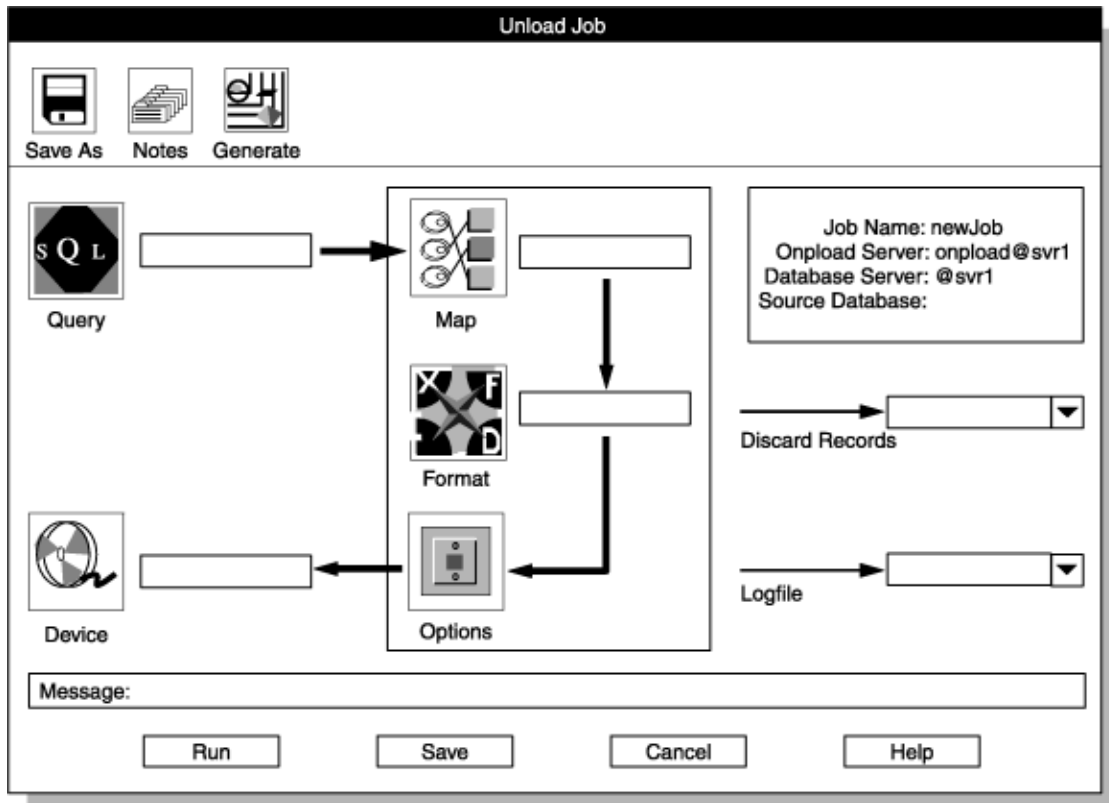
Figure 98. The Unload Job Select window



2. Click **Create** in the **Selection Type** group.
3. Choose a name for this unload job and type the name in the **Job Name** text box.
4. Optionally check the **Write/read to/from tape until end of device** check box.
For more information, see [Specify to write to the end of the tape on page 147](#).
5. Click **OK**.

The **Unload Job** window appears, as the following figure shows. [The ipload utility windows on page 45](#), provides detailed descriptions of the buttons in the **Unload Job** window.

Figure 99. The Unload Job window



6. Type appropriate values for all of the unload components.

If you click a component button, the corresponding view window opens, and you can create or select the component.

7. Specify the file that contains rejected record by using one of these methods:

Choose from:

- Type the name of the rejected file in the **Discards Records** text box.
- Click the down arrow next to the **Discard Records** text box to select the file name from the file-selection list.

8. Select the file that contains the unload status log by using one of these methods:

Choose from:

- Type the name of the log file in the **Logfile** text box.
- Click the down arrow next to the **Logfile** text box to select a file name from the **File-Selection** window.

9. Click **Options** to change unload options.

For more information, see [Changing the unload options on page 148](#).

10. Click **Save** to save this unload job.

(If you click **Run** to run the job immediately, the job is saved automatically.)

11. Now you can either run the unload job or exit and run the job later.

Choose from:

- Click **Run** to run the job.
- Click **Cancel** to exit to the **Unload Job Select** window.

Results

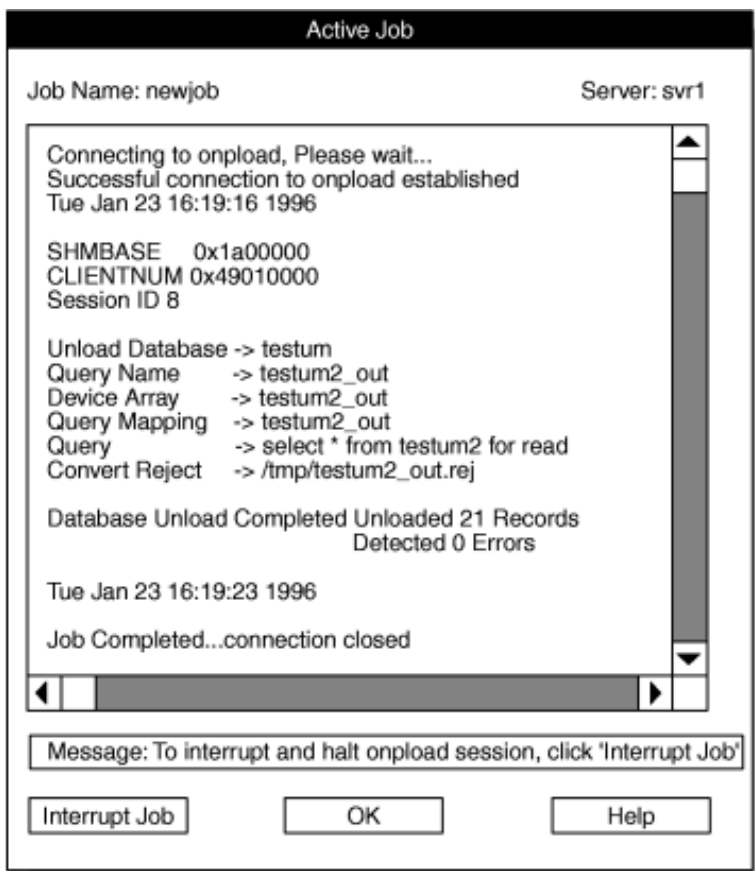
! **Important:** Use Ext Type String Length data type or Ext Type Binary Length data type if you unload data that contains null UDT values.

Run the unload job

If you click **Run** in the **Unload Job** window, the **Active Job** window appears, as the following figure shows. The **Active Job** window displays the progress of your job and indicates when the job completes. When the **Active Job** window indicates that the job is complete, click **OK** to return to the **Unload Job Select** window.

The information that onpload displays in the **Active Job** window is also stored in the log file whose name you selected in [step 8 on page 145](#).

Figure 100. The Active Job window



Related reference

[View the status of a load job or unload job on page 175](#)

Problems during an unload job

If you encounter any problems during the unload, examine the various files that onpload creates.

! **Important:** If a write to a file fails because a disk is out of space, the operating system does not return information about how much of the write succeeded. In this situation, the onpload utility cannot accurately report the number of records that were written to disk. Thus, the number of records that are logged as unloaded in the log file is imprecise.

Related reference

[The HPL browsing options on page 170](#)

Specify to write to the end of the tape

On the **Unload Job Select** window, you can specify to write to the tape until the end of the device with the **Write/read to/from tape until end of device** check box. When a tape device is full, you are prompted to provide the next tape, until the unload job is complete.

! **Important:** If you select this option, you must also select it when loading from the **Load Job** window or specify the **-Z** option from the command line; otherwise, the loaded and unloaded data might be inconsistent.

If you check this check box, it supersedes any tape size information you enter in the **Device-Array Definition** window or at the command line.

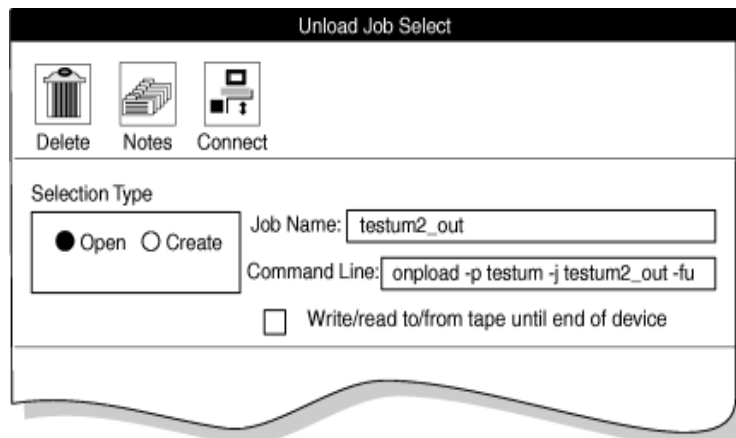
! **Important:** You must provide the same tapes in the same order on the same devices for both unload jobs and load jobs to ensure consistency.

The command-line information

If you select an existing job in the **Unload Job Select** window, the **Command Line** text box shows the onpload command that ipload generated for that unload job.

The following figure shows the **Command Line** portion of an **Unload Job Select** window. The **Command Line** text box displays the onpload command generated for the job shown in [Figure 100: The Active Job window on page 146](#).

Figure 101. Fragment of the Unload Job Select window



The command line, `onpload -p testum -j testum2_out -fu`, contains the following arguments.

-p testum

The project where the job is stored.

-j testum2_out

The name of the job.

-fu

The job that unloads (rather than loads) data.

You can copy the onpload command from the **Command Line** text box and paste it at a system prompt to run the unload job. If you need to run the unload job multiple times (for example, every evening at 5:00 p.m.), you can save the onpload command and run it later.

You do not need to start ipload to run a job from the system prompt. Both ipload and onpload use the onpload database, but each one uses it independently.

Changing the unload options

About this task

The **Unload Options** window contains the following options.

Table 16. The Unload Options window options

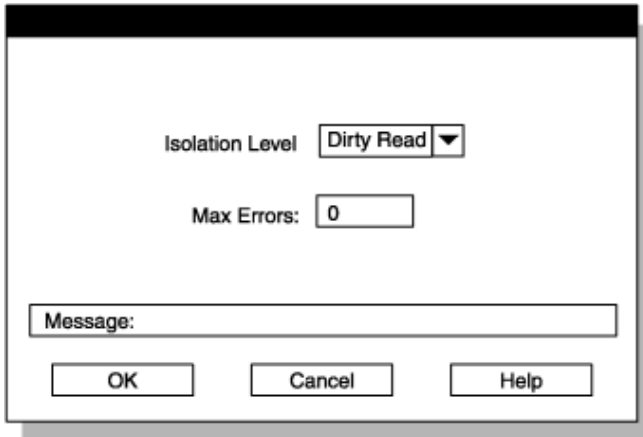
Option	Description
Isolation Level	<p>The criteria for how the query selects records. The four levels of isolation (from highest to lowest) are as follows:</p> <ul style="list-style-type: none"> • Committed • Cursor Stability • Repeatable Read • Dirty Read <p>The higher the isolation level, the lower the unload performance. For a more detailed definition of isolation levels, see the <i>HCL® Informix® Guide to SQL: Syntax</i>.</p>
Max Errors	<p>The maximum number of error conditions to be encountered. If the number of unload errors exceeds this number, the unload job stops.</p>

To change unload options:

1. Display the **Unload Job** window.
See the instructions for [Creating an unload job on page 144](#).
2. Click **Options**.

The **Unload Options** window appears, as the following figure shows.

Figure 102. The Unload Options window



3. Change the options that you want.
4. Click **OK** to return to the **Unload Job** window.

Editing an unload job

After you save an unload job, you can return to the unload job and modify it.

About this task

To edit an unload job:

1. Choose **Jobs > Unload** from the HPL main window.
2. Click **Open** in the **Selection Type** group.
3. Select a job from the **Job Information** list box.
4. Click **OK** to display the **Unload Job** window.
5. Make the appropriate changes to the entries in the **Unload Job** window.
6. Click **Options** to change unload options.
For more information, see [Changing the unload options on page 148](#).
7. Click **Save** to save this unload job.
8. Now you can either run the unload job or exit and run the job later.

Choose from:

- Click **Run** to run the job.
- Click **Cancel** to exit.

Generate options for an unload job

Instead of individually creating the components that are required on the **Unload Job** window, you can use the Generate options to create an unload job.

You can click **Generate** in the **Unload Job** window, or you can choose **Components > Generate** from the HPL main window.

The Generate options do not give you as much flexibility as the **Unload Job** window, but the options let you create the components quickly. In addition, the Generate options let you create formats (Binary, Fixed Internal, and No Conversion) that are not available from the **Format-Definition** window.

Related reference

[The Generate options of the iupload utility on page 160](#)

Load data to a database table

This section describes the load process.

Related information

[Load Job and Unload Job windows on page 52](#)

Load jobs

A load job loads data from a set of one or more files into a single database table. A record format, which defines each field of a record, specifies the layout of the input data. A load map specifies how the record fields are mapped to the columns of the target table. During the load process, the onpload utility converts data from record field to table column.

Components of the load job

The High-Performance Loader (HPL) lets you define the individual components of a data load individually or lets you use the generate option to define the components automatically.

The components of the load job specify:

- The device array where the source data files resides
- The format of the data files
- The filter that accepts or rejects source-file records for the load
- The map that specifies the relationship between the data-file format and the database table schema

When you run a load job, you select which individual components to use. The collection of the various components for a specific load is called the load job. You can assign a name to a load job, save the job, and then retrieve and rerun it as often as you need to. You can modify an existing job or save it under another job name.


You can define as many different load jobs as you need. You can group your load jobs under one or more projects to make the tasks easier to manage.

Related reference

[Define onpladm utility jobs on page 202](#)

Choose the database server

You must run the load job on the *target server*. The target server is the database server that contains the database into which you load the data. The target database must be on the same database server as the onpload program that updates it.

 **Tip:** The onpload database and the ipload interface can be on different computers. You can run the ipload interface on any computer that can connect to the database server that contains the onpload database.

Run multiple jobs

You can run only one express-load job at a time on the same table, although you can run multiple unload jobs concurrently. Because the High-Performance Loader (HPL) is designed to maximize the use of system resources, running concurrent jobs might overload the system.

If you are using a UNIX™ **cron** job to run the load or unload jobs, let one job finish before you start the next.

Prepare user privileges and the violations table

You must make sure that the user who runs a load job has sufficient privileges to manage the constraints and the violations table.

The following table summarizes the actions that you must take. The following sections discuss these actions in more detail.

Table Status	User Privileges	Action
Owned by user		No further action is required.
Not owned by user	User has DBA privileges on the table.	No further action is required.
Not owned by user	User does not have DBA privileges on the table.	User must have: <ul style="list-style-type: none"> • Resource privileges on database. • Alter privileges on table. Owner must start violations table.

For detailed information about user privileges and violations tables, see the *HCL® Informix® Guide to SQL: Syntax* and the *HCL® Informix® Guide to SQL: Reference*.

Set user constraints

To modify any constraint, index, or trigger, a user must have both Alter privileges on the table and the Resource privilege on the database. The user must also have these privileges to start or stop a violations table. You use the GRANT statement to set these privileges.

Manage the violations and diagnostics tables

You can turn on or turn off the generation of constraint-violation information. If you turn on the generation of constraint-violation information, onpload writes the information to the violations and diagnostics tables.

The High-Performance Loader (HPL) manages the violations and diagnostics tables in the following manner:

1. Starts the load job.
2. Starts the violations and diagnostics tables if they do not exist. (If a violations and diagnostics table exists, the HPL uses that table).

The HPL uses the following SQL statement to start the violations table:

```
START VIOLATIONS TABLE FOR tablename
```

3. Performs the load job.
4. Stops the violations and diagnostics tables if they were started at [step 2 on page 152](#).

The HPL uses the following SQL statement to stop the violations and diagnostics tables:

```
STOP VIOLATIONS TABLE FOR tablename
```

5. Drops the violations table if the violations table is empty.

The START VIOLATIONS TABLE statement creates the violations and diagnostics tables and associates them with the load table. The STOP VIOLATIONS TABLE statement dissociates the violations and diagnostics tables from the load table.

The violations table (***tablename_vio***) and the diagnostics table (***tablename_dia***) are always owned by the owner of the table with which they are associated. The Resource privilege lets a user start and stop a violations table, but it does not let the user drop a table that the user does not own. Thus, the HPL cannot drop the violations table in step 5 if the user is not the owner.

Failure to drop the violations table does not cause the load job to fail. However, this failure leaves in the database a violations table that is not associated with a table. If the user tries to run the job again, the START VIOLATIONS TABLE statement in step 2 fails because the table ***tablename_vio*** exists.

To solve this problem, the owner of the table or the database administrator must explicitly create the violations and diagnostics tables by using the START VIOLATIONS statement. When the owner creates the violations table, the following actions take place:

- In [step 2 on page 152](#), the HPL uses the existing violations table.
- In [step 4 on page 152](#), the HPL does not stop the violations table because the table was not started in step 2.
- In [step 5 on page 152](#), the HPL does not drop the violations table because the user does not own the table.

After the load job is complete, an active violations table remains in the database. This table might be empty, but does no harm. When the user runs the load job a second time, the violations table is available, and the load job succeeds.

Related information

[Changing the load options on page 158](#)

[START VIOLATIONS TABLE statement on page](#)

[STOP VIOLATIONS TABLE statement on page](#)

The Load Job windows

You can create, save, and run a load job from the **Load Job Select** window and the **Load Job** window. The **Load Job** window visually represents the various components of a load. After you select the components, you can save the load job for future use or run it immediately.

The ipload utility assigns path names for the log files that document the load and that capture records that do not pass the specified filter or that do not pass conversion.

When you use ipload to create a job, ipload stores information for the job in a row in the **session** table of the onpload database. The ipload utility stores information about the components of the load job in other tables of the onpload database, including **format**, **maps**, **filters**, and so on. When you use the onpload command, columns in the **session** table reference the components to assemble the information necessary for the job.

Related reference

The onpload database

The session table in the onpload database

Creating a load job

Use the **Load Job Select** window to create a load job or select an existing job to edit.

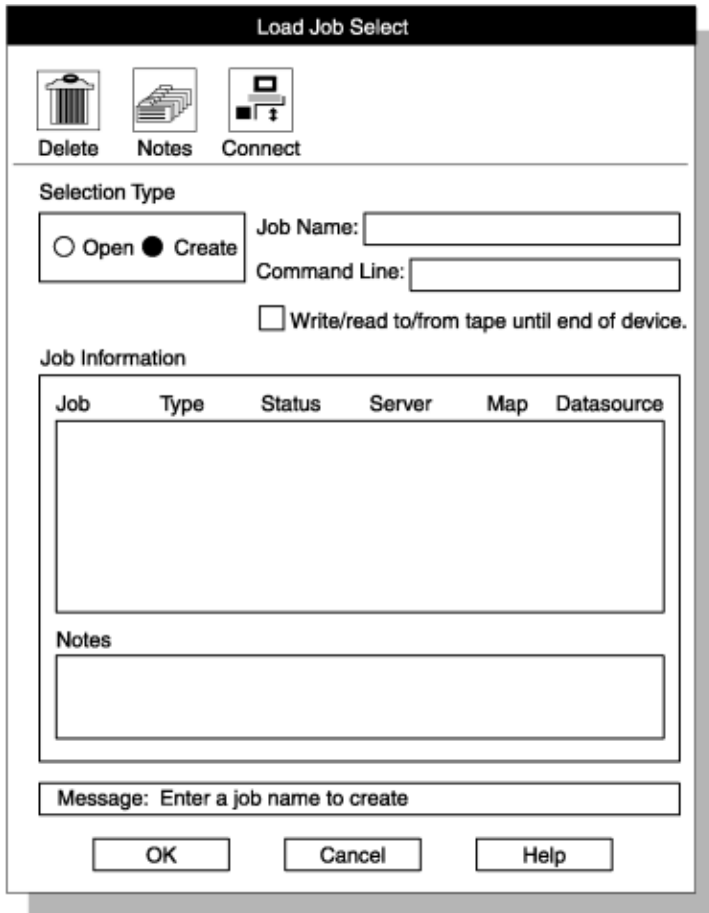
About this task

To create a load job:

1. Choose **Jobs > Load** from the HPL main window.

The **Load Job Select** window appears, as the following figure shows.

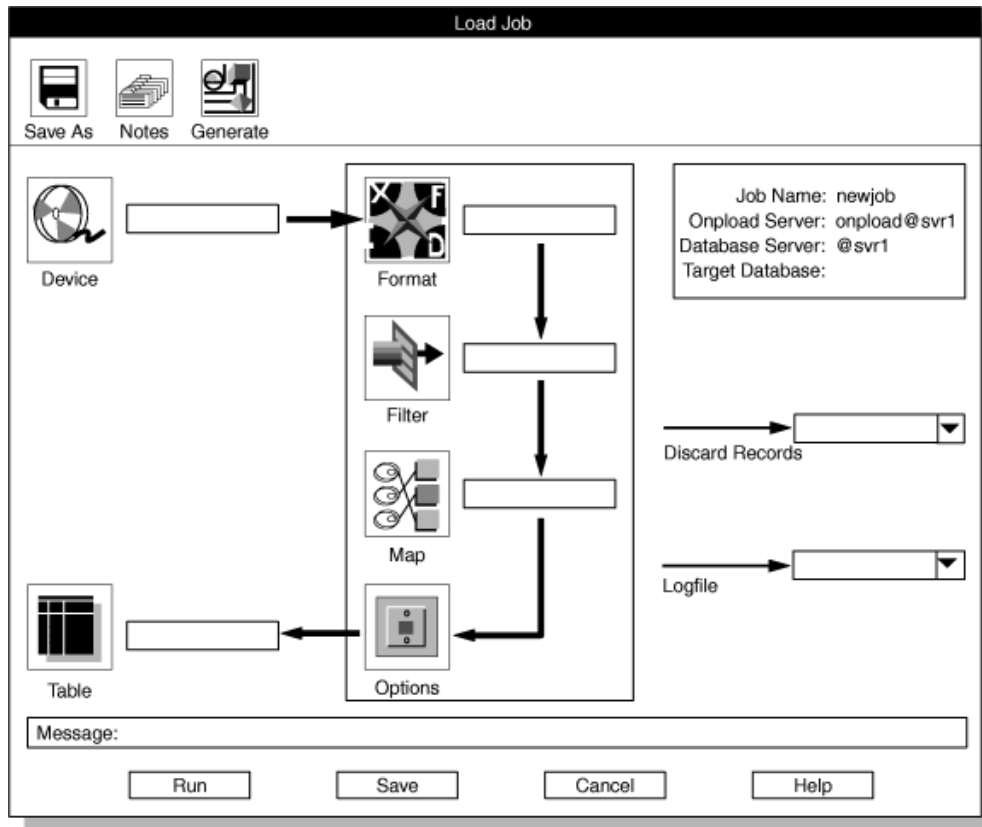
Figure 103. The Load Job Select window



2. Click **Create** in the **Selection Type** group.
3. Select a name for the job and type it in the **Job Name** text box.
4. **Optional:** Check the **Write/read to/from tape until end of device** check box.
For more information, see [Specify to read to the end of the tape on page 157](#).
5. Click **OK**.

The **Load Job** window appears, as the following figure shows.

Figure 104. The Load Job window



6. Type the appropriate values for the components of the load.

The [HPL ipload utility icon buttons on page 63](#) describes the icons that represent the components of the load. For detailed information about these components, see the individual topics on device arrays, formats, filters, and maps.

7. Select a base name for the files that contain rejected records and type it in the **Discard Records** text box.

[Reviewing records that the conversion rejected on page 173](#) gives information about rejected records.

8. Choose a name for the file that contains the load job status log and type it in the **Logfile** text box.

For more information about the log file, see [View the status of a load job or unload job on page 175](#).

9. Click **Options** to change the load options.

For more information about these options, see [Changing the load options on page 158](#).

10. Click **Save** to save this load job.

(If you click **Run** to run the job immediately, the job is saved automatically.)


11. Now you can either run the load job or exit and run the job later.

Choose from:

- Click **Run** to run the job.
- Click **Cancel** to exit to the **Load Job Select** window.

Run the load job

If you click **Run** in the **Load Job** window, the **Active Job** window appears, as [Figure 23: The Active Job window on page 39](#) shows. The **Active Job** window displays the progress of your job and indicates when the job completes. When the **Active Job** window indicates that the load job is complete, click **OK** to return to the **Load Job Select** window.

 **Tip:** Before you run a load job, you might want to view the data-file records according to a specified format to check your definitions.

After you run an express-mode load, you must make a level-0 backup before you can access the table that you loaded.

Related information

[Preview data-file records on page 170](#)

Make a level-0 backup

Express-mode loads do not log loaded data. After an express-mode load, onpload sets the table to read-only as a protective measure. To make the table available for write access, you must do a level-0 backup for all the dbspaces that the fragments of the loaded table occupy. The level-0 backup allows data recovery for the table in case of future database corruption.

If you do not need to provide for data recovery, you can use `/dev/null` as the backup device for the level-0 backup. This strategy makes the table available for write access without actually backing up the data. If a user attempts to write into the table before you make a level-0 backup, the database server issues an error message.

If you run several express-load jobs on different tables in a database, you can complete all of the loads before you perform the level-0 backup. However, if you try to do a second load on the same table without making a level-0 backup, the database server issues an error message.


For discussions of table fragments and dbspaces, see the *HCL® Informix® Administrator's Guide*. For information about making backups, see the *HCL® Informix® Backup and Restore Guide*.

Related information

[How the express and deluxe modes work on page 179](#)

Problems during a load job

If you encounter any problems during the load, examine the various files that onpload creates.

 **Important:** Because of operating-system limitations, the onpload utility cannot load successfully from a file (on disk) that is larger than 2 GB on a platform that is less than a 64-bit platform. If you try to read a file that is larger than 2 GB, onpload HPL fails only after it processes the first 2 GB of data. The HPL log file reports the following error:



Cannot read file `/some_dir/a_long_file` - aio error code 27

Related reference

[The HPL browsing options on page 170](#)

Specify to read to the end of the tape

From the **Load Job Select** window, you can specify to read from the tape until the end of the device with the **Write/read to/from tape until end of device** check box. When a tape device is empty, you are prompted to provide a new tape, until the load job is complete.



Important: If you select this option, you must also select it when unloading from the **Unload Job** window, or specify the **-Z** option from the command line; otherwise, the loaded and unloaded data might be inconsistent.

If you check this check box, it supersedes any tape size information you enter in the **Device-Array Definition** window or at the command line.



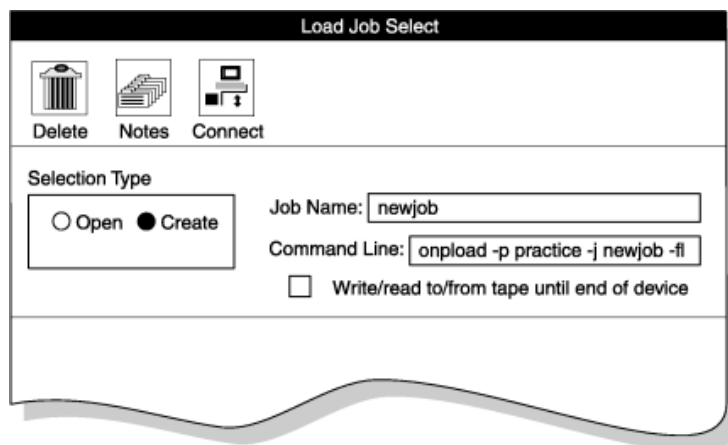
Important: You must provide the same tapes in the same order on the same devices for both the unload jobs and the load jobs to ensure consistency.

The command-line information

If you select an existing job in the **Load Job Select** window, the **Command Line** text box shows the onpload command that ipload generated for that load job.

The following figure shows the **Command Line** portion of a **Load Job Select** window. The **Command Line** text box displays the onpload command generated for the load job that [Figure 21: The Load Job window with all required component boxes completed on page 37](#) shows.

Figure 105. Fragment of the Load Job Select window



The command line, `onpload -p practice -j newjob -fl`, contains the following arguments.

-p practice

The project where the job is stored.

-j newjob

The name of the job.

-fl

The job that loads (rather than unloads) data.

You can copy the `onpload` command from the **Command Line** text box and paste it at a system prompt to run the load job. If you need to run the load job multiple times, you can save the `onpload` command and run it later.

You do not need to start `ipload` to run a job from the system prompt. The `ipload` and `onpload` utilities both use the `onpload` database, but each utility uses it independently.

Changing the load options

Before you begin

Before you run a load job, you can review or change any load options. The load options include specifying the number of records to load, the starting record number, and the loading mode.

About this task

The `ipload` utility stores option information in the **session** table of the `onpload` database.

The **Load Options** window contains the following option text boxes.

Table 17. The Load Options window options

Option	Description
Load Mode	The mode for the load: express, deluxe, or deluxe without replication
Generate Violations Records	Whether or not to generate violations records
Tapes	The number of tapes that contain source data
Number Records	The number of records to process in the data file
Start Record	The record number in the data file from which to start loading
Max Errors	The maximum number of error conditions to be encountered If the number of load errors exceeds this number, the load stops.
Commit Interval	The number of records to load before logging the transaction

Table 17. The Load Options window options (continued)

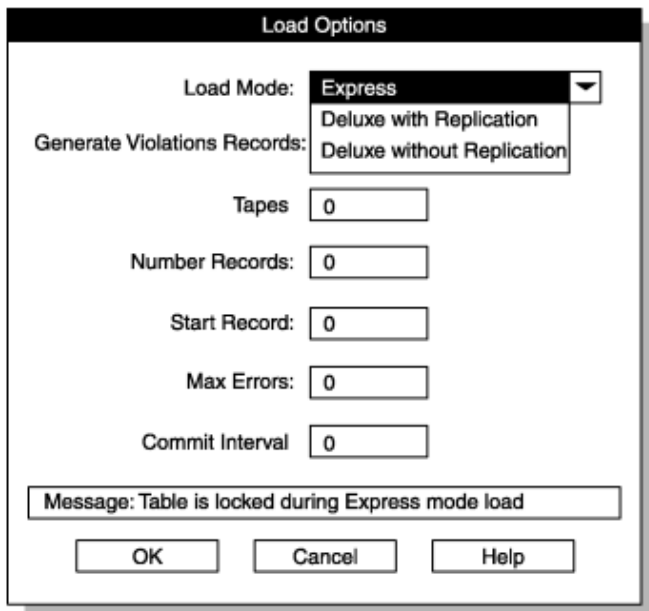
Option	Description
	If you set the commit interval to 0, onpload uses the default value of 10. You can use this option only with deluxe mode.

To change load options:

1. Display the Load Job window.
See [Creating a load job on page 153](#).
2. Click **Options**.

The **Load Options** window appears, as the following figure shows.

Figure 106. The Load Options window



3. Change the options that you want.
4. Click **OK** to return to the **Load Job** window.

Related reference

The onpload database

Related information

[The HPL ipload utility icon buttons on page 63](#)

[Manage the violations and diagnostics tables on page 152](#)

Editing a load job

After you create and save a load job, you can later return and modify that load job.

About this task

To edit a load job:

1. Choose **Jobs > Load** from the HPL main window to display the **Load Job Select** window.
2. Click **Open** in the **Selection Type** group.
3. Select a job from the **Job Information** list box.
4. Click **OK** to display the **Load Job** window.
5. Make appropriate changes to the entries in the **Load Job** window.
6. Click **Options** to change load options.
7. Click **Save** to save this load job.
8. Run or cancel the load, as follows:

Choose from:

- Click **Run** to run the data load.
- Click **Cancel** to exit.

Generate options for a load job

Instead of individually creating the components that are required on the **Load Job** window, you can use the Generate options to create a load job.

You can click **Generate** in the **Load Job** window, or you can choose **Components > Generate Job** from the HPL main window.

The Generate options do not give you as much flexibility as creating each component individually, but these options let you create the components quickly. (After you generate the components, you can edit the components individually by accessing them through the **Components** menu.) In addition, the Generate options let you create formats (Fixed Internal and No Conversion) that are not available from the **Format-Definition** window.

Related reference

[The Generate options of the iupload utility on page 160](#)

The Generate options of the iupload utility

This section describes the Generate options for the iupload utility.

Related reference

[The iupload utility Generate options on page 40](#)

[Generate options for an unload job on page 149](#)

[Generate options for a load job on page 160](#)

Related information[Formats on page 85](#)[Components of the unload job on page 142](#)

Overview of the ipload Generate options

Use the generate options of ipload to automatically generate components of a load or unload job. The generate options can save you time when you create formats, maps, queries, and load and unload jobs.

When you generate a load or unload job for a database, ipload creates a format for the data file and a map that associates the columns of the table with the fields of the data-file records. Although the generated components might not match your database schema or data-file records exactly, the components created by the generate options provide useful starting points for building HPL components. After you generate default components, you can modify the components to match your specific needs.

Tasks that generate load or unload components

Use the ipload utility to perform the following tasks:

- Generate load components from the **Load Job** window.
- Generate unload components from the **Unload Job** window.
- Generate both load and unload components from the **Components** menu.

Generate from the Load Job window

Use the **Generate** button in the **Load Job** window to save time when the format of the data file corresponds to the format of the database table.

When you generate from the **Load Job** window, ipload makes the following assumptions about the file (or device array) that contains the data:

- The file is an ASCII file.
- The file uses the same locale as the database.
- The file uses a vertical bar (|) for the field delimiter and a new line for the record delimiter.
- The fields in each record of the file correspond one-to-one to the columns of the target table.
- All records in the file should be loaded.

Generating a job from the Load Job window

When you generate from the **Load Job** window, ipload creates a format, a map, a job, and, if needed, a device array.

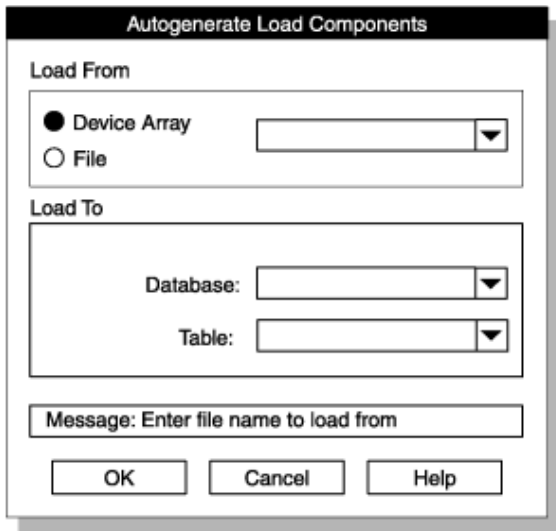
About this task

To generate a job from the **Load Job** window:

1. Choose **Jobs > Load** from the HPL main window to display the **Load Job Select** window.
2. Click **Create** in the **Selection Type** group.
3. Select a name for the load job and type it in the **Job Name** text box.
4. Click **OK** to display the **Load Job** window.
5. Click **Generate**.

The **Autogenerate Load Components** window appears, as the following figure shows.

Figure 107. The Autogenerate Load Components window



6. Click **Device Array** or **File** to indicate the location of the source data

Choose from:

- To load from an existing device array, click **Device Array** and type the name of the device array.
- To load from a file, click **File** and type the full path name of the file. The ipload utility automatically generates a device array that includes the file.

7. In the **Load To** group, type the name of the database and table that will receive the data.
8. Click **OK** to generate the components of the load and return to the **Load Job** window.
9. If needed, click **Filter** to prepare a filter.
10. If you want, change the path names in the **Discard Records** and **Logfile** text boxes.
11. Click **Save** to save the components and the job.
12. Click **Run** to run job or **Cancel** to exit.

Generate from the Unload Job window

Use the **Generate** button in the **Unload Job** window to save time when the format of the data file is similar to the format of the database table.

When you generate from the **Unload Job** window, ipload makes the following assumptions about the file (or device array) into which the data is unloaded:

- The file is an ASCII file.
- The file uses the same locale as the database.
- The file uses a vertical bar (|) for the field delimiter and a new line for the record delimiter.

Generating a job that uses a query

When you generate from the **Unload Job** window, ipload creates a format, a map, a job, and, if needed, a device array. You can generate an unload job that uses a query to select from one or more tables or that unloads an entire table.

About this task

To generate a job that uses a query

1. Follow the instructions in [Creating a query on page 109](#) to create a query.
2. Choose **Job > Unload** from the HPL main window to display the **Unload Job Select** window.
3. Click **Create** in the **Selection Type** group.
4. Select a name for the unload job and type it in the **Job Name** text box.
5. Click **OK** to display the **Unload Job** window.
6. Click **Generate**.

The **Autogenerate Unload Components** window appears, as the following figure shows.

Figure 108. The Autogenerate Unload Components window

The screenshot shows a dialog box titled "Autogenerate Unload Components". It is divided into two main sections: "Unload From" and "Unload To".

- Unload From:**
 - There are two radio buttons: "Table" (which is selected) and "Query".
 - Next to "Table", there is a "Database:" label and a dropdown menu showing "stores_demo".
 - Next to "Query", there is a "Table:" label and a text box containing "customer".
 - Below the "Table:" text box is a "Query:" label and an empty dropdown menu.
- Unload To:**
 - There are two radio buttons: "Device Array" and "File" (which is selected).
 - Next to "Device Array" is a dropdown menu showing "/work/cust_out".

At the bottom of the dialog, there is a message box that says "Message: Enter file name to unload into". Below the message box are three buttons: "OK", "Cancel", and "Help".

7. Click **Query** in the **Unload From** group.
8. Enter the name of the query.
You can use the down arrow to see selection lists. When you unload from a table, you do not enter a query.
9. Click **Device Array** or **File** in the **Unload To** group.

Choose from:

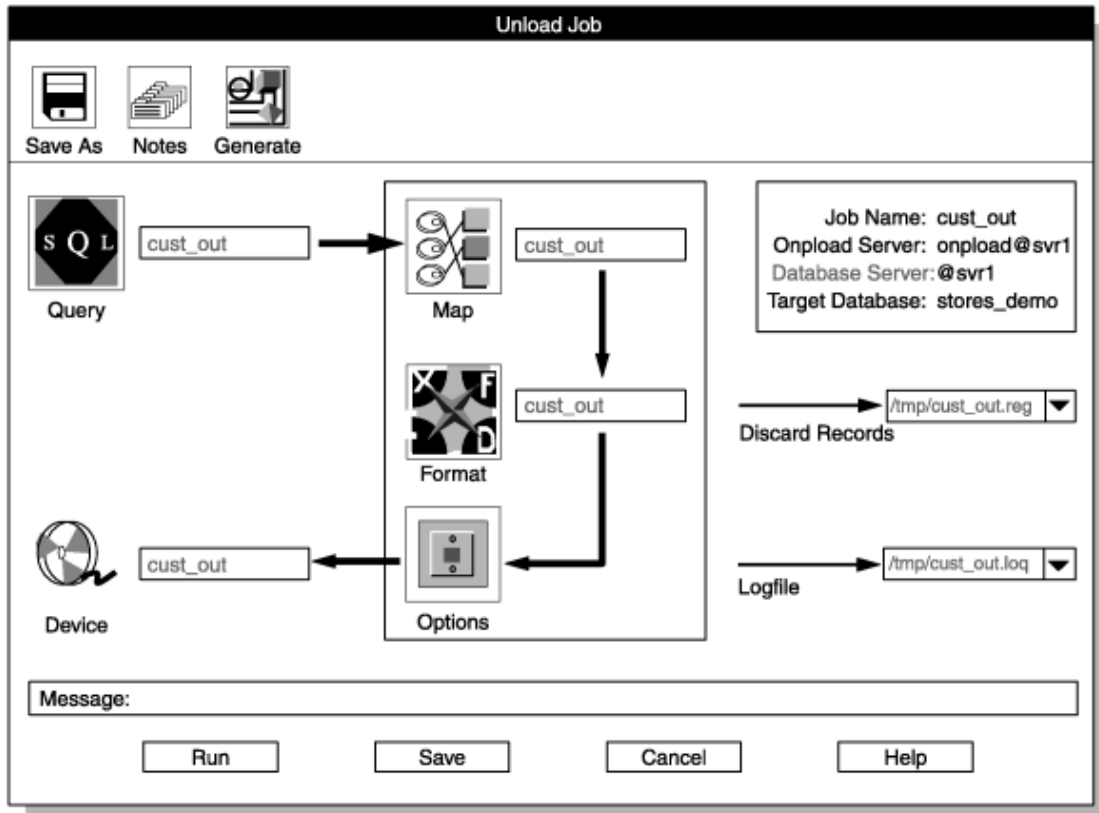
- If you click **Device Array**, you can use the down arrow to see a list of the available device arrays.
- If you click **File**, ipload creates a device array of the same name as the unload job and inserts the specified file into that device array.

! **Important:** If you are executing onpload from the command line during an unload job, you must first create the file.

10. Click **OK** to generate the components of the unload job.

The display returns to the **Unload Job** window. The ipload utility completes the **Unload Job** window. If you chose **cust_out** for the unload job name (step 4 on page 163), the **Unload Job** window appears as the following figure shows.

Figure 109. The Unload Job window



11. Click **Save** to save this Unload Job.

You can click **Run** to run the job, or click **Cancel** to exit and run the job later.

12. To run the job, click **Run** to display the **Active Job** window.

13. When the **Active Job** window displays **Job Completed**, click **Cancel** to return to the main HPL window.

Generating a job that unloads an entire table

About this task

To generate a job that unloads an entire table:

1. Choose **Jobs > Unload** from the HPL main window to display the **Unload Job Select** window.
2. Click **Create** in the **Selection Type** group.
3. Choose a name for the unload job and type it in the **Job Name** text box.
4. Click **OK** to display the **Unload Job** window.
5. Click **Generate**.

The **Autogenerate Unload Components** window appears, as [Figure 108: The Autogenerate Unload Components window on page 163](#) shows.

6. Click **Table** in the **Unload From** group.
7. Enter the database that you want in the **Database** text box and the table that you want in the **Table** text box.
You can use the down arrows to see selection lists. When you unload from a table, you do not enter a query.
8. Click **Device Array** or **File** in the **Unload To** group.
Choose from:
 - If you click **Device Array**, you can use the down arrow to see a list of the available device arrays.
 - If you click **File**, ipload creates a device array of the same name as the unload job and inserts the specified file into that device array.
9. Click **OK** to generate the components of the unload job.
The **Unload Job** window reappears, with the components of the job completed.
10. Click **Save** to save this Unload Job.
You can click **Run** to run the job, or click **Cancel** to exit and run the job later.
11. To run the job, click **Run** and the **Active Job** window appears.
12. When the **Active Job** window displays `Job Completed`, click **Cancel** to return to the main HPL window.

Generate from the Components menu

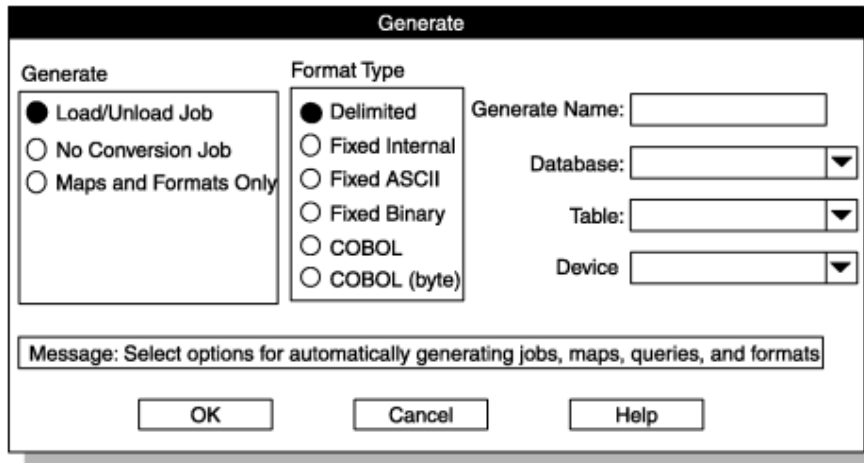
To generate all of the components for both load and unload jobs in one operation, choose **Components > Generate Job** from the main HPL window. This Generate option lets you choose formats that are not available from the **Format-Definition** window.

The Generate window

You can specify the characteristics of the components that ipload creates from the **Generate** window.

The following figure shows the **Generate** window. This window generates all of the components required for a load job and an unload job: format, load map, unload map, query, and device array.

Figure 110. The Generate window



Generate group

The **Generate** group specifies the type of generate to perform.

The **Generate** group has the following choices.

Choice	Effect	See
Load/Unload Job	Generates both load and unload jobs	Generating load and unload components on page 167
No Conversion Job	Generates a job that treats an entire database record as one entity	Using the No Conversion Job option on page 169
Maps and Formats only	Generates only a format, a load map, and an unload map	


Format type group

The **Format Type** group specifies the format of the data file.

The **Format Type** group has the following choices.

Choice	Description	See
Delimited	The fields of a data-file record are separated by a field delimiter, and records are separated by a record delimiter. The default delimiter for fields is a vertical bar (). The default delimiter for records is a new line.	Modifying delimited-format options on page 104

Choice	Description	See
Fixed Internal	The data file uses HCL Informix® internal format. The only changes to the data that you can make when you use this format are ALTER TABLE changes: modify the order of columns, delete or add columns, or change the data type. The HPL loads and unloads data in this format more efficiently than data in the Delimited and Fixed ASCII formats.	Other formats on page 103
Fixed ASCII	All records are the same length. Each record contains characters in fixed-length fields. This format is the same as the Fixed format choice of the Record Formats window.	Fixed-length records on page 86
Fixed Binary	The data file records contain data in fixed-length fields. Character-oriented data is in character fields. Numeric data (integer, float, and so on) is in machine-dependent binary values. Use this format for loading or unloading data for an application that has or requires data in binary format. Data in binary format is much more compact than data in ASCII format.	Fixed-length records on page 86
COBOL	The data file is formatted according to COBOL 86 standards. All COBOL data types are supported.	COBOL records on page 100
COBOL (byte)	The data file is formatted with byte alignments for COMP-4 data type. All COBOL data types are supported.	

 **Tip:** To generate EBCDIC data, select the Delimited or Fixed ASCII format and use the format options to change the code set.

Related reference

[Format options on page 103](#)

[Alter the schema of a table on page 185](#)

Generating load and unload components

When you choose **Components > Generate**, you can generate all of the components required for a load job and an unload job: format, load map, unload map, query, and device array.

About this task

To generate the components for loading or unloading a database:

1. Choose **Components > Generate Jobs** from the main HPL window to display the **Generate** window.
2. Click **Load/Unload Job** in the **Generate** group.
3. Select a format for the data file in the **Format Type** group.
4. Select a name for the generated components and type it in the **Generate Name** text box.
This name is used for each of the components that this option creates.
5. Type the name of the database that you will load or unload in the **Database** text box. Or, click the down arrow to select a database from the selection list.
6. Type the name of the table within the database in the **Table** text box. Or, click the down arrow to select a table from the selection list.
7. Type the name of a device array in the **Device** text box. Or, click the down arrow to select a device from the selection list.

If you enter the name of a device (file) instead of a device array, ipload creates a device array of the same name as the unload job and inserts the specified device into that device array.

When you specify a file instead of a device array in the **Device** text box, ipload makes the following assumptions about the data file:

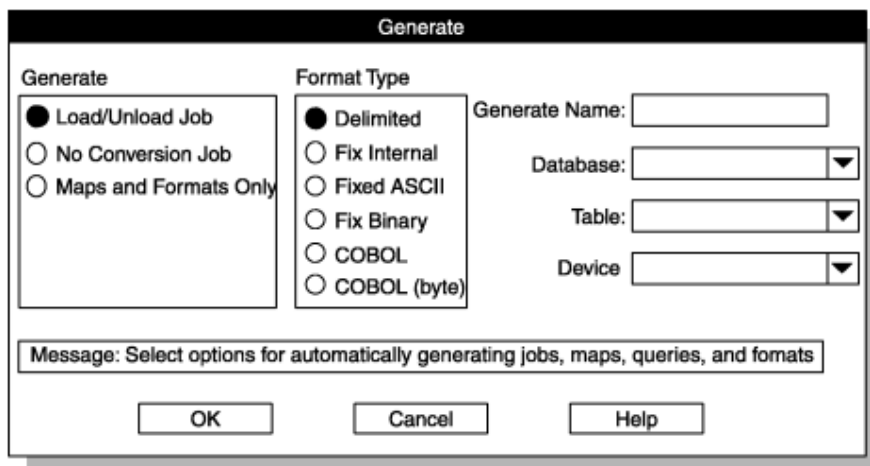
- The file is an ASCII file.
- The file uses the same locale as the database.
- The file uses a vertical bar (|) for the field delimiter and a new line for the record delimiter.
- The fields in the data file are in the same order as the columns of the target table.

8. Click **OK**.

Results

The following figure shows appropriate choices for generating load and unload jobs for delimited output from the **state** table of the **stores_demo** database. After ipload creates the components, you can run the job or use the **Component-Definition** windows to make any necessary changes.

Figure 111. The Generate window



For information about generating a job from a .csv file, see [Testing the import of a CSV file on page 106](#).

Using the No Conversion Job option

The **No Conversion Job** option uses the HCL Informix® internal format to unload data from a table. Jobs loaded or unloaded with this option are sometimes called raw loads or raw unloads.

About this task

The **No Conversion Job** option treats an entire database record as one entity by using the Informix® internal format. It does not generate formats or maps. The **No Conversion Job** option is the fastest option that you can use for loading and unloading data. Use this option to transport data or when you need to reorganize the disks on your computer. No-conversion jobs are always completed in express mode. You cannot use the onpload command line to convert the running job to deluxe mode when using a no-conversion job.

When you run a job that you created with the **No Conversion Job** option, ipload displays a **Fast Job Startup** window instead of the usual **Load Job** or **Unload Job** window.

To use the **Fast Job Startup** window:

1. Choose **Components > Generate Job** from the main HPL window.
2. Click **No Conversion Job** in the **Generate** group.
3. Select a name for the job and type it in the **Generate Name** text box.
4. Type the names of the database, table, and device array in the **Database**, **Table**, and **Device** text boxes.
5. Click **OK**.

The display returns to the HPL main window.

6. Choose **JobsLoad** (or **Jobs > Unload**) from the main HPL window.

The **Load Job Select** or **Unload Job Select** window appears.

7. Select the job from the **Job Information** list box.
8. Click **OK**.

The **Fast Job Startup** window appears, as the following figure shows.

Figure 112. The Fast Job Startup window for a load job

The screenshot shows a dialog box titled "Fast Job Startup". At the top, it displays "Job Name: a_fast_job". Below this are three dropdown menus: "Database:" with the value "stores_demo", "Table:" with the value "customer", and "Device Array" with the value "some_array". At the bottom of the dialog, there is a message box that says "Message: Enter database to unload". Below the message box are three buttons: "Run", "Cancel", and "Help".

9. Click **Run** to run the job and the **Active Job** window appears.
10. When the **Active Job** window displays `Job Completed`, click **Cancel** to return to the main HPL window.

Related reference

[Settings for a no-conversion load or unload job on page 186](#)

The HPL browsing options

This section describes the browsing options that are available for the High-Performance Loader (HPL).

Related reference

[Problems during an unload job on page 146](#)

[Problems during a load job on page 156](#)

Browsing options

Use the browsing options of the High-Performance Loader (HPL) to preview records from the data file, review records after you perform a load or unload job, review various files associated with the HPL, and view the status of a load or unload job.

Related reference

[View error records on page 182](#)

Related information

[The Browse button on page 58](#)

Preview data-file records

Before you actually run a load job, you can use the **Record Browser** window to check your definition of the format. The display shows errors such as incorrect field lengths or missing fields. You can edit the format to correct your format definitions.

Related reference

[Run the load job on page 156](#)

Related information

[Formats of supported datafile records on page 85](#)

[Editing a format on page 91](#)

Using the Record Browser window

You can review records in a specified format, search the list of available formats, or edit a format from the **Record Browser** window.

About this task

Reviewing data-file records in a selected format

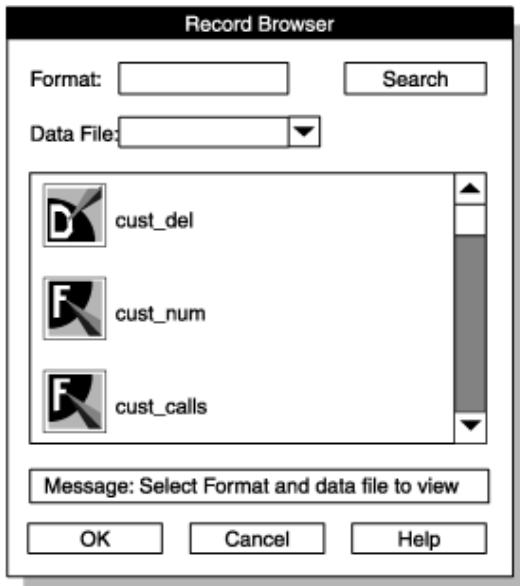
About this task

To review data-file records in a selected format:

1. In the HPL main window, select the project that contains your load job.
2. Choose **Browsers > Record**.

The **Record Browser** window appears, as the following figure shows.

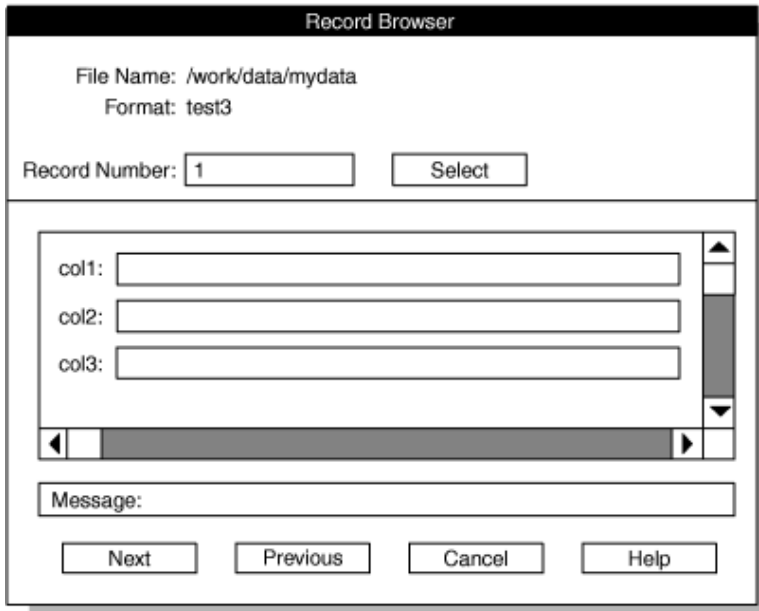
Figure 113. The Record Browser window



3. Type the name of the format to be applied to the source data file or click the format name in the list box.
4. In the **Data File** text box, type the name of the data file that you plan to load, or click the down arrow and select a file from the selection list.
5. Click **OK**.

The second **Record Browser** window appears, as the following figure shows. This **Record Browser** window displays each of the fields in the format, followed by the value of the field for the given Record Number.

Figure 114. The Record Browser window



6. You can take the following actions:

Choose from:

- Type the record number that you want to view. Click **Select**.
- Click **Next** to display the next record.
- Click **Previous** to display the previous record.

7. When you finish browsing, click **Cancel** to return to the HPL main window.

Searching and editing a format

About this task

To search for and edit a format:

1. In the HPL main window, select the project that contains your load job.
2. Choose **Browsers > Record** to display the **Record Browser** window.
3. In the **Format** text box, type the format name or partial format name that you want to find.
You can use wildcards (for example, **cust**).
4. Click **Search**.
The *ipload* utility displays all formats of the current project that include the letters *cust*.
5. Click **Cancel** to return to the HPL main window.

Editing a format

About this task

To edit a format:

1. Select the project that contains your load job.
2. Choose **Browsers > Record** from the HPL main window.
3. Click a format button to edit the format.

The ipload utility displays the **Format-Definition** window.

Related information

[Editing a format on page 91](#)

Reviewing records that the conversion rejected

When you run a load job, onpload creates a file that contains information about records of the data file that the conversion rejected.

About this task

This file is named *basename.rej* where *basename* is the base name that you selected in [step 7 on page 155 of Creating a load job on page 153](#). When you use a generate option to create the components for a load job, *basename* is / *tmp/jobname* where *jobname* is the name that you selected for the unload job.

To review rejected records:

1. On the HPL main window, select the project that contains your load job.
2. Choose **Browser > Record** to display the **Record Browser** window.
3. Type the name of the format to be applied to the rejected-records file in the **Format** text box. Or, click the format name in the list box.
4. Type the name of the rejected-records file in the **Data File** text box. Or, click the down arrow and select a data file from the selection list.
5. Click **OK** to display the second **Record Browser** window.
6. You can take the following actions:
 - **Choose from:**
 - Type the record number that you want to view. Click **Select**.
 - Click **Next** to display the next record.
 - Click **Previous** to display the previous record.
7. Click **Cancel** to return to the HPL main window.

Viewing the violations table

The load job also creates two tables in the target database that contain information about records that passed the conversion but that the database server rejected. The tables are named **tablename_vio** and **tablename_dia**, where *tablename* is the name of the table being loaded.

About this task

View the violations table to browse through records that passed the filter and conversion but that the database server rejected. The High-Performance Loader (HPL) writes these records into the violations table (**tablename_vio**). The data in the violations table has the same format as the database table.

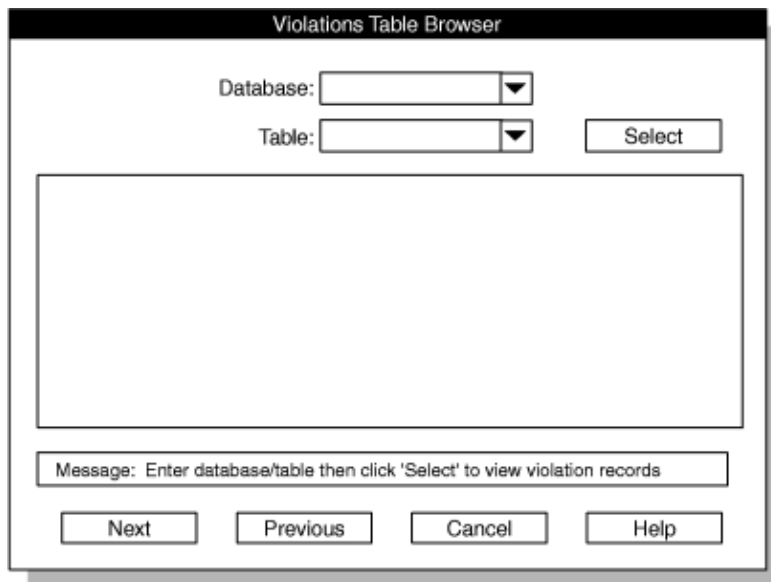
The *HCL® Informix® Guide to SQL: Syntax* discusses in detail the information found in the violations table.

To view the violations table:

1. Choose **Browsers > Violations** from the HPL main window.

The **Violations Table Browser** window appears, as the following figure shows.

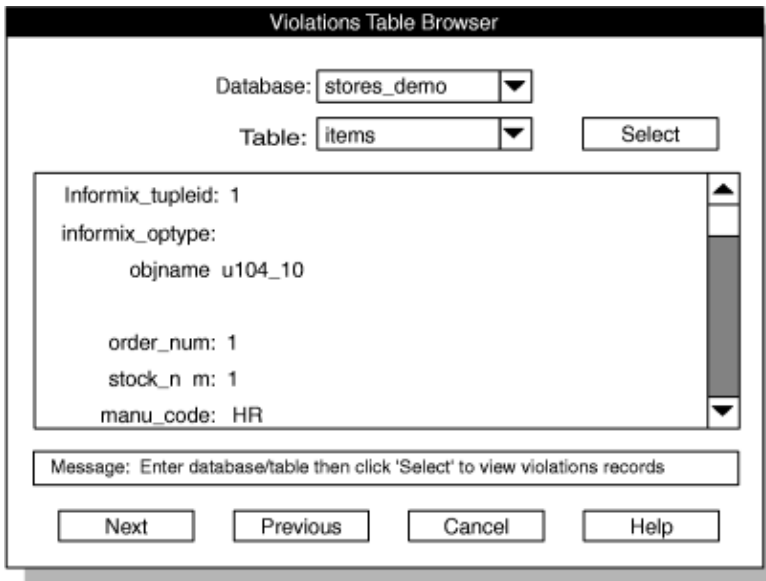
Figure 115. The Violations Table Browser window



2. Type the name of the database and table that you want to review the violations or click the down arrows to make your choices from selection lists.
3. Click **Select**.

The following figure shows the first record of a violations table.

Figure 116. A record in the Violations Table Browser window



4. Click **Next** and **Previous** to move forward and backward through the violations table.
5. Click **Cancel** to return to the HPL main window.

View the status of a load job or unload job

When a load or unload job is complete, onpload writes a record of the load or unload job into a log file.

Related reference

[Run the unload job on page 146](#)

The HPL log-file and pop-up messages

Viewing the log file

The default name for a log file is `/tmp/jobname.log`, where *jobname* is the name that you chose for the job. You can specify a different name for the log file in the **Load Job** window or **Unload Job** window.

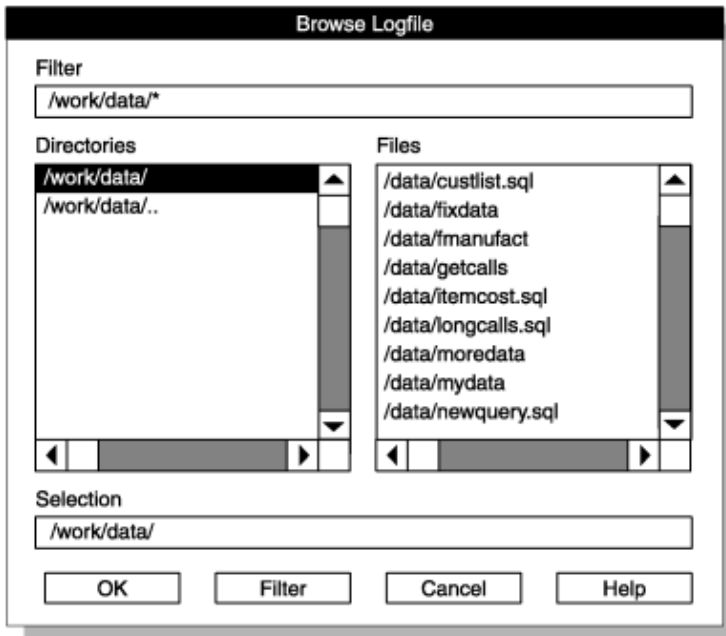
About this task

To view the log file:

1. Choose **Browsers > Logfile** from the HPL main window.

The **Browse Logfile** window appears, as the following figure shows. When the window appears, the **Filter** text box and the **Selection** text box show the directory from which ipload was started.

Figure 117. The Browse Logfile window



2. In the **Filter** text box, type the full path name of the directory that contains the log.
You can use wildcards to select only certain files from that directory.
3. Click **Filter**.

The **Files** list box shows a list of the files that match the path that you entered in the **Filter** text box.

4. In the **Files** list box, click the name of the file that you want to examine.
The full path name of the selected file appears in the **Selection** text box.
5. Click **OK**.

A **Browse** window appears that displays the contents of the selected file.

6. Review the log with the scroll bar to move through the log.
7. Click **Cancel** to return to the HPL main window.

Results

Alternatively, if you know the full path name of the log file, you can simply type the path name in the **Selection** text box and click **OK**.

Sample log file

The following example shows a sample log file entry:

```
Sat Mar 11 13:52:42 1995

Session ID 1

Unload Database -> stores_demo
Query Name -> f_manufact
```



```

Device Array-> fmanufact
Query Mapping-> f_manufact
Query-> select * from manufact
Convert Reject  -> /work/data/f_manu_unl

Database Unload Completed -- Unloaded 9 Records  Detected 0 Errors

Sat Mar 11 13:52:50 1995

```

You can review the log file to determine load status and to see where any errors occurred. The log file is a simple ASCII file. You can print it if necessary.

Manage the High-Performance Loader

Related information

[HPL loading modes on page 7](#)

Manage modes, errors, and performance

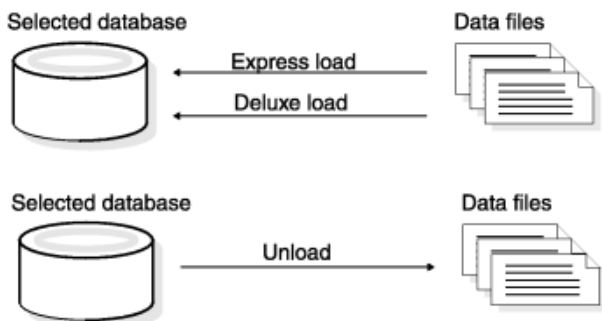
When you manage the High-Performance Loader (HPL), you manage modes, violations (errors), and performance. You also need to avoid the limitation that occurs when using an Excalibur Text DataBlade® module index

HPL modes

The High-Performance Loader (HPL) offers two load modes, deluxe mode and express mode, and one unload mode. The express mode is faster, and the deluxe mode is more flexible.

The following figure shows the load and unload modes of the HPL.

Figure 118. The load and unload modes of the HPL



Related information

[Express-mode limitations on page 183](#)

The HPL deluxe mode

The deluxe mode has the following features:

- Performs row-by-row referential and constraint checking as the data is loaded
- Allows loading of data while other users are working (no table locking) if the table already has an associated violations table before the load and if either of the following conditions apply:
 - The index is set to FILTERING WITHOUT ERROR mode
 - The table does not have unique indexes.
- Allows users to access and update the table during a load

Loaded data is immediately visible to the user.

- Logs data, but also offers a no-logging option
- Updates indexes
- Evaluates triggers
- Sets constraints to FILTERING WITHOUT ERROR
- Sets the isolation mode as if for an insert cursor
- Simulates an INSERT statement, except that the HPL allows the load to handle parallel data streams
- Allows loading of data without replication

The deluxe mode has the following limitations:

- Does not support loads without conversion
- Does not support loads that have conversion but do not generate a violations table
- Does not support the **-fv** option

When you use the deluxe mode without replication to load data into the table that was used to define the replicates, the data in that table is not replicated.

Related information

[The onpload utility deluxe-mode process on page 14](#)

[Comparison between an express-mode and a deluxe-mode load operation on page 180](#)

The HPL express mode

The express mode for loading is faster than the deluxe mode, but has some limitations

The express mode has the following features:

- Locks tables for exclusive use by the load utility
- Disables referential and constraint checking on the table
- Requires a level-0 backup
- Sets all objects (such as indexes or constraints) to disabled before loading
- Re-enables all objects after loading, if possible, and flags objects that cannot be re-enabled in the violations and diagnostic tables
- Supports loading of raw tables
- Supports the loading of logged databases and ANSI mode databases

However:

- If you load ANSI mode databases in express mode, you cannot use DB-Access to select data from the database tables.
- If you load logged databases in express mode, you must perform a level-0 backup before you can write to the target database.

The express mode does not support:

- Tables that contain smart large objects (BLOB or CLOB data types)
- Tables that contain simple large objects (TEXT and BYTE) or extended data types
- Loading of data on a heterogeneous data replication (HDR) replicated table from a database that has transactions
- A table that has a fragmentation strategy and has a WITH ROW IDs clause to enable by using row IDs with fragmentation
- Tables with primary key constraints when child table records refer to the load table
- Rows that are larger than the system page size
- The static-hash access method
- Triggers on the loaded data, because express mode cannot invoke the triggers



Important: If your load job has any of these conditions, you must use deluxe mode to load your data.

Related information

[The onpload utility express-mode load process on page 15](#)

[Express-mode limitations on page 183](#)

[Comparison between an express-mode and a deluxe-mode load operation on page 180](#)

How the express and deluxe modes work

A deluxe-mode load simulates an INSERT statement, except that the High-Performance Loader (HPL) allows the load to handle parallel data streams.

The sequence of events when you run an express-mode load is as follows:

1. The onpload utility locks the table with a shared lock.

Other users can read data in existing rows.

2. The onpload utility creates new extents and fills them with the new rows. However, onpload does not update the database structures that track extents.

The new extents are not visible to the user.

3. At the end of the express load, onpload updates the internal structures of the database.
4. The onpload utility sets the table to read-only.

This setting occurs because in express mode onpload does not log data, and therefore, the table is in an unrecoverable state.

5. The onpload utility unlocks the table and enables the constraints. The new rows become visible to the user for read only.
6. The user performs a level-0 backup.
7. Your database server sets the table to read/write.

If the load fails, onpload discards the extents and clears the internal information that says the table is unrecoverable.

Related information

[Make a level-0 backup on page 156](#)

[INSERT statement on page](#)

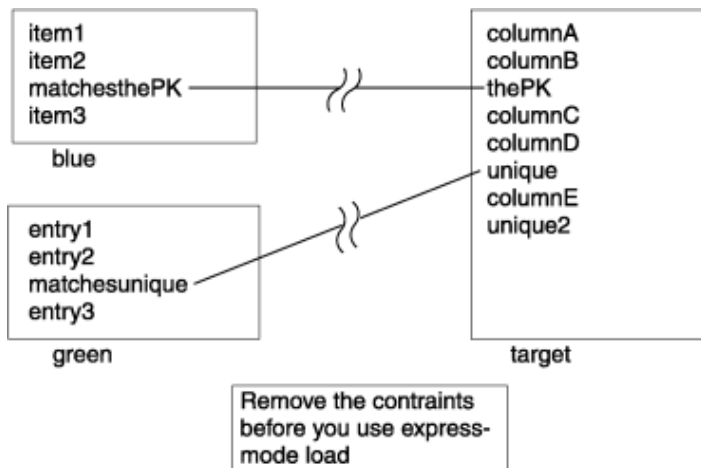
[The INSERT statement on page](#)

Foreign-key constraints

Express® mode cannot disable primary constraints or unique constraints that are referenced as foreign keys that are active on other tables. If you want to load data into such a table, you must first use SET CONSTRAINTS DISABLED statements to disable the foreign-key constraints in the referencing table or tables. After the load is finished, re-enable the foreign-key constraints.

The following figure shows an example of foreign-key constraints. The table **target** has a primary key (**thePK**) and a unique key (**unique**) that table **blue** and table **green** reference. Before you perform an express-mode load into the table **target**, you must disable the foreign-key constraints in both table **blue** and table **green**.

Figure 119. Foreign-key constraints



Comparison between an express-mode and a deluxe-mode load operation

The following table contrasts the operation of an express-mode and a deluxe-mode operation.

Express® mode action	Performed by	Deluxe mode action	Performed by
Set objects to disabled	onpload	Set constraints to filtering	onpload
Load records from the data file into the table (including rows that would cause violations if constraints were on)	onpload	Load records from data file into the table. Write records that violate constraints into the violations table and not into the target table	onpload
Enable objects. Detect violators and copy them into the _vio table.	onpload	Set constraints to non-filtering	onpload
Complete a level-0 backup to make the database writable	user		
Resolve violators	user	Resolve violators	user

Related information

[The HPL deluxe mode on page 177](#)

[The HPL express mode on page 178](#)

HPL load and unload errors

When you load records from a data file, some of the records might not meet the criteria that you established for the database table.

For example, the data file might contain:

- Null values where the table specifies NOT NULL
- Values in an incorrect format (for example, alphabetic characters in a numeric field)
- Records that do not have the expected number of fields

The way that the HPL treats these errors depends on the mode (deluxe or express) and the type of job (load or unload).

The HPL separates errors into the following two classes:

- Rejected records from the input file

These records include:

- Records that the filter rejected
- Records that cannot be converted
- Constraint violations

Rejected records from the input file

Input-file records that the High-Performance Loader (HPL) rejects because they could not be converted include records in an incorrect format, records that do not have the expected number of fields, and records whose fields contain null values for columns that do not allow null values.

The onpload utility writes these records into a file with the suffix `.rej`. The onpload utility writes records that are rejected because they do not match the filter criteria into a file with the suffix `.flt`.

Constraint violations

When the onpload utility starts a deluxe-mode load, it runs the following SQL statement:

```
SET CONSTRAINTS ON mytable FILTERING
```

This statement has two results:

- The utility adds two tables, **mytable_vio** and **mytable_dia**, to the database that contains **mytable**.
- All of the constraints that are associated with the table are set to filtering. Filtering mode causes any record that does not meet the constraint requirements to be added to the **mytable_vio** table instead of generating an error.

The use of filtering mode for constraints is covered in detail in the *HCL® Informix® Guide to SQL: Syntax*.

View error records

Choose from the **Browsers** menu in the HPL main window to view the error records that onpload generates.

Related information

[Browsing options on page 170](#)

HPL performance

You can improve the HPL performance by preparing an environment that is optimized for the particular load or unload job that you are performing.

You must consider the following aspects of your load and unload jobs:

- Configuration-parameter values
- Mode (express or deluxe)
- Devices for the device array
- Usage models

Related reference

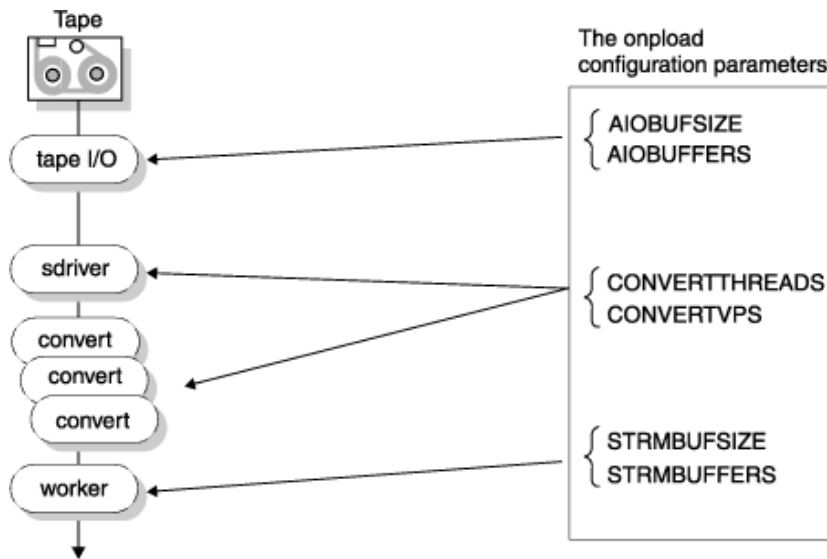
[Configure the ipload utility on page 70](#)

The onpload configuration parameters

The onpload configuration parameters control the number of threads that onpload starts and the number and size of the buffers that are used to transfer data.

The following figure shows which part of the onpload process is affected by each configuration parameter.

Figure 120. The onpload configuration parameters



The `AIOBUFSIZE` and `AIOBUFFERS` parameters control the number and size of the buffers that onpload uses for reading from the input device. `CONVERTTHREADS` and `CONVERTVPS` control the amount of CPU resources to apply to data conversion. `STRMBUFSIZE` and `STRMBUFFERS` control the number and size of the buffers used to transport data between onpload and the database server.

The onpload configuration parameters are stored in the following files:

- `$INFORMIXDIR/etc/$PLCONFIG` (UNIX™)
- `%INFORMIXDIR%\etc\%PLCONFIG` (Windows™)

Related reference

[HPL usage tasks on page 185](#)

[HPL performance hints on page 187](#)

High-Performance Loader configuration file

Express-mode limitations

You cannot use express mode in certain situations. For example, express mode does not support the loading of simple large objects (Simple LOs) or Ext Type Data and smart-large-object data (Ext Type data types), ensuring constraints, or invoking triggers.

In addition, you cannot use DB-Access to select data from ANSI database tables if the data was loaded in express mode. If you attempt to select data from such tables, the database server rejects the SELECT statements because the tables were not archived. Only read-only access is allowed to the tables until a level-0 archive is performed.

Related information

[The HPL express mode on page 178](#)

[HPL modes on page 177](#)

The onstat options for onpload

The onstat utility has two options that you can use to observe the behavior of database server threads during express-mode loads.

Use the following command to display light-append information:

```
onstat -g lap
```

The onstat -j option provides an interactive mode that lets you gather special information about an onpload job.

Related reference

[The onstat -j option](#)

Devices for the device array

On a data-load job, each device runs independently of other devices. Thus mixing fast and slow devices does not adversely affect the speed of the load.

In most unload jobs, all devices receive equal amounts of data. Thus the speed of all devices is limited by the speed of the slowest device. If you have several fast devices and one or two slow devices, it might be advantageous to remove the slow devices.

When CPU resources are plentiful during an HPL job, the device controllers are a potential bottleneck. If you have configured extra converter threads and extra converter VPs, CPU use should be close to 100 percent. If CPU use is not close to 100 percent, the cause might be one of the following situations:

- The device controller is managing too many devices.
- The devices themselves are slow.

Related reference

[Define device arrays on page 214](#)

HPL usage tasks

Three tasks that you might need to perform for the High-Performance Loader (HPL) are:

- Reorganizing computer configuration
- Altering the schema of a table
- Assessing information for loading or unloading external data

Related information

[The onpload configuration parameters on page 183](#)

Reorganize computer configuration

If you are not changing the table schema, use a No-Conversion Job to unload and load when you need to reorganize the configuration of your computer or change to a different computer. The no-conversion mode is the fastest means of performing an unload job or a load job, because rows are unloaded in HCL Informix® internal format with no conversion and reloaded in the same fashion.



Restriction: You cannot use a No-Conversion Job when the source and target computers use different internal byte representations. For information about the byte-order type, see [The Machines window on page 77](#).

For information about preparing for a no-conversion unload/load with ipload, see [Using the No Conversion Job option on page 169](#). To set no-conversion mode when you are using the onpload utility at the command line, use the **-fn** option. For more information, see [The onpload utility on page 189](#).

Alter the schema of a table

When you need to alter a table (add, drop, or change the data type of columns), use the Fixed Internal format. In Fixed Internal format, rows are unloaded in HCL Informix® internal format on a column-by-column basis. Thus you can drop, add, or modify columns and still minimize conversion overhead.

Related reference

[Format type group on page 166](#)

Assess information for loading or unloading external data

When you load or unload data from an external source, you must assess the type of data and the amount of conversion that is required so that you can choose appropriate configuration parameters.

The following sections show possible configuration parameters for two different types of load jobs. Suppose your hardware has the following configuration:

- Eight CPU symmetric multiprocessors, each about 40 MIPS
- Several (say 16) 300-MB disks (for the database)
- Four 1.2-MB tape devices (to load to and unload from)
- 512-MB memory

You also have this information about your system:

- The database server is running with seven CPU virtual processors.
- The data that you want to load has a row of about 150 bytes with a mix of **INT**, **DATE**, **DECIMAL**, **CHAR**, and **VARCHAR** data types. (This is similar to the LINEITEM table in the TPC-D database).

Settings for a no-conversion load or unload job

A no-conversion load or unload is not highly CPU intensive because no conversion is involved. In this case, the load or unload is expected to be limited by the speed of the tape devices. No-conversion loads and unloads are always completed in express mode.

The following table lists sample values of configuration parameters for a raw load.

Configuration parameter	Suggested value	Comment
CONVERTVPS	4	This process is not CPU intensive.
CONVERTTHREADS	1	This process is not CPU intensive.
STRMBUFSIZE	128	Choose some multiple of the AIOBUFSIZE, up to about 8*AIOBUFSIZE.
STRMBUFFERS	5	Should be CONVERTTHREADS + 4.
AIOBUFSIZE	32	Choose a buffer size to match the best block size for the tape drive. Large buffers increase performance if sufficient memory is available.
AIOBUFFERS	5	CONVERTTHREADS + 4 or 2 * CONVERTTHREADS, whichever is larger

Related information

[Using the No Conversion Job option on page 169](#)

Run no-conversion load jobs with tables with hidden columns

If you run no-conversion jobs, the physical image of the table in the load file must match the physical image of the target table and all columns must exist in the same order and be of the same type.

By default a no-conversion unload job does not contain the hidden columns, because the job resulted from unloading a "SELECT *" query and this type of SELECT query does not return any hidden columns. Under these circumstances, the target

table where the load will occur must not contain any hidden columns. If it does contain hidden columns, the image of the file will not be a match.

If your application requires that the unload file contains the hidden columns (for example, if the target table has hidden columns and it is not possible to alter the table to remove the columns while the load occurs) , you must use a SELECT statement that explicitly indicates the columns in the select list.

The target table should not contain hidden columns, because these columns retain internal information for internal activities of the engine. Loading hidden columns can create unexpected results.

Hidden columns tables include but are not restricted to fragmented tables with rowid columns, tables with VERCOLS, and tables with CRCOLS. Also fragmented tables with rowid columns do not support no-conversion jobs.

An express-mode load with delimited ASCII

In an express-mode load with delimited ASCII, the load or unload job is limited by the available CPU. The conversion to or from HCL Informix® types to ASCII is the most expensive portion of these operations

Because this conversion is expensive, the following configuration might be more appropriate. The following table lists sample values of configuration parameters for an express-mode load with delimited ASCII.

Configuration parameter	Sample value	Comment
CONVERTVPS	8	One convert VP per CPU
CONVERTTHREADS	2	Four devices * 2 thread/device = 8 threads
STRMBUFSIZE	32	Should be some multiple of AIOBUFSIZE. For this CPU-intensive case, 1 or 2 * AIOBUFSIZE is sufficient.
STRMBUFFERS	4	CONVERTTHREADS + 4
AIOBUFSIZE	32	Choose a buffer size to match the best block size for the tape drive. Large buffers increase performance if sufficient memory is available.
AIOBUFFERS	5	CONVERTTHREADS + 4 or 2 * CONVERTTHREADS, whichever is larger

HPL performance hints

In general, the performance of the High-Performance Loader (HPL) depends on the underlying hardware resources: CPU, memory, disks, tapes, controllers, and so on. Any of these resources could be a bottleneck, depending on the speed of the resource, the load on the resource, and the particular nature of the load or unload.

For example, load and unload jobs that perform no conversions consume minimal CPU resources. These jobs are thus likely to be limited by device or controller bandwidth. Alternatively, ASCII loads and unloads are CPU intensive because of the overhead of conversion to and from ASCII. This section discusses some topics that you should consider when you try to improve performance.

Related information

[The onpload configuration parameters on page 183](#)

Choose an efficient format

When you load data to or unload data from a source that is not HCL Informix®, you can use fixed or delimited format in an appropriate code set such as ASCII or EBCDIC.

In general, ASCII loads and unloads are the fastest. If you are using onpload for a machine or schema reorganization, choose the no-conversion format. Delimited and fixed ASCII formats are comparable in behavior except when VARCHAR data is present. If the schema contains VARCHAR data and the length of the VARCHAR data varies greatly, you might want to choose delimited format.

Ensure enough converter threads and VPs

Loads and unloads other than raw and fast-format ones are likely to be CPU intensive due to conversion overhead. In such cases, conversion speed is likely to determine the load or unload speed. It is thus important to use sufficient conversion resources (that is, enough converter threads and VPs).

The number of converter threads that is required for a device depends on the relative speeds of the device and the CPU as well as the data types in the table being loaded or unloaded. CHAR and VARCHAR formats are the cheapest to convert. INT, DATE, SMFLOAT, and FLOAT are more expensive. DECIMAL and MONEY are among the most expensive formats to convert.

The `plconfig` file specifies the number of converter threads per device. You can override this value on the onpload command line with the **-M** option.

The number of converter VPs should be based on the conversion intensity of the load or unload and the number of physical CPUs on the computer. If the load or unload is expected to be highly intensive, you might want to specify the number of convert VPs to be the number of physical CPUs (or one fewer) to take advantage of all of the available CPUs. You can set the number of converter VPs in the onpload configuration file.

The database server and onpload client VPs might both be competing for the same physical CPU resources. To reduce contention, run only the number of VPs that are necessary on both the database server and onpload sides. However, if the number of database server VPs is already specified, you might have a choice only in the number of onpload VPs. In this case, the suggestions in the previous paragraph apply.

Too few converter threads and VPs can make conversion a bottleneck. However, too many converter threads can waste time in scheduling threads in and out of the VPs. In general, more than ten converter threads per VP is too many.

Ensure enough memory

To maximize available memory and scan resources, the High-Performance Loader (HPL) automatically sets the **PDQPRIORITY** environment variable to `100`, if it is not already set. If the **PDQPRIORITY** environment variable is set, HPL uses that value.

If the **PDQPRIORITY** environment variable is set to 0 or if the **PDQPRIORITY** environment variable is disabled on the database server, then HPL cannot unload multiple devices.

Ensure enough buffers of adequate size

You set the size and number of buffers in the `plconfig` configuration file with the `onpload` utility.

For adequate performance, you should provide at least two (preferably three) AIO and stream buffers per converter thread. The size of AIO buffers should be at least as large as the device block size, and the size of the stream buffers should be large (32 or 64 KB), as the following table shows.

Configuration Parameter	Size	Comment
STRMBUFFSIZE	64	Should be some multiple of AIOBUFSIZE for best performance
STRMBUFFERS	2 * CONVERTTHREADS	
AIOBUFSIZE	64	
AIOBUFFERS	2 * CONVERTTHREADS	

Increase the commit interval

In the deluxe modes, the commit interval specifies the number of records that should be loaded before the transaction is committed (see [Figure 106: The Load Options window on page 159](#)). Low values (frequent commits) degrade performance and high values improve performance. If you increase the commit interval, you might need to increase the size of your logical-log buffers.

While larger commit intervals can speed up loads, larger commit intervals require larger logical-log space and increase the checkpoint time. These side effects impact other system users during `onpload` operations.

Limitation when using the Excalibur Text DataBlade® Module indexes

You cannot create a load job with multiple devices to insert rows into a table that has an Excalibur Text DataBlade® module index. A multiple-device load fails because the Excalibur Text DataBlade® module does not handle concurrency correctly.

To avoid this limitation:

- Create a load with a single device.
- Complete the load by using multiple devices and then create the Excalibur Text DataBlade® module index on the table.

The onpload utility

This section describes how to use the `onpload` utility.

Related information

[The onpload utility on page 8](#)

[Unload jobs on page 142](#)

Overview of the onpload utility

After you create the onpload database with the ipload interface, the onpload utility allows you to perform loads and unloads directly from the command line. However, you cannot load or unload binary large objects (BLOBs) or character large objects (CLOBs) to or from multiple files.

The onpload command uniquely specifies a row in the **session** table in the onpload database. Each row in the **session** table specifies all the components and options that are associated with a job.

The onpload file name size limitations on UNIX™

On UNIX™, you cannot have a file name size greater than 256 characters.

Suppose you have a 128-character database name and a 128-character table name and automatically generate the project (auto job generation), the generated device file names have the following format:

```
directory_specified/databasename_tablename.unl
```

In this situation, the device name would exceed 256 characters and the job would not run. To work around this problem, modify the device file names (assuming the specification file name is `device.hpl`):

1. Describe the device with the following command:

```
onpladm describe device devicename -F device.hpl
```

2. Modify the `device.hpl` specification file to shorten the names of `.unl` files.
3. Modify the device attributes with the following command:

```
onpladm modify device devicename -F device.hpl
```

Alternatively, do not use the auto generation option of `onpladm` when you have both database and table names that would produce a generated name exceeding the 256 UNIX™ file name limit. Instead, manually provide a device file name.

Related reference

[The onpladm utility on page 199](#)

Start the onpload utility

You can start onpload from the command line or from ipload.

When you click **Run** in the Load Job window ([Figure 104: The Load Job window on page 155](#)) or in the Unload Job window ([Figure 99: The Unload Job window on page 145](#)), ipload uses information from the onpload database to start onpload.

Typically, you use `ipload` to start a job when you plan to perform that particular load or unload once. For a job that needs to be run periodically, such as a weekly report, you might choose to run the job from the command line.

The information that you give to `onpload` as command-line arguments takes precedence over any information that is in the `onpload` database. The information from the command-line arguments is effective only for a single load. The command-line arguments do not affect the values in the `onpload` database or in the `plconfig` configuration file.

Using the onpload utility

In most cases, use the Load Job window (Figure 104: The Load Job window on page 155) or Unload Job window (Figure 99: The Unload Job window on page 145) to prepare the load or unload job.

After you prepare the job, the **Command Line** text box on the Load Job Select window (Figure 103: The Load Job Select window on page 154) or the Unload Job Select window (Figure 98: The Unload Job Select window on page 144) shows you the command line that `ipload` prepared for the job. You can copy that command line and use it to run the job at a later time. You can also use the command-line options shown in this section to modify the basic command line that `ipload` prepared.

i Tip: Enter `onpload` with no options at the command line to display a command-line listing of all `onpload` options and their functions.

When you use the `onpload` utility, you cannot load or unload smart large objects (BLOB or CLOB data types) to or from multiple files.

The `onpload` and `onpladm` utilities include support for long object names up to 128 characters, but the `ipload` utility does not.

The following sections give additional information about the syntax and individual options of the `onpload` utility.

The onpload utility syntax

```
onpload { -v | { -jjobname [ -asource ] [ -pprojectname ] | -mmap -asource [ -pprojectname ] <Other Options> } }
```

Other Options

```
" <Setting the run mode> <Modifying parameter size> "
```

```
" <Overriding database values> [ -z ] "
```

For more information about the options shown in the diagram, see [Set the onpload run mode with the -f option on page 193](#), [Modify the size of onpload database parameters on page 195](#), and [Override the onpload database values on page 198](#).

The following table contains an explanation of the elements in the diagram.

Element	Purpose	Key considerations
-V	Displays the current version number and the software serial number.	This option is available only from the command line.
-d source	Sets the path name of the file, tape, or pipe (UNIX™ only) or the name of the device array to use for the load or unload session	If the -f option is not set to a , d , or p , onpload assumes that the data source is a file. To use ipload, see Interpret the onpload -d and -f options together on page 194 .
-j jobname	Names a load or unload job from the onpload database	When using the -j option, you can override only a few values from the onpload database. See Override the onpload database values on page 198 To set by using ipload, see Components of the unload job on page 142 and Components of the load job on page 150 .
-m map	Names a map from the onpload database	To use ipload, see Figure 80: Using a map on page 120 .
-p projectname	Identifies the project where the format and map are stored	To use ipload, see Project organization on page 67 If you use this feature, you must use it for both the load and unload commands. Otherwise, the unloaded data might not match the loaded data.
-Z	Enables writing to or reading from a tape until the end of the device. The onpload utility prompts you for additional tapes until the load or unload is completed	If the -Z option is not set, the onpload uses the tape size provided on the command line. If the -Z option is set, it supersedes the tape size information provided. This option is equivalent to checking the Write/read to/from tape until end of device check box on the Load Job Select or Unload-Job Select windows.

The pload command line assumes the default project unless otherwise specified with the **-p** option.

For example, you might use the Load Job window to prepare the following command:

```
onpload -j bigload -p zz
```


If you receive a tape that you know contains data with bugs, you might choose to modify the command to allow errors and to save the log in a special place, as follows:

```
onpload -j bigload -p zz -fl -e 1000 -l /mylogs/buggytape.log
```

For information about the **-fl** option, see [Set the onpload run mode with the -f option on page 193](#).

Set the onpload run mode with the -f option

The **-f** option lets you set the type of source data and the type of mode. Possible modes are express load, deluxe load, deluxe load without replication, or unload. If used along with the **-j** option, you can override only a few values from the **onpload** database.

See [Override the onpload database values on page 198](#).

Setting the Run Mode

```
" -f[ {d | p | a} ] [ {l [c] | u} ] [ {n | q | v | M | N} ] "
```

Element	Purpose	Key considerations
M	Displays the program module or line number in messages.	This flag is available only from the command line. This flag is used for debugging.
N	Allows deluxe mode load without replication	This flag works only if the -j job argument, which specifies a session table job to run, is left blank.
a	Treats data source as a device-array.	The definition of the device array is extracted from the onpload database. To use ipload, see Device arrays on page 79 .
c	Sets mode to deluxe mode.	If this flag is not set, onpload uses express mode. To use ipload, see HPL modes on page 177 .
d	Treats data source as a device (tape or file).	To set this option by using ipload, see Device arrays on page 79 .
l	Loads data into database.	This is the default flag, as opposed to u , which unloads data from the database. To use ipload, see Components of the load job on page 150 .
n	Specifies that onpload does not need to perform data conversion.	The target table for the load must have the same schema as the table from which the data is extracted.

Element	Purpose	Key considerations
		If onpload generated the input data file as a data file in HCL Informix® format, you do not need to perform data conversion when you reload data. To use ipload, see Using the No Conversion Job option on page 169 .
p	Treats data source as a program to run and reads interface to the program by way of a pipe (on UNIX™ only).	To use ipload, see Device arrays on page 79 .
q	Tells onpload not to generate status messages while a job is running.	None.
u	Unloads data from database.	If this flag is not set, onpload loads data into the database. To use ipload, see Components of the unload job on page 142 .
v	Tells onpload not to generate violations records.	This flag is available only from the command line, and is not supported with deluxe mode load.

Type the onpload -f flags

When you combine **-f** flags into one group, do not put spaces between the flags. For example, use **-f acq**.

If you prefer, you can use multiple occurrences of the **-f** option instead of combining all of the possible **-f** flags into one group. For example, the following two command lines are equivalent:

```
onpload -m mymap -d mydev -flnc
onpload -m mymap -d mydev -fl -fn -fc
```

Interpret the onpload -d and -f options together

The argument of the **-d** option gives the name of the data source. You can specify the device type of the data source with flags of the **-f** option, as follows:

- If the command line does not specify a device type, onpload treats the data source as the path name of a cooked file on disk. Because no device type is specified, the following onpload command treats **filename** as the name of a file:

```
onpload -m mapname -d filename
```

- The **-fd** option in the following command causes onpload to treat `/dev/rmt/rst11b` as the name of a tape device:

```
onpload -m mapname -d /dev/rmt/rst11b -fd
```

The tape device name must be Berkeley Software Distribution (BSD) compliant.

- The **-fa** option in the following command causes onpload to treat **tapearray3** as the name of a device array. The device array is described in the onpload database.

```
onpload -m mapname -d tapearray3 -fa
```

- In an UNIX™ environment, the **-fp** option in the following command causes onpload to treat **apipename** as the name of a pipe. When onpload starts executing, it causes the pipe process to start executing.

```
onpload -m mapname -d apipename -fp
```

The same semantics apply for an unload job. If you use the **u** flag of the **-f** option to indicate an unload job, the interpretation of the data-source name is as described previously. For example, the following command specifies that onpload should unload data to the device `/dev/rmt/rst11`:

```
onpload -m mapname -d /dev/rmt/rst11 -fdu
```

Modify the size of onpload database parameters

The options that are described in this section let you enter size information that overrides existing parameters in the onpload database.

Modifying Parameter Size

```
" [ { -Atapehead | -Bblocksize | -Gswapbytes | -Icommit_int | -aiobuFSIZE | -b bufsize | -emaxerrors |
-i prog_interval | -n numrecs | -s startrec | -t numtapes } ] "
```

Element	Purpose	Key considerations
-A <i>tapehead</i>	Tells onpload to skip the specified number of bytes on the tape before it starts reading data records.	This option is available only from the command line. For specific details on this option, see The session table in the onpload database.
-B <i>blocksize</i>	Sets the tape I/O block size (bytes).	If the data source is a device array, this setting is ignored. To use ipload, see Figure 50: The tape parameters group on page 83 .
-G <i>swapbytes</i>	Sets the number of bytes in a swap group.	This option is available only from the command line. This option globally reverses the byte order in the input data stream. Each

Element	Purpose	Key considerations
		group of bytes is swapped with the group of bytes that follows it.
-l <i>commit_int</i>	Sets the number of records to process before doing a commit.	This option applies only to deluxe mode. To use ipload, see Figure 106: The Load Options window on page 159 .
-a <i>iobufsize</i>	Sets the size (kilobytes) of the asynchronous I/O buffers, the memory buffers used to transfer data to and from tapes and files.	This option is available only from the command line This value overrides the value (AIOBUFSIZE) set in the HPL configuration file (<code>plconfig</code>). For specific details on this option, see The AIOBUFSIZE configuration parameter.
-b <i>bufsize</i>	Sets the size (kilobytes) of the server stream buffer, the memory buffer used to write records to the database.	This option is available only from the command line. Larger buffers result in more efficient data exchange with the database. This value overrides the value (STRMBUFFSIZE) set in the HPL configuration file (<code>plconfig</code>). For specific details on this option, see The STRMBUFFSIZE configuration parameter.
-e <i>maxerrors</i>	Sets the error threshold that causes the load or unload session to shut down.	If no number is specified, the default is to process all records. To use ipload, see Figure 106: The Load Options window on page 159 .
-i <i>prog_interval</i>	Sets the number of records to process before making an entry in the log file specified by the -l option.	This option is available only from the command line. If no log file is specified, progress messages are sent to stdout . If the -i option is omitted, the default number is

Element	Purpose	Key considerations
		1000. To set, see The onpload -i option on page 197 .
-n numrecs	Sets the number of records to load.	If no number is specified, all records are processed. The -n option does not affect unload operations because pload cannot maintain row ordering. To use ipload, see Figure 106: The Load Options window on page 159 .
-s startrec	Sets the starting record to load.	This option is used to skip records. For example, setting -s to 10 starts the loading at the 10th record, so that if the file contains 20 records, 11 records are loaded. Setting -s to 0 or 1 the load starts with the first record. If you do not set this option, the load starts with the first record. The -s option does not affect unload operations because pload cannot maintain row ordering. To use ipload, see Figure 106: The Load Options window on page 159 .
-t numtapes	Specifies the number of tapes to load.	If you do not set this option, the default value is 1. To use ipload, see Figure 106: The Load Options window on page 159 .

The onpload -i option

You can specify the number of records to process before onpload reports the progress in an entry in the log file with the **-i** option.

The onpload utility calculates the progress message count in the pload log file as equal to the number of rows processed, but rounded down to the nearest multiple of the value of *progress_interval*. For example, if the number of rows processed is 910 and the value of *progress_interval* is 100, then the progress message count is 900.

The onpload utility updates the row count only once for each stream buffer of data that it processes. Thus, reducing the row count on the **-i** option does not necessarily increase the number of progress messages in the log file. For example, if the stream buffer holds 910 rows of data, setting *row_count* to 10, 100, and 900 has the same effect: onpload writes one progress message.

Override the onpload database values

The options that are described in this topic let you enter size information that overrides existing parameters in the onpload database. You can override only these parameters. You cannot override other options, such as run mode.

Overriding the onpload database values

```
" [ { -ccaseconvert | -doverride_db | -ffilter | -ltrace_level | -mconverters | -rrejectfile | -sservername |
-rtarget_db | -llogfile } ] "
```

Element	Purpose
-C <i>caseconvert</i>	Sets the case-conversion option that converts all character information
-D <i>override_db</i>	Overrides the database specified in the map used for the load
-F <i>filter</i>	Identifies the filter that onpload uses for screening load records
-L <i>trace_level</i>	Sets the amount of information logged during the load
-M <i>converters</i>	Sets the maximum number of conversion threads per device
-R <i>rejectfile</i>	Identifies the file destination for rejected records
-S <i>servername</i>	Sets the onpload database server
-T <i>target_db</i>	Sets the target database serve
-l <i>logfile</i>	Specifies the name of a file to which onpload sends messages

Load data into collection data type columns

The onpload utility might need to create temporary smart large objects while loading data into collection data type columns. To create temporary smart large objects, onpload obtains sbspace information from the SBSPACETEMP configuration parameter, or if that parameter is not set, from the SBSPACENAME configuration parameter. For more information about temporary sbspaces, see the *HCL® Informix® Administrator's Guide*.

If onpload does not find a temporary sbspace, the load job fails with the following error message in the onpload log file:

```
Fatal error in server row processing - SQL error -9810 ISAM error -12053
```

If you see the error message shown previously in onpload log file, configure a temporary sbspace by using the SBSPACETEMP configuration parameter or a default sbspace by using the SBSPACENAME configuration parameter and then restart the load job.

The onpladm utility

This section describes how to use the onpladm utility.

Related reference

[Examples of loading and unloading jobs using the ipload utility on page 18](#)

[Start the ipload utility on page 20](#)

[The ipload utility GUI or the onpladm command-line interface on page 45](#)

Related information

[The onpladm utility on page 9](#)

[The onpload file name size limitations on UNIX on page 190](#)

Overview of the onpladm utility

The onpladm command-line interface is equivalent to the ipload utility. You can use the onpladm utility from the command line to create, modify, and delete High-Performance Loader (HPL) objects. The HPL objects include projects, jobs, maps, formats, queries, filters, device arrays, and machines.

You can use the onpladm utility on both UNIX™ and Windows™ computers.

While the onpload and onpladm utilities include support for object names that contain up to 128 characters, the ipload utility does not. If you use long database, table or column names and create jobs by using onpladm, you cannot run these jobs with the ipload utility. For ipload, database, table and column names cannot exceed 18 characters.

Related reference

[Cannot create shared-memory pool: errno UNIX_error_num](#)

The onpladm utility features

You can create and maintain the onpload database with the onpladm utility and the ipload utility. The onpload database stores information about the High-Performance Loader (HPL) objects of load and unload jobs.

The first time that you issue a **create** command, the onpladm utility creates the onpload database, if it does not yet exist.

The onpladm utility can perform the functions that the following table describes.

Object	Operation
Project	Create, delete, describe, list, run
Job	Create, delete, describe, list, modify, run
Map	Create, delete, describe, list, modify

Object	Operation
Format	Create, delete, describe, list, modify
Query	Create, delete, describe, list, modify
Device array	Create, delete, describe, list, modify
Computer	Create, delete, describe, list, modify
Target server	Configure and list default settings
Filter	Create, delete, describe, list, modify

The onpladm utility also allows you to:

- Create jobs and other objects with a single, minimum-input command.
- Specify object characteristics in specification files.
- Create, load, or unload jobs for all tables in a database.
- Create jobs for an entire database and group the jobs into a project with a single command.
- Run individual jobs or projects independently.



Important: The most effective way to use onpladm is to create and modify HPL objects is to:

1. Create the default objects by using default job creation options.
2. Use `describe object` syntax to describe the objects.
3. Manually modify these objects with correct values for different configuration parameters.
4. Use `modify object` syntax to correctly create these objects.

Specification-file conventions

Use specification files to create, modify, and describe the HPL objects. When you enter the onpladm command to describe an object, the contents of the specification file appear.

When you create a job or map with a quick command, the onpladm utility uses default attributes to create the job or map. If you create a job or map with a specification file, you can specify attribute values.

The following diagram illustrates the syntax to create or modify an object by using a specification file.

Creating or modifying an object with a specification file

```
onpladm { create | modify } object [ -F specfilename ] [ -s servername ]
```

Element	Purpose	Key considerations
<code>-F specfilename</code>	Sets the specification file	The default value is the standard output.

Element	Purpose	Key considerations
-S <i>servername</i>	Sets the onpload database server	The default is the value of the INFORMIXSERVER environment variable.

Use the following conventions when you create specification files:

- Begin object definitions with `BEGIN OBJECT` and end them with `END OBJECT`.
- If object definitions contain variable items, begin each variable item with `BEGIN SEQUENCE` and end each item with `END SEQUENCE`.

For example, you might use the following specification file to create a device array that consists of a file and a pipe:

```
BEGIN OBJECT DEVICEARRAY mydevice
# Optional Attributes
BEGIN SEQUENCE
TYPE FILE
FILE /work/data.unl
TAPEBLOCKSIZE 0
TAPEDEVICESTRIDE 0
PIPECOMMAND
END SEQUENCE
BEGIN SEQUENCE
TYPE PIPE
FILE
TAPEBLOCKSIZE 0
TAPEDEVICESTRIDE 0
PIPECOMMAND /work/bin/datacreate.sh
END SEQUENCE
END OBJECT
```

For more information about attributes and their possible values, see the description of each specification file.

- Precede comments in specification files with a pound sign (`#`).
- List attributes in the exact order in which the specification-file format displays them.
- Use the following syntax to refer to the attributes of an object or the attributes of elements of an object:

Attribute_name Attribute_value



Important: Do not use this for `BEGIN` and `END` statements and comment statements.

You must always provide an attribute name. You must provide both the attribute name and the attribute value to describe a required attribute, but you only have to provide the attribute name if the attribute is optional.

Attributes and their values depend on their object type. For more information about attributes and object types, see the corresponding specification files.

- Enclose attribute values that contain spaces in double quotation marks.
- Precede double quotation marks in attribute values with a double quotation mark.

For example, to enter a `MATCH` condition for `" CA "` in a filter object, include the following line: `MATCH = " "CA" "`

For more information about file conventions, see individual specification-file formats.

Related reference

[Create maps on page 217](#)

Error handling

The onpladm commands have two return values:

0

If the command is successful, a success message appears.

-1

If the command fails, an error message appears.

Define onpladm utility jobs

You can create, modify, describe, list, run, and delete jobs with the onpladm utility.

Related reference

[Components of the load job on page 150](#)

Create onpladm jobs

You can create two types of jobs:

- Conversion jobs

Use a conversion job to process data that requires conversion before loading and that is not in internal format. Conversion jobs have associated maps and format objects.

- No-conversion jobs (fast jobs)

Use a no-conversion job to process data that does not require conversion before loading and that uses the HCL Informix® internal format. No maps or formats are associated with a no-conversion job. Consequently, a no-conversion job is faster than a conversion job. No-conversion jobs are always completed in express mode.

Create conversion jobs

When you use command-line parameters to create a conversion load or unload job, all maps, formats, and query objects automatically have the same name as the job name. If you delete a job, the **map.format** or the query objects will remain intact unless you specifically delete the format or query objects.

When you create a conversion job with detailed specification files, the High-Performance Loader (HPL) objects can have different names from the job name.

Create conversion jobs by using a quick command

When you create a conversion job by using a quick command, the onpladm utility creates all High-Performance Loader (HPL) objects associated with the job. The HPL objects that it creates have the same name as the job.

The following diagram illustrates the syntax to create a conversion job from the command line.

Creating a conversion job

```
onpladm create job job [ -p project ] -d device -D database -t table [ <Setting the run mode> ] [ { -n <Setting the format> } ] [ -r target ] [ -s server ] [ -M devicesize ] [ -B blocksize ]
```

Element	Purpose	Key considerations
-B blocksize	Sets the tape I/O block size (bytes)	No default value
-d device	Sets the name of device, such as a file, device array, tape, or pipe	No default value
-D database	Name of the target database that contains the information to be loaded or unloaded	No default value.
job	Names a load or unload job from the onpload database	None
-M devicesize	Tape device size in kilobytes	The device size must be greater than zero.
-n	Sets no-conversion express job	None
-p project	Identifies the project where the format and map are stored	The default is the project created when the onpload database is built.
-S server	Sets the onpload database server	The default is the value of the INFORMIXSERVER environment variable.
-t table	Name of the table to be loaded or unloaded	None
-T target	Name of the target server to which the data will download	The default is the value of the INFORMIXSERVER environment variable.

The following diagram illustrates the syntax to set the run mode with the **-f** option.

Setting the Run Mode

`"-f"`

`"[{d | p | a}]"`

`"[{l [c] | u}]"`

`"[n]"`

`"[N]"`

Element	Purpose	Key considerations
a	Treats data source as a device-array	The definition of the device array is extracted from the onpload database.
c	Sets mode to deluxe mode	If this flag is not set, onpladm uses express mode.
d	Treats data source as a tape device	None
l	Loads data into database	This is the default flag, as opposed to u , which unloads data from the database.
n	Specifies that onpladm does not need to perform data conversion	<p>The target table for the load must have the same schema as the table from which the data is extracted.</p> <p>If onpladm generated the input data file as a data file in HCL Informix® format, you do not need to perform data conversion when you reload data.</p>
N	Allows deluxe mode load without replication	This flag works only if the <i>job</i> argument, which specifies a session table job to run, is left blank.
p	Treats data source as a program to run and reads interface to the program by way of a pipe (on UNIX™ only)	None
u	Unloads data from database	If this flag is not set, onpladm loads data into the database.

The following diagram illustrates the syntax to set the format type with the `-z` option.



Element	Purpose	Key considerations
D	Sets the format to delimited	This is the default value. See Delimited records on page 97 .
FI	Sets the format to fixed internal	See Fixed-length records on page 86 .
FA	Sets the format to fixed ASCII	See Fixed-length records on page 86 .
FB	Sets the format to fixed binary format.	See Fixed-length records on page 86 .
C	Sets the format to COBOL	See COBOL records on page 100 .
CB	Sets the format to COBOL (byte)	See COBOL records on page 100 .

Create conversion jobs by using detailed specification files

You can also specify job details in specification files that you reference from the command line. When you create a job by using a specification file, you must create all associated High-Performance Loader (HPL) objects; the `onpladm` utility does not create these objects for you.

Related reference

[Describe a job on page 211](#)

Create a conversion-load job

Use the syntax shown in [Specification-file conventions on page 200](#) to create a conversion-load job from a specification file.

Use the following syntax to create a specification file for a conversion-load job:

```

BEGIN OBJECT LOADJOB jobname
# Compulsory Attributes
PROJECT projectname
DEVICE device_array_name
MAP mapname

FILTER filename
SERVER targetservername
DATABASE targetdatabasename
```

```

FLTFILE filtered_records_filename
REJECTFILE rejected_records_filename
LOGFILE job_progress_logfilename
RUNMODE runmode_type
GENERATEVIORECS violation_records_option
TAPES sourcetape_num
NUMRECORDS records_num
STARTRECORD start_record
MAXERRORS max_error_num

END OBJECT
    
```

The following table lists the attributes and their attribute values.

Attribute	Attribute value
<i>device_array_name</i>	Device-array name You must create the device array; onpladm does not create it for you.
<i>filtername</i>	Filter name
<i>jobname</i>	Job name
<i>job_progress_logfilename</i>	Path to the job-progress log file
<i>mapname</i>	Map name You must create the map before you use this option; onpladm does not create the map for you.
<i>max_error_num</i>	Maximum number of errors; if exceeded, load ends
<i>projectname</i>	Name of an existing project
<i>records_num</i>	Number of records to be processed in the data file
<i>rejected_records_filename</i>	Rejected-records file name (rejected by database server)
<i>filtered_records_filename</i>	Filtered-records file name (rejected by filter)
<i>runmode_type</i>	Type of run mode: E Express® mode D Deluxe mode without replication DR Deluxe mode with replication
<i>sourcetape_num</i>	Number of tapes that contain source data

Attribute	Attribute value
<i>start_record</i>	Number of the first record to begin a load
<i>targetdatabasename</i>	Name of the database that the records will be loaded or unloaded to; if set, this value overrides the database value in the load or unload map
<i>targetservername</i>	Name of the target database server
<i>violation_records_option</i>	Specify: Y to generate violations records or N not to generate violations records

Create a conversion unload job

Use the syntax shown in [Specification-file conventions on page 200](#) to create a conversion unload job from a specification file.

Use the following syntax to create a conversion unload job:

```
BEGIN OBJECT UNLOADJOB jobname
# Compulsory Attributes
PROJECT projectname
DEVICE device_array_name
MAP mapname

FILTER filtername
SERVER targetservername
DATABASE targetdatabasename
REJECTFILE rejected_records_filename
LOGFILE job_progress_logfilename
ISOLATIONLEVEL isolation_level
MAXERRORS max_error_num

END OBJECT
```

The following table lists the arguments and their attribute values.

Attribute	Attribute value
<i>device_array_name</i>	Device-array name You must create the device array; onpladm does not create it for you.
<i>filtername</i>	Filter name
<i>isolation_level</i>	Unload isolation level: DR Dirty Read

Attribute	Attribute value
	<p>CR Committed Read</p> <p>CS Cursor Stability</p> <p>RR Repeatable Read</p>
<i>jobname</i>	Job name
<i>job_progress_logfilename</i>	Path to the job-progress log file
<i>mapname</i>	<p>Map name</p> <p>You must create the map before you use this option; onpladm does not create the map for you.</p>
<i>max_error_num</i>	Maximum number of errors; if exceeded, unload ends
<i>projectname</i>	Name of an existing project
<i>rejected_records_filename</i>	Rejected-records file name (rejected by database server)
<i>targetdatabasename</i>	Name of the database that the records will be loaded or unloaded to; if set, this value overrides the database value in the load or unload map
<i>targetservername</i>	Name of the target database server

Create no-conversion jobs

A no-conversion job is faster than a conversion job because onpload uses the internal format of the target database and does not create maps or formats. No-conversion jobs are always run in express mode.

Create no-conversion jobs by using a quick command

When you create a job by using a quick command, the onpladm utility creates all High-Performance Loader (HPL) objects associated with that job.

The following diagram illustrates the syntax to create a no-conversion job.

Creating a no-conversion job

```
onpladm create job job [ -pproject ] -n -advice -ddatabase -ttable [ [ -sserver ] [ -rtarget ] <Setting the run mode> ]
```


Element	Purpose	Key considerations
-d <i>device</i>	Sets the name of device, such as a file, device array, tape, or pipe	No default value.
-D <i>database</i>	Name of the target database that contains the information to be loaded or unloaded	No default value.
<i>job</i>	Names a load or unload job from the onpload database	None
-n	Sets no-conversion express job	None
-p <i>project</i>	Identifies the project where the format and map are stored	The default is the project created when the onpload database is built.
-S <i>server</i>	Sets the onpload database server	The default is the value of the INFORMIXSERVER environment variable.
-t <i>table</i>	Name of the table to be loaded or unloaded	None
-T <i>target</i>	Name of the target server to which the data will download	The default is the value of the INFORMIXSERVER environment variable.

The following diagram illustrates the syntax to set the run mode with the **-f** option.

Setting the run mode

```
" -f "
```

```
" [{d | p | a}] "
```

```
" [{l | u}] "
```

Element	Purpose	Key considerations
a	Treats the data source as a device-array	The definition of the device array is extracted from the onpload database.
d	Treats the data source as a tape device	None
l	Loads data into the database	This is the default flag, as opposed to u , which unloads data from the database.
p	Treats the data source as a program to run, and reads the interface to the program by way of a pipe (UNIX™ only)	None

Element	Purpose	Key considerations
u	Unloads data from the database	If this flag is not set, onpladm loads data into the database.

Create no-conversion jobs by using detailed specification files

You can also specify job details in specification files that you reference from the command line. When you create a job by using a specification file, you must create all associated High-Performance Loader (HPL) objects; the onpladm utility does not create these objects for you.

Create a no-conversion load job

Use the syntax shown in [Specification-file conventions on page 200](#) to create a no-conversion load job with specification files.

Use the following syntax to create a specification file for a no-conversion load job:

```
BEGIN OBJECT FASTLOADJOB jobname
# Compulsory Attributes
PROJECT projectname
DEVICE device_array_name
DATABASE targetdatabasename
TABLE tablename

SERVER targetservername

END OBJECT
```

The following table lists the attributes and their values.

Attribute	Attribute value
<i>device_array_name</i>	Device-array name You must create the device array; onpladm does not create it for you.
<i>jobname</i>	Job name
<i>projectname</i>	Name of an existing project
<i>tablename</i>	Table name
<i>targetdatabasename</i>	Name of the database that the records will be loaded or unloaded to; if set, this value overrides the database value in the load map
<i>targetservername</i>	Name of the target database server

Create a no-conversion unload job

Use the syntax shown in [Specification-file conventions on page 200](#) to create a no-conversion unload job with specification files.

Use the following syntax to create a specification file for a no-conversion unload job:

```
BEGIN OBJECT FASTUNLOADJOB jobname
# Compulsory Attributes
PROJECT projectname
DEVICE device_array_name
DATABASE targetdatabasename
QUERY queryname

SERVER targetservername

END OBJECT
```

The following table lists the attributes and their values.

Attribute	Attribute value
<i>device_array_name</i>	Device-array name You must create the device array; onpladm does not create it for you.
<i>jobname</i>	Job name
<i>projectname</i>	Name of an existing project
<i>queryname</i>	The SQL SELECT statement, in quotation marks, that contains unload criteria
<i>targetdatabasename</i>	Name of the database that the records will be loaded or unloaded to; if set, this value overrides the database value in the load map
<i>targetservername</i>	Name of the target database server

Modify a job by using a detailed specification file

Use the syntax shown in [Specification-file conventions on page 200](#) to modify a job by using a specification file.

For information about the job-specification file, see [Create no-conversion jobs by using detailed specification files on page 210](#).

Describe a job

The following diagram illustrates the syntax to describe a job.

Describing a job

```
onpladm describe jobjobname -f [{ l | u }][{-fspecfilename | -pprojectname | -R | -sservername }]
```

Element	Purpose	Key considerations
-F <i>specfilename</i>	Sets the specification file	The default value is the standard output.
-f	Flags to specify the type of job	The default is load job.
l	Specifies a load job	None
u	Specifies an unload job	None
<i>jobname</i>	Names a job from the onpload database	None
-p <i>projectname</i>	Identifies the project where the format and map are stored	None
-R	Deletes all associated objects if they are not referenced by other objects	None
-S <i>servername</i>	Sets the onpload database server	The default is the value of the INFORMIXSERVER environment variable.

Related reference

[Create conversion jobs by using detailed specification files on page 205](#)

List all jobs in a project

The following diagram illustrates the syntax to display all the jobs in a project.

Displaying all jobs in a project

```
onploadm list job [-p projectname] [-f [{l | u}]] [-s servername]
```

Element	Purpose	Key considerations
-f	Flags to specify the type of job	The default is load job.
l	Specifies a load job	None
u	Specifies an unload job	None
-p <i>projectname</i>	Identifies the project where the format and map are stored	None
-S <i>servername</i>	Sets the onpload database server	The default is the value of the INFORMIXSERVER environment variable.

Run a job

The following diagram illustrates the syntax to run a job and display the job progress or send it to a log file.

Running a job

```
onpladm run job jobname [-p projectname] -f [{l | u}][{-l logfile | -s servername | -z}]
```

Element	Purpose	Key considerations
-f	Flags to specify the type of job	The default is load job.
l	Specifies a load job	None
u	Specifies an unload job	None
<i>jobname</i>	Names a load or unload job from the onpload database	None
-l logfile	Sets the path for a log file where job progress is recorded	None
-p projectname	Identifies the project where the format and map are stored	None
-S servername	Sets the onpload database server	The default is the value of the INFORMIXSERVER environment variable.
-Z	Enables writing to or reading from a tape until the end of the device. The onpladm utility prompts you for additional tapes until the load or unload is completed.	<p>If you use this feature, you must use it for both load and unload jobs. Otherwise, the unloaded data might not match the loaded data.</p> <p>If the -Z option is not set, the onpladm uses the tape size specified with the -M option of the create job command. If the -Z option is set, it supersedes the tape size information provided.</p>

Related information

Running onpladm on UNIX with the database server running on Windows

The onpladm utility when referential constraints are on tables

The onpladm utility unloads and loads an entire database by creating individual table unload and load jobs. If the load mode is **express**, the default load mode, the onpladm utility disables referential constraints during load jobs. If database tables have constraints on them, problems can occur. For example, if one table has a referential constraint to another table and that table is not yet loaded, a violation that prevents a table from loading can occur.

If tables have constraints, you can manually disable the constraints after the load job is complete.

Delete a job

The following diagram illustrates the syntax to delete a job.

Deleting a Job

```
onpladm delete job jobname -f [{l | u}][{-p projectname | -s servername | -R}]
```

Element	Purpose	Key considerations
-f	Flags to specify the type of job	The default is load job.
l	Specifies a load job	None
u	Specifies an unload job	None
<i>jobname</i>	Names a load or unload job from the onpload database	None
-p <i>projectname</i>	Identifies the project where the format and map are stored	None
-S <i>servername</i>	Sets the onpload database server	The default is the value of the INFORMIXSERVER environment variable.
-R	Deletes all associated objects if they are not referenced by other objects	None

Define device arrays

You can create, modify, describe, list, and delete device arrays with the onpladm utility.

Related information

[Device arrays on page 79](#)

[Devices for the device array on page 184](#)

Create a device array

You can only create a device array by using a detailed specification file.

Use the syntax shown in [Specification-file conventions on page 200](#) to create a device array.

Use the following syntax to create a device array:

```
BEGIN OBJECT DEVICEARRAY device_array_name
# Compulsory Attributes
BEGIN SEQUENCE
TYPE device_type
FILE device_path
TAPEBLOCKSIZE tapeblock_size
```

```
TAPEDEVICESTR tapedevice_size
PIPECOMMAND pipe_commandname
END SEQUENCE

END OBJECT
```

The following table lists the attributes and their values.

Attribute	Attribute value
<i>device_array_name</i>	Device-array name You must create the device array; onpladm does not create it for you.
<i>device_path</i>	Path to the device; only valid if device type is file or tape
<i>device_type</i>	Type of device, in uppercase letters (for instance, PIPE, FILE, or TAPE)
<i>tapeblock_size</i>	Tape-block size
<i>tapedevice_size</i>	Tape-device size in megabytes
<i>pipe_commandname</i>	Pipe command name

Related reference

[Modify a device array on page 215](#)

[Describe a device array on page 215](#)

Modify a device array

You can only modify a device array with a specification file.

Use the syntax shown in [Specification-file conventions on page 200](#) to modify a device array.

Related reference

[Create a device array on page 214](#)

Describe a device array

The following diagram illustrates the syntax to describe a device array.

Describing a device array

```
onpladm describe device devicename devicename [-F specfilename] [-s servername]
```

Element	Purpose	Key considerations
<i>devicename</i>	Sets the name of device, such as a file, device array, tape, or pipe	None
-F <i>specfilename</i>	Sets the specification file	The default value is the standard output.
-S <i>servername</i>	Sets the onpload database server	The default is the value of the INFORMIXSERVER environment variable.

Related reference

[Create a device array on page 214](#)

List project device arrays

To list all the device arrays in a project, use the following syntax.

Listing project device arrays

```
onpladm list device [-p projectname] [-s servername]
```

Element	Purpose	Key considerations
-p <i>projectname</i>	Identifies the project where the format and map are stored	None
-S <i>servername</i>	Sets the onpload database server	The default is the value of the INFORMIXSERVER environment variable.

Deleting a device array

The following diagram illustrates the syntax to delete a device array.

Deleting project device arrays

```
onpladm delete device devicename [-s servername]
```

Element	Purpose	Key considerations
<i>devicename</i>	Sets the name of device, such as a file, device array, tape, or pipe	None
-S <i>servername</i>	Sets the onpload database server	The default is the value of the INFORMIXSERVER environment variable.

Define maps

You can create, modify, describe, list, and delete maps with the `onpladm` utility.

Related information

[Load and unload maps on page 119](#)

Create maps

You can create maps by using a quick command or a specification file.

Related reference

[Specification-file conventions on page 200](#)

Create a map by using a quick command

When you create a map by using a quick command, the `onpladm` utility creates a format object with the same name as the map, plus the suffix `-fmt`. The generated format name (as for all `onpladm` objects) has a maximum length of 18 characters.

For example, if the map name is `mymap`, the format name is `mymap-fmt`. If the map name is `123456789123456789`, the format name is `12345678912345-fmt`.

The create map command also creates a query object for the unload map. The following diagram illustrates the syntax to create a map from the command line.

Creating a map

```
onpladm create map mapname [-p project] -D database -t table [{-s servername | -T targetservername <Setting the
format> | -f [{l | u}]]
```

Element	Purpose	Key considerations
<code>-D <i>database</i></code>	Name of the target database that contains the information to be loaded or unloaded	No default value
<code>-f</code>	Flags to specify the type of job	The default is load job.
<code>l</code>	Specifies a load job	None
<code>u</code>	Specifies an unload job	None
<code><i>mapname</i></code>	Sets the map	None
<code>-p <i>project</i></code>	Identifies the project where the format and map are stored	The default is the project created when the onpload database is built.

Element	Purpose	Key considerations
-S server	Sets the onpload database server	The default is the value of the INFORMIXSERVER environment variable.
-t table	Name of the table to be loaded or unloaded	None
-T target	Name of the target server to which the data will download	The default is the value of the INFORMIXSERVER environment variable.

The following diagram illustrates the syntax to set the format type with the **-z** option.

Setting the format

```

-z

[{D|FI|FA|FB|C|CB}]
    
```

Element	Purpose	Key considerations
D	Sets the format to delimited	This is the default value. See Delimited records on page 97 .
FI	Sets the format to fixed internal	See Fixed-length records on page 86 .
FA	Sets the format to fixed ASCII	See Fixed-length records on page 86 .
FB	Sets the format to fixed binary	See Fixed-length records on page 86 .
C	Sets the format to COBOL	See COBOL records on page 100 .
CB	Sets the format to COBOL (byte)	See COBOL records on page 100 .

Create a map with a detailed specification file

When you create a map, the onpladm utility creates a format object with the same name as the map. When you create a map with a specification file, you must create all associated High-Performance Loader (HPL) objects; the onpladm utility does not create these objects for you.

Use the syntax shown in [Specification-file conventions on page 200](#) to create maps with specification files.

Use the following syntax to creating a load map with a specification file:

```

BEGIN OBJECT LOADMAP mapname

# Compulsory Attributes
PROJECT projectname
FORMAT formatname
DATABASE targetdatabasename
TABLE targettablename
    
```

```

BEGIN SEQUENCE
COLUMNNAME columnname
FIELDNAME fieldname
JUSTIFICATION justification
CASECONVERT caseconversion
DEFAULTVALUE defaultvalue
TRANSFERBYTES byte_transfer
COLUMNOFFSET column_offset
FIELDOFFSET field_offset
FIELDMINIMUM field_minimum
FIELDMAXIMUM field_maximum
FILLCHARACTER fillcharacter
PICTURE picture
FUNCTION record_function
STORAGECODING storage_format
BLOBCOLUMN blob_columnname
END SEQUENCE

END OBJECT

```

The following table lists the attributes and their values.

Attribute	Attribute value
<i>blob_columnname</i>	The column that contains the name of the file where BYTE or TEXT data is stored See Simple LO data in a separate file on page 96 .
<i>byte_transfer</i>	Number of bytes to transfer from record field to database column
<i>caseconversion</i>	Enter UPPER for all uppercase data, LOWER for all lowercase data, and PROPER for data with an initial capital letter
<i>columnname</i>	Name of the column to be mapped
<i>column_offset</i>	Amount of offset from the beginning of the column to the location on the column from which data transfer begins
<i>defaultvalue</i>	Value when no field is mapped to the column
<i>fieldname</i>	Field corresponding to the record format to be mapped
<i>field_maximum</i>	Largest acceptable numeric-column value
<i>field_minimum</i>	Smallest acceptable numeric-column value
<i>field_offset</i>	Amount of offset from the start of the field record to the location in the record from which data transfer begins
<i>fillcharacter</i>	Character used to pad contents of a field
<i>formatname</i>	Associated format name

Attribute	Attribute value
	You must create the format; onpladm does not create it for you.
<i>justification</i>	Enter left, right, or CENTER to position text within a record
<i>mapname</i>	Map name
<i>picture</i>	Reformats and masks data from the field of a record before data is transferred to database
<i>projectname</i>	Name of existing project
<i>record_function</i>	User-defined function in a shared library that is called for every record that is processed See Custom-conversion functions.
<i>storage_format</i>	The format in which to store BYTE or TEXT data
<i>targetdatabasename</i>	Name of the database that the records will be loaded and unloaded to
<i>targettablename</i>	Target-table name

Use the following syntax to create an unload map with a specification file:

```
BEGIN OBJECT UNLOADMAP mapname

# Compulsory Attributes
PROJECT projectname
FORMAT formatname
DATABASE targetdatabasename
QUERY queryname

BEGIN SEQUENCE
COLUMNNAME columnname
FIELDNAME fieldname
JUSTIFICATION justification
CASECONVERT caseconversion
DEFAULTVALUE defaultvalue
TRANSFERBYTES byte_transfer
COLUMNOFFSET column_offset
FIELDOFFSET field_offset
FIELDMINIMUM field_minimum
FIELDMAXIMUM field_maximum
FILLCHARACTER fillcharacter
PICTURE picture
FUNCTION record_function
STORAGECODING storage_format
BLOBCOLUMN blob_columnname
END SEQUENCE
```

END OBJECT

The following table lists the attributes and their values.

Attribute	Attribute value
<i>blob_columnname</i>	The column that contains the name of the file where BYTE or TEXT data is stored See Simple LO data in a separate file on page 96 .
<i>byte_transfer</i>	Number of bytes to transfer from record field to database column
<i>caseconversion</i>	Enter UPPER for all uppercase data, LOWER for all lowercase data, and PROPER for data with an initial capital letter
<i>columnname</i>	Name of the column to be mapped
<i>column_offset</i>	Amount of offset from the beginning of the column to the location on the column from which data transfer begins
<i>defaultvalue</i>	Value when no field is mapped to the column
<i>fieldname</i>	Field corresponding to the record format to be mapped
<i>field_maximum</i>	Largest acceptable numeric-column value
<i>field_minimum</i>	Smallest acceptable numeric-column value
<i>field_offset</i>	Amount of offset from the start of the field record to the location in the record from which data transfer begins
<i>fillcharacter</i>	Character used to pad contents of a field
<i>formatname</i>	Associated format name
<i>justification</i>	Enter left, right, or CENTER to position text within a record
<i>mapname</i>	Map name
<i>picture</i>	Reformats and masks data from the field of a record before data is transferred to database
<i>projectname</i>	Name of existing project
<i>queryname</i>	Query name
<i>record_function</i>	User-defined function in dynamically linked library that is called for every record that is processed See Custom-conversion functions.
<i>storage_format</i>	The format in which to store BYTE or TEXT data
<i>targetdatabasename</i>	Name of the database that the records will be loaded and unloaded to

Related reference

[Describe a map on page 222](#)

[Modify a map by using a detailed specification file on page 223](#)

Delete a map

The following diagram illustrates the syntax to delete a map.

Deleting a map

```
onpladm delete map mapname -f [{ l | u }][{ | -p projectname | -s servername }]
```

Element	Purpose	Key considerations
-f	Flags to specify the type of job	The default is load job.
l	Specifies a load job	None
u	Specifies an unload job	None
mapname	Sets the map name	None
-p projectname	Identifies the project where the format and map are stored	None
-S servername	Sets the onpload database server	The default is the value of the INFORMIXSERVER environment variable.

Describe a map

The following diagram illustrates the syntax to describe a map.

Describing a map

```
onpladm describe map mapname -f [{ l | u }][{ | -p projectname | -F specfilename | -s servername }]
```

Element	Purpose	Key considerations
-f	Flags to specify the type of job	The default is load job.
l	Specifies a load job	None
u	Specifies an unload job	None
-F specfilename	Sets the specification file	The default value is the standard output.
mapname	Sets the map name	None

Element	Purpose	Key considerations
-p <i>projectname</i>	Identifies the project where the format and map are stored	None
-S <i>servername</i>	Sets the onpload database server	The default is the value of the INFORMIXSERVER environment variable.

Related reference

[Create a map with a detailed specification file on page 218](#)

Modify a map by using a detailed specification file

Use the syntax shown in [Specification-file conventions on page 200](#) to modify an existing map with a specification file.

Related reference

[Create a map with a detailed specification file on page 218](#)

List all maps in a project

The following diagram illustrates the syntax to list all the maps in a project.

Listing all maps in a project

```
onpladm list map [-f {l | u}] [{-p projectname | -s servername}]
```

Element	Purpose	Key considerations
-f	Flags to specify the type of job	The default is load job.
l	Specifies a load job	None
u	Specifies an unload job	None
-p <i>projectname</i>	Identifies the project where the format and map are stored	None
-S <i>servername</i>	Sets the onpload database server	The default is the value of the INFORMIXSERVER environment variable.

Define formats

You can create, modify, describe, list, and delete formats with the onpladm utility.

Related information

[Formats of supported datafile records on page 85](#)

Create a format

You can only create formats by using a detailed specification file.

Use the syntax shown in [Specification-file conventions on page 200](#) to create a format by using a detailed specification file.

Use the following syntax to create a fixed-format object from a specification file:

```
BEGIN OBJECT FIXEDFORMAT formatname

# Compulsory Attributes
PROJECT projectname
CHARACTERSET data_codeset
MACHINE machine_type
BEGIN SEQUENCE
FIELDNAME fieldname
DATATYPE datatype
BYTES field_bytes
DECIMALS decimal_places
OFFSET offset_bytes
END SEQUENCE

END OBJECT
```

The following table lists the attributes and their values.

Attribute	Attribute value
<i>data_codeset</i>	Code set used to translate data in the data table For more information about data code sets, see the following file on your product CD: \$INFORMIXDIR/gls/cm3/registry
<i>datatype</i>	Type of field data See Data types allowed in a fixed format on page 89 .
<i>decimal_places</i>	Number of decimal places for Float and Double data types
<i>field_bytes</i>	Number of bytes that the field occupies in the record
<i>fieldname</i>	Field name in the format
<i>formatname</i>	Format name
<i>machine_type</i>	Type of computer that produced the data, such as a SPARCstation See List all existing machine types on page 236 .

Attribute	Attribute value
<i>offset_bytes</i>	Number of bytes of the field offset in the record
<i>projectname</i>	Name of existing project

Use the following syntax to create a COBOL-format object from a specification file:

```
BEGIN OBJECT COBOLFORMAT formatname

# Compulsory Attributes
PROJECT projectname
CHARACTERSET data_codeset
MACHINE machine_type
DRIVER driver_type
BEGIN SEQUENCE
FIELDNAME fieldname
PICTURE picture_description
USAGE usage_description
END SEQUENCE

END OBJECT
```

The following table lists the attributes and their values.

Attribute	Attribute value
<i>data_codeset</i>	Code set used to translate data in the data table For more information about data code sets, see the following file on your product CD: <code>\$INFORMIXDIR/gls/cm3/registry</code>
<i>driver_type</i>	Type of driver: COBOL (default) or COBOL_b
<i>fieldname</i>	Field name in the format
<i>formatname</i>	Format name
<i>machine_type</i>	Type of computer that produced the data, such as a SPARCstation
<i>picture_description</i>	Picture description that matches the record FD from the COBOL program See Picture strings.
<i>projectname</i>	Name of existing project
<i>usage_description</i>	Number of bytes that the field occupies in the record

Use the following syntax to create a delimited-format object:

```
BEGIN OBJECT DELIMITEDFORMAT formatname

# Compulsory Attributes
PROJECT projectname
CHARACTERSET data_codeset
```

```

RECORDSTART recordstart_delimit_character
RECORDEND recordend_delimit_character
FIELDSTART fieldstart_delimit_character
FIELDEND fielndelimit_character
FIELDSEPARATOR fieldseparator_delimit_character
BEGIN SEQUENCE
FIELDNAME format_fieldname
FIELDTYPE field_datatype
END SEQUENCE

END OBJECT
    
```

The following table lists the attributes and their values.

Attribute	Attribute value
<i>data_codeset</i>	Code set used to translate data in the data table For more information about data code sets, see the following file on your product CD: <code>\$INFORMIXDIR/gls/cm3/registry</code>
<i>field_datatype</i>	Type of field data See Data types allowed in a delimited format on page 98 .
<i>fielndelimit_character</i>	Delimiting character that specifies the end of the field, in hexadecimal or decimal format Begin a hexadecimal character with <code>0x</code> .
<i>fieldstart_delimit_character</i>	Delimiting character that specifies the start of the field, in hexadecimal or decimal format Begin a hexadecimal character with <code>0x</code> .
<i>fieldseparator_delimit_character</i>	Delimiting character that specifies the field separator, in hexadecimal or decimal format Begin a hexadecimal character with <code>0x</code> .
<i>format_fieldname</i>	Format field name
<i>formatname</i>	Format name
<i>recordend_delimit_character</i>	Delimiting character that specifies the end of the record, in hexadecimal or decimal format Begin a hexadecimal character with <code>0x</code> .
<i>recordstart_delimit_character</i>	Delimiting character that specifies the start of the record, in hexadecimal or decimal format

Attribute	Attribute value
	Begin a hexadecimal character with 0x.
<i>projectname</i>	Name of existing project

Related reference

[Modify a format by using a specification file on page 227](#)

[Describe a format on page 227](#)

Modify a format by using a specification file

You can only modify a format by using a specification file.

Use the syntax shown in [Specification-file conventions on page 200](#) to modify a format.

Related reference

[Create a format on page 224](#)

Describe a format

The following diagram illustrates the syntax to describe a format to a file.

Describing a format

```
onploadm describe format formatname [ { | -p projectname | -F specfilename | -s servername } ]
```

Element	Purpose	Key considerations
-F <i>specfilename</i>	Sets the specification file	The default value is the standard output.
<i>formatname</i>	Sets the format name	None
-p <i>projectname</i>	Identifies the project where the format and map are stored	None
-S <i>servername</i>	Sets the onpload database server	The default is the value of the INFORMIXSERVER environment variable.

Related reference

[Create a format on page 224](#)

List all formats in a project

The following diagram illustrates the syntax to list all formats in a project to standard output.

Listing all formats in a project

```
onpladm list format [{|-p projectname |-s servername}]
```

Element	Purpose	Key considerations
-p <i>projectname</i>	Identifies the project where the format and map are stored	None
-S <i>servername</i>	Sets the onpload database server	The default is the value of the INFORMIXSERVER environment variable.

Delete a format

The following diagram illustrates the syntax to delete a format.

Deleting all formats in a project

```
onpladm delete format formatname [{|-p projectname |-s servername}]
```

Element	Purpose	Key considerations
<i>formatname</i>	Sets the format name	None
-p <i>projectname</i>	Identifies the project where the format and map are stored	None
-S <i>servername</i>	Sets the onpload database server	The default is the value of the INFORMIXSERVER environment variable.

Define queries

You can create, modify, describe, list, and delete queries with the onpladm utility.

Creating a query

You can only create a query by using a detailed specification file.

Use the syntax shown in [Specification-file conventions on page 200](#) to create a query.

Use the following syntax to create a specification file for a query:

```
BEGIN OBJECT QUERY queryname
# Compulsory Attributes
PROJECT projectname
DATABASE targetdatabasename
SELECTSTATEMENT sql_statement
```

END OBJECT

The following table lists the attributes and their values.

Attribute	Attribute value
<i>queryname</i>	Query name
<i>projectname</i>	Name of an existing project
<i>sql_statement</i>	SQL SELECT statement, in quotation marks, that contains unload criteria
<i>targetdatabasename</i>	Name of the database that the records will be loaded or unloaded to

Related reference

[Modify a query on page 229](#)

[Describing a query on page 229](#)

Modify a query

You can only modify a query by using a detailed specification file.

Use the syntax shown in [Specification-file conventions on page 200](#) to modify a query by using a specification file.

Related reference

[Creating a query on page 228](#)

Describing a query

The following diagram illustrates the syntax to describe a query.

Describing a query

```
onpladm describe query queryname [ { | -p projectname | -F specfilename | -s servername } ]
```

Element	Purpose	Key considerations
-F <i>specfilename</i>	Sets the specification file	The default value is the standard output.
<i>queryname</i>	Sets the query name	None
-p <i>projectname</i>	Identifies the project where the format and map are stored	None
-S <i>servername</i>	Sets the onpload database server	The default is the value of the INFORMIXSERVER environment variable.

Related reference[Creating a query on page 228](#)

List all queries in a project

The following diagram illustrates the syntax to list all queries in a project to standard output.

Listing all queries in a project

```
onpladm list query [ { | -p projectname | -s servername } ]
```

Element	Purpose	Key considerations
-p <i>projectname</i>	Identifies the project where the format and map are stored	None
-S <i>servername</i>	Sets the onpload database server	The default is the value of the INFORMIXSERVER environment variable.

Delete a query

The following diagram illustrates the syntax to delete a query.

Deleting a query

```
onpladm delete query queryname [ { | -p projectname | -s servername } ]
```

Element	Purpose	Key considerations
-p <i>projectname</i>	Identifies the project where the format and map are stored	None
<i>queryname</i>	Sets the name of the query	None
-S <i>servername</i>	Sets the onpload database server	The default is the value of the INFORMIXSERVER environment variable.

Define filters

You can create, modify, describe, list, and delete filters with the onpladm utility.

Related reference[Example of using filter on page 135](#)

Create a filter

You can only create a filter by using a specification file.

Use the syntax shown in [Specification-file conventions on page 200](#) to create a filter by using a specification file.

Use the following syntax to create a filter by using a specification file:

```
BEGIN OBJECT FILTER filtername
# Compulsory Attributes
PROJECT projectname
FORMAT formatname
BEGIN SEQUENCE
FIELDNAME data_file_fieldname
STATUS record_status
MATCH match_criteria
END SEQUENCE
END OBJECT
```

The following table lists the attributes and their values.

Attribute	Attribute value
<i>data_file_fieldname</i>	Data-file field to be used in the match condition
<i>formatname</i>	Associated format name
<i>filtername</i>	Filter name
<i>match_criteria</i>	Match criteria, in quotation marks. See Match condition operators and characters.
<i>projectname</i>	Name of existing project
<i>record_status</i>	Type a K to keep records that meet a match condition or D to discard them.

Related reference

[Modify a filter on page 231](#)

[Describe a filter on page 232](#)

Modify a filter

You can only modify a filter by using a detailed specification file.

Use the syntax shown in [Specification-file conventions on page 200](#) to modify a filter.

Related reference

[Create a filter on page 231](#)

Describe a filter

The following diagram illustrates the syntax to describe a filter.

Describing a filter

```
onpladm describe filter filtername [ { | -p projectname | -F specfilename | -s servername } ]
```

Element	Purpose	Key considerations
-F <i>specfilename</i>	Sets the specification file	The default value is the standard output.
<i>filtername</i>	Sets the filter name	None
-p <i>projectname</i>	Identifies the project where the format and map are stored	None
-S <i>servername</i>	Sets the onpload database server	The default is the value of the INFORMIXSERVER environment variable.

Related reference

[Create a filter on page 231](#)

List all filters in a project

The following diagram illustrates the syntax to list all the filters in a project.

Listing all filters in a project

```
onpladm list filter [ { | -p projectname | -s servername } ]
```

Element	Purpose	Key considerations
-p <i>projectname</i>	Identifies the project where the format and map are stored	None
-S <i>servername</i>	Sets the onpload database server	The default is the value of the INFORMIXSERVER environment variable.

Delete a filter

The following diagram illustrates the syntax to delete a filter.

Deleting a filter

```
onpladm delete filter filtername [ { | -p projectname | -s servername } ]
```


Element	Purpose	Key considerations
<i>filtername</i>	Sets the name of the filter	None
-p <i>projectname</i>	Identifies the project where the format and map are stored	None
-S <i>servername</i>	Sets the onpload database server	The default is the value of the INFORMIXSERVER environment variable.

Define projects

You can create a project, run all jobs in a project, list all projects, and delete a project with the onpladm utility.

Related information

[Project organization on page 67](#)

Create a project

The following diagram illustrates the syntax to create a new, empty project to which you can add the HPL jobs.

Creating a project

```
onpladm create project projectname [-s servername]
```

Element	Purpose	Key considerations
-p <i>projectname</i>	Identifies the project where the format and map are stored	None
-S <i>servername</i>	Sets the onpload database server	The default is the value of the INFORMIXSERVER environment variable.

Run all jobs in a project

The following diagram illustrates the syntax to run all load or unload jobs in a project.

Running all jobs in a project

```
onpladm run project projectname -f[{l | u}][{-l logfile | -s servername}]
```

Element	Purpose	Key considerations
-f	Flags to specify the type of job	The default is load job.
l	Specifies a load job	None
u	Specifies an unload job	None

Element	Purpose	Key considerations
<code>-l logfile</code>	Sets the path for a log file where job progress is recorded	None
<code>projectname</code>	Identifies the project where the format and map are stored	None
<code>-S servername</code>	Sets the onpload database server	The default is the value of the INFORMIXSERVER environment variable.

Related reference

The onpladm utility on Windows

List all projects

The following diagram illustrates the syntax to list all your projects.

Listing all projects

```
onpladm list project [-s servername]
```

Element	Purpose	Key considerations
<code>-S servername</code>	Sets the onpload database server	The default is the value of the INFORMIXSERVER environment variable.

Delete a project

The following diagram illustrates the syntax to delete a project, all of its corresponding jobs, and other HPL objects that it holds.

Deleting a project

```
onpladm delete project projectname [-s servername]
```

Element	Purpose	Key considerations
<code>projectname</code>	Identifies the project where the format and map are stored	None
<code>-S servername</code>	Sets the onpload database server	The default is the value of the INFORMIXSERVER environment variable.

Define machine types

You can create, modify, describe, list, and delete machine types with the `onpladm` utility.

Create a machine type

You can only create a machine type by using a specification file.

Use the syntax shown in [Specification-file conventions on page 200](#) to create a machine type.

Use the following syntax to create a machine object by using a specification file:

```
BEGIN OBJECT MACHINE machinename
# Compulsory Attributes
BYTEORDER machinetype_byteorder
SHORTSIZE shortinteger_bytes
INTEGERSIZE integer_bytes
LONGSIZE longinteger_bytes
FLOATSIZE float_bytes
DOUBLESIZE double_bytes

END OBJECT
```

The following table lists the attributes and their values.

Attribute	Attribute value
<i>double_bytes</i>	Double integer size
<i>float_bytes</i>	Float size
<i>integer_bytes</i>	Integer size
<i>machinename</i>	Machine name
<i>longinteger_bytes</i>	Long integer size
<i>machinetype_byteorder</i>	Computer byte-order type Enter <code>MSB</code> for most-significant bit or <code>LSB</code> for least-significant bit.
<i>shortinteger_bytes</i>	Short integer size

Related reference

[Modify a machine type on page 235](#)

[Describe a machine on page 236](#)

Modify a machine type

You can only modify a machine type by using a detailed specification file.

Use the syntax shown in [Specification-file conventions on page 200](#) to modify a machine type.

Related reference

[Create a machine type on page 235](#)

Describe a machine

The following diagram illustrates the syntax to describe a machine.

Describing a machine

```
onpladm describe machine machinename [{ | -F specfilename | -s servername }]
```

Element	Purpose	Key considerations
-F <i>specfilename</i>	Sets the specification file	The default value is the standard output.
<i>machinename</i>	Sets the machine name	None
-S <i>servername</i>	Sets the onpload database server	The default is the value of the INFORMIXSERVER environment variable.

Related reference

[Create a machine type on page 235](#)

List all existing machine types

The following diagram illustrates the syntax to list the machine types.

Listing all machines

```
onpladm list machine [-s servername]
```

Element	Purpose	Key considerations
-S <i>servername</i>	Sets the onpload database server	The default is the value of the INFORMIXSERVER environment variable.

Delete a machine type

You can only delete a machine type by using a specification file.

The following diagram illustrates the syntax to delete a machine type.

Deleting a Machine

```
onpladm delete machine machinename [-s servername ]
```

Element	Purpose	Key considerations
<i>machinename</i>	Sets the machine type	None
-S <i>servername</i>	Sets the onpload database server	The default is the value of the INFORMIXSERVER environment variable.

Define database operations

You can perform the following database operations with the onpladm utility:

- Create load and unload jobs for all tables in a database with a single command.
- Load or unload all tables in a database.

Create a database project

When you create a database project with a single command, the onpladm utility:

- Creates all load or unload jobs for every table in a database
- Creates all the associated project HPL objects required for the jobs
- Groups the load or unload jobs and associated project HPL jobs into a single project that has the same name as the database

When you create a database project, you must specify the data-files source directory name or the tape-device path.

If you specify the data-files source directory, the files that the onpladm utility creates have the following format:

```
PREFIX_DATABASE_TABLE.unl
```

PREFIX is an option that you specify on the command line, *DATABASE* is the name of the target database, and *TABLE* is the name of the target-table name.

If you do not specify a prefix, the onpladm utility creates files of the following format:

```
DATABASE_TABLE.unl
```

The onpladm utility truncates the format file name if it is longer than the maximum file name length of 18 characters.

The following diagram illustrates the syntax to create a project for all the tables in a database.

Creating a database project

```
onpladm create project projectname -adevice -ddatabase [ <Setting the run mode> ] [ { -n <Setting the format> } ] [ -pprefix ] [ -rtarget ] [ -sserver ] [ -mdevicesize ] [ -bblocksize ]
```

Element	Purpose	Key considerations
-B <i>blocksize</i>	Sets the tape I/O block size (bytes)	No default value.
-d <i>device</i>	Sets the name of device, such as a file, device array, tape, or pipe	No default value.
-D <i>database</i>	Name of the target database that contains the information to be loaded or unloaded	No default value.
-M <i>devicesize</i>	Tape device size in kilobytes	The device size must be greater than zero.
-n	Sets no-conversion express job	None
-P <i>prefix</i>	Filename prefix for the files to be used as devices	None
<i>projectname</i>	Identifies the project where the format and map are stored	None
-S <i>server</i>	Sets the onpload database server	The default is the value of the INFORMIXSERVER environment variable.
-T <i>target</i>	Name of the target server to which the data will download	The default is the value of the INFORMIXSERVER environment variable.

The following diagram illustrates the syntax to set the run mode with the **-f** option.

Setting the run mode

“ -f ”

“ [a] ”

“ [c] ”

“ [N] ”

Element	Purpose	Key considerations
c	Sets mode to deluxe mode.	If this flag is not set, onpladm uses express mode.
d	Treats data source as a tape device	None
N	Allows deluxe mode load without replication	None

The following diagram illustrates the syntax to set the format type with the **-z** option.

Setting the format

```
" -z "
```

```
" [{ D | FI | FA | FB | C | CB } ] "
```

Element	Purpose	Key considerations
D	Sets the format to delimited	This is the default value. See Delimited records on page 97 .
FI	Sets the format to fixed internal	See Fixed-length records on page 86 .
FA	Sets the format to fixed ASCII	See Fixed-length records on page 86 .
FB	Sets the format to fixed binary	See Fixed-length records on page 86 .
C	Sets the format to COBOL	See COBOL records on page 100 .
CB	Sets the format to COBOL (byte)	See COBOL records on page 100 .

Configure target-server attributes

To configure target-server attributes, you must set target-server attribute values and list target-server defaults.

Set target-server attribute values

The following diagram illustrates the syntax to set default values for target server attributes.

Setting target server attribute values

```
onpladm configure defaults [ -s servername ] -m machinetype -c data_codeset [ -s servername ]
```

Element	Purpose	Key considerations
-c data_codeset	Character set for data files	The character set of the database is determined by the DB_LOCALE environment variable. For information about locales and code sets, see the <i>HCL® Informix® GLS User's Guide</i> .
-m machinetype	Machine type	None
-s servername	Database server for which defaults are set	If a server is not specified, the default information within the onpload database that describes all database servers is modified.

Element	Purpose	Key considerations
-S <i>servername</i>	Sets the onpload database server	The default is the value of the INFORMIXSERVER environment variable.

List target-server defaults

The following diagram illustrates the syntax to list target-server default values.

Listing target server defaults

```
onpladm list defaults [-s servername] [-s servername]
```

Element	Purpose	Key considerations
-s <i>servername</i>	Database server for which defaults are set	If a server is not specified, the default information within the onpload database that describes all database servers is modified.
-S <i>servername</i>	Sets the onpload database server	The default is the value of the INFORMIXSERVER environment variable.

Index

Special Characters

.flt file 182

A

Active Job window 39, 146, 146, 156, 163, 164, 169
AIO error code 27 156
AIOBUFFERS configuration parameter 189
 affecting onpload processes 183
 example 186
AIOBUFSIZE configuration parameter 189
 affecting onpload process 183
 example 186
Alter table schema 185
ALTER TABLE statement
 format to use 166
Assigning records to devices 80
Autogenerate Unload Components
 window 40, 161, 163

B

Binary data
 format 76
Block size of tape 195, 203, 237
Browse option
 defined 171
 log file 175
Browsers menu
 defined 46
 Logfile option 175
 Record option 171
 Violations option 173
Buffer size
 I/O, with onpload 195
 server stream buffer 195
BYTE data
 amount to transfer 128
 order of binary information 73, 77
 size of data type 89
 size of variables 77

C

cadiload threads 15
Carriage returns, in fixed format 92
Case conversion 128, 198
Changing
 unload job options 158
CHAR data type 188
Character
 case conversion 198
 set, modifying 104, 104
COBOL format 101
 byte
 setting with onpladm 203, 217, 237
 used with generate 166
 creating 101
 definition window 100
 records 100
 setting with onpladm 203, 217, 237
 used with generate 166
Code sets
 conversion 142
 data file 73
 database 73, 73
 delimited formats 104
 Fixed and COBOL formats 104
 Global Language Support (GLS) 73, 104, 104, 142

 modify 104, 104
 setting with onpladm 224, 239
Column Selection window 111, 111
Columns
 characteristics 131
 default values 128
 drop, add, modify 185
 offset, in mapping options 128
Commit
 interval
 load job 158
Components menu
 defined 46
 devices 80
 filter 136
 formats 86
 generate 167
 maps 122, 124
 query 109, 115
Computer
 configuration, reorganize 185
 modifying 76
Configuration file
 HPL 198
 onpload 12
Configuration parameters
 thread control 183
Configure menu 46
Configuring
 ipload utility 70
Confirm delete window 59
Confirmation window
 file overwrite 117
Connect Server window 71, 71
Constraints
 checking 178
 table-level referential ones when using
 onpladm 213
 violations 181
Converter threads 14, 188, 198, 198
CONVERTTHREADS configuration parameter
 affecting onpload process 183
 example 186
CONVERTVPS configuration parameter
 affecting onpload process 183
 example 186
Copy Data window
 illustration 58
 using 58
Copy existing format 58
cron
 job 143, 151
Custom driver
 adding 76
Custom file handling software 75
Custom input to pipes 75

D

Data
 unload using onpload 193
Data conversion
 onpload 193
Data file
 formats supported by ipload 85
 structure 85
Data masking 128
Data source, for onpload 193, 193
Data types
 COBOL 100
 fixed format 89
Database servers
 limitations 151
 listing attributes with onpladm 240
 selecting 71
 setting attributes with onpladm 239
 target server 143, 151
Database Views window 118
Databases
 create for example 19
 creating project with onpladm 237
 name, override in onpload 198
 unloading records 144
DATE data type 188
DB_LOCALE environment variable 73
DBDELIMITER environment variable 104
DBONPLOAD environment variable 10, 11, 72
 defined 11
DECIMAL data type 188
Decimals 89
Defaults
 name of log file 175
 name of rejects file 175
 values, example 21
Defaults window 56, 73
Define format
 delimited records 97, 101
 editing a format 91
 fixed-length records 86
 modifying a format 103
Definition window 101
Delete existing format 59
Delimited format
 creating 97, 101
 modifying 104
 setting with onpladm 203, 217, 237
 using a .csv file 106
 using simple large objects 98
Delimited Format window 27, 98, 99
Delimited record, defined 97
Delimiter characters
 changing 104
 defined 97
Delimiter Options window 104
Deluxe mode
 choosing from load options 158
 compare to express 180
 defined 177
 in High-Performance Loader, defined 7
 INSERT statements 177
 mentioned 177
 without replication 177
Device array
 defined 79
 device types 80, 81
 editing 84
 example 24
 improving performance 184
 onpladm
 creating 214
 deleting 216
 describing 215
 listing all in a project 216
 modifying 215
 using 203, 208
 onpload
 using 193

- speed 184
- tape parameters 81
- Device Array Definition window 25, 50, 81, 81
- Device Array Selection window 24, 48, 80
- Dirty Read isolation level 148
- Driver
 - class, format type 75
 - modifying 104
 - name of custom driver 75
- Drivers window
 - using 76, 79

E

- EBCDIC data, generating 166
- Environment variables
 - DB_LOCALE 73
 - DBDELIMITER 104
 - DBONPLOAD 10, 11, 11
 - IFX_ONPLOAD_AUTO_UPGRADE 10
 - INFORMIXDIR 10
 - INFORMIXSERVER 10
 - LD_LIBRARY_PATH 10, 10
 - ONCONFIG 10
 - PDQPRIORITY 10
 - PLCONFIG 10, 12
 - PLOAD_LO_PATH 10, 13
 - PLOAD_SHMAT 10
 - PLOAD_SHMBASE 10, 12, 13
- Error code 27 156
- Errors
 - constraint violations 182
 - maximum number allowed 148, 158, 195
 - onpladm utility 202
- Exporting a query 116
- Express mode 177
 - choosing from load options 158
 - compare to deluxe 180, 180
 - foreign key constraints 180
 - in High-Performance Loader, defined 7
 - level-0 backup 156
 - load example 187
 - page size limitation 178
 - sequence of events 179

F

- Fast format 103
- Fast job
 - defined 103
- Fast Job Startup window 169
- Field
 - set minimum and/or maximum 128
 - set offset 128
- Figure
 - extracting data from a table 108
 - foreign-key constraints 180
 - load and unload modes 177
 - maps found by a search 134
 - view indicator 130
- File descriptor, COBOL 102
- Files
 - .flt 182
 - COBOL 100
 - default onpload configuration 12
 - import/export queries 116
 - onpload.std 12
 - path name for I/O 191
 - plconfig 198
- Fill character, mapping options 128
- Filter
 - adding 140
 - conversion of code set 142
 - creating 136, 136

- defined 135
- deleting 141
- editing 139, 139, 139
- example 135
- inserting items 140
- match conditions 138
- mode, with constraints 182
- onpladm
 - creating 231
 - deleting 232
 - describing 232
 - listing all in project 232
 - modifying 231
 - rejected records 182
- Filter Views window 141
- Filters window 136
- Find button 130
- Find Node window 130
- Fixed binary format
 - format type group 166
 - setting with onpladm 203, 217, 237
- Fixed format
 - data types 89
 - defined 86
 - setting with onpladm 203, 217, 237
 - using carriage returns 92
 - using simple large objects 93, 95
- Fixed Format Definition window 86, 92, 93, 95
- Fixed Format edit window 86
- Fixed Format Options window 104
- Fixed internal format
 - mentioned 103
 - setting with onpladm 203, 217, 237
 - used in generate 166
- Fixed length 85
- FLOAT data type 188
- Foreign key constraint 180
- Format
 - COBOL 100, 166, 166
 - copy 58
 - create 86
 - defined 85
 - delete 59
 - delimited 85
 - fast job 103
 - fast, defined 103
 - file structure supported in HPL 85
 - fixed internal 166
 - onpladm
 - creating 224
 - deleting 228
 - describing 227
 - listing all in project 228
 - modifying 227
 - performance 188
 - steps for editing 91, 91, 91, 92
 - testing 170
 - types supported by HPL 85
 - types used by generate 166
- Format Views window 26, 54, 107
- Fragmenter threads 15
- Functions
 - user-defined in mapping options 128

G

- Generate
 - assumptions 167
 - defined 161, 161
 - EBCDIC data 166
 - format types 166
 - from unload job 162

- no-conversion job 169
- types of load and unload tasks 161
- Generate window 165, 167
- Global Language Support (GLS) 73, 104, 104, 142

H

- Help
 - High-Performance Loader menu
 - description 46
 - using online help 66
- High-Performance Loader
 - components 8
 - configuration file 198
 - configuring it 70
 - data-load process 5
 - data-unload process 6
 - features 5
 - ipload utility 8, 19, 20, 45, 67
 - main window 20
 - managing 177
 - modes 7, 37, 177
 - onpladm utility 9, 199
 - onpload database 9, 67, 72
 - onpload utility 8, 13, 74, 190
 - overview 5
 - prerequisites 10
 - usage tasks 185

I

- I/O
 - buffer size 195
 - number of tapes to load 195
 - tape block size 195
- IFX_ONPLOAD_AUTO_UPGRADE environment variable 10
- Import/Export File Selection window 116
- Importing query 116
- INFORMIXDIR environment variable 10
- INFORMIXSERVER environment variable 10
- Input
 - starting record 195
- INSERT statements 177
- INT data type 188
- Internal format
 - limitations 103
 - use with generate 169
- ipload utility
 - configuring 70
 - creating a project 67, 69
 - defined 5, 8
 - GUI 46
 - Projects window 68
 - purpose 8
 - starting 20, 45, 45
 - using 19, 45
- Isolation level
 - committed 148
 - cursor stability 148
 - Repeatable Read 148
 - setting with onpladm 207
 - unload option 148

J

- Jobs
 - conversion 202
 - menu, defined 46
 - no-conversion 202
- Justification of data in mapping options 128

L

- LD_LIBRARY_PATH environment variable 10, 10
- Level-0 backup
 - express mode 39, 179
- Limitations, database server 151
- Load and unload session
 - maximum errors 195
- Load data
 - onpladm 203, 208
 - onpload 193
- Load job
 - changing options 158, 158
 - commit interval 158
 - components 150
 - creating 153, 153
 - defined 150
 - device-array speed 184
 - editing 160
 - example 22
 - from the command line 190
 - generate violations records 158
 - log file 175
 - maximum errors 158
 - mode options 158
 - multiple jobs 143, 151
 - number of records 158
 - onpladm
 - conversion 202
 - creating for every table 237
 - creating no-conversion 208
 - deleting 214
 - describing 211
 - listing all in project 212
 - modifying 211
 - running 212
 - preview records 170
 - run example 38, 44
 - running 156, 160
 - server considerations 143, 151
 - starting record 158
 - status log 153
 - tapes, number of 158
 - using cron 143, 151
- Load Job Select window
 - command line information 157, 157
 - illustration 22, 153
- Load Job window 22, 29, 34, 52, 153
- Load log
 - examining 175
- Load map
 - how to create 122
- Load Options window 37, 158
- Load Record Maps window 31
- Log files
 - created by ipload 153
 - for load job 175
 - messages 156
 - sample entry 176
 - setting 144
- Lowercase conversion 128, 198

M

- Machine type
 - High-Performance Loader default 73
 - modifying 76, 104
 - onpladm
 - creating 235
 - deleting 236
 - describing 236
 - listing all 236
 - modifying 235

- Machines window
 - illustration 77
 - using 78
- Map
 - columns and fields of same name 122, 124
 - defined 119
 - in-line blobs 93
 - onpladm
 - creating 217
 - deleting 222
 - describing 222
 - listing all in project 223
 - modifying 223
- Map Views window 31, 132, 133
- Map-definition window 34, 34, 120
- Map-Definition window 34, 122, 124, 126, 130
- Map-edit window
 - defined 121
 - purpose 120
 - using the find button 130
- Mapping options
 - bytes to transfer 128
 - case conversion 128
 - column offset 128
 - default column value 128
 - defining 127
 - field minimum and/or maximum 128
 - field offset 128
 - fill character 128
 - function, user-defined 128
 - justification 128
 - picture format 128
 - steps to define 127
 - summary 127
 - symbol 127
- Mapping Options window 127
- Masking data 128
- Match conditions
 - defined 135
 - filter 138
 - WHERE clause 114
- Maximum
 - number of errors 148
- Message log file
 - path name in onpload 198
- Message window 56
- Mode options, load job 158
- Modify format
 - COBOL 104
 - delimited 104
 - fixed 104
- MONEY data type 188
- Multiple load or unload jobs 143, 151

N

- No-conversion job
 - changing computer configuration 185
 - defined 103
 - load example 186
 - option 169
 - restrictions 103
 - run fast job 169
 - running the job 169
 - using onpladm 202
 - using onpload 193
 - with hidden columns 186
- NOT NULL violation 181
- Notes window 60
- null UDT data 89
- Number of conversion threads 198, 198
- Number of records in a load job 158

- Number of records to process
 - assigned in onpload 195
- Number of tapes to load 195

O

- ONCONFIG environment variable 10
- Online help
 - High-Performance Loader menu 46
 - how to use 66
- onpladm command
 - B option 203
 - f option 203
 - F option 200
 - M option 203, 237
 - n option 203
 - P option 237
 - R option 211
 - S option 200, 203
 - T option 203
 - z option 203, 217, 237
 - Z option 212
 - setting format type 203, 217, 237
 - setting run mode 203
 - specification file syntax 200
- onpladm utility
 - creating no-conversion jobs 208
 - creating onpload database 199
 - database project, creating 237
 - defined 5, 9, 199
 - defining jobs 202
 - device arrays, defining 214
 - error handling 202
 - features of 199
 - filters, defining 230
 - formats, defining 223
 - machine types, defining 235
 - maps, defining 217
 - projects, defining 233
 - queries, defining 228
 - specification file conventions 200
 - target server attributes, listing 240
 - target server attributes, setting 239
- Onpladm utility
 - using when tables have constraints 213
- onpload command
 - d option 194
 - i option 197
 - M option 188
 - Z option 191
 - generated for Load Job 157
 - generated for Unload Job 147
 - starting 190
 - syntax 191
 - usage 190
 - using with collection columns 198
- onpload database
 - connection to 71
 - creating 72
 - creating with onpladm 199
 - creation of 20
 - default settings 72
 - defined 5, 9
 - multiple 11
 - purpose of 67
 - select server 71
- onpload utility
 - architecture 13
 - configuration file 12
 - defined 5, 8, 190
 - deluxe mode 14
 - express mode 15

- filename size limits on UNIX 190
- flag for a fast job 103
- load data 193
- specifying defaults 74
- starting 190
- syntax 191
- using with collection columns 198
- onpload.std file 12
- onstat utility 184
- Options
 - load job 158
 - unload job 148
- Options symbol 127
- Organization of a Record that Includes In-Line TEXT Data 95

P

- Page size in express mode 178
- PDQPRIORITY environment variable 10, 188
- Performance
 - converter threads 188
 - hints for HPL 187
 - improving in HPL 182
 - VPs 188
- Picture description, COBOL 102
- Picture format, in mapping options 128
- Pipe
 - device arrays 75
 - in a device array 81
 - starting command 80
 - use with onpladm 203, 208
 - use with onpload 193
- pL_wkr threads 15
- plconfig configuration file
 - HPL value override 198
- PLCONFIG environment variable 10, 12
- PLCONFIG file
 - override I/O buffer size 195, 195, 198
- PLOAD_LO_PATH environment variable 10, 13
- PLOAD_SHMAT environment variable 10, 13
- PLOAD_SHMBASE
 - avoiding shared memory collision 12
- PLOAD_SHMBASE environment variable 10, 12
- Print button 62
- Privileges 151
- Problems during a load job 146, 156
- Project name, in onpload 191, 228, 233
- Projects
 - creating a new project 69
 - onpladm
 - creating 233
 - creating for all tables 237
 - deleting 234
 - listing all 234
 - running all jobs 233
- Projects window 68
- Proper-name case conversion 198
- Proper-name conversion 128

Q

- Query
 - defined 108
 - export to a file 116, 117
 - for unload map 124
 - import from a file 116, 116
 - onpladm
 - creating 228
 - deleting 230
 - describing 229
 - listing all in project 230
 - modifying 229
 - steps for defining 109

- using the Table button 111
- Query window 109
- Query-definition window 109, 111, 114

R

- Raw load and unload 103, 169
- Reading to end of device 157, 191, 212
- Record Browser window 171, 171
- Record filter 198
- Record Formats window 27, 60, 86
- Record map, assigned by onpload 191
- Records, number to process 195
- Rejected records 181
 - conversion errors 182
 - filter conditions 182
 - reviewing 173
- Reorganize computer configuration 185
- Repeatable read isolation level 148

S

- Schema, of database table 161
- sdriver threads 14
- SELECT clause, preparing 109
- Server name, High-Performance Loader default 73
- SET CONSTRAINTS statement
 - DISABLED 180
 - ON 182
- setrw threads 15
- Simple large objects
 - as inline data 93, 95, 95
 - BYTE data type 178
 - in delimited records 98
 - in fixed format 93, 95
 - in separate files 94, 96
 - TEXT and BYTE 178
- Smart large objects 178
- SMFLOAT data type 188
- Specification file for onpladm
 - conventions 200
 - conversion job, creating 205
 - device arrays, creating 214
 - device arrays, modifying 215
 - filters, creating 231
 - filters, modifying 231
 - formats, creating 224
 - formats, modifying 227
 - jobs, modifying 211
 - machine types, creating 235
 - machine types, deleting 236
 - machine types, modifying 235
 - maps, creating 218
 - maps, modifying 223
 - no-conversion jobs, creating 210
 - queries, creating 228
 - queries, modifying 229
 - syntax 200
- Specifications window 131
- Specs button 57, 131
- SQL statements, use in HPL 108
- sqlhosts file 71
- Start record
 - input 195
 - load job 158
- Stream threads 15
- STRMBUFFERS configuration parameter 189
 - example 186
- STRMBUFFERS parameter
 - affecting onpload process 183
- STRMBUFFERSIZE configuration parameter 189
 - affecting onpload process 183
 - example 186

- Swap bytes 195
- Symbol, mapping options 127
- Synonyms 111, 122
- Syntax
 - onpload utility 191

T

- Table
 - create for example 19
- Tape I/O threads 14
- Tape parameters, specifying 81
- Tape size
 - setting 81
- Tapes
 - block size 195
 - number of 158
 - number to load 195
 - reading to end 157, 191, 212
 - writing to end 147, 191, 212
- Target server 71, 143, 151
- Testing formats 170
- TEXT data type 178
- Threads
 - cadiload 15
 - convert 14
 - fragmenter 15
 - pL_wkr 15
 - sdriver 14
 - setrw 15
 - stream 15
 - tape I/O 14
 - ulstrm 17
 - unload-stream 17
 - worker 14
- Transfer bytes, in mapping options 128
- Troubleshooting
 - HPL load errors 181

U

- ulstrm threads 17
- Unload data
 - using onpload 193, 203, 208
- Unload job
 - changing the options 148
 - components of 142
 - creating 143
 - defined 142
 - example 40
 - from the command line 190
 - generate option 162
 - log file 175
 - multiple jobs 143, 151
 - onpladm
 - conversion 202
 - creating for every table 237
 - creating no-conversion 208
 - deleting 214
 - describing 211
 - listing all in project 212
 - modifying 211
 - running 212
 - options 148, 158
 - using cron 143
- Unload Job Select window
 - command line information 147
 - illustration 144
- Unload Job window 40, 40, 144, 163
- Unload map
 - defined 124
 - how to create 124
 - steps to create 124
- Unload Options window 148, 148

- Unload Record Maps window 124, 124
- Uppercase conversion 128, 198
- Usage description, COBOL 102
- Usage models for HPL 185

V

- VARCHAR data type 188
- View icon
 - defined 63
- View indicator, figure 130
- Views 111
- Violations
 - when loading records from a data file 181
- Violations table
 - generate from load job 158
 - viewing 173
- Violations Table Browser window 173, 173
- VPs, performance 188

W

- WHERE clause
 - match conditions 114
 - preparation 114
- Window
 - Delimited Format definition 98
 - Delimiter Options 104
 - Device Array Selection 80
 - device-array definition 81
 - Filter Views 141
 - Filters 136
 - Fixed Format 86
 - Fixed Format definition 92, 93
 - Fixed Format definition window 86
 - HPL main window 20
 - Load Job 22, 29
 - Load Job Select 22
 - Load Options 37
 - Machines 77
 - map definition 34
 - query definition 109
 - Record Browser 171
 - Record Format 86
- Worker threads 14
- Writing to end of device 147, 191, 212