HCL Informix 15.0.0

Informix SNMP Subagent Guide

Contents

Chapter 1. SNMP Subagent Guide

The describes the Simple Network Management Protocol (SNMP) and the software that you need to use SNMP to monitor and manage HCL Informix® database servers and databases.

These topics are written for the following users:

- Database server administrators
- Backup operators
- Performance engineers

These topics assumes 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 with database server administration, operating-system administration, or network administration

You must install additional software to use the HCL Informix® implementation of SNMP. For specific requirements, see Informix implementation of SNMP on page 8.

The onsnmp utility cannot be run on HDR secondary servers, remote standalone (RS) secondary servers, or shared disk (SD) secondary servers.

SNMP concepts

This section provides a brief introduction to Simple Network Management Protocol (SNMP).

What is SNMP?

The Simple Network Management Protocol (SNMP) is a published, open standard for network management. SNMP lets hardware and software components on networks provide information to network administrators.

Purpose of the SNMP

Although the original purpose of the Simple Network Management Protocol (SNMP) was to let network administrators remotely manage an Internet system, the design of SNMP lets network administrators manage applications and systems.

SNMP provides the following capabilities:

- Hides the underlying system network
- Lets you manage and monitor all network components from one console

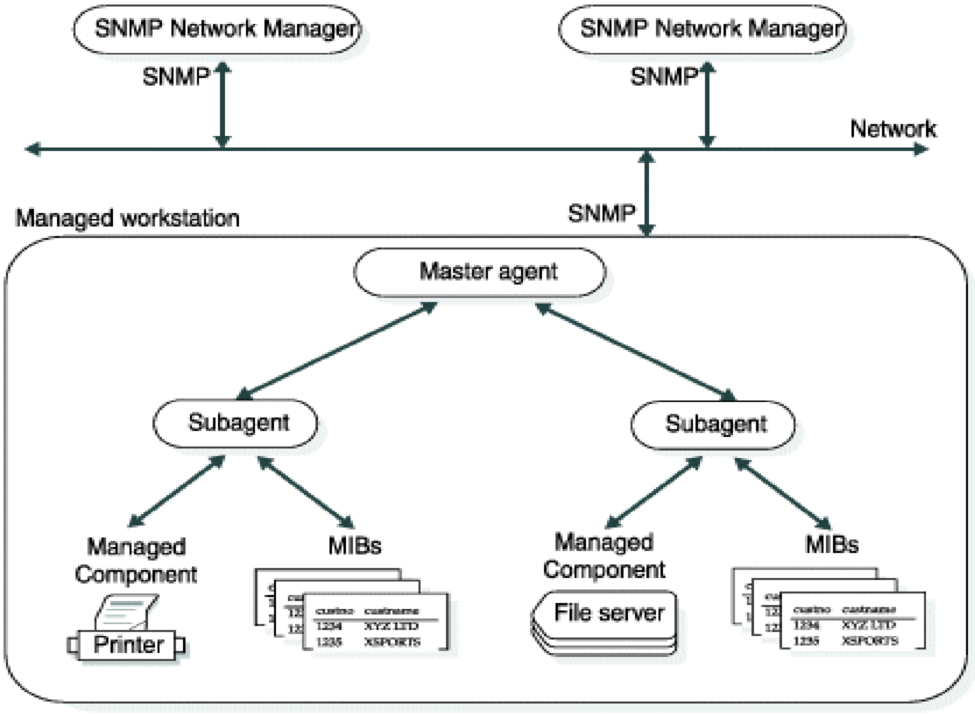
The SNMP architecture

The Simple Network Management Protocol (SNMP) architecture includes four layers.

As the following figure illustrates, the SNMP architecture includes the following layers:

- SNMP Network Managers
- Master agents
- Subagents
- Managed components

Figure 1. SNMP architecture



A network can have multiple SNMP Network Managers. Each workstation can have one master agent. The SNMP Network Managers and master agents use SNMP protocols to communicate with each other. Each managed component has a corresponding subagent and MIBs. SNMP does not specify the protocol for communications between master agents and subagents.

SNMP network managers

An SNMP Network Manager is a program that asks for information from master agents and displays that information. You can use most SNMP Network Managers to select the items to monitor and the form in which to display the information.

An SNMP Network Manager typically provides the following features:

- Remote monitoring of managed components
- Low-impact sampling of the performance of a managed component
- Correlation of managed component metrics with related system and network metrics
- Graphical presentation of information

Many hardware and network services have created SNMP Network Managers. For example:

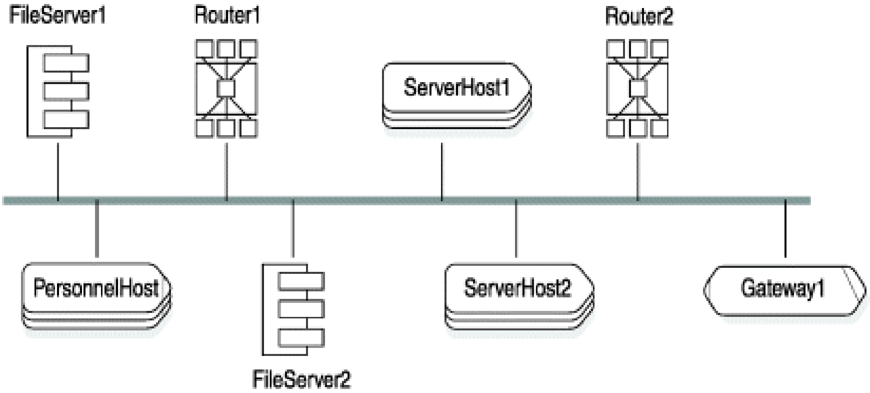
- CA-Unicenter
- Hewlett-Packard Open View
- IBM® NetView®/6000
- Novell Network Management System
- Sun Solstice
- Tivoli® TME 10 NetView®

SNMP Network Managers use a connectionless protocol, which means that each exchange between an SNMP Network Manager and a master agent is a separate transaction. A connectionless protocol allows the SNMP Network Manager to perform the following actions:

- Gather information without putting an excessive load on the network
- Function in an environment where heavy traffic can cause network problems

Most SNMP Network Managers provide a graphical user interface (GUI) such as the one that the following figure illustrates. With this SNMP Network Manager, you select a node to monitor and then choose specific information from a menu.

Figure 2. SNMP Network Manager example



The following code shows how an SNMP Network Manager might display information about the databases on a network. In this example, the network has only one database.

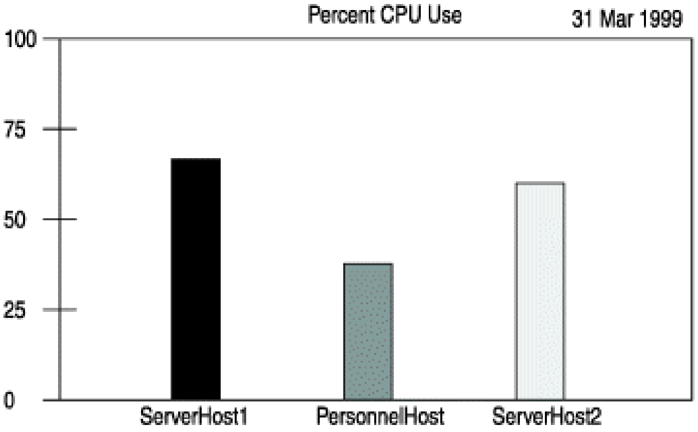
```
Feb 17 1999 [ smoke ] : RDBMS-MIB.rdbmsDbTable
KEY = 72000003
rdbmsDbName = CustomerData
rdbmsDbName.72000003 = AnotherData
rdbmsDbPrivateMibOID = 1.3.6.1.4.1.893
rdbmsDbVendorName = IBM Corporation
rdbmsDbName = CustomerData
rdbmsDbContact = John Doe
```

The following code shows how a different SNMP Network Manager could display the same information.

```
rdbmsDbPrivateMibOID.72000003 = 1.3.6.1.4.1.893
rdbmsDbVendorName.72000003 = IBM Corporation
rdbmsDbName.72000003 = CustomerData
rdbmsDbContact.72000003 = John Doe
```

In addition to text, an SNMP Network Manager might also display graphs or charts, as the following figure illustrates.

Figure 3. Example of monitoring information



Master agents

A master agent is a software program that provides the interface between an SNMP Network Manager and a subagent.

Each workstation that includes a managed component needs to have a master agent. Each managed workstation can have a different master agent. A master agent performs the following tasks:

1. Parses requests from the SNMP Network Manager
2. Routes requests from the SNMP Network Manager to the subagents
3. Collects and formats responses from the subagents
4. Returns the responses to the SNMP Network Manager
5. Notifies the SNMP Network Manager when a request is invalid or information is unavailable

Subagents

A subagent is a software program that provides information to a master agent.

Each managed component has a corresponding subagent. A subagent performs the following tasks:

1. Receives requests from the master agent
2. Collects the requested information
3. Returns the information to the master agent
4. Notifies the master agent when a request is invalid or information is unavailable

Managed components

A managed component is hardware or software that provides a subagent. For example, database servers, operating systems, routers, and printers can be managed components if they provide subagents.

Event notification

When an event occurs that affects the performance or availability of a managed component, the SNMP Network Manager can alert you to that condition.

The following list describes some of the decisions that you can make about event notification:

- Define the conditions that need to be monitored.
- Specify how frequently to poll for each condition.

  When you determine the polling frequency, you must balance the need for prompt notification of an undesirable condition and the burden that polling puts on the network.
- Specify how the SNMP Network Manager notifies you of an event.

  You might choose to have an icon flash or change colors when an event occurs.

Data requests

A data request can be a one-time request or a periodic request. A one-time request is useful for comparing the data for two managed components. Periodic requests are useful for accumulating statistical information about a managed component.

Traps

You can configure the SNMP Network Manager to detect extraordinary events and notify you when they occur.

The following list describes some of the decisions that you can make about traps:

- Define the conditions that need to generate a trap.
- Specify how the SNMP Network Manager notifies you of a trap.

  You might choose to have an icon flash or change colors when a trap occurs.
- Specify how the SNMP Network Manager responds to a trap.

  The SNMP Network Manager can query the managed component to determine the cause and extent of the problem.

Management Information Bases

A Management Information Base (MIB) is a group of tables that specify the information that a subagent provides to a master agent. MIBs follow SNMP protocols.

MIBs use a common interface definition language. The Structure of Management Information (SMI) defines this language and dictates how to use Abstract Syntax Notation One (ASN.1) to describe each table in the MIBs.

MIB table naming conventions

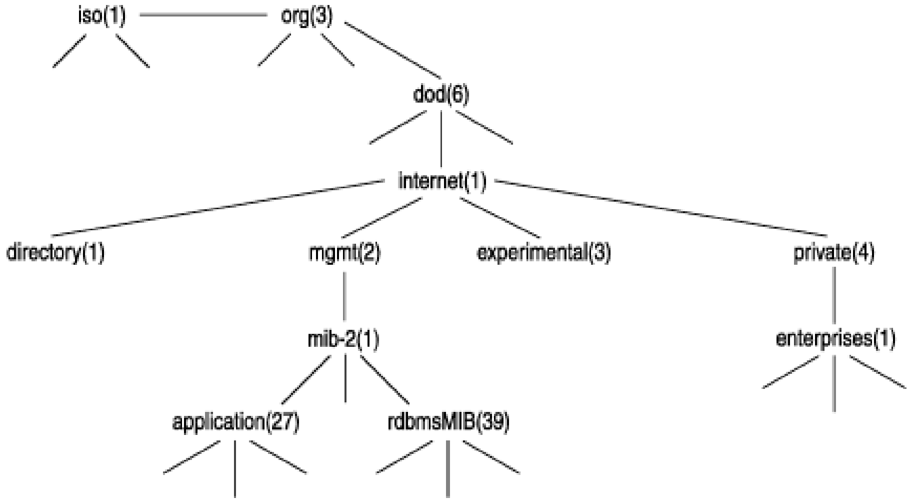
The name of each MIB table starts with the name of the MIB. Thus each table in the RDBMS MIB starts with **rdbms**. For example, the RDBMS MIB includes tables that are named **rdbmsSrvTable** and **rdbmsDbInfoTable**.

The name of each column in an MIB table starts with the name of the table, excluding **Table**. Thus, each column in **rdbmsSrvTable** starts with **rdbmsSrv**. For example, **rdbmsSrvVendorName** and **rdbmsSrvProductName** are columns in **rdbmsSrvTable**.

The MIB hierarchy

All MIBs are part of an information hierarchy that the Internet Assigned Numbers Authority (IANA) defines. The hierarchy defines how to name tables and columns and how to derive the numeric object identifiers (OIDs). The following figure shows the MIB hierarchy.

Figure 4. MIB hierarchy



Even though you rarely see the full path to a table, column, or value, the path is important because the SNMP components use the numeric equivalent of the path to locate data. For example, the following value is the path to the Application MIB:

```
iso.org.dod.internet.mgmt.mib-2.application
```

An OID is the numeric equivalent of a path. It uniquely describes each piece of data that an SNMP Network Manager can obtain and is written as a string of numbers separated by periods (.). For example, the following value is the OID for the Application MIB:

```
1.3.6.1.2.1.27
```

The following value is the OID for a value in the Application MIB:

```
1.3.6.1.2.1.27.1.1.8.2
```

The first part of this OID is the OID for the Application MIB. The final part of the OID assigns values sequentially to each table in the MIB, each column in the table, and each value in a column.

Informix® implementation of SNMP

The HCL Informix® implementation of SNMP lets database administrators monitor Informix® database servers and databases.

Components of the Informix® implementation

The HCL Informix® implementation consists of the following components:

- Master agent
    - On UNIX™, a master agent is provided through licensing agreements with vendors.
    - On Windows™, install the Microsoft™ SNMP Extendible master agent.
- Subagent

  The subagent for Informix® database servers is OnSNMP.

- Managed components

  In the Informix® implementation of SNMP, each database server is a managed component.

- MIBs

  OnSNMP uses several MIBs.

Purpose of Informix® SNMP

Event notification

You can configure an SNMP Network Manager to notify you when a specific event occurs.

An event usually has a corresponding object in an MIB table. The following table describes four possible events and the MIB objects that correspond to them.

Table 1. Possible events and the corresponding MIB objects

Event	MIB object
A database server is not available.	**onServerMode**
Database availability changed.	**rdbmsRelState**
A chunk failed.	**onChunkStatus**
A table is running out of space.	**onTablePagesAllocated** **onTablePagesUsed**

For example, you might discover that an application that uses the HCL Informix® database server stopped responding. You can send email to the help desk to report this problem. The help desk can tell you about the problem, and you can look at **onSessionTable** to determine the cause of the problem.

Data requests

You can issue a one-time data request to compare the configuration parameters of two database servers. You can issue periodic data requests to provide statistical information for assessing database performance or resource allocation.

For example, even if you use a database that is on a local host, you can call a remote technical support representative to report a problem. The problem might be that the data for the transactions running in a particular situation is less than expected. From the remote location, the technical support representative can query an SNMP Network Manager to determine the database server configuration, monitor the database server performance, and identify the bottleneck. OnSNMP provides this information to SNMP Network Managers through the master agent.

Traps

When the status of the database server changes from its current status to any status that is less available, OnSNMP sends a message to the SNMP Network Managers. For example, if a dbspace goes down, the database server status changes from full to limited availability. The message that OnSNMP sends is **rdbmsStateChange**, which is an unsolicited trap. When an SNMP Network Manager notifies you that it received an **rdbmsStateChange** trap, you can query the database server that generated the trap to determine the cause and extent of the problem.

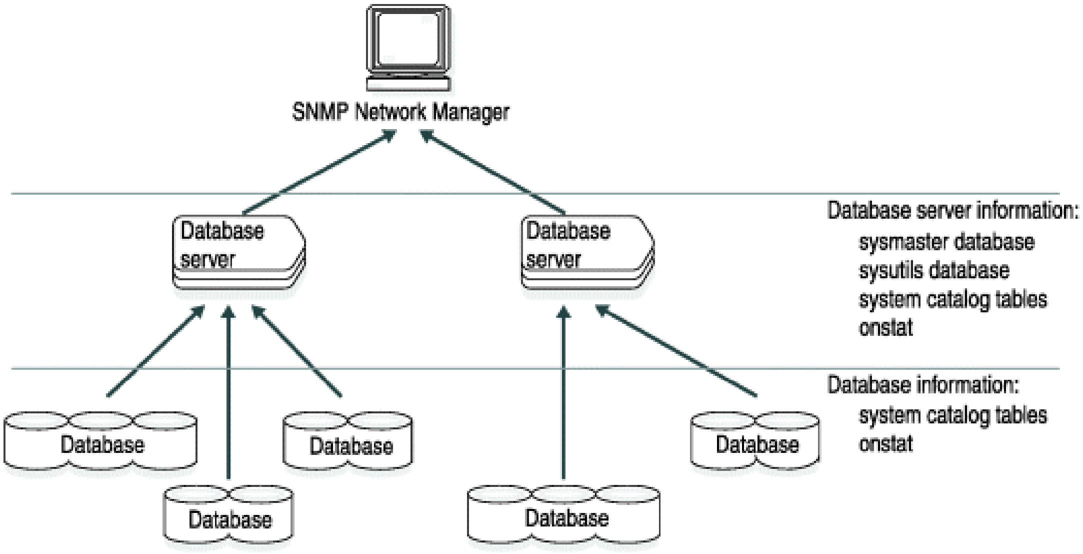
For example, the logical logs for a database server might become full and cause the database server to become unavailable. OnSNMP can notice that the database server is unavailable and send an **rdbmsStateChange** trap to an SNMP Network Manager. The SNMP Network Manager can make an icon flash to notify you of the problem. You can then send data requests to determine the cause of the failure.

Information that OnSNMP provides

All the information that OnSNMP provides is available from other sources, such as the system catalog tables, the **sysmaster** and **sysutils** databases, dbaccess calls, and the onstat utility.

However, the system catalog tables and the onstat utility refer only to a single database, and the **sysmaster** and **sysutils** databases refer only to a single database server. OnSNMP provides information that lets an SNMP Network Manager monitor all the HCL Informix® databases that are on a network. The following figure illustrates this concept.

Figure 5. Monitoring Informix® databases



SNMP standard

The SNMP standard has two versions: SNMPv1 and SNMPv2.

The following table lists the versions of the SNMP standard with which OnSNMP complies.

Table 2. Versions of the SNMP standard

Operating system	Version of the SNMP standard
UNIX™	SNMPv1 and SNMPv2
Windows™	SNMPv1

SNMP architecture

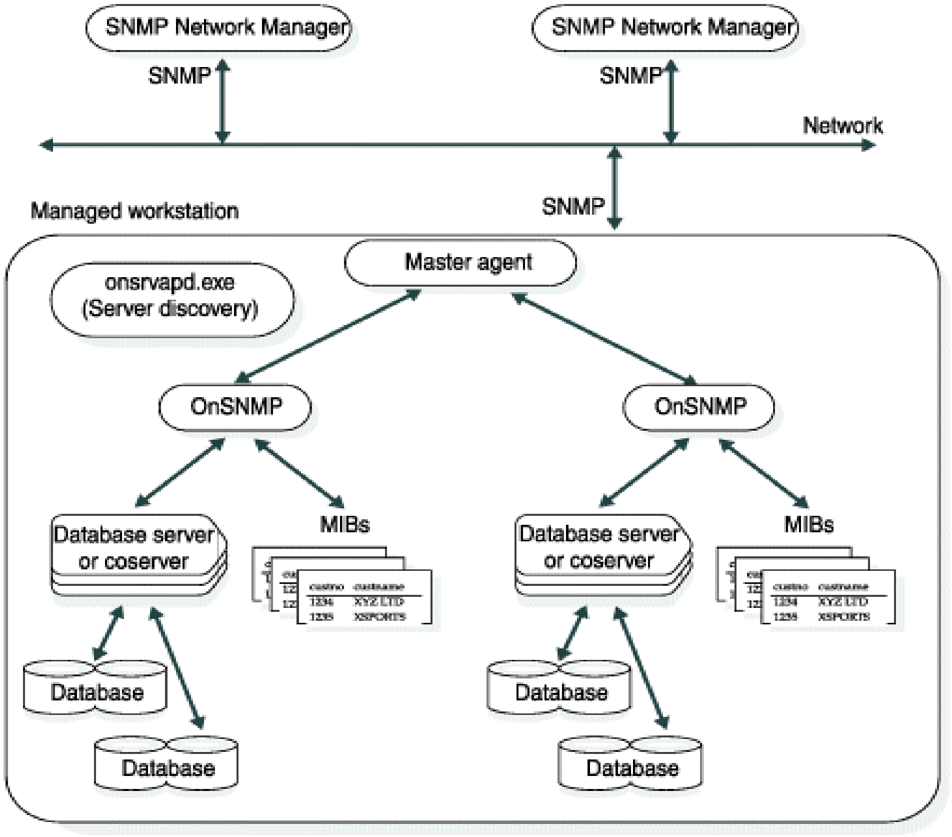
The architecture for the HCL® Informix® implementation of SNMP depends on your operating system.

SNMP is incompatible on High-Availability Data Replication (HDR) secondary servers, remote standalone (RS) secondary servers, or shared disk (SD) secondary servers.

Informix® SNMP architecture on UNIX™

The following figure shows the SNMP architecture for Informix® database servers on UNIX™. Each managed workstation runs one master agent and one server discovery process. Each database server has one OnSNMP process.

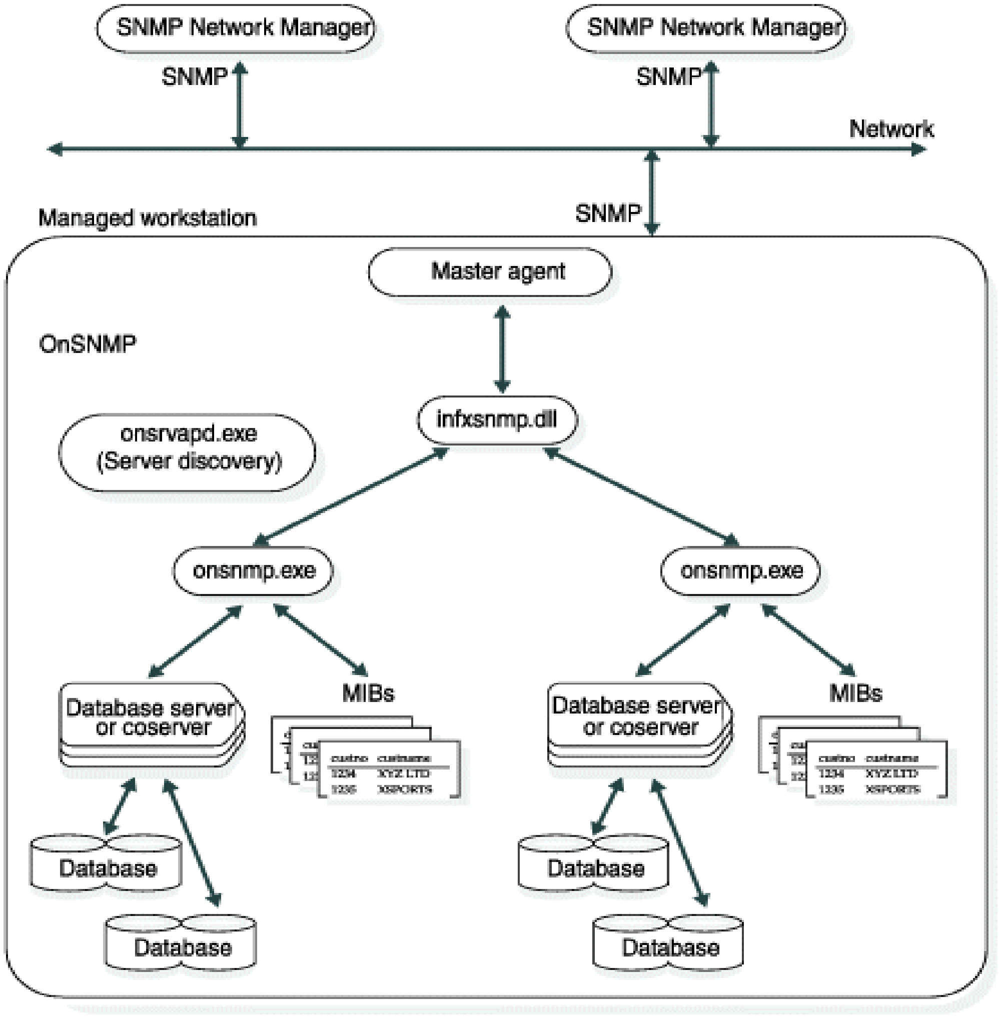
Figure 6. Informix® SNMP architecture on UNIX™



Informix® SNMP architecture on Windows™

The following figure shows the SNMP architecture for Informix® database servers on Windows™. Each managed workstation runs one master agent. The master agent and the SNMP Network Manager use SNMP to communicate with each other. Each managed workstation runs one server discovery process and one `infxsnmp.dll`. One instance of the **onsnmp** subagent is started for each instance of Informix® that runs on the managed workstation. OnSNMP and the master agent do not need to use SNMP to communicate with each other.

Figure 7. Informix® SNMP architecture on Windows™



Informix® implementation of SNMP on UNIX™ or Linux™

To use the HCL Informix® implementation of SNMP on UNIX™ or Linux™, you must install and start the following software:

- `runsnmp.ksh`
- An SNMP Network Manager on a network management workstation
- A master agent on each workstation that includes the HCL Informix® database server
- the HCL Informix® database server

When you install the Informix® database server, the installation procedure installs the OnSNMP subagent and the server discovery process as well as the files needed for SNMP support.

The discovery process discovers multiple server instances running on the host. These instances might belong to different versions that are installed on different directories. Whenever a server instance is brought online, the discovery process detects it and creates an instance of OnSNMP to monitor the database server.

The runsnmp.ksh script

The `runsnmp.ksh` script on UNIX™ ensures that both the SNMP master agent and the **onsrvapd** server-discovery daemon are running on a host.

The `runsnmp.ksh` file is in the `$INFORMIXDIR/snmp` directory. You must correctly set the **INFORMIXDIR** environment variable to the latest installed version of the product and run the script as **root**.

runsnmp.ksh [{ **-m***master_agent_args* | **-s***server_disc_args* }] [*stop*] [*start*]

Issue the `runsnmp.ksh` commands that the following diagram shows.

Table 3. The runsnmp.ksh commands

Option	Description
-m *master_agent_args*	The master-agent arguments can be either **stop** or valid master-agent arguments.
-s *server_disc_args*	The server-discovery arguments can be either **stop** or valid **onsrvapd** arguments.
start	Starts **snmpdm** and **onsrvapd** if they are not running. This option is the default.
stop	Stops **snmpdm** and **onsrvapd** if they are already running and exits.

The *master_agent_args* and the *server_disc_args* are not checked for correctness.

The following examples illustrate how to use `runsnmp.ksh`:

- Start **snmpdm** and **onsrvapd** if they are not running.

  ```
  runsnmp.ksh
  ```

- Stop **onsrvapd** and **subagents** and then exit.

  ```
  runsnmp.ksh -s stop
  ```

- Stop **onsrvapd** and any **subagents** and then restart **onsrvapd**.

  ```
  runsnmp.ksh -s stop start
  ```

- Stop **snmpdm**, **onsrvapd**, and any **subagents** and then exit.

  ```
  runsnmp.ksh stop
  ```

- Stop **snmpdm** or **snmpdp**, **onsrvapd**, and any **subagents** and then restart **snmpdm** or **snmpdp** and **onsrvapd**.

```
runsnmp.ksh stop start
```

- Start **snmpdm** if it is not running, and then start **onsrvapd** with the `none` option, if it is not running.

```
runsnmp.ksh -s "
-rnone"
```

UNIX™ master agents

On UNIX™, master agents are provided through licensing agreements.

The following table lists these master agents.

Master Agent	Company	Website
EMANATE, Version 14.2	SNMP Research	www.snmp.com

For some UNIX™ platforms, you might be able to use a master agent other than the one provided with the database server. To see whether this applies to your platform, see your release notes.

Assuring compatibility

The following guidelines assure master agent compatibility:

- Only one master agent is provided, usually EMANATE, for each UNIX™ platform type.
- The subagent that works with the master agent is also provided with the database server.
- In some cases, the platform vendor also supplies a master agent that works with the subagent provided with the database server. This is generally true only if the platform vendor supplies the same type of master agent as that provided with the database server and if the version number of the vendor-supplied master agent is greater than or equal to that of the version provided with the database server.
- Only run one instance of a master agent on a platform. You can run multiple instances of subagents, including multiple instances of **onsnmp**, if multiple database server instances exist.
- HCL Informix® subagents can coexist with subagents that platform or third-party vendors supply if all the subagents share a common, compatible master agent.

Starting and stopping a master agent

Start the master agent before you start the HCL Informix® database server, and stop all HCL Informix® database servers on a workstation before you stop the master agent.

About this task

The best way to start a master agent is to run the `runsnmp.ksh` script as part of the startup procedure for the system. Similarly, the best way to stop a master agent is to run the `runsnmp.ksh` script as part of the shutdown procedure. However, you can start or stop a master agent manually if you prefer. Additionally, while a master agent is running, you can make sure that it is running correctly.

The `runsnmp.ksh` script automatically starts the EMANATE master agent at startup and stops it at shutdown.

If you bought a master agent from another vendor, follow the instructions that the vendor provides.

Stopping a master agent manually

About this task

To stop a master agent manually:

1. Log in as **root**.

   If you do not have **root** user privileges, ask your system administrator to stop the master agent.
2. Kill the following process:

   For EMANATE, **snmpdm**

Results

The following table describes the command-line options that you can include in the `snmpdm` command for the EMANATE master agent.

Table 4. The snmpdm command-line options

Option	Description
-apall	Turn on all messages.
-aperror	Turn on error messages. Error messages are already turned on by default.
-aptrace	Turn on trace messages.
-apwarn	Turn on warning messages. Warning messages are already turned on by default.
-d	Run the master agent in the foreground.

Making sure that a master agent is running correctly

About this task

To make sure that a master agent is running correctly:

1. Check the master agent log file to verify that the master agent has not generated any errors.
   The log file is located in the /tmp directory unless the environment variable mentioned in #unique_27 is set to a different directory.
2. Verify that the process is running:

   For EMANATE, **snmpdm**

UNIX™ subagent

When you install the HCL Informix® database server on UNIX™, the installation procedure installs OnSNMP. OnSNMP consists of the **onsnmp** program.

Under normal circumstances, you do not need to start or stop OnSNMP explicitly. If you experience abnormal circumstances and need to start or stop OnSNMP explicitly, contact Technical Support.

The following additional files are provided with the database server for SNMP support.

Table 5. Additional files provided with the database server

Program	Description
onsrvapd daemon	When you start the Informix® database server that is on this workstation, **onsrvapd** detects this event and starts OnSNMP for the database server. When the database server halts, **onsrvapd** stops OnSNMP for that database server.
`runsnmp.ksh` script	This script starts **onsrvapd**. It also starts the master agent that is appropriate for the platform. If you want to run OnSNMP, you need to run `runsnmp.ksh` each time that you reboot.

UNIX™ server discovery process

The `runsnmp.ksh` script automatically starts the UNIX™ server discovery process. This section provides procedures for working manually with **onsrvapd**. Some of these procedures include instructions on how to configure OnSNMP.

The principles for starting and stopping **onsrvapd** manually are the same as the principles for a master agent: start **onsrvapd** before you start the HCL Informix® database server, and stop all HCL Informix® database servers on a workstation before you stop **onsrvapd**.

Preparing onsrvapd manually

If you do not use `runsnmp.ksh` to automatically prepare and start **onsrvapd**, perform the steps in this procedure.

About this task

To prepare **onsrvapd**:

1. Make sure that the owner of **onsrvapd** is **root** and that the group is **informix**.
2. Make sure that the setuid (sticky) bit is set for the **onsrvapd** file.

Issue the onsrvapd command

You can specify the **onsrvapd** command-line options that the following syntax shows. Some of these options affect OnSNMP.

```
onsrvapd [ { -d | -glogginglevel | -k lingermnts | -lpathname | -ppollsecs | -rserver_disc_args | -slevel | -v } ]
```

Table 6. The onsrvapd command-line options

Option	Description
-d	Flag that tells UNIX™ to run **onsrvapd** once and terminate it instead of starting it as a daemon.
-g *logginglevel*	Logging level to which OnSNMP logs debug information. Valid values are 2, 4, 8, 16, 32, and 64. The default value is 32. The lower the value, the higher the amount of logging. The **onsrvapd** daemon passes this value to OnSNMP.
-k *lingerings*	Number of minutes that **onsrvapd** waits after a database server goes down before **onsrvapd** kills the corresponding OnSNMP. If *lingermnts* is 0, **onsrvapd** waits indefinitely.
-l *pathname*	Directory for the error log files. The file name of the OnSNMP error log is `onsnmp.`*`servername`*`.log`. For example, if your server name is MyServer, the file name of the OnSNMP error log is `onsnmp.MyServer.log`. The file name of the **onsrvapd** error log is `onsrvapd.log`.
-p *pollsecs*	Frequency, in seconds, with which OnSNMP polls the database server. The default value is `5` seconds. The **onsrvapd** daemon passes this value to OnSNMP.
-r *level*	Refresh control value. For a description, see Refresh control value on page 26.
-V	Prints the OnSNMP version number.

Starting onsrvapd manually

About this task

To start **onsrvapd** manually:

1. Stop or kill any daemons that are running on the workstation.
2. Enter the command: `onsrvapd`.

Results

To stop **onsrvapd** manually, kill the **onsrvapd** process.

Making sure that onsrvapd is running correctly

About this task

To make sure that **onsrvapd** is running correctly:

1. Check the log file to verify that **onsrvapd** has not generated any errors. The log file is located in the `/tmp` directory.
2. Verify that **onsrvapd** is running.

Choose an installation directory

When you have multiple HCL Informix® installation directories on a host computer, you must set the latest installation directory as **INFORMIXDIR** before you run the `runsnmp.ksh` script to start OnSNMP. If all the directories are for the same type of database server, use the installation directory that has the latest database server version number.

One way to determine the latest directory to use with different types of database server lines is to find the latest version of the SNMP master agent. The EMANATE master agent displays the version when you run it.

Informix® implementation of SNMP on Windows™

To use the HCL® Informix® implementation of SNMP on Windows™, you must install and start the following software:

- Microsoft™ SNMP service on each workstation that includes the Informix® database server
- The Informix® database server

  When you install an Informix® database server, the installation procedure installs the OnSNMP subagent and the server discovery process as well as the files needed for SNMP support.

Windows™ master agent

The Microsoft™ TCP/IP custom installation procedure installs the Microsoft™ SNMP Extendible master agent.

About this task

For information about this master agent, see the Microsoft™ TCP/IP Help.

To start the Microsoft™ TCP/IP Help:

1. Choose **Start > Help**.
2. Choose the **Index** tab.
3. Enter the following phrase in the text box: `SNMP`

   In response to this search request, the help system displays a **Topics Found** dialog box.
4. Choose **TCP/IP Procedures Help**.

Results

Important: To start or stop the Microsoft™ SNMP Extendible master agent, you must be a member of the **Administrator Group** on the host workstation.

Windows™ subagent

On Windows™, OnSNMP comprises the following files. The table also lists the directories in which the HCL® Informix® installation procedure installs each file.

Table 7. OnSNMP files and associated directories

File	Description	Directory
`infxsnmp.dll`	Library that provides the interface between `onsnmp.exe` and the master agent. The HCL® Informix® installation procedure installs one `infxsnmp.dll` on each workstation. The initialization process for the master agent loads `infxsnmp.dll`.	`%Windows%\system32`
`onsnmp.exe`	Subagent program. The HCL® Informix® installation procedure installs an `onsnmp.exe` file for each database server.	`%INFORMIXDIR%\bin`
`onsrvapd.exe`	Server discovery process, which starts `onsnmp.exe` for each database server that starts. The HCL® Informix® installation procedure performs the following tasks for `onsrvapd.exe`: • Installs one `onsrvapd.exe` on each workstation • Creates the Informix® Server Discovery Process for SNMP in the control panel and configures it to start automatically when the system reboots	32-bit platforms: `%Windows%\system32` 64-bit platforms: `Windows\SysWOW64`

When you install the Informix® database server, the installation procedure automatically installs OnSNMP. When you start the Informix® database server that is on a network that uses SNMP, `onsrvapd.exe` detects this event and starts OnSNMP for the database server. When the database server halts, `onsrvapd.exe` stops OnSNMP for that database server.

Start and stop OnSNMP

Under normal circumstances, you do not need to start or stop OnSNMP explicitly. If you are experiencing abnormal circumstances and need to start or stop OnSNMP explicitly, contact Technical Support.

Configure OnSNMP

The HCL Informix® installation procedure creates a registry key, **OnSnmpSubagent**, under `HKEY_LOCAL_MACHINE\SOFTWARE\Informix`.

The following table describes the **OnSnmpSubagent** arguments that you can change.

Table 8. OnSnmpSubagent arguments that can be changed

Argument	Value	Description
Environment\LINGER_TIME	*lingermnts*	Number of minutes that the master agent waits after a database server goes down before the master agent kills the corresponding OnSNMP. If *lingermnts* is 0, the master agent waits indefinitely.
Environment\LOGDIR	*pathname*	Complete path of the OnSNMP error-log file, including file name
Environment\REFRESH_TIME	*pollsecs*	Frequency, in seconds, with which OnSNMP polls the database server
Environment\LOGLEVEL	*loglevel*	Logging level to which OnSNMP logs debugging information. The default value is 3. The **onsrvapd** daemon passes this value to OnSNMP.

The following table describes the **OnSnmpSubagent** arguments that you not change.

Table 9. OnSnmpSubagent arguments that do not get changed

Argument	Value	Description
Pathname	*pathname*	Complete path of `infxsnmp.dll`, including file name
MIBS\APPLMIB	*apploid*	OID for the Application MIB
MIBS\ONMIB	*onoid*	OID for the Online MIB
MIBS\RDBMSMIB	*rdbmsoid*	OID for the RDBMS MIB

The HCL Informix® installation procedure also creates an argument, INFXSNMP, under `HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\SNMP\Parameters\ExtensionAgents`. This new argument specifies the location of the **OnSnmpSubagent** registry key, including the name of the key.

To change the OnSNMP configuration, change the values for these arguments.

Windows™ registry key for the OnSNMP logging level

On Windows™, there is a registry entry to specify the logging level to which OnSNMP logs debugging information.

The logging levels that you can specify are:

- 6 (unrecoverable error conditions)
- 5 (major error conditions)
- 4 (warnings in the program)
- 3 (general information)
- 2 (debug information)
- 1 (dump all information)

Windows™ server discovery process

The HCL Informix® Server Discovery Process for SNMP is known as **onsrvapd**. It is installed as a Windows™ service that runs under the Informix® user.

The discovery process discovers multiple server instances running on the host. These instances might belong to different versions that are installed on different directories. Whenever a server instance is brought online, the discovery process detects it and creates an instance of OnSNMP to monitor the database server.

Start and stop onsrvapd

You can start **onsrvapd** from the services folder in the control panel or from a command prompt.

To start and stop **onsrvapd** from a command prompt, enter the following commands:

- To start **onsrvapd**, enter:

  ```
  net start onsrvapd
  ```

- To stop **onsrvapd**, enter:

  ```
  net stop onsrvapd
  ```

The OnSNMP Discovery Process (`onsrvapd.exe`) is installed as an Windows™ service and starts and stops automatically. You do not need to issue commands at the command line. In the event you want to issue commands from the command line, see the command-line syntax listed in Issue the onsrvapd command on page 16.

Ensure that onsrvapd is running correctly, by checking the log file to verify that **onsrvapd** has not generated any errors. For location of the log files, see your release notes. Verify that **onsrvapd** is running.

Installing the Informix® SNMP agent

If you install the Microsoft™ SNMP Extendible master agent after you install the HCL Informix® database server, the Informix® installation procedure cannot create INFXSNMP. To correct this problem, run a program called **inssnmp** to complete the OnSNMP installation.

About this task

To run **inssnmp**:

1. Start a Command Prompt session.
2. Go to `%INFORMIXDIR%\bin`.
3. Enter the following command: `inssnmp`

Results

Tip: If you install a Windows™ service pack on your computer before you install the Microsoft™ SNMP Extendible master agent, you might need to reinstall the service pack.

GLS and SNMP

HCL Informix® products include a Global Language Support (GLS) feature, which lets you work with languages that use code sets other than the standard English code set. However, the SNMP protocols that OnSNMP supports (SNMPv1 and SNMPv2) do not recognize these different code sets.

OnSNMP uses the US English locale when it sends information to the master agent. If OnSNMP cannot convert the code set of the database to the US English locale, it fails and returns error -23101 with the following message:

```
Unable to load locale categories.
```

OnSNMP sends only 7-bit characters. If an eighth bit is present, OnSNMP truncates it. Thus, when an SNMP Network Manager requests character information, OnSNMP returns a value. However, the value might not reflect the name of the database or table.

OnSNMP sends numeric information correctly, regardless of the code set that the database uses.

MIB types and objects

This section describes the types of MIBs and the types of MIB objects that the HCL Informix® database server uses.

OnSNMP uses the following MIBs:

- Application MIB
- Relational Database Management System (RDBMS) MIB
- HCL Informix® Private MIB
- Online MIB in the HCL Informix® Private MIB

Application MIB

The Application MIB is a public MIB, which means that the Internet Engineering Task Force (IETF) specifies the structure of the MIB and the MIB tables. A public MIB is the same for all managed components on an SNMP network, not just for HCL® Informix® products.

OnSNMP uses only **applTable**, which is the portion of the Application MIB that the RDBMS MIB requires.Figure 4: MIB hierarchy on page 8 shows the position of the Application MIB in the MIB hierarchy.

The following value is the path to the Application MIB:

```
iso.org.dod.internet.mgmt.mib-2.application
```

The following value is the OID for the Application MIB:

```
1.3.6.1.2.1.27
```

RDBMS MIB

The Relational Database Management System (RDBMS) MIB is a public MIB, which means that the IETF specifies the structure of the MIB and the MIB tables.

A public MIB is the same for all managed database components. However, some of the definitions in the RDBMS MIB are purposely vague to let each vendor tailor the entries to a specific database server. For example, **rdbmsSrvLimitedResourceTable** contains information about the resources that a database server uses. Each database server vendor can decide which resources to include in this table. Figure 4: MIB hierarchy on page 8 shows the position of the RDBMS MIB in the MIB hierarchy.

The following value is the path to the RDBMS MIB:

```
iso.org.dod.internet.mgmt.mib-2.rdbmsMIB
```

The following value is the OID for the RDBMS MIB:

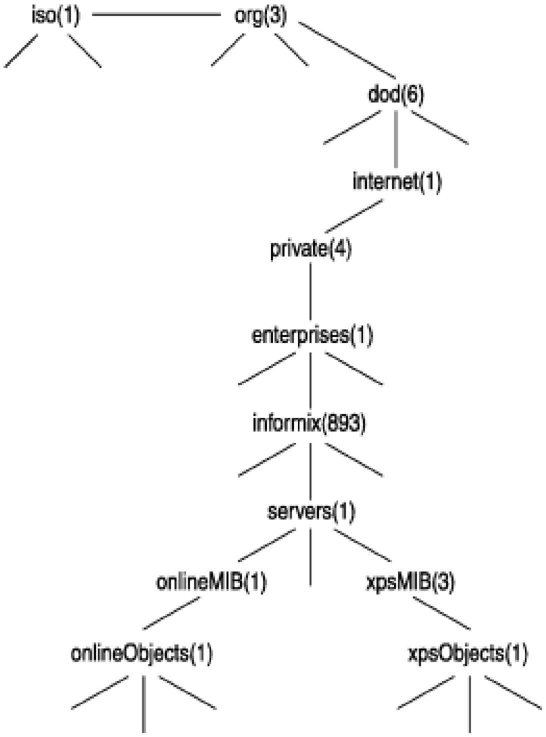
```
1.3.6.1.2.1.39
```

Informix® Private MIB

The Informix® Private MIB is a private MIB, which means that a private enterprise defines and uses it.

The Internet Assigned Numbers Authority (IANA) assigns a unique enterprise identifier to each company that uses the SNMP protocol. The Informix® Private MIB describes information that is relevant to the specific architecture and features of Informix® database servers and databases. The following figure shows the MIB hierarchy for the Informix® Private MIB.

Figure 8. MIB hierarchy for the Informix® Private MIB



The following value is the path to the Informix® Private MIB:

```
iso.org.dod.internet.private.enterprises.informix
```

The following value is the OID for the Informix® Private MIB:

```
1.3.6.1.4.1.893
```

Online MIB

The Online MIB is in the Informix® Private MIB. The Online MIB contains information for all Informix® database servers.

In the Online MIB, all tables are after the following node:

```
servers.onlineMIB.onlineObjects
```

The OID for each table in the Online MIB starts with the following value:

```
1.3.6.1.4.1.893.1.1.1
```

MIB objects

An MIB object is similar to a column in a table.

The Informix® implementation of SNMP recognizes the following types of MIB objects:

- Traps are defined as MIB objects, but they cannot be retrieved. Instead, when a certain condition is detected, OnSNMP issues an event that includes the object ID that the trap defines.
- Catalog-based MIB objects exist only if the refresh control value (described in Refresh control value on page 26) is `once` or `all`.
- objects are tables that exist only if a database server is configured to participate in .

Table indexing

In the description of the MIBs in Management Information Base reference on page 28, the header for each table specifies how each row in the table is indexed. A table can have one or more indexes. For example, the header for **rdbmsSrvTable** is **rdbmsSrvTable[applIndex]**, which means that the table has one index called **applIndex**.

Each index value is concatenated to the column OID with periods between each value. If a MIB table has several indexes, the indexes are concatenated one after the other. Most SNMP Network Managers display only the final portion of the OID that relates to the table being displayed. Some SNMP Network Managers display the OID as part of the information about each individual item; other SNMP Network Managers display the OID as part of a header for a list of values.

Numeric index values

The following line is an example of indexed information:

```
rdbmsRelActiveTime.72000003.893072000 = 11/16/98 12:34:08
```

The following table describes how to interpret the example. For more information about these values, see rdbmsRelTable on page 33.

Table 10. Values to interpret the example

Index subvalue	Description
rdbmsRelActiveTime	Name of the column
72000003	**rdbmsDbIndex**
893072000	**applIndex**

Alphabetical index values

When an index is an alphabetic string, such as the name of a configuration parameter, the OID for that index consists of the following elements, all separated by periods:

- Number of letters in the name
- ASCII value for each letter

The following line is an example of alphabetical indexed information:

```
rdbmsSrvParamCurrValue.893072000.4.76.82.85.83.1 = 8
```

The following table describes how to interpret this example. For more information about these values, see rdbmsSrvParamTable on page 36.

Table 11. Values to interpret the example

Index subvalue	Description
rdbmsSrvParamCurrValue	Name of the column
893072000	**applIndex**
4.76.82.85.83	**rdbmsSrvParamName**: • 4 = Number of letters • 76 = L • 82 = R • 85 = U • 83 = S
1	**rdbmsSrvParamSubIndex**

Refresh control value

As a background task, OnSNMP periodically updates the contents of MIB tables that it derives from catalog information. The refresh control value determines the amount of time that OnSNMP spends refreshing these MIB tables versus the amount of time that it spends responding to queries from the master agent.

Specify the refresh control value with the `runsnmp.ksh` -s -r command-line option or the **onsrvapd** -r command-line option. The following table lists the MIB tables that this value affects.

Table 12. MIB tables affected by options

Database-related MIB tables	Table-related MIB tables
rdbmsDbInfoTable **rdbmsDbTable** **rdbmsRelTable** **onBarTable** **onDatabaseTable**	**onActiveTableTable** **onFragmentTable** **onTableTable**

The following table describes the possible values for the refresh control value.

Table 13. Possible values for refresh control value

Value	Description
`a` or all	Refresh the database-related and table-related tables periodically.
`n` or none	Do not fill or refresh any of the catalog-based tables. Instead, leave the catalog-based tables empty.
`o` or once	Fill the database-related and table-related tables once at startup.

The following table lists the default refresh control value for each operating system.

Table 14. Default refresh control values

Operating system	Default refresh control value
UNIX™	once
Windows™	all

The best value to use depends on the environment and how you use OnSNMP. If the list of tables and databases changes frequently, it is probably best to use a value of `all` to make sure that the MIB tables are accurate. If the environment includes many tables and databases, it is probably best to use a value of `once` to let OnSNMP respond to queries.

Files installed for SNMP

This section lists the files that are typically installed for the HCL® Informix® implementation of SNMP on UNIX™ and Windows™.

Files installed on UNIX™ or Linux™

The `runsnmp.ksh` file exists for all UNIX™ versions of SNMP support.

The following files are installed in `$INFORMIXDIR/bin`.

Table 15. Files installed in $INFORMIXDIR/bin

File name	Description
`onsnmp`	OnSNMP executable file
`onsrvapd`	Server discovery process
`snmpdm`	EMANATE executable or a dummy file for UNIX™ platforms that EMANATE does not support

The following files are installed in `$INFORMIXDIR/snmp`.

Table 16. Files installed in $INFORMIXDIR/snmp

File name	Description
`./snmpr/snmpd.cnf`	EMANATE configuration file or a dummy file for UNIX™ platforms that EMANATE does not support
`.runsnmp.ksh`	Script that starts the master agent and **onsrvapd**

OnSNMP uses the following log files by default.

Table 17. Default log files

File name	Description
`snmp.log`	Log file for EMANATE; not installed on UNIX™ platforms that EMANATE does not support
`onsrvapd.log`	Log file for **onsrvapd**.
`onsnmp.*.log` For HCL Informix®, the path is **onsnmp.servername.log**	Log file for **onsnmp**.

Files installed on Windows™

The following files are created in `%Windows%\system32`.

Table 18. Files created in %Windows% \system32

File name	Description
`infxsnmp.dll`	DLL for OnSNMP
`onsrvapd.exe`	Server discovery process

The following file is created in `%INFORMIXDIR%\bin`.

Table 19. Files created in %INFORMIXDIR%\bin

File name	Description
`onsnmp.exe`	OnSNMP executable

In addition, log files are created in the directories that are specified in the registry.

Management Information Base reference

An SNMP Network Manager hides most of the structures of the Management Information Base (MIB). However, an understanding of this structure can help you comprehend the information that an SNMP Network Manager displays.

The descriptions in this section are brief. For detailed descriptions, see the online MIB files. The following table lists the directories for the MIB files.

Table 20. Directories for MIB files

Operating system	MIB directory
UNIX™	`$INFORMIXDIR/snmp`
Windows™	`%INFORMIXDIR%\etc`

Many MIB values are for database servers, depending on the types of database servers that you are using.

This section presents the MIB tables in alphabetical order. For the logical order, see the MIB files. The following table summarizes the MIB tables that OnSNMP uses and indicates the topics that contains more information.

Table 21. MIB tables that OnSNMP uses

MIB	Table	Description
Application	**applTable**	Attributes for each database server

Table 21. MIB tables that OnSNMP uses (continued)

MIB	Table	Description
RDBMS	**rdbmsDbInfoTable**	Information about databases
	rdbmsDbTable	Information about databases
	rdbmsRelTable	Information about the relationship between a database and the database server with which it is associated
	rdbmsSrvInfoTable	Information about the database server since it was started
	rdbmsSrvLimited-ResourceTable	Information about the limited resources for each database server
	rdbmsSrvParamTable	Information about the configuration parameters for each database server
	rdbmsSrvTable	Information about a database server
	rdbmsTraps	Information about the traps that OnSNMP can send to the SNMP Network Manager
Online	**onActiveBarTable**	Information about the current activity
	onActiveTableTable	Information about the open and active database tables
	onBarTable	Information about the backup and restore history
	onChunkTable	Information about the chunks that the database servers use
	onDatabaseTable	Information about active databases
	onDbspaceTable	Information about dbspaces
	onErQueueTable	Information about the queue
	onErSiteTable	Information about the site
	onFragmentTable	Information about the fragments that are in fragmented database tables
	onLockTable	Information about the active locks that database servers are using
	onLogicalLogTable	Information about logical logs
	onPhysicalLogTable	Information about physical logs
	onServerTable	Status and profile information about each active database server
	onSessionTable	Information about each session
	onSqlHostTable	Copy of the connection information

Table 21. MIB tables that OnSNMP uses (continued)

MIB	Table	Description
	onTableTable	Information about a database table

Application MIB

HCL Informix® uses one table from the application MIB. This table provides general-purpose attributes for each database server.

applTable

The following list summarizes this table:

Contents:

Attributes for each database server

Index:

applIndex

Scope of a row:

One database server

The table has the following MIB objects.

Table 22. MIB objects for applTable

MIB object	Description
applIndex	Unique integer index that identifies each database server. This value is the sum of the following values: • HCL Informix® Enterprise ID * 1,000,000 The HCL Informix® Enterprise ID is 893. Therefore, Enterprise ID * 1,000,000 is 893,000,000. • SERVERNUM * 1000
applName	Name of the database server.
applDirectoryName	No OnSNMP support for this MIB object.
applVersion	Version of the database server.
applUptime	Time when the database server was last initialized. This time is the system time according to the master agent. If the database server was last initialized before OnSNMP was last initialized, this value is 0.

Table 22. MIB objects for applTable (continued)

MIB object	Description
applOperStatus	Operating status of the database server: • up (1) • down (2) • halted (3) • (4): OnSNMP does not use this value. • restarting (5)
applLastChange	Time when the database server entered its current state. This time is the system time according to the master agent. If the database server was last initialized before OnSNMP was last initialized, this value is 0.
applInboundAssociations	Number of current SQLCONNECT actions.
applOutboundAssociations	OnSNMP does not support this MIB object.
applAccumulatedInboundAssociations	Number of SQLCONNECT actions that have occurred so far.
applAccumulatedOutboundAssociations	OnSNMP does not support this MIB object.
applLastInboundActivity	Time for the most recent attempt to start or stop a session with a database server. This time is the system time according to the master agent.
applLastOutboundActivity	OnSNMP does not support this MIB object.
applRejectedInboundAssociations	Number of times that the database server rejected an input connection due to administrative reasons or resource limitations.
applFailedOutboundAssociations	OnSNMP does not support this MIB object.

RDBMS MIB

The Relational Database Management System (RDBMS) MIB defines several tables that provide information about managed database servers and their databases.

OnSNMP does not support the tables **rdbmsDbLimitedResourceTable** and **rdbmsDbParamTable**.

rdbmsDbInfoTable

The following list summarizes this table:

Contents:

Information about databases

Index:

rdbmsDbIndex

Scope of a row:

One database that does not have an access state of **unavailable**

The **rdbmsRelState** value indicates the access state for the database.

The table has the following MIB objects.

Table 23. MIB objects for rdbmsDbInfoTable

MIB object	Description
rdbmsDbIndex	See rdbmsDbTable on page 32.
rdbmsDbInfoProductName	Name of the database product. For example, this value might be HCL Informix®.
rdbmsDbInfoVersion	Version number of the database server that created or last restructured this database
rdbmsDbInfoSizeUnits	Units for **rdbmsDbInfoSizeAllocated** and **rdbmsDbInfoSizeUsed**: • Bytes (1) • Kilobytes (2) • Megabytes (3) • Gigabytes (4) • Terabytes (5)
rdbmsDbInfoSizeAllocated	Estimated size allocated for this database in the units that **rdbmsDbInfoSizeUnits** specifies
rdbmsDbInfoSizeUsed	Estimated size in use for this database in the units that **rdbmsDbInfoSizeUnits** specifies
rdbmsDbInfoLastBackup	Date and time when the latest backup of the database was performed. If the database has never been backed up, this value is noSuchInstance (SNMPv2) or noSuchName (SNMPv1).

rdbmsDbTable

The following list summarizes this table:

Contents:

Information about databases

Index:

rdbmsDbIndex

Scope of a row:

One database

The table has the following MIB objects.

Table 24. MIB objects for rdbmsDbTable

MIB object	Description
rdbmsDbIndex	Unique integer index that identifies a database. This value is the sum of the following values: • SERVERNUM * 1,000,000 If SERVERNUM is 0, OnSNMP uses 256 instead of 0. • Database number
rdbmsDbPrivateMibOID	OID for the HCL Informix® Private MIB: 1.3.6.1.4.1.893
rdbmsDbVendorName	Name of the database vendor: IBM® Corporation
rdbmsDbName	Name of the database
rdbmsDbContact	Login name of the person who created the database

rdbmsRelTable

The following list summarizes this table:

Contents:

Information about the relationship between a database and the database server with which it is associated

The table has the following MIB objects.

Table 25. MIB objects for rdbmsRelTable

MIB object	Description
rdbmsDbIndex	See rdbmsDbTable on page 32.
applIndex	See applTable on page 30.

Table 25. MIB objects for rdbmsRelTable (continued)

MIB object	Description
rdbmsRelState	Access state between the database server and the database: • Other (1): The database server is online, but one of the dbspaces of the database is down. • Active (2): The database server is actively using the database. The database server is online, and a user opened the database. • Available (3): The database server could use the database if asked to do so. The database server is online, but the database is not open. • Restricted (4): The database is not available. The database server is online, and a user opened the database in exclusive mode. • Unavailable (5)
rdbmsRelActiveTime	Date and time that the database server made the database active. If **rdbmsRelState** is not active, this value is noSuchInstance (SNMPv2) or noSuchName (SNMPv1).

rdbmsSrvInfoTable

The following list summarizes this table:

Contents:

Information about the database server since it was started

Index:

applIndex

Scope of a row:

One database server

The table has the following MIB objects.

Table 26. MIB objects for rdbmsSrvInfoTable

MIB Object	Description
applIndex	See applTable on page 30.
rdbmsSrvInfoStartupTime	Date and time when the database server was last started
rdbmsSrvInfoFinishedTransactions	Number of transactions completed, either with a commit or with an abort
rdbmsSrvInfoDiskReads	Number of reads from the physical disk
rdbmsSrvInfoLogicalReads	Number of logical reads
rdbmsSrvInfoDiskWrites	Number of writes to the physical disk

Table 26. MIB objects for rdbmsSrvInfoTable (continued)

MIB Object	Description
rdbmsSrvInfoLogicalWrites	Number of logical writes
rdbmsSrvInfoPageReads	Number of page reads
rdbmsSrvInfoPageWrites	Number of page writes
rdbmsSrvInfoDiskOutOfSpaces	Number of times that the database server has been unable to obtain the desired disk space
rdbmsSrvInfoHandledRequests	Number of requests made to the database server on inbound associations
rdbmsSrvInfoRequestRecvs	Number of receive operations that the database server made while it was processing requests on inbound associations
rdbmsSrvInfoRequestSends	Number of send operations that the database server made while it was processing requests on inbound associations
rdbmsSrvInfoHighwaterInbound-Associations	Greatest number of inbound associations that have been open at the same time
rdbmsSrvInfoMaxInbound-Associations	Greatest number of inbound associations that can be open at the same time

rdbmsSrvLimitedResourceTable

The following list summarizes this table:

Contents:

Information about the limited resources for each database server

Index:

applIndex, **rdbmsSrvLimitedResourceName**

Scope of a row:

One limited resource

The table has the following MIB objects.

Table 27. MIB objects for rdbmsSrvLimitedResourceTable

MIB Object	Description
applIndex	See applTable on page 30.
rdbmsSrvLimitedResourceName	Name of the limited resource:

Table 27. MIB objects for rdbmsSrvLimitedResourceTable (continued)

MIB Object	Description
	• BUFFERS • DS_MAX_QUERIES • DS_MAX_SCANS • DS_TOTAL_MEMORY • LOCKS • LTXEHWM • LTXHWM • STACKSIZE • LOGFILES • DBSPACES • CHUNKS
rdbmsSrvLimitedResourceID	OID or vendor name for the HCL Informix® Private MIB: 1.3.6.1.4.1.893 or informix
rdbmsSrvLimitedResourceLimit	Maximum value that this limited resource can attain
rdbmsSrvLimitedResourceCurrent	The current value for this limited resource
rdbmsSrvLimitedResourceHighwater	Maximum value that this limited resource has attained since **applUptime** was reset. This value is 0 for DBSPACES and CHUNKS.
rdbmsSrvLimitedResourceFailures	Number of times that the database server tried to exceed the maximum value for this limited resource since **applUptime** was reset. This value is 0 for DBSPACES and CHUNKS.
rdbmsSrvLimitedResourceDescription	Description of the limited resource. This description includes the units for the value for the limited resource.

rdbmsSrvParamTable

The following list summarizes this table:

Contents:

Information about the configuration parameters for each database server

Index:

applIndex, **rdbmsSrvParamName**, **rdbmsSrvParamSubIndex**

Scope of a row:

One configuration parameter that is listed in the configuration file for the database server

The **ONCONFIG** environment variable specifies the file name of the configuration file. The following table lists the location of the configuration file for each operating system. For more information about the configuration file, see your *HCL® Informix® Administrator's Guide* and the *HCL® Informix® Administrator's Reference*. For more information about the **ONCONFIG** environment variable, see the *HCL® Informix® Guide to SQL: Reference*.

Table 28. Location of the configure files

Operating system	Location of configuration file
UNIX™	`$INFORMIXDIR/etc/$ONCONFIG`
Windows™	`%INFORMIXDIR%\etc\%ONCONFIG%`

The table has the following MIB objects.

Table 29. MIB objects for rdbmsSrvParamTable

MIB object	Description
applIndex	See applTable on page 30.
rdbmsSrvParamName	Name of a configuration parameter
rdbmsSrvParamSubindex	Subindex for the configuration parameter. This value is 1 for every configuration parameter except DATASKIP, DBSPACETEMP, DBSERVERALIASES, and NETTYPE.
rdbmsSrvParamID	OID or vendor name for the HCL Informix® Private MIB: 1.3.6.1.4.1.893 or informix
rdbmsSrvParamCurrValue	Value of the configuration parameter. OnSNMP obtains this value from the configuration file. Therefore, it does not reflect dynamic changes that you might make to the configuration parameter.
rdbmsSrvParamComment	Purpose of the configuration parameter

rdbmsSrvTable

The following list summarizes this table:

Contents:

Information about a database server

Index:

applIndex

Scope of a row:

One database server

The table has the following MIB objects.

Table 30. MIB objects for rdbmsSrvTable

MIB object	Description
applIndex	See applTable on page 30.
rdbmsSrvPrivateMibOID	OID for the HCL Informix® Private MIB: 1.3.6.1.4.1.893
rdbmsSrvVendorName	Name of the database server vendor: IBM® Corporation
rdbmsSrvProductName	Name of the database server product. For example, this value might be HCL Informix®.
rdbmsSrvContact	Name of the database server contact: informix

rdbmsTraps

This MIB object contains information about traps that an SNMP subsystem that supports the RDBMS MIB can generate. In this case, the SNMP subsystem is OnSNMP.

frdbmsStateChange trap

When a database server changes from its status to any less-available status, OnSNMP sends a **rdbmsStateChange** trap message to configured network hosts through the master agent.

The following list summarizes this trap:

Contents:

The **rdbmsRelState** MIB object

Index:

rdbmsDbIndex, **applIndex**

Scope of a row:

If the status of the HCL Informix® database server becomes unavailable, it generates one trap for each database.

Online MIB in the Informix® Private MIB

The Online MIB defines several tables that provide information that is relevant for HCL Informix® database servers and their databases.

onActiveBarTable

The following list summarizes this table:

Contents:

Information about the current activity

Index:

applIndex, **onActiveBarIndex**

Scope of a row:

One activity

The table has the following MIB objects.

Table 31. MIB objects for onActiveBarTable

MIB Object	Description
applIndex	See applTable on page 30.
onActiveBarIndex	A number that OnSNMP assigns
onActiveBarActivityType	Type of activity: • dbspaceBackup (1) • dbspaceRestore (2) • logBackup (3) • logRestore (4) • systemBackup (5) • systemRestore (6)
onActiveBarActivityLevel	Level of activity: • completeBackup (1) • incrementalLevelOne (2) • incrementalLevelTwo (3)
onActiveBarElapsedTime	Length of time since the activity started, in hundredths of seconds
onActiveBarActivitySize	Total number of used pages to scan OnSNMP updates this value as the activity progresses.
onActiveBarActivityScanned	Number of used pages that the activity has scanned so far
onActiveBarActivityCompleted	Number of scanned pages that the activity has transferred for archiving so far
onActiveBarActivityStatus	Status of the activity

onActiveTableTable

The following list summarizes this table:

Contents:

Information about the open and active database tables

Index:

applIndex, **rdbmsDbIndex**, **onTableIndex**

Scope of a row:

One open and active database table

For a fragmented database table, the values in this table are summaries of the values from all the fragments of the database table. The table has the following MIB objects.

Table 32. MIB objects for onActiveTableTable

MIB Object	Description
applIndex	See applTable on page 30.
rdbmsDbIndex	See rdbmsDbTable on page 32.
onTableIndex	See onDbspaceTable on page 45.
onActiveTableStatus	Status of the table: • not Busy (1): The table is not in use. • busy (2): The table is in use. • dirty (3): The table has been modified.
onActiveTableIsBeingAltered	State of the table: • Yes (1): The table is being altered. (An index is being added or dropped, an ALTER TABLE statement is being executed, the alter page count is being updated, or pages are being altered to conform to the latest schema.) • No (2): The table is not being altered.
onActiveTableUsers	Number of users accessing the table
onActiveTableLockRequests	Number of lock requests
onActiveTableLockWaits	Number of lock waits
onActiveTableLockTimeouts	Number of lock timeouts
onActiveTableIsamReads	Number of reads from the database table
onActiveTableIsamWrites	Number of writes to the database table
onActiveTableBufferReads	Number of buffer reads

Table 32. MIB objects for onActiveTableTable (continued)

MIB Object	Description
onActiveTableBufferWrites	Number of buffer writes

onBarTable

The following list summarizes this table:

Contents:

Information about the backup and restore history

Index:

applIndex, **onBarActivityIndex**, **onBarObjectIndex**

Scope of a row:

One object that participated in a backup or restore activity

For information about backup and restore, see the *HCL® Informix® Backup and Restore Guide*.

The table has the following MIB objects.

Table 33. MIB objects for onBarTable

MIB object	Description
applIndex	See applTable on page 30.
onBarActivityIndex	Index to the history
onBarObjectIndex	Index to the object
onBarName	Name of the object
onBarType	Type of object: • blobspace (1) (Only HCL Informix® provides blobspaces.) • rootDbspace (2) • criticalDbspace (3) • noncriticalDbspace (4) • logicalLog (5)
onBarLevel	Level of the backup action: • completeBackup (1) • incrementalLevelOne(2) • incrementalLevelTwo (3)

Table 33. MIB objects for onBarTable (continued)

MIB object	Description
onBarStatus	Status of the action on the object: - 0 = successful - Nonzero = error number
onBarTimeStamp	Ending time stamp for the action

onChunkTable

The following list summarizes this table:

Contents:

Information about the chunks that the database servers use

Index:

applIndex, **onDbspaceIndex**, **onChunkIndex**

Scope of a row:

One chunk

The table has the following MIB objects.

Table 34. MIB objects for onChunkTable

MIB object	Description
applIndex	See applTable on page 30.
onDbspaceIndex	See rdbmsDbInfoTable on page 32.
onChunkIndex	Unique integer index for this chunk The database server generates this value.
onChunkFileName	Path name for the chunk
onChunkFileOffset	Offset into the device, in pages
onChunkPagesAllocated	Chunk size, in pages
onChunkPagesUsed	Number of pages used
onChunkType	Type of chunk:

Table 34. MIB objects for onChunkTable (continued)

MIB object	Description
	• regularChunk (1) • blobChunk (2) • stageBlob (3)
onChunkStatus	Status of the chunk: • offline (1) • online (2) • recovering (3) • inconsistent (4) • dropped (5)
onChunkMirroring	Mirroring status of the chunk: • notMirrored (1) • mirrored (2) • newlyMirrored (3)
onChunkReads	Number of physical-read operations
onChunkPageReads	Number of page reads
onChunkWrites	Number of physical-write operations
onChunkPageWrites	Number of page writes
onChunkMirrorFileName	Path name of the mirror chunk If the chunk is not mirrored, this value is noSuchInstance (SNMPv2) or noSuchName (SNMPv1).
onChunkMirrorFileOffset	Offset of the mirror, in pages If the chunk is not mirrored, this value is noSuchInstance (SNMPv2) or noSuchName (SNMPv1).
onChunkMirrorStatus	Mirroring status: • offline (1) • online (2) • recovering (3)

Table 34. MIB objects for onChunkTable (continued)

MIB object	Description
	- inconsistent (4) - dropped (5)

If the chunk is not mirrored, this value is noSuchInstance (SNMPv2) or noSuchName (SNMPv1).

onDatabaseTable

The following list summarizes this table:

Contents:

Information about active databases

Index:

applIndex, **rdbmsDbIndex**

Scope of a row:

One active database

This table does not provide information about an active database if one of the dbspaces for the database is down. (The **rdbmsRelState** MIB object for each database in **rdbmsRelTable** indicates whether a database is active and whether one of its dbspaces is down.)

The table has the following MIB objects.

Table 35. MIB objects for onDatabaseTable

MIB object	Description
applIndex	See applTable on page 30.
rdbmsDbIndex	See rdbmsDbTable on page 32.
onDatabaseDbspace	Default dbspace
onDatabaseCreated	Creation date and time
onDatabaseLogging	Logging status: - none (1) - buffered (2)

Table 35. MIB objects for onDatabaseTable (continued)

MIB object	Description
	• unbuffered (3) • ansi (4)
onDatabaseOpenStatus	Database status: • notOpen (1) • open (2) • openExclusive (3)
onDatabaseUsers	Number of users

onDbspaceTable

The following list summarizes this table:

Contents:

Information about dbspaces

Index:

applIndex, **onDbspaceIndex**

Scope of a row:

One dbspace

The table has the following MIB objects.

Table 36. MIB objects for onDbspaceTable

MIB object	Description
applIndex	See applTable on page 30.
onDbspaceIndex	Unique integer index for this dbspace. The database server generates this value.
onDbspaceName	Name of the dbspace
onDbspaceOwner	Login name of the owner
onDbspaceCreated	Creation date
onDbspaceChunks	Number of chunks in the dbspace
onDbspaceType	Type of dbspace:

Table 36. MIB objects for onDbspaceTable (continued)

MIB object	Description
	- regularDbspace (1) - temporaryDbspace (2) - blobDbspace (3)
onDbspaceMirrorStatus	Mirroring status: - notMirrored (1) - mirrored (2) - mirrorDisabled (3) - newlyMirrored (4)
onDbspaceRecoveryStatus	Recovery status: - noRecoveryNeeded (1) - doneRecovery (2) - physicallyRecovered (3) - logicallyRecovering (4)
onDbspaceBackupStatus	Backup status: - yes (1): The dbspace is backed up. - no (2): The dbspace is not backed up.
onDbspaceMiscStatus	Miscellaneous status: - none (1): no more information - aTableDropped (2)
onDbspacePagesAllocated	Size of all the primary chunks in the dbspace
onDbspacePagesUsed	Number of pages used in all the primary chunks in the dbspace
onDbspaceBackupDate	Date when the latest backup was performed. If the dbspace has never been backed up, this value is noSuchInstance (SNMPv2) or noSuchName (SNMPv1).
onDbspaceLastBackupLevel	Level of the last backup. If the dbspace has never been backed up, this value is noSuchInstance (SNMPv2) or noSuchName (SNMPv1).

Table 36. MIB objects for onDbspaceTable (continued)

MIB object	Description
onDbspaceLastFullBackupDate	Date and time of the last full backup (level 0). If the dbspace has never had a full backup, this value is noSuchInstance (SNMPv2) or noSuchName (SNMPv1).

onErQueueTable

The following list summarizes this table:

Contents:

Information about the replication queues for all database servers that participate in

Index:

applIndex, **onErQueueReplIndex**

Scope of a row:

One replication queue

The table has the following MIB objects.

Table 37. MIB objects for onErQueueTable

MIB object	Description
applIndex	See applTable on page 30.
onErQueueReplIndex	Unique integer index that identifies a replicant
onErQueueSiteIndex	Unique integer that identifies a database server
onErQueueReplName	Display string that describes the replicant or collection of replicants
onErQueueSiteName	Name of the database server
onErQueueSize	Current® number of bytes in the send queue
onErQueueLastCommit	Date and time when last transaction was committed
onErQueueLastAck	Date and time when last data was acknowledged

onErSiteTable

The following list summarizes this table:

Contents:

Information about all the remote database servers that participate in

Index:

applIndex, **onErSiteIndex**

Scope of a row:

A single replication queue

The table has the following MIB objects.

Table 38. MIB objects for onErSiteTable

MIB object	Description
applIndex	See applTable on page 30.
onErSiteIndex	Integer that uniquely identifies a database server as defined in the group entry in **sqlhosts**
onErSiteName	Name of the replication site
onErSiteState	State of the replication activity for this site: • inactive (1) • active (2) • suspend (3) • quiescent (4) • hold (5) • delete (6) • failed (7) • unknown (8)
onErSiteConnectionState	State of the connection to this site: • idle (1) • connected (2) • disconnected (3) • timeout (4) • shutdown (5) • error (6) • unknown (7)
onErSiteConnectionChange	Date and time when the connection state last changed
onErSiteIdleTimeout	Time limit for to wait for new data to send or receive. Value is set when database server is defined. Connection is closed if time limit is exceeded.
onErSiteOutMsgs	Total number of messages transmitted from the current database server to this site
onErSiteOutBytes	Total number of bytes transmitted from the current database server to this site
onErSiteInMsgs	Total number of messages received by the current database server from this site

Table 38. MIB objects for onErSiteTable (continued)

MIB object	Description
onErSiteInBytes	Total number of bytes received by the current database server from this site
onErSiteTransactions	Total number of transactions received from this site
onErSiteCommits	Total number of transactions received and committed from this site
onErSiteAborts	Total number of transactions aborted from this site
onErSiteLastReceived	Date and time when the last transaction was processed from this site
onErSiteRowCommits	Total number of rows committed from this site
onErSiteRowAborts	Total number of rows aborted from this site
onErSiteRcvLatency	Average latency between the source commit time and target receive time; performance measure of network queueing delay
onErSiteCommitLatency	Average latency between source and target commit time; performance measure of network and database server delay
onErSiteClockErrors	Number of transactions received from this site with a time that is ahead of our current time; indicates system clock synchronization problems

onFragmentTable

The following list summarizes this table:

Contents:

Information about the fragments that are in fragmented database tables

Index:

applIndex, **rdbmsDbIndex**, **onTableIndex**, **onFragmentIndex**

Scope of a row:

One fragment of a fragmented database table

The table has the following MIB objects.

Table 39. MIB objects for onFragmentTable

MIB object	Description
applIndex	See applTable on page 30.
rdbmsDbIndex	See rdbmsDbTable on page 32.

Table 39. MIB objects for onFragmentTable (continued)

MIB object	Description
onTableIndex	See onDbspaceTable on page 45.
onFragmentIndex	Unique integer index for the fragment
onFragmentType	Type of database table: • fragmentedIndex (1) • fragmentedTable (2)
onFragmentDbspace	Dbspace name for the fragment
onFragmentExpression	Expression text used for fragmentation of the table or index This value is blank if the fragmentation scheme is round-robin.
onFragmentIndexName	Index identifier
onFragmentExtents	Number of extents used
onFragmentPagesAllocated	Total (extent) size allocated to the fragment, in pages
onFragmentPagesUsed	Number of pages used
onFragmentIsamReads	Number of reads from the fragment If the fragment is not active, this value is noSuchInstance (SNMPv2) or noSuchName (SNMPv1).
onFragmentIsamWrites	Number of writes to the fragment If the fragment is not active, this value is noSuchInstance (SNMPv2) or noSuchName (SNMPv1).
onFragmentUsers	Number of user threads that access the fragment.
onFragmentLockRequests	Number of locks of any type requested for this fragment.
onFragmentLockWaits	Number of times an initial lock request failed because the lock could not be granted initially for the fragment.
onFragmentLockTimeouts	Number of deadlock timeouts for the fragment.

onLockTable

The following list summarizes this table:

Contents:

Information about the active locks that database servers are using

Index:

applIndex, **onSessionIndex**, **onLockIndex**

Scope of a row:

One lock

A row exists for each lock that the session is using and for each lock on which the session is waiting.

The table has the following MIB objects.

Table 40. MIB objects for onLockTable

MIB object	Description
applIndex	See applTable on page 30.
onSessionIndex	See onServerTable on page 54.
onLockIndex	Index to this row
onLockDatabaseName	Name of the database that is using or waiting for this lock
onLockTableName	Name of the table that is using or waiting for this lock
onLockType	Type of the lock: • byte (1) • intentShared (2) • shared (3) • sharedByRepeatableRead (4) • update (5) • intentExclusive (6) • sharedIntentExclusive (7) • exclusive (8) • exclusiveByRepeatableRead (9) • waiting (10)
onLockGranularity	Granularity of the lock: • table (1) • page (2) • row (3) • index (4)
onLockRowId	rowid of the locked row

Table 40. MIB objects for onLockTable (continued)

MIB object	Description
onLockWaiters	Number of sessions that are waiting for the lock
onLockGrantTime	Time when the lock was granted if the session is using the lock If no transaction exists, this value is noSuchInstance (SNMPv2) or noSuchName (SNMPv1).

onLogicalLogTable

The following list summarizes this table:

Contents:

Information about logical logs

Index:

applIndex, **onLogicalLogIndex**

Scope of a row:

One logical log

The table has the following MIB objects.

Table 41. MIB objects for onLogicalLogTable

MIB Object	Description
applIndex	See applTable on page 30.
onLogicalLogIndex	Index for the logical-log file
onLogicalLogID	Unique integer identification number for the logical-log file
onLogicalLogDbspace	Dbspace name where the log file was created
onLogicalLogStatus	Status of the logical-log file: • newlyAdded (1) • free (2) • current (3) • used (4) • backedUpButNeeded (5)
onLogicalLogContainsLastCheckpoint	Checkpoint status:

Table 41. MIB objects for onLogicalLogTable (continued)

MIB Object	Description
	- yes (1): The logical-log file contains the last checkpoint. - no (2): The logical-log file does not contain the last checkpoint.
onLogicalLogIsTemporary	Temporary status: - yes (1): The logical-log file is temporary. - no (2): The logical-log file is not temporary.
onLogicalLogPagesAllocated	Size of the logical-log file, in pages
onLogicalLogPagesUsed	Number of pages used in the logical-log file
onLogicalLogFillTime	Date and time when the logical-log file last filled up If the log file has never been full, this value is noSuchInstance (SNMPv2) or noSuchName (SNMPv1).
onLogicalLogTimeUniqueIDChanged	Time stamp when a new unique ID was assigned to this logical-log entry
onLogicalLogTimeLastBackupDate	Date and time of the last backup for this logical-log entry

onPhysicalLogTable

The following list summarizes this table:

Contents:

Information about physical logs

Index:

applIndex

Scope of a row:

One physical log

The table has the following MIB objects.

Table 42. MIB objects for onPhysicalLogTable

MIB object	Description
applIndex	See applTable on page 30.
onPhysicalLogDbspace	Dbspace name where the physical log was created

Table 42. MIB objects for onPhysicalLogTable (continued)

MIB object	Description
onPhysicalLogBufferSize	Size of the physical-log buffer, in pages
onPhysicalLogBufferUsed	Number of pages of the physical-log buffer that are used
onPhysicalLogPageWrites	Number of pages written to the physical log
onPhysicalLogWrites	Number of (disk) writes to the physical log
onPhysicalLogPagesAllocated	Size of the physical log, in pages
onPhysicalLogPagesUsed	Number of pages used

onServerTable

The following list summarizes this table:

Contents:

Status and profile information about each active database server

Index:

applIndex

Scope of a row:

One database server

The table has the following MIB objects.

Table 43. MIB objects for onServerTable

MIB Object	Description
applIndex	See applTable on page 30
onServerMode	Mode of the database server: • initializing (1) • quiescent (2) • fastRecovery (3) • backingUp (4) • shuttingDown (5) • online (6) • aborting (7) • onlineReadOnly (8)

Table 43. MIB objects for onServerTable (continued)

MIB Object	Description
onServerCheckpointInProgress	Checkpoint status: • yes (1): A checkpoint is in progress. • no (2): A checkpoint is not in progress.
onServerPageSize	Size of a page, in bytes
onServerThreads	Number of active threads
onServerVPs	Number of virtual processors
onServerVirtualMemory	Total virtual memory used, in kilobytes
onServerResidentMemory	Total resident memory used, in kilobytes
onServerMessageMemory	Total message memory used, in kilobytes
onServerIsamCalls	Sum of all reads, writes, rewrites, deletes, commits, and rollbacks to and from the database table
onServerLatchWaits	Number of latch waits
onServerLockRequests	Number of lock requests
onServerLockWaits	Number of lock waits
onServerBufferWaits	Number of buffer waits
onServerCheckpointWaits	Number of checkpoint waits
onServerDeadLocks	Number of deadlocks
onServerLockTimeouts	Number of deadlock time outs
onServerLogicalLogRecords	Number of logical-log records
onServerLogicalLogPageWrites	Number of logical-log page writes
onServerLogicalLogWrites	Number of logical-log writes
onServerBufferFlushes	Number of buffer flushes
onServerForegroundWrites	Number of foreground writes
onServerLRUWrites	Number of LRU writes
onServerChunkWrites	Number of chunk writes
onServerReadAheadPages	Number of read-ahead pages This value includes data and index read-ahead pages.
onServerReadAheadPagesUsed	Number of read-ahead pages used

Table 43. MIB objects for onServerTable (continued)

MIB Object	Description
onServerSequentialScans	Number of sequential scans
onServerMemorySorts	Number of memory sorts
onServerDiskSorts	Number of disk sorts
onServerMaxSortSpace	Maximum disk space that a sort uses, in pages
onServerNetworkReads	Number of network reads
onServerNetworkWrites	Number of network writes
onServerPDQCalls	Number of parallel-processing actions performed
onServerTransactionCommits	Number of committed transactions
onServerTransactionRollbacks	Number of rolled-back transactions
onServerTimeSinceLastCheckpoint	Length of time since the last checkpoint, in hundredths of second
onServerCPUSystemTime	Amount of CPU time that the database server has used in System Mode, in hundredths of second
onServerCPUUserTime	Amount of CPU time that the database server has used in User Mode, in hundredths of second

onSessionTable

The following list summarizes this table:

Contents:

Information about each session

Index:

applIndex, **onSessionIndex**

Scope of a row:

One session

The table has the following MIB objects.

Table 44. MIB objects for onSessionTable

MIB Object	Description
applIndex	See applTable on page 30.
onSessionIndex	Unique integer index for the session
onSessionUserName	Name of the user, in the form name@host(tty)

Table 44. MIB objects for onSessionTable (continued)

MIB Object	Description
onSessionUserProgramVersion	Version of the database server
onSessionUserProcessId	Process ID for the session
onSessionUserTime	Length of time that the user has been connected to the database server, in hundredths of seconds
onSessionState	State of the session: • idle (1) • active (2) • waitingOnMutex (3) • waitingOnCondition (4) • waitingOnLock (5) • waitingOnBuffer (6) • waitingOnCheckPointing (7) • waitingOnLogicalLogWrite (8) • waitingOnTransaction (9)
onSessionDatabase	Connected database
onSessionCurrentMemory	Memory usage, in bytes
onSessionThreads	Number of active threads
onSessionCurrentStack	Average size of the stack for all threads
onSessionHighwaterStack	Maximum amount of memory that any thread has used so far
onSessionLockRequests	Number of lock requests
onSessionLocksHeld	Number of locks held
onSessionLockWaits	Number of lock waits
onSessionLockTimeouts	Number of timeouts for locks
onSessionLogRecords	Number of log records
onSessionIsamReads	Number of reads from database tables
onSessionIsamWrites	Number of writes to database tables
onSessionPageReads	Number of page reads
onSessionPageWrites	Number of page writes
onSessionLongTxs	Number of long transactions

Table 44. MIB objects for onSessionTable (continued)

MIB Object	Description
onSessionLogSpace	Logical-log space used, in bytes
onSessionHighwaterLogSpace	Maximum logical-log space that this session has ever used
onSessionSqlStatement	Latest SQL statement, truncated to 250 characters if necessary
onSessionSqlIsolation	SQL isolation level: • noTransactions (1) • dirtyReads (2) • readCommitted (3) • cursorRecordLocked (4) • repeatableRead (5)
onSessionSqlLockWaitMode	Action to take if the isolation level requires a wait: • `-1` = Wait forever. • `0` = Do not wait. • `>0` = Wait for specified number of seconds.
onSessionSqlEstimatedCost	Estimated cost of the SQL statement according to SQLEXPLAIN
onSessionSqlEstimatedRows	Estimated number of rows that the SQL statement selects according to SET EXPLAIN
onSessionSqlError	Error number for the last SQL statement
onSessionSqlIsamError	ISAM error number for the last SQL statement
onSessionTransactionStatus	Status of the transaction: • none (1) • committing (2) • rollingBack (3) • rollingHeuristically (4) • waiting (5)
onSessionTransactionBeginLog	Unique ID of the logical-log file in which the BEGIN WORK record was logged If no transaction exists, this value is noSuchInstance (SNMPv2) or noSuchName (SNMPv1).

Table 44. MIB objects for onSessionTable (continued)

MIB Object	Description
onSessionTransactionLastLog	Unique ID of the logical-log file in which the last record was logged If no transaction exists, this value is noSuchInstance (SNMPv2) or noSuchName (SNMPv1).
onSessionOriginatingSessionId	Local session ID of the global session on the server for which this local session runs

onSqlHostTable

The following list summarizes this table:

Contents:

Copy of the connection information

Index:

applIndex, **onSqlHostIndex**

Scope of a row:

One connectivity value

As the following table shows, the location of the connection information depends on the operating system.

Table 45. Location of connection information

Operating system	Location of connectivity information
UNIX™	The **INFORMIXSQLHOSTS** environment variable specifies the full path name and file name of the connection information. The default location is `$INFORMIXDIR/etc/sqlhosts`. For information about **INFORMIXSQLHOSTS**, see the *HCL® Informix® Guide to SQL: Reference*.
Windows™	The connectivity information is in a key in the Windows™ registry called `HKEY_LOCAL_MACHINE\SOFTWARE\Informix\SQLHOSTS`.

For details about the connection information, see your *HCL® Informix® Administrator's Guide*.

The table has the following MIB objects.

Table 46. MIB objects for onSqlHostTable

MIB object	Description
applIndex	See applTable on page 30.
onSqlHostIndex	Index to the entry in the connectivity information

Table 46. MIB objects for onSqlHostTable (continued)

MIB object	Description
onSqlHostName	Host name of the database server
onSqlHostNetType	Connection type
onSqlHostServerName	Name of the database server or its alias
onSqlHostServiceName	Service name
onSqlHostOptions	List server options in the form of key=value pairs

onTableTable

The following list summarizes this table:

Contents:

Information about a database table

Index:

applIndex, **rdbmsDbIndex**, **onTableIndex**

Scope of a row:

One database table

For a fragmented database table, the values in this table are summaries of the values from all the database table fragments. The table has the following MIB objects.

Table 47. MIB objects for onTableTable

MIB object	Description
applIndex	See applTable on page 30.
rdbmsDbIndex	See rdbmsDbTable on page 32.
onTableIndex	Table number This value is the same as **tabid** in the system catalog table **systables**
onTableName	Table name
onTableOwner	Table owner
onTableType	Type of table: • table (1) • view (2)

Table 47. MIB objects for onTableTable (continued)

MIB object	Description
	• privateSynonyn (3) • synonym (4)
onTableLockLevel	Locking level of the table: • page (1) • row (2)
onTableCreated	Creation date, in string format
onTableFirstDbspace	Name of the first (or only) dbspace for the table
onTableRowSize	Length of a row
onTableRows	Number of rows
onTableColumns	Number of columns
onTableIndices	Number of indexes
onTableExtents	Number of extents in use
onTablePagesAllocated	Total (extent) size allocated to the table, in pages
onTablePagesUsed	Number of pages in use
onTableFragments	Number of fragments
onTableFragmentStrategy	Fragmentation strategy: • roundRobin (1) • byExpression (2) • tableBased (3)

If the table is not fragmented, this value is noSuchInstance (SNMPv2) or noSuchName (SNMPv1).

onTableActiveFragments

Number of active fragments

If the table is not fragmented, this value is noSuchInstance (SNMPv2) or noSuchName (SNMPv1).

Index