IBM Tivoli Asset Discovery for z/OS
Version 8 Release 1

# *Administration Guide and Reference*

SC22-5474-00

Before using this information and the product it supports, read the information in "Notices" on page 233.

**May 2016**

This edition applies to version 8, release 1, modification 0 of IBM Tivoli Asset Discovery for z/OS and to all subsequent releases and modifications until otherwise indicated in new editions.

# Contents

# Chapter 1. Overview of IBM Tivoli Asset Discovery for z/OS

IBM® Tivoli® Asset Discovery for z/OS® is built on the concept of remote and central mainframe components which work together to produce reports on z/OS mainframe products and their usage. This section provides you with a high-level overview of the Tivoli Asset Discovery for z/OS core architecture.

Tivoli Asset Discovery for z/OS runs on z/Architecture® mainframes that use the z/OS operating system. Its purpose is to:

- Discover and identify products for the z/OS platform.
- Monitor software usage and trends.
- Report on the MSU capacity of each system under which the product runs.
- Provide reporting for assets and usage.

The benefits of using this software are:

- Used and unused software are identified.
- Users of software are identified.
- Obsolete versions of software are identified and the usage of these versions determined.
- Usage trends of software and libraries are identified.

In an IBM z/OS environment software is contained in load libraries or as z/OS UNIX executable files. Tivoli Asset Discovery for z/OS scans the content of these libraries and executable files to determine which software products are installed. Tivoli Asset Discovery for z/OS also monitors the loaded programs and executable files to measure software usage.

The discovered load libraries and executable files are then checked against a global database of product information. Tivoli Asset Discovery for z/OS uses this information to determine which products are installed and used on each system.

The Tivoli Asset Discovery for z/OS Usage Monitor gathers information about events for modules and executable files which are then attributed to each product.

The workflow is illustrated in Figure 1 on page 2, followed by a brief description of the components.

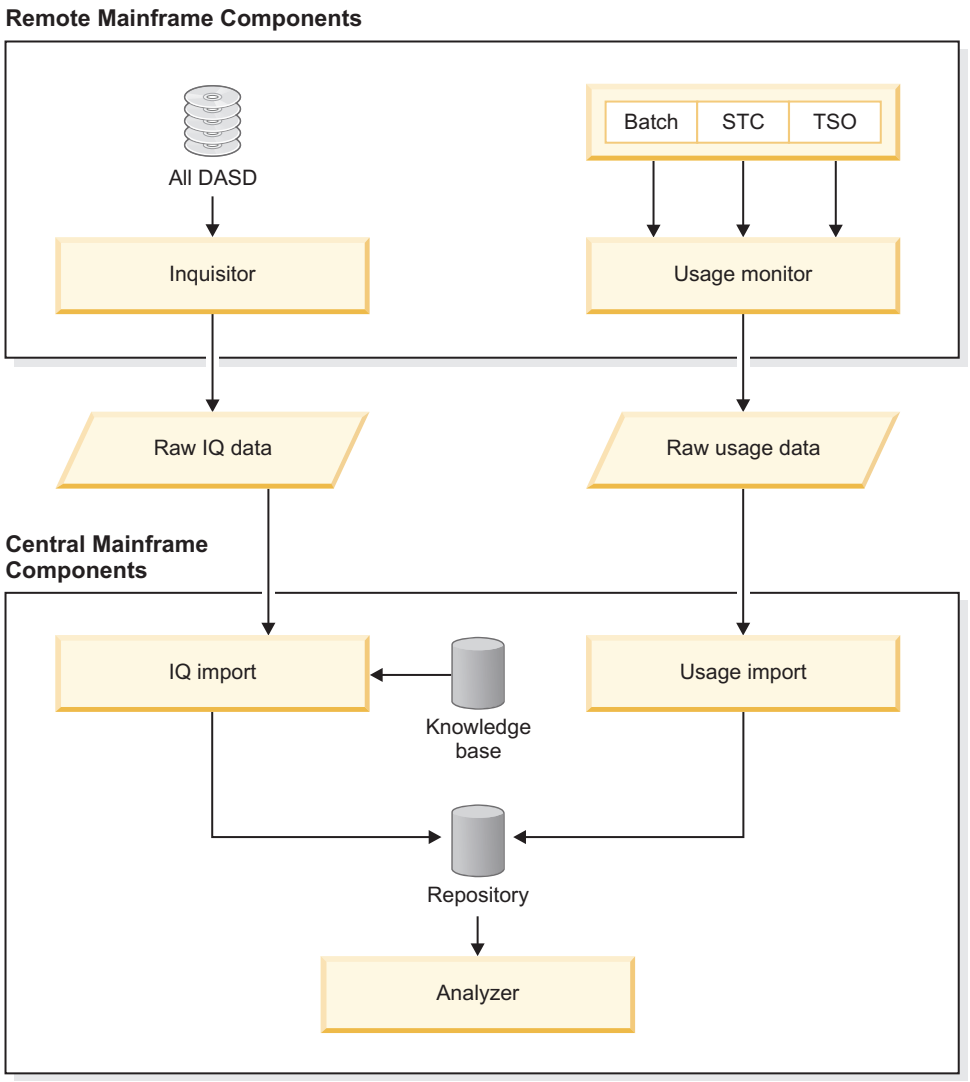



*Figure 1. Product workflow*

**Inquisitor**

The Inquisitor is a batch job that discovers loadable programs in z/OS data sets and z/OS UNIX System Services file systems. A program locates load libraries on z/OS DASD devices and captures information about the load modules. The process can be targeted to specific devices, libraries, or groups of libraries. The program creates a compressed data set, which is then used as input to the Inquisitor Import procedure.

An additional process locates and scans z/OS UNIX directories for program objects and captures this information. The process creates a compressed data set that is then used as input to the Inquisitor Import for z/OS UNIX procedure.

**Usage Monitor**

The Usage Monitor is a started task or batch job that monitors and records loaded modules of batch jobs, started tasks, TSO users, and z/OS UNIX executable files.

**Knowledge Base**

The Global Knowledge Base (GKB) is a database that is provided with

Tivoli Asset Discovery for z/OS. The GKB has a list of all z/OS globally-identified products that are used by the product in the process of matching.

**Inquisitor (IQ) Import**
The Inquisitor Import is a batch job that loads Inquisitor data into database tables on z/OS for z/OS load modules and z/OS UNIX program objects. The imported Inquisitor data is then matched against the Global Knowledge Base.

**Usage Import**
The Usage Monitor is a batch job that imports Usage Monitor data into the Repository. The data is matched against load modules and z/OS UNIX executable files and the data is then aggregated with installed software products. After this process has been completed, you can view the usage data with the Tivoli Asset Discovery for z/OS Analyzer reports.

**Repository**
The Repository is a set of database tables for z/OS data that stores information about the software products discovered and their usage data.

**Analyzer**
The Analyzer queries the Tivoli Asset Discovery for z/OS database and displays Analyzer online reports. The Analyzer runs as a started task or batch job on the same z/OS system where the DB2® subsystem or SQLite database runs. The output formats can be HTML, Excel, Text, or Comma Separated Value (CSV). You can logon to the Analyzer from a web browser to display interactive reports. You can also run the Analyzer in batch mode and save the results to an output data set on z/OS.

### Process flow

Data is collected on the target systems by the Inquisitor and the Usage Monitor batch programs. You can then import this data into the Repository database tables. The database is located on a z/OS system within a DB2 subsystem or a SQLite database.

Following is a summary of the workflow tasks:

1. Importing and matching the data collected by the Inquisitor.
2. Importing the collected usage data into the Repository.
3. Running utilities to manage and maintain your data. This task is optional.
4. Reporting using the Analyzer, which consists of online and batch components.

## Summary of changes

The updates provided in this manual are in relation to the APAR/PTFs listed below since version 8, release 1, was released.

All changes are marked with a vertical bar ( | )to the left of the change.

*Table 1.*

| APAR/PTF | Abstract |
|---|---|
| OA42950/UA70679 | ADDITIONAL CICS EXIT TO DETECT DYNAMICALLY CALLED PROGRAMS |

*Table 1. (continued)*

| APAR/PTF | Abstract |
|---|---|
| OA43016/UA71035 | HSIC020E USAGE IMPORT ENCOUNTERED ERRORS. ERROR CODE = 6800 HSI9999I AT LEAST ONE REPOSITORY TABLE FAILED IN INITIALISATION |
| OA43642/UA71608 | BUFFER OVERFLOW CAUSED BY LARGE SQL STATEMENTS. SQLCODE -803 |
| OA44118/UA72439 | HSISINQU DOES NOT SUPPORT PARAMETER PLX=Y |
| OA44416/UA73126 | PROVIDE AN OPTION TO BYPASS UPDATE OF PRODUCTS USAGE COUNTS IN COMMONLY RUN JOBS |
| OA45545/UA74298 | ABEND0C4 IN PROGRAM HSIPINQ WHEN RUNNING SCANPGM FULLIDR |
| OA46093/UA75220 | ALLOW PR/SM LPAR TIME OFFSET TIMESTAMP ADJUSTMENT IN UMON AND IQDATA |
| OA46267/UA75637 | ZCAT NOT PROCESSING JOB ID CORRECTLY |
| OA46668/UA76862 | INQUISITOR IMPORT DOES NOT ISSUE NON-ZERO RC WHEN NO DATA IS IMPORTED |
| OA48007/UA77856 | UPDATE MIGRATION JOBS FOR V7.5 TO V8.1 |
| OA48006/UA77910 | PROVIDE CAPABILITY TO COORDINATE GKB AND CODE MODIFICATIONS |
| OA48462/UA78443 | ALLOW DUPLICATE IQ IMPORTS AND THOSE THAT ARE SUPERSEDED BY USAGE TO BE PROCESSED |
| OA49152/UA81097 | USAGE MONITOR TO EXPLOIT 64-BIT STORAGE TO DELIVER VSCR |
| OA50305/UA81550 | USAGE IMPORT REPLACES SYSPLEX NAME TO ITS ORIGINAL NAME |

# Implementing Tivoli Asset Discovery for z/OS with SQLite database

You can implement Tivoli Asset Discovery for z/OS with a SQLite database if you do not have a DB2 for z/OS license or if you are currently unable to deploy one. If you implement Tivoli Asset Discovery for z/OS with a SQLite database there are certain functional and performance limitations.

SQLite is an in-process library that implements a self-contained, serverless, transactional SQL database engine. SQLite is an embedded SQL database engine that is unlike many other SQL databases because it does not have a separate server process. SQLite reads and writes data directly to ordinary disk files. When you implement SQLite with Tivoli Asset Discovery for z/OS, the complete SQL database is contained within a single z/OS UNIX zFS file.

## Limitations

Tivoli Asset Discovery for z/OS with SQLite database is implemented with the following configuration:

- 500 products identified from 15 LPARs
- 3 months usage data (about 6 million usage records)
- Only 1 repository within a zFS file

This configuration represents the limitation for SQLite support with Tivoli Asset Discovery for z/OS. Only one repository within a zFS file is supported. If you

require a database with a larger configuration supporting multiple repositories, consider implementing Tivoli Asset Discovery for z/OS with DB2 for z/OS.

SQLite has limited concurrency because it uses read/write locks on the entire database file. Therefore, if any process is reading from any part of the database, all other processes are prevented from writing to any other part of the database. Similarly, if any one process is writing to the database, all other processes are prevented from reading any other part of the database.

# What's new in IBM Tivoli Asset Discovery for z/OS version 8.1

New features and capabilities in IBM Tivoli Asset Discovery for z/OS version 8.1 help your organization achieve greater efficiency in asset discovery.

## Support for SQLite database

This release adds support for the SQLite database as an alternative to the DB2 database. This new database support provides a low-administration alternative for customers who do not have DB2 or do not have the resources to maintain a DB2 database.

## Changes to the Inquisitor

Tivoli Asset Discovery for z/OS, version 8.1 include:

- Addition of the SCANDEV command to collect information about the system I/O configuration.
- Bypassing scans of uncataloged SMS-managed data sets that eliminates many data set error conditions.
- Changing several data set error conditions to warnings to reduce their severity level.
- Addition of new record types describing specified selection filters to improve compliance auditability.
- The Inquisitor can run in multitasking mode to substantially reduce runtime.
- Improved performance for VTOC scans implemented by scanning up to 200 VTOCs concurrently.

The UNIX Inquisitor now ignores the SYMLNK keyword in the program parameter.

## Changes to the Usage Monitor

Tivoli Asset Discovery for z/OS, version 8.1 include:

- Removal of the cache in favor of a scheme to improve the performance of updating every entry.
- Ability to capture program usage data for programs that started before the current collection cycle.
- Resolution of UNIX symbolic links into real path names.
- Addition of a default data set exclusion list.
- Ability to accept UNIX path name masks as filtering criteria.
- Reporting of mask matching statistics in the regular status reports.
- Addition of new record types describing specified selection filters to improve compliance auditability.

## Changes to the Inquisitor Import

During Inquisitor import, you can now set a parameter to mark libraries, products, and load modules as deleted if existing libraries in the repository are not found in the scanned Inquisitor file. For shared libraries, the scanned Inquisitor file can be from any system ID that belongs to the sysplex. For non-shared libraries, the scanned Inquisitor file must be from the same system ID.

## Changes to Automation Server

Automation Server action statements now include a MNTH operand so that you can schedule actions to be performed, for example, on an annual or quarterly basis.

## Changes in reports

Tivoli Asset Discovery for z/OS, version 8.1 provides new reports and adds additional information to existing reports:

- New Analyzer report enables users to see what has changed over time for:
  - New product release installations
  - Product release upgrades in the same library
- Improved history logging for inquisitor and usage imports into the Tivoli Asset Discovery for z/OS repository. New reports provide information about when inquisitor scans were run and the date they were imported and for usage data that is imported. This feature enables you to see when scans are missed for a system.
- The Product Use by Machine and IBM Value Units Report is a new report that shows the value units calculation for applicable products.
- A number of reports show additional hardware information:
  - Model Capacity
  - MSU value for LPAR
  - Show IFLs
  - What processors are online at the time of the scan
  - Show machine resources
- Create product annotations for specified products and show the annotations in Inventory reports.
- The Libraries with Unknown Modules report is updated to show a best guess as to what possible product the unknown module is associated with. This enhanced reporting feature enables sites to find modules from licensed products that have been copied to private libraries for possible breach of license.
- The Storage Subsystem Hardware report is a new report that shows storage subsystems and the channels that are attached to them.
- New administration reports are available:
  - New reports to view IQ and Usage import history
  - Allow the user to create EOS dates for ISV software
  - Ability to delete old Hardware
  - Create Annotations to view in Product Inventory and Discovered Product Inventory reports

## Custom product names

In most cases, Tivoli Asset Discovery for z/OS uses commonly-used product names. For customers who support older product names, this feature enables users to enter an alternate name for a product that displays in reports in Tivoli Asset Discovery for z/OS.

## Product license verification

Tivoli Asset Discovery for z/OS, version 8.1 adds a license verification spreadsheet that allows users to verify at a high-level if they are compliant with licensed software that is installed on their systems.

## CICS Transaction Server monitoring

Tivoli Asset Discovery for z/OS, version 8.1 introduces the ability to monitor CICS transactions from specific CICS regions.

## Performance improvement and DASD savings

Tivoli Asset Discovery for z/OS, version 8.1 includes better and faster product identification algorithms and DASD savings due to removal of inquisitor tables.

# Chapter 2. Planning for deployment

Before you deploy IBM Tivoli Asset Discovery for z/OS, consider which deployment option is best suited to your environment.

**Related tasks**:

Chapter 4, “Implementing deployment scenarios,” on page 29
Most implementations of Tivoli Asset Discovery for z/OS are based on one of the common deployment scenarios. An example is provided for implementing each of these common deployment scenarios with a DB2 Repository database. You can adapt an example for use with a SQLite Repository database.

## Predeployment considerations

You can deploy Tivoli Asset Discovery for z/OS to use a single Repository or multiple Repositories.

Implementing a deployment in a single Repository is relatively straightforward because the data from all systems is imported into a single Repository. Before you deploy with a single Repository, plan the following aspects of the deployment:

- How frequently will the Inquisitor scan each system?
- If you have systems that are located at remote sites, what mechanism will transfer collected Inquisitor and usage data via file transfer protocol (FTP) to the central site, and how frequently will these transfers occur?
- How often is it necessary to load Inquisitor and usage data from each system into the Repository?

To deploy with multiple Repositories, plan the following aspects of the deployment:

- How frequently will the Inquisitor scan each system?
- How many Repositories to include in the deployment and are all of these Repositories located at the central site?
- For each system, which Repository loads the Inquisitor and usage data for the system?
- If you have systems that are located at remote sites, what mechanism will transfer collected Inquisitor and usage data via file transfer protocol (FTP) to the central site, and how frequently will these transfers occur?
- How often is it necessary to load Inquisitor and usage data from each system into its specified Repository?

## Deployment data processes

Tivoli Asset Discovery for z/OS is structured on several key data processes.

**Inquisitor data**
The Inquisitor scans DASD volumes for libraries containing load modules and HFS/zFS files for z/OS UNIX program objects and produces Inquisitor data. These load modules and program objects are matched and associated to a particular vendor and product and the matched information is then loaded into the Repository tables. These processes are performed by running the Inquisitor Import job.

**Usage event**
A usage event describes a unique load of a load module or program object for an address space that can contain an account code. The Usage Monitor records these usage events as they occur on a particular operating system. After the usage data is imported into the Repository, each usage event is identified by the load module name, library name, and volume. It can then be associated to a particular product discovered on that system.

**Repository**
The Repository is a collection of database tables that contain processed Inquisitor and Usage Monitor data. To ensure that accurate data is stored in the Repository tables, the following criteria must be met:

- The DASD VOLSERs of the data being imported must be unique unless the DASD VOLSERS are shared or are clones of each other with identical contents.
- The data imported must be from systems with unique SMF IDs.

When you are designing the scope of a Repository, there are a few common scenarios that most installations fit into. It is common to define the scope of a Repository based upon a data center. In this scenario, each data center in the organization has a separate Repository.

**CAUTION:**
**Import only DASD volumes with a unique VOLSER into your Repository.**

The only way to prevent this sharing from taking place is to divide the z/OS systems with conflicting DASD/SMF IDs into separate Repositories. This can entail running one Repository for each sysplex or stand-alone z/OS system. With Tivoli Asset Discovery for z/OS, it is common for IT service providers to define separate Repositories for each customer. This definition also satisfies the need for separation of data and ease of reporting.

It is recommended to have a central DB2 subsystem or SQLite databases that contain all the Repositories in your entire enterprise. The usage and Inquisitor data that require processing should be transmitted to this central DB2 subsystem or SQLite database by using the Tivoli Asset Discovery for z/OS Automation Server or equivalent automation product.

## Deployment for a single Repository

The recommended procedure for deploying the Inquisitor and Usage Monitor to collect raw data is to deploy both components on every system in your organization.

After you deploy both components to each system in your organization, perform data collection in the following sequence:

1. Use the Inquisitor Job to scan all available DASD on each z/OS System.
2. Import Inquisitor data by running the Inquisitor Import job.
3. Ensure that the Usage Monitor is active on all z/OS systems, directly after IPL.
4. Import Usage data by running the Usage Import job. Run this job after Inquisitor data has been imported.

Tivoli Asset Discovery for z/OS displays products that have been discovered. Usage data collected from every system by the Usage Monitor is imported and usage events are assigned to the discovered products, enabling analysis of product use by system.

The first step in deploying Tivoli Asset Discovery for z/OS is to run the SMP/E installation of the product, followed by the customization and creation of the database resources.

The next step is to create a test Repository. This deployment exercise is useful as it helps you to:

- Gain familiarity with the product.
- Check that your Repositories are defined correctly in terms of your business requirements and that the DASD VOLSERs and SMF IDs are unique.
- Ensure that data-sizing is adequate.
- Analyze the integrity of the data.

As part of this test implementation, you can then deploy the Inquisitor and Usage Monitor to all systems in your organization. It is advisable to first start the Usage Monitor on every system, in order to gather a significant amount of usage data. Place the test repository on a test or development DB2 subsystem.

At this point you can start the Tivoli Asset Discovery for z/OS Analyzer and connect to the Repository. To verify the data collected by the Inquisitor and Usage Monitor, log on to the Analyzer and navigate to the Discovery menu tab. From this menu you can proceed to various reports on discovered products and module usage.

After you move your Repositories to their final location, you should consider setting up automation of the product.

## Deployment for multiple Repositories

Multiple Repositories can be required to provide support for more than one data center, for different geographical regions, and for running multiple customers.

You can locate multiple Repositories in one central location, or you can locate them in geographically dispersed locations. Multiple Repositories may be organized as follows:

1. For a central location
2. For geographically dispersed locations

### Central location

Each Repository contains data that is divided up into logical units, for example:

- Data center
- Outsourced customer
- Sysplex

Each Repository has its own database. For DB2, all repositories must reside in the same DB2 subsystem but for SQLite, each repository must reside in its own SQLite database. For DB2 only, the advantage of this configuration is that reporting can be performed on data across all repositories. With this configuration, all repositories

can share the same Global Knowledge Base (GKB) and you only have to maintain a single copy of the GKB.

## Geographically dispersed locations

Each Repository is defined with its own database at a specific geographic site as a stand alone operation. Reporting can only be performed for each specific Repository. The disadvantage with this configuration is that it can be necessary to consolidate Repository data to a central site for reporting purposes.

# Chapter 3. Installing and customizing IBM Tivoli Asset Discovery for z/OS

The product installation involves downloading the product and available updates, preparing the database, and configuring and populating a test database. After verifying that all components are correctly installed, you duplicate the test database to create a production database where you automate data collection and import tasks.

## Installation prerequisites

Before you install IBM Tivoli Asset Discovery for z/OS, verify that the required hardware and software requirements are available in the installation environment.

### Hardware requirements

The hardware requirements for running Tivoli Asset Discovery for z/OS are a z/Architecture machine capable of running z/OS Version 1 Release 11 or later.

### Software requirements

The software requirements for running Tivoli Asset Discovery for z/OS are:

- z/OS Version 1 Release 11 or later.
- Database can be either:
  - DB2, Version 9, Release 1 or DB2 Version 10, Release 1 if you choose DB2 for your Tivoli Asset Discovery for z/OS database
  - SQLite, Version 3.2.6.23.1, that is embedded in Tivoli Asset Discovery for z/OS

  If you do not have a DB2 license, contact IBM support in order to install TADz with SQLite only. It is not necessary to install the database on all of your z/OS systems but it must be installed on at least one z/OS system
- Language Environment® for z/OS.
- Browser can be either:
  - Firefox ESR Version 10.0.10 with JavaScript and cookies enabled
  - Internet Explorer, Version 9 with JavaScript and cookies enabled
- Microsoft Excel 2003

## Security and authorization prerequisites

A z/OS user ID is required with appropriate RACF® access to submit the batch jobs used in the customizing and operation of Tivoli Asset Discovery for z/OS. Additional security and authorization configurations can be necessary, depending on your environment.

### RACF authorizations

The following table lists the RACF authority required to run Tivoli Asset Discovery for z/OS Started Tasks, Usage Monitor, Analyzer, and Automation Server. Consult with your RACF administrator to define the required RACF authority.

*Table 2. RACF authority required for each started task*

| Started task name | SHSIMOD1 | PARMLIB | SHSIANL1 | SHSIANL2 | ACDS | (DB2 only) SDSNLOAD and SDSNEXIT | HLQIDS data set | Usage Monitor output data sets |
|---|---|---|---|---|---|---|---|---|
| Usage Monitor | READ | READ | n/a | n/a | n/a | n/a | READ | ALTER |
| Analyzer | READ | READ | READ | READ | n/a | READ | n/a | n/a |
| Automation Server | READ | READ | n/a | n/a | CONTROL | n/a | n/a | n/a |

The started task should be defined in the resource class STARTED, with additional detail in the STDATA segment of the resource. It can also be defined in the started task table ICHRIN03, but this requires an IPL to add or update a task definition. For example:

```
RDEFINE STARTED HSI*.* UACC(NONE)  +
STDATA (USER(uuuuuuu))
```

Replace *uuuuuuu* with the name of the started task user for Tivoli Asset Discovery for z/OS

```
SETROPTS RACLIST(STARTED) REFRESH
```

For non-RACF security products, consult your Security Administrator.

## z/OS UNIX security

Both the Usage Monitor and the z/OS UNIX Inquisitor need sufficient authority to navigate the UNIX file system. The writer task of the Usage Monitor requires access to resolve symbolic links, while the UNIX Inquisitor is tasked with discovering executable files.

The HSIPHOST module is called by the Usage Monitor writer task and by both Inquisitor programs to collect the system TCP/IP host name and IP address. This action requires a security user profile which has an associated UNIX uid value. The call of HSIPHOST can be disabled by relevant Usage Monitor and Inquisitor settings, if necessary.

## APF

The Inquisitor and Usage Monitor use z/OS authorized system services. These programs are contained in the PDSE Load Library SHSIMOD1, which must be authorized using APF in order to run the Usage Monitor and/or the Inquisitor when the latter is not being run with PARM=NOAPF.

## MEMLIMIT

The Usage Monitor creates memory objects, which are areas of virtual storage that have addresses greater than 2GB and can only be addressed in 64-bit addressing mode. This storage management scheme means that the Usage Monitor no longer needs to stage a lot of the collected data through ECSA, and does not need to create data spaces to hold collected data. (The Usage Monitor is no longer a consumer of the MAXCAD resource.)

The MEMLIMIT setting, which applies to the Usage Monitor address space, must be set at a value high enough to allow the Usage Monitor to create all memory objects necessary for operations. It is recommended that MEMLIMIT=NOLIMIT is used for the Usage Monitor address space.

The actual size of the memory objects that the Usage Monitor creates depends on the SIZ and QSZ settings. A z/OS system programmer must have the necessary authorities to perform this task.

### DB2 authorization

You need DB2 privileges to perform the following tasks:

- DBADM authority to access the product database. You may need to drop and create DB2 resources.
- BIND plans and packages.
- EXECUTE authority to execute plans and packages.
- SELECT authority to access the DB2 Catalog tables.
- LOAD, REPAIR, and STATS privileges to run DB2 utilities LOAD, REPAIR, and RUNSTATS.
- GRANT USE OF BUFFERPOOL privilege to use specific buffer pools.
- GRANT USE of STOGROUP privilege to use a specific storage group.
- Access to work file database or TEMP database for Declared Global Temporary table.

### SQLite authorization

To perform an installation with a SQLite database requires that authority to perform the following tasks:

- Allocate, format and mount a zFS file system.
- Grant access to z/OS OMVS groups

## Checklist of installation and customization tasks

This checklist includes a set of procedures that include installing the product, creating a test database, populating data, and validating the test installation. When you complete all of these procedures you are ready to create a production environment for IBM Tivoli Asset Discovery for z/OS.

*Table 3. Checklist of installation and customization tasks*

| Step | Description | Data sets and members |
|---|---|---|
| 1 | Install target libraries.<br><br>A z/OS system programmer performs this task.<br><br>Installing target libraries | hsi= IBM Tivoli Asset Discovery for z/OS product prefix<br>• hsi.SHSIANL1<br>• hsi.SHSIANL2<br>• hsi.SHSIEXEC<br>• hsi.SHSIGKB1<br>• hsi.SHSIMJPN<br>• hsi.SHSIMOD1<br>• hsi.SHSIPARM<br>• hsi.SHSIPROC<br>• hsi.SHSISAMP |

*Table 3. Checklist of installation and customization tasks (continued)*

| Step | Description | Data sets and members |
|---|---|---|
| 2 | Prepare database prerequisites.<br><br>A DB2 database administrator performs this task.<br><br>"Preparing DB2 database prerequisites" on page 18<br><br>The SQLite database is embedded in Tivoli Asset Discovery for z/OS and the prerequisites are already configured. If you plan to use the SQLite database for your implementation, you do not have to perform this task. | DB2 SDSNSAMP data set members:<br>• DSNTIJTM<br>• DSNTIJCL |
| 3 | Prepare local environment settings.<br><br>Tasks include editing the HSISCUST member in the SHSISAMP target library, changing the SYSIN DD entry for local settings, and running the HSISCUST job. This job generates JCL jobs that you run in subsequent tasks.<br><br>A database administrator and a Tivoli Asset Discovery for z/OS administrator perform this task.<br><br>Preparing local environment settings | hsiinst=hlq for JCLLIB, and PARMLIB libraries<br><br>hsi.SHSISAMP data set member: HSISCUST generates hsiinst.&DB.JCLLIB hsiinst.&DB.PARMLIB |
| 4 | Create a test Repository database.<br><br>A database administrator and a Tivoli Asset Discovery for z/OS administrator perform this task.<br>• "Creating a test Repository database in DB2" on page 23<br>• "Creating a test Repository database in SQLite" on page 23 | JCLLIB data set member:<br>• HSISDB01<br>• HSISDB02<br>• HSISDB03<br>• HSISGKBL<br>• HSISGRNT |
| 5 | Collect data and import it into the test Repository database.<br><br>A Tivoli Asset Discovery for z/OS administrator performs this task.<br><br>"Populating the test Repository database with data" on page 23 | JCLLIB data set members:<br>• HSISINQZ: Gather Inquisitor data.<br>• HSISINQU: Gather Inquisitor UNIX data.<br>• HSISUMON: Gather Usage Monitor data.<br>• HSISIQIM: Import Inquisitor data.<br>• HSISUIMP: Import usage data. |

*Table 3. Checklist of installation and customization tasks (continued)*

| Step | Description | Data sets and members |
|---|---|---|
| 6 | Create the production Repository database and arrange for regular maintenance.<br>• "Creating a production Repository database" on page 25<br>• "Maintaining the production Repository database" on page 27 | JCLLIB data set members:<br>• HSISCUST<br>• HSISDB01<br>• HSISDB02<br>• HSISDB03<br>• HSISGKBL<br>• HSISGRNT<br>• HSISGRTB<br>• HSIJMON<br>• HSIASALC<br>• HSIJAUTO<br>• HSIJANLO<br>• HSISANS1<br>• HSISANS2<br>• HSISANS3<br>• HSISINQZ<br>• HSISINQU<br>• HSISUMON<br>• HSISIQIM<br>• HSISUIMP<br>• HSISUDEL<br>• HSISUSUM<br>• HSISLDEL<br>• HSISTPRM<br>• HSISUN81<br>• HSISLO81<br>• HSISUT01<br>• HSISUT02<br>• HSISUT03<br>• HSISUT04<br>• HSISIVPD<br><br>PARMLIB data set member:<br>• HSISMNPM<br>• HSIAPARM<br>• HSISANP1 |

## Installing target libraries

Before you can install Tivoli Asset Discovery for z/OS in a production environment, you can create a test environment.

### Before you begin

The installation must be performed by a z/OS system programmer that has access to ShopzSeries to download the product.

### Procedure

1. Download IBM Tivoli Asset Discovery for z/OS, Version 8.1, and all available maintenance components from ShopzSeries.
2. Follow the Receive and Apply instructions in the *Tivoli Asset Discovery for z/OS Program Directory* to install the target libraries. The following libraries are installed:

| Data set low level qualifier (LLQ) | Description |
|---|---|
| SHSIANL1 | Analyzer reports for Tivoli Asset Discovery for z/OS. |
| SHSIANL2 | Java script for Tivoli Asset Discovery for z/OS. |
| SHSIEXEC | REXX code for Tivoli Asset Discovery for z/OS. |
| SHSIGKB1 | Global Knowledge base data for Tivoli Asset Discovery for z/OS. |
| SHSIMJPN | Message templates in Japanese. |
| SHSIMOD1 | Load modules for Tivoli Asset Discovery for z/OS. |
| SHSIPARM | Templates that the HSISCUST job uses to populate &HSIINST..PARMLIB library. |
| SHSIPROC | JCL PROCs for Tivoli Asset Discovery for z/OS. |
| SHSISAMP | Templates that the HSISCUST job use to populate the &HSIINST..JCLLIB library. |

3. Install all PTF maintenance packages available on the Preventive Service Planning website.
4. Ensure that the target libraries are available to the LPAR where you intend to configure the test DB2 for z/OS database.
5. Specify that the SHSIMOD1 data set is authorized by the Authorized Program Facility (APF). For example, you can enter the following command:
   ```
   SETPROG APF,ADD,DSN=hsi.SHSIMOD1,SMS
   ```
   or
   ```
   SETPROG APF,ADD,DSN=hsi.SHSIMOD1,VOL=xxxxxx
   ```
6. Schedule a change request to roll out target libraries to all z/OS LPARs where IBM Tivoli Asset Discovery for z/OS is used and include APF authorization for SHSIMOD1. For example, update the appropriate PROG*xx* member.

## Preparing DB2 database prerequisites

The DB2 environment for the test z/OS installation includes various prerequisites that you must configure.

### Before you begin

DB2 database administrator and Tivoli Asset Discovery for z/OS administrator privileges are required to perform this task.

DB2 for z/OS, Version 9 or Version 10, must be installed. DB2 must have access to a minimum of 1600 cylinders of 3390 DASD space.

### Procedure

1. Run the DSNTIJCL job from DB2 SDSNSAMP to bind the DSNACLI plan and enable the Call Library Interface (CLI/ODBC) DB2 plan. If you encounter a SQL error, code 805, rebind this plan with the latest DB2 maintenance package and include the following package in the job:
   BIND PACKAGE (DSNAOCLI) MEMBER(DSNCLIMS) - CURRENTDATA(YES) ENCODING(EBCDIC) SQLERROR(CONTINUE)
2. Run the DSNTIJTM job from DB2 SDSNSAMP to bind the DSNREXX plan and enable the REXX DB2 plan.

## Preparing local environment settings

After installation, you can create a custom version of any job in the JCLLIB library or any parameter in the PARMLIB library, by copying and editing the relevant job in the HSISCUST member in the hsi.SHSISAMP data set.

Depending on your environment, you can define parameters for the following environments:

- DB2
- SQLite
- Remote configuration

The **DBTYPE** parameter determines the environment and creates the jobs to customize and run the product in that environment.

Review the HSISCUST job parameters before you begin. A database administrator and a system programmer are required to perform the customization. After you make the required changes, submit the job. The JCL creates or reuses two output PDSE libraries and two sequential data sets.

The job creates the following PDSE libraries:

- The JCLLIB library contains a Job Control Language (JCL) script that implements and operates the product.
- The PARMLIB library contains predefined parameters that the JCL script references.

The sequential data sets are:

- The UM.HLQIDS sequential data set is referenced by the Usage Monitor on creation, and contains a single record.
- The TADZLOCK sequential data set is a dummy file used for serialization.

### General parameters

The following table lists the general parameters that you must consider for all environments.

*Table 4. General customization parameters*

| Parameter | Description |
|---|---|
| SET HSI | You must set this JCL parameter to the high-level qualifiers of the target libraries created by the SMP/E installation process. The default parameter is HSI.V810. |

*Table 4. General customization parameters (continued)*

| Parameter | Description |
|---|---|
| SET ISP | The customization tool uses ISPF services to customize the parameters and JCL for the user. This parameter specifies the high-level qualifiers for the ISPF target libraries. The default parameter begins with ISP. |
| DBTYPE | This parameter determines the environment and creates the JCL and parameters for that environment:<br>• DB2<br>• SQLITE<br>• REMOTE: The product collects Inquisitor and Usage Monitor data at remote sites and no database is required. |

## Required settings for all database types

The following table lists the required settings for all databases.

*Table 5. Required settings for all databases*

| Parameter | Description |
|---|---|
| CLASS | CLASS |
| MSGCLASS | JES message class |
| MSGLEVEL | JES message level. |
| CEERUN | This parameter specifies the fully qualified Language Environment CEERUN data set. |
| CBCDLL | This parameter specifies the fully qualified Language Environment CBCDLL C++ runtime data set. |
| HSIINST | This parameter specifies the high-level qualifiers of the JCLLIB and PARMLIB data sets that are created by running the HSISCUST job. If the JCLLIB and PARMLIB data sets exist, they are reused and you can replace members with updated information. Two other sequential data sets are either created or reused. The name specified for this parameter must be less than, or equal to, 19 characters in length. |

## Settings for DB2 and SQLite databases

The following table lists the settings for DB2 and SQLite databases.

*Table 6. Settings for DB2 and SQLite databases*

| Parameter | Description |
|---|---|
| SYS | System where database resides |
| REPZSCHM | This parameter is used as a full qualifier for the tables and index definitions in the repository, and as a part qualifier for the tables and index definitions in the local knowledge base, and local knowledge base for z/OS UNIX. The REPZSCHM name must be less than, or equal to, 8 characters in length.<br><br>If you are migrating from Tivoli Asset Discovery for z/OS, Version 7.5 to Version 8.1, the value specified for this parameter must be the same as defined for the **DB** parameter. If you specify a different value, the migration will fail. |

*Table 6. Settings for DB2 and SQLite databases (continued)*

| Parameter | Description |
|---|---|
| GKBZSCHM | This parameter is part of the table qualifier and the index definitions qualifier for the GKB, GKB for z/OS UNIX, and Inquisitor filters. The GKBZSCHM name must be less than, or equal to, 8 characters in length.<br><br>If you are migrating from Tivoli Asset Discovery for z/OS, Version 7.5 to Version 8.1, the value specified for this parameter must be the same as defined for the **DBGKB** parameter. If you specify a different value, the migration will fail. |
| DBADMIN | DBADMIN is an optional parameter. For a DB2 database, this parameter specifies the list of user IDs that are granted administrator access to the database and its contents. Specify an empty string if you do not want to grant adminstrator access to user IDs for the database specified in DB and DBGKB. For SQLite, this parameter specifies the list of user IDs that can connect to the z/OS RACF group. |
| SIZE | This parameter specifies the initial space allocations for DB2 and SQLite table spaces of the three largest tables. The default value of SIZE is 1. |

## DB2 database settings

The following table lists the DB2 database settings.

*Table 7. DB2 database settings*

| Parameter | Description |
|---|---|
| DB | This parameter specifies the name of the repository database that the product uses to store all of the information that it gathers other than from the GKB. The DB name must be less than, or equal to, 8 characters in length. |
| DBGKB | This parameter defines a single GKB database that is accessed by multiple repositories under the same DB2 subsystem. The DBGKB name must be less than, or equal to, 8 characters in length, and must not have the same name as the name defined for the DB. |
| DB2LOAD | This parameter specifies the fully qualified SDSNLOAD data set name. |
| DB2EXIT | This parameter specifies the fully qualified SDSNEXIT data set name. If the DB2EXIT library does not exist, use the same value as the DB2LOAD parameter. |
| DBSSID | This parameter specifies the DB2 subsystem ID on the z/OS System. |
| LOC | This parameter specifies the CLI/ODBC location for the DB2 subsystem ID on the z/OS system. You can use the DB2 DISPLAY DDF command to determine the Location. |
| SETSQLID | This parameter is used in SET CURRENT SQLID to allow a different user to define DB2 objects. This parameter is optional. The SETSQLID value must be less than, or equal to, 8 characters in length. |

*Table 7. DB2 database settings (continued)*

| Parameter | Description |
|---|---|
| SGHSITAB | This parameter specifies the storage group name for small tables in the database. The default value is SGHSITAB (same as the parameter name). Consult your DB2 database administrator for security implications and naming conventions. See the SQL statement CREATE STOGROUP for more information. |
| SGHSIBIG | This parameter specifies the storage group name for large tables in the database. The default value is SGHSIBIG (same as the parameter name). Consult your DB2 database administrator for security implications and naming conventions. See the SQL statement CREATE STOGROUP for more information. |
| SGHSIIDX | This parameter specifies the storage group name for indexes in the database. The default value is SGHSIIDX (same as the parameter name). Consult your DB2 database administrator for security implications and naming conventions. See the SQL statement CREATE STOGROUP for more information. |
| SGTABCAT | This parameter specifies the VCAT of the DB2 table space data set names for small tables in the database. Consult your DB2 database administrator for security implications and disk storage requirements. This parameter is referenced by storage group name parameter SGHSITAB. |
| SGTABVOL | This parameter specifies the names of the volumes that the table space data sets for small tables are allocated on. This parameter is referenced by storage group name parameter SGHSITAB. |
| SGBIGCAT | This parameter specifies the VCAT of the DB2 table space data set names for large tables in the database. Consult your DB2 database administrator for security implications and disk storage requirements. This parameter is referenced by storage group name parameter SGHSIBIG. |
| SGBIGVOL | This parameter specifies the names of the volumes that the table space data sets for large tables are allocated on. This parameter is referenced by storage group name parameter SGHSIBIG. |
| SGIDXCAT | This parameter specifies the VCAT of the DB2 data set names for indexes in the database. Consult your DB2 database administrator for security implications and disk storage requirements. This parameter is referenced by storage group name parameter SGHSIIDX. |
| SGIDXVOL | This parameter specifies the names of the volumes that the data sets, for indexes, are allocated on. This parameter is referenced by storage group name parameter SGHSIIDX. |
| • BPDB<br>• BPTS<br>• BPIX | These parameters specify the buffer pool definitions for the database, table spaces, and indexes. See Appendix D, “Performance and tuning,” on page 241. |

## SQLite database settings

The following table lists SQLite database settings.

*Table 8. SQLite database settings*

| Parameter | Description |
|---|---|
| SQLTZFS | zFS linear vsam dataset name that is used for Tivoli Asset Discovery for z/OS SQLite databases. |

*Table 8. SQLite database settings (continued)*

| Parameter | Description |
|---|---|
| SQLTPATH | USS directory where the SQLTZFS dataset is mounted. The HSISDB01 job in JCLLIB creates this path at a later time. |

# Configuring a test Repository database

Configuring the test Repository database includes setting up database objects, creating the Global Knowledge Base (GKB) database, and configuring access to the Repository and the GKB databases. Most sites maintain both a test Repository database and a production Repository database. After you configure and validate the test Repository database, repeat this task to create the production Repository database.

## Creating a test Repository database in DB2

Creating the Tivoli Asset Discovery for z/OS database includes setting up storage groups, the database name, and the administrator logon details. You also create the Global Knowledge Base (GKB) environment and then grant access to the database.

### Procedure

1. Run the HSISDB01 job to create the storage groups.
2. Run the HSISDB02 job to create the database objects for the GKB.
3. Run the HSISDB03 job to create the Repository database and database objects.
4. Run the HSISGKBL job to load the GKB.
5. Run the HSISGRNT job to grant DBADMIN access to the Tivoli Asset Discovery for z/OS administrator to the Repository and the GKB databases.

## Creating a test Repository database in SQLite

Creating the test Repository database includes allocating, formatting and mounting a zFS file system and also grant access to z/OS OMVS groups. Creating the Tivoli Asset Discovery for z/OS database includes setting up storage groups, the database name, and the administrator logon details. You also create the Global Knowledge Base (GKB) environment and then grant access to the database.

### Procedure

1. Run the HSISDB01 job to allocate, format, and mount a zFS file system.
2. Run the HSISDB02 job to create the database objects for the Global Knowledge Base (GKB).
3. Run the HSISDB03 job to create the Repository database and database objects.
4. Run the HSISGKBL job to load the GKB.
5. Run the HSISGRNT job to grant access to the z/OS OMVS groups that the Tivoli Asset Discovery for z/OS administrator is a member of.

## Populating the test Repository database with data

You can populate the test Repository database in stages. Begin by collecting and importing Inquisitor and Usage Monitor data on the local LPAR, and then verify that this process is successful before collecting and importing data from other LPARs.

**Collecting and importing data to the test Repository database:**

After creating the databases and database objects, you are ready to collect Inquisitor and Usage Monitor data. You can then import the collected data into the test Repository database.

**Procedure**

1. Run the HSISINQZ job to scan the DASD for z/OS product modules and generate output to the DD `HSIPZIP` output file. For large sites, this operation can take up to an hour. You can perform steps 3 and 4 while the job is running.
2. Run the HSISINQU job to scan z/OS UNIX files and generate output to the DD `HSIXZIP` output file. For large sites, this operation can take up to an hour. You can perform steps 3 and 4 while the job is running.
3. To run the Usage Monitor to gather initial usage data, perform the following tasks:
   a. Run the HSISUMON job to start the Usage Monitor as a batch job. The Usage Monitor is typically run as a started task, but you can run it as a batch job for this test. This job runs continually until you stop it manually and most of the time this job is idle.
   b. Stop the Usage Monitor to generate the `hsiinst.UM&SMF.D*.T*` output file. For example, enter the following command to stop the started task:
   `P HSIJMON`
4. To import Inquisitor (IQ) data into the test Repository database, perform the following tasks:
   a. Verify that the HSISINQZ and HSISINQU jobs that you started in steps 1 and 2 have completed. If the jobs are still running, wait until they are completed. The output logs from these jobs provide information on the number of records collected.
   b. Run the HSISIQIM job to import the data from the `HSIPZIP` and `HSIXZIP` output files that were created by the HSISINQZ and HSISINQU jobs. For large sites, this job can take at least 2 hours to run the first time. Performance is 90 per cent faster on subsequent runs.
5. Run the HSISUIMP job to import usage data from the `hsiinst.UM&SMF.D*.T*` file.

**Verifying the results of the data import with the Analyzer:**

After you complete the collection and import of Inquisitor and Usage Monitor data, use the Analyzer to verify that the import was successful.

**Procedure**

1. Review the HSISANP1 PARMLIB library settings and modify if necessary. These settings specify the Tivoli Asset Discovery for z/OS administrator user id and password.
2. Run the HSISANLO JCLLIB job on the test Repository database. This job, typically, runs continually but you can enter the `F HSISANLO, STOP` command to stop it.
3. On your PC browser, logon to the Analyzer utility with the values specified in the HSISANP1 PARMLIB library.
4. Review the Analyzer reports to confirm that all expected products have been identified. If a product is missing, perform the following tasks to identify the reason why a product is not included:

- Check that the product is in the GKB and report any missing product to IBM support so that they can provide an updated GKB for the product.
- If the product exists in the GKB, check that the product is installed on the test z/OS. If the product is not installed on the test z/OS, run the Inquisitor utility on a system where the product is installed and then import that data into the test database.

**Collecting and importing data from other systems:**

After you verify that all components are correctly installed on the test Repository database, you can now discover and import Inquisitor and Usage Monitor data from other z/OS logical partitions (LPARs).

**Procedure**

1. Run the following jobs to collect Inquisitor and Usage data from other systems:
   a. Run the HSISINQZ job to scan all other LPARs and generate output to the `hsiinst.HSIPZIP.Z&SMF` file.
   b. Run the HSISINQU job and generate output to the `hsiinst.HSIUZIP.U&SMF` file.
   c. Run the HSISUMON job to start the Usage Monitor as a batch job on the other LPARS.
2. Transfer collected data to the central site via file transfer protocol (FTP).
3. Run the following jobs to import Inquisitor and Usage data at the central site:
   a. Run the HSISIQIM job to import Inquisitor data from the `hsiinst.HSIPZIP.Z&SMF` and `hsiinst.HSIUZIP.U&SMF` files for each LPAR.
   b. Run the HSISUIMP job to import Usage data from the `hsiinst.UM &SMF.D*.T*` file for each LPAR.

# Configuring a production Repository database

Most implementations include a test Repository database and a production Repository database. Configuring a production Repository database involves creating the database and importing data, configuring security, and automating data collection activities

## Creating a production Repository database

The production Repository database runs on a development logical partition (LPAR) and it is not necessary to run it on a business workload LPAR. You can duplicate the content of test Repository database to populate production Repository database without collecting and importing Inquisitor and Usage Monitor data again.

### About this task

You can create the production Repository database on a DB2 or SQLite database. This procedure combines instructions for both database environments. Refer to the instructions for creating a test Repository database if you require database-specific instructions.

### Procedure

1. Run the HSISDB01 job.
   For DB2, the job creates storage groups.
   For SQLite, the job allocates the zFS file system.

2. Run the HSISDB02 and HSISDB03 jobs to create the Global Knowledge Base (GKB) and Repository databases and database objects.
3. Run the HSISGKBL job to load the GKB.
4. Run the HSISGRNT job.
   For DB2, the job grants DBADMIN access to the Tivoli Asset Discovery for z/OS administrator for the Repository and GKB databases.
   For SQLite, the job grants access to the z/OS OMVS groups.
5. Run the HSISGRTB job.
   For DB2, this job grants SELECT access to database tables.
6. To populate the production Repository database, repeat the procedure for collecting and importing data that you performed to populate the test Repository database.

### What to do next

Configure security for the production Repository database.

## Configuring security for the production Repository database

Resource Access Control Facility (RACF) security provides authentication, authorization, and auditing control for working with z/OS systems.

### Procedure

1. Define a profile in the STARTED class to associate a user ID with the HSIJMON, HSIJAUTO, and HSIJANLO started tasks.
2. Specify that user IDs have the following access permissions:
   a. READ access to hsi** data sets
   b. ALTER access to hsiinst.** data sets

### What to do next

Configure the automation of data collection activities on the production Repository database.

## Automating data collection and reporting activities

When you configure the Usage Monitor, the Automation Server, and the Analyzer to run as started tasks, these data collection and reporting activities are automated.

### Procedure

1. Configure the Usage Monitor utility to start automatically:
   a. In the HSISMNPM member of the PARMLIB data set, modify settings if necessary so that the **DSN(hsiinst.UM&SMF)** command generates hsiinst.UM&SMF.D*.T* data sets.
   b. Schedule a change request to roll out the new HSIJMON started task on all z/OS LPARs.
   c. Copy the HSIJMON started task from the JCLLIB libray to the system PROCLIB data set.
   d. Arrange for the HSIJMON started task to start early in the initial program load (IPL) cycle to ensure that all usage activity is recorded.
2. Configure the Automation Server utility to start automatically and to automate data collection and import tasks:
   a. Schedule a change request to roll out a new HSIJAUTO started task on all z/OS LPARs.

b. Run the HSIASALC job to define the automation control Virtual Storage Access Method (VSAM) data set.
c. Configure the HSIAPARM settings to perform the following tasks every weekend:
   - Remote hosts: Runs an Inquisitor scan job to collect data, runs the ZCAT to amalgamate usage data, and transfers collected data via file transfer protocol (FTP).
   - Database host: Runs an Inquisitor import job, runs a usage import job, and aggregates the data.
d. Optional: If necessary, run the HSIASSCT job to mark existing data sets as being already processed in the automation control data set.
e. Copy the HSIJAUTO started task from the JCLLIB library to the system PROCLIB data set.
f. Arrange for the HSIJAUTO to start automatically at any time in the IPL cycle.

3. Configure the Analyzer utility to start automatically:
   a. Schedule a change request to roll out the new HSIJANLO started task to the production database host.
   b. Copy the HSIJANLO started task from the JCLLIB data set to the system PROCLIB data sets.
4. Configure the Analyzer utility to work with a secure socket layer (SSL) for HTTPS transport and to logon with a RACF user ID and password:
   a. In the HSISANP2 member of the PARMLIB data set, change the security parameter to SECURITY=SYSTEM,
   b. Review and edit the comments in the HSISANS1, HSISANS2, and HSISANS3 members of the JCLLIB data set to create a digital certificate that is required for SSL.
   c. Configure the HTTPPORT parameter, if you require a value other than the default value.
   d. Review the Analyzer reports to confirm that all expected products are identified.

## Maintaining the production Repository database

You must perform regular maintenance tasks on the production Repository database to ensure that performance is optimal. The maintenance tasks cull obsolete and unwanted data and reorganize the database as necessary.

### About this task

A database administrator or system programmer performs these maintenance tasks.

### Procedure

1. Run the following jobs on a regular basis to delete old usage data, save space, and improve processing time:
   a. Run the HSISUDEL job to delete usage data that are older than a specified period.
   b. Run the HSISUSUM job to summarize usage data and compress records into monthly periods.
2. Run the HSISLDEL job to delete obsolete discovery and usage data for a specified system (LPAR).

3. Run the HSISTPRM job to reset the status flag back to normal for tables in the production Repository database, following a failure.
4. Run the HSISIVP job to verify database changes since the product was released.
5. Run the following jobs on a regular basis to maintain the integrity and performance of data in the production Repository database:
   a. Run the HSISUT01 job to backup the Repository database in DB2 or backup the zFS file system in SQLite.
   b. Run the HSISUT02 job to restore the Repository database in DB2 or restore the zFS file system in SQLite.
   c. Run the HSISUT03 job to reorganize the Repository database in DB2.
   d. Run the HSISUT04 job to update Runstats statistics for the Repository database in DB2.

# Chapter 4. Implementing deployment scenarios

Most implementations of Tivoli Asset Discovery for z/OS are based on one of the common deployment scenarios. An example is provided for implementing each of these common deployment scenarios with a DB2 Repository database. You can adapt an example for use with a SQLite Repository database.

**Related concepts**:

Chapter 2, "Planning for deployment," on page 9
Before you deploy IBM Tivoli Asset Discovery for z/OS, consider which deployment option is best suited to your environment.

## Scenario 1: Implementing a single Repository database with a single GKB database

The most common deployment scenario is an implementation with a single Repository database and a single global knowledge base (GKB) database.

### About this task

The example deployment is for a DB2 database environment and includes the key parameters that influence this scenario.

### Procedure

1. Customize an instance of the HSISCUST member in the hsi.SHSISAMP data set with the following parameters:
   - **DBTYPE**=DB2
   - **REPZSCHM**=TADZRE1
   - **GKBZSCHM**=TADZGK1
   - **DB**=TADZREP1
   - **DBGKB**=TADZGKB1
2. Submit the HSISCUST job.
3. Create the Repository and GKB databases and grant access to them:
   a. Run the HSISDB01 job to create storage groups.
   b. Run the HSISDB02 job to create the GKB database and database objects.
   c. Run the HSISDB03 job to create the Repository database and database objects.
   d. Run the HSISGKBL job to load GKB data.
   e. Run the HSISGRNT job to grant DBADMIN access to Tivoli Asset Discovery for z/OS administrator.
   f. Run the HSISGRTB job to grant SELECT access to database tables.
4. Collect Inquisitor and Usage Monitor data:
   a. Run the HSISINQZ job on all z/OS LPARs to collect Inquisitor data.
   b. Run the HSISINQU job on all z/OS LPARs to collect Inquisitor data for UNIX.
   c. Run the HSISUMON job on all z/OS LPARs to collect usage data.
5. Transfer the collected Inquisitor and Usage Monitor data to the central site via file transfer protocol (FTP).

6. Import Inquisitor and Usage Monitor data at the central site:
   a. Run the HSISIQIM job to import Inquisitor data into the Repository database for each LPAR.
   b. Run the HSISUIMP job to import Usage data into the Repository database for each LPAR.

## Scenario 2: Implementing multiple Repositories with a shared GKB database

This deployment scenario implements two Repositories in a single DB2 subsystem that share a single global knowledge base (GKB) database. The advantage of sharing the same GKB is that you need only apply monthly updates to a single GKB database.

### About this task

The example deployment is for two Repositories in the same DB2 subsystem to enable the Analyzer to browse both Repositories at the same time.

### Procedure

1. Customize an instance of the HSISCUST member in the hsi.SHSISAMP data set with the following parameters:
   - **DBTYPE**=DB2
   - **REPZSCHM**=TADZRE1
   - **GKBZSCHM**=TADZGK1
   - **DB**=TADZREP1
   - **DBGKB**=TADZGKB1
2. Submit the HSISCUST job.
3. Create the Repository and GKB database and grant access to them:
   a. Run the HSISDB01 job to create storage groups.
   b. Run the HSISDB02 job to create the GKB database and database objects.
   c. Run the HSISDB03 job to create the Repository database and database objects.
   d. Run the HSISGKBL job to load GKB data.
   e. Run the HSISGRNT job to grant DBADMIN access to Tivoli Asset Discovery for z/OS administrator.
   f. Run the HSISGRTB job to grant SELECT access to database tables.
4. Collect Inquisitor and Usage Monitor data:
   a. Run the HSISINQZ job on all z/OS LPARs to collect Inquisitor data.
   b. Run the HSISINQU job on all z/OS LPARs to collect Inquisitor data for UNIX.
   c. Run the HSISUMON job on all z/OS LPARs to collect usage data.
5. Transfer the collected Inquisitor and Usage Monitor data to the central site via file transfer protocol (FTP).
6. Import Inquisitor and Usage Monitor data at the central site:
   a. Run the HSISIQIM job to import Inquisitor data into the Repository database for each LPAR.
   b. Run the HSISUIMP job to import Usage data into the Repository database for each LPAR.

7. Customize another instance of the HSISCUST member in the hsi.SHSISAMP data set with the following parameters:
   - **DBTYPE**=DB2
   - **REPZSCHM**=TADZRE2
   - **GKBZSCHM**=TADZGK1
   - **DB**=TADZREP2
   - **DBGKB**=TADZGKB1
8. Create the second Repository and grant access to it:
   It is not necessary to run jobs to create and populate the GKB database in this step because the second Repository shares the GKB that you created in Step 2.
   a. Run the HSISDB01 job to create storage groups.
   b. Run the HSISDB03 job to create the Repository database and database objects.
   c. Run the HSISGRNT job to grant DBADMIN access to Tivoli Asset Discovery for z/OS administrator.
   d. Run the HSISGRTB job to grant SELECT access to database tables.
9. Collect Inquisitor and Usage Monitor data to add to the second Repository database:
   a. Run the HSISINQZ job on all z/OS LPARs to collect Inquisitor data.
   b. Run the HSISINQU job on all z/OS LPARs to collect Inquisitor data for UNIX.
   c. Run the HSISUMON job on all z/OS LPARs to collect usage data.
10. Transfer the collected Inquisitor and Usage Monitor data to the central site via file transfer protocol (FTP).
11. Import Inquisitor and Usage Monitor data at the central site:
    a. Run the HSISIQIM job to import Inquisitor data into the second Repository database for each LPAR.
    b. Run the HSISUIMP job to import Usage data into the second Repository database for each LPAR.

### What to do next

Repeat steps 7 -11 for each additional Repository that you want to create, changing the values for the **REPZSCHM** and **DB** parameters for each new Repository.

## Scenario 3: Implementing multiple Repositories with multiple GKB databases

This deployment scenario implements two Repositories in a single DB2 subsystem, each with its own global knowledge base (GKB) database. This deployment scenario is not common because you must apply monthly updates to each GKB database.

### About this task

The example deployment is for two Repositories in the same DB2 subsystem to enable the Analyzer to browse both Repositories at the same time.

### Procedure

1. Customize an instance of the HSISCUST member in the hsi.SHSISAMP data set with the following parameters:

- **DBTYPE**=DB2
- **REPZSCHM**=TADZRE1
- **GKBZSCHM**=TADZGK1
- **DB**=TADZREP1
- **DBGKB**=TADZGKB1

2. Submit the HSISCUST job.
3. Create the first Repository and GKB database and grant access to them:
   a. Run the HSISDB01 job to create storage groups.
   b. Run the HSISDB02 job to create the GKB database and database objects.
   c. Run the HSISDB03 job to create the Repository database and database objects.
   d. Run the HSISGKBL job to load GKB data.
   e. Run the HSISGRNT job to grant DBADMIN access to Tivoli Asset Discovery for z/OS administrator.
   f. Run the HSISGRTB job to grant SELECT access to database tables.
4. Collect Inquisitor and Usage Monitor data:
   a. Run the HSISINQZ job on all z/OS LPARs to collect Inquisitor data.
   b. Run the HSISINQU job on all z/OS LPARs to collect Inquisitor data for UNIX.
   c. Run the HSISUMON job on all z/OS LPARs to collect usage data.
5. Transfer the collected Inquisitor and Usage Monitor data to the central site via file transfer protocol (FTP).
6. Import Inquisitor and Usage Monitor data at the central site:
   a. Run the HSISIQIM job to import Inquisitor data into the Repository database for each LPAR.
   b. Run the HSISUIMP job to import Usage data into the Repository database for each LPAR.
7. Customize another instance of the HSISCUST member in the hsi.SHSISAMP data set with the following parameters:
   - **DBTYPE**=DB2
   - **REPZSCHM**=TADZRE2
   - **GKBZSCHM**=TADZGK2
   - **DB**=TADZREP2
   - **DBGKB**=TADZGKB2
8. Submit the HSISCUST job.
9. Create the second Repository and second GKB database grant access to them:
   a. Run the HSISDB01 job to create storage groups.
   b. Run the HSISDB02 job to create the GKB database and database objects.
   c. Run the HSISDB03 job to create the Repository database and database objects.
   d. Run the HSISGKBL job to load GKB data.
   e. Run the HSISGRNT job to grant DBADMIN access to Tivoli Asset Discovery for z/OS administrator.
   f. Run the HSISGRTB job to grant SELECT access to database tables.
10. Collect Inquisitor and Usage Monitor data for the second Repository database:
    a. Run the HSISINQZ job on all z/OS LPARs to collect Inquisitor data.

b. Run the HSISINQU job on all z/OS LPARs to collect Inquisitor data for UNIX.
c. Run the HSISUMON job on all z/OS LPARs to collect usage data.

11. Transfer the collected Inquisitor and Usage Monitor data to the central site via file transfer protocol (FTP).
12. Import Inquisitor and Usage Monitor data at the central site:
    a. Run the HSISIQIM job to import Inquisitor data into the second Repository database for each LPAR.
    b. Run the HSISUIMP job to import Usage data into the second Repository database for each LPAR.

### What to do next

Repeat steps 7- 12 for each additional Repository and GKB database that you want to create, changing the values for **REPZSCHM**, **GKBZSCHM**, **DB**, and **DBGKB** parameters for each new Repository and GKB database.

## Scenario 4: Collecting and transferring Inquisitor and usage data from remote sites

This scenario extends each of the deployment scenarios to collect data from remote sites and transfer the data back to the central site for processing.

### Procedure

1. At the remote site, install the target libraries.
2. Customize an instance of the HSISCUST member in the hsi.SHSISAMP data set with the following parameter:
   **DBTYPE**=REMOTE
3. Submit the HSISCUST job.
4. Collect Inquisitor and Usage Monitor data and transfer the files to the central site for processing:
   a. Run the HSISINQZ job on all z/OS LPARs to collect Inquisitor data.
   b. Run the HSISINQU job on all z/OS LPARs to collect Inquisitor data for UNIX.
   c. Run the HSISUMON job on all z/OS LPARs to collect usage data.
   d. Transfer the collected Inquisitor and Usage Monitor data to the central site via file transfer protocol (FTP).

## Scenario 5: Implementing in a sysplex environment

This deployment scenario is for a sysplex environment where the DASD is fully shared across all z/OS LPARs that belong to the sysplex. This special deployment is similar to the deployment scenarios 1, 2, or 3 but the implementation steps are slightly different. The reason for using this approach is to achieve operational efficiency by just processing a single z/OS LPAR within a sysplex.

### About this task

The example deployment is for a DB2 database environment and includes the key parameters that influence this scenario. For this scenario, assume that the sysplex contains four z/OS LPARs: MVSA, MVSB, MVSC, and MVSD.

## Procedure

1. Customize an instance of the HSISCUST member in the hsi.SHSISAMP data set with the following parameters:
   a. DBTYPE=DB2
   b. REPZSCHM=TADZRE1
   c. GKBZSCHM=TADZGK1
   d. DB=TADZREP1
   e. DBGKB=TADZGKB1
2. Submit the HSISCUST job.
3. Create the Repository and GKB databases and grant access to them:
   a. Run the HSISDB01 job to create storage groups.
   b. Run the HSISDB02 job to create the GKB database and database objects.
   c. Run the HSISDB03 job to create the Repository database and database objects.
   d. Run the HSISGKBL job to load GKB data.
   e. Run the HSISGRNT job to grant DBADMIN access to Tivoli Asset Discovery for z/OS administrator.
   f. Run the HSISGRTB job to grant SELECT access to database tables.
4. Collect and import Inquisitor data for for all z/OS LPARs the first time:
   a. Run the HSISINQZ job on all four z/OS LPARs to collect Inquisitor data: MVSA, MVSB, MVSC, and MVSD.
   b. Transfer the collected Inquisitor data to the central site via file transfer protocol (FTP).
   c. Run the HSISIQIM job to import Inquisitor data into the Repository database for each z/OS LPAR.
5. Collect and import Inquisitor data only for a single z/OS LPAR in subsequent scans:
   a. Set PLX=Y in the Inquisitor HSISINQZ job.
   b. Run the HSISINQZ job on the first z/OS LPAR, MVSA, to collect Inquisitor data.
   c. Transfer the collected Inquisitor data to the central site via FTP.
   d. Run the HSISIQIM job to import Inquisitor data into the Repository database for the z/OS LPAR MVSA only.
   e. Repeat steps a - d for z/OS LPAR MVSA every time a new scan is required.
6. Collect and import Inquisitor data for UNIX for all z/OS LPARs:
   a. Run the HSISINQU job on all four z/OS LPARs to collect Inquisitor data for UNIX.
   b. Transfer the collected Inquisitor data for UNIX to the central site via FTP.
   c. Run the HSISIQIM job to import Inquisitor data for UNIX into the Repository database for each z/OS LPAR.
   d. Repeat steps a - c for each z/OS LPAR every time a new scan is required.
7. Collect and import Usage Monitor data:
   a. Run the HSISUMON job on all z/OS LPARs to collect usage data.
   b. Transfer the collected Usage Monitor data to the central site via FTP.
   c. Run the HSISUIMP job to import Usage data into the Repository database for each LPAR.

# Chapter 5. Migrating to IBM Tivoli Asset Discovery for z/OS, version 8.1

When you migrate to the latest version of Tivoli Asset Discovery for z/OS from an earlier version, you must convert existing data to be compatible with your new environment.

## Migrating to Tivoli Asset Discovery for z/OS from an earlier version

You can upgrade to Tivoli Asset Discovery for z/OS, version 8.1 from version 7.5 or version 7.2. This release introduces support for SQLite database, and you can migrate to either a DB2 Repository database or a SQLite database.

### Migrating from version 7.5 to Tivoli Asset Discovery for z/OS version 8.1 ( DB2 database)

When you upgrade to Tivoli Asset Discovery for z/OS version 8.1 for DB2 database, you do not have to port any data from the Repository database. The migration tasks focus on defining new DB2 objects and dropping obsolete DB2 objects.

#### Before you begin

Make a backup of your Tivoli Asset Discovery for z/OS version 7.5 Repository database.

Make a backup or rename your JCLLIB and PARMLIB data sets.

#### About this task

Perform these migration tasks for every DB2 Repository in your Tivoli Asset Discovery for z/OS environment.

#### Procedure

1. In Tivoli Asset Discovery for z/OS version 8.1, make a copy of the HSISCUST member in the hsi.SHSISAMP data set and modify the following parameters:
   a. Set the value of the new **DBTYPE** parameter to DB2.
   b. Set the value of the new **SYS** parameter to the system where the Repository database is located.
   c. Set the value of the **DB** parameter to the same value that is defined for your existing version 7.5 Repository database.
   d. Set the value of the **DBGKB** parameter to the same value that is defined for your existing 7.5 Global Knowledge Base (GKB) database.
   e. Set the value of the new **REPZSCHM** parameter to the same value that is defined for the **DB** parameter.
   f. Set the value of the new **GKBZSCHM** parameter to the same value that is defined for the **DBGKB** parameter.
2. Submit the HSISCUST job.
3. Edit and update jobs in the JCLLIB library and parameters in the PARMLIB library if there are special site requirements.

4. Run the following migration jobs:
   a. Submit the HSISMI76 job to verify whether two version 7.5 PTFs have been implemented with database changes in the version 7.5 repository.

      PTF UA65570 in version 7.5 adds a new column, MODEL_CAPACITY to table NODE_CAPACITY.

      PTF UA70726 modifies the three tables below:

| | |
|---|---|
| **TUSEMTD** | Column FMTDID data type changed to FLOAT. |
| **TUSEPOV** | Column FUSEPOVINVID data type changed to FLOAT. |
| **TUSEPOVLIB** | Column FUSEPOVLIBID data type changed to FLOAT. |

   b. Submit the HSISMI81 job to add new DB2 objects to the Repository database. A condition code of 0 is expected.
   c. Submit the HSISMI82 job to populate records and also delete obsolete records in some Repository tables. A condition code of 0 is expected.
   d. Submit the HSISMI83 job to drop obsolete DB2 objects from the Repository database. A condition code of 0 is expected
   e. Submit the HSISMI84 job to verify that the previous migration tasks have been successfully implemented. A condition code of 0 is expected
5. Submit the HSISGKBL job to populate the GKB database. GKB level 20150529 is shipped with this migration. To download the latest GKB level, refer to topic "Updating the Global Knowledge Base" on page 41.

## What to do next

After migration, use the following approach to manage the implementation to the new version.:

- Continue to use existing version 7.5 Inquisitor fully-scanned files as inputs for the version 8.1 HSISIQIM Inquisitor Import job.
  1. For each repository run all HSISIQIM jobs with setting of FULLREMATCH=Y. Please read the comments in the "Performance consideration" section of job HSISIQIM job before you proceed
  2. For the last HSISIQIM job, update the Aggregator jobstep with COUNTUSAGEFULL=Y. For example:

     ```
     //AGGR EXEC HSIJSQLE,PROG=HSICTLAG,TPARAM=HSISAGP1
     //USERPARM DD *
     COUNTUSAGEFULL=Y
     ```

  3. Run the last HSISIQIM job with COUNTUSAGEFULL=Y for the Aggregator job step.
  4. Repeat steps 1 to 3 for the next repository.
  5. After running the last HSISIQIM job, set COUNTUSAGEFULL=N (the default setting).
- Continue to use existing version 7.5 usage data files as inputs for the version 8.1 HSISUIMP Usage Import job.
- Configure APF authorization for the version 8.1 SHSIMOD1 load library
- When the version 8.1 Inquisitor scans and Usage Monitors are ready for use, you can run 8.1 operational jobs and you can discontinue version 7.5 tasks.

# Migrating from version 7.5 to Tivoli Asset Discovery for z/OS version 8.1 (SQLite database)

Tivoli Asset Discovery for z/OS version 8.1 introduces support for SQLite database. When you migrate from version 7.5 you can port your data from the DB2 Repository to the SQLite database.

## Before you begin

Make a backup or rename your JCLLIB and PARMLIB data sets. Before considering porting your data from the DB2 Repository to the SQLite database, refer to the Product Overview section on the limitations of using SQLite

## Procedure

1. In Tivoli Asset Discovery for z/OS version 8.1, make a copy of the HSISCUST member in the hsi.SHSISAMP data set and modify the following parameters:
   a. Set the value of the new **DBTYPE** parameter to SQLITE.
   b. Set the value of the new **SYS** parameter to the system where the SQLite Repository database is located.
   c. Set the value of the new **REPZSCHM** parameter to the name of the table owner for the SQLite Repository objects.
   d. Set the value of the new **GKBZSCHM** parameter to the name of the table owner for the GKB objects.
   e. Set the value of the new **SQLTZFS** parameter to the name of the zFS linear VSAM data set.
   f. Set the value of the new **SQLTPATH** parameter to the path of the USS directory.
2. Submit the HSISCUST job.
3. Edit and update jobs in the JCLLIB library and parameters in the PARMLIB library if there are special site requirements.
4. Submit the following jobs:
   a. Submit the HSISDB01 job to define the zFS VSAM linear data set
   b. Submit the HSISDB02 job to create the GKB database.
   c. Submit the HSISDB03 job to create the Repository database.
   d. Submit the HSISGKBL job to populate the GKB database. GKB level 20150529 is shipped with this migration. To download the latest GKB level, refer to topic "Updating the Global Knowledge Base" on page 41.
5. Run the HSISUNLD job from the version 7.5 JCLLIB library to unload data from the version 7.5 DB2 Repository database.
6. In version 8.1, submit the HSISMI8Q job to load the unloaded data into the SQLite Repository.

## What to do next

After migration, use the following approach to implement the new version:

- Continue to use existing version 7.5 Inquisitor fully-scanned files as inputs for the version 8.1 HSISIQIM Inquisitor Import job.
  1. For each repository run all HSISIQIM jobs with setting of FULLREMATCH=Y. Please read the comments in the "Performance consideration" section of job HSISIQIM job before you proceed.

2. For the last HSISIQIM job, update the Aggregator job step with COUNTUSAGEFULL=Y. For example:

```
//AGGR EXEC HSIJSQLE,PROG=HSICTLAG,TPARAM=HSISAGP1
//USERPARM DD *
COUNTUSAGEFULL=Y
```

3. Run the last HSISIQIM job with COUNTUSAGEFULL=Y for the Aggregator job step.
4. Repeat steps 1 to 3 for the next repository.
5. After running the last HSISIQIM job, set COUNTUSAGEFULL=N (the default setting).

- Continue to use the usage data files from version 7.5 as inputs for the version 8.1 HSISUIMP Usage Import job.
- Configure APF authorization for the version 8.1 SHSIMOD1 load library.
- When the version 8.1 Inquisitor scans and Usage Monitor data files are ready for use, run the version 8.1 operational jobs and discontinue version 7.5 tasks.

## Migrating from version 7.2 to Tivoli Asset Discovery for z/OS version 8.1 (DB2 database)

When you upgrade to Tivoli Asset Discovery for z/OS version 8.1 for DB2, you must define new Global Knowledge Base (GKB) and Repository databases. You can port version 7.2 usage data across to version 8.1 for DB2.

### Procedure

1. In Tivoli Asset Discovery for z/OS version 8.1, make a copy of the HSISCUST member in the hsi.SHSISAMP data set and modify the following parameters:
   a. Set the value of the new **DBTYPE** parameter to DB2.
   b. Set the value of the new **SYS** parameter to the system where the Repository database is located.
   c. Set the value of the **DB** parameter to the name of the Repository database.
   d. Set the value of the **DBGKB** parameter to the name of the GKB database.
   e. Set the value of the new **REPZSCHM** parameter to the name of the table owner of **DB** Repository tables.
   f. Set the value of the new **GKBZSCHM** parameter to the name of the table owner for the **DBGKB** tables.
2. Submit the HSISCUST job.
3. Edit and update jobs in the JCLLIB library and parameters in the PARMLIB library if there are special site requirements.
4. Submit the following jobs to create the databases:
   a. Submit the HSISDB01 job to define DB2 storage groups.
   b. Submit the HSISDB02 job to create the GKB database.
   c. Submit the HSISDB03 job to create the Repository database
   d. Submit the HSISGKBL job to populate the GKB database.
   e. Submit the HSISGRNT job to grant DBADM access to databases.
5. Submit the HSISMI75 migration job to export usage data from the version 7.2 Repository database.
6. Submit the HSISUIMP usage import job to import usage data from the previous step.

### What to do next

After migration, use the following approach to manage the implementation to the new version:

- Configure APF authorization for version 8.1 of the SHSIMOD1 load library.
- Run version 8.1 of the HSISINQZ job on all z/OS LPARs to collect Inquisitor data.
- Run version 8.1 of the HSISINQU job on all z/OS LPARs to collect Inquisitor data for UNIX.
- Run version 8.1 of the HSISUMON job on all z/OS LPARs to collect usage data.
- Transfer the collected Inquisitor and Usage Monitor data to the central site via file transfer protocol (FTP).
- Run version 8.1 of the HSISIQIM job to import Inquisitor data into the Repository database for each LPAR.
- Run version 8.1 of the HSISUIMP job to import Usage data into the Repository database for each LPAR.

Data from version 7.2 of the Inquisitor and Usage files cannot be imported into a version 8.1 repository.

## Migrating from version 7.2 to Tivoli Asset Discovery for z/OS version 8.1 (SQLite database)

Tivoli Asset Discovery for z/OS version 8.1 introduces support for SQLite database. When you migrate from version 7.2 you can port your data from the DB2 Repository to the SQLite database.

### Procedure

1. In Tivoli Asset Discovery for z/OS version 8.1, make a copy of the HSISCUST member in the hsi.SHSISAMP data set and modify the following parameters:
   a. Set the value of the new **DBTYPE** parameter to SQLITE.
   b. Set the value of the new **SYS** parameter to the system where the SQLite Repository database is located.
   c. Set the value of the new **REPZSCHM** parameter to the name of the table owner for the SQLite Repository objects.
   d. Set the value of the new **GKBZSCHM** parameter to the name of the table owner for the GKB objects.
   e. Set the value of the new **SQLTZFS** parameter to the name of the zFS linear VSAM data set.
   f. Set the value of the new **SQLTPATH** parameter to the path of the USS directory.
2. Submit the HSISCUST job.
3. Edit and update jobs in the JCLLIB library and parameters in the PARMLIB library if there are special site requirements.
4. Submit the following jobs:
   a. Submit the HSISDB01 job to define the zFS VSAM linear data set
   b. Submit the HSISDB02 job to create the GKB database.
   c. Submit the HSISDB03 job to create the Repository database.
   d. Submit the HSISGKBL job to populate the GKB database.
   e. Submit the HSISGRNT job to grant access to z/OS OMVS groups.

5. Submit the HSISMI75 migration job to export usage data from the version 7.2 Repository database.
6. Submit the HSISUIMP usage import job to import usage data from the previous step.

### What to do next

After migration, use the following approach to manage the implementation to the new version:

- Configure APF authorization for the version 8.1 SHSIMOD1 load library.
- Run version 8.1 of the HSISINQZ job on all z/OS LPARs to collect Inquisitor data.
- Run version 8.1 of the HSISINQU job on all z/OS LPARs to collect Inquisitor data for UNIX.
- Run version 8.1 of the HSISUMON job on all z/OS LPARs to collect usage data.
- Transfer the collected Inquisitor and Usage Monitor data to the central site via file transfer protocol (FTP).
- Run version 8.1 of the HSISIQIM job to import Inquisitor data into the Repository database for each LPAR.
- Run version 8.1 of the HSISUIMP job to import Usage data into the Repository database for each LPAR.

Data from version 7.2 Inquisitor and Usage files cannot be imported into a version 8.1 repository.

## Migrating from Tivoli License Compliance Manager for z/OS, version 4.2 to Tivoli Asset Discovery for z/OS. version 8.1

Tivoli License Compliance Manager for z/OS, version 4.2, customers must implement Tivoli Asset Discovery for z/OS, version 8.1, as a new install using either a DB2 database or SQLite database. Porting of Tivoli License Compliance Manager for z/OS version 4.2 Surveyor and Monitor data directly to Tivoli Asset Discovery for z/OS version 8.1 is not supported.

# Chapter 6. Collecting and importing data with IBM Tivoli Asset Discovery for z/OS

Tivoli Asset Discovery for z/OS includes programs that collect system and usage data, import, filter and match this data, update the Repository tables, and make the data available for review and query.

## Updating the Global Knowledge Base

IBM provides monthly updates to the Global Knowledge Base (GKB) so that you can keep your product inventory definitions up-to-date. You can also submit items to IBM support for inclusion in GKB updates.

### About this task

Updates to the GKB are available from the Fix Central website and you can register for notifications when updates are posted. Each update includes the following files:

- The `TADZ81KB.XMI` file contains a list of products that were added to the GKB since the last update. The file may also contain special processing instructions, such as running supplied SQL before doing any updates.
- The `GKBLVEL`*`yymmdd`*`.TXT` file contains instructions for applying the update.

### Procedure

1. From the Fix Central website, download the `TADZ81KB.XMI` file.
2. Upload the `TADZ81KB.XMI` file into a preallocated file on the mainframe with the attributes **`FB 80`**.
3. Issue the following TSO command to receive the file:
   ```
   RECEIVE INDATASET(TADZ81KB.XMI)
   ```
4. Enter `DA (`*`filename`*`)` when you are prompted for additional information.
5. In the GKB load job, HSISGKBL, update the SET INDSN= value with the name of the file that you received, and then submit the job.

## Collecting scanned libraries with the Inquisitor for z/OS

The Inquisitor is a program that scans and collects information about partitioned data set (PDS) and partitioned data set extended (PDSE) program libraries. The Inquisitor Import program takes the collected data as input to form the basis of your software inventory.

**Related tasks**:

"Importing Inquisitor data" on page 83
The Inquisitor Import reads data from Inquisitor scans, where the data is filtered and matched to products. The filtered, matched data is then copied to the Repository tables where it can be viewed and queried by the Analyzer reporting utility.

### Running the Inquisitor program

The HSISINQZ job in the JCLLIB library performs the Inquisitor collection. This job is generated from the HSISCUST post-installation customization job.

### About this task

The length of time it takes this job to run depends on the number of volumes and libraries to be scanned. Run this job during off-peak periods.

### Procedure

1. In the HSISINQZ job, check the values for the following parameters and change if necessary:
   - The **ALLMSG** parameter requests both DSNMSG and PGMMSG message logging.
   - The **PLX** parameter is set to Y (yes) if you plan to collect data for a sysplex and otherwise set to N (no).
     Review information about the **PLX** parameter before you set this option.
   - The **LLQ** parameter is set to `Z&SMF`. You can change this value if you want to generate data sets with unique names without changing the JCL library.

   These values are set when the HSISINQZ job is created.
2. Optional: In the program parameter string, you can specify a report message level and an override to the system identifier. Use commas to separate the various settings specified within the program parameter string.
3. Run the HSISINQZ job.

## PLX parameter of the Inquisitor program

The **PLX** parameter can reduce the time it takes to scan and process different SIDs that are completely shared and are, therefore, identical. When you set **PLX**=`Y`, the Inquisitor Import detects libraries that are mirrors or libraries that have not changed and quickly processes scans of these shared SIDs.

Plan your Repository to receive scans of system identifiers (SIDs) containing libraries that are unique in library name and volume, except when identically-named libraries are copies or are shared. If you have libraries that are identical in library name and volume name but are intended to have different content, place these libraries in different Repositories so that they can be processed separately.

If a library with the same library name and volume name is encountered in different SID scans, the Inquisitor Import considers the first instance that it encounters on the first SID to be the base. The Inquisitor Import treats any subsequent instances on different SIDs as mirrors.

The locations of all SIDs for a given library are recorded, but module discovery information is only calculated and stored when the library is encountered on its base SID. This approach ensures consistency in matching if the copies are not synchronized. The approach also saves processing time when SIDs are identical. If the Inquisitor Import encounters mirror libraries, it displays their names, current SIDs, and base SIDs in the log file, and reports their number at the end of the run.

If an SID is decommissioned and is no longer available for scanning, you can run the system deletion job to remove the SID and any libraries, modules, and products that are exclusively attached to the deleted SID. For shared libraries, only the record of the library that is attached to the specified SID is removed. The contents of the library are then attached to the subsequent SID in the list which then functions as the base SID.

If the **PLX=Y** option is specified during the Inquisitor run, the Inquisitor Import applies the results of the scan from the Inquisitor file to all SIDs that the were previously processed and share the same sysplex ID with the SID in the Inquisitor file. An existing library is processed on its base SID but is recorded as seen on all SIDs of the sysplex. A new library is processed on the current SID which becomes its base SID and is recorded as seen on all SIDs of the sysplex.

The Inquisitor Import requires that the Inquisitor file is more recent than any Inquisitor file that it previously processed for the same SID. If you specify the **PLX=Y** option during the Inquisitor scan, the Inquisitor file must be more recent than previously processed files of all SIDs of the sysplex.

The default value for the **PLX** parameter is N (no). If you intend to use the **PLX**=Y option to save scanning time, you must scan all SIDs at least once and present all the scans to the Inquisitor Import, so that it can determine how the different SIDs are shared.

When you specify the **PLX**=Y option, the Inquisitor Import processes the file in the following manner:

- Treats the content of all SIDs that share the sysplex ID with the currently-scanned SID as being identical to each other.
- Applies the scan results to all SIDs of the sysplex.

Because the Inquisitor Import process does not verify that all SIDs are identical, incorrect results can occur if the SIDs of the sysplex have different content. Use the **PLX**=Y option only if you are sure that all SIDs of the sysplex are identical in content at the time of the scan.

## Inquisitor program parameters and files

The Inquisitor program has mandatory and optional parameters that affect how data is collected. The program uses some mandatory files as well as some optional files.

*Table 9. Parameter settings for the Inquisitor*

| Parameter | Description |
|---|---|
| DSNMSG | Requests that messages relating to processed data sets, which might otherwise be suppressed, are to be logged in the SYSPRINT report. |
| PGMMSG | Requests that messages relating to processed programs, which might otherwise be suppressed, are to be logged in the SYSPRINT report. |
| ALLMSG | Requests both DSNMSG and PGMMSG message logging. |
| NOAPF | Specifies that the Inquisitor is to run in an environment which is not APF authorized. |
| NOHOST | Requests that the call to HSIPHOST to collect the TCPIP host name and IP address is bypassed. |
| SID= | The value is up to 4 characters long, and specifies the system identifier to be contained in the data output from the Inquisitor. If the SID identifier override is omitted, the system SMF identifier is used. The SID parameter setting is used when the SMF system identifier of a system is not unique. For example: `SID=SYS2` |
| PLX= | The parameter is used to identify if the Inquisitor data being collected is part of a SYSPLEX. The value is either Y or N.<br><br>If the PLX parameter is not used, the default value of N is created in the Inquisitor header record. |
| PLEXNAME= | The value is up to 8 characters long, and specifies the sysplex identifier to be contained in the data output from the Inquisitor. If the PLEXNAME identifier override is omitted, the actual sysplex name is used. The primary purpose of the PLEXNAME parameter is to provide a means for controlling the scope of sysplex-wide inventory updates. |

*Table 9. Parameter settings for the Inquisitor (continued)*

| Parameter | Description |
|---|---|
| LLQ= | This parameter is used to specify a suffix string made up of one or more data set name qualifiers to be appended to the data set name of the HSIPZIP and HSIPOUT data set. Its maximum length is 44 characters. It may contain both static and dynamic system symbols, and the user symbols &SMF. (SMF system identifier) and &SYSLPAR. (LPAR name) supplied by the Inquisitor. Use the LLQ setting when you need to create uniquely named data sets without changing the JCL. |

*Table 10. Files used by the Inquisitor*

| Filename | Description |
|---|---|
| SYSPRINT | A mandatory report file. |
| TAGREP | An optional report file that summarizes tag data collected by the Inquisitor. |
| SYSIN | A mandatory request input file. It processes fixed length, variable length, and undefined record formats. Records shorter than 72 bytes will be logically extended by the Inquisitor with blanks. |
| HSIPZIP | An optional output file that contains compressed Inquisitor data. It is written using a variable length record format. You must provide DCB information to ensure optimal use of DASD space. |
| HSIPOUT | An optional output file that contains uncompressed Inquisitor data. It is not specified in the packaged sample, as the use of HSIPZIP is preferred, due to its reduced space requirements. HSIPOUT also contains variable length records. The program supplies the appropriate LRECL. By default, system determined block size is used.<br><br>If you want to the direct the Inquisitor output to a compressible extended-format data set, then you should use the HSIPOUT file. The HSIPZIP file employs update-in-place processing, which prevents the use of DFSMS compression. |
| MCDS | An optional file that allocates the DFHSM MCDS data set, and is required if any requests contain the REMIGRATE or NOML2 operands. Further, if supplied for other requests, you can use it to avoid recalling data sets which are not load libraries. If the DFHSM MCDS is spread over more than one data set, use the DD names MCDS2, MCDS3, and MCDS4 consecutively. This allocates all the MCDS data sets in key range order. |
| ABRIN | An optional SYSIN file belonging to the FDRABRP utility program that is required if any requests contain the ABRMIG or ABRARC operands. It is primed by the Inquisitor during execution. For this reason, a single track VIO file is an ideal allocation. |
| ABRPRINT | An optional SYSPRINT file belonging to the FDRABRP utility program that is required if any requests contain the ABRMIG or ABRARC operands. It is an output-only file, and is not processed by the Inquisitor. |

# Inquisitor program command syntax

The Inquisitor program includes SYSIN commands and optional command operands.

## SYSIN commands

The Inquisitor program uses the SCANCMD, SCANDIR, and SCANPGM SYSIN commands that are described in the following table.

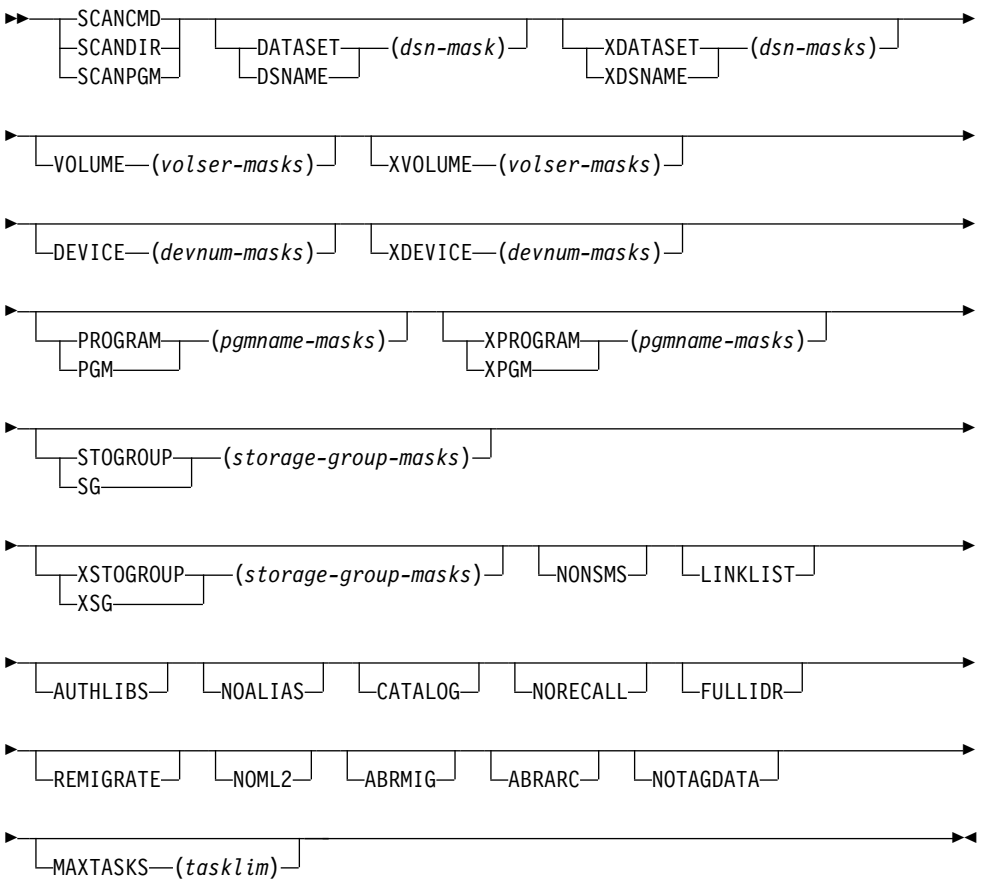
*Table 11. SYSIN commands used by the Inquisitor*

| Command | Description |
|---|---|
| SCANCMD | Allows command syntax and operand consistency to be checked by the Inquisitor without initiating an actual scan for program libraries. It performs a parse only operation, although output files are opened.<br><br>Error messages relating to syntax and operand errors are produced as usual. This verb is useful if you are formulating the best request combination when implementing on any given system. |
| SCANDIR | Collects data from program library directory entries. Contents of program members are not accessed.<br><br>Compared to SCANPGM, its reduced data collection allows it to run faster. Although all syntactically correct operands are allowed, some operands relating to data from member contents are ignored during processing. SCANDIR collects all of the information needed for automated software identification, and is the command of choice for a production environment. |
| SCANPGM | Collects all data collected by SCANDIR, and information from member contents. Such information relates to program structure and history.<br><br>Your IBM representative might request SCANPGM output data to assist with problem diagnosis and resolution. |
| SCANDEV | Collects information about the input and output (I/O) configuration of the z/OS system including online I/O devices, control units, and related channel path connectivity. The SCANDEV command has no operands. |

The Inquisitor can process multiple requests in a single program run. The output of these requests is contained in the same file.

This syntax diagram shows the SYSIN commands and their operands.

**Syntax diagram of library scan commands**




Operand defaults are:

```
DSNAME(*) VOLUME(*) DEVICE(*) PROGRAM(*)
```

All operands are optional. They are:

**DATASET Alias: DSNAME**

This operand specifies one or more 1 to 44 byte data set name masks. Only data sets with names matching any masks specified here are processed. Data sets with names not matching any masks specified here are not processed. Multiple masks must be separated by one or more delimiters. This operand can be specified more than once in a request, whereupon all masks specified in all occurrences of this operand are checked for selection matching. The precise treatment of asterisks in these masks is altered by the presence of the CATALOG keyword in the request. When CATALOG is specified, mask matching becomes qualifier aware and a single asterisk represents one, or part of, one qualifier only. When CATALOG is specified, use a double asterisk to specify any number of qualifiers. The data set name selection mask is the only mask affected by the CATALOG keyword. When the CATALOG keyword is present, exactly one DSNAME mask must be specified.

**XDATASET Alias: XDSNAME**

This operand specifies one or more 1 to 44 byte data set name masks. Data

sets with names matching any mask specified here are not processed. Multiple masks must be separated by one or more delimiters. This operand can be specified more than once in a request, whereupon all masks specified in all occurrences of this operand are checked for exclusion matching. If this operand is used, each mask must specify a subset of a DATASET mask.

**VOLUME**
This operand specifies one or more 1 to 6 byte volume serial number masks. Only volumes with serial numbers matching any mask specified here are processed. Volumes with serial numbers not matching any mask specified here, are not processed. Multiple masks must be separated by one or more delimiters. This operand can be specified more than once in a request, whereupon all masks specified in all occurrences of this operand are checked for selection matching. A volume serial number mask of six asterisks specifies the current IPL volume, which is ascertained during execution.

**XVOLUME**
This operand specifies one or more 1 to 6 byte volume serial number masks. Volumes with serial numbers matching any mask specified here are not processed. Multiple masks must be separated by one or more delimiters. This operand can be specified more than once in a request, whereupon all masks specified in all occurrences of this operand are checked for exclusion matching. If this operand is used, each mask must specify a subset of a VOLUME mask. A volume serial number mask of six asterisks specifies the current IPL volume, which is ascertained during execution.

**DEVICE**
This operand specifies one or more 1 to 4 byte device number masks. Only volumes with device numbers matching any mask specified here are processed. Volumes with device numbers not matching any mask specified here, are not processed. Multiple masks must be separated by one or more delimiters. This operand can be specified more than once in a request, whereupon all masks specified in all occurrences of this operand are checked for selection matching. Standard character string mask matching is used. The use of characters which are not hexadecimal digits will not be detected by the program.

**XDEVICE**
This operand specifies one or more 1 to 4 byte device number masks. Volumes with device numbers matching any mask specified here are not processed. Multiple masks must be separated by one or more delimiters. This operand can be specified more than once in a request, whereupon all masks specified in all occurrences of this operand are checked for exclusion matching. If this operand is used, each mask must specify a subset of a DEVICE mask. Standard character string mask matching is used. The use of characters which are not hexadecimal digits will not be detected by the program.

**PROGRAM Alias: PGM**
This operand specifies one or more 1 to 8 byte program name masks. Only programs with names matching any mask specified here are processed. Programs with names not matching any mask specified here, are not processed. Multiple masks must be separated by one or more delimiters.

This operand can be specified more than once in a request, whereupon all masks specified in all occurrences of this operand are checked for selection matching.

**XPROGRAM Alias: XPGM**
This operand specifies one or more 1 to 8 byte program name masks. Programs with names matching any mask specified here are not processed. Multiple masks must be separated by one or more delimiters. This operand can be specified more than once in a request, whereupon all masks specified in all occurrences of this operand are checked for exclusion matching. If this operand is used, each mask must specify a subset of a PROGRAM mask.

**STOGROUP Alias: SG**
This operand specifies one or more 1 to 8 byte storage group name masks. SMS-managed volumes in a storage group with a name matching any mask specified here are processed. SMS-managed volumes in a storage group with a name that does not match any mask specified here, are not processed. Multiple masks must be separated by one or more delimiters. This operand can be specified more than once in a request, whereupon all masks specified in all occurrences of this operand are checked for selection matching. Volumes which are not SMS-managed are not processed unless the NONSMS keyword operand is specified.

**XSTOGROUP Alias: XSG**
This operand specifies one or more 1 to 8 byte storage group name masks. SMS-managed volumes in a storage group with a name matching any mask specified here are not processed. Multiple masks must be separated by one or more delimiters. This operand can be specified more than once in a request, whereupon all masks specified in all occurrences of this operand are checked for exclusion matching. If both this mask and a STOGROUP mask are used, then each mask must specify a subset of a STOGROUP mask.

**NONSMS**
This keyword operand specifies that volumes which are not SMS-managed are eligible for processing. The presence of this operand means that SMS-managed volumes are not processed unless the STOGROUP operand was used to supply a storage group name mask.

**LINKLIST**
This keyword operand specifies that all link list data sets are to be unconditionally included for processing.

**AUTHLIBS**
This keyword operand specifies that all APF authorized data sets are to be unconditionally included for processing.

**NOALIAS**
This keyword operand specifies that any program member marked as an alias is to be excluded from processing.

**CATALOG**
This keyword operand specifies that data sets to be processed are located from a catalog search rather than VTOC searches. Data set alias names are not processed. The Inquisitor triggers and waits for a RECALL operation for each migrated data set which passes data set name mask processing, unless NORECALL is also specified.

**NORECALL**
This keyword specifies that migrated data sets are not to be recalled and are excluded from processing. This operand only has effect when the CATALOG operand is also specified. Data sets with a catalog entry indicating a volume serial number of MIGRAT, or ARCIVE, are deemed to be migrated.

**FULLIDR**
This keyword operand specifies that a full scan of CESD and IDR records is to be performed, even when a module would not have been selected for such processing. Depending upon the exact nature of the request being run, this operand can significantly elongate the elapsed time of Inquisitor runtime.

This operand is ignored for a SCANDIR request.

**REMIGRATE**
This keyword operand specifies that when a data set which had to be recalled has been processed, DFHSM is requested to migrate the data set again asynchronously. Migrated data sets can only be processed when the CATALOG operand is also specified. Only data sets with a catalog entry indicating a volume of MIGRAT are remigrated.

The presence of this operand requires that the MCDS file is allocated to the DFHSM MCDS. Access to the MCDS allows the Inquisitor to avoid recalls for data sets which are not partitioned, do not have an undefined record format, and do not have a block size of at least 1024.

**NOML2**
This keyword operand specifies that data sets migrated to level two are not to be recalled and are excluded from processing. Migrated data sets can only be processed when the CATALOG operand is also specified. Only data sets with a catalog entry indicating a volume of MIGRAT are checked for level two status.

The presence of this operand requires that the MCDS file is allocated to the DFHSM MCDS. Access to the MCDS allows the Inquisitor to avoid recalls for data sets which are not partitioned, do not have an undefined record format, and do not have a block size of at least 1024.

**ABRMIG**
This keyword operand indicates that when a catalog entry with a volume of MIGRAT is encountered, the FDRABR product is to be invoked to determine whether a recallable archived copy of the data sets is available or not. If it is, then the data set is processed. If not, then the data set is not processed.

The NORECALL operand takes precedence over this operand.

The effect of ABRMIG is not affected by the ABRARC operand.

The presence of this operand requires that the ABRIN and ABRPRINT files are allocated.

**ABRARC**
This keyword indicates that, when a cataloged data set cannot be found on the volume, the FDRABR product is to be invoked in order to determine whether a recallable archived copy of the data set is available. If it is, then the data set is processed. If not, the data set is not processed.

The NORECALL operand takes precedence over this operand.

The effect of ABRARC is not affected by the ABRMIG operand.

The presence of this operand requires that the ABRIN and ABRPRINT files are allocated.

**NOTAGDATA**
This keyword indicates that data written to program libraries by the Product Tagger is not to be collected and written to the Inquisitor output file. Use this operand only when you do not want to update the Local Knowledge Base during the import process with the latest Tagger data that could be found by the Inquisitor.

**MAXTASKS**
This operand specifies the maximum number of VTOC-scanning subtasks to be activated by the Inquisitor. These subtasks reduce the elapsed time of an Inquisitor scan by enabling the concurrent processing of multiple volumes. Reducing the number of subtasks reduces the demand on storage in the region and does not impact performance unless the main task has to wait longer for VTOC scan results. The operand value is a single decimal number in the 1 to 200 range. The default value is 10. The actual number of subtasks used does not exceed the number of volumes to be scanned. VTOC-scanning subtasks are not used for CATALOG requests.

## SYSIN syntax rules for the Inquisitor

Syntax rules are as follows:

- Only the first 72 bytes of an input record are ever scanned.
- Short records are extended to 72 bytes with blanks.
- Blanks and commas are equivalent.
- Subparameters of value operands are specified in parentheses.
- A continuation to the next record is requested by a plus or a hyphen when it follows a delimiter, or is at the start of a record.
- A continuation cannot be requested in the middle of a word or value.
- The part of the record following a continuation character is ignored and can be used for comments.
- Records beginning with an asterisk are comment records.
- Records containing only blanks or commas are comment records.
- Comment records are ignored by syntax parsing logic, and do not alter continuation status.
- TSO conventions apply to abbreviations. That is, operands can be abbreviated to the minimum unambiguous length. Verbs cannot be abbreviated.
- If the input record contains an ampersand, the system symbol substitution routine ASASYMBM is called to perform symbol substitution processing.
- All input requests are parsed and stored before the first request is processed.
- If a syntax error is encountered, no requests are processed. This is to reduce the instance of incorrect or unproductive requests triggering lengthy DASD subsystem scans. The error is in the last record echoed in SYSPRINT.
- Value masks are character strings which are compared to data found at run time. Comparison is performed one byte at a time, from left to right. For a match, the characters must compare equal, unless a generic mask character is found.
- System static symbols, system dynamic symbols, and &SMF (SMF system identifier) and &SYSLPAR (LPAR name), can be used to construct value masks. &SYSLPAR may resolve to a null string if z/OS is running in a virtual machine.

- Valid generic mask characters are a percent (%), to flag a match for any single character, and an asterisk (*), to flag a match for any character string segment of zero or greater length.

## Inquisitor examples

These examples show some possible scenarios where you can customize the scope and type of processing when you run the Inquisitor program.

### Example 1

These three statements are equivalent, and request data collection for all programs on all online DASD volumes.

```
SCANDIR
SCANDIR DA(*) PGM(*)
SCANDIR VOL(*) DS(*)
```

### Example 2

To scan all SMS-managed volumes except volumes in storage group SGWORK use:

```
SCANDIR STOGROUP(*) XSTOGROUP(SGWORK)
```

### Example 3

To scan all volumes except volumes in storage groups with names beginning with SGW use:

```
SCANDIR XSTOGROUP(SGW*) NONSMS
```

### Example 4

To scan all volumes with serial numbers beginning with TSO and WRK, these two requests are used in a single program run:

```
SCANDIR VOLUME(TSO*)
SCANDIR VOLUME(WRK*)
```

### Example 5

To scan all volumes except those with serial numbers beginning with TSO and WRK use:

```
SCANDIR XVOLUME(TSO* WRK*)
```

### Example 6

To scan all volumes with serial numbers beginning with USR which are also in SMS storage groups with names beginning with SG for programs with names beginning with UTIL, use: .

```
SCANDIR VOLUME(USR*) STOGROUP(SG*) PROGRAM(UTIL*)
```

### Example 7

To scan all data sets with high level qualifiers of SYS1, SYS2, SYS3, except z/OS distribution libraries, use:

```
SCANDIR DSNAME(SYS%.*) XDSNAME(SYS1.A*)
```

## Example 8

To restrict the data in the previous example to cataloged data sets, use:

```
SCANDIR DSNAME(SYS%.**) XDSNAME(SYS1.A*) CATALOG
```

**Note:** Note the extra asterisk in the data set name selection mask. Without this, only data set names with two qualifiers are selected. Data set name exclusion processing is not changed by the CATALOG operand.

## Example 9

To scan the current IPL volume, and any other link, list, and APF authorized libraries use:

```
SCANDIR VOLUME(******) LINKLIST AUTHLIBS
```

## Example 10

To scan the single cataloged data set SYS1.PPLIB without a lengthy DASD subsystem scan use:

```
SCANDIR DATASET(SYS1.PPLIB) CATALOG
```

## Example 11

To scan all cataloged SYS1 and SYS2 data sets use (a) two requests in a single program run, or (b) a single request. The two approaches exhibit similar resource consumption:

```
SCANDIR DA(SYS1.**) CAT
SCANDIR DA(SYS2.**) CAT

SCANDIR DS(SYS%.**) CAT XDSN(SYS3.*,SYS4.*,SYSA.*)
```

The XDSN values are coded as shown under the assumption that SYS1, SYS2, SYS3, SYS4 and SYSA are the only 4 character high-level qualifiers beginning with SYS on the system being scanned.

**Note:** SCANDIR DS(SYS1.**,SYS2.**) CAT is not allowed.

## Example 12

These examples are all equivalent. They scan the entire DASD subsystem for all data sets with a first qualifier of SYS1 or SYS2, excluding those with a second qualifier beginning with A.

(a)

```
SCANDIR DA(SYS1.*,SYS2.*) XDA(SYS1.A*,SYS2.A*)
```

(b)

```
SCANDIR DA(SYS1.*    +
SYS2.*)   +
XDA(SYS1.A*   +
SYS2.A*)
```

(c)

```
SCANDIR DA(SYS1.*)   +
DA(SYS2.*)   +
XDA(SYS1.A*)  +
XDA(SYS2.A*)
```

(d)

```
SCANDIR DA(SYS1.*)  XDA(SYS1.A*)  +
DA(SYS2.*)  XDA(SYS2.A*)
```

(e)

```
SCANDIR DA(SYS1.*)  XDSN(SYS1.A* SYS2.A*)  DS(SYS2.*)
```

## Designing Inquisitor requests

When constructing statements for the Inquisitor SYSIN file, try to combine all selection and exclusion criteria to form a single SCANDIR request. A single Inquisitor request will not scan a VTOC or a library more than once.

It can be difficult to formulate a system scan into a single CATALOG request, meaning that when the CATALOG operand is used, multiple requests are coded. Ensure that no data set will be scanned by more than one SCANDIR CATALOG request by excluding as many data set name patterns from each request as necessary. Data set name exclusions may not be necessary if all CATALOG search selection masks represent disjoint parts of the name space.

The example shown here uses the XDA operand to prevent SYS1.LINKLIB from being scanned more than once:

```
SCANDIR DA(SYS1.**) CATALOG
SCANDIR DA(SYS%.LINKLIB) XDA(SYS1.LINKLIB) CATALOG
```

As well as using the selection and exclusion facilities to ensure completeness, they can also be used to improve performance and efficiency by excluding DASD volumes which do not contain program libraries. Although a volume with no program libraries can be scanned quickly, processing duration might be reduced if such volumes can be excluded from an Inquisitor scan.

For example, volumes that only contain databases, or temporary data sets, do not have any files suitable for Inquisitor processing, but the VTOCs of those volumes are still read unless excluded by the appropriate selection criteria.

To illustrate this further, consider a system with these DASD subsystem usage elements:

**System platform**
Non-SMS and storage group SYSTEM.

**Work pool**
Storage group TEMP containing temporary and short-lived (two days) permanent files.

**TSO** Storage groups TSOONE and TSOTWO.

**Non-DB application**
Non-SMS and storage groups BATCH1 and BATCH2.

**Databases**
Non-SMS volumes DBA001 to DBA099 and SMS storage groups DB01, DB02, and DB03.

The scanning of this configuration is to be carried out with the following assumptions:

- No need for data from libraries that do not exist for more than two days.
- No program libraries on database volumes.
- Applications combine their program libraries and non-database files.
- TSO users can have program libraries.
- Management requires information regarding all potentially permanent executable software.

To acquire Inquisitor data from all useful sources without processing volumes more than once, and without processing irrelevant volumes, you can specify multiple requests in a single Inquisitor run. For example:

```
SCANDIR SG(SYSTEM)
SCANDIR SG(TSO*)
SCANDIR SG(BATCH*)
SCANDIR NONSMS XVOL(DB*)
```

This can be consolidated into a single request giving the same result. For example:

```
SCANDIR SG(SYSTEM TSO* BATCH*) NONSMS XVOL (DB*)
```

## Scanning migrated libraries

The Inquisitor locates load libraries by either scanning the VTOC of online volumes, or by searching the system catalog (CATALOG) for relevant data sets. When you use the Catalog Search Interface, you can return data sets for migrated libraries. VTOC scans do not find migrated data sets.

When the keyword CATALOG is specified in a request statement, the Inquisitor passes the data set name selection mask to the Catalog Search Interface (CSI) to search for the catalog entries. It is possible that one or more of the catalog entries returned by the CSI are for a data set that has been migrated. In contrast, VTOC scans do not find migrated data sets.

Inquisitor processing of migrated data sets found by the CSI involves dynamic allocation which then triggers the recall of the data set. Recalls increase Inquisitor processing time. The processing leaves the data set in a recalled status.

The Inquisitor looks at the volume serial number in the catalog entry to determine if a data set is migrated or not. A data set is considered to have been migrated if its catalog entry indicates a volume serial number of either MIGRAT or ARCIVE.

To suppress the processing of all migrated data sets, specify the NORECALL keyword on each Inquisitor request.

### Integration with DFHSM

If you are using the MCDS file allocation, and a data set cataloged on volume MIGRAT is encountered, the Inquisitor can read the data set record from the DFHSM Migration Control Data Set (MCDS) to verify that the data has the attributes of a program library. If the MCDS record is not found, the data set is ignored and processing is bypassed, avoiding a DFHSM error condition. If the data set does not have partitioned organization, an undefined record format, and a block size of at least 1024, the Inquisitor ignores the data set, avoiding the recall of many data sets which are not program libraries.

For systems with DFHSM space management functions, you can use the request keywords NOML2 and REMIG. The MCDS file allocation is a prerequisite for using the following keywords:

**NOML2**
Specifies that data sets migrated to level 2 are excluded from the scan.

**REMIG**
Specifies that after a recalled data set is processed by the Inquisitor, the Inquisitor requests DFHSM to remigrate the data set. The Inquisitor does not wait for the migration to complete, but begins to process the next data set immediately after making the request to DFHSM. Migration level 2 is never specified by the Inquisitor for the migration, even if the data set was recalled from ML2. (However, it might be selected by DFHSM as a result of SMS management class settings.)

**Note:**
Any combination of REMIGRATE, NOML2, and NORECALL is valid. Specifying NORECALL means NOML2 and REMIGRATE have no effect.

In the case where you want to scan all relevant migrated program libraries and do not want any such libraries explicitly remigrated afterward, you would not code any of the NORECALL, NOML2 and REMIGRATE keywords. In this instance, the MCDS file allocation, though optional, can still be used to great advantage.

## Scanning generation data sets

Inquisitor CSI requests are limited to NONVSAM type A catalog entries. Generation data sets (which are members of a generation data group) are not scanned by Inquisitor CATALOG requests but can be processed by Inquisitor VTOC scans. Consider excluding generation data sets if you back up program libraries using generation data sets.

To exclude generation data sets from a VTOC scan request, specify a suitable data set exclusion mask, for example:

```
XDA(*.G%%%%V00)
```

## Collecting information about the I/O configuration

The Inquisitor can scan the input and output (I/O) configuration of a z/OS system to collect information about the use of hardware assets such as storage devices. The SCANDEV command is used to request such a scan.

When you run the SCANDEV command, two additional types of records are generated which describe device groups and channel paths. Consideration of I/O devices is limited to those devices which are online at the time of the scan.

A device group is a contiguous block of device numbers, not including offline devices, where the device type, control unit type and serial number, and online channel path connectivity is the same. The channel path type is collected for each channel path used to connect to an online I/O device.

A channel that does not provide an online path to any online device is not reported even if the channel is configured online to the z/OS system.

# Collecting UNIX files with the Inquisitor for z/OS UNIX

The Inquisitor for z/OS UNIX is a program that collects information about executable software existing in HFS and zFS data sets currently mounted and accessible to z/OS UNIX. The Inquisitor Import program takes the collected data as input to form the basis of your z/OS UNIX software inventory.

## Inquisitor for z/OS UNIX overview

The Inquisitor for z/OS UNIX produces a set of record types which are different from those produced by the Inquisitor for z/OS. However, both programs collect the same types of information about installed software.

The Inquisitor for z/OS UNIX processes the hierarchical file system (HFS) root directory, as well as all subdirectories. For this reason, the program must run with a UID that allows access to all directories and programs to be examined. If the Inquisitor for z/OS UNIX does not have permission to access a directory, then no information is collected from that directory, or any of its subdirectories.

The HSIXROOT file is used to nominate one or more directories to be considered root directories. When specified, only the nominated directories and their subdirectories are processed. This facility is useful when only a subset of the file hierarchy needs to be scanned.

The HSIXOMIT file is used to nominate one or more directories which are to be omitted or excluded from the scan, together with all of their subdirectories. This facility can be used to reduce resource consumption by preventing parts of the UNIX file hierarchy known not to have any executable software from being scanned.

## Running the Inquisitor for z/OS UNIX program

The HSISINQU job in the JCLLIB library performs the Inquisitor for z/OS UNIX collection. This job is generated from the HSISCUST post-installation customization job.

### About this task

Run-time for this job depends on the size and complexity of the UNIX directory structure to be scanned. Run this job during off-peak periods.

### Procedure

1. In the HSISINQU job, check the values for the following parameter and change if necessary:
   - The **LLQ** parameter is set to `Z&SMF`. You can change this value if you want to generate data sets with unique names without changing the JCL library.

   This value is set when the HSISINQU job is created.
2. Optional: In the program parameter string, you can specify a report message level, an override to the system identifier, and whether you want compressed or uncompressed output. Use commas to separate the various settings specified within the program parameter string.
3. Run the HSISINQU job.

## Inquisitor for z/OS UNIX program parameters and files

The Inquisitor for z/OS UNIX program has mandatory and optional parameters that affect how data is collected. The program uses some mandatory files as well as some optional files.

*Table 12. Parameter settings for Inquisitor for z/OS UNIX*

| Parameter | Description |
|---|---|
| PTHMSG | Requests that a message is written to HSIXMSG each time a directory is opened or closed. |
| PGMMSG | Requests that a message is written to HSIXMSG each time an executable file is processed. |
| ALLMSG | Requests both PTHMSG and PGMMSG message logging. |
| NOHOST | Requests that the call to HSIPHOST to collect the TCPIP host name and IP address is bypassed. |
| SID= | The value is up to 4 characters long and specifies the system identifier to be contained in the data output from the Inquisitor. If the SID identifier override is omitted, the system SMF identifier is used. The SID parameter setting is used when the SMF system identifier of a system is not unique. For example: `SID=SYS2` |
| OUT= | Specifies output file usage. The default value is Z.<br>• A value of Z requests zipped output to HSIXZIP.<br>• A value of T requests text output to HSIXOUT.<br>• A value of B requests output to both HSIXZIP and HSIXOUT files. |
| PLEXNAME= | The value is up to 8 characters long, and specifies the sysplex name to be contained in the data output from the Inquisitor. If the PLEXNAME identifier override is omitted, the actual sysplex name is used. The primary purpose of the PLEXNAME parameter is to provide a means for controlling the scope of sysplex-wide inventory updates. |
| LLQ= | This parameter is used to specify a suffix string made up of one or more data set name qualifiers to be appended to the data set name of the HSIXZIP and HSIXOUT data set. Its maximum length is 44 characters. It may contain both static and dynamic system symbols, and the user symbols &SMF. (SMF system identifier) and &SYSLPAR. (LPAR name) supplied by the Inquisitor. Use the LLQ setting when you need to create uniquely named data sets without changing the JCL. |

*Table 13. Files used by the Inquisitor for z/OS UNIX*

| Filename | Description |
|---|---|
| HISIXMSG | Report file used by HSIXINQ. |
| SYSPRINT | Used by Language Environment (LE), which is required to be in the standard module search path, and by IDCAMS when LLQ= is specified. |
| SYSOUT | Used by Language Environment (LE), which is required to be in the standard module search path. |
| HSIXZIP | An optional output file that contains compressed Inquisitor for z/OS data. It is written using a variable length record format. You need to provide DCB information to ensure optimal use of DASD space.<br><br>The HSIXZIP file must never undergo any translation when being transferred, whatever the architecture of the target system. That is, only BINARY transfers are to be used to transport the file. |
| HSIXOUT | An optional output file that contains uncompressed Inquisitor for z/OS UNIX data. It is not specified in the packaged sample, as the use of HSIPZIP is preferred, due to its reduced space requirements. HSIXOUT also contains variable length records. The program supplies the appropriate LRECL. By default, system determined block size is used.<br><br>If you want to the direct the Inquisitor for z/OS UNIX output to a compressible extended-format data set, then you should use the HSIPOUT file. The HSIPZIP file employs update-in-place processing, which prevents the use of DFSMS compression. |
| HSIXROOT | An optional file which can contain one or more records; each of which specifies a directory path to be considered as a root directory to be processed. If HSIXROOT is not allocated or empty, then a forward slash (/) is considered to be the only root directory to be processed. |

*Table 13. Files used by the Inquisitor for z/OS UNIX (continued)*

| Filename | Description |
|---|---|
| HSIXOMIT | An optional file which can contain one or more records; each of which specifies a directory path which is to be omitted from the scan. Root directories cannot be omitted. |

The HSIXROOT and HSIXOMIT files have the following characteristics and attributes in common:

- There is no requirement for the file to be allocated.
- The file might be empty or allocated to DUMMY.
- The file might contain fixed length or variable length records.
- Records must not contain more than 1024 bytes of data.
- Blank records are deemed to be comments and discarded.
- Leading and trailing blanks are discarded when the directory name is extracted.
- Records with an asterisk as the first nonblank are deemed to be comments and discarded.
- If the directory path does not end in a slash, then one is appended.

## Security considerations

If you want to collect all relevant z/OS UNIX data, you must have access to all UNIX directories, including the root directory. This access ensures that all z/OS UNIX data is collected.

To allow the Inquisitor unrestricted read access to all z/OS UNIX files, consider using the UNIXPRIV RACF Resource Class, which alleviates the need for UID(0).

The following sample definition can be used by your Security Administrator to define, permit, activate, and RACLIST the RACF UNIXPRIV Class:

```
RDEL UNIXPRIV SUPERUSER.FILESYS.**
RDEF UNIXPRIV SUPERUSER.FILESYS.** UACC(NONE) OWNER(IBMUSER)
PE  SUPERUSER.FILESYS.** CLASS(UNIXPRIV) RESET
PE  SUPERUSER.FILESYS.** CLASS(UNIXPRIV) ID(USERONE) ACCESS(READ)
SETR CLASSACT(UNIXPRIV)
SETR RACLIST(UNIXPRIV)
SETR RACLIST(UNIXPRIV) REFR
```

# Collecting usage data with the Usage Monitor

The Usage Monitor is a server address space that runs as a started task. Work is queued to the Usage Monitor from all address spaces where programs are used. The Usage Monitor moves captured program usage data into the collection repository and periodically writes the accumulated data to a sequential file. The Usage Monitor runs APF authorized and is non swappable.

**Related concepts**:

"Importing usage data" on page 86
The Usage import job imports data generated by the Usage Monitor and aggregates usage data for discovered or identified modules in the Repository tables in the database.

## Setting up the Usage Monitor

The Usage Monitor uses the HSISUMON job in the JCLLIB library. This job is generated from the HSISCUST post-installation customization job. This job will call the HSIJMON procedure from the JCLLIB library.

Parameters for the Usage Monitor job are HSISMNPM member in the PARMLIB library.

## Files used by the Usage Monitor

The Usage Monitor has three product-specific files. They are:

**HSIZIN**
A sequential file consisting of fixed length 80 byte records. It contains initial commands which are run before data collection becomes active. It must contain the data set prefix to be used for dynamically created output files. The prefix can be changed later by an operator MODIFY command.

HSIZIN is opened, read, and closed during initialization processing. Do not specify FREE=CLOSE in the JCL for HSIZIN, or refresh processing is not possible.

Dataset &HSIINST..&DB..UM.HLQIDS contains the high-level qualifier listing for products and is populated by the IQ Import job (HSISIQIM). To minimize the number of records to be created by the Usage Monitor, only usage events that match the list of products in this dataset are generated. To activate this facility, in //HSIZIN, uncomment dataset &HSIINST..&DB..UM.HLQIDS as described in PROC HSIJMON.

**HSIZMSG**
A log file which contains the initial commands issued, and which indicates their degree of success. It also contains regular status reports, refresh reports (when appropriate), and a termination report. It consists of fixed length 121 byte records.

**SYSOUT**
A report file used by the SORT program.

Output files containing program usage data are dynamically allocated by the Usage Monitor. The data set name prefix, the allocation unit, and the primary and secondary space allocation quantities (in tracks), need to be customized for the target system. This is done in the PARMLIB member HSISMNPM.

## Using exclusion masks to reduce data

The data from a significant number of program usage events does not contribute meaningfully to the task of managing the software inventory. To reduce the processing of this unnecessary data, two mechanisms which allow some data to be excluded from collection have been provided. They are exclusion masking based on program name, and exclusion masking based on data set name.

## Filtering by program name

A program name exclusion table exists which contains program name masks. When a program usage event is detected by the Usage Monitor, the program name is checked against entries in the program name exclusion table. When a match is found, the usage event data is discarded. Program name exclusion filtering occurs before the data set name of the program library is determined by the Usage Monitor which makes it more efficient than data set name filtering.

Each table entry contains a program name comparison string up to 8 bytes long. The string is either an 8 byte program name, or a shorter program name prefix. When entering these strings with the EXC command, a prefix is denoted by using an asterisk as the last character.

The program name exclusion table resides in key zero common storage, and its size is always a multiple of 4,096 bytes. The minimum table size can house up to 253 entries, and the table size increases dynamically, as required. The default program name exclusion table contains entries to exclude data pertaining to the usage of many programs which are part of the operating system.

In order to add, reset, remove, or display the entries to the table, use these commands:

**EXC** To add entries to the program name exclusion table, or to reset the table to its default contents.

**DEL** To remove some, or all, entries from the table.

**D-X** To display the current contents of the table.

Unlike masks added by the EXC command, default program name exclusion masks do not exclude job step program usage events. So, for example, the IEF* default exclusion mask excludes dynamic calls and loads of program IEFBR14, but usages where IEFBR14 is invoked by JCL are not excluded by this mask.

## Filter by data set name

After the Usage Monitor has ascertained the name of the data set from which a used program is fetched, it is used to decide if the usage data is retained for collection or discarded. To perform this process, three lists of data set name masks are scanned; the first is the default data set name exclusion list, the second is the dynamic data set name inclusion list, and the third is the dynamic data set name exclusion list.

The default data set name exclusion list is built during Usage Monitor initialization, and consists of the SCEERUN library, the SCEERUN2 library, SYS1.CMDLIB (containing TSO commands) and SYS1.CSSLIB (containing callable services modules). The data set names of the SCEERUN and SCEERUN2 Language Environment libraries are determined by searching the link list for specific LE modules. You can use the XDD command to deactivate any of these default exclusion entries. You can use the XDS(*DFLT*) command to reactivate the default data set exclude list without affecting the status of masks in other lists.

The other two lists are constructed from commands you specify either in the HSIZIN file or dynamically via the system MODIFY command.

To avoid excessive storage and processor resource consumption, it is preferable to keep the number of elements in each list to a minimum. This is achieved by using generic masks to cover many data set names. The inclusion mask list is provided so that specific exceptions to broad exclusion rules can be specified. If you do not supply any data set name exclusion masks, the inclusion list does not affect data collection, but can still be used as a convenient way to collect relative usage statistics from the regular Usage Monitor status reports.

Data set name filtering occurs in the following sequence:

1. Excludes usage if the data set name matches a default exclusion mask, otherwise proceeds to step 2.
2. Include usage if the data set name matches a mask supplied by an IDS command, otherwise proceeds to step 3.
3. Excludes usage if the data set name matches a mask supplied by an XDS command, otherwise proceeds to step 4.
4. Includes usage if the data set name does not match any of the masks.

Data set mask elements reside in key zero common storage. Each element occupies 56 bytes, and contains a data set name mask up to 44 bytes in length. You can use the percent sign as a wildcard to match a single character. You can use a trailing asterisk to match the rest of the data set name.

In order to add, reset, remove, or display the entries to the tables, use these commands:

**XDS** To add a data set name mask to the exclusion list.

**IDS** To add a data set name mask to the inclusion list.

**XDD** To deactivate a data set name exclusion mask.

**IDD** To deactivate a data set name inclusion mask.

**D-D** To display all active data set name masks.

Both of the non-default lists have no elements until an XDS or IDS command is processed. Storage is dynamically acquired for each element as required. To ensure system integrity, XDD and IDD commands do not cause the storage of a deactivated element to be freed, but mark the element as inactive. When a deactivated mask is reactivated, the existing entry is marked as active without the further acquisition of storage.

When the Usage Monitor address space first initializes, all elements of lists that remain in storage from a previous run are freed before the processing of initial commands and the commencement of data collection.

There is no requirement to use either data set name mask list at any stage.

## Filtering by UNIX path name

If the mask value specified in an IDS, XDS, IDD or XDD command contains at least one slash, the value is deemed to be a UNIX path name mask and not a data set name mask. During processing, multiple consecutive slashes are reduced to a single slash.

UNIX path name masks entered via IDS and XDS commands are compared to the path names specified by applications at run time and may not correspond to the path names against which usage is attributed. The main cause for this difference in path names is the use of symbolic links. The Usage Monitor writer task converts path names with symbolic links to real path names in order to match inventory discovered by the Inquisitor.

Do not use an IDD or XDD command that specifies a UNIX path name mask because the only use for such a command is to dynamically delete a UNIX path name mask. Most UNIX path name masks contain lower case alphabetics. The system MODIFY command interface usually changes lower case characters to upper case which prevents the mask matching the relevant active mask. To delete a

UNIX path name mask you must either recycle the Usage Monitor or use the REF command to refresh the settings from the HSIZIN file. In either case, all UNIX path name masks are deactivated and the necessary change is to remove the IDS or XDS command that you want to deactivate from the HSIZIN file.

Similarly, because of the prevalence of lower case alphabetics in UNIX path names, you only specify IDS and XDS commands with path name masks as HSIZIN file input rather than via the MODIFY system command interface.

The length limit of 44 characters also applies to UNIX path name masks.

## Starting and stopping the Usage Monitor

A Usage Monitor member named HSIJMON is provided in SHSIPROC. If you want to start HSIJMON as a started task, copy the customized member from the JCLLIB to an authorized PROCLIB.

### Procedure

1. To start the Usage Monitor in normal mode, enter the following command:
   ```
   S HSIJMON
   ```
2. To fully stop the Usage Monitor, enter one of the following commands:
   ```
   P HSIJMON
   F HSIJMON,STOP
   F HSIJMON,END
   ```
   These commands cause the Usage Monitor to stop data collection, attach a writer task to process the existing data in the collection repository, wait for the writer task to output the data, and then terminate.
3. To perform an immediate termination, enter the following command:
   ```
   F HSIJMON,CAN
   ```
   This command causes the server address space to stop data collection, detaches any running writer task which renders the output data in the data set unusable, deletes the current collection repository without writing out its contents, and terminates. If you use the z/OS system command CANCEL to stop the Usage Monitor, its collection repository remains in storage. To clear the collection repository from storage, you must restart the Usage Monitor.

## Refresh processing for the Usage Monitor

The Usage Monitor includes commands that you can issue dynamically to alter processing but that are active only for the duration of the current Usage Monitor session. To implement a change to both the running Usage Monitor and to the initialization commands for starting subsequent Usage Monitor sessions, you can use the refresh facility.

Refresh processing involves the execution of the command stream placed in the HSIZIN file, without the requirement of stopping and restarting the Usage Monitor. As a result, refresh processing can verify the validity of the initialization command stream so that changes are made and tested dynamically. This ensures that future Usage Monitor sessions do not encounter initialization command stream errors.

Some commands set a switch for logic control, or set a numeric value to be used during processing. These commands specify the values to be used in the future. Other commands pertaining to inclusion and exclusion masking add a mask to, or remove a mask from, the active mask list, so are part of an accumulation of commands which specify future processing.

Consider the example where several exclusion masks are active, and a change to deactivate one of the masks is required. A command to deactivate the mask might be issued dynamically, but if this change is to be made permanent, then the HSIZIN file needs to be updated. The alternative is to remove the command setting the exclusion from the HSIZIN file, and to then issue the Usage Monitor REF command to initiate a refresh.

Before the first HSIZIN command is run during refresh processing, the program mask exclusion list is set to the default list. Further, all data set name exclusion masks are deactivated, and all data set name inclusion masks are deactivated. This order of deactivation ensures that there is no loss of data that would otherwise be collected. However, there is the possibility that data which would have been excluded is collected during the short window between the reset of the mask lists and the processing of the HSIZIN commands.

The response to each command in the HSIZIN file is written to the HSIZMSG file. A summary WTO message, indicating whether any errors are found or not, is issued after refresh processing has finished.

Stopping the Usage Monitor and restarting it, produces the same active exclusion masks as a refresh. It also produces a data collection outage. For more information, see the REF command in the next topic for a list of the processes performed during a refresh operation.

## Usage Monitor commands

The Usage Monitor commands are passed to the Usage Monitor from the HSIZIN input file, or by an operator MODIFY command.

The syntax rules are as follows:

- All commands are three characters long.
- Operands or subparameters are specified in parentheses.
- Multiple subparameters are separated by commas.
- The command must not contain any embedded blanks.
- Commands must start in column one.

To record the settings the Usage Monitor is using, place the display commands at the end of the HSIZIN file.

Details of each command follow.

### CAP - Set hardware capacity collection status

CAP is used to specify if the Usage Monitor is to produce records containing information about the hardware capacity of the system. Collecting this information is important when hardware capacity changes dynamically.

A change to this setting does not take effect until the next collection repository switch.


**Y** Specifies that hardware capacity data is collected and written out.

**N** Specifies that hardware capacity is not collected or written out.

If no CAP command is issued, the default is CAP(Y).

*Table 14. Examples of using the CAP command*

| Command purpose | Example code |
|---|---|
| Collect hardware capacity data. | `F HSIJMON,CAP(Y)` |
| Do not collect hardware capacity data. | `F HSIJMON,CAP(N)` |

## CIC – Allow or disable program usage data from CICS regions

The CIC command provides a system-wide control mechanism to allow or disallow program usage data to be collected by the Usage Monitor CICS global user exit (GLUE) program.

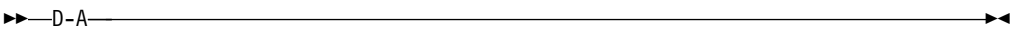

**Y** Specifies that customized CICS regions are able to present program usage data to the Usage Monitor for collection.

**N** Specifies CICS program usage monitoring is disabled throughout the operating system image.

If no CIC command is issued, the default is CIC(Y).

## D-A - Display output allocation parameters

D-A is used to display dynamic allocation details to be used in the creation of output data files. The data set name, DCB attributes, primary and secondary space quantities, and unit and optional volume serial number are shown.




The following code example displays the current dynamic allocation values.

```
F HSIJMON,D-A
```

## D-C - Display the counters and statistics

D-C is used to display the Usage Monitor activity and status indicators. The purpose of this command is to assist IBM technical support in problem diagnosis. The meaning of the output generated by this command is not published.

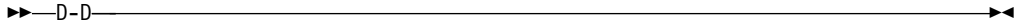

The following code example displays the current value of internal Usage Monitor counters.

```
F HSIJMON,D-C
```

## D-D - Display the data set name inclusion and exclusion lists

D-D is used to display the data set name masks in the inclusion list, followed by the data set name masks in the exclusion list.

The inclusion and exclusion lists do not need to be populated in order to collect data. The absence of any entries in the exclusion list means that data collection is not filtered by program library data set names.




The following code example displays the current data set name inclusion and exclusion lists.

```
F HSIJMON,D-D
```

## D-I - Display the system identifier

D-I is used to display the system identifier, which is written in the output header record. It can be altered by the **SID** command.


The following example code displays the current system identifier used by the Usage Monitor.

```
F HSIJMON,D-I
```

## D-S - Display the status settings

D-S is used to display several miscellaneous settings. Other commands are used to alter the individual settings, but this command provides a convenient way to list the current values.


Place at the end of the HSIZIN file to confirm monitoring settings.

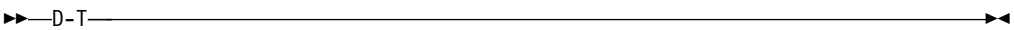
The following example code displays the current values of settings.

```
F HSIJMON,D-S
```

## D-T - Display the automatic switch-and-write time setting

D-T is used to display the time-of-day specified for automatic collection repository switching and consequent writer task creation. When data from after this time-of-day is detected, data collection is automatically switched to a new repository, and write-out of data in the old repository is started.

The UTC or GMT switch time is calculated using local time current at collection repository creation time. The time when a collection repository is terminated is set when it is created. Changes to the system local time offset, such as those caused by a change to daylight saving time, do not alter the UTC or GMT that the current collection repository is closed. The time of the switch after the next switch is calculated using the new local time.

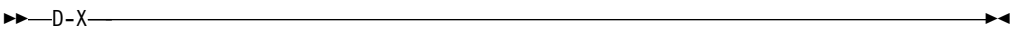



The following example code displays the current automatic switch-and-write time setting.

```
F HSIJMON,D-T
```

## D-X - Display the active exclude list

D-X is used to display the active program name mask exclude list. Data is not collected for programs with names that match the mask in any active entry in the exclude list.




The following example code displays the current exclude list entries.

```
F HSIJMON,D-X
```

## DCB - Set output DCB attributes

DCB is used to set DCB attributes, which are optimal for a specific device type.


If no DCB command is issued, the default is DCB(3390).

**DCB(3390)**

Sets the output DCB to

```
RECFM=VB,LRECL=27994,BLKSIZE=27998
```

Use when the output device has 3390 compatible geometry.

**DCB(3380)**

Sets the output DCB to

```
RECFM=VB,LRECL=23472,BLKSIZE=23476
```

Use when the output device has 3380 compatible geometry.

**DCB(UNKN)**

Sets the output DCB to

```
RECFM=VBS,LRECL=32756,BLKSIZE=0
```

The system determines the optimal block size for the device used by dynamic allocation. Use when the output device type is not known until allocation time.

Some FTP products do not process a file with RECFM=VBS correctly, even when no records are actually spanned.

## DEL - Deleting program mask entries

DEL is used to remove program name masks from filter tables. Both default and user-added entries can be removed. The required operand specifies one or more program name masks.


***mask*** Specifies a 1 - 8 character program name mask. Any wildcard characters in the mask are treated as literals for the purposes of finding the mask to delete.

***ALL*** Specifies every currently active mask. This mask cannot be specified with any other mask.

Except for short test periods, it is expected that default exclusion masks such as IGG* remain active.

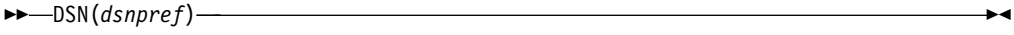
*Table 15. Examples of using the DEL command*

| Command purpose | Example code |
|---|---|
| Remove all entries, so that all possible programs are monitored. | `F HSIJMON,DEL(*ALL*)` |
| Remove exclusion masks to monitor LE and REXX modules. | `F HSIJMON,DEL(CEE*,IRX*)` |
| Remove an exclusion mask to monitor the program called CEE. | `F HSIJMON,DEL(CEE)` |

## DSN - Setting the data set name prefix

DSN is used to specify the first part of the data set names used for the output files. The prefix is specified in the required operand. The HSIZIN file must contain a DSN command.

You can use symbols in the construction of the data set name prefix. Available symbols include all z/OS static symbols, &SMF, the SMF identifier for the system, and &SYSLPAR, the logical partition name for the system.




*dsnpref* Specifies a 1 - 26 character data set name prefix. It can contain one or more data set qualifiers, and must not end in a period after any symbol substitution.

Usage Monitor needs RACF ALTER access to the data sets to be able to create them.

The following example code shows how to get output files with names of the form `SYS3.HSI.HSIJMON.Dyyyyddd.Thhmmsst`:

```
F HSIJMON,DSN(SYS3.HSI.HSIJMON)
```

## DUR - Set execution duration

DUR is used to specify a fixed short-term execution duration of the Usage Monitor started task. When the specified time has elapsed the Usage Monitor will terminate automatically. The Usage Monitor stop time is calculated by adding the specified duration to the current time when the command is processed.

Any subsequent WRT commands are ignored.

The DUR command is not normally used in standard operations where the Usage Monitor is to remain active until system shutdown. When it is used, it is normally placed in the HSIZIN file to specify a predetermined length of execution for sampling or testing purposes.


*hhmm* Specifies a time duration in hour and minute notation. The value must be four decimal digits. The minimum value is 0001 and the maximum value is 2400. The last two digits (mm) must be in the 00 - 59 range.

The following example code instructs the Usage Monitor to stop after 150 minutes.

```
F HSIJMON,DUR(0230)
```

## EXC - Adding program mask exclusion entries

EXC is used to add program name masks to the exclusion table. The required operand specifies one or more program name masks.


*mask* Specifies a 1 - 8 character program name mask. If the mask ends in an asterisk only, characters before the asterisk are compared. Otherwise, an exact program name is deemed to have been specified.

***DFLT***
Specifies every supplied default entry in the exclusion table is to be made active, and all user-added entries are to be removed from the exclusion table. This mask cannot be specified with any other mask.

Except for short test periods, it is expected that default exclusion masks such as IGG* would remain active.

*Table 16. Examples of using the EXC command*

| Command purpose | Example code |
|---|---|
| Reset the exclusion table to its default status. | `F HSIJMON,EXC(*DFLT*)` |
| Exclude the collection of data for Language Environment modules and REXX modules. | `F HSIJMON,EXC(CEE*,IRX*)` |
| Exclude the collection of data for the program CEE. | `F HSIJMON,EXC(CEE)` |

## HOF - Adjust for hypervisor STCK TOD clock offset

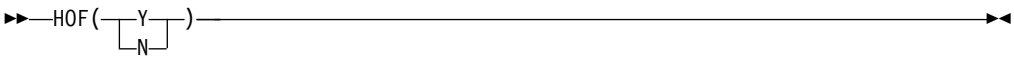
HOF is used to control whether the TOD clock offset in a logical partition is to be applied to collected data, or not. When HOF(N) is set, data timestamps are derived from the local time as supplied by z/OS. When HOF(Y) is set, the hypervisor STCK date and time offset from field SMF89HOF in SMF type 89 records is subtracted from z/OS local time to form the collected timestamp values.

For Usage Monitor data, the HOF setting at the time that the writer task is attached after the closure (or switch) of a collection repository is used.

For Inquisitor data, the HOF setting active at the time of the Inquisitor program initialization is used.

HOF(Y) will not cause any change to data timestamp values unless the Usage Monitor has processed a type 89 SMF record. For this to occur, SMF parameter settings must specify the collection of type 89 records, and at least one SMF interval must have ended while the Usage Monitor is active before the ouput file data generation commenced.

If the Usage Monitor has been stopped before an Inquisitor scan commences, the Inquisitor program uses the HOF status current at the time of Usage Monitor termination.



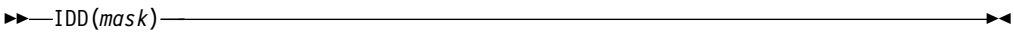

**Y** Specifies the hypervisor STCK TOD clock offset will be used to adjust date and time values present in collected data.

**N** Specifies the date and time values present in collected data will be based wholly on the local time, as maintained by z/OS.

HOF(N) is the default setting that will be used if no HOF command has been issued since IPL.

## IDD - Deleting data set name inclusion entries

IDD is used to remove data set name masks previously added by the **IDS** command.




*mask* Specifies a 1 - 44 character data set name mask. Any wildcard characters in the mask are treated as literals for the purposes of finding the mask to delete.

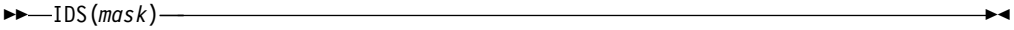
The following example code deactivates the SYS3.LINKLIB inclusion mask.

```
F HSIJMON,IDD(SYS3.LINKLIB)
```

## IDS - Adding data set name inclusion entries

IDS is used to supply data set name masks, which specify data set names to be excluded from exclusion processing. Program usage data fetched from data sets with names matching inclusion masks, is collected without reference to the data set name mask exclusion list.

Inclusion masks only affect data collection if there are active exclusion masks. An inclusion mask is normally expected to match a subset of data set names, which would match an exclusion mask.




*mask* Specifies a 1 - 44 character data set name mask. If the mask ends in an asterisk only characters before the asterisk are compared. Percent signs in the mask indicate that any character in that location is considered a match. If the mask contains a slash character (/), the value is considered to be a UNIX path name mask rather than a data set name mask.

You can use the following example code if your intention is to not collect program usage data for data sets with a high-level qualifier of SYS3, except for SYS3.LINKLIB. SYS3.LINKLIB is the only data set with a high-level qualifier of SYS3 for which program usage data is to be collected.

```
XDS(SYS3.*)
IDS(SYS3.LINKLIB)
```

## IPH – Control collection of TCPIP Host details

JIPH is used to control the reporting of the TCPIP host name and IP address. Program usage data sets created by the Usage Monitor will normally have the TCPIP host details present in the header information, but this data can be suppressed by specifying IPH(N).


**Y** Specifies that the Usage Monitor will call HSIPHOST to procure TCPIP host details.

**N** Specifies that the Usage Monitor will not report TCPIP host details in the program usage files.

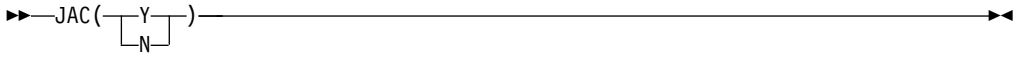
If no IPH command is issued, the default is IPH(Y).

## JAC - Set job account collection status

JAC is used to specify if the Usage Monitor is to consider the account code of jobs significant when aggregating data. The Usage Monitor normally aggregates data based on the program name, the job name, and the user ID. This setting is used to add the job account, truncated after 20 characters, to the aggregation key.

Do not instruct the Usage Monitor to collect and preserve all job account codes if they are not important to the administration of your system. Collecting and preserving job accounts can significantly increase data volumes.

A change to this setting does not take effect until the next collection repository switch.



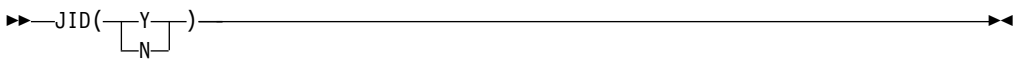

**Y** Specifies that job account codes are used.

**N** Specifies that job account codes are ignored.

If no JAC command is issued, then job accounts are not used. The default is JAC(N).

## JID - Control the preservation of batch job identifiers

JID is used to control whether all batch job identifiers are to be preserved or not. Normally usage data for each program is aggregated by job name and user ID with only the most recent job identifier being retained. JID provides the option of keeping all batch job identifiers so that the number of jobs using a program can be counted, and usage can be attributed to specific individual jobs. Job identifier aggregation for started tasks and TSO user sessions is always equivalent to JID(N) and is not affected by this setting.




**Y** Specifies that batch job identifiers should not be overlaid and that different batch job identifiers should prevent data aggregation.

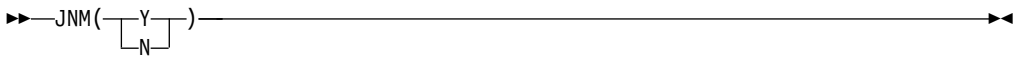
**N** Specifies that normal aggregation by job name and user ID is to proceed without considering job identifier differences.

The default setting of JID(N) applies each time the Usage Monitor is started.

## JNM - Control the collection of job names

JNM is used to specify whether the Usage Monitor collects the names of jobs which use programs or not. If the names of jobs which use the various programs are not considered to be important, you can dispense with the collection of these names. The advantage of not collecting individual job names is the reduction in processing times and data volumes caused by the aggregation of data into fewer records. When individual job names are not collected, usage is summed over broad address space categories, such as JOB, STC, TSO, and SYS. The total usage counts collected by the Usage Monitor for each program are not affected by this setting.

A change to this setting takes effect at the next collection repository switch.



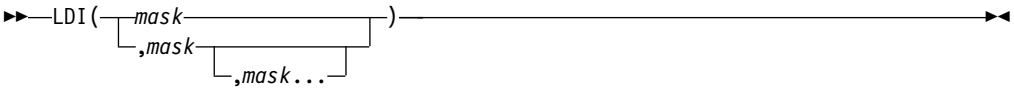

**Y** Specifies that the name of each job running a program is to be collected.

**N** Specifies that only a broad address space category of each job running a program is to be collected, instead of the individual job name.

If no JNM command has been issued, then job names are collected. JNM(Y) is the default.

## LDI – Set LOAD exclusion inactive

LDI is used to deactivate a LOAD exclusion entry previously activated by the LDX command. Names specified as operand values should exactly match names already used in LDX commands.




***mask*** Specifies a 1 - 8 character program name mask

## LDX – Add or activate a LOAD exclusion

Normally when programs are loaded into storage, the Usage Monitor deems that the loaded programs are being used. However, there are software packages which load programs as part of a process to analyze these programs for software control purposes. When this happens, usage is attributed to each of the loaded programs even though they were not given control or actually used. In such cases, you may choose to specify the name of the program performing the analysis in an LDX command in order to exclude the referencing of the loaded programs from data collected by the Usage Monitor.

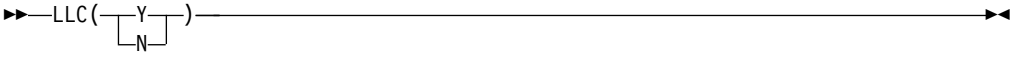
When a program is explicitly loaded into and deleted from storage, the Usage Monitor checks the active program name of the task against the list of active LDX entries. If the name of the active program – the program assumed to be issuing the LOAD and DELETE requests – matches the value stored in an active LDX entry, then the Usage Monitor will exclude the event from the collected usage data.

Usage collected from events where the operating system gives control to programs (such as ATTACH and LINK requests) are not affected by LDX entries.


***mask*** Specifies a 1 - 8 character program name mask

## LLC - Link list correction

LLC is used where sites make a number of dynamic link list changes. This command updates the HSIJMON data to point to the correct load library. Use this command only if you enable dynamic link list updates, which alter the relative concatenation numbers of persisting libraries.




**Y** A BLDL is performed at write time by the writer task and, if found, the data set name is overlaid. To avoid performance problems, especially during system shutdown, ensure that LLA remains active until the Usage Monitor has terminated.

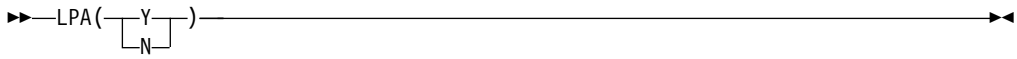
**N** Do not check for dynamic list updates.

If no LLC command is issued, then the default setting of LLC(N) is current.

A change to this setting does not take effect until the next collection repository switch.

## LPA - Set link pack area program monitoring status

LPA is used to specify whether the monitoring of programs in the Link Pack Area (LPA) is to occur or not. All types of LPA are included in this category.




**Y** Specifies that LPA program usage is to be monitored.

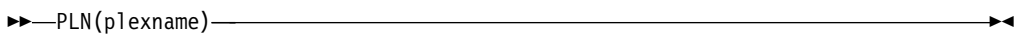
**N** Specifies that LPA program usage is not to be monitored.

If no LPA command is issued, then LPA program usage data is collected. LPA(Y) is the default setting.

A change to this setting does not take effect until the next collection repository switch.

## PLN - Set the sysplex name

PLN is used to override the name of the sysplex contained in the output header record. The actual sysplex name is used as a norm, but an override allows control over which systems have their inventory updated when the PLX=Y Inquisitor setting is used. The value specified here should match the PLEXNAME= value specified for the corresponding Inquisitor scans. Overriding the sysplex name is not usually needed unless PLX=Y is used and the sysplex grouping does not match the shared DASD grouping. Symbols can be employed in the construction of the sysplex name. Available symbols include all z/OS system symbols, &SMF, the SMF identifier for the system, and &SYSLPAR, the logical partition name for the system.




**plexname**
Specifies a string which is to be resolved to an identifier 1-8 bytes in length.

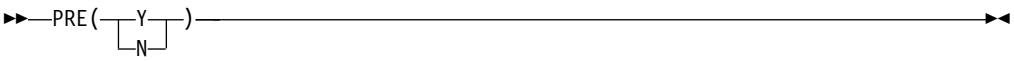
## PRE - Collect usage for long running programs

PRE is used to specify if the Usage Monitor is to collect usage for programs which started before the current collection cycle. Without this data collection a Usage Monitor collection cycle will have no usage data for programs which started running before the cycle started and remain running when the cycle ends. If a job or task runs for more than two days, most days will not have any usage recorded for the main program unless this additional data collection is enabled.

When the additional data collection is enabled, previously fetched programs resident in the regions of started task and batch job address spaces where SMF

interval recording is active have usage recorded in each collection cycle which encompassed the end of at least one SMF interval.

This setting can affect usage figures. For example, the main program of a constantly running task can accrue a usage count of around 30 over a month even though it was really only used once for an extended period.



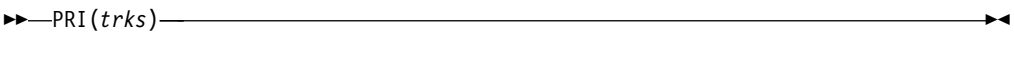

**Y** Specifies that usage for previously running programs is to be collected.

**N** Specifies that usage for previously running programs is not to be collected.

The default setting of PRE (Y) applies each time the Usage Monitor is started.

## PRI - Set the data set space primary allocation

PRI is used to specify the primary space allocation quantity in tracks. It is used for output data set allocations.




*trks* Specifies a number of tracks from 0 to 150,000.

If no PRI command is issued, the primary space allocation is 750 tracks. The Usage Monitor uses the RLSE space allocation attribute.

The following example code sets the primary space allocation to 900 tracks.

```
F HSIJMON,PRI(900)
```

## PRS - Set registered software activity data collection status

PRS is used to specify if the Usage Monitor is to output records containing information about the activity of registered software. Registered software uses the system Register service. The data contains information about the usage of registered software, and information about software registration settings from the PARMLIB member IFAPRDxx.

A change to this setting does not take effect until the next collection repository switch.


**Y** Specifies that registered software information is collected and output.

**N** Specifies that registered software information is neither collected or output.

If no PRS command is issued, then registered software data is collected. PRS(Y) is the default.

## QSZ – Specify collection element queue area size

QSZ is used to specify the virtual storage size of the SCOPE=COMMON memory object which forms the area where collected usage data is queued to the Usage Monitor address space for storing into the collection repository. The QSZ value specifies the number of storage segments the area occupies, where a segment is one megabyte in size.

The QSZ value used is fixed for the life of the Usage Monitor address space. To change the QSZ value the Usage Monitor started task must be recycled.

```
►►—QSZ(segments)———————————————————————————————————►◄
```

*segments*
Specifies a number of segments from 1 to 200.

If no QSZ command is issued, a 10MB queue area will be used. The queue area is processed as a LIFO stack, which means that only the necessary number of pages needed to hold the peak queue length will need to be backed by physical storage, no matter how large the QSZ value is set.

## REF - Refresh Usage Monitor settings

REF is used at any time to reset Usage Monitor settings according to commands in the HSIZIN file, without stopping and starting the Usage Monitor. The detailed results of the refresh operation are written to the HSIZMSG file.

The processes of a refresh operation include:

- Verify that HSIZIN is still allocated.
- Open HSIZIN.
- Set the program exclusion list to the default list.
- Deactivate all data set exclusion list elements.
- Deactivate all data set inclusion list elements.
- Process the commands in HSIZIN.
- Close HSIZIN.
- Issue either HSIZ059I or HSIZ060I, as appropriate.

```
►►—REF———————————————————————————————————————————►◄
```

The following example code changes Usage Monitor settings to updated values from HSIZIN.

```
F HSIJMON,REF
```

## SEC - Set the data set space secondary allocation

SEC is used to specify the secondary space allocation quantity in tracks. It is used for output data set allocations.

```
►►—SEC(trks)———————————————————————————————————————►◄
```

*trks* Specifies a number of tracks from 0 to 150,000.

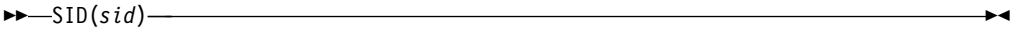
If no SEC command is issued, the secondary space allocation is 300 tracks. The Usage Monitor uses the RLSE space allocation attribute.

The following example code sets the secondary space allocation to 600 tracks.

```
F HSIJMON,SEC(600)
```

## SID - Set the Usage Monitor system identifier

SID is used to override the system identifier contained in the output header record. The SMF system identifier is used as a norm, but an override enables the data from separate systems to be differentiated in all instances where duplicate SMF identifiers are in use. Symbols can be employed in the construction of the system identifier. Available symbols include all z/OS system symbols, &SMF, the SMF identifier for the system, and &SYSLPAR, the logical partition name for the system.




*sid* Specifies a string which is to be resolved to an identifier 1-4 bytes in length.

*Table 17. Examples of using the EXC command*

| Command purpose | Example code |
|---|---|
| Set the output system identifier to PROD. | `F HSIJMON,SID(PROD)` |
| Set the header record system identifier to the current LPAR name. The LPAR name must not exceed four characters in length. | `F HSIJMON,SID(&SYSLPAR)` |

## SIZ - Set the data space repository size

SIZ is used to specify the maximum number of entries that the collection repository can hold.


*entries* Specifies a number of entries from 100 to 6,000,000.

If no SIZ command is issued, a data space capacity of 200,000 entries is used. Each entry occupies 232 bytes and contains a pointer to a separate part of the repository dedicated to holding data set and UNIX file names. Storage is conserved by only storing a single copy of each collected data set and UNIX file name. As each repository page has data placed in it for the first time, that page must be backed physically by the system. When a collection repository is full, a repository switch is triggered automatically. A repository switch also occurs when data stamped after the switch time is detected, or when a manual switch is requested by the **SWI** command.

The following example code sets the size of future collection repositories to 1,000,000 entries.

```
F HSIJMON,SIZ(1000000)
```

## SJS - Controlling spawned job suffix preservation

When a spawned address space is created by a unit of work with a job name that is shorter than eight characters, the system appends a sequence digit in the 1 to 9 range to the job name, and this becomes the job name of the spawned address space. This approach means that the usage of programs generated by jobs with a specific name can be logged under as many as ten different job names. The system-generated job names usually do not assist in identifying the source of the work because there is often no other reconciliation data which also uses these generated names.

The SJS setting can be used to remove the spawned sequence number suffix so that all usage events for programs are logged under the original job name, resulting in fewer Usage Monitor records and reduced processing time. If the spawning job name is eight characters long and ends in a digit in the 1 to 9 range, then activity in spawned address spaces (but not the original address space) can be reported under a job name which is only the first seven characters of the original job name. If this is likely to present a problem then use SJS(N).


**Y** Spawned job name suffix digit is truncated.

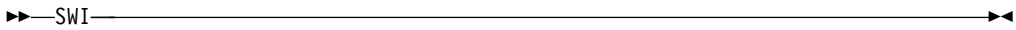
**N** No editing is performed of spawned address space job names.

SJS(Y) is the default if no SJS command has been issued since the Usage Monitor started.

## SWI - Switch to a new collection repository

SWI causes a new collection repository to be created and used for subsequent data collection. A writer task processes the data contents of the repository that is being used at the time that the SWI command is issued.

The SWI command has no operands. It is invalid in the HSIZIN initial command file. As well as the switch caused by an explicit SWI command, automatic switches occur when a repository becomes full, and when data stamped after the switch time is detected. The SWI command might be rejected if the writer task is busy.




The following example code manually switches to a new repository.

```
F HSIJMON,SWI
```

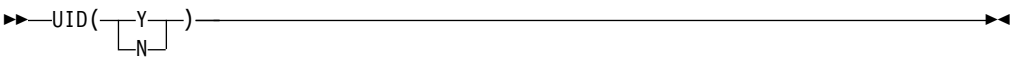
## UID - Control the collection of user details

UID is used to specify whether the Usage Monitor collects the identifiers and names of users who use programs or not. If the details of users who use the various programs are not considered to be important, then you can dispense with the collection of this information. The advantage of not collecting user information is the reduction in processing times and data volumes.

When user information is not collected, the user ID data item remains blank, and user names are not output, regardless which UNM setting is current. The total usage counts collected by the Usage Monitor for each program are not affected by this setting.

If you want program usage attributed to individual users but do not want the names of users to be retained, use UID(Y) and UNM(N).

A change to this setting does not take effect until the next collection repository switch.



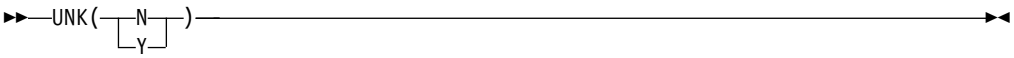

**Y** Specifies that details of each user using a program are to be collected.

**N** Specifies that details of each user using a program are not to be collected.

If no UID command is issued after IPL, user details are collected. UID(Y) is the default.

## UNK - Set the unknown event collection switch

UNK is used to specify whether events with incomplete data are to be collected or not. The database content is not affected. Collecting extra data is useful in determining why some usage events are not captured. It must be set only when requested by IBM support.




**Y** Specifies that the "unknown" events are to be collected.

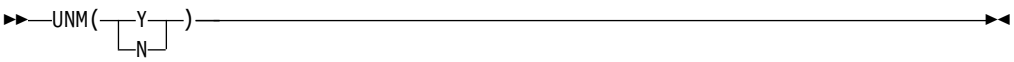
**N** Specifies that the "unknown" events are not to be collected.

If no UNK command is issued, the unknown events are not collected. UNK(N) is the default setting.

A change to this setting does not take effect until the next collection repository switch.

## UNM - Set user name collection status

Software security packages, such as RACF, have a name field for each user ID defined to the system. The Usage Monitor collects the user ID (up to eight characters long), and the contents of the name field (up to 20 characters long), as part of the data collection performed when programs are used. UNM is used to specify whether the names of users collected from the security package are output. The output of the user ID is controlled by the **UID** setting. This setting is checked by the writer task when the data in a collection repository is being processed for output.



**Y** Specifies that collected user names are written to the output file.

**N** Specifies that collected user names are discarded.

If no UNM command is issued, then user names are collected. UNM(Y) is the default.

A change to this setting does not take effect until the next collection repository switch.

## UNT - Set the data set allocation unit

UNT is used to specify the allocation unit to be used for output data set allocations.


*Unitname*
Specifies a 1 - 8 character long unit name.

If no UNT command is issued, SYSALLDA is used.

The following example code sets the allocation unit to WORKDA.

```
F HSIJMON,UNT(WORKDA)
```

## USS - Set UNIX program monitoring status

USS is used to determine if the programs retrieved from Hierarchical File System (HFS) files are to be monitored.


**Y** Programs fetched from HFS files are to be monitored.

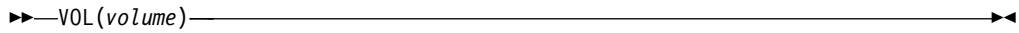
**N** Programs fetched from HFS files are not to be monitored.

If no USS command is issued, the programs retrieved from HFS files are not monitored. USS(N) is the default setting.

A change to this setting does not take effect until the next collection repository switch.

## VOL - Set the data set allocation volume

VOL is used to specify the allocation volume to be used for output data set allocations. The explicit nomination of a specific volume is necessary when there are no PUBLIC or STORAGE volumes in the allocation unit pool.




*volume* specifies a 1 - 6 character long volume serial number.

If no VOL command is issued, a specific volume is not explicitly requested. You must then have PUBLIC or STORAGE volumes in the public allocation pool, unless the data sets are managed by SMS.

The following example code sets the allocation volume to SCR001.

```
F HSIJMON,VOL(SCR001)
```

## WRT - Set the automatic switch-and-write time of day

WRT is used to specify a time-of-day to end data collection for the current collection repository, and automatically switch to a new one. The data write-out for the closed repository is also initiated at the same time. These events are triggered when data from after the specified time is detected.

The UTC or GMT switch time is calculated using the local time when the repository is created. The time that a data space is terminated is set when it is created. Changes to the system local time offset, such as those caused by a change to daylight saving time status, do not alter the UTC or GMT time that the current repository is closed. The time of the switch, after the next switch, is calculated using the new local time.


*hhmm* Specifies a 24-hour time-of-day in hour and minute notation. The value must be four decimal digits. The first two digits (hh) must be in the 00 - 23 range. The last two digits (mm) must be in the 00 - 59 range.

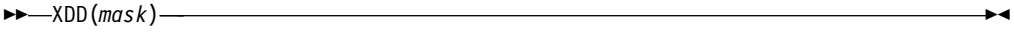
If no WRT command is issued, the automatic switch time of midnight is used. That is, WRT(0000) is the default.

The following example code sets the automatic switch-and-write time to 10 minutes before midnight.

```
F HSIJMON,WRT(2350)
```

## XDD - Deleting data set name exclusion entries

XDD is used to remove data set name masks which were added by the XDS command. XDD can also deactivate entries from the default exclusion list that was automatically created by the Usage Monitor.




*mask* Specifies a 1 - 44 character data set name mask. Any wildcard characters in the mask are treated as literals for the purposes of finding the mask to delete.

The following example code deactivates the SYS3.* exclusion mask.

```
F HSIJMON,XDD(SYS3.*)
```

## XDS - Adding data set name exclusion entries

XDS is used to supply data set name masks which specify data set names to be excluded from data collection. Program usage data for programs fetched from data sets with names matching exclusion masks is discarded. When the captured data

set name has been matched to an inclusion mask set by the IDS command, the data is collected without reference to the exclusion mask list.

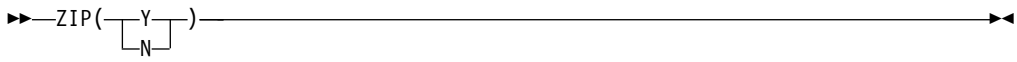

*mask* Specifies a 1 - 44 character data set name mask. If the mask ends in an asterisk, only characters before the asterisk are compared. Percent signs in the mask indicate that any character in that location is considered a match. If the mask contains a slash character (/), the value is considered to be a UNIX path name mask rather than a data set name mask.

The following example code excludes program usage data from collection for programs fetched from data sets with a high-level qualifier of SYS3.

```
F HSIJMON,XDS(SYS3.*)
```

### ZIP - Set the compressed output data switch

ZIP is used to control whether the writer task is to compress output data or not. Compressing the output data reduces data volume, in turn reducing data transfer time and storage space requirements.




**Y** Specifies that output data is to be compressed.

**N** Specifies that output data is not to be compressed.

If no ZIP command is issued, then compressed data is output. ZIP(Y) is the default setting.

## Monitoring usage in CICS regions

The CICS Transaction Server for z/OS performs much of its program management outside of the contents supervisor framework that most applications use. For the Usage Monitor to accurately detect and record the use of programs in a CICS region, you must customize each CICS region where you require detailed program usage monitoring.

To prepare a CICS region to enable detailed monitoring, you must install the following components:

- CICS global user exist (GLUE) programs
- An enabling program to activate these user exit programs
- An entry in the program list table (PLT) that triggers the enabling program

The customized JCLLIB library contains the following sample jobs that you can copy and use in your customization:

- The HSISENAX member contains a sample job to translate, assemble and bind the enabling program.
- The HSISPLTX member contains a sample job to create a PLT with the required entry to trigger the enabling program. If you use this sample job, verify the name of the enabling program and the PLT suffix before you submit the job.

The CICS program monitoring facility does not support releases earlier than CICS Transaction Server version 3, release 2. Different releases of the CICS Transaction Server require different versions of the GLUE programs. You must ensure that the correct version of these programs is used for each CICS region. You must also take care when upgrading regions to a later release of CICS so that the correct version of this module will be used with the newer software. The following table lists the required GLUE programs for different versions of the CICS transaction server.

| CICS Transaction Server release | exits program |
|---|---|
| Version 3, release 2 | HSIZFTC0, HSIZEII0 |
| Version 4, release 1 | HSIZFTC1, HSIZEII1 |
| Version 4, release 2 | HSIZFTC2, HSIZEII2 |
| Version 5, releases 1, 2, and 3 | HSIZFTC2, HSIZEII2 |

When you implement this CICS Transaction Server customization, the Usage Monitor can collect and record data related to program name and data set name. The collected data is subject to the Usage Monitor program name and data set name selection and exclusion filters. You can stop data collections from all HSIZFTC*x* and HSIZEII*x* GLUE programs with the CIC(N) Usage Monitor setting. CIC(Y) is the default setting if you do not issue a CIC Usage Monitor command.

Depending upon the level of program usage detail you require, you may find that the HSIZFTC*x* exit produces sufficient data for your needs without also installing the HSIZEII*x* exit. If you want to access more detailed CICS data such as particulars of transactions and the end users involved, a specialized CICS monitor such as IBM Tivoli OMEGAMON XE for CICS on z/OS is required.

## Customizing a CICS region to provide usage data

### Procedure

1. Copy the appropriate HSIZFTC*x* and HSIZEII*x* global user exit (GLUE) programs from the SHSIMOD1 library to a DFHRPL library of the CICS region.
2. Customize and submit the HSISENAX job to create a program that enables the HSIZFTC*x* and HSIZEII*x* exit programs:
   a. Customize the sample job for translating, assembling, and binding the enabling program that is provided in the HSISENAX member in the customized JCLLIB library. For convenience, you can name this program HSIZENA*x*, where *x* is the same suffix character as the suffix of the HSIZFTC*x* and HSIZEII*x* programs that it enables.
   b. Check that the name specified in the PROGRAM operand of EXEC CICS ENABLE statement is the name of the enabling program.
   c. Check that the name specified in EXEC CICS ENABLE PROGRAM statement is the name of the of the GLUE programs.
   d. Link the HSIZENA*x* enabling program into the same DFHRPL library where you copied the HSIZFTC*x* and HSIZEII*x* GLUE programs.
   e. Submit the HSIZENA*x* job.
3. Add an entry in the following format to the active program library table (PLT) of the CICS Transaction Server to install the HSIZENA*x* module:
   DFHPLT TYPE=ENTRY,PROGRAM=HSIZENA*x*
   Place the entry before the DFHPLT TYPE=ENTRY,PROGRAM=DFHDELIM entry so that it loads early during CICS initialization to minimize the need for program resource definitions.

4. Ensure that the PLTPI setting for the CICS region specifies your newly updated PLT.
5. Optional: Use the HSITAGP tagger program to tag non-vendor application programs that you want to be identified in usage reports.

### Results

When you complete this task, the use of programs that are given control by various mechanisms in the CICS Transaction Server are attributed to the CICS region address spaces that invoke them.

### What to do next

You can stop data collection from all HSIZFTC*x* and HSIZEII*x* glue programs with the CIC(N) Usage Monitor setting. The CIC(Y) option is the default if you do not issue a CIC Usage Monitor command.

## Importing Inquisitor data

The Inquisitor Import reads data from Inquisitor scans, where the data is filtered and matched to products. The filtered, matched data is then copied to the Repository tables where it can be viewed and queried by the Analyzer reporting utility.

**Related concepts**:

"PLX parameter of the Inquisitor program" on page 42
The **PLX** parameter can reduce the time it takes to scan and process different SIDs that are completely shared and are, therefore, identical. When you set **PLX**=Y, the Inquisitor Import detects libraries that are mirrors or libraries that have not changed and quickly processes scans of these shared SIDs.

**Related tasks**:

"Collecting scanned libraries with the Inquisitor for z/OS" on page 41
The Inquisitor is a program that scans and collects information about partitioned data set (PDS) and partitioned data set extended (PDSE) program libraries. The Inquisitor Import program takes the collected data as input to form the basis of your software inventory.

## Running the Inquisitor import

The HSISIQIM job in the JCLLIB library performs the Inquisitor import. This job is generated from the HSISCUST post-installation customization job.

### About this task

Run-time for the HSISIQIM job depends on the number of modules to be imported into the database Inquisitor tables. Because the processing is memory-intensive, run the HSISIQIM job during off-peak periods.

If the HSISIQIM job processes the same data that is generated by the Inquisitor scan for a specified system, the job terminates with an error to indicate that the input file is a duplicate input file.

### Procedure

1. In the HSISIQIM job, update the following parameters, according to your requirements:

- **FULLREMATCH**: Set to N (no) to skip import from scanned libraries where no member directories have changed since a previous Inquisitor import to the same Repository. Set to Y (yes) to import and match process all libraries.

  When **FULLREMATCH** is set to Y, an extra check is performed on libraries in the repository. For a given system ID (SID), existing libraries in the repository are marked as deleted unless the libraries are found in the scanned Inquisitor file. For shared libraries (where PLX=Y), the scanned Inquisitor file can be from any SID that belongs to the sysplex. For non-shared libraries (where PLX=N), the scanned Inquisitor file must be from the same SID. The deletions include products, libraries and modules.
- **PRODUCTONLY**: Set to N (no) to import all modules, including unidentified modules. Set to Y (yes) to import only matched modules.

The settings for both of these parameters influence the duration of the import process.

2. Optional: Review and modify other parameters, as required, in the PARMLIB member.
3. Submit the HSISIQIM job.

## Import filters and matching

When you import data collected by the Inquisitor, the import procedure reads the data and filters and matches the data before copying the data to the Repository tables.

The Inquisitor Import loads data and performs the following tasks:

1. Reads Inquisitor data generated from Inquisitor scans. To exclude importing specific libraries, the Inquisitor data is filtered against a set of supplied Inquisitor filter tables. These Inquisitor filter tables are updated monthly, together with the knowledge databases. The filtering excludes, for example, the ISV distribution libraries.
2. Matches load modules to best fitting products at the version, release, and modification (VRM) level. Best matches for modules are found based on module names and sizes, and information in the Global Knowledge Base (GKB) and Local Knowledge Base (LKB). Temporary scorecard tables are used to hold all the possible scorecards for modules in a given library while they are matched.
3. Loads matched load modules, including matching information, into the Repository tables. Data from the Repository tables are now ready for viewing or reporting using the Analyzer reporting facility.
4. Aggregates usage data for rediscovered modules in the Repository tables.

The Inquisitor Import uses memory intensively in order to efficiently match many Knowledge Base scorecards to library modules. The maximum memory requirements depend on the number of modules in a library of an Inquisitor import file, and the number of scorecards in the GKB and LKB that affect the processed libraries. To estimate a requirement, allow 5M +1.5k per module. For example, for an Inquisitor file containing a maximum library size of 30000 modules, the requirement is approximately 5M + (0.0015*30000) = 50M.

## TPARAM parameters

The TPARAM parameters that you specify for Inquisitor import define what data is included in the import.

**COMMIT=**
Default is 1000. Number of records stored before issuing a COMMIT.

**COUNTUSAGE=**

The parameter controls when product usage totals are counted. Default value is Y and indicates that they will be counted in the Inquisitor Import step for modules that have usage and are identified for the first time perhaps because usage was imported before discovery was performed. These recounts of product usage because of modules that are identified can be time consuming. Specify N to skip this task so that it can be performed only once in the Aggregator step.

The default value is provided for backward compatibility for existing setups. It is more efficient and common in most environments to run a series of Inquisitor Import and Usage Import jobs followed by a single run of the Aggregator as the last job before the Analyzer is used to view the repository data. Specify COUNTUSAGE=N in your Inquisitor Import jobs if you are running the Aggregator as the last step so that time consuming counts occur only once during aggregation.

The Inquisitor Import deletes count entries for products for which at least one module is identified for the first time and that have usage details as these counts are invalidated by the identification. This is done regardless of the setting of the COUNTUSAGE parameter. These entries will be recalculated by the Aggregator if not done earlier by a job where COUNTUSAGE=Y behavior is in effect.

When you select COUNTUSAGE=N, you must run the Aggregator as the last step, or select COUNTUSAGE=Y in the last submitted job to ensure that usage count totals entries are complete.

**DSN=** DB2 location. Value assigned, as defined in job HSISCUST

**FILTERSCHEMA=**
Inquisitor Import filter qualifier. Name of qualifier is *&GKBZSCHM*_IQF7

**FULLREMATCH=**
Default is N, which means import and match processing will be skipped for scanned libraries that have had no member directory changes since a previous Inquisitor Import and into the same Repository. Y means that all libraries will be imported and matched.

**GKBSCHEMA=**
Global Knowledge Base qualifier for z/OS. Name of qualifier is *&GKBZSCHM*_GKB7

**GKUSCHEMA=**
Global Knowledge Base qualifier for z/OS UNIX. Name of qualifier is *&GKBZSCHM*_GKU7

**LKBSCHEMA=**
Local Knowledge Base qualifier for z/OS. Name of qualifier is *&REPZSCHM*_LKB7

**LKUSCHEMA=**
Local Knowledge Base qualifier for z/OS UNIX. Name of qualifier is *&REPZSCHM*_LKU7

**PRODUCTONLY=**
Default is N, which means all modules, including unidentified modules,

are loaded into the Repository. Y means only modules that have been matched to known products are loaded into the Repository, meaning, application modules are excluded.

**REPSCHEMA=**
Repository qualifier. Name of qualifier is *&REPZSCHM*.

## Importing usage data

The Usage import job imports data generated by the Usage Monitor and aggregates usage data for discovered or identified modules in the Repository tables in the database.

The HSISUIMP job in the JCLLIB library performs the Usage import. This job is generated from the HSISCUST post-installation customization job. Because run-time for this job depends on the volume of usage data to load, run this job during off-peak periods.

Usage data files can be either outputs from the Usage Monitor or condensed outputs from the ZCAT utility. These output files can be concatenated as a single input in the job. When you are priming a new repository, use a single small file as input when loading usage data for the first time.

If the HSISUIMP job processes the same data that is generated by the Usage Monitor for a specified system, the job terminates with an error to indicate that the input file is a duplicate input file.

**Related tasks**:

"Collecting usage data with the Usage Monitor" on page 58
The Usage Monitor is a server address space that runs as a started task. Work is queued to the Usage Monitor from all address spaces where programs are used. The Usage Monitor moves captured program usage data into the collection repository and periodically writes the accumulated data to a sequential file. The Usage Monitor runs APF authorized and is non swappable.

### TPARAM parameters

The TPARAM parameters that you specify for Usage Import define what data is included in the import.

**COMMIT=**
Default is 1000. Number of records stored before issuing a COMMIT.

**COUNTUSAGE=**

The parameter controls when product usage totals are counted. Default value is Y and indicates that they will be counted in the Usage Import step for systems and periods that are processed and receive new usage details. Specify N to skip this task so that it can be performed only once in the Aggregator step.

The default value is provided for backward compatibility for existing setups. It is more efficient and common in most environments to run a series of Inquisitor Import and Usage Import jobs followed by a single run of the Aggregator as the last job before the Analyzer is used to view the repository data. Specify COUNTUSAGE=N in your Usage Import jobs if you are running the Aggregator as the last step so that time consuming counts occur only once during aggregation.

The Usage Import deletes count entries for systems and periods that receive new usage details and are invalidated by the new details, regardless of the setting of the COUNTUSAGE parameter. These entries will be recalculated by the Aggregator if not done earlier by a job that runs where COUNTUSAGE=Y behavior is in effect.

When you select COUNTUSAGE=N, you must run the Aggregator as the last step, or select COUNTUSAGE=Y in the last submitted job to ensure that usage count totals entries are complete.

**DSN=** DB2 location. Value assigned, as defined in job HSISCUST

**FILTERSCHEMA=**
Inquisitor Import filter qualifier. Name of qualifier is *&GKBZSCHM*_IQF7

**FULLREMATCH=**
Default is N, which means import and match processing will be skipped for scanned libraries that have had no member directory changes since a previous Inquisitor Import and into the same Repository. Y means that all libraries will be imported and matched.

**GKBSCHEMA=**
Global Knowledge Base qualifier for z/OS. Name of qualifier is *&GKBZSCHM*_GKB7

**GKUSCHEMA=**
Global Knowledge Base qualifier for z/OS UNIX. Name of qualifier is *&GKBZSCHM*_GKU7

**LKBSCHEMA=**
Local Knowledge Base qualifier for z/OS. Name of qualifier is *&REPZSCHM*_LKB7

**LKUSCHEMA=**
Local Knowledge Base qualifier for z/OS UNIX. Name of qualifier is *&REPZSCHM*_LKU7

**PRODUCTONLY=**
Default is N, which means all modules, including unidentified modules, are loaded into the Repository. Y means only modules that have been matched to known products are loaded into the Repository, meaning, application modules are excluded.

**REPSCHEMA=**
Repository qualifier. Name of qualifier is *&REPZSCHM*.

## Aggregating usage and discovery data

The Aggregator is a program that populates the Asset tables.

These tables are accessed by Analyzer reports and batch jobs, and offer a higher level view of product discovery and usage at a version rather than a release or module level. The Aggregator also calculates product discovery and usage count totals that are used to speed up the Analyzer queries.

The Aggregator should be run as the final job in a batch run of Inquisitor Import and Usage Import jobs, and before the repository is accessed by the Analyzer. This ensures that asset reports and counts are synchronized with the latest collected discovery and usage details.

It is usual for asset level usage details to be kept longer than module usage details. This is controlled by the KEEPAGGR parameter of the Usage Deletion program, which has a longer default period than the KEEPDETAIL parameter. The aggregated entries are normally kept for the longer KEEPAGGR period and are usually only recalculated for periods where module usage details are available.

The Aggregator run be time consuming since it is run for the entire repository and periods rather than an individual SID (as in the case of the Inquisitor Import), or system and period (as in the case of the Usage Import). It is therefore more efficient to run the Aggregator only once following several imports of usage or discovery data. Ensure also that you do not keep usage details for longer than is necessary for your site, and that Usage Deletion is run on a regular basis.

An example of how to run the Aggregator can be seen in sample jobs that are generated by HSISCUST, such as `HSISIQIM` (Inquisitor Import) and `HSISUIMP` (Usage Import).

## TPARAM parameters

The TPARAM parameters that you specify for aggregating usage and discovery data is included in the import.

**COMMIT=**
Default is 1000. Number of records stored before issuing a COMMIT.

**COUNTUSAGEFULL=**

Default value is N and means that only entries for systems and periods that are affected by recent activities, and are skipped by the Usage Import and/or the Inquisitor Import because COUNTUSAGE=N was in effect, are calculated by the Aggregator.

A value of Y means that usage count totals for all systems and periods are deleted and recalculated.

The default value of N is chosen for backward compatibility considerations and for more efficient operations. It is more reliable, however, to recalculate all totals preferably with every run of the Aggregator, and at least occasionally if the COUNTUSAGEFULL=Y process is too time consuming. The time taken to refresh the usage counts can be seen in the Aggregator log.

**DSN=** Database location. Value assigned, as defined in job HSISCUST

**IXBUFFERPOOL=**
The buffer pool for indexes created on declared global temporary tables that are used by the Aggregator. If not supplied, the buffer pool defined for the database where the global temporary tables belong is used. This value is normally only needed by sites which have a requirement to specify different buffer pool values for different applications or invocations, and have used specific values during the creation of the database repository.

**IXSTOGROUP=**
The storage group for indexes created on declared global temporary tables that are used the Aggregator. Default is SGHSIIDX. This value need only be specified by sites that have a procedural requirement of not using the default DB2 storage group value, and have used a different STOGROUP value during the creation of the product database repository.

**GKBSCHEMA=**
Global Knowledge Base qualifier for z/OS as defined during the installation of the z/OS GKB database.

**REPSCHEMA=**
Repository qualifier as defined during HSISCUST and the subsequent creation of the product repository database.

# Activating the Automation Server

The Automation Server discovers new data sets and processes them by starting a set of predefined actions that associate the data with data set name masks that form a catalog search. This search determines if any data set names matching the mask are to be processed.

## Automation Server overview

The Automation Server provides the ability to select data sets if you have data set names that are variable, such as those created by the Usage Monitor, which have low-level qualifiers containing time stamps.

The Automation Server runs as a started task in its own address space.

The user ID for the Automation Server must have an OMVS segment and a UID, or there must be a default UID configured.

The Automation Server issues a system-wide ENQ to ensure that there is only one instance of it in a z/OS image. A single instance of the Automation Server continuously references all data sets, catalogs, and volumes that are accessible from all systems in a sysplex so it is unnecessary for the Automation Server to run on more than one system.

Input control statements define the processing to be performed by the Automation Server. There are two types of control statements, *action* statements and *DSN* statements:

***action***
Action statements name the template member which forms the basic input for the action to be performed when a relevant data set is newly discovered by a catalog search. They have optional operands to specify time-of-day, day-of-week, day-of-month, and month-of-year restrictions.

**DSN** DSN statements provide a data set name mask to be associated with the preceding action statement. There can be many DSN statements after each action statement.

There are currently two types of action statement:

**FTP** Starts the FTP utility to perform a file transfer.

For the FTP action, the template member is read and, after symbol substitution processing, is written to the file defined by the INPUT DD statement. The INPUT file is allocated to a temporary data set. The FTP program is attached as a subtask and scans the INPUT file to process the FTP requests. The report messages it generates are written to the OUTPUT file.

Upon completion of the FTP subtask, the Automation Server examines the completion code. If the program ends normally with a zero return code,

the Automation Server deems the action to have been successful and updates the action status in the HSIACDS file so the action is not repeated for this data set.

If the FTP program abends, the Automation Server deems the action to have failed. A failed transfer is tried again at a later time. A retry is subject to specified scheduling constraints. The OUTPUT FTP report file contains information to track the exact cause of a transfer failure.

**JOB** Submits a batch job.

For the JOB action, the template member is read and, after symbol substitution processing, is written to the file defined by the INTRDR DD statement. This file is directed to the internal reader used by the system, and the jobs submitted by the Automation Server become available for JCL conversion as soon as the INTRDR file is closed, or another JOB card image is found by the reader.

The Automation Server deems all JOB submissions successful, so there are no retries. Any failure should be investigated using the appropriate procedures used by your installation.

**Note:** The job stream in a JOB action template member may define more than one job.

The Automation Server does not check template member records for either FTP or JCL validity.

Before initiating an action for a detected data set, the Automation Server attempts to exclusively allocate the data set. If the data set is in use, the action is not performed and the awaiting retry status is stored in the control data set. The action is performed when the data set is next found to be available for use within any specified scheduling restrictions. Dynamic allocation failures due to other reasons do not inhibit the action.

## Running the Automation Server

To run the Automation Server, you must create a control data set, configure the Automation Server as a started task, design request control statements, and exclude data sets from processing.

### Creating the Automation Server control data set

To create the Automation Server control data set, use member HSIASALC in the JCLLIB. This member is generated from the HSISCUST post-installation customization job.

#### Procedure

1. Allocate sufficient space for the Automation Server to handle the workload required by the installation. One 96 byte record (including the 52 byte key) is required for each data set processed by the Automation Server.
2. Create the control data set by running the HSIASALC job that contains the IDCAMS JCL and control statements.
3. In the HSIACSD ddname, allocate the VSAM KSDS control data set to the Automation Server.

## Copying the started JCL task to a library

The Automation Server is run by the HSIJAUTO job in the JCLLIB library that is supplied in the SHSIPROC data set. To start the HSIJAUTO job as a started task, you must customize the JCLLIB member and copy it to an authorized PROCLIB library.

### Procedure

1. Set values for the following parameters in the HSIJAUTO task in the JCLLIB library:
   - **HSI**: Set high-level qualifiers for the installation target libraries months
   - **HSIINST**: Set high-level qualifiers for the &HSIINST..PARMLIB data set created by the HSISCUST job.
   - **ACDS**: Set the data set name of the Automation Control Data Set (ACDS).
2. Copy the HSIJAUTO member from the JCLLIB library to an authorized PROCLIB library.

**Files used by the Automation Server:**

Several files must be available for use by the Automation Server.

**STEPLIB**
Load library containing the product software. Not required if Tivoli Asset Discovery for z/OS is installed into the system link list.

**HSIACNTL**
Partitioned data set containing fixed length 80 byte records. Member HSIAPARM of this partitioned data set contains the Automation Server control statements that specify the actions to be performed. For each action in the HSIAPARM member, there is a corresponding member of the same name containing the template data for that action. The template data is made up of JCL or an FTP command stream containing symbolic references, to be resolved by the Automation Server when the action is performed.

**HSIACDS**
A VSAM KSDS control data set used by the Automation Server.

**HSIAMSG**
Specifies the message report file for the Automation Server. Initialization statements, error messages, and activity logging messages, are written to this file.

**SYSPRINT**
Specifies the message report file for Language Environment.

**SYSOUT**
Specifies the message report file for Language Environment.

**OUTPUT**
Specifies the message report file for the FTP program. The contents are determined by the FTP program installed in the system.

**INPUT**
Specifies a fixed length 120 byte record file containing FTP commands read by the FTP program. The FTP commands are written to this file before the Automation Server FTP action is performed.

**INTRDR**
Specifies a fixed length 80 byte record file to be directed to the internal

reader used by the system. The Automation Server writes a job stream to this file whenever a JOB action is to be performed.

## Designing request control statements

Automation Server action requests are specified in the HSIAPARM member of the SHSIPARM file.

### Syntax rules

Syntax rules are as follows:

- Records with an asterisk in column 1 are comments.
- Blank records are comments.
- A parameter record has one or more parameters, each with a value specified within parentheses after the parameter name.
- The first parameter specifies the statement type.
- All parameters must begin before column 72.
- Blanks can be used before and after parameter names, parentheses, and parameter values.
- Continuations on to subsequent records are not possible.

### Statement syntax

**Action statement**

Each statement requests that an action is performed for a data set when it matches an associated data set name mask, and is detected for the first time. An action is performed once for each match, but the presence of a data set triggers the action for each specified data set name mask it matches.

Action statements have several optional operands to provide control over when Automation Server processing is to occur.

These operands can specify:

- time-of-day window
- day-of-week control string
- day-of-month window
- month-of-year control string

When all these constraints have been satisfied, the Automation Server searches the catalog for data sets with names that match the masks associated with an action.

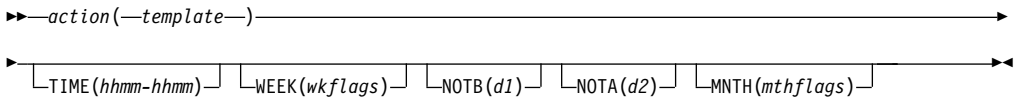
**Data set name mask statement**

Each data set name mask statement associates the specified data set name mask with the preceding action statement. It is invalid for the HSIAPARM member to begin with a data set name mask statement. When a data set with a name matching the specified mask is first located, the action specified in the preceding action statement is triggered.

The data set name mask of NULLFILE is an exception. When a data set name mask with this exact value is processed by the Automation Server, a catalog search is not performed, but the associated action is triggered as if a new cataloged data set matching the mask has been located. Automation Server symbols for the data set name, and for the first qualifier of the data set name have values of the 8 byte string NULLFILE. Use the data set

name mask of NULLFILE to trigger scheduled actions which do not depend on the creation of a particular data set.

### Action statement syntax




***action*** FTP or JOB

***template***
Name of a member in the HSIACNTL file.

**TIME** This operand is optional, and the default is TIME(0000-2400), which specifies no time-of-day constraint.

***hhmm-hhmm***
Specifies a time-of-day range. Each *hhmm* value is four contiguous decimal digits that specify a time-of-day using the 24 hour clock. The minimum value is 0000 and the maximum value is 2400; the last two digits must not exceed 59. The two values are separated by a hyphen. Zero or more additional blanks are also permitted. The first *hhmm* specifies the time-of-day window start, while the second specifies the time-of-day window end. The window includes times which are after the window start and before the window end. However, if the second *hhmm* value is lower than the first, the window includes times which are after the window start or before the window end.

**WEEK** This operand is optional. The default is WEEK(YYYYYYY), which specifies that the *action* can be run on every day of the week.

***wkflags***
Specifies a single contiguous 7 byte string, consisting of the uppercase characters Y or N. Each Y or N corresponds to a day of the week depending on its position in the string; the first corresponding to Sunday, the last to Saturday. If the character corresponding to a day of the week is N, the action is not processed on that day.

**NOTB** This operand is optional. The default NOTB(1) specifies that the monthly window starts on the first possible day of the month. NOTB means "not before".

***d1*** Specifies a one or two digit decimal number in the 1-31 range. This number denotes the first possible day of the month on which the action is permitted.

**NOTA** This operand is optional. The default NOTA(31) specifies that the monthly window extends to the last day of the month. NOTA means "not after".

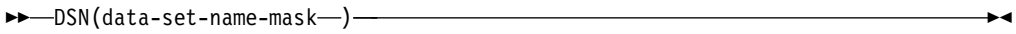
***d2*** Specifies a one or two digit decimal number in the 1-31 range. This number denotes the last possible day of the month on which the action is permitted.

**MNTH**
This operand is optional. The default value enables processing in every month of the year.

***mthflags***
Specifies a single contiguous 12 byte string, consisting of the uppercase characters Y or N. Each Y or N corresponds to a month of the year

depending on its position in the string; the first corresponding to January, the last to December. If the character corresponding to a month of the year is N, the action is not processed in that month.

### DSN statement syntax




**DSN** Data set name.

***data-set-name-mask***
Specifies a data set name mask pattern which does not exceed 44 characters in length, and is used by the Catalog Search Interface. The generic match mask for a single character is the percent sign. The generic match mask variable number of characters is the asterisk. A double asterisk can be used to match a variable number of data set name qualifiers. The catalog search is restricted to entry type A non-VSAM data sets and entry type H generation data sets.

## Control statement examples

**Example 1:**
Files created by the Usage Monitor undergo two independent processes, both within the 8:00 p.m. to 11:30 p.m. window. They are processed by a job based on the JCL contained in member HSISJOB1, and are separately transferred to a z/OS system using the FTP commands in member HSISFTP1. All members are pointed to by the HSIACNTL ddname.

```
* TRANSFER USAGE MONITOR FILES TO Z/OS SYSTEM
   JOB(HSISJOB1)  TIME(2000-2330)
   DSN(USER.OMU*.D*.T*)
   FTP(HSISFTP1)  TIME(2000-2330)
   DSN(USER.OMU*.D*.T*)
```

**Example 2:**
Files created by the Usage Monitor are to be imported to the appropriate database.

```
* PERFORM USAGE MONITOR IMPORT
JOB(HSISUIMP)
DSN(USER.UMON.*.*)
```

In this example HSISUIMP contains the necessary JCL to run Usage Import on a z/OS system.

**Note:** The JCL can route the job to any connected NJE node, or specify an affinity to any system sharing the SPOOL. You do not need to run the job on the z/OS system where the Automation Server is running. The template name, HSISUIMP in this example, does not need to match the job name submitted by the Automation Server action.

**Example 3:**
A job stream stored in member WED2MNTH is to be submitted unconditionally on the second Wednesday of every month.

```
* RUN MONTHLY JOBSTREAM ON THE SECOND WEDNESDAY OF EVERY MONTH
 JOB ( WED2MNTH )   WEEK ( NNNYNNN )   NOTB ( 8 )   NOTA ( 14 )
 DSN ( NULLFILE )
```

**Example 4:**
A job stream stored in member NEWSHIFT is to run every day at 6:00 a.m, 2:00 p.m., and 10:00 p.m.

```
* RUN SHIFT TASK LIST REPORT AT THE START OF EACH SHIFT
 JOB(NEWSHIFT) TIME(0600-0630)
 JOB(NEWSHIFT) TIME(1400-1430)
 JOB(NEWSHIFT) TIME(2200-2230)
```

## Automation Server symbol processing

Whenever an action is performed, the contents of the template member are written to an appropriate output file. Each 80 byte record is written unchanged, unless symbol substitution is required. If an ampersand character is present in a record from the template member, the system symbol substitution routine ASASYMBM is called to process the record before it is written. You can use more than one symbol in a record. If an ampersand character does not denote the start of a recognized symbol, then that part of the data remains unchanged. Symbols available for use in template members include all z/OS system symbols and symbols defined locally by the Automation Server. Most Automation Server local symbols are derived from the catalog entry data set name which, when discovered, triggers the instance of the action.

System symbols supplied by the operating system, as well as the &SMF and &SYSLPAR symbols supplied by the Automation Server, are available for use in the HSIAPARM member. The &SYSLPAR symbol might resolve to a null string if the system is running in a virtual machine.

Automation Server local symbols are provided in the following table:

*Table 18. Automation Server local symbols*

| Symbol | Description |
|---|---|
| &SMF | System SMF identifier. |
| &SYSLPAR | System LPAR name. |
| &DATASETNAME. | The entire data set name. |
| &QUAL1. | The first qualifier of the data set name. |
| &QUAL2. | The second qualifier of the data set name. |
| &QUAL3. | The third qualifier of the data set name. |
| &QUAL4. | The fourth qualifier of the data set name. |
| &QUAL5. | The fifth qualifier of the data set name. |
| &QUAL6. | The sixth qualifier of the data set name. |
| &QUAL7. | The seventh qualifier of the data set name. |
| &QUAL8. | The eighth qualifier of the data set name. |
| &QUAL9. | The ninth qualifier of the data set name. |

**Example:**

The data set triggering a JOB action is IBMUSER.IQ.ZIP. As a result, JCL DD statements referencing the data set in a template member can be represented as shown in this example:

```
//*-------------------------------------------------------------***
//* Sample JCL demonstrating the use of Automation Server local***
//* symbols derived from the data set name.                     ***
//*-------------------------------------------------------------***
//BR14      EXEC  PGM=IEFBR14
//DD1       DD    DSN=&DATASETNAME.,DISP=SHR
//DD2       DD    DSN=&QUAL1..&QUAL2..&QUAL3.,DISP=SHR
```

Both JCL DD statements would be resolved by symbol substitution to:

```
DSN=IBMUSER.IQ.ZIP,DISP=SHR
```

This is the **DSN=** JCL statement output to the internal reader.

As symbol substitution is performed before the job is submitted, z/OS system symbols that cannot be used in batch job JCL, can be used in the Automation Server templates. The symbols are resolved using the system executing the Automation Server, which may not be the system where the submitted job executes.

### Starting and stopping the Automation Server

When you install the Automation Server as a started task, you can run operator commands to start it and stop it. This task requires RACF security access.

#### Procedure

1. Assign RACF CONTROL access to the VSAM data set that is configured for the user ID that is assigned to the Automation Server.
2. Issue the system **START** command to start the Automation Server.
3. Issue the system **STOP** command to stop the Automation Server running as a started task or from running a batch job.

### Excluding data sets

You can exclude data sets from Automation Server processing. To exclude a data set, it must have a record in the Automation Server control data set, with an indication in the record that the data set is already processed.

In the HSIAPARM member you define actions to be performed, and supply data set name masks specifying the data sets to be processed by the Automation Server. Data sets with these name patterns might already exist and have been processed before the Automation Server was implemented.

To exclude a data set, the name of the data must satisfy a selection mask pattern. To implement the exclusion, you can use the Automation Server data set name scouting program. The HSISCUST post-installation job creates the HSIASSCT member in the JCLLIB data set. Run the HSIASSCT job to start the scouting program.

The program reads the HSIAPARM member, searches the catalog for every specified data set name mask, and writes a record for each data set that it discovers. The job then sorts the records into key order and copies them into the VSAM control data set. Every record loaded into the control data set in this way indicates a specific action with a status of complete.

If you want to continue processing some of the data sets, you can manually delete them from the sequential data set before the data is copied into the control data set.

## Automation Server control data set maintenance

A record is kept for every data set processed by the Automation Server in the Automation Server control data set, ACDS. The purpose of this record is to prevent the repeated processing of a data set for the same data set name mask. As records accrue, the size of the data in the ACDS continues to grow.

If a processed data set is deleted, or a data set name mask is removed from the set of masks processed by the Automation Server, then there is no reason to keep a record of that data set in the ACDS. The Automation Server performs a cleanup cycle for the ACDS on a daily basis. This cleanup cycle consists of reading the ACDS sequentially, and deleting records for data sets which have not been found by catalog search. This is based on the relevant data set name mask in the current calendar month, or in the prior calendar month.

As with most VSAM data sets with ongoing record insertion and deletion activity, it is advisable to periodically reorganize the ACDS.

# Chapter 7. Reporting with the Analyzer

The primary reporting facility in IBM Tivoli Asset Discovery for z/OS is the Analyzer.

The Analyzer runs as a started task or batch job on the same z/OS host as the DB2 Subsystem or SQLite database that contains the Tivoli Asset Discovery for z/OS database(s).

The Analyzer has two modes:

**Online mode**
A PC Browser, for example Firefox, is used to communicate with the Analyzer for interactive queries.

**Batch mode**
This mode uses the Analyzer to generate the report to an output file. The Batch mode is useful when you want to automate reports or develop your own reports. Batch mode is also useful when you want to select multiple criteria, such as multiple libraries or multiple users which you cannot do online from some reports.

All Analyzer reports can be run in online and batch modes and can produce the following output formats:

- HTML (htm)
- Excel (Excel)
- Text line (txt)
- Comma Separated Value (csv)

## Analyzer prerequisites

The Analyzer uses the DB2 Call Library Interface (ODBC/CLI), also used by the Inquisitor Import, Usage Import and other batch components, and the z/OS socket application programming interface. For the SQLite database, the Analyzer uses an internal ODBC interface.

There is no dependency on any other middleware components. For example, no dependency exists on the HTTP Server, WebSphere® Application Server, or Java™.

The Analyzer has been designed with minimal prerequisites. These are:

- The Analyzer must be run on the same z/OS host as the DB2 subsystem or SQLite database that contains the Tivoli Asset Discovery for z/OS repositories.
- The user ID of the Analyzer address space must have previously been granted access to the databases. See the HSISGRNT job in the JCLLIB for sample JCL to grant access.
- When running the Analyzer in the online mode, you need access to a TCP/IP port. The default is port 9000.
- When running the Analyzer in online mode with SECURITY=SYSTEM, the Analyzer SHSIMOD1 load library must be defined to the z/OS Authorized Program Facility (APF). In addition, all data sets in the Analyzer STEPLIB, or JOBLIB DD concatenation, must be defined to APF.

- You can run the Analyzer in online mode while Inquisitor Import or Usage Import is also updating data into the repositories. However the Analyzer reports may not display the correct information on the latest updates due to concurrency issues in DB2. To ensure that the latest correct information are displayed, do not run operational jobs that update data in the repositories while users are running reports with the Analyzer. For the SQLite database, this is not an issue as only single thread is allowed.

## Analyzer JCLLIB and PARMLIB members

Several JCLLIB and PARMLIB members are used when you run the Analyzer to generate reports.

The members in the JCLLIB contain sample JCL to run the Analyzer.

*Table 19. JCLLIB members for Analyzer*

| Member | Description |
|---|---|
| HSIJANLO | Analyzer PROC for online mode. Copy this PROC from the JCLLIB to a system PROCLIB data set to run the Analyzer as a started task |
| HSISANLB | Analyzer batch job for batch mode |
| HSISANLO | Analyzer batch job for online mode |
| HSISANS1 | Define the Analyzer security profiles in RACF (only applicable for Analyzer SECURITY=SYSTEM setting) |
| HSISANS2 | Generate the Analyzer SSL certificate in RACF (only applicable for Analyzer SECURITY=SYSTEM setting) |
| HSISANS3 | Connect the Analyzer user ID to an existing SSL certificate in RACF (only applicable for Analyzer SECURITY=SYSTEM setting) |

The following members in the PARMLIB contain sample configuration settings for the Analyzer in online mode. These members are referenced with the TPARAM setting in the HSIJANLO PROC.

*Table 20. PARMLIB members for Analyzer*

| Member | Description |
|---|---|
| HSISANP1 | SECURITY=BASIC<br><br>HTTP communications with basic security |
| HSISANP2 | SECURITY=SYSTEM<br><br>HTTPS (SSL encrypted) communications with z/OS system security (SAF/RACF).<br><br>Refer to members HSISANS1/2/3 in JCLLIB for sample JCL to define RACF profiles/certificates |

## Running the Analyzer in online mode

The primary reporting facility in Tivoli Asset Discovery for z/OS is the Analyzer. You can use the Analyzer in online mode to view reports, run queries, and drill down to related reports.

### About this task

HSISANLO job in the JCLLIB is used to run Analyzer in online mode as a batch job.

```
//HSISANLO EXEC HSIJANLO,TPARAM=HSISANP1
```

To run the Analyzer in online mode as a Started Task, copy the HSIJANLO from the JCLLIB to a system PROCLIB data set.

```
//HSIJANLO PROC HSI='TADZ.V750',       TADz Target library HLQ.
//            HSISCLI='HSISCLI',       DB2 CLI Parms
//            TPARAM='HSISANP1'        TPARAM input parms
//*
//ANALYZER EXEC PGM=HSICANLZ,REGION=0M,TIME=NOLIMIT
//STEPLIB  DD DISP=SHR,DSN=&HSI..SHSIMOD1
//         DD DISP=SHR,DSN=&DB2EXIT
//         DD DISP=SHR,DSN=&DB2LOAD
//SYSPRINT DD SYSOUT=*,HOLD=YES,LRECL=500
//HSIANL1  DD DISP=SHR,DSN=&HSI..SHSIANL1
//HSIANL2  DD DISP=SHR,DSN=&HSI..SHSIANL2
//DSNAOINI DD DISP=SHR,DSN=&HSIINST..PARMLIB(&HSISCLI)
//HSICUST  DD DISP=SHR,DSN=&HSIINST..PARMLIB(HSISANCQ)
//*HSINLS   DD DISP=SHR,DSN=&HSI..SHSIANL1(HSINLSJP)
//TPARAM   DD DISP=SHR,DSN=&HSIINST..PARMLIB(&TPARAM)
```

When the Analyzer is run with online mode, configuration options must be defined in the TPARAM DD, including the communication port and security mode.

## Analyzer communication port

### About this task

The Analyzer communication port is defined by using the HTTPPORT setting. Both sample PARMLIB members HSISANP1 (basic security), and HSISANP2 (system security), have the following:

```
************************************************************************
* HTTPPORT defines the TCP/IP port used for communications.           *
*                                                                      *
*   If HTTPPORT = 9000  is defined on a system with a TCP/IP host      *
*   called sys1.mycompany.com, to access the TADz Analyzer the user    *
*   would specify the following URL in their PC Browser:               *
*        http://sys1.mycompany.com:9000  if SECURITY=BASIC             *
*    or https://sys1.mycompany.com:9000  if SECURITY=SYSTEM            *
*                                                                      *
*   The port specified must be available on your system.               *
*                                                                      *
*   TSO NETSTAT can be used to check if a port is available e.g.:      *
*   TSO NETSTAT (PORT 9000       --*  is port 9000 in use?            *
*   TSO NETSTAT PORTL(PORT 9000  --*  is port 9000 reserved?          *
*                                                                      *
*   If no entries are returned from these NETSTAT commands, the port   *
*   is most probably available.  At some sites, you may need your      *
*   Network Systems Programmer to reserve a port for TADz Analyzer.    *
*                                                                      *
************************************************************************
HTTPPORT = 9000
```

If HTTPPORT is not specified, or is set to 0, the Analyzer runs in batch mode instead of in online mode

## Analyzer security

You can view Analyzer reports in a web browser, such as Firefox, and you can communicate with the Analyzer utility to perform interactive queries.

Some of the Analyzer reports contain a large amount of information and it is recommended that you use a screen resolution of at least 1440 x 900 pixels to view them.

The following table describes the security modes that you can configure for accessing Analyzer online.

*Table 21. Security modes for accessing Analyzer online*

| Security configuration | Communication mode | Access ID and password | Access permissions |
|---|---|---|---|
| SECURITY=BASIC | HTTP | Standard user ID and password. Default values are:<br>• User: `tadzusr` and password `TADZ`<br>• Admin: `tadzadm` and password `TADZ` | User ID tadzusr has limited access and user ID tadzadm has full access |
| SECURITY=SYSTEM | HTTPS | z/OS system user ID and password<br><br>Default: User TSO ID and password | Depends on access given to TSO ID |

## Analyzer BASIC security

HSISANP1 in the PARMLIB defines basic user ID security settings for running the Analyzer.

User IDs TADZADM and TADZUSR can be used without any prior configuration. User ID AUID001 is a sample of how to restrict a user ID to certain databases.

```
************************************************************************
* SECURITY=BASIC  - HTTP communications                                *
*                     with basic security defined in TPARAM DD         *
*                                                                      *
************************************************************************
 SECURITY = BASIC

************************************************************************
* The following settings are only applicable for                       *
* SECURITY=BASIC:                                                      *
*                                                                      *
*   AUTH_USER defines Userids and passwords for Analyzer logon         *
*   AUTH_DB   defines the databases access                             *
*   AUTH_MENU defines the menus access                                 *
*                                                                      *
* The sample settings profile:                                         *
*  - TADZADM userid:                                                   *
*       - Password TADZ                                                *
*       - Access to all databases                                      *
*       - Access to all menu tabs                                      *
*  - TADZUSR userid:                                                   *
*       - Password TADZ                                                *
*       - Access to all databases                                      *
*       - Access to menu tabs ASSET, DISC + CUSTOM only (not ADMIN)    *
*  - AUID001 userid:                                                   *
*       - Password PW01                                                *
*       - Access to databases AUDB01 + AUDB02 only                     *
*       - Access to menu tab ASSET only                                *
************************************************************************
```

```
*--------------------------------------------------------------------
*AUTH_USER= USERID   , PASSWORD                                      *
*--------------------------------------------------------------------
AUTH_USER = TADZADM  , TADZ
AUTH_USER = TADZUSR  , TADZ
AUTH_USER = AUID001  , PW01

*--------------------------------------------------------------------*
*AUTH_DB  = DATABASE , LIST OF USERIDS AUTHORIZED TO SEE THE DATABASE *
*--------------------------------------------------------------------*
AUTH_DB   = *        , TADZADM TADZUSR
AUTH_DB   = AUDB01   , AUID001
AUTH_DB   = AUDB02   , AUID001

*--------------------------------------------------------------------*
*AUTH_MENU= MENU_TAB , LIST OF USERIDS AUTHORIZED TO SEE THE MENU TAB *
*--------------------------------------------------------------------*
AUTH_MENU = ASSET    , TADZADM TADZUSR AUID001
AUTH_MENU = DISC     , TADZADM TADZUSR
AUTH_MENU = ADMIN    , TADZADM
AUTH_MENU = CUSTOM   , TADZADM TADZUSR
```

## Analyzer SYSTEM security

HSISANP2 in the PARMLIB defines the system security settings for running the Analyzer.

The following system security settings are defined:

```
**********************************************************************
* SECURITY=SYSTEM - HTTPS (SSL encrypted) communications             *
*                    with z/OS system security (SAF/RACF).           *
*                    Refer to HSISANS1/2/3 in JCLLIB for sample JCL  *
*                    to define RACF profiles/certificates.           *
*                                                                    *
**********************************************************************
SECURITY = SYSTEM

**********************************************************************
* The following settings are only applicable for                     *
* SECURITY=SYSTEM:                                                   *
*                                                                    *
*   AUTH_HLQ         defines SAF/RACF profile high level qualifier   *
*                                                                    *
*   AUTH_UPPERCASE=Y Analyzer will uppercase passwords when          *
*                    invoking SAF/RACF password authentication       *
*   AUTH_UPPERCASE=N Analyzer will pass through mixed case passwords *
*                    when invoking SAF/RACF password authentication  *
*                                                                    *
*   GSK_KEYRING_FILE defines SAF/RACF Keyring name of SSL Certificate *
*   GSK_KEY_LABEL    defines SAF/RACF Label   name of SSL Certificate *
*   GSK_....         defines optional z/OS SSL environment variables. *
*                    The z/OS Cryptographic Cryptographic Services   *
*                    Secure Sockets Layer Programming manual         *
*                    SC24-5901-07 explains the environment variables. *
*                    For example, define GSK_HW_CRYPTO = 32          *
*                    for SHA-256 digest generation.                  *
*                                                                    *
* JCLLIB(HSISANS1) contains sample JCL to define RACF profiles, using *
* a high level qualifier of 'TADZ'.  If you have changed HSISANS1,   *
* you may also need to change the AUTH_HLQ TPARAM setting.           *
*                                                                    *
* JCLLIB(HSISANS2/3) contains sample JCL to define RACF SSL          *
* Certificates.  If you have changes HSISANS2/3, you may also need to *
* change the GSK_KEYRING_FILE and GSK_KEY_LABEL TPARAM settings.     *
*                                                                    *
**********************************************************************
```

```
AUTH_HLQ         = TADZ
AUTH_UPPERCASE   = Y
GSK_KEYRING_FILE = TADZ_KEYRING
GSK_KEY_LABEL    = TADZCERT
```

HSISANS1 in the JCLLIB has sample JCL to define RACF security profiles.

**Note:** The RACF ID can be an existing RACF group (which user IDs have been connected to) and/or existing RACF user IDs.

If your z/OS system has been set up to use a third party alternative to RACF, you must define comparable settings in your third party security product.

```
/*-------------------------------------------------------------*/
/* TADZ ANALYZER DATABASE PROFILES                             */
/*-------------------------------------------------------------*/
 RDELETE FACILITY  TADZ.DB.AU*
 RDEFINE FACILITY  TADZ.DB.AU*           UACC(NONE)
 PERMIT            TADZ.DB.AU*           ACCESS(READ) -
   CLASS(FACILITY) ID(TADZADM,TADZUSR,AUID001)

 RDELETE FACILITY  TADZ.DB.*
 RDEFINE FACILITY  TADZ.DB.*             UACC(NONE)
 PERMIT            TADZ.DB.*             ACCESS(READ) -
   CLASS(FACILITY) ID(TADZADM,TADZUSR)
 PERMIT            TADZ.DB.*             ACCESS(NONE) -
   CLASS(FACILITY) ID(AUID001)

/*-------------------------------------------------------------*/
/* TADZ ANALYZER MENU PROFILES                                 */
/*-------------------------------------------------------------*/
 RDELETE FACILITY  TADZ.MENU.ASSET
 RDEFINE FACILITY  TADZ.MENU.ASSET       UACC(NONE)
 PERMIT            TADZ.MENU.ASSET       ACCESS(READ) -
   CLASS(FACILITY) ID(TADZADM,TADZUSR,AUID001)

 RDELETE FACILITY  TADZ.MENU.DISC
 RDEFINE FACILITY  TADZ.MENU.DISC        UACC(NONE)
 PERMIT            TADZ.MENU.DISC        ACCESS(READ) -
   CLASS(FACILITY) ID(TADZADM,TADZUSR)

 RDELETE FACILITY  TADZ.MENU.ADMIN
 RDEFINE FACILITY  TADZ.MENU.ADMIN       UACC(NONE)
 PERMIT            TADZ.MENU.ADMIN       ACCESS(READ) -
   CLASS(FACILITY) ID(TADZADM)

 RDELETE FACILITY  TADZ.MENU.CUSTOM
 RDEFINE FACILITY  TADZ.MENU.CUSTOM      UACC(NONE)
 PERMIT            TADZ.MENU.CUSTOM      ACCESS(READ) -
   CLASS(FACILITY) ID(TADZADM,TADZUSR)

  SETROPTS RACLIST(FACILITY) REFRESH
```

## SSL Certificates

When the Analyzer is running with SYSTEM=SECURITY, you must have an SSL Certificate defined in your SAF/RACF security system. You can either generate your own certificate, or connect to an existing certificate.

HSISANS2 in JCLLIB has sample JCL to generate SSL certificates in RACF.

```
//*****************************************************************
//*                                                               *
//* To enable TADz Analyzer to use HTTP secure (HTTPS) the following  *
//* steps should be implemented by your site's RACF Administrator:    *
//* 1. Delete KEYRING(TADZ_KEYRING) and certificate with the          *
```

```
//*    LABEL('TADZCERT').                                          *
//* 2. Activate RACF Classes required for digital certificates.    *
//* 3. Define Keyring TADZ_KEYRING.                                *
//* 4. Generate certificate.                                       *
//* 5. Connect to Keyring.                                         *
//* 6. Refresh RACF Classes required for digital certificates.     *
//* 7. Permit access to the Facility Class profiles.               *
//*                                                                *
//*                                                                *
//* The following JCL demonstrates a sample implementation:        *
//* 1. Update all occurrences of "Userid-running-HSISANLO" to reflect *
//*    your TADz HTTPS environment.                                *
//*                                                                *
//* Do not change the RACF keyring 'TADZ_KEYRING' or label 'TADZCERT' *
//* unless you update the corresponding values in analyzer PARMLIB *
//* member HSISANP2 and restart the Analyzer STC/Job.              *
//*----------------------------------------------------------------*
//RACFDEF    EXEC  PGM=IKJEFT01,DYNAMNBR=30
//SYSTSPRT   DD    SYSOUT=*
//SYSTSIN    DD    *
 PROF NOPREF

 RACDCERT DELETE(LABEL('TADZCERT'))
 RACDCERT ID(CMACN) DELRING(TADZ_KEYRING)

 SETROPTS CLASSACT(DIGTCERT,DIGTNMAP)
 SETROPTS RACLIST(DIGTCERT,DIGTNMAP)

 RACDCERT ID(Userid-running-HSISANLO) ADDRING(TADZ_KEYRING)

 RACDCERT ID(Userid-running-HSISANLO) CERTAUTH GENCERT -
 SUBJECTSDN( O('Your Organization')   -
 CN('Your Domain')                    -
 C('US')) TRUST                       -
 WITHLABEL('LOCALCA')                 -
 KEYUSAGE(CERTSIGN)

 RACDCERT ID(Userid-running-HSISANLO) GENCERT -
 SUBJECTSDN (CN('TADZCERT')                   -
 OU('Your Dept.')                             -
 C('US'))                                     -
 WITHLABEL('TADZCERT')                        -
 SIGNWITH(CERTAUTH                            -
 LABEL('LOCALCA'))

 RACDCERT ID(Userid-running-HSISANLO)         -
 CONNECT(ID(Userid-running-HSISANLO)          -
 LABEL('TADZCERT')                            -
 RING(TADZ_KEYRING)                           -
 DEFAULT                                      -
 USAGE(PERSONAL))

 RACDCERT ID(Userid-running-HSISANLO)         -
 CONNECT(ID(Userid-running-HSISANLO) CERTAUTH -
 LABEL('LOCALCA')                             -
 RING(TADZ_KEYRING)                           -
 USAGE(CERTAUTH))

 SETROPTS RACLIST(DIGTCERT,DIGTNMAP) REFRESH
/*
//PERMIT     EXEC  PGM=IKJEFT01,DYNAMNBR=30
//SYSTSPRT   DD    SYSOUT=*
//SYSTSIN    DD    *
 PROF NOPREF

  RDEL FACILITY IRR.DIGTCERT.LIST
  RDEL FACILITY IRR.DIGTCERT.LISTRING
```

```
  SETR RACLIST(FACILITY) REFRESH

  RDEFINE FACILITY IRR.DIGTCERT.LIST  UACC(NONE)
  RDEFINE FACILITY IRR.DIGTCERT.LISTRING  UACC(NONE)
  SETR RACLIST(FACILITY) REFRESH

  PERMIT IRR.DIGTCERT.LIST CLASS(FACILITY)     -
  ID(Userid-running-HSISANLO) AC(READ)

  PERMIT IRR.DIGTCERT.LISTRING CLASS(FACILITY) -
  ID(Userid-running-HSISANLO) AC(READ)

  SETR RACLIST(FACILITY) REFRESH
/*
```

HSISANS3 in JCLLIB has sample JCL to connect to existing SSL certificates in RACF.

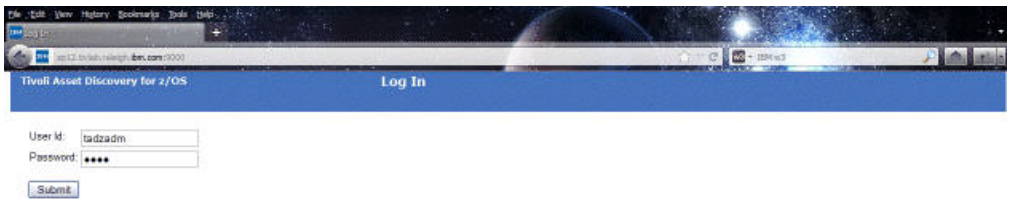
```
//*********************************************************************
//*                                                                   *
//* To enable TADz Analyzer to use HTTP secure (HTTPS) using an       *
//* existing CA certificate, 'Entrust Secure Server Root CA' in our   *
//* example, the following steps should be implemented by your site's *
//* RACF Administrator:                                               *
//*                                                                   *
//* 1. Delete KEYRING(TADZ_KEYRING) and certificate with the          *
//*    LABEL('TADZCERT').                                             *
//* 2. Activate RACF Classes required for digital certificates.       *
//* 3. Define Keyring TADZ_KEYRING.                                   *
//* 4. Connect the existing CA certificate to the Keyring.            *
//* 5. Refresh RACF Classes required for digital certificates.        *
//* 6. Permit access to the Facility Class profiles.                  *
//*                                                                   *
//*                                                                   *
//* The following JCL demonstrates a sample implementation:           *
//* 1. Update all occurrences of "Userid-running-HSISANLO" to reflect *
//*    your TADz HTTPS environment.                                   *
//*                                                                   *
//* Do not change the RACF keyring 'TADZ_KEYRING' or label 'TADZCERT' *
//* unless you update the corresponding values in analyzer PARMLIB    *
//* member HSISANP2 and restart the Analyzer STC/Job.                 *
//*-------------------------------------------------------------------*
//RACFDEF    EXEC  PGM=IKJEFT01,DYNAMNBR=30
//SYSTSPRT   DD    SYSOUT=*
//SYSTSIN    DD    *
 PROF NOPREF

 RACDCERT DELETE(LABEL('TADZCERT'))
 RACDCERT ID(CMACN) DELRING(TADZ_KEYRING)

 SETROPTS CLASSACT(DIGTCERT,DIGTNMAP)
 SETROPTS RACLIST(DIGTCERT,DIGTNMAP)

 RACDCERT ID(Userid-running-HSISANLO) ADDRING(TADZ_KEYRING)

 RACDCERT ID(Userid-running-HSISANLO) GENCERT -
 SUBJECTSDN (CN('TADZCERT')                   -
 OU('Your Dept.')                             -
 C('US'))                                     -
 WITHLABEL('TADZCERT')

 RACDCERT ID(Userid-running-HSISANLO)         -
 CONNECT(ID(Userid-running-HSISANLO)          -
 LABEL('TADZCERT')                            -
 RING(TADZ_KEYRING)                           -
 DEFAULT                                      -
 USAGE(PERSONAL))
```

```
 RACDCERT ID(Userid-running-HSISANLO)          -
 CONNECT(ID(Userid-running-HSISANLO) CERTAUTH -
 LABEL('Entrust Secure Server Root CA')        -
 RING(TADZ_KEYRING)                            -
 USAGE(CERTAUTH))

 SETROPTS RACLIST(DIGTCERT,DIGTNMAP) REFRESH
/*
//PERMIT     EXEC  PGM=IKJEFT01,DYNAMNBR=30
//SYSTSPRT   DD    SYSOUT=*
//SYSTSIN    DD    *
 PROF NOPREF

  RDEL FACILITY IRR.DIGTCERT.LIST
  RDEL FACILITY IRR.DIGTCERT.LISTRING
  SETR RACLIST(FACILITY) REFRESH

  RDEFINE FACILITY IRR.DIGTCERT.LIST  UACC(NONE)
  RDEFINE FACILITY IRR.DIGTCERT.LISTRING  UACC(NONE)
  SETR RACLIST(FACILITY) REFRESH

  PERMIT IRR.DIGTCERT.LIST CLASS(FACILITY)     -
  ID(Userid-running-HSISANLO) AC(READ)

  PERMIT IRR.DIGTCERT.LISTRING CLASS(FACILITY) -
  ID(Userid-running-HSISANLO) AC(READ)

  SETR RACLIST(FACILITY) REFRESH
/*
```

## Online login to the Analyzer

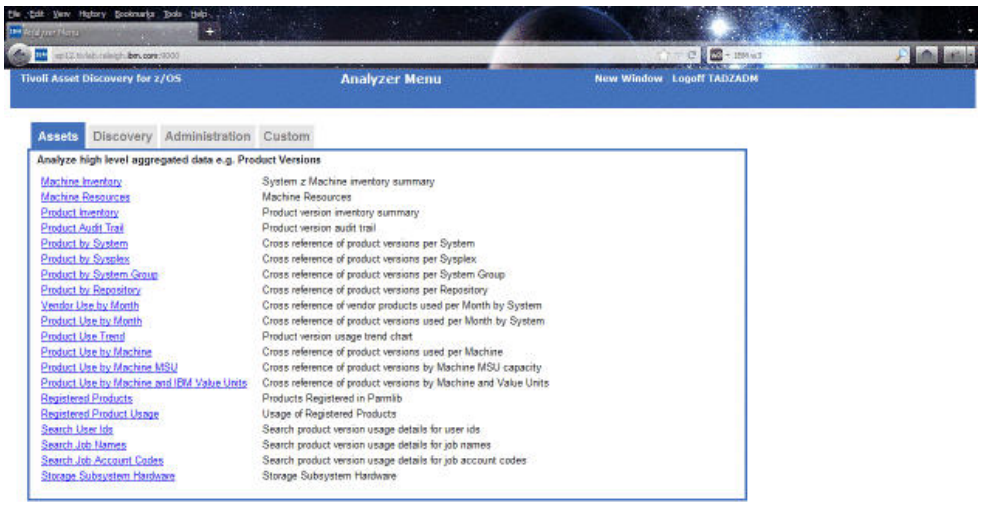
With the Analyzer reporting utility, you can log in with a browser to gain access to the Analyzer Asset, Discovery, and Administration reports and to any Custom reports that you create.

To access the Analyzer online, enter the URL including the host name and port number, in the address bar of a browser. The example URL in the following image is `sp12.tivlab.raleigh.ibm.com:9000`, and provides the user ID and password that are associated with the default basic security option.

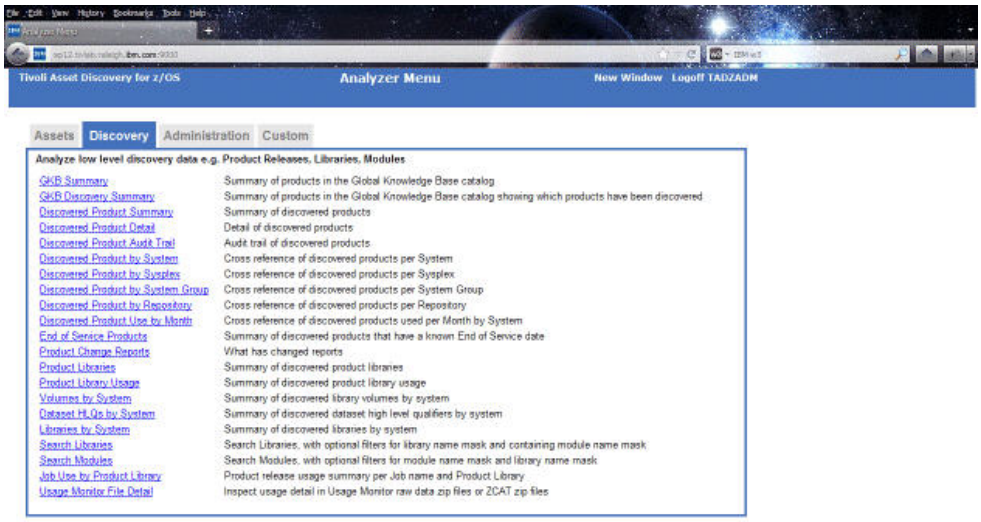



When you login to the Analyzer online, the Analyzer Menu window includes the following tabs:

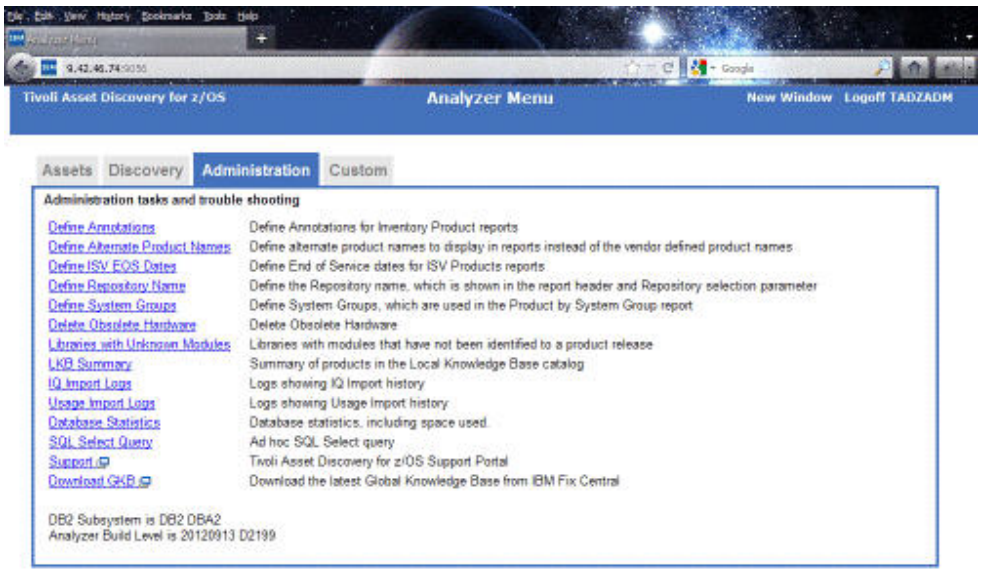
- The **Assets** tab contains reports that query high level aggregated data, such as product versions. This level of data is useful if you are reconciling product licenses.



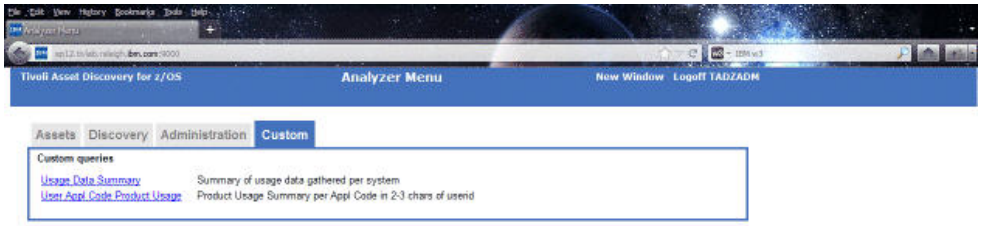

- The **Discovery** tab contains reports that query low-level discovery data, such as product releases, libraries, and modules. This level of data is useful if you support z/OS systems.




- The **Administration** tab contains administration tasks and troubleshooting reports. These reports are designed for Tivoli Asset Discovery for z/OS administrators and users only see this menu if they are granted specific access.




- The **Custom** tab contains your local custom reports. Two example custom reports are provided.




From any of the tabs, when you click the link to a report, the next window opens that contains parameter selection lists based on the data in your database. Select items in the parameter lists to construct a query. Hold down the Ctrl or Shift key to select multiple items from a list. When you have selected all required parameters, click **Submit** to run the query.

At the end of every report, the report name and parameters are shown in the same syntax that you can copy and paste into the HSISANLB batch job SYSIN DD deck to run the report in batch mode.

When you construct a query, if you choose the option `Output format and select Browser` as the output format, the report includes hyperlinks that you can use to drill down for more information.

You can download the content of a report, including the embedded content, in the following file formats:

- Excel
- HTML
- Comma separated value (CSV)
- Text (txt)

## Controlling the Analyzer address space

The Analyzer supports several z/OS modify commands, including STOP, REFRESH, and TRACE.

The following tables shows the z/OS modify commands that the Analyzer supports.

*Table 22. z/OS modify commands*

| Command | Description |
|---|---|
| STOP | Stops the Analyzer address space. For example `/F HSISANLO,STOP`<br><br>You can also issue this via the z/OS Stop command `/P HSISANLO` |
| REFRESH | Refresh Analyzer report templates and NLS text. For example `/F HSISANLO,REFRESH`. This is typically used to load new Custom queries |
| TRACE | Toggles on/off tracing. For example/`F HSISANLO,TRACE`. This should only be used when requested by IBM Support. |

## Running the Analyzer in batch mode

If you want to automate report generation, you can run the Analyzer in batch mode.

HSISANLB in JCLLIB contains sample JCL.

```
//*
//   SET OUTFMT=TXT
//*  SET OUTFMT=XLS
//*  SET OUTFMT=CSV
//*  SET OUTFMT=HTM
//*
//   SET OUTDSN=&SYSUID..TADZANLZ.&OUTFMT  Output dsn
//*
//ALLOC   EXEC PGM=IEFBR14
//OUTDSN   DD DISP=(MOD,CATLG),DSN=&OUTDSN,
//            DCB=(DSORG=PS,RECFM=VB,LRECL=1000,BLKSIZE=0),
//            UNIT=SYSALLDA,SPACE=(CYL,(5,10))
//*
//ANALYZER EXEC PGM=HSICANLZ
//STEPLIB  DD DISP=SHR,DSN=HSIDEV.V810.SHSIMOD1
//         DD DISP=SHR,DSN=DB2V10.DEA1.SDSNEXIT
//         DD DISP=SHR,DSN=DB2V10.SDSNLOAD
//         DD DISP=SHR,DSN=CEE.SCEERUN
//         DD DISP=SHR,DSN=CBC.SCLBDLL
//SYSPRINT DD SYSOUT=*,HOLD=YES,LRECL=500
//HSIANL1  DD DISP=SHR,DSN=HSIDEV.V810.SHSIANL1
//HSIANL2  DD DISP=SHR,DSN=HSIDEV.V810.SHSIANL2
//DSNAOINI DD DISP=SHR,DSN=HSIDEV.V810.PARMLIB(HSISCLI)
//HSICUST  DD DISP=SHR,DSN=HSIDEV.V810.PARMLIB(HSISANCQ)
//*HSINLS   DD DISP=SHR,DSN=HSIDEV.V810.SHSIANL1(HSINLSJP)
//WORK0      DD DSN=&&WORK0,DISP=(NEW,DELETE),
//              UNIT=SYSALLDA,SPACE=(CYL,(100,50),RLSE)
//TPARAM   DD DUMMY
//OUTPUT1  DD DISP=OLD,DSN=&OUTDSN
//SYSIN    DD *
/asset/machine_inventory
show1pars   = on
showconfig  = on
respository = TADZREPZ
/*
```

The report name and parameters are specified in the SYSIN DD and the output goes to the OUTPUT1 DD.

The simplest way to know what report name and parameters to specify is to run the report first using Analyzer in online mode. At the end of every report, the

report name and parameters are listed in the syntax needed for batch mode. You can cut and paste this syntax into the batch SYSIN DD.

Alternatively, you can directly type in the parameters. Wildcard filters have been enabled to assist in this case.

## Analyzer globalization support

By default, the Analyzer uses English for all report titles, headings, and descriptions. To change to Japanese, define the HSINLS DD to point to the HSINLSJP member in the SHSIANL1 data set.

You can also define your own custom language settings with the HSINLS DD. HSINLSEN member in SHSIANL1 data set provides a template. It contains English key phrases that are assigned to text that is used on the reports.

# Chapter 8. Running the utilities provided with Tivoli Asset Discovery for z/OS

Tivoli Asset Discovery for z/OS provides utilities that you run to perform routine functional tasks to maintain the product lifecycle.

## Condensing usage data with the ZCAT utility

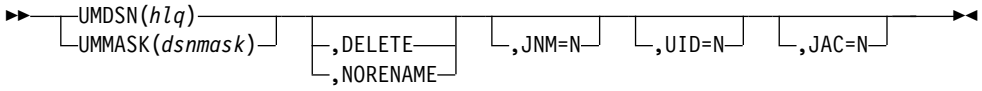
The ZCAT utility concatenates and condenses Usage Monitor data sets and generates a file that is then processed by the Usage Import program. When you condense the data produced by the Usage Monitor program, you can save storage space and improve the performance of the Usage Import program.

The Usage Monitor started task produces at least one usage data set per day. You can design a work flow that runs the ZCAT utility on the data sets on a weekly, fortnightly, or monthly basis before the Usage Import program processes them. Running the ZCAT utility on a weekly basis is useful, but depends on the amount of data that is produced and processed at your site. The Usage Monitor program collects detail about which job, account ID, and user ID are using each module of a particular library on a specified date. This information is output into multiple files that are produced on a daily basis. The ZCAT utility condenses the files in the following manner:

- Usage data across multiple files is condensed to a monthly granularity, as are the records stored in the Repository database.
- Redundant records in files and records that are not stored in the database, are omitted.
- Optionally, condensation can apply to user IDs, job names, or account ID details.
- The ZCAT output file is compressed and ready to be transmitted for Usage Import processing.

The following diagram shows the syntax of program parameters to run the ZCAT utility.

**ZCAT utility syntax**




### Mandatory parameters

The UMDSN or the UMMASK parameter must be specified.

**UMDSN**(*hlq*)
*hlq* is the Usage Monitor data set high-level qualifier. When the **UMDSN** parameter is specified, ZCAT concatenates all data sets having names of hlq.Dyyyyddd.Thhmmsst where yyyyddd and hhmmsst are the timestamp patterns of data sets produced by the Usage Monitor. The *hlq* can contain wildcard characters of percent or asterisk. The percent character denotes a

single character mask, and the asterisk character denotes all characters. For example UMDSN(TADZ.**) would search for all data set names of TADZ.**.D%%%%%%%.T%%%%%%%

**UMMASK(** *dsnmask*)
*dsnmask* is the full dsn mask search criteria. It can be used to search for a pattern of files that differ from the files produced by the Usage Monitor. This parameter is useful if the files produced by the Usage Monitor have been renamed, but still need processing. Specifying UMMASK(hlq.D%%%%%%%.T%%%%%%%) is equivalent to specifying UMDSN(hlq)

## Optional parameters

One or more optional parameters can follow the mandatory parameters.

**DELETE**
Delete the input data sets after the output data set is successfully generated. The default is to retain the input data sets, which are renamed by default.

**NORENAME**
Do not rename input data sets from hlq.D*.T* to hlq.D*.S* after the output data set is successfully generated. The default is to rename these input data sets to stop them being reprocessed by the ZCAT utility. Use this option only to rename the data sets before further ZCAT processing. This option stops double counting of usage data. This parameter is automatically set when UMMASK is used.

The RENAME option is ignored, if **DELETE** is also specified.

## Optional condensation parameters

Improvements in performance and data storage space are gained by using the ZCAT utility options to carry out further condensation of data, ignoring data differences that are not important at your site, and do not appear in your regular reporting. You can still point the Usage Monitor File Detail Report to the saved archive of the concatenated detail file (ZCATDETL), or to the Usage Monitor output files. ZCATDETL is produced by the ZCAT utility.

**JNM**=*N*
Condense data for different job names to one of the following generic names based on the job type: -STC-, -JOB-, -TSO- or -SYS-. The default is to retain the job name, and to condense data that belongs to the same job name only. To maximize aggregation of records, the shipped version of the HSISZCAT sample job specifies N.

**UID**=*N*
Condense data for different user IDs, which are converted to blank. The default is to retain the user ID, and to condense data that belongs to the same user ID only. To maximize aggregation of records, the shipped version of the HSISZCAT sample job specifies N.

**JAC**=*N*
Condense data for different job account codes which are converted to blank. The default is to retain the job account code, and to condense data that belongs to the same job account code only. To maximize aggregation of records, the shipped version of the HSISZCAT sample job specifies N.

**Note:** Due to the various consolidating options, records that are ignored in the ZCATOUT data set are still written to the ZCATDETL output data set, which can be retained for archiving.

### DD statements

**ZCATOUT**

Specifies the name of the ZCAT output data set. This data set can then be used as the input to the Usage Import program, where usage details are imported into the database. If the ZCATOUT DD card is omitted, ZCAT by default writes to a data set having the name hlq.Dyyyyddd.Uhhmmsst (U instead of T implied by the high level qualifier (hlq) option for input data sets), where yyyyddd and hhmmsst refer to the date and time timestamp of the first processed input data set.

**ZCATDETL**

If the ZCATDETL DD is allocated, all records are written to this data set. This data set includes any non condensed and non diagnostic data that is not written to the ZCATOUT data set. It enables the Job name, user ID, and job account details (which are ignored due to ZCAT options and are not written to the ZCATOUT file) to be archived into the detail file.

The ZCATDETL and ZCATOUT data sets are compressed data sets written by the ZCAT utility.

```
//ZCAT     EXEC PGM=HSIZCAT,PARM='UMDSN(TADZ.**),JNM=N'
//STEPLIB  DD  DISP=SHR,DSN=TADZ.V810.SHSIMOD1
//ZCATOUT  DD  DSN=&SYSUID..TADZ.ZCATOUT,
//             DISP=(NEW,CATLG),UNIT=SYSDA,SPACE=(CYL,(50,50),RLSE),
//             DCB=(DSORG=PS,RECFM=VB,LRECL=27994,BLKSIZE=27998)
//ZCATDETL DD  DSN=&SYSUID..TADZ.ZCATDETL,
//             DISP=(NEW,CATLG),UNIT=SYSDA,SPACE=(CYL,(50,50),RLSE),
//             DCB=(DSORG=PS,RECFM=VB,LRECL=27994,BLKSIZE=27998)
//SYSPRINT DD  SYSOUT=*,HOLD=YES
```

In this example, all data sets having names of TADZ.*.*.D%%%%%%%.T %%%%%%% are processed due to the **UMDSN** parameter. The condensed output is written to sysuid.TADZ.ZCATOUT where the SYSUID system symbol is the user ID of the person submitting the job. This file is then transmitted for Usage Import processing. The **JNM=N** parameter instructs the utility to condense job names and ignore the original job name distinction. All valid records are written to the ZCATDETL DD card sysuid.TADZ.CATDETL, which is then archived for reference purposes.

## Summarizing usage data with the Usage Summary utility

The Usage Summary utility summarizes usage data in the repository. The process deletes detailed usage records and creates monthly summary records by reducing the number of DB2 rows used to represent your old data. After each time that you run the Usage Summary utility, the usage data is aggregated to update the asset tables.

To minimize space utilization and improve SQL query performance, it is recommended that you keep detailed module usage data for the last three months and summarize all detailed module usage data older than three months. It is also recommended to delete summarized module usage data older than 18 months. Please see job HSISUDEL (Usage Deletion) for more details.

If you have not run the Usage Summary job for some time, then select a period of a few months at a time, in order to keep the run times down to a reasonable time.

## Running the Usage Summary utility

To run the Usage Summary, use the job HSISUSUM, in the JCLLIB. This job is generated from the HSISCUST post-installation customization job.

## TPARAM parameters

**COMMIT=**
Default is 1000. Number of records stored before issuing of COMMIT.

**DSN=** DB2 location. Value assigned, as defined in job HSISCUST.

**SUMBY=**
Use this parameter to condense the usage data.

**SUMBY=1**
Data is summarized at the Product, LPAR, Period, Module ID, and Job Type level.

User ID and job ID distinctions are ignored. Instead of Job IDs, events are attributed to Job Types (BATCH, TSO, DB2...).

**SUMBY=2**
Data is summarized at Product, LPAR, Period, Job ID, User ID level.

Load module and program names are ignored.

**SUMBY=3**
The rules for SUMBY=1 and SUMBY=2 apply.

Data is summarized by Product, LPAR, Period, Job Type.

**KEEPDETAIL=**
Default is 2. Number of months prior to the current month for which usage records are not summarized. Prior usage records are summarized. If KEEPDETAIL=0 is specified, all usage records, excluding those records for the current month, are summarized.

**FIRSTDATE=**
Start of the first date range. This is in the form YYYYMM. Only complete months are chosen.

**LASTDATE=**
End of the last date range. This is in the form YYYYMM.

**Note:** The date range of summarization is inclusive of the month specified in the FIRSTDATE and LASTDATE parameters.

**MINUSAGETHRESHOLD**
Default is 1000. Sets a value for Usage Summary to ignore summarization of usage records. If this parameter is set to 1000, then any product with a usage count of 1000 or less for any given month, does not have its usage records summarized. This allows you to view the usage records for low usage products.

**REPSCHEMA=**
Repository qualifier. Name of qualifier is *&REPZSCHM*.

# Deleting usage data with the Usage Deletion utility

You use the Usage Deletion utility to delete detailed, summarized, and aggregated usage data for a specified period for all systems in the repository. Each time you run the utility, usage data is aggregated to update the asset tables.

To minimize space utilization and improve SQL query performance, keep no more than 3 months of detailed module usage data and 13 months of aggregated product usage data. If you want to keep more than 3 months of detailed module usage data, run job HSISUSUM (Usage Summary) to summarize the detailed module usage data older than three months.

If you do not run the Usage Deletion utility for some time, select a period of a few months, in order to keep the run times down to a reasonable time.

## Running the Usage Deletion utility

To run the Usage Deletion, use the job HSISUDEL, in the JCLLIB. This job is generated from the HSISCUST post-installation customization job.

## TPARAM parameters

**COMMIT=**
Default is 1000. Number of records stored before issuing of COMMIT.

**DSN=** DB2 location. Value assigned, as defined in job HSISCUST.

**KEEPDETAIL=**
Default is 2. Number of months prior to the current month for which detailed and summarized module usage data are kept. KEEPDETAIL=0 means all detailed and summarized module usage data excluding those from the current month are deleted.

**KEEPAGGR=**
Default is 12. Number of months prior to the current month for which aggregated product usage data are kept. KEEPAGGR=0 means all aggregated product usage data, excluding those from the current month are deleted.

**FIRSTDATE=**
Start of the first date range. This is in the form YYYYMM. Only complete months are chosen.

**LASTDATE=**
End of the last date range. This is in the form YYYYMM.

**Note:** The date range of deletion is inclusive of the month specified in the FIRSTDATE and LASTDATE parameters.

**REPSCHEMA=**
Repository qualifier. Name of qualifier is *&REPZSCHM*.

**SID=** System Identifier of system for which usage should be deleted. Specify SID=ALLSIDS to delete usage data for all SIDs.

**Note:** If KEEPDETAIL is set to a value, then FIRSTDATE / LASTDATE will be ignored. If detailed usage data are to be deleted within a certain date range, then comment out KEEPDETAIL and define dates for FIRSTDATE / LASTDATE. For further details, please see comments described in job HSISUDEL.

## Deleting a specific system with the System Deletion utility

It can be necessary to delete data for a specified system to reconcile data and delete unreferenced records. The System Deletion utility deletes discovery, usage, and hardware data for a specified system.

Because libraries that are shared with other systems are not deleted, data for a specified system can become outdated.

You can also use the System Deletion utility to delete a system that was accidentally imported into the repository or to delete a system that is decommissioned.

### Running the System Deletion utility

To run the System Deletion, use the job HSISLDEL, in the JCLLIB. This job is generated from the HSISCUST post-installation customization job.

### TPARAM parameters

**DSN=** DB2 location. Value assigned, as defined in job HSISCUST.

**REPSCHEMA=**
Repository qualifier. Name of qualifier is &REPZSCHM.

**SID=** System Identifier of system to be deleted.

## Listing high-level qualifiers for the Usage Monitor utility

Tivoli Asset Discovery for z/OS collects large amounts of usage data. The High-level Qualifier Listing for the Usage Monitor utility creates a list of high-level qualifiers for the products that are to be identified.

Following are some examples that exclude all usage, but include some usage for the specified high-level qualifiers:

```
XDS(*)
IDS(DB2.*)
IDS(IMS.*)
IDS(CICS.*)
IDS(SYS1.*)
```

The high-level qualifier listing process is automated in the Inquisitor Import job. The high-level qualifier listing is written to a data set, and this data set is concatenated to the `HSIZIN` control file for the Usage Monitor program.

### Running the High-level Qualifier Listing for the Usage Monitor utility

To run the High Level Qualifier for the Usage Monitor utility, use the job HSISLLST in the JCLLIB. This job is generated from the HSISCUST post-installation customization job.

### TPARAM parameters

**DSN=** DB2 location. Value assigned, as defined in HSISCUST.

**REPSCHEMA=**
Repository qualifier. Name of qualifier is *&REPZSCHM*.

## Updating the TPARAM table

The TPARAM table in the repository can be set to an inconsistent state due to failures in jobs that update the repository tables. You can reset a parameter in the TPARAM table to rectify this inconsistent state.

To run the TPARAM table update, use the job HSISTPRM, in the JCLLIB. This job is generated from the HSISCUST post-installation customization job.

### SYSIN parameter

```
UPDATE &REPZSCHM.TPARAM SET FVALUE = '0' WHERE FKEY = 'PROCRUN';
```

## Tagging unidentified products with the Product Tagging utility

The Product Tagging utility helps you to identify software that has not been predefined in the Global Knowledge Base (GKB). You can also use the Product Tagging utility to supersede GKB entries without changing the contents of the GKB.

### Product tagging process

Product tagging is a manual process where you must provide the product name, the vendor, and the location of the programs. The Product Tagging utility uses the same method as the Inquisitor program to scan the programs and records the results in dedicated program members.

The SYSIN file contains the control statements that describe which licensed programs are to be tagged. This file contains the program name, vendor name, product identifier, and product version. The program library which contains the software to be tagged is allocated to the SYSLIB file.

You can have only one set of identifying attributes for each program name. If conflicting attributes are found for one or more program names, the Product Tagging utility issues a message and stops.

Information about all discovered programs relating to the nominated product is compiled into a single object module. This module is written to the scanned library allocated to SYSLIB file or to the program library allocated to the optional HSIREDIR file. Using the HSIREDIR file, you can nominate to keep all tag data separate from licensed program software. The HSIREDIR file data sets must be included in the standard Inquisitor scan processing, even if these data sets contain no other program.

The tag data members created by the Product Tagging utility are recognized by the Inquisitor (by their SSI value) during normal program library scanning. The Inquisitor program extracts the tag data from the member contents and writes it to an output file. The Inquisitor import process uses these program tags to maintain entries for the programs in the local knowledge base. The match engine can then accurately identify the tagged product level, regardless of which library the product is deployed to and which system the data is collected from.

Each time you run the Product Tagging utility, it scans a single library and tags a single software product, or optional feature of a product. For products with multiple program libraries, each library is processed in a separate job or step. To ensure effective software identification by the match engine as it processes each

library, use the OPTION statement to differentiate the identification entities between the different libraries of a product. Do not tag distribution libraries.

You can override the default output member name of @HSIPTAG by specifying a TAGMEM statement. All output members from the Product Tagging utility are flagged with an SSI value of X'D7E3C1C7', which is 'PTAG' in EBCDIC.

If there is no preexisting member of the same name, the Product Tagging utility creates a new program member to contain the tag data. If a member exists, the new tag data is added to the existing data that relates to other products or optional features. Any data relating to the same software identified by {VENDOR + PRODUCT + OPTION + VERSION} is replaced. The data relating to each software piece resides in its own control section. Tag data members contain no executable code, and are bound with the only loadable attribute. These data members are bound as reentrant, with a residence mode of ANY, to minimize the impact of being placed in a library which is loaded into the Link Pack Area.

To erase the effects of processing with the Product Tagging utility, delete the tag data members which are identified by their SSI value. If you are using ISPF, employ the **SORT SSI** member list command.

The software processed when you run the Product Tagging utility has a key of {VENDOR + PRODUCT + OPTION + VERSION}. If non-key data items, such as the values specified in the PPNUM or LICENSED statements are incorrect, you can correct them by fixing the input statement values and rerunning the utility. This action replaces all non-key tag data. However, if a key data item is incorrect, it will not be erased by running the Product Tagging utility with the correct data.

If you are processing libraries that are not dedicated to a single licensed program, use member name masking to prevent tagging programs not related to that product. Some installations place multiple software products in a combined common library. If the products are tagged before they are combined, you must use different tag data member names.

## Product tagging job and control statements

You use the HSISPTAG job in the JCLLIB to run the Product Tagging utility. This job is generated from the HSISCUST post-installation customization job. You input control statements using the `SYSIN` file.

General syntax rules are:

- Fixed length, variable length, and undefined record formats are processed.
- Short records are extended to 72 bytes of data, with blanks if necessary.
- Only the first 72 bytes of data for each record are processed by the Tagger.
- Records beginning with an asterisk are treated as comments and do not alter continuation status.
- The first nonblanks of a statement must identify the statement type.
- One or more blanks must follow the statement type.
- A statement with no value or operand specified is invalid.
- For statement types other than SELECT, the specified value is deemed to start with the first nonblank after the statement type name.
- Statements can be placed in any order. All statements are processed before any tagging activity commences.

- SELECT is the only statement type which can be supplied more than once in an input file.
- SELECT is the only statement type which can be continued over more than one record.

The following table lists all of the statement types that you can use with the Product Tagging utility:

*Table 23. Product Tagging utility statement types*

| Statement Type | Value | Default Value | Required | Maximum length |
|---|---|---|---|---|
| VENDOR | Vendor name | - | Yes | 30 bytes |
| PRODUCT | Product name | - | Yes | 50 bytes |
| PPNUM | Licensed program number | blanks | No | 16 bytes |
| OPTION | Optional feature name | BASE | No | 30 bytes |
| VERSION | Software level | - | Yes | 8 bytes |
| LICENSED | Separately licensed feature? (YES or NO) | NO | No | 3 bytes |
| TAGMEM | Output member name | @HSIPTAG | No | 8 bytes |
| SELECT | Program name filter | PGM(*) | No | 8 bytes per mask |

SELECT is not a value-oriented statement type. It has operands which have values specified in parentheses. The PROGRAM or PGM inclusion operand can be abbreviated to P. The XPROGRAM or XPGM exclusion operand can be abbreviated to XP.

The Tagger stops parsing a SELECT record and the current statement continues on to the next record whenever a continuation character is encountered. Valid continuation characters are plus and hyphen. A continuation cannot occur within an operand name, or a value mask.

### SELECT syntax


***member-mask***

A string up to 8 bytes in length, representing one or more possible member names of a PDS or PDSE. Use a percent sign to indicate that any single character is to be considered a match in the exact location of the compared character string. Use an asterisk to indicate that any zero or more characters are a match.

## Product tagging examples

Three examples are provided to show the ways that you can use the Product Tagging utility to tag unidentified products.

## Example 1

A company called ISV has created a build of several programs (build 97) it is developing under the Swisho4U brand. The data sets created by this build have their own disk volume called BLD097. The tag data is to be redirected to a data set dedicated for this purpose.

```
//STEP1   EXEC PGM=HSITAGP
//SYSPRINT DD  SYSOUT=*
//SYSLIB   DD  DSN=S4U.LOADLIB,DISP=SHR,UNIT=3390,VOL=SER=BLD097
//HSIREDIR DD  DSN=S4U.TAGLIB,DISP=SHR
//SYSIN    DD  *
VENDOR   ISV
PRODUCT  Swisho4U
VERSION  BUILD097
/*
```

## Example 2

The BigBiz Inc. data center is about to deploy the contractor data processing component for Version 4 Release 2 of its internally developed human resources application called HU-MAN. The software is tagged in its own library, but the default tag member name is not used in case it is later loaded into a program library common to several applications. All programs in HU-MAN have names beginning with HU, but the contractor component is the only component which has program names beginning with HUC. The relevant program library can be accessed by using the catalog.

```
//TAGRUN  EXEC PGM=HSITAGP
//SYSPRINT DD  SYSOUT=*
//SYSLIB   DD  DSN=HUMAN.V4R2M0.LOAD,DISP=SHR
//SYSIN    DD  *
VENDOR   BIGBIZ INCORPORATED
PRODUCT  HU-MAN Human Resources Management
OPTION   Contractor Handling
VERSION  04.02.00
TAGMEM   HUMANT@G
SELECT   PGM(HUC*)
/*
```

## Example 3

Version 1.5 of the product MVSBLOAT from MiscWare has been deployed on a system which has a dedicated tag data library called SYS2.TAGLIB. Link list programs for the product have been placed in SYS2.LINKLIB and ISPF application modules have been placed in SYS2.ISPLLIB. The product does not have optional features, but only the base component installed. All the installed programs have names beginning with MVSB. The OPTION statement is used to ensure that the contents of each library can be identified by the Match Engine.

```
//STEP1   EXEC PGM=HSITAGP
//SYSPRINT DD  SYSOUT=*
//SYSLIB   DD  DSN=SYS2.LINKLIB,DISP=SHR
//HSIREDIR DD  DSN=SYS2.TAGLIB,DISP=SHR
//SYSIN    DD  *
VENDOR  MiscWare
PRODUCT MVSBLOAT
OPTION  BASE (Batch)
VERSION 01.05.00
TAGMEM  $$OEMTAG
SELECT  PGM(MVSB*)
/*
//STEP2   EXEC PGM=HSITAGP
//SYSPRINT DD  SYSOUT=*
```

```
//SYSLIB   DD  DSN=SYS2.ISPLLIB,DISP=SHR
//HSIREDIR DD  DSN=SYS2.TAGLIB,DISP=SHR
//SYSIN    DD  *
VENDOR  MiscWare
PRODUCT MVSBLOAT
OPTION  BASE (Dialogs)
VERSION 01.05.00
TAGMEM  $$OEMTAG
SELECT  PGM(MVSB*)
/*
```

# Importing subcapacity reporting data with the SCRT Import utility

The SCRT Import utility reads data created by the IBM Subcapacity Reporting Tool and generates CSV files that you can then import to the repository. You can use the Analyzer to query this data for trending of SCRT data and to compare the data with the corresponding usage data.

## Running the SCRT Import utility

To run the SCRT import utility, use the job HSISSCRT, in the JCLLIB. This job is generated from the HSISCUST post-installation customization job.

## Data input

DDNAME CSVIN contains the CSV output from the IBM SCRT tool which can be from a data set with DSORG of PS or PO. Binary uploaded CSV files are supported. DDNAME SIDMAP maps duplicate SMFIDs to a unique SID. The SCRT Import utility handles data where the same SMFID is used on multiple machines concurrently.

## Data input example

Map SMFID on specific machines to your desired SID. As described in this example, when processing data for CPU serial 11111, SMFID IP01, use SID QIP1, and so on.

```
//SIDMAP   DD *
11111-IP01=QIP1
11111-IP02=QIP2
11111-IP03=QIP3
/*
```

**CPU serial**
: 5 alphanumeric characters

**SMFID**
: 1 to 4 alphanumeric characters

**Unique SID**
: 1 to 4 alphanumeric characters. This must be the same as the SID value being used by the Usage Monitor for that z/OS system.

## Data output

Several DB2 tables are populated from the data contained in CSVIN, including NODE, NODE_CAPACITY, and PRODUCT_NODE_CAPACITY. Ensure that the CSVIN DD points to the .CSV output file created by the SCRT tool. This may be a DSORG=PO or PS data set.

### TPARAM parameters

**SSID=**
DB2 subsystem name. Value assigned, as defined in job HSISCUST

**REPSCHEMA=**
Repository qualifier. Name of qualifier is *&REPZSCHM*.

**GKB=**
Global Knowledge Base qualifier. Name of qualifier is *&GKBZSCHM*_GKB7

## Capturing historical SMF data with the SMF Scanner utility

You can run the SMF Scanner utility to get historical usage information from existing SMF data. This SMF data enables you to view trending results from before Tivoli Asset Discovery for z/OS is installed.

To start the process you need to run two jobs to capture scanned data (Inquisitor) and historical usage data (SMF Scanner). The output from the SMF Scanner (usage data) can then be processed to produce historical trending.

A sample job HSISIBM can take a file from either the Inquisitor or the Usage Monitor and filter out non-IBM programs. You might use this function when sending data to IBM Support for diagnosis.

The output of the SMF Scanner may also be used as input to HSISIBM. The SMF Scanner only tracks usage of the Job Step EXEC PGM modules and does not include modules that have been invoked from within the task.

### Running the SMF Scanner utility

To run the SMF scanner utility, use the job HSISSMF in the JCLLIB. This job is generated from the HSISCUST post-installation customization job.

## Extracting data with the XML Export utility

The XML Export utility extracts information in XML format that you can then import into SmartCloud Control Desk.

The extracted information can be either:

- A catalog of the products that are installed in your system.
- A catalog of the products defined in the Global Knowledge Base (GKB).

### Running the XML Export utility

To run the XML export utility, use the job HSISKBT, in the JCLLIB. This job is generated from the HSISCUST post-installation customization job.

The output XML file generated from this utility needs to be transferred by FTP to a distributed environment and then loaded into SmartCloud Control Desk. The XML file must be translated from EBCDIC to ASCII.

### TPARAM parameters

**SSID=**
DB2 subsystem name. Value assigned, as defined in job HSISCUST.

**SCHEMA=**

Repository qualifier or Global Knowledge Base qualifier.

a) Using the Repository qualifier value means that a catalog of products installed on your site is selected

b) You can also use the Global Knowledge Base qualifier value. This would mean that a catalog of all products defined in the Global Knowledge Base is selected.

## Transferring output XML by FTP

The output XML file that is generated when you run the XML Export utility must be transferred by FTP to a distributed environment before you can load it into SmartCloud Control Desk.

### Procedure

1. To connect to the host system, in a command line, enter the following command:
   `C:\temp ftp host name`.
2. When prompted, enter your user name and password.
3. To set the input to ASCII format, enter the following command:
   `ftp > quote type a`.
4. Optional: To transfer non-ASCII characters, enter an ENCODING command before you enter the GET command:
   `quote site ENCODING=MBCS MBDATACONN=(IBM-939,UTF-8)`. This example specifies encoding for a Japanese codepage.
5. To specify the location of the file to transfer, enter the following command:
   `ftp > get 'hsiinst.SWKBT.XML' C:\XML.FILE`.
6. To complete the FTP transfer, enter the following command:
   `ftp > exit`.

# Compressing and decompressing data sets with the HSIZIP utility

HSIZIP is a utility program that can compress a sequential or partitioned data set into a zip archive, decompress the contents of a zip archive into a sequential or partitioned data set, and report on the contents of a zip archive.

In this context, an archive is a sequential file that contains one or more logical files for the purpose of reducing the space occupied by the data. The archive can serve as a backup and convenient transport format for the data it contains. The Inquisitor and Usage Monitor components usually create zip archives to contain the data that they collect.

The HSIZIP utility has two compress and decompress functions; one for text data and one for binary data.

## Text data processing with the HSIZIP utility

The HSIZIP utility processes text data to be compatible with the zip processing utilities available on other platforms.

When compressing a text record, the HSIZIP utility performs the following tasks:

- Translates extended binary coded decimal interchange code (EBCDIC) line feed (LF) characters (x'25') to periods.

- Translates EBCDIC data to American standard code for information interchange (ASCII) data.
- Appends an ASCII carriage return line feed (CRLF) sequence (x'0D0A') to encode the record extent.
- Compresses the data and writes it to the archive.

Each compressed member is marked as an ASCII text file and the internal attribute value of the central file header is set to 1.

When decompressing text data, the HSIZIP utility performs the following tasks:

- Accumulates data until an ASCII LF (x'0A') is encountered.
- Truncates the trailing ASCII carriage return (CR) (x'0D') if present in accumulated data.
- Translates the ASCII data to EBCDIC and writes the data as a single record.

During compression, records read from data sets with fixed-length records have their trailing blanks truncated before being compressed. After being decompressed, short records to be written to data sets with fixed-length records are extended with blanks to the required length.

The translation tables used for conversion between EBCDIC and ASCII that are originally sourced from the EZAESENU member in the SEZATCPX library are reciprocal so that applying one translate table and then the other yields the original data. Consequently, all EBCDIC single byte character set (SBCS) and double byte character set (DBCS) text can undergo a ZIP and UNZIP cycle without corruption.

## Binary data processing with the HSIZIP utility

The HSIZIP utility processes binary data in order to preserve record boundaries, while other platforms typically consider binary data to be a byte stream without structure.

When compressing a record of binary data, the HSIZIP utility performs the following tasks:

- Prefixes the data with a multiple virtual storage (MVS) type of record descriptor word (RDW), where the first two bytes contain the length of the record including the RDW, and the third and fourth bytes contain zeros.
- Compresses the data and writes it to the archive.

Each compressed member is marked as a binary file and the internal attribute value of the central file header is set to 0.

When decompressing binary data, the HSIZIP utility performs the following tasks:

- Decompresses 4 bytes from the archive and extracts the record length.
- Decompresses the RDW-indicated length minus 4 bytes and writes it as a record.

During decompression of binary data, the embedded RDWs are checked for validity. If an RDW does not indicate a positive length greater than 4 or does not end with two bytes of zeros, the HSIZIP utility switches to byte stream mode. In byte stream mode, the utility considers data as a stream of bytes without an inherent record structure. If the RDW that fails the validity test is the first four bytes of the file, the resultant decompression is broadly compatible with the decompression that most other platforms perform and the utility issues an informational message. If the RDW that fails the validity test is not at the start of

the file, the utility issues a warning message, sets the final condition code to be greater than zero, but continues processing so that the output data is available for any necessary data recovery activity.

## HSIZIP program parameters

The HSIZIP utility can accept up to two program parameters. The first parameter specifies the function the program is to perform and the second parameter can provide a data definition override list for programs that dynamically invoke the utility.

When you invoke the HSIZIP utility as a stand-alone batch program, the PARM value on the EXEC statement specifies the functional request. DD statements define the details of the following files:

- The SYSPRINT report file
- The SYSUT1 input file
- The SYSUT2 output file

You can specify program parameters in the function request in mixed case. The following information describes valid program parameters.

**LIST** If you specify this parameter, the utility produces a list of the contents of the input archive.

**TEST** This function is similar to the LIST function except the report contains data from both the local file headers and the central file directory, including file offsets, so that the integrity of the archive can be verified.

**ZIP or ZIP=filename.ext**
Use this parameter to compress a partitioned data set into an archive where each member is loaded as a separate zipped file within the archive. A sequential input file is processed as a single member stored in the archive under the name specified in the parameter. If no name is specified in the parameter, the name **seq.txt** is used. The data is treated as text.

**ADD or ADD=filename.ext**
This parameter performs the same function as ZIP except that the output file must be an existing zip archive. The utility writes the compressed data as additional member(s) and prints a report of the original contents of the output archive before it starts to process any new data. The data is treated as text. There is no dependency on the text or binary nature of the existing zipped files in the archive.

**UNZIP or UNZIP=filenamemask**
Use this parameter to decompress an archive into a partitioned data set and load each zipped file into a separate member. The parameter restores data sets from archives made by the HSIZIP utility with **PARM=ZIP**. If the output data set is sequential, only the first file in the archive is unzipped. You can use the file name mask specification to filter the files to be unzipped.

**ZIPBIN or ZIPBIN=filename.ext**
Use this parameter to compress a partitioned data set into an archive and load each member as a separate zipped file within the archive. A sequential input file is processed as a single member stored in the archive under the name specified in the parameter. If no name is specified in the parameter then the name **seq.bin** is used. The data is treated as binary data and no translation is performed.

**ADDBIN or ADDBIN=filename.ext**

This parameter performs the same function as the **ZIPBIN** parameter except that the output file must be an existing zip archive. The utility writes the compressed data as additional member(s) and prints a report of the original contents of the output archive before it starts to process any new data. The data is treated as binary and no translation is performed. There is no dependency on the text or binary nature of the existing zipped files in the archive.

**UNZIPBIN or UNZIPBIN=filenamemask**

Use this parameter to decompress an archive into a partitioned data set and load each zipped file into a separate member. The parameter restores data sets from archives made by the HSIZIP utility with **PARM=ZIPBIN**. If the output data set is sequential, only the first file in the archive is unzipped. Use the file name mask specification to filter the files to be unzipped.

The filenames and filename masks that you specify in program parameters must not exceed 128 bytes in length. File name mask matching is case insensitive. The following characters are generic masking characters for filename masks:

- ? (question mark) matches any single character.
- * (asterisk) matches any zero or more contiguous characters.

If the function request is absent or invalid, the utility writes usage notes to the report file. If the request is absent, the utility attempts to run the LIST function.

## HSIZIP files

The HSIZIP issues report files, input files and output files.

The HSIZIP utility uses the following files:

- SYSPRINT is a report file. RECFM=VBA and LRECL=137 are used in the DCB.
- SYSUT1 is an input file that describes the data set that contains data to be zipped or the zip archive that contains data to be listed or unzipped.
- SYSUT2 is an output file that contains the results of a compression or a decompression operation. This file is not required by the LIST and TEST functions.

The HSIZIP utility does not support spanned records for any file. The main compression and decompression input and output to archive files uses the queued sequential access method (QSAM) locate mode. Apart from the lack of support for spanned records, an input archive allocated to SYSUT1 can have any valid record format and reside on any device that can be read by QSAM. An archive allocated to SYSUT2 must have variable-length records and support update-in-place processing. In effect, a SYSUT2 file must be an MVS DASD data set that is not also a compressible extended-format data set.

## Dynamic invocation of the HSIZIP program by other programs

Other programs can call the HSIZIP utility to perform compression and decompression processing requests. When the HSIZIP utility receives control, it examines the program parameter list and proceeds accordingly.

The first program parameter must begin with a halfword counter indicating the length of the function request text that immediately follows. The format is the same format as the system uses to pass the parameter specified in the PARM operand of the EXEC statement in JCL.

A second program parameter can be specified to override the default filenames used by the HSIZIP utility. If the value of the halfword length indicator at the start of the parameter is not a multiple of 8 or is not less than 256, the HSIZIP utility ignores it. A series of 8-byte file name entries immediately follow the length indicator and each can specify the DD name to use instead of the default name. Set a slot to 8 bytes of zeros to avoid overriding that particular default file name. SYSPRINT, SYSUT1 and SYSUT2 correspond to the sixth, eighth and ninth file name slots respectively.

## HSIZIP data set support

The data control block (DCB) attributes of the original data set that the HSIZIP utility compresses are not encoded into the archive. The success of a compress and decompress cycle requires the user to supply suitable DCB attributes for the ultimate destination of the data.

The following points are provided to help you to assess whether the HSIZIP utility can successfully process a data set:

- There is no internal limit on the size of the uncompressed data. The file headers within the archive contain 32-bit counters for the uncompressed and compressed data byte counts. It is important that the low-order 32 bits of the uncompressed file size is recorded accurately in these headers. It is not important if the counter wraps around one or more times other than being able to correctly report the uncompressed files size from the header data.
- The compressed byte count of an archive member cannot exceed the number that can be stored in an unsigned 32-bit binary integer. This limit is a fundamental HSIZIP limit.
- When processing a whole partitioned data set, the file name specified after ZIP= or ADD= is ignored because the member names are used to label the archived files.
- When ZIP processing detects that a PDS member is a zip archive, the member is stored as a byte stream as is without attempting further compression or record boundary preservation.
- ZIPBIN processing of PDS members containing zip archives usually causes the compressed size to be larger than the uncompressed size, due to the inability to further compact the data and the insertion of RDWs to preserve record boundaries. So, if the only non-text data in a PDS is in members which are themselves zip archives, specify ZIP rather than ZIPBIN to minimize the resultant file size.
- When using ADD or ADDBIN, avoid duplicate file names in the resultant archive.
- You can use the ADD and ADDBIN parameters to create an archive with a mixture of text and binary file members.
- The binary or text nature of an unzip process is set by the program parameter and not from the attribute values in the file header.
- When the HSIZIP utility creates a zip archive, the data set name of the input file is stored as the zip archive comment.
- PDS member user data such as system status information (SSI), ISPF statistics, and load module attributes are stored in the comment field of the central file header of the archive member and can be restored during unzip operations.
- Alias members are stored as files with zero bytes. The alias member data is preserved only if the real member associated with the alias member is also processed.

- Use ZIPBIN and UNZIPBIN when processing load module libraries.
- Segment overlay programs are not restored properly, unless the TTRs happen to match, because the TTRs in the segment tables are not updated by the HSIZIP utility.
- The HSIZIP utility cannot restore program PDSE data sets because only the program binder can write to program PDSEs. There is no restriction on data PDSEs.

## HSIZIP return codes

When you run the HSIZIP utility, several codes are returned that indicate whether the program ran successfully.

*Table 24. HSIZIP utility return codes*

| Return code | Description |
|---|---|
| 0 | Request processed successfully. |
| 2 | Warning message issued. The warning is for a condition that does not affect the operation of the current request, but will probably impact on the intended use of the file created by the request. |
| 4 | No data was found to process, or an I/O error was encountered. |
| 8 | An error occurred. Look at SYSPRINT for more details. |
| Other | As set by another routine. Look at SYSPRINT for more details. |

# Verifying database changes since the product was released

This utility verifies database changes that were introduced after the product was released.

To run the verification, use job HSISIVPD in the JCLLIB. This job is generated from the HSISCUST post-installation job.

# Chapter 9. Configuring language support

Tivoli Asset Discovery for z/OS includes Japanese language support for MVS™ Message Service (MMS) message information. You can also configure the Analyzer utility to create and view reports in Japanese.

## Configuring Japanese messages

To configure messages in the Japanese language, there are two areas to customize. You must compile the Japanese language MMS messages and then you must enable messages in the Japanese language in the language environment.

### About this task

Japanese language MMS message information is contained in the hsi.SHSIMJPN program directory. You must compile the message file and then install the most recent system runtime message file.The HSISMCMP job in the JCLLIB library compiles MMS messages. This job is generated by the HSISCUST post-installation customization job.

### Procedure

1. Run the HSISCMP job to compile the MMS files into a system runtime message file, for example the `his.MMSJPN99` file. The HSISCMP job is generated by the HSISCUST post-intallation customization task.
2. Create an entry named **`MMSLSTJ9`** in the z/OS PARMLIB library, with the following values:
   ```
   DEFAULTS LANGCODE(JPN)
   LANGUAGE LANGCODE(JPN) DSN(SYS2.MMSJPN99) CONFIG(CNLJPN00)
   ```
3. Enter the following MVS system command to install the system runtime message file:
   ```
   SET MMS=J9
   ```
4. Optional: Enable Japanese messages for the Inquisitor Import and the Usage Import, Summary, and Deletion components. To configure the language environment, refer to the following documents:
   - For information about the Language Environment `MSGFILE` and `NATLANG` options, refer to the *Language Environment Programming Reference (SA227562).*
   - For information about specifying Language Environment runtime options, refer to the *Language Environment Programming Guide (SA227561).*
   - For information about setting `NATLANG(JPN)` as an installation default, refer to the *Language Environment Customization (SA22-7564).*

## Enabling the Analyzer utility for Japanese

You can configure the Analyzer utility so that you can view and create reports in Japanese.

### Procedure

1. In the Analyzer HSIJANLO started task, update the DD statements with the following commands:
   ```
   //HSICUST hsi.SHSIPARM(HSISANCJ)
   //HSINLS hsi.SHSIANL1(HSINLSJP)
   ```

2. Optional: Perform the following tasks to customize your Japanese Analyzer reports:
   a. Copy the **hsi.SHSIPARM(HSISANCJ)** parameter to the **hsiinst.PARMLIB(HSISANCJ)** library.
   b. Modify the **hsiinst.PARMLIB(HSISANCJ)** library to customize your reports.
   c. Enter the following command to update the Analyzer HSIJANLO started task:
      `//HSICUST hsiinst.PARMLIB(HSISANCJ)`

## Configuring the Japanese DB2 subsystem for use with Tivoli Asset Discovery for z/OS

### About this task

Tivoli Asset Discovery for z/OS is implemented with a Japanese DB2 subsystem that is configured with the MCCSID=939 code page (Japanese extended English).

If your DB2 for z/OS system is configured with MCCSID=930, MCCSID=1390, or MCCSID=5026 code page (Japanese extended Katakana), additional customization is required in setting up Tivoli Asset Discovery for z/OS. Customization steps are found in INFO APAR II14738

# Chapter 10. Reference information for Tivoli Asset Discovery for z/OS

Reference information includes messages, repository table layouts, and performance and tuning.

## Repository table layouts

This topic describes the tables in the Repository including column names, types, and length.

*Table 25. NODE*

| Column name | Column type | Column length | Description |
|---|---|---|---|
| NODE_KEY | Char | 32 | Global Unique ID (GUID) for this entry |
| NODE_TYPE | Char | 4 | Entry Type: HW or LPAR |
| HW_TYPE | Char | 4 | System z® Hardware Type, for example 2096 |
| HW_MODEL | Char | 3 | System z Hardware Model, for example P03 |
| HW_PLANT | Char | 2 | System z Hardware Plant, for example 02 |
| HW_SERIAL | Char | 12 | System z Hardware Serial, for example. 000000013EED |
| HW_NAME | Char | 10 | Configured Hardware Name |
| HW_VENDOR | Char | 10 | System z Hardware Vendor, for example IBM |
| LPAR_NUMBER | Integer | | Logical Partition Number, for example 1 |
| LPAR_NAME | Char | 10 | Logical Partition Name, for example LPARSYS1 |
| VMGUEST_NAME | Char | 10 | z/VM® Guest Name (if z/OS is running under z/VM) |
| HW_NODE_KEY | Char | 32 | NODE_KEY for related hardware parent |
| LAST_UPDATE_TIME | Timestamp | | Time stamp entry was last updated |

*Table 26. NODE_CAPACITY*

| Column name | Column type | Column length | Description |
|---|---|---|---|
| NODE_KEY | Char | 32 | NODE GUID |
| PERIOD | Date | | Month for this entry |
| START_TIME | Timestamp | | First date that this entry is applicable for this Month |
| END_TIME | Timestamp | | Last date that this entry is applicable for this Month |
| METRIC_TYPE | Char | 10 | Metric Type: MSU, SUBCAPMSTY |
| LAST_UPDATE_TIME | Timestamp | | Time stamp entry was last updated |
| QUANTITY | Integer | | Metric Value |
| MODEL_CAPACITY | Char | 4 | CPU model for capacity planning purpose |

*Table 27. PRODUCT*

| Column name | Column type | Column length | Description |
|---|---|---|---|
| SW_KEY | Char | 32 | Global Unique ID (GUID) for this entry. For SW_TYPE=VERSION this will be the same value as VERSION_GUID For SW_TYPE=FEATURE this will be the same value as FEATURE_GUID |
| SW_TYPE | Char | 8 | Entry type - VERSION or FEATURE |
| VENDOR_NAME | Char | 50 | Vendor name |
| PRODUCT_NAME | Char | 50 | Product name, which is a normalized form of Version Name in order to group different versions of products under the same product name |
| VERSION | Integer | | Version |
| VERSION_NAME | Char | 50 | Product Version Title |
| FEATURE_NAME | Char | 50 | Product Feature Title |
| PID | Char | 16 | Product Identifier |
| EID | Char | 8 | Entitlement Identifier for the Product Feature |
| SSPID | Char | 8 | Subscription & Support Product Identifier |
| SSEID | Char | 8 | Subscription & Support Entitlement Identifier for the Product Feature |
| PRICETYPE | Char | 10 | Price Type (not used in 7.2) |
| SUBCAPACITY | Char | 20 | IPLA Subcapacity type: Execution-based, Reference-based, z/OS-based, Not eligible, NULL |
| ICA | Char | 1 | Y or N: IBM Company Agreement license |
| IPLA | Char | 1 | Y or N: International Program License Agreement |
| VUE | Char | 8 | IPLA Value Unit Exhibit |
| VENDOR_GUID | Char | 32 | Globally Unique ID for VENDOR_NAME |
| PRODUCT_GUID | Char | 32 | Globally Unique ID for VENDOR_NAME + PRODUCT_NAME |
| VERSION_GUID | Char | 32 | Globally Unique ID for VENDOR_NAME + VERSION_NAME + VERSION |
| FEATURE_GUID | Char | 32 | Globally Unique ID for VENDOR_NAME + VERSION_NAME + VERSION + FEATURE_NAME |
| LAST_UPDATE_TIME | Timestamp | | Time stamp entry was last updated |
| ALT_PRODUCT_NAME | Char | 50 | Alternate product name |
| FOBSERVEDELETED | Timestamp | | Date and time that the library was deleted from Inquisitor data. |
| BUNDLE_NAME | Char | 50 | Product suite |

*Table 28. PRODUCT_INSTALL*

| Column name | Column type | Column length | Description |
|---|---|---|---|
| SW_KEY | Char | 32 | Product GUID |
| SYSTEM_KEY | Char | 32 | System GUID |

*Table 28. PRODUCT_INSTALL (continued)*

| Column name | Column type | Column length | Description |
|---|---|---|---|
| INSTALL_DATE | Date | | Date the product was first observed to be installed on this System |
| UNINSTALL_DATE | Date | | Date the product was first observed to be missing from this System |
| LAST_USED_DATE | Date | | Date the product was last used on this System |
| LAST_UPDATE_TIME | Timestamp | | Time stamp entry was last updated |

*Table 29. PRODUCT_NODE_CAPACITY*

| Column name | Column type | Column length | Description |
|---|---|---|---|
| SW_KEY | Char | 32 | Product GUID |
| NODE_KEY | Date | | Node GUID |
| PERIOD | Timestamp | | Month for this entry |
| START_TIME | Timestamp | | First date that this entry is applicable for this Month |
| END_TIME | Timestamp | | Last date that this entry is applicable for this Month |
| METRIC_TYPE | Integer | | Metric Type: INSTALLED, JOBNAMES, MODULES, USERS, SUBCAPMSU |
| LAST_UPDATE_TIME | Timestamp | | Time stamp entry was last updated |
| QUANTITY | Float | | Metric Value |

*Table 30. PRODUCT_USE*

| Column name | Column type | Column length | Description |
|---|---|---|---|
| PERIOD | Date | | Month for this entry |
| SYSTEM_KEY | Char | 32 | System GUID |
| SW_KEY | Char | 32 | Product GUID |
| FLPARID | Integer | | TLPAR.FLPARID for convenient linking with PRODUCT_USE_DETAIL |
| HW_NODE_KEY | Char | 32 | NODE GUID for Hardware NODE that this System was last running on in this month |
| USER_CNT | Integer | | MAX distinct Userid count |
| JOBNAME_CNT | Integer | | MAX distinct Job Name count |
| ACCOUNT_CNT | Integer | | MAX distinct Account Code count |
| SCRT_MSU | | | Sub-capacity Reporting Tool MSU (millions of service units per hour) |
| EVENT_CNT | Double | | SUM of Module usage |
| START_DATE | Date | | Date within this Period that usage was for first recorded |
| END_DATE | Date | | Date within this Period that usage was for last recorded |
| LAST_UPDATE_TIME | Timestamp | | Time stamp entry was last updated |

*Table 31. PRODUCT_USE_DETAIL*

| Column name | Column type | Column length | Description |
|---|---|---|---|
| PERIOD | Date | | Month for this entry |
| FLPARID | Integer | | System TLPAR.FLPARID for convenient linking with PRODUCT_USE |
| VERSION_GUID | Char | 32 | Product Version GUID |
| FEATURE_GUID | Char | 32 | Product Feature GUID |
| USERNAME | Char | 8 | User ID |
| JOBNAME | Char | 8 | Job Name |
| ACCOUNTCODE | Char | 20 | First 20 chars of the Job Account Code |
| EVENT_CNT | Double | | SUM of Module usage |
| START_DATE | Date | | Date within this Period that usage was for first recorded |
| END_DATE | Date | | Date within this Period that usage was for last recorded |

*Table 32. SYSTEM*

| Column name | Column type | Column length | Description |
|---|---|---|---|
| SYSTEM_KEY | Char | 32 | Global Unique ID (GUID) for this entry |
| LAST_UPDATE_TIME | Timestamp | | Time stamp entry was last updated |
| SID | Char | 4 | Product system ID. By default this is the SMFID. In cases where the same SMFID is used on different systems, the SID must be defined to a unique value for the customer enterprise in the Usage Monitor |
| SMFID | Char | 4 | z/OS SMF ID |
| SYSPLEX | Char | 8 | z/OS Sysplex name |
| IPADDR | Varchar | 45 | Host TCP/IP address |
| HOSTNAME | Varchar | 256 | Host TCP/IP name |

*Table 33. SYSTEM_NODE*

| Column name | Column type | Column length | Description |
|---|---|---|---|
| SYSTEM_KEY | Char | 32 | System GUID |
| NODE_KEY | Char | 32 | Node GUID |
| PERIOD | Date | | Month this entry is for |
| START_TIME | Timestamp | | Time it was first observed that this system is using this Node in this month period |
| END_TIME | Timestamp | | Time it was last observed that this system is using this Node in this month period |
| LAST_UPDATE_TIME | Timestamp | | Time stamp entry was last updated |

*Table 34. TACCOUNT*

| Column name | Column type | Column length | Description |
|---|---|---|---|
| FACCOUNTID | Integer | | Account ID |
| FACCOUNTCODE | Char | 20 | Job Account Code, truncated to 20 characters |

*Table 35. TALTERNATE*

| Column name | Column type | Column length | Description |
|---|---|---|---|
| PRODUCT_NAME | Char | 50 | Product name, which is a normalized form of Version Name in order to group different versions of products under the same product name |
| ALT_PRODUCT_NAME | Char | 50 | Alternate product name |
| BUNDLE_NAME | Char | 50 | Product suite |

*Table 36. TANNOTATE*

| Column name | Column type | Column length | Description |
|---|---|---|---|
| FTYPE | Char | 1 | Annotation type:<br>• A – Asset<br>• D- Discovery<br>• B - Both |
| PRODUCT_GUID | Char | 32 | Globally Unique ID for VENDOR_NAME + PRODUCT_NAME |
| FANNOTATION | Varchar | 255 | Text that is annotation |

*Table 37. TCHANNEL_PATH*

| Column name | Column type | Column length | Description |
|---|---|---|---|
| SID | Char | 4 | Product system ID. By default this is the SMFID |
| CP_CHPID | Char | 4 | Channel path identifier |
| CP_TYPE | Char | 5 | Channel path type acronym |
| PERIOD | Date | | Month for this entry |
| CP_DESC | Char | 32 | Channel path description |

*Table 38. TCONTROL_UNIT*

| Column name | Column type | Column length | Description |
|---|---|---|---|
| SID | Char | 4 | Product system ID. By default this is the SMFID |
| PERIOD | Date | | Month for this entry |
| CU_DEV1 | Char | 4 | First device number |
| CU_DEV2 | Char | 4 | Last device number |
| CU_DEVCT | Integer | | Device count for CU record |
| CU_UNTNM | Char | 8 | Generic device unit name |
| CU_CHN1 | Char | 4 | Channel path 1 |
| CU_CHN2 | Char | 4 | Channel path 2 |
| CU_CHN3 | Char | 4 | Channel path 3 |
| CU_CHN4 | Char | 4 | Channel path 4 |
| CU_CHN5 | Char | 4 | Channel path 5 |
| CU_CHN6 | Char | 4 | Channel path 6 |
| CU_CHN7 | Char | 4 | Channel path 7 |

*Table 38. TCONTROL_UNIT (continued)*

| Column name | Column type | Column length | Description |
|---|---|---|---|
| CU_CHN8 | Char | 4 | Channel path 8 |
| CU_TYPE | Char | 6 | Control unit type |
| CU_MODEL | Char | 3 | Control unit model |
| CU_MFR | Char | 3 | Control unit manufacturer |
| CU_PLANT | Char | 2 | Control unit plant of manufacture |
| CU_SEQNUM | Char | 12 | Control unit sequence number |

*Table 39. TISVEOS*

| Column name | Column type | Column length | Description |
|---|---|---|---|
| FPOVID | Integer | | Product ID |
| FEOSDATE | Timestamp | | End of Service Date |

*Table 40. TJOBDATA*

| Column name | Column type | Column length | Description |
|---|---|---|---|
| FJOBNAME | Char | 8 | Job Name |
| FJOBID | Integer | | Job ID |
| FJOBTYPE | Char | 6 | Job Type |

*Table 41. TLIBRARY*

| Column name | Column type | Column length | Description |
|---|---|---|---|
| FLIBID | Integer | | Library ID |
| FLIBNAME | Char | 128 | Library name |
| FINVID | Integer | | Inventory ID |
| FCREATIONDATE | Timestamp | | Library creation date on Mainframe |
| FLIBDEVNUM | Char | 4 | DASD device number |
| FREFERENCEDATE | Timestamp | | Date library last referenced |
| FLIBVOLSER | Char | 8 | Volser library resides on |
| FTRACKSALLOC | Char | 10 | Number of allocated tracks |
| FTRACKSUSED | Char | 10 | Number of used tracks |
| FORIGIN | Char | 1 | Blank - PDS, E - PDSE, V - VTOC |
| FCATALOG | Char | 1 | S - SMS managed, C - Cataloged, U uncataloged W - cataloged on wrong volume |
| FLINKLIST | Char | 1 | Is this a link listed library? |
| FLINKPACK | Char | 1 | Is this library in the Linkpack |
| FAPFAUTH | Char | 1 | Is this library APF authorized |
| FLASTUSAGE | Date | | 1st month of the most recent usage applied to any module in this library |
| FUSEFLAG | Smallint | | Flag for library usage |
| FMODCNT | Integer | | Number of modules in library |
| FOBSERVEFIRST | Timestamp | | Date and time that library was first observed |
| FOBSERVELAST | Timestamp | | Date and time that library was last observed |

*Table 41. TLIBRARY (continued)*

| Column name | Column type | Column length | Description |
|---|---|---|---|
| FOBSERVEDELETED | Timestamp | | Date and time that library was deleted from Inquisitor data. |
| FCHECKSUM | Char | 40 | A checksum of module names and sizes in a given library used to determine whether a library has changed. |
| FSTORAGEGROUP | Char | 8 | The storage group the library belongs to |

*Table 42. TLIBSYS*

| Column name | Column type | Column length | Description |
|---|---|---|---|
| FLIBID | Integer | | Library ID |
| FLPARID | Integer | | LPAR ID |
| FOBSERVEFIRST | Timestamp | | Date and time that library was first observed |
| FOBSERVELAST | Timestamp | | Date and time that library was last observed |
| FOBSERVEDELETED | Timestamp | | Date and time that library was deleted from Inquisitor data |

*Table 43. TLOGIQ*

| Column name | Column type | Column length | Description |
|---|---|---|---|
| FIQDATE | Timestamp | | Inquisitor date |
| FIMPORTDATE | Timestamp | | Inquisitor Import date |
| FSID | Char | 4 | Product system ID. By default this is the SMFID |
| FSYSPLEXID | Char | 8 | z/OS Sysplex name |
| FFULLREMATCH | Char | 1 | Y or N. Described in job HSISIQIM |
| FPRODUCTONLY | Char | 1 | Y or N. Described in job HSISIQIM |
| FOSTYPE | Char | 4 | z/OS or USS |
| FPLX | Char | 1 | PLX option (Y or N) as defined in Inquisitor scan |
| FVERSIONGKB | Char | 15 | The version of the GKB that the IQ is matched with |
| FSPNUM | Char | 8 | System Product number of the z/OS |
| FFMID | Char | 8 | FMID of the z/OS |
| FOSNUM | Char | 8 | Product ID of the z/OS |
| FOSVERSION | Char | 8 | Version of the z/OS |
| HW_TYPE | Char | 4 | System z hardware type |
| HW_MODEL | Char | 4 | System z hardware model |
| HW_SERIAL | Char | 12 | System z hardware serial |
| FFILTER | Varchar | 256 | List of filters in the Inquisitor file |

*Table 44. TLOGUI*

| Column name | Column type | Column length | Description |
|---|---|---|---|
| FUMONDATE | Timestamp | | Usage Monitor date |

*Table 44. TLOGUI (continued)*

| Column name | Column type | Column length | Description |
|---|---|---|---|
| FIMPORTDATE | Timestamp | | Usage Import date |
| FSID | Char | 4 | Product system ID. By default this is the SMFID |
| FSYSPLEXID | Char | 8 | z/OS Sysplex name |
| HW_TYPE | Char | 4 | System z Hardware Type |
| HW_MODEL | Char | 4 | System z Hardware Model |
| HW_SERIAL | Char | 12 | System z Hardware Serial |
| FFILTER | Varchar | 256 | List of filters in the UMON file |

*Table 45. TLPAR*

| Column name | Column type | Column length | Description |
|---|---|---|---|
| FLPARID | Integer | | LPAR ID |
| FLPARNAME | Char | 20 | Name of the LPAR |
| FUSEFLAG | Smallint | | Indicates if usage has been attributed to this LPAR |
| FEDITFLAG | Smallint | | Has this LPAR record been updated manually |
| FMANF | Char | 10 | Machine manufacturer |
| FMACHINE | Char | 12 | CPU Model |
| FSERIALNO | Char | 12 | CPU Serial number |
| FSYSPLEXID | Char | 8 | Sysplex name if in a Sysplex |
| FMIPS | Integer | | Number of MIPS for LPAR |
| FHW_NAME | Char | 10 | Configured hardware name |
| FIPADDR | Varchar | 45 | Host TCP/IP address |
| FHOSTNAME | Varchar | 256 | Host TCP/IP name |

*Table 46. TMACHINE_RESOURCE*

| Column name | Column type | Column length | Description |
|---|---|---|---|
| HW_NODE_KEY | Char | 32 | NODE GUID for Hardware NODE that this System was last running on in this month |
| LPAR_NAME | Char | 10 | Logical partition name |
| PERIOD | Date | | Month for this entry |
| LPAR_NUMBER | Integer | | Logical partition number |
| LPC_NAME | Char | 10 | LPC name |
| OSTYPE | Char | 8 | OS name |
| ENGINE_CNT1 | Char | 4 | CPU available count |
| ENGINE_CNT2 | Char | 4 | CPU online count |
| ENGINE_CNT3 | Char | 4 | CPU dedicated count |
| ENGINE_CNT4 | Char | 4 | zAAP available count |
| ENGINE_CNT5 | Char | 4 | zAAP online count |
| ENGINE_CNT6 | Char | 4 | zAAP dedicated count |
| ENGINE_CNT7 | Char | 4 | IFL available count |

*Table 46. TMACHINE_RESOURCE (continued)*

| Column name | Column type | Column length | Description |
|---|---|---|---|
| ENGINE_CNT8 | Char | 4 | IFL online count |
| ENGINE_CNT9 | Char | 4 | IFL dedicated count |
| ENGINE_CNT10 | Char | 4 | ICF available count |
| ENGINE_CNT11 | Char | 4 | ICF online count |
| ENGINE_CNT12 | Char | 4 | ICF dedicated count |
| ENGINE_CNT13 | Char | 4 | zIIP available count |
| ENGINE_CNT14 | Char | 4 | zIIP online count |
| ENGINE_CNT15 | Char | 4 | zIIP dedicated count |
| ENGINE_CNT16 | Char | 4 | Other available count |
| ENGINE_CNT17 | Char | 4 | Other online count |
| ENGINE_CNT18 | Char | 4 | Other dedicated count |

*Table 47. TMODULE*

| Column name | Column type | Column length | Description |
|---|---|---|---|
| FMODID | Integer | | Module ID |
| FMODNAME | Char | 40 | Module name |
| FLIBID | Integer | | Library ID |
| FPOVLIBID | Integer | | Product Library ID |
| FMODFLAG | Smallint | | Module indication flag as to which product version it belongs to and whether it has been superseded. |
| FFMID | Char | 12 | FMID |
| FMODSIZE | Char | 8 | Module size |
| FUSEFLAG | Smallint | | Flag for module usage |
| FMODTYPE | Smallint | | Type of module |
| FOBSERVEFIRST | Timestamp | | Date and time that module was first observed |
| FOBSERVELAST | Timestamp | | Date and time that module was last observed |
| FOBSERVEDELETED | Timestamp | | Date and time that module was deleted from Inquisitor data |
| LINKEDITDATE | Date | | Link edit date of module |

*Table 48. TPARAM*

| Column name | Column type | Column length | Description |
|---|---|---|---|
| FKEY | Char | 64 | Parameter Key |
| FVALUE | Char | 254 | Parameter Value |

*Table 49. TPERIODS*

| Column name | Column type | Column length | Description |
|---|---|---|---|
| FPERIOD | Date | | Calendar month for usage |
| FINVID | Integer | | Inventory ID |
| FSUMMARISED | Smallint | | Summary status |

*Table 50. TPOVINV*

| Column name | Column type | Column length | Description |
|---|---|---|---|
| FPOVINVID | Integer | | Unique ID |
| FPOVID | Integer | | Product ID |
| FINVID | Integer | | Inventory ID |
| FPOVGID | Integer | | Global Knowledge Base Version ID |
| FOBSERVEFIRST | Timestamp | | First time observation was made |
| FOBSERVELAST | Timestamp | | Last time observation was made |
| FOBSERVEDELETED | Timestamp | | First time observation was not found in this library |
| FPRODUCTID | Integer | | Product ID |
| FVENDORID | Integer | | Vendor ID |
| FPRODINVID | Integer | | Product and Inventory ID |
| FPATCHLIST | Varchar | 254 | List of current patches applied to z/OS UNIX product. |

*Table 51. TPOVLIB*

| Column name | Column type | Column length | Description |
|---|---|---|---|
| FPOVLIBID | Integer | | Unique ID |
| FPOVINVID | Integer | | Product inventory ID |
| FLIBID | Integer | | Library ID |
| FPOVLIBPID | Integer | | Previous Product Version ID |
| FMATCHCODE | Char | 3 | Matching code |
| FMATCHID | Integer | | Link to Inquisitor Decision table |
| FPRODUCTPCT | Integer | | Product percentage used during match |
| FVERSIONPCT | Integer | | Version percentage used during match |
| FOBSERVEFIRST | Timestamp | | First time observation was made |
| FOBSERVELAST | Timestamp | | Last time observation was made |
| FOBSERVEDELETED | Timestamp | | First time observation was not found in this library |
| FMODCNT | Integer | | Number of load modules in library for product |

*Table 52. TPRODUCT*

| Column name | Column type | Column length | Description |
|---|---|---|---|
| FPRODUCTID | Integer | | Product ID |
| FPRODUCTNAME | Char | 50 | Product Name (could be alias name) |
| FGLOBALNAME | Char | 50 | Product Name (always Global Name if Alias is used) |
| FOPTIONNAME | Char | 30 | Option Name |
| FVENDORID | Integer | | Vendor ID |
| FPRODSTATUS | Smallint | | Billable Status |
| FCATEGORY | Char | 30 | Product Category |
| FDESCRIPTION | Varchar | 254 | Product Description |

*Table 52. TPRODUCT (continued)*

| Column name | Column type | Column length | Description |
|---|---|---|---|
| FOBSERVEDELETED | Timestamp | | Date and time that the library was deleted from Inquisitor data. |

*Table 53. TPRODUCT_REGISTRATION*

| Column name | Column type | Column length | Description |
|---|---|---|---|
| SID | Char | 4 | Product system ID. By default this is the SMFID |
| PID | Char | 16 | Product Identifier |
| PRODUCT_OWNER | Char | 16 | Product owner or vendor |
| PRODUCT_NAME | Char | 50 | Product name |
| PRODUCT_FEATURE | Char | 16 | Product feature or option name |
| PRODUCT_VERSION | Char | 6 | Product version |
| PRODUCT_FLAGS | Char | 8 | Register state entry flags |
| PERIOD | Date | | Month for this entry |

*Table 54. TREGISTERED_PRODUCT_USAGE*

| Column name | Column type | Column length | Description |
|---|---|---|---|
| SID | Char | 4 | Product system ID. By default this is the SMFID |
| PRODUCT_OWNER | Char | 16 | Product owner or vendor |
| PRODUCT_NAME | Char | 50 | Product name |
| PRODUCT_VERSION | Char | 8 | Product version |
| PRODUCT_QUALIFIER | Char | 8 | Product qualifier |
| PERIOD | Date | | Month for this entry |
| PID | Char | 16 | Product Identifier |
| PRODUCT_FLAGS | Char | 8 | Register usage entry flags |
| SMFRECORDS | Integer | | SMF record count |
| TCBTIME | Integer | | Product TCB time |
| SRBTIME | Integer | | Product SRB time |

*Table 55. TUIMPORTCTRL*

| Column name | Column type | Column length | Description |
|---|---|---|---|
| FINVID1 | Integer | | Primary Inventory ID |
| FINVID2 | Integer | | Second ID |
| FINVID3 | Integer | | Third ID |
| FINVID4 | Integer | | Fourth ID |
| FINVID5 | Integer | | Fifth ID |
| FINVID6 | Integer | | Sixth ID |
| FINVID7 | Integer | | Seventh ID |
| FINVID8 | Integer | | Eighth ID |
| FMODVPOV | Char | 1 | If non 0 allows relaxed VPOV assignment |

*Table 55. TUIMPORTCTRL (continued)*

| Column name | Column type | Column length | Description |
|---|---|---|---|
| FLPARNAME | Char | 20 | LPAR name |

*Table 56. TUSELIB*

| Column name | Column type | Column length | Description |
|---|---|---|---|
| FUSELIBID | Integer | | Unique ID |
| FLPARID | Integer | | LPAR ID |
| FLIBID | Integer | | Library ID |

*Table 57. TUSEMTD*

| Column name | Column type | Column length | Description |
|---|---|---|---|
| FMTDID | Float | | Unique ID |
| FLPARID | Integer | | LPAR ID |
| FMODID | Integer | | Module ID |
| FJOBID | Integer | | Job ID |
| FUSERID | Integer | | User ID |
| FPOVLIBID | Integer | | Product Library ID |
| FEVENTCNT | Float | | Total calls to module for this month |
| FPERIOD | Date | | Calendar month that usage occurred |
| FFIRSTDATE | Date | | First day of usage in the month |
| FLASTDATE | Date | | Last day of usage in the month |
| FPROVIDER | Char | 4 | Provider Service |
| FPOVINVID | Integer | | Unique ID |
| FPRODINVID | Integer | | Product and Inventory ID |
| FACCOUNTID | Integer | | Account ID |
| FJESID | Char | 8 | Last JES job ID updated for the month |
| FTCBCNT | Float | | TCB count |
| FTCBTIME | Float | | TCB time |

*Table 58. TUSEPOV*

| Column name | Column type | Column length | Description |
|---|---|---|---|
| FUSEPOVINVID | Float | | Unique ID |
| FLARPID | Integer | | LPAR ID |
| FPOVINVID | Integer | | POVINV ID |
| FJOBCNT | Integer | | Number of distinct Jobs for a product |
| FUSERCNT | Integer | | Number of distinct Users for a product |
| FEVENTCNT1 | Float | | Sum of calls to this product current month |
| FEVENTCNT3 | Float | | Sum of calls to this product previous 3 month |
| FEVENTCNT6 | Float | | Sum of calls to this product previous 4-6 month |
| FEVENTCNT9 | Float | | Sum of calls to this product previous 7-9 month |

*Table 58. TUSEPOV (continued)*

| Column name | Column type | Column length | Description |
|---|---|---|---|
| FEVENTCNT12 | Float | | Sum of calls to this product previous 10-12 month |
| FPERIOD | Date | | Calendar month in which usage occurred |
| FFIRSTUSED | Date | | The earliest usage date in month |
| FLASTUSED | Date | | The most recent usage date in month |
| FPRODINVID | Integer | | Product Inventory ID |
| FSEQNO | Smallint | | Internal use only |
| FACCCNT | Integer | | Number of distinct account codes |
| FTCBCNT | Float | | TCB count |
| FTCBTIME | Float | | TCB time |

*Table 59. TUSEPOVLIB*

| Column name | Column type | Column length | Description |
|---|---|---|---|
| FUSEPOVLIBID | Float | | Unique ID |
| FLARPID | Integer | | LPAR ID |
| FPOVLIBID | Integer | | POVLIB ID |
| FJOBCNT | Integer | | Number of distinct Jobs for a product |
| FUSERCNT | Integer | | Number of distinct Users for a product |
| FEVENTCNT1 | Float | | Sum of calls to this product current month |
| FEVENTCNT3 | Float | | Sum of calls to this product previous 3 month |
| FEVENTCNT6 | Float | | Sum of calls to this product previous 4-6 month |
| FEVENTCNT9 | Float | | Sum of calls to this product previous 7-9 month |
| FEVENTCNT12 | Float | | Sum of calls to this product previous 10-12 month |
| FPERIOD | Date | | Calendar month in which usage occurred |
| FFIRSTUSED | Date | | The earliest usage date in month |
| FLASTUSED | Date | | The most recent usage date in month |
| FSEQNO | Smallint | | Internal use only |
| FACCCNT | Integer | | Number of distinct account codes |
| FTCBCNT | Integer | | TCB count |
| FTCBTIME | Integer | | TCB time |
| FTCBCNT | Float | | TCB count |
| FTCBTIME | Float | | TCB time |

*Table 60. TUSEPRS*

| Column name | Column type | Column length | Description |
|---|---|---|---|
| FUSEPRSID | Integer | | Unique ID for TUSEPRS table |
| FREGVEND | Char | 16 | Product Registration Vendor name |
| FREGPROD | Char | 16 | Product Registration Product name |

*Table 60. TUSEPRS (continued)*

| Column name | Column type | Column length | Description |
|---|---|---|---|
| FREGFEAT | Char | 16 | Product Registration Feature name |
| FREGVRN | Char | 6 | Product Registration Version |
| FREGPID | Char | 8 | Product Registration Product identifier |
| FREGFLAGS | Char | 8 | Product Registration flags |
| FLPARID | Integer | | LPAR ID |
| FPERIOD | Date | | Calendar month when usage occurred |
| FFIRSTDATE | Date | | The earliest usage date in month |
| FLASTDATE | Date | | The most recent usage date in month |

*Table 61. TUSERDATA*

| Column name | Column type | Column length | Description |
|---|---|---|---|
| FUSERID | Integer | | User ID |
| FUSERNAME | Char | 10 | User Name |
| FORGNAME | Char | 8 | Owning Organization |
| FREALNAME | Char | 20 | Real person's name |

*Table 62. TVENDOR*

| Column name | Column type | Column length | Description |
|---|---|---|---|
| FVENDORID | Integer | | Vendor ID |
| FVENDORNAME | Char | 50 | Vendor Name (could be alias name) |
| FGLOBALNAME | Char | 30 | Reserved |
| FVENDORGUID | Char | 32 | Vendor globally unique ID |
| FOBSERVEDELETED | Timestamp | | Date and time that the library was deleted from Inquisitor data. |

*Table 63. TVERSION*

| Column name | Column type | Column length | Description |
|---|---|---|---|
| FPOVID | Integer | | Version ID |
| FVERSIONNAME | Char | 44 | Version name |
| FPPNUMNAME | Char | 16 | PPNUM |
| FPRODUCTID | Integer | | Product Option ID |
| FMINUSAGE | Float | | Minimum usage threshold |
| FVERSIONGUID | Char | 32 | Product version globally unique identifier. PRODUCT.SW_KEY for SW_TYPE = VERSION |
| FFEATUREGUID | Char | 32 | Product feature globally unique identifier. PRODUCT.SW_KEY for SW_TYPE = FEATURE |
| FOBSERVEDELETED | Timestamp | | Date and time that the library was deleted from Inquisitor data. |

# Post-installation jobs

After installation, you can create a custom version of any job in the JCLLIB library or any parameter in the PARMLIB library, by copying and editing the relevant job in the HSISCUST member in the hsi.SHSISAMP data set.

## Jobs generated in JCLLIB for a DB2 environment

The custom JCLLIB members that you create with post-installation customization in a DB2 environment are used to submit jobs to implement the product.

### Post installation jobs

The following table lists the post-installation jobs generated in the JCLLIB library when the DBTYPE is set to DB2.

*Table 64. Post-installation jobs generated for a DB2 database*

| Job | Description |
|---|---|
| HSISDB01 | Job to define DB2 storage groups |
| HSISDB02 | Job to define the GKB database |
| HSISDB03 | Job to define the repository database |
| HSISGKBL | Job to populate the GKB, GKB for z/OS UNIX, and Inquisitor filters. |
| HSISGRNT | Job to grant administrator access to the repository and GKB databases |
| HSISGRTB | Job to grant SELECT access to the repository and GKB tables |
| HSISANS1 | Job to set up RACF security profiles in the Analyzer utility. |
| HSISANS2 | Job to set up new SSL certificates in the Analyzer utility. |
| HSISANS3 | Job to use existing SSL certificates in the Analyzer utility. |

### Operations jobs

The following table lists the operations jobs generated in the JCLLIB library when the DBTYPE is set to DB2.

*Table 65. Operations jobs generated for a DB2 database*

| Job | Description |
|---|---|
| HSISGKBL | Job to populate the GKB, GKB for z/OS UNIX, and Inquisitor filters. To be run when monthly updates are provided. |
| HSISINQZ | Job to run the Inquisitor. |
| HSISINQU | Job to run the Inquisitor for z/OS UNIX. |
| HSISIQIM | Job to run the Inquisitor Import for z/OS and z/OS UNIX. |
| HSIJMON | Started task - Usage Monitor |
| HSISUMON | Job to run the Usage Monitor. |
| HSISUIMP | Job to run the Usage Import. |
| HSIJAUTO | Started task - Automation Server |
| HSIASALC | Job to allocate the Automation Server control file. |
| HSIASSCT | Job to run the Automation Server Scout utility. |

## Reporting jobs

The following table lists the reporting jobs generated in the JCLLIB library when the DBTYPE is set to DB2.

*Table 66. Reporting jobs generated for a DB2 database*

| Job | Description |
|---|---|
| HSIJANLO | Started task - Analyzer online mode. |
| HSISANLO | Job to run Analyzer reporting in online mode. |
| HSISANLB | Job to run Analyzer reporting in batch mode. |

## Utility jobs

The following table lists the Utility jobs generated in the JCLLIB library when the DBTYPE is set to DB2.

*Table 67. Utility jobs generated for a DB2 database*

| Job | Description |
|---|---|
| HSISZCAT | Job to concatenate and aggregate Usage Monitor data sets. |
| HSISPTAG | Job to run the Product Tagging utility. |
| HSISUDEL | Job to run the usage deletion |
| HSISUSUM | Job to run the Usage Summary. |
| HSISLDEL | Job to run the system deletion. |
| HSISLLST | Job to create an HLQ listing for the Usage Monitor. |
| HSISSCRT | Job to read SCRT CSV files and populate the repository. |
| HSISKBT | Job to run the XML utility.<br><br>The XML output is for SmartCloud Control Desk. |
| HSISSMF<br><br>HSISIBM | Job to get the historical usage information from existing SMF data.<br><br>Job to filter out non-IBM programs from the Inquisitor utility and usage data. |
| HSISPLTX | Job to create a PLT entry for CICS. |
| HSISENAX | Job to enable CICS GLUE program. |
| HSISUT01 | Sample job to backup the repository database. |
| HSISUT02 | Sample job to restore the repository database. |
| HSISUT03 | Sample job to reorganize the repository database. |
| HSISUT04 | Sample job to run the RUNSTATS utility on the repository database. |
| HSISIVPD | Diagnostic. Job to verify database changes since the product was released. |
| HSISTPRM | Diagnostic. Job to update the repository TPARAM table. |
| HSISCSI | Diagnostic. Job to gather SMP/E diagnostics data |

## Jobs for porting data between repositories

The following table lists the jobs generated in the JCLLIB library for porting data between repositories when the DBTYPE is set to DB2.

*Table 68. Jobs generated for porting data between repositories for a DB2 database*

| Job | Description |
|---|---|
| HSISUN81 | Job to unload the repository database. |
| HSISLO81 | Job to load the repository database. |

### Migration jobs

The following table lists the migration jobs generated in the JCLLIB library when the DBTYPE is set to DB2.

*Table 69. Migration jobs generated for a DB2 database*

| Job | Description |
|---|---|
| HSISMI75 | Job to export V7.2 Usage data to V.81. |
| HSISMI76 | Job to verify that PTFs UA65570 and UA70726 have been applied in the existing version 7.5 repository. |
| HSISMI81 | Job to add new DB2 objects in existing V7.5 repository. |
| HSISMI82 | Job to populate records and also delete obsolete records in existing V7.5 repository |
| HSISMI83 | Job to drop obsolete DB2 objects in existing V7.5 repository. |
| HSISMI84 | Job to verify that the migration tasks for the existing V7.5 repository have been successfully implemented. |

### Globalization jobs

The following table lists the globalization jobs generated in the JCLLIB library when the DBTYPE is set to DB2.

*Table 70. Globalization jobs generated for a DB2 database*

| Job | Description |
|---|---|
| HSISMCMP | Job to compile Japanese messages for MMS. |

## Jobs generated in JCLLIB for a remote environment

The custom JCLLIB members that you create with post-installation customization in a remote environment are used to submit jobs to implement the product.

### Operations jobs

The following table lists the operations jobs generated in the JCLLIB library when the DBTYPE is set to REMOTE.

*Table 71. Operations jobs generated for a remote environment*

| Job | Description |
|---|---|
| HSISINQZ | Job to run the Inquisitor. |
| HSISINQU | Job to run the Inquisitor for z/OS UNIX. |
| HSIJMON | Started task - Usage Monitor |
| HSISUMON | Job to run the Usage Monitor. |
| HSIJAUTO | Started task - Automation Server |
| HSIASALC | Job to allocate the Automation Server control file. |

*Table 71. Operations jobs generated for a remote environment (continued)*

| Job | Description |
|---|---|
| HSIASSCT | Job to run the Automation Server Scout utility. |

### Utility jobs

The following table lists the Utility jobs generated in the JCLLIB library when the DBTYPE is set to REMOTE.

*Table 72. Utility jobs generated for a remote environment*

| Job | Description |
|---|---|
| HSISZCAT | Job to concatenate and aggregate Usage Monitor data sets. |
| HSISPTAG | Job to run the Product Tagging utility. |
| HSISSMF<br><br>HSISIBM | Optional. Job to get the historical usage information from existing SMF data.<br><br>Optional. Job to filter outnon-IBM programs from the Inquisitor and usage data. |
| HSISPLTX | Optional. Job to create a PLT entry for CICS. |
| HSISENAX | Optional. Job to enable CICS GLUE program. |
| HSISCSI | Diagnostic. Job to gather SMP/E diagnostics data |

### Globalization jobs

The following table lists the globalization jobs generated in the JCLLIB library when the DBTYPE is set to REMOTE.

*Table 73. Globalization jobs generated for a remote environment*

| Job | Description |
|---|---|
| HSISMCMP | Job to compile Japanese messages for MMS. |

## Jobs generated in JCLLIB for a SQLite environment

The custom JCLLIB members that you create with post-installation customization in a SQLite environment are used to submit jobs to implement the product.

### Post installation jobs

The following table lists the post-installation jobs generated in the JCLLIB library when the DBTYPE is set to SQLITE.

*Table 74. Post-installation jobs generated for a SQLite database*

| Job | Description |
|---|---|
| HSISDB01 | Job to create SQLite zFS file system. |
| HSISDB02 | Job to define the GKB database |
| HSISDB03 | Job to define the repository database |
| HSISGKBL | Job to populate the GKB, GKB for z/OS UNIX, and Inquisitor filters. |
| HSISGRNT | Job to grant access to z/OS OMVS groups. |
| HSISANS1 | Job to set up RACF security profiles in Analyzer |

*Table 74. Post-installation jobs generated for a SQLite database (continued)*

| Job | Description |
|---|---|
| HSISANS2 | Job to set up new SSL certificates in Analyzer |
| HSISANS3 | Job to to use existing SSL certificates in Analyzer |

## Operations jobs

The following table lists the operations jobs generated in the JCLLIB library when the DBTYPE is set to SQLITE.

*Table 75. Operations jobs generated for a SQLite database*

| Job | Description |
|---|---|
| HSISGKBL | Job to populate the Global Knowledge Base, Global Knowledge<br><br>Base for z/OS UNIX, and Inquisitor filters. To be run when monthly updates are provided. |
| HSISINQZ | Job to run the Inquisitor. |
| HSISINQU | Job to run the Inquisitor for z/OS UNIX. |
| HSISIQIM | Job to run the Inquisitor Import for z/OS and z/OS UNIX. |
| HSIJMON | Started task - Usage Monitor |
| HSISUMON | Job to run the Usage Monitor. |
| HSISUIMP | Job to run the Usage Import. |
| HSIJAUTO | Started task - Automation Server |
| HSIASALC | Job to allocate the Automation Server control file. |
| HSIASSCT | Job to run the Automation Server Scout utility. |

## Reporting jobs

The following table lists the reporting jobs generated in the JCLLIB library when the DBTYPE is set to SQLITE.

*Table 76. Reporting jobs generated for a SQLite database*

| Job | Description |
|---|---|
| HSIJANLO | Started task - Analyzer online mode. |
| HSISANLO | Job to run Analyzer reporting in online mode. |
| HSISANLB | Job to run Analyzer reporting in batch mode. |

## Utility jobs

The following table lists the Utility jobs generated in the JCLLIB library when the DBTYPE is set to SQLITE.

*Table 77. Utility jobs generated for a SQLite database*

| Job | Description |
|---|---|
| HSISZCAT | Job to concatenate and aggregate Usage Monitor data sets. |
| HSISPTAG | Job to run the Product Tagging utility. |
| HSISUDEL | Job to run the Usage Deletion |

*Table 77. Utility jobs generated for a SQLite database (continued)*

| Job | Description |
|---|---|
| HSISUSUM | Job to run the Usage Summary. |
| HSISLDEL | Job to run the System Deletion. |
| HSISSCRT | Job to read SCRT CSV files and populate the Repository. |
| HSISSMF | Job to get the historical usage information from existing SMF data. |
| HSISIBM | Job to filter out non-IBM programs from the Inquisitor and usage data. |
| HSISPLTX | Job to create a PLT entry for CICS. |
| HSISENAX | Job to enable CICS GLUE program. |
| HSISUT01 | Sample job to backup SQLite zFS file system. |
| HSISUT02 | Sample job to restore SQLite zFS file system. |
| HSISIVPD | Diagnostic. Job to verify database changes since the product was released. |
| HSISTPRM | Diagnostic. Job to update the Repository TPARAM table. |
| HSISSQLT | Diagnostic. Job to display SQLite database files. |
| HSISMNT | Diagnostic. Job to mount SQLite zFS file system and optionally delete the zFS file system. |
| HSISMNTU | Diagnostic. Job to unmount SQLite zFS file system after an IPL. |

## Jobs for porting data between repositories

The following table lists the jobs generated in the JCLLIB library for porting data between repositories when the DBTYPE is set to SQLITE.

*Table 78. Jobs generated for porting data between repositories for a SQLite database*

| Job | Description |
|---|---|
| HSISUN81 | Job to unload the repository database. |
| HSISLO81 | Job to load the repository database. |

## Migration jobs

The following table lists the migration jobs generated in the JCLLIB library when the DBTYPE is set to SQLITE.

*Table 79. Migration jobs generated for a SQLite database*

| Job | Description |
|---|---|
| HSISMI75 | Job to export V7.2 Usage data to V.81. |

## Globalization jobs

The following table lists the globalization jobs generated in the JCLLIB library when the DBTYPE is set to SQLITE.

*Table 80. Globalization jobs generated for a SQLite database*

| Job | Description |
|---|---|
| HSISMCMP | Job to compile Japanese messages for MMS. |

# Database performance and tuning

You can perform various configurations to tune your database to provide the best performance for your Tivoli Asset Discovery for z/OS environment.

## DB2 performance and tuning

Various configuration options are available to assist you in optimizing performance for your environment.

**Initial space allocation**

This section is useful for the database administrator who must determine space requirements for Tivoli Asset Discovery for z/OS. Listed in the table Table 81 are guidelines for the initial spaces allocation based on the number of LPARs. The value for the SIZE parameter is specified in HSISCUST.

*Table 81. Initial space allocation for the product*

| SIZE= | Initial space allocation | Number of LPARs |
|---|---|---|
| 1 | 1600 cylinders | 1-10 |
| 2 | 3750 cylinders | 11-20 |
| 3 | 12600 cylinders | >20 |

*Table 82. Initial space allocations for the 3 largest tables*

| SIZE= | VMODULE (Repository modules) | VUSEMTD (Repository usage records) | VPRODDET (Repository Product Use Detail Records) |
|---|---|---|---|
| 1 | 120,960 KB for 1,000,000 modules | 72,000 KB for 1,000,000 records | 13,680 KB for 100,000 records |
| 2 | 302,400 KB for 2,500,000 modules | 432,000 KB for 6,000,000 records | 68,400 KB for 500,000 records |
| 3 | 846,720 KB for 7,000,000 modules | 1,800,000 KB for 25,000,000 records | 684,000 KB for 5,000,000 records |

For some sites, table space VUSEMTD can be large. For performance and space management requirements, you should consider defining the table space as a partitioned table space.

**Choosing a DB2 subsystem for this product**

The DB2 resources required for this product do not need to be defined in a production DB2 subsystem in order to minimize competition for mainframe resources in the DB2 production environments. To avoid competing for mainframe resources, run the jobs for the Inquisitor Import and Usage Import during off-peak periods. In addition, run the utilities Usage Summary and Usage Deletion during off-peak periods.

**Buffer pools**

By allocating the appropriate buffer pool to the respective table spaces and indexes, as defined in HSISCUST, you can manage your system resources accordingly. For DB2 performance, first investigate the buffer pools. Check with your site specialist on the types and size of buffer pools that are defined for this product.

**Space allocation and utilization**

In terms of space utilization, -1 has been set for all SECQTY to enforce Sliding Secondary Extents. This enables DB2 to manage secondary extents efficiently, and minimizes extension failures. You need to extrapolate the PRIQTY for the table spaces and indexes for the large tables according to your requirements. Definitions for these DB2 objects are listed in the respective jobs in JCLLIB.

Repository tables with the biggest impact on performance due to size are TMODULE, TUSEMTD, and TJOBDATA. Data for the TMODULE table is populated during Inquisitor Import process. TUSEMTD, and TJOBDATA tables are populated during Usage Import. For example, you might have more than 300 million usage records in the TUSEMTD table, and more than 20 million modules identified in the TMODULE table. To minimize space utilization and improve SQL query performance, you must prune your usage records by running the Usage Deletion job HSISUDEL.

**Declared Global Temporary tables**
Declared Global Temporary tables are used during the Asset Aggregator process. The Work file table spaces must be large enough to handle this process. When the Aggregator job step is run, indexes on declared global temporary tables are created. By default, the bufferpool used by the index is dependent on the bufferpool defined for the Work file database. Parameter IXBUFFERPOOL in PARMLIB member HSISAGP1 can be used to substitute the default value.

**Work file database**
When you run some of the SQL queries, they can produce a large amount of output. In order to avoid any excessive output, increase the number and size of the table spaces in the work file database.

**Reorganization and RUNSTATS**
It is important to run reorganization of the Repository table spaces periodically, especially after Inquisitor Imports, Usage Imports, and Usage Deletion After reorganization of the Repository table spaces, it is also a good idea to run RUNSTATS for these table spaces.

# SQLite performance and tuning

## General zFS performance queries

zFS performance is dependent on many factors. To help you to optimize performance, zFS provides performance information to help determine bottlenecks. You can enter the following system commands to get information about the current operation of zFS:

- `F ZFS,QUERY,SETTINGS`
- `F ZFS,QUERY,ALL`

To query and reset performance counters, enter the following z/OS UNIX System Services command:
`zfsadm` command to query and reset performance counters.

## Resource Management Facility (RMF) support for zFS

RMF support is available for zFS. When you are considering zFS performance, investigate the zFS components that are involved in I/O processing to or from a zFS file system. In a shared file system environment it is better for performance if you can mount a file system as read-only rather than as read-write. If a file system is mounted as read-write, but is accessed primarily from a single system, such as SYS1, it can improve performance if that file system is z/OS UNIX owned on system SYS1.

You can also optimize zFS performance by tailoring the size of its caches to reduce I/O rates and path length. It is also important to monitor DASD performance to

ensure that no disc volumes or channels are required to perform beyond the intended capacity.

### Monitoring and tuning cache size to improve zFX performance

You can improve the performance of zFS by controlling the size of the caches that hold file system and log data. Monitor the following caches so that you can control them effectively to reduce I/O rates:

- The user file cache is used for all user files and performs I/O for all user files greater than 7 KB.
- The metadata cache performs I/O for all user files that are smaller than 7 KB.
- The log file cache stores file record transactions that describe changes to the file system.

## Database resources used by Tivoli Asset Discovery for z/OS

Some database resources are affected when you migrate from an earlier version to a more recent version of Tivoli Asset Discovery for z/OS.

The following table lists the &GKBZSCHM_GKB7 database resources that are affected by application data migration. The PARMLIB members provides definitions for these database resources.

*Table 83. &GKBZSCHM_GKB7 database resources affected by migration*

| Version 8.1 qualifier | Table name | Version 7.2 table space | Version 7.5 table space | Version 8.1 table space |
|---|---|---|---|---|
| &GKBZSCHM_GKB7 | PRODUCT | WVDRGKB | WVDRGKB | WVDRGKB |
| &GKBZSCHM_GKB7 | TPARAM | WRULGKB | WRULGKB | WRULGKB |
| &GKBZSCHM_GKB7 | TPRODLINK | WVDRGKB | WVDRGKB | WVDRGKB |
| &GKBZSCHM_GKB7 | TPRODUCT | WPDTGKB | WPDTGKB | WPDTGKB |
| &GKBZSCHM_GKB7 | TPRSMAP | WRULGKB | WRULGKB | WRULGKB |
| &GKBZSCHM_GKB7 | TPTFFMID | WRULGKB | WRULGKB | WRULGKB |
| &GKBZSCHM_GKB7 | TRULES | WRULGKB | WRULGKB | WRULGKB |
| &GKBZSCHM_GKB7 | TSCORPAT | WSCPGKB | WSCPGKB | WSCPGKB |
| &GKBZSCHM_GKB7 | TVENDOR | WVDRGKB | WVDRGKB | WVDRGKB |
| &GKBZSCHM_GKB7 | TVERSION | WVERGKB | WVERGKB | WVERGKB |

The following table lists the &GKBZSCHM_GKU7 database resources that are affected by application data migration. The PARMLIB members provides definitions for these database resources.

*Table 84. &GKBZSCHM_GKU7 database resources affected by migration*

| Version 8.1 qualifier | Table name | Version 7.2 table space | Version 7.5 table space | Version 8.1 table space |
|---|---|---|---|---|
| &GKBZSCHM_GKU7 | TPARAM | WRULGKU | WRULGKU | WRULGKU |
| &GKBZSCHM_GKU7 | TPRODUCT | WPDTGKU | WPDTGKU | WPDTGKU |
| &GKBZSCHM_GKU7 | TPTFFMID | WRULGKU | WRULGKU | WRULGKU |
| &GKBZSCHM_GKU7 | TRULES | WRULGKU | WRULGKU | WRULGKU |
| &GKBZSCHM_GKU7 | TSCORPAT | WSCPGKU | WSCPGKU | WSCPGKU |

*Table 84. &GKBZSCHM_GKU7 database resources affected by migration (continued)*

| Version 8.1 qualifier | Table name | Version 7.2 table space | Version 7.5 table space | Version 8.1 table space |
|---|---|---|---|---|
| &GKBZSCHM_GKU7 | TVENDOR | WVDRGKU | WVDRGKU | WVDRGKU |
| &GKBZSCHM_GKU7 | TVERSION | WVERGKU | WVERGKU | WVERGKU |

The following table lists the &GKBZSCHM_IQF7 database resources that are affected by application data migration. The PARMLIB members provides definitions for these database resources.

*Table 85. &GKBZSCHM_IQF7 database resources affected by migration*

| Version 8.1 qualifier | Table name | Version 7.2 table space | Version 7.5 table space | Version 8.1 table space |
|---|---|---|---|---|
| &GKBZSCHM_IQF7 | TCOMPILERS | WIQFILTR | WIQFILTR | WIQFILTR |
| &GKBZSCHM_IQF7 | TIQFILTERS | WIQFILTR | WIQFILTR | WIQFILTR |
| &GKBZSCHM_IQF7 | TPARAM | WIQFILTR | WIQFILTR | WIQFILTR |
| &GKBZSCHM_IQF7 | TUXFILTERS | WIQFILTR | WIQFILTR | WIQFILTR |
| &GKBZSCHM_IQF7 | TXPCMODULES | WIQFILTR | WIQFILTR | WIQFILTR |
| &GKBZSCHM_IQF7 | TXPCSPEC | WIQFILTR | WIQFILTR | WIQFILTR |
| &GKBZSCHM_IQF7 | TXVENDORS | WIQFILTR | WIQFILTR | WIQFILTR |

The following table lists the &REPZSCHM_LKB7 database resources that are affected by application data migration. The PARMLIB members provides definitions for these database resources.

*Table 86. &REPZSCHM_LKB7 database resources affected by migration*

| Version 8.1 qualifier | Table name | Version 7.2 table space | Version 7.5 table space | Version 8.1 table space |
|---|---|---|---|---|
| &REPZSCHM_LKB7 | TPARAM | WLOCLKB | WLOCLKB | WLOCLKB |
| &REPZSCHM_LKB7 | TPRODUCT | WLOCLKB | WLOCLKB | WLOCLKB |
| &REPZSCHM_LKB7 | TPTFFMID | WLOCLKB | WLOCLKB | WLOCLKB |
| &REPZSCHM_LKB7 | TRULES | WLOCLKB | WLOCLKB | WLOCLKB |
| &REPZSCHM_LKB7 | TSCORPAT | WLOCLKB | WLOCLKB | WLOCLKB |
| &REPZSCHM_LKB7 | TVENDOR | WLOCLKB | WLOCLKB | WLOCLKB |
| &REPZSCHM_LKB7 | TVERSION | WLOCLKB | WLOCLKB | WLOCLKB |

The following table lists the &REPZSCHM_LKU7 database resources that are affected by application data migration. The PARMLIB members provides definitions for these database resources.

*Table 87. &REPZSCHM_LKU7 database resources affected by migration*

| Version 8.1 qualifier | Table name | Version 7.2 table space | Version 7.5 table space | Version 8.1 table space |
|---|---|---|---|---|
| &REPZSCHM_LKU7 | TPARAM | WLOCLKU | WLOCLKU | WLOCLKU |
| &REPZSCHM_LKU7 | TPRODUCT | WLOCLKU | WLOCLKU | WLOCLKU |
| &REPZSCHM_LKU7 | TPTFFMID | WLOCLKU | WLOCLKU | WLOCLKU |

*Table 87. &REPZSCHM_LKU7 database resources affected by migration (continued)*

| Version 8.1 qualifier | Table name | Version 7.2 table space | Version 7.5 table space | Version 8.1 table space |
|---|---|---|---|---|
| &REPZSCHM_LKU7 | TRULES | WLOCLKU | WLOCLKU | WLOCLKU |
| &REPZSCHM_LKU7 | TSCORPAT | WLOCLKU | WLOCLKU | WLOCLKU |
| &REPZSCHM_LKU7 | TVENDOR | WLOCLKU | WLOCLKU | WLOCLKU |
| &REPZSCHM_LKU7 | TVERSION | WLOCLKU | WLOCLKU | WLOCLKU |

The following table lists the &REPZSCHM database resources that are affected by application data migration. The PARMLIB members provides definitions for these database resources.

*Table 88. &REPZSCHM database resources affected by migration*

| Version 8.1 qualifier | Table name | Version 7.2 table space | Version 7.5 table space | Version 8.1 table space |
|---|---|---|---|---|
| &REPZSCHM | NODE | VAGGR | VAGGR | VAGGR |
| &REPZSCHM | NODE_CAPACITY | VAGGR | VAGGR | VAGGR |
| &REPZSCHM | PRODUCT | VAGGR | VPRODUCT | VPRODUCT |
| &REPZSCHM | PRODUCT_INSTALL | VAGGR | VPRODINS | VPRODINS |
| &REPZSCHM | PRODUCT_NODE_ CAPACITY | VAGGR | VAGGR | VAGGR |
| &REPZSCHM | PRODUCT_USE | VAGGR | VPRODUSE | VPRODUSE |
| &REPZSCHM | PRODUCT_USE_ DETAIL | VSHARE | VPRODDET | VPRODDET |
| &REPZSCHM | SYSTEM | VAGGR | VAGGR | VAGGR |
| &REPZSCHM | SYSTEM_NODE | VAGGR | VAGGR | VAGGR |
| &REPZSCHM | TACCOUNT | VSHARE | VSHARE | VSHARE |
| &REPZSCHM | TALTERNATE | N/A | N/A | VADMIN |
| &REPZSCHM | TANNOTATE | N/A | N/A | VADMIN |
| &REPZSCHM | TCHANNEL_PATH | N/A | N/A | VCHANNEL |
| &REPZSCHM | TCONTROL_UNIT | N/A | N/A | VCONTROL |
| &REPZSCHM | TINVCTL | VSHARE | VSHARE | deprecated |
| &REPZSCHM | TINVREG | VSHARE | VSHARE | deprecated |
| &REPZSCHM | TIQHISTORY | VSHARE | VSHARE | deprecated |
| &REPZSCHM | TISVEOS | N/A | N/A | VADMIN |
| &REPZSCHM | TJOBDATA | VJOBDATA | VJOBDATA | VJOBDATA |
| &REPZSCHM | TLIBRARY | VSHARE | VLIBRARY | VLIBRARY |
| &REPZSCHM | TLIBSYS | N/A | VTLIBSYS | VTLIBSYS |
| &REPZSCHM | TLOGIQ | N/A | N/A | VLOGIQ |
| &REPZSCHM | TLOGUI | N/A | N/A | VLOGUI |
| &REPZSCHM | TLPAR | VSHARE | VSHARE | VSHARE |
| &REPZSCHM | TMACHINE_RESOURCE | N/A | N/A | VMACHINE |
| &REPZSCHM | TMODULE | VMODULE | VMODULE | VMODULE |
| &REPZSCHM | TPARAM | VSHARE | VSHARE | VSHARE |

*Table 88. &REPZSCHM database resources affected by migration (continued)*

| Version 8.1 qualifier | Table name | Version 7.2 table space | Version 7.5 table space | Version 8.1 table space |
|---|---|---|---|---|
| &REPZSCHM | TPERIODS | VSHARE | VSHARE | VSHARE |
| &REPZSCHM | TPOVINV | VSHARE | VPOVINV | VPOVINV |
| &REPZSCHM | TPOVLIB | VSHARE | VPOVLIB | VPOVLIB |
| &REPZSCHM | TPRODUCT | VSHARE | VSHARE | VSHARE |
| &REPZSCHM | TPRODUCT_ REGISTRATION | N/A | N/A | VPRODREG |
| &REPZSCHM | TREGCLASS | VSHARE | VSHARE | deprecated |
| &REPZSCHM | TREGION | VSHARE | VSHARE | deprecated |
| &REPZSCHM | TREGISTERED_ PRODUCT_USAGE | N/A | N/A | VREGUSAG |
| &REPZSCHM | TREGLEAF | VSHARE | VSHARE | deprecated |
| &REPZSCHM | TUIMPORTCTRL | VSHARE | VSHARE | VSHARE |
| &REPZSCHM | TUSELIB | VSHARE | VUSELIB | VUSELIB |
| &REPZSCHM | TUSEMTD | VUSEMTD | VUSEMTD | VUSEMTD |
| &REPZSCHM | TUSEPO | VSHARE | VUSEPO | deprecated |
| &REPZSCHM | TUSEPOV | VSHARE | VUSEPOV | VUSEPOV |
| &REPZSCHM | TUSEPOVLIB | VSHARE | VUPOVLIB | VUPOVLIB |
| &REPZSCHM | TUSEPRS | VSHARE | VUSEPRS | VUSEPRS |
| &REPZSCHM | TUSERDATA | VSHARE | VUSRDATA | VUSRDATA |
| &REPZSCHM | TVENDOR | VSHARE | VSHARE | VSHARE |
| &REPZSCHM | TVERSION | VSHARE | VSHARE | VSHARE |

# Chapter 11. Troubleshooting, messages, and support

To isolate and resolve problems with your IBM software, you can use the troubleshooting, messages, and support information. This information contains instructions for using the problem determination resources that are provided with your IBM products.

## Troubleshooting a problem

*Troubleshooting* is a systematic approach to solving a problem. The goal of troubleshooting is to determine why something does not work as expected and how to resolve the problem.

The first step in the troubleshooting process is to describe the problem completely. Problem descriptions help you and the IBM technical-support representative know where to start to find the cause of the problem. This step includes asking yourself basic questions:

- What are the symptoms of the problem?
- Where does the problem occur?
- When does the problem occur?
- Under which conditions does the problem occur?
- Can the problem be reproduced?

### What are the symptoms of the problem?

When starting to describe a problem, the most obvious question is "What is the problem?" This question might seem straightforward; however, you can break it down into several more-focused questions that create a more descriptive picture of the problem. These questions can include:

- Who, or what, is reporting the problem?
- What are the error codes and messages?
- How does the system fail? For example, is it a loop, hang, crash, performance degradation, or incorrect result?

The answers to these questions typically lead to a good description of the problem, which can then lead you a problem resolution.

Two main ways to approach any problem you encounter are understanding messages and using log files.

### Messages

Messages are issued when unexpected events occur. Messages can have any of these severities:

**Informational**
The message confirms an event that was requested or describes another normal occurrence. Informational messages generally do not require any action. The identifier of an informational message ends with the letter I.

**Warning**
The message describes an event that might indicate a problem. Read the

message text and determine whether the event is normal or a problem. The identifier of a warning message ends with the letter W.

**Error** The message describes an event that requires a response. Read the message description and the suggested response. The identifier of an error message ends with the letter E.

**Severe Error**
The message describes an event that requires a response. Read the message description and the suggested response. The identifier of an error message ends with the letter S.

**Unrecoverable Error**
The message indicates that an unrecoverable error was encountered and no requests were processed. Read the message description. The identifier of an error message ends with the letter U.

You can find a message description easily by entering its identifier into the Search box in the information center.

# Problems and solutions

Solution information helps you to understand the causes of an issue with your product and learn what to do to diagnose or resolve the problem.

## Resolving SQLCODE -805 errors

If you receive a SQLCODE -805 error, operational jobs fail.

### Symptoms

Tivoli Asset Discovery for z/OS operational jobs are failing.

### Causes

The DSNACLI plan is not bound with the latest DB2 maintenance package, or the plan references a missing package.

### Diagnosing the problem

Check the log files for error messages in the Tivoli Asset Discovery for z/OS operational jobs. The following log entry provides an example of the failure of the DSNCLIQR package:

```
Native Error Code: -805
{DB2 FOR OS/390}{ODBC DRIVER}{DSN10015}
 DSNT408I SQLCODE = -805,
ERROR:  DBRM OR PACKAGE NAME DBA2..DSNCLIQR.18F920E31-
          3F8F1D9
NOT FOUND IN PLAN DSNACLI. REASON 03
```

### Resolving the problem

**System administrator response:** Rerun DSNTIJCL from hlq.SDSNSAMP for all packages that the DSNACLI plan requires to be bound at the same time. The list of packages is slightly different for each release of DB2 for z/OS.

**System administrator response:** Ensure that the DSNAOCLI package has the following parameters:

```
BIND PACKAGE (DSNAOCLI) MEMBER(DSNCLIMS) -
   CURRENTDATA(YES) ENCODING(EBCDIC)
   SQLERROR(CONTINUE)
```

## Insufficient space in the DB2 work file database

When processing large amounts of data, you can encounter insufficient space in the DB2 work file database.

### Symptoms

Tivoli Asset Discovery for z/OS operational jobs are failing.

### Causes

When you run SQL queries that process large amounts of data, including sorts and joins, there is insufficient space in the work file database that is used for temporary storage.

### Diagnosing the problem

**Database administrator response:** Check for examples of following messages in the DB2 MSTR address space:

```
DSNT501I  -DE91 DSNIXWKF RESOURCE UNAVAILABLE  553
           CORRELATION-ID=XXXXXX
           CONNECTION-ID=DB2CALL
           LUW-ID=AUIBMQXP.OMU1DE81.C4729058740C=0
           REASON 00C900A5
           TYPE 00000230
           NAME 4K
DSNT501I  -DE91 DSNIWKFL RESOURCE UNAVAILABLE  554
           CORRELATION-ID=XXXXXX
           CONNECTION-ID=DB2CALL
           LUW-ID=AUIBMQXP.OMU1DE81.C4729058740C=0
           REASON 00C90084
           TYPE 00000230
           NAME 4K
```

**Database administrator response:** View and check the sizes and extents of the physical data sets allocated to the table spaces in the work file database.

### Resolving the problem

**Database administrator response:** Increase the sizes of the table spaces associated with the work file database. The sample DB2 hlq.SDSNSAMP(DSNTIJTM) job provides examples of how to increase the sizes of the table spaces.

## Preventing timeouts and deadlocks during Inquisitor or Usage imports

To prevent timeouts or deadlocks, batch import jobs from the Inquisitor or the Usage Monitor require exclusive access to tables in the Repository database.

### Symptoms

Tivoli Asset Discovery for z/OS operational jobs for Inquisitor or Usage import are failing.

### Causes

When running Inquisitor import or Usage import batch jobs, multiple users using the Analyzer to access the Repository database can lead to timeouts or deadlocks. These batch jobs require exclusive accesses to the data in the tables. SQLCODE -904 error occurs with reason code 00C90083 or reason code 00C9008E.

### Diagnosing the problem

**Database administrator response:** Check for examples of following messages in the DB2 MSTR address space:

```
DSNT501I  - DSNIDBET RESOURCE UNAVAILABLE  656
                          CORRELATION-ID=AAAAAA
                       CONNECTION-ID=DB2CALL
                         LUW-ID=NETA.GGGGGG.UUUUD99=81188
                      REASON 00C90083
                         TYPE 00000200
                       NAME XXXXXX.YYYYY

  DSNT501I  - DSNIDBET RESOURCE UNAVAILABLE  656
                           CORRELATION-ID=BBBBB
                       CONNECTION-ID=DB2CALL
                         LUW-ID=NETA.GGGGGG.IIIII222=81188
                      REASON 00C9008E
                         TYPE 00000200
                       NAME XXXXXX.YYYYY
```

### Resolving the problem

**Database administrator response:** You can perform one or more of the following changes to resolve the issue:

- Run these jobs during off-peak periods.
- Reduce the number of users that need to use the Analyzer during peak periods.
- Reduce the value of the COMMIT=1000 attribute to COMMIT=500 in PARMLIB members HSISIQP1 and HSISUIP1.
- Define the DSNACLI plan and the dependent packages to use uncommitted reads.
  - Modify all packages and plan in hlq.SDSNSAMP(DSNTIJCL) to have ISOLATION(UR) and then run the job.
  - When you change the settings for DSNACLI, ensure that this plan is not used by other applications or create the plan with a different name.
  - By setting ISOLATION(UR), when you run Analyzer reports, it is likely that the correct or latest information is not be displayed due to concurrency issues in DB2.

## Updating your Global Knowledge Base (GKB) database

Tivoli Asset Discovery for z/OS provides a monthly update to the GKB that you can download from Fix Central. When you run the Inquisitor import, product identification is up-to-date if the latest copy of the GKB database is referenced.

### Symptoms

Products displayed in the Analyzer reports are not correct.

### Causes

The GKB database does not contain the latest updates

### Diagnosing the problem

**Operator response:** Check the Analyzer reports to determine the accuracy of the products displayed. In the Analyzer Administration tab check the IQ Import logs report where the GKB version for each HSISIQIM job run is displayed.

**Database administrator response:** To verify the latest version of the GKB database, check the log in HSISIQIM job for GKB Version = yymmdd. yymmdd represents the version of GKB monthly update that is applied at your site.

### Resolving the problem

**System administrator response:** Download the latest GKB monthly update file from Fix Central and apply it in your environment.

**Applying updates of the GKB database:**

Monthly Global Knowledge Base (GKB) updates are available on Fix Central. After you download an update file, you run the GKB load job to apply the updates to your environment.

**Procedure**

1. Login to Fix Central with a valid IBM user ID and password.
2. Specify the following values:

| Field | Value |
|---|---|
| Product Family | Tivoli |
| Product | IBM Tivoli Asset Discovery for z/OS |
| Installed Version | 8.1.0 |
| Platform | z/OS |

3. Display all fixes. The format of the fix is 8.1.0 Tiv-TADZ-z0S-LV020215. The last six digits signify the fix level and is in YYMMDD format.
4. Select the most recent version of the GKB update file, and download the file as binary. The name of the update file is `TADZ81KB.XMI`.
5. Upload the `TADZ81KB.XMI` file to the mainframe to a preallocated file with the attributes FB 80.
6. After the file is uploaded, receive the file, RECEIVE INDATASET(TADZ81KB.xmi).
7. When prompted for additional information enter, DA(filename).
8. In the GKB load job, HSISGKBL, update the SET INDSN= value with the name of the file that you received, and then submit the job.

## Overcoming space limits for very large DB2 sites

For very large DB2 sites where the space for usage data can exceed 64GB, you can overcome this limit by defining the VUSEMTD table space as a partition-by-growth table space.

### Symptoms

The Tivoli Asset Discovery for z/OS HSISUIMP operational job fails.

### Causes

This issue can occur for the following reasons:

- Retaining too much usage data for long periods and not performing regular housekeeping tasks.
- Importing all usage data into a single repository instead of spreading the usage data across multiple repositories.

### Diagnosing the problem

**Database administrator response:**

- Run the Database Statistics report from the **Administration** tab in the Analyzer online and check the statistics for the VUSEMTD table space and dependent objects.
- From ISPF, enter `&SGBIGCAT.DSNDBD.&DB.VUSEMTD` to list the physical allocation of the VUSEMTD table space.

### Resolving the problem

**Database administrator response:**
You can configure the VUSEMTD table space as a partition-by-growth table space and introduce regular database housekeeping tasks to resolve the problem. Configuring the table space as a partition-by-growth table space has the following prerequisites:

- You are running DB2 for z/OS version 9 or later, with new function mode.
- To specify a value greater than 4G, the following conditions must be true:
  - DB2 is running with DFSMS Version 1 Release 5.
  - The data sets for the table space are associated with a DFSMS data class that has been specified with extended format and extended addressability.

The configuration is a significant task that requires appropriate preparation. Your environment must meet the prerequisites and you need to prepare an implementation plan before you proceed with the following task:

1. Run DB2 Runstats on the VUSEMTD table space.
2. Run the Database Statistics report from the **Administration** tab in the Analyzer online and keep a copy of the results.
3. Run the DB2 Unload utility to unload the TUSEMTD table.
4. Drop the VUSEMTD table space.
5. Create a VUSEMTD table space as a partition-by-growth table space.
6. Change the values for the table space based on site requirements. Refer to the sample definition.
7. Create the TUSEMTD table and dependent indexes. Definitions are available in the HSISSQ18 member in the PARMLIB.
8. Run the DB2 Load utility to load the TUSEMTD table.
9. Run DB2 Runstats on the VUSEMTD table space.
10. Run the Database Statistics report from the **Administration** tab on the Analyzer online.
11. Compare the number of records for the TUSEMTD table and its dependent indexes against the earlier Analyzer report. The results must be identical.

The following code provides a sample definition for partition-by-growth table space:

```
CREATE TABLESPACE VUSEMTD
      IN TADZREP1
      USING STOGROUP SGHSIBIG
       PRIQTY 140000
       SECQTY   -1
       ERASE NO
      DSSIZE 4G
      MAXPARTITIONS 64
      LOCKSIZE PAGE   LOCKMAX SYSTEM
      BUFFERPOOL BP1 SEGSIZE 64
      CLOSE NO
      CCSID EBCDIC;
COMMIT;
```

**Database administrator response:**
Perform regular maintenance, including the following activities:

1. Run the following jobs on a regular basis to delete old usage data, to save space, and to improve processing time:
   a. Run the HSISUDEL job to delete usage data that is older than a specified period.

b. Run the HSISUSUM job to summarize usage data and compress records into monthly periods.
2. Run the HSISLDEL job to delete obsolete discovery and usage data for a specified system (LPAR).

# Tivoli Asset Discovery for z/OS messages

You can identify the type of message by the message prefix.

## HSIA - Automation Server messages

**Return codes**

*Table 89. Return codes and their meaning*

| Return code | Description |
|---|---|
| 0 | No errors encountered. All requests processed successfully. |
| 16 | Unrecoverable error. No requests processed. SYSIN or HSIPZIP or INQSOUT File cannot be used, or unsupported operating system. |

**Message suffix codes**

*Table 90. Message suffix codes and associated condition codes*

| Suffix | Meaning | Raises minimum condition code to: |
|---|---|---|
| I | Information message | 0 |
| W | Warning message | 4 |
| E | Error message | 8 |
| S | Severe error message | 12 |
| U | Unrecoverable error message | 16, 20 |

**Message texts and explanations**

All numeric completion codes of system services reported in these messages are in hexadecimal unless otherwise stated.

**HSIA001E EXPECTED CLOSE PARENTHESIS WAS NOT FOUND IN INPUT RECORD**

**Explanation:** Parsing did not detect the expected close parenthesis.

**System action:** Terminates the processing of the HSIAPARM member contents.

**Operator response:** Correct the HSIAPARM member contents.

**System programmer response:** None.

**Module:** HSIAUTO HSIADSN

**HSIA002E EXPECTED VALUE FOR *parm* NOT FOUND BEFORE THE CLOSE PARENTHESIS**

**Explanation:** No subparameter value was specified within the parentheses.

In the message text:

*parm*
name of parameter being processed.

**System action:** Terminates the processing of the HSIAPARM member contents.

**Operator response:** Correct the HSIAPARM member contents.

**System programmer response:** None.

**Module:** HSIAUTO HSIADSN

**HSIA003E THE *parm* PARAMETER IS NOT RECOGNIZED**

**Explanation:** A parameter was detected which is not valid for the type of statement being processed.

In the message text:

*parm*
name of parameter being processed.

**System action:** Terminates the processing of the HSIAPARM member contents.

**Operator response:** Correct the HSIAPARM member contents.

**System programmer response:** None.

**Module:** HSIAUTO

---

**HSIA004E THE VALUE SPECIFIED FOR *parm* IS INVALID**

**Explanation:** The named parameter had a value which did not conform to syntax requirements.

In the message text:

*parm*
name of parameter being processed.

**System action:** Terminates the processing of the HSIAPARM member contents.

**Operator response:** Correct the HSIAPARM member contents.

**System programmer response:** None.

**Module:** HSIAUTO

---

**HSIA005E NO "FTP" OR "JOB" STATEMENT BEFORE "DSN" STATEMENT**

**Explanation:** There is no preceding action to relate the dsname mask to.

**System action:** Terminates the processing of the HSIAPARM member contents.

**Operator response:** Correct the HSIAPARM member contents.

**System programmer response:** None.

**Module:** HSIAUTO HSIADSN

---

**HSIA006E *parm* IS AN UNRECOGNIZED STATEMENT TYPE**

**Explanation:** A statement type other than FTP, JOB, or DSN was specified.

In the message text:

*parm*
name of parameter.

**System action:** Terminates the processing of the HSIAPARM member contents.

**Operator response:** Correct the HSIAPARM member contents.

**System programmer response:** None.

**Module:** HSIAUTO HSIADSN

---

**HSIA007E TERMINATING - AUTOMATION SERVER IS ALREADY ACTIVE**

**Explanation:** The Automation Server is already running. Only one concurrent copy can run in an operating system image.

**System action:** Program terminates with condition code 16. The established Automation Server continues.

**Operator response:** None.

**System programmer response:** None.

**Module:** HSIAUTO

---

**HSIA008E TERMINATING - COULD NOT INITIALISE WITH BAD PARAMETERS**

**Explanation:** The HSIAPARM contents contained an error so the Automation Server could not initialize.

**System action:** The program terminates with condition code 16.

**Operator response:** Correct the HSIAPARM member contents.

**System programmer response:** None.

**Module:** HSIAUTO HSIADSN

---

**HSIA009E REFRESH IGNORED - COULD NOT PROCESS BAD PARAMETERS**

**Explanation:** The HSIAPARM contents contained an error so the Automation Server could not update its operational parameters.

**System action:** Terminates the processing of HSIAPARM contents.

**Operator response:** Correct the HSIAPARM member contents.

**System programmer response:** None.

**Module:** HSIAUTO HSIADSN

---

**HSIA010E NO FUNCTIONS WERE REQUESTED**

**Explanation:** No actions were specified. The Automation Server has no work to do.

**System action:** Terminates the processing of HSIAPARM contents.

**Operator response:** Correct the HSIAPARM member contents.

**System programmer response:** None.

**Module:** HSIAUTO HSIADSN

---

**HSIA011E NO DATA SET NAME MASKS WERE SPECIFIED**

**Explanation:** No work was requested. The Automation Server has no work to do.

**System action:** Terminates the processing of HSIAPARM contents.

**Operator response:** Correct the HSIAPARM member contents.

**System programmer response:** None.

**Module:** HSIAUTO HSIADSN

---

**HSIA012E NUMBER OF ACTIONS EXCEEDS MAXIMUM OF 1000**

**Explanation:** Too many actions were requested.

**System action:** Terminates the processing of HSIAPARM contents.

**Operator response:** Reduce the number of actions specified.

**System programmer response:** None.

**Module:** HSIAUTO HSIADSN

---

**HSIA013E NUMBER OF DATA SET NAME MASKS EXCEEDS THE MAXIMUM OF 2000**

**Explanation:** Too many dataset name masks were specified.

**System action:** Terminates the processing of HSIAPARM contents.

**Operator response:** Reduce the number of dataset name masks.

**System programmer response:** None.

**Module:** HSIAUTO HSIADSN

---

**HSIA014E MEMBER *mbr* WAS NOT FOUND IN THE HSIACNTL FILE**

**Explanation:** Member HSIAPARM was found to be missing from the HSIACNTL file.

In the message text:

*mbr*
name of missing member.

**System action:** Terminates the processing of the member. If the member is HSIAPARM the Automation Server terminates. For template members the Automation Server continues processing.

**Operator response:** Create the required member in the HSIACNTL library.

**System programmer response:** None.

**Module:** HSIAUTO HSIADSN

---

**HSIA015E INPUT LOGICAL RECORD LENGTH WAS NOT 80**

**Explanation:** A record was read from the HSIACNTL library which was not 80 bytes long.

**System action:** The program terminates and takes no actions.

**Operator response:** Ensure the HSIACNTL library only contains fixed length 80 byte records.

**System programmer response:** None.

**Module:** HSIAUTO HSIADSN

---

**HSIA016E EXPECTED OPEN PARENTHESIS WAS NOT FOUND IN INPUT RECORD**

**Explanation:** Parsing did not detect the expected open parenthesis.

**System action:** Terminates the processing of the HSIAPARM member contents.

**Operator response:** Correct the HSIAPARM member contents.

**System programmer response:** None.

**Module:** HSIAUTO

---

**HSIA017E VALUE SPECIFIED FOR *parm* IS TOO LONG**

**Explanation:** A parameter value was specified which has a length greater than the maximum allowed for the named parameter.

In the message text:

*parm*
name of parameter.

**System action:** Terminates the processing of the HSIAPARM member contents.

**Operator response:** Correct the HSIAPARM member contents.

**System programmer response:** None.

**Module:** HSIAUTO HSIADSN

---

**HSIA018E THE "NOTA" VALUE IS LESS THAN THE "NOTB" VALUE**

**Explanation:** The action can never be performed because all days of the month have been excluded by the combination of the NOTA (not after) and NOTB (not before) specifications.

**System action:** Terminates the processing of the HSIAPARM member contents.

**Operator response:** Correct the HSIAPARM member contents.

**System programmer response:** None.

**Module:** HSIAUTO

---

**HSIA019I AUTOMATION SERVER INITIALIZATION COMPLETE**

**Explanation:** The Automation Server is commencing normal operations.

**System action:** Processing continues.

**Operator response:** None.

**System programmer response:** None.

**Module:** HSIAUTO

---

**HSIA020I AUTOMATION SERVER HAS NOW TERMINATED**

**Explanation:** The Automation Server is ceasing normal operations.

**System action:** Processing concludes.

**Operator response:** None.

**System programmer response:** None.

**Module:** HSIAUTO

---

**HSIA999U HSIMSG/HSIAMSG FAILURE - MSGID=*msgid* RC=*rc* RS=*rs***

**Explanation:** HSIMSG was called to produce a message text, but the call failed.

In the message text:

*msgid*
identifier of the failing message.

*rc* HSIMSG return code.

# HSIF - Conversion messages

### Return codes

*Table 91. Return codes and their meaning*

| Return code | Description |
|---|---|
| 0 | No errors encountered. All requests processed successfully. |
| 4 | Input/Output error in one or more program libraries. |
| 8 | Error - Incomplete data. Processing continues. OPEN or other system service error. |
| 12 | Syntax error. |
| 16 | Unrecoverable error. |
| 20 | Disastrous error. No requests processed. SYSPRINT file cannot be used. |

### Message suffix codes

*Table 92. Message suffix codes and associated condition codes*

| Suffix | Meaning | Raises minimum condition code to: |
|---|---|---|
| I | Information message | 0 |
| W | Warning message | 4 |
| E | Error message | 8 |
| S | Severe error message | 12 |
| U | Unrecoverable error message | 16 |

### Message texts and explanations

All numeric completion codes of system services reported in these messages are in hexadecimal unless otherwise stated.

---

**HSIF000U PROCESSING TERMINATED - NO USABLE SYSPRINT FILE**

**Explanation:** The OPEN of the SYSPRINT file failed.

Note: This message is issued by WTO with ROUTCDE=(2,11).

**System action:** Terminates with a condition code of 20.

**Operator response:** Ensure a usable SYSPRINT file is allocated.

**System programmer response:** None.

**Module:** HSIIM2D HSIIS2D

---

**HSIF001S PROCESSING TERMINATED - STORAGE OBTAIN ENCOUNTERED AN ERROR, RC=*rc***

**Explanation:** Unable to acquire required amount of storage.

In the message text:

*rc* return code from STORAGE macro.

**System action:** Terminates with a condition code of 16.

**Operator response:** None.

**System programmer response:** Try increasing the region size specified in the region parameter on the JOB or EXEC statement in the JCL for the job. For additional information, examine the return code shown in the message and use for problem determination/resolution.

**Module:** HSIIM2D

---

**HSIF002S PROCESSING TERMINATED - DD FOR SYSPRINT MISSING**

**Explanation:** DD statement missing for SYSPRINT. Note: This message is issued by WTO with ROUTCDE=(2,11).

**System action:** Terminates with a condition code of 20.

**Operator response:** Ensure DD statement in the JCL is correct.

**System programmer response:** None.

**Module:** HSIIM2D HSIIS2D

---

**HSIF004S PROCESSING TERMINATED - CANNOT OPEN FILE "*file*"**

**Explanation:** The OPEN of the file failed.

In the message text:

*file*
name of file that failed the OPEN request.

**System action:** Terminates with a condition code of 16.

**Operator response:** Ensure a usable file is allocated.

**System programmer response:** None.

**Module:** HSIIM2D HSIIS2D

---

**HSIF005S PROCESSING TERMINATED - DD FOR FILENAME "*file*" MISSING**

**Explanation:** DD statement missing for the file.

In the message text:

*file*
name of file with missing DD statement.

**System action:** Terminates with a condition code of 16.

**Operator response:** Ensure DD statement in the JCL is correct.

**System programmer response:** None.

**Module:** HSIIM2D HSIIS2D

---

**HSIF007S PROCESSING TERMINATED - INVALID SYSIN PARAMETER**

**Explanation:** Parsing detected an invalid parameter.

**System action:** Terminates with a condition code of 12.

**Operator response:** The final parameter displayed in the SYSPRINT report is in error. Correct the SYSIN file contents and rerun the program.

**System programmer response:** None.

**Module:** HSIIM2D HSIIS2D

---

**HSIF008S PROCESSING TERMINATED - INVALID SYSIN PLX= PARAMETER. MUST BE Y OR N**

**Explanation:** The SYSIN PLX= parameter must be set to Y or N.

**System action:** Terminates with a condition code of 12.

**Operator response:** Correct the SYSIN file contents and rerun the program.

**System programmer response:** None.

**Module:** HSIIM2D HSIIS2D

---

**HSIF009S PROCESSING TERMINATED - INVALID SYSIN SYSPLEX= PARAMETER. THE SYSPLEX NAME MUST BE ENTERED**

**Explanation:** The SYSIN SYSPLEX NAME must be entered.

**System action:** Terminates with a condition code of 12.

**Operator response:** Correct the SYSIN file contents and rerun the program.

**System programmer response:** None.

**Module:** HSIIM2D

---

**HSIF010S PROCESSING TERMINATED - INVALID SYSIN SSID= PARAMETER. THE SMF SYSTEM IDENTIFIER MUST BE ENTERED**

**Explanation:** The SYSIN SSID= parameter must be entered.

**System action:** Terminates with a condition code of 12.

**Operator response:** Correct the SYSIN file contents and rerun the program.

**System programmer response:** None.

**Module:** HSIIM2D HSIIS2D

---

**HSIF011S PROCESSING TERMINATED - INVALID SYSIN SSMF= PARAMETER. THE SMF SYSTEM IDENTIFIER MUST BE ENTERED**

**Explanation:** The SYSIN SSMF= parameter must be entered.

**System action:** Terminates with a condition code of 12.

**Operator response:** Correct the SYSIN file contents and rerun the program.

**System programmer response:** None.

**Module:** HSIIM2D HSIIS2D

---

**HSIF012S PROCESSING TERMINATED - INVALID SYSIN LPARNAME= PARAMETER. THE LPAR NAME MUST BE ENTERED**

**Explanation:** The SYSIN LPAR NAME must be entered.

**System action:** Terminates with a condition code of 12.

**Operator response:** Correct the SYSIN file contents and rerun the program.

**System programmer response:** None.

**Module:** HSIIM2D

---

**HSIF013S PROCESSING TERMINATED - INVALID SYSIN SERIAL= PARAMETER. THE SERIAL NUMBER MUST BE ENTERED**

**Explanation:** The SYSIN SERIAL NUMBER parameter must be entered.

**System action:** Terminates with a condition code of 12.

**Operator response:** Correct the SYSIN file contents and rerun the program.

**System programmer response:** None.

**Module:** HSIIM2D

---

**HSIF014S PROCESSING TERMINATED - INVALID SYSIN HWTYPE= PARAMETER. THE HARDWARE TYPE MUST BE ENTERED**

**Explanation:** The SYSIN HARDWARE TYPE parameter must be entered.

**System action:** Terminates with a condition code of 12.

**Operator response:** Correct the SYSIN file contents and rerun the program.

**System programmer response:** None.

**Module:** HSIIM2D

---

**HSIF015S PROCESSING TERMINATED - INVALID SYSIN HWMODEL= PARAMETER. THE HARDWARE MODEL MUST BE ENTERED**

**Explanation:** The SYSIN HARDWARE MODEL must be entered.

**System action:** Terminates with a condition code of 12.

**Operator response:** Correct the SYSIN file contents and rerun the program.

**System programmer response:** None.

**Module:** HSIIM2D

---

**HSIF016S PROCESSING TERMINATED - INVALID SYSIN UNM= PARAMETER. MUST BE Y OR N**

**Explanation:** The SYSIN UNM= parameter must be set to Y or N.

**System action:** Terminates with a condition code of 12.

**Operator response:** Correct the SYSIN file contents and rerun the program.

**System programmer response:** None.

**Module:** HSIIM2D

---

**HSIF017S PROCESSING TERMINATED - INVALID SYSIN JAC= PARAMETER. MUST BE Y OR N**

**Explanation:** The SYSIN JAC= parameter must be set to Y or N.

**System action:** Terminates with a condition code of 12.

**Operator response:** Correct the SYSIN file contents and rerun the program.

**System programmer response:** None.

**Module:** HSIIM2D

**HSIF018S PROCESSING TERMINATED - INVALID SYSIN PLANT= PARAMETER. THE PLANT MUST BE ENTERED**

**Explanation:** The SYSIN PLANT must be entered.

**System action:** Terminates with a condition code of 12.

**Operator response:** Correct the SYSIN file contents and rerun the program.

**System programmer response:** None.

**Module:** HSIIM2D

**HSIF019S PROCESSING TERMINATED - OUTPUT HEADER NOT CREATED. NO STATISTICS OR CAPACITY RECORD IN INPUT FILE**

**Explanation:** The input file contains no Statistics or Capacity record.

**System action:** Terminates with a condition code of 16.

**Operator response:** Ensure a usable input file exists and rerun the program.

**System programmer response:** None.

**Module:** HSIIM2D

**HSIF020S PROCESSING TERMINATED - HSIMONZP *func* FAILURE, RC=*rc***

**Explanation:** HSIMONZP file processing failure.

In the message text:

*func*
file processing in error.

*rc* return code from HSIMONZP.

**System action:** Terminates with a condition code of 16.

**Operator response:** Ensure the correct DD statements exist for the file and rerun the program. For additional information, examine the return code shown in the message and use for problem determination/resolution.

**System programmer response:** None.

**Module:** HSIIM2D

**HSIF021S PROCESSING TERMINATED - NO DD STATEMENT DETECTED FOR HSIINQOT OR HSIINQZP**

**Explanation:** At least one of the DD statements must be included.

**System action:** Terminates with a condition code of 16.

**Operator response:** Correct the JCL and rerun the program.

**System programmer response:** None.

**Module:** HSIIS2D

**HSIF022S PROCESSING TERMINATED - *func* FAILURE, RC=*rc***

**Explanation:** Processing failure.

In the message text:

*func*
processing in error.

*rc* return code.

**System action:** Terminates with a condition code of 16.

**Operator response:** None.

**System programmer response:** Contact IBM support.

**Module:** HSIIS2D

**HSIF023S PROCESSING TERMINATED - HSIINQZP *func* FAILURE, RC=*rc***

**Explanation:** HSIINQZP file processing failure.

In the message text:

*func*
file processing in error.

*rc* return code.

**System action:** Terminates with a condition code of 16.

**Operator response:** None.

**System programmer response:** Examine the return code shown in the message and use for problem determination/resolution.

**Module:** HSIIS2D

**HSIF024S PROCESSING TERMINATED - HSIINQOT *func* FAILURE, RC=*rc***

**Explanation:** HSIINQOT file processing failure.

In the message text:

*func*
file processing in error.

*rc* return code.

**System action:** Terminates with a condition code of 16.

**Operator response:** None.

**System programmer response:** Examine the return code shown in the message and use for problem determination/resolution.

**Module:** HSIIS2D

**HSIF025S PROCESSING TERMINATED - TABLE #LPRTBL# *func* FAILURE, RC=*rc***

**Explanation:** Table processing failure.

In the message text:

*func*
table operation in error.

*rc* return code.

**System action:** Terminates with a condition code of 16.

**Operator response:** None.

**System programmer response:** Examine the return code shown in the message and use for problem determination/resolution.

**Module:** HSIIS2D

**HSIF999U HSIMSG/HSIFMSG FAILURE - MSGID=*msgid* RC=*rc* RS=*rs***

**Explanation:** HSIMSG was called to produce a message text, but the call failed.

In the message text:

*msgid*
identifier of the failing message.

*rc* HSIMSG return code.

*rs* HSIMSG reason code.

**System action:** Terminates with a condition code of 20.

**Operator response:** Inform the system programmer.

**System programmer response:** Ensure Joblib/Steplib contains the library where the HSIFMSG message module resides. If you cannot resolve this issue then contact IBM support.

**Module:** HSIIM2D HSIIS2D

# HSII - REXX utility messages

### Return codes

*Table 93. Return codes and their meaning*

| Return code | Description |
|---|---|
| 0 | No errors encountered. All requests processed successfully. |
| 4 | Input/Output error in one or more program libraries. |
| 8 | Error - Incomplete data. Processing continues. OPEN or other system service error. |
| 12 | Syntax error. |
| 16 | Unrecoverable error. |
| 20 | Disastrous error. No requests processed. SYSPRINT file cannot be used. |

### Message suffix codes

*Table 94. Message suffix codes and associated condition codes*

| Suffix | Meaning | Raises minimum condition code to: |
|---|---|---|
| I | Information message | 0 |

*Table 94. Message suffix codes and associated condition codes (continued)*

| Suffix | Meaning | Raises minimum condition code to: |
|---|---|---|
| W | Warning message | 4 |
| E | Error message | 8 |
| S | Severe error message | 12 |
| U | Unrecoverable error message | 16 |

**Message texts and explanations**

All numeric completion codes of system services reported in these messages are in hexadecimal unless otherwise stated.

**HSII001I READ FAILED FOR SYSIN, RC=*rc***

**Explanation:** The HSIICUST program could not read commands from the SYSIN DDname.

In the message text:

*rc* return code from EXECIO

**System action:** The program terminates and takes no actions.

**Operator response:** Correct the JCL and provide a SYSIN DD with valid control statements.

**System programmer response:** None.

**Module:** HSIICUST

**HSII002I REQUIRED PARAMETER *prm* IS MISSING FROM SYSIN**

**Explanation:** The HSIICUST program did not find the required parm in the SYSIN supplied by the user.

In the message text:

*prm*
name of the parm that is missing.

**System action:** The program terminates and takes no actions.

**Operator response:** Correct the SYSIN and supply the required parm.

**System programmer response:** None.

**Module:** HSIICUST

**HSII003I THE DEFAULT VALUE "*prm*=*dft*" IS BEING USED.**

**Explanation:** The submitted HSISCUST Job SYSIN did not contain this parameter and the default value will now be used.

In the message text:

*prm*
name of the parameter.

*dft*
supplied default.

**System action:** The program continues and uses the default as documented in the message.

**Operator response:** If the default value is to be overridden supply the parameter value in the HSISCUST Job SYSIN stream then resubmit.

**System programmer response:** None.

**Module:** HSIICUST

**HSII004I ALLOCATION OF DATASET *dsn* TO *dd* FAILED, RC=*rc***

**Explanation:** The HSIICUST program could not allocate the dataset to the ddname.

In the message text:

*dsn*
name of the dataset that failed allocation.

*dd* DD name to be allocated.

*rc* return code from the allocate command.

**System action:** The program terminates and takes no actions.

**Operator response:** Check the return code from the allocate command in the TSO commands manual. Correct the options and try running the program again. The probable option in error is HSIINST.

**System programmer response:** None.

**Module:** HSIICUST

**HSII005I *func* FAILED FOR *rsc*, RC=*rc***

**Explanation:** The HSIICUST/HSIISCRT program could not perform a required ISPF function because an error occured during the function execution.

In the message text:

*func*
name of ISPF function that failed.

*rsc*
resource that caused the failure.

*rc* return code from the ISPF function.

**System action:** The program terminates and takes no actions. The program may have written out JCL and parameter members. These members should be treated as suspect as they might contain errors in them due to the nature of this error.

**Operator response:** Check that the options specified do not exceed the field length requirements of the various options. If you cannot resolve this issue then gather appropriate diagnostic materials and contact IBM support.

**System programmer response:** None.

**Module:** HSIICUST HSIISCRT

---

**HSII006I MODEL DATASET SHSISAMP HAS NOT BEEN FOUND.**

**Explanation:** The HSIICUST program could not find the SHSISAMP dataset created during installation.

**System action:** The program terminates and takes no actions.

**Operator response:** Check that the HSI option is correctly specified and that the installation target libraries are available to the customization program.

**System programmer response:** None.

**Module:** HSIICUST

---

**HSII007I CUSTOMIZATION TERMINATES ...**

**Explanation:** The HSIICUST program encountered an error during execution.

**System action:** The program terminates and takes no further action.

**Operator response:** Check the previous message which will identify the component causing the problem.

**System programmer response:** None.

**Module:** HSIICUST

---

**HSII008I** *prm* **PARAMETER VALUE** *pval* **IS >** *plen* **CHARACTERS.**

**Explanation:** The HSIICUST program found a parameter value with a length greater than the allowed value for that parm.

In the message text:

*prm*
name of HSISCUST parameter that failed.

*pval*
contents of the PARM.

*plen*
length of Parameter value.

**System action:** The program terminates and takes no actions.

**Operator response:** Check the parameter in question and re-submit the HSISCUST JCL after correcting the length error.

**System programmer response:** None.

**Module:** HSIICUST

---

**HSII009I** *prm* **DATASET** *dsn* **WAS NOT FOUND ON SYSTEM** *systm*

**Explanation:** The HSIICUST program found a parameter value, which requires a Dataset name. This Dataset name could not be found on the system.

In the message text:

*prm*
name of HSISCUST parameter that failed.

*dsn*
dataset name associated with the PARM.

*systm*
Operating System name.

**System action:** The program terminates and takes no actions.

**Operator response:** Check the parameter in question and re-submit the HSISCUST JCL after correcting the DSN error.

**System programmer response:** None.

**Module:** HSIICUST

---

**HSII010I** *verb* **STATEMENT VERB NOT ONE OF** *list*

**Explanation:** During syntax parsing for a statement the verb found does not match any of the valid verbs for the program.

In the message text:

*verb*
word that is not a valid verb.

*list*
list of valid verbs.

**System action:** The program terminates and takes no actions.

**Operator response:** Update the statements to the program to correct the verb in error and supply a correct verb.

**System programmer response:** None.

**Module:** HSIIKBT HSIIODEL HSIISCRT

---

**HSII011I** *word* **NOT VALID FOR VERB** *verb*

**Explanation:** During syntax parsing for a statement, a word was found that does not match the syntax of the statement for the verb that is being processed.

In the message text:

*word*
word that is not valid for the statement syntax for a verb.

*verb*
the verb of the statement that encountered the error.

**System action:** The program terminates and takes no actions.

**Operator response:** Update the statements to the program to correct the statement in error.

**System programmer response:** None.

**Module:** HSIIKBT HSIIODEL HSIISCRT

---

**HSII012I** **PARAMETER** *prm* **IS NOT A VALID HSISCUST PARAMETER**

**Explanation:** An invalid HSISCUST SYSIN parameter has been found.

In the message text:

*prm*
parameter that is invalid.

**System action:** The program terminates.

**Operator response:** Remove the invalid parameter from the HSISCUST Job SYSIN then resubmit.

**System programmer response:** None.

**Module:** HSIICUST

---

**HSII013I** **PARAMETER** *prm* **HAS A NULL VALUE**

**Explanation:** The submitted HSISCUST Job SYSIN has encountered a parameter with a NULL value.

In the message text:

*prm*
name of the parameter that is null.

**System action:** The program terminates.

**Operator response:** Ensure that the HSISCUST parameter has a valid parameter value in the HSISCUST Job SYSIN then resubmit.

**System programmer response:** None.

**Module:** HSIICUST

---

**HSII014I** **PARAMETER VALUE FOR** *prm* **HAS UNBALANCED QUOTES, PARAMETER VALUE IS** *pval*

**Explanation:** An HSISCUST parameter contains unbalanced quotes.

In the message text:

*prm*
name of the parameter with unbalanced quotes.

*pval*
parameter value.

**System action:** The program terminates

**Operator response:** Ensure that the Parameter value is enclosed within single quotation marks then resubmit the HSISCUST Job.

**System programmer response:** None.

**Module:** HSIICUST

---

**HSII015I** **DATASET** *dsn stat*

**Explanation:** The HSIICUST program identifies the status of the output datasets that it is going to use.

In the message text:

*dsn*
name of output dataset.

*stat*
status of the output dataset.

**System action:** The HSIICUST program continues processing.

**Operator response:** Informational message, no action required.

**System programmer response:** None.

**Module:** HSIICUST

---

**HSII016I** **UNMATCHED COMMENT DELIMITER IN HSISCUST STATEMENT:** *stmnt*

**Explanation:** The HSIICUST program found an error in the comment delimiters passed from the HSISCUST SYSIN stream.

In the message text:

*stmnt*
statement where the error occurred.

**System action:** The program terminates and takes no actions.

**Operator response:** Correct the HSISCUST SYSIN statements and provide valid comment delimiters, `/* */`.

**System programmer response:** None.

**Module:** HSIICUST

---

**HSII017I PARAMETER *prm* MUST HAVE THE 1ST CHARACTER AS A VALUE BETWEEN A AND Z**

**Explanation:** The HSISCUST Job has encountered a parameter in the SYSIN DD stream where the 1st character is not alphabetic. This parameter value must start with a value between A and Z.

In the message text:

*prm*
name of the parameter that has a non alphabetic 1st character.

**System action:** The program terminates.

**Operator response:** Ensure that the HSISCUST parameter has a value between A and Z then resubmit the Job.

**System programmer response:** None.

**Module:** HSIICUST

---

**HSII018I *prm* VALUE *pval* MUST BE WITHIN THE VALID RANGE OF *val1* TO *val2***

**Explanation:** The HSIICUST program encountered a parameter that was outside the valid range of values allowed.

In the message text:

*prm*
name of the parameter in error.

*pval*
value of parameter in error.

*val1*
minimum value of valid range.

*val2*
maximum value of valid range.

**System action:** The program terminates and takes no actions.

**Operator response:** Correct the parameter in error and rerun the HSISCUST Job.

**System programmer response:** None.

**Module:** HSIICUST

---

**HSII019I DBTYPE MUST BE THE FIRST PARM IN THE SYSIN STREAM. CURRENT VALUE IS *prm***

**Explanation:** The HSIICUST program found an error in the first parameter passed from the HSISCUST SYSIN stream.

In the message text:

*prm*
statement where the error occurred.

**System action:** The program terminates and takes no actions.

**Operator response:** Correct the HSISCUST SYSIN statements and ensure that DBTYPE= is the first entry. Comment statements and blank lines may precede the DBTYPE= entry.

**System programmer response:** None.

**Module:** HSIICUST

---

**HSII020I TAILORING PARAMETERS:**

**Explanation:** HSIICUST progress message.

**System action:** The program continues.

**Operator response:** This is an informational progress message and no further action is required.

**System programmer response:** None.

**Module:** HSIICUST

---

**HSII021I CREATING POST-INSTALLATION DATASETS:**

**Explanation:** HSIICUST progress message.

**System action:** The program continues.

**Operator response:** This is an informational progress message and no further action is required.

**System programmer response:** None.

**Module:** HSIICUST

---

**HSII022I APPLYING TAILORING INFORMATION TO POST-INSTALLATION MEMBERS:**

**Explanation:** HSIICUST progress message.

**System action:** The program continues.

**Operator response:** This is an informational progress message and no further action is required.

**System programmer response:** None.

**Module:** HSIICUST

---

**HSII023I POST-INSTALLATION CUSTOMIZATION COMPLETE.**

**Explanation:** HSIICUST completion message.

**System action:** The program ends successfully.

**Operator response:** This is an informational progress message and no further action is required.

**System programmer response:** None.

**Module:** HSIICUST

**HSII024I** **PARAMETER *prm* CONTAINS AN EXTRANEOUS VALUE: *xval***

**Explanation:** An HSISCUST parameter contains an extraneous value.

In the message text:

*prm*
name of the parameter with an extraneous value.

*xval*
the extraneous value(s).

**System action:** The program terminates

**Operator response:** Ensure that the Parameter value is correctly defined and that any comments are enclosed within comment delimiters.

**System programmer response:** None.

**Module:** HSIICUST

**HSII025I** **PARAMETER VALUE FOR *prm* MUST BE IN QUOTES, PARAMETER VALUE IS *pval***

**Explanation:** An HSISCUST parameter value contains no quotes.

In the message text:

*prm*
name of the parameter.

*pval*
parameter value.

**System action:** The program terminates

**Operator response:** Ensure that the Parameter value is enclosed within single quotation marks then resubmit the HSISCUST Job.

**System programmer response:** None.

**Module:** HSIICUST

**HSII026I** **PARAMETER *prm* MUST BE A VALUE BETWEEN A-Z OR 0-9**

**Explanation:** The HSISCUST Job has encountered a parameter in the SYSIN DD stream where the 1st character is not alphanumeric. This parameter value must be a value between A-Z or 0-9

In the message text:

*prm*
name of the parameter that has a non alphanumeric first character.

**System action:** The program terminates.

**Operator response:** Ensure that the HSISCUST parameter has a value between A-Z or 0-9 then resubmit the Job.

**System programmer response:** None.

**Module:** HSIICUST

**HSII027I** ***prm* VALUE *pval* FAILED. VALUE MUST BE DB2, SQLITE OR REMOTE.**

**Explanation:** The HSIICUST program encountered an invalid DBTYPE value.

In the message text:

*prm*
name of the parameter in error.

*pval*
value of parameter in error.

**System action:** The program terminates and takes no actions.

**Operator response:** Correct the DBTYPE value and rerun the HSISCUST Job.

**System programmer response:** None.

**Module:** HSIICUST

**HSII028I** **PARAMETERS DB AND DBGKB CANNOT HAVE THE SAME DATABASE NAME: *db***

**Explanation:** The HSIICUST program found that the values of DB and DBGKB are the same.

In the message text:

*db* duplicate Database name.

**System action:** The program terminates and takes no actions.

**Operator response:** Correct the HSISCUST SYSIN statements and provide unique values for both DB and DBGKB.

**System programmer response:** None.

**Module:** HSIICUST

**HSII100I** ***prm* MISSING FROM CONFIGURATION.**

**Explanation:** The utility requires the parameter in the TPARAM/SYSIN stream.

In the message text:

*prm*
parameter that is missing.

**System action:** The program terminates and takes no actions.

**Operator response:** Update the parameters in the TPARAM/SYSIN DD to add the missing parameter.

**System programmer response:** None.

**Module:** HSIIKBT HSIIODEL HSIISCRT

**HSII107I** *svc* **FROM** *rsc* **FAILED, RC=***rc*

**Explanation:** An error has occurred executing the service for the resource specified. The service issued the return code mentioned.

In the message text:

*svc*
service that failed.

*rsc*
resource that failed.

*rc* return code from service.

**System action:** The program stops processing statements. No changes have been made.

**Operator response:** Report this error to the System Programmer.

**System programmer response:** For "EXECIO READ" service, this means that the resource specified (ddname) is missing or empty. If you cannot resolve this issue then gather appropriate diagnostic materials and contact IBM support.

**Module:** HSIIKBT HSIIODEL HSIISCRT

---

**HSII108I** **SQL** *verb* **FAILED, SQLCODE=***sqlcode*

**Explanation:** An error has occurred executing the SQL verb for the table specified.

In the message text:

*verb*
SQL verb and table name.

*sqlcode*
SQLCODE from failing statement.

**System action:** The program stops processing statements. The current statement changes to DB2 tables are backed out.

**Operator response:** Report this error to the System Programmer.

**System programmer response:** If you cannot resolve this issue then gather appropriate diagnostic materials and contact IBM support.

**Module:** HSIIKBT HSIIODEL HSIISCRT

---

**HSII202I** **MISSING PID** *pid***.**

**Explanation:** The SCRT Import utility has detected a missing PID.

In the message text:

*pid*
missing PID.

**System action:** The program continues.

**Operator response:** Supply IBM support with the missing PID number and request a Global Knowledge Base (GKB) refresh.

**System programmer response:** None.

**Module:** HSIISCRT

---

**HSII300I** **ERROR WRITING TO** *ddn***.**

**Explanation:** The XML Export utility has a problem writing the XML file.

In the message text:

*ddn*
DDNAME of the file

**System action:** The program terminates.

**Operator response:** Check the Return Code and any preceding messages.

**System programmer response:** None.

**Module:** HSIIKBT

---

**HSII301I** **NUMBER OF LINES WRITTEN TO SWKBTXML DD IS** *ocnt***.**

**Explanation:** No. of Lines written to SWKBTXML DD by the XML Export utility.

In the message text:

*ocnt*
lines written to SWKBTXML DD.

**System action:** The program continues and takes no actions.

**Operator response:** Informational message, no action required.

**System programmer response:** None.

**Module:** HSIIKBT

---

**HSII302I** **SQL WARNING FOR** *warn***.**

**Explanation:** SQL warning was issued from a command

In the message text:

*warn*
SQL warning.

**System action:** The program continues.

**Operator response:** None.

**System programmer response:** None.

**Module:** HSIIKBT

---

**HSII303I SQL ERROR FOR *err*.**

**Explanation:** The XML Export Utility has encountered an error.

In the message text:

*err*
SQL Error.

**System action:** The program terminates and takes no actions.

**Operator response:** Examine the SQL return code to determine the cause of the error. Inform the system programmer.

**System programmer response:** If you cannot resolve this issue then gather appropriate diagnostic materials and contact IBM support.

**Module:** HSIIKBT

**HSII304I *err*.**

**Explanation:** The XML Export Utility has encountered a DSNREXX error.

In the message text:

*err*
DSNREXX error.

**System action:** The program terminates and takes no actions.

**Operator response:** Examine the preceding error messages to determine the error. Inform the system programmer.

**System programmer response:** If you cannot resolve this issue then gather appropriate diagnostic materials and contact IBM support.

**Module:** HSIIKBT

**HSII305I INVALID SCHEMA *sch*.**

**Explanation:** The XML Export Utility has encountered a problem with an invalid Schema.

In the message text:

*sch*
schema name

**System action:** The program terminates and takes no actions.

**Operator response:** Ensure that the correct Schema is being used.

**System programmer response:** None.

**Module:** HSIIKBT

**HSII999U MODULE HSIIMSG FAILED - MSGID=*msgid* RC=*rc* RS=*rs***

**Explanation:** HSIMSG was called to produce a message text, but the call failed.

In the message text:

*msgid*
identifier of the failing message.

*rc* HSIMSG return code.

*rs* HSIMSG reason code.

**System action:** Terminates with a condition code of 20.

**Operator response:** Inform the system programmer.

**System programmer response:** Contact IBM Support.

**Module:** HSIIINQ

# HSIP - Inquisitor for z/OS messages and codes

### Return codes

*Table 95. Return codes and their meaning*

| Return code | Description |
|---|---|
| 0 | No errors encountered. All requests processed successfully. |
| 4 | Input/Output error in one or more program libraries. |
| 8 | Error - Data collection is incomplete. Processing continues. The error is in a system service such as OPEN or DYNALLOC. Data that is collected can be processed normally. |
| 12 | Syntax error. |
| 16 | Unrecoverable error. No requests processed. SYSIN or HSIPZIP or HSIPOUT File cannot be used, or unsupported Operating System. |
| 20 | Disastrous error. No requests processed. SYSPRINT file cannot be used. |

### Message suffix codes

*Table 96. Message suffix codes and associated condition codes*

| Suffix | Meaning | Raises minimum condition code to: |
|---|---|---|
| I | Information message | 0 |
| W | Warning message | 4 |
| E | Error message | 8 |
| S | Severe error message | 12 |
| U | Unrecoverable error message | 16 |

### Message texts and explanations

All numeric completion codes of system services reported in these messages are in hexadecimal unless otherwise stated.

**HSIP000U NO USABLE SYSPRINT FILE**

**Explanation:** The OPEN of the SYSPRINT file failed. Note: This message is issued by WTO with ROUTCDE=(2,11). All other messages are written to the SYSPRINT file.

**System action:** Terminates with a condition code of 20.

**Operator response:** Ensure a usable SYSPRINT file is allocated. The program overrides any unacceptable DCB values.

**System programmer response:** None.

**Module:** HSIPINQ

**HSIP001U CANNOT OPEN SYSIN FILE**

**Explanation:** The OPEN of the SYSIN file failed.

**System action:** Terminates with a condition code of 16.

**Operator response:** Ensure a usable SYSIN file is allocated.

**System programmer response:** None.

**Module:** HSIPINQ

**HSIP004S UNKNOWN VERB "*verb*"**

**Explanation:** Parsing detected unrecognised data when looking for a verb.

In the message text:

*verb*
name of the encountered verb.

**System action:** Terminates with a condition code of 12.

**Operator response:** Correct the SYSIN file contents and rerun the program.

**System programmer response:** None.

**Module:** HSIPINQ

---

**HSIP005S UNKNOWN OPERAND "*op*"**

**Explanation:** Parsing detected unrecognised data when looking for an operand.

In the message text:

*op* name of the encountered operand.

**System action:** Terminates with a condition code of 12.

**Operator response:** Correct the SYSIN file contents and rerun the program.

**System programmer response:** None.

**Module:** HSIPINQ

---

**HSIP006S UNEXPECTED OPEN PARENTHESIS ENCOUNTERED**

**Explanation:** Parsing detected an open parenthesis at an unexpected location.

**System action:** Terminates with a condition code of 12.

**Operator response:** Correct the SYSIN file contents and rerun the program.

**System programmer response:** None.

**Module:** HSIPINQ

---

**HSIP007S UNEXPECTED CLOSE PARENTHESIS ENCOUNTERED**

**Explanation:** Parsing detected a close parenthesis at an unexpected location.

**System action:** Terminates with a condition code of 12.

**Operator response:** Correct the SYSIN file contents and rerun the program.

**System programmer response:** None.

**Module:** HSIPINQ

---

**HSIP008S EXPECTED OPEN PARENTHESIS MISSING**

**Explanation:** Parsing did not detect the expected open parenthesis.

**System action:** Terminates with a condition code of 12.

**Operator response:** Correct the SYSIN file contents and rerun the program.

**System programmer response:** None.

**Module:** HSIPINQ

---

**HSIP009S EXPECTED CLOSE PARENTHESIS MISSING**

**Explanation:** Parsing did not detect the expected close parenthesis.

**System action:** Terminates with a condition code of 12.

**Operator response:** Correct the SYSIN file contents and rerun the program.

**System programmer response:** None.

**Module:** HSIPINQ

---

**HSIP010U OPERATING SYSTEM NOT SUPPORTED - CODE "*code*"**

**Explanation:** The value of the byte at CVTDCB was not X'9B'.

In the message text:

*code*
hexadecimal value of first byte of CVTDCB.

**System action:** Terminates with a condition code of 16.

**Operator response:** This version of the Inquisitor cannot be run on this Operating System. If necessary, gather appropriate diagnostic materials and contact IBM support.

**System programmer response:** None.

**Module:** HSIPINQ

---

**HSIP011I MISSING CLOSE PARENTHESIS ASSUMED**

**Explanation:** End-of-file was detected for SYSIN before an expected close parenthesis was detected.

**System action:** The request is accepted and processing continues.

**Operator response:** Correct the SYSIN file contents to avoid this message.

**System programmer response:** None.

**Module:** HSIPINQ

---

**HSIP012S MISSING OPERAND SUBPARAMETER FOR *spm***

**Explanation:** A required subparameter of an operand was not specified.

In the message text:

*spm*
name of the operand being processed.

**System action:** Terminates with a condition code of 12.

**Operator response:** Correct the SYSIN file contents and rerun the program.

**System programmer response:** None.

**Module:** HSIPINQ

---

**HSIP013S E-O-F INSTEAD OF EXPECTED CONTINUATION**

**Explanation:** End-of-file was detected for SYSIN instead of an expected record required to continue the current statement being parsed.

**System action:** Terminates with a condition code of 12.

**Operator response:** Correct the SYSIN file contents and rerun the program.

**System programmer response:** None.

**Module:** HSIPINQ

---

**HSIP014I COMPLETED REQUEST NUMBER** *rno* **- PROCESSING STATISTICS ARE:**

**Explanation:** Processing of a request has been completed. One or more messages follow containing the statistics for the request.

In the message text:

*rno*
sequence number of the request.

**System action:** Processing continues.

**Operator response:** None.

**System programmer response:** None.

**Module:** HSIPINQ

---

**HSIP015I VOLUMES=***vol* **DATASETS=***ds* **BAD-D/S=***dsbad* **PROGRAMS=***pgms*

**Explanation:** Processing of a request has been completed. Statistics related to the request are shown.

In the message text:

*vol*
count of volumes scanned for this request.

*ds* count of data sets successfully processed.

*dsbad*
count of data sets which could not be processed.

*pgms*
count of programs processed for this request.

**System action:** Processing continues.

**Operator response:** None.

**System programmer response:** None.

**Module:** HSIPINQ

---

**HSIP016I ACCEPTED REQUEST NUMBER** *rno*

**Explanation:** Parsing of a request has been completed successfully. The request is stored for subsequent processing.

In the message text:

*rno*
sequence number of the request.

**System action:** Processing continues.

**Operator response:** None required.

**System programmer response:** None.

**Module:** HSIPINQ

---

**HSIP017E DYNALLOC FAILURE: RC=***rc* **ERROR=***err* **INFO=***inf* **VOLUME=***vol*

**Explanation:** A data set could not be dynamically allocated. See message HSIP080I for the name of the dataset that incurred the problem.

In the message text:

*rc* return code of the DYNALLOC macro.

*err*
dynamic allocation return code (DARC).

*inf*
dynamic allocation information code.

*vol*
volume serial number of the data set.

**System action:** Processing of this data set is terminated.

**Operator response:** If necessary, rerun when the file is available for use. Note: The meanings of many DARC values are usually available in Appendix A of the ISPF Tutorial.

**System programmer response:** None.

**Module:** HSIPINQ

---

**HSIP018W VTOC DYNALLOC FAILURE: RC=***rc* **ERROR=***err* **INFO=***inf* **VOLUME=***vol*

**Explanation:** A VTOC could not be dynamically allocated.

In the message text:

*rc* return code of the DYNALLOC

*err*
dynamic allocation return code (DARC).

*inf*
dynamic allocation information code.

*vol*
volume serial number of the data set.

**System action:** Processing of this volume is terminated.

**Operator response:** If necessary, rerun when the VTOC is available for use to process this volume. Note: The meanings of many DARC values are usually available in Appendix A of the ISPF Tutorial.

**System programmer response:** None.

**Module:** HSIPINQ

---

**HSIP020I** *ocnt* **INQUISITOR OUTPUT RECORDS WRITTEN**

**Explanation:** Processing has concluded and all data files have been closed.

In the message text:

*ocnt*
number of records written.

**System action:** Termination continues.

**Operator response:** None required.

**System programmer response:** None.

**Module:** HSIPINQ

---

**HSIP021S INVALID OPERAND SUBPARAMETER FOR** *spm*

**Explanation:** The specified subparameter of an operand was not valid.

In the message text:

*spm*
name of the operand being processed.

**System action:** Terminates with a condition code of 12.

**Operator response:** Correct the SYSIN file contents and rerun the program.

**System programmer response:** None.

**Module:** HSIPINQ

---

**HSIP022W I/O ERR MEMBER** *mbr* **IN** *dsn*

**Explanation:** An I/O error was encountered while reading the contents of a load module.

In the message text:

*mbr*
name of the program being processed.

*dsn*
name of the data set being processed.

**System action:** Processing of this member continues.

**Operator response:** None required.

**System programmer response:** None.

**Module:** HSIPINQ

---

**HSIP023W ABEND** *abend* **IN OPEN FOR** *dsn*

**Explanation:** An abnormal end occurred while opening a data set.

In the message text:

*abend*
hexadecimal system abend and reason

*dsn*
name of the data set being processed.

**System action:** Processing of this data set is terminated.

**Operator response:** None required, but you may wish to exclude the data set from processing, or correct the cause of the abend.

**System programmer response:** None.

**Module:** HSIPINQ

---

**HSIP024S BAD UCBSCAN RETURN CODE OF HEX** *rc*

**Explanation:** An unexpected return code was received from UCBSCAN.

In the message text:

*rc* hexadecimal return code from UCBSCAN

**System action:** Processing of volume scanning for this request is terminated.

**Operator response:** Rerun the program when no dynamic reconfiguration changes are being implemented.

**System programmer response:** None.

**Module:** HSIPINQ

---

**HSIP025U CANNOT OPEN HSIPOUT FILE**

**Explanation:** The OPEN of the HSIPOUT file failed.

**System action:** Terminates with a condition code of 16.

**Operator response:** Ensure that the allocated HSIPOUT file is usable, or omit the HSIPOUT file in favour of using the HSIPZIP file.

**System programmer response:** None.

**Module:** HSIPINQ

---

**HSIP026E I/O ERROR ENCOUNTERED READING VTOC OF VOLUME *vol* ON DEVICE *dev***

**Explanation:** An I/O error was encountered while reading a VTOC.

In the message text:

*vol*
volume serial number being processed.

*dev*
device number of the volume.

**System action:** Processing of this track of the VTOC is terminated.

**Operator response:** None required, but you may wish to exclude the volume from processing, or correct the cause of the I/O error.

**System programmer response:** None.

**Module:** HSIPINQ

**HSIP028U CANNOT OPEN HSIPDMP FILE**

**Explanation:** The OPEN of the HSIPDMP file failed after DUMPTEXT was specified.

**System action:** Terminates with a condition code of 16.

**Operator response:** Ensure a usable HSIPDMP file is allocated, or remove all DUMPTEXT operand's from the contents of the SYSIN file. The DUMPTEXT operand should only be specified at the request of IBM support.

**System programmer response:** None.

**Module:** HSIPINQ

**HSIP029I TEXT-DUMPS=*cnt***

**Explanation:** Processing of a request with DUMPTEXT specified has completed. This message follows HSIP015I.

In the message text:

*cnt*
count of load module text blocks written.

**System action:** Processing continues.

**Operator response:** None required. The DUMPTEXT operand should only be specified at the request of IBM support.

**System programmer response:** None.

**Module:** HSIPINQ

**HSIP030I "DUMPTEXT" OPERAND IGNORED FOR "SCANDIR" VERB**

**Explanation:** A DUMPTEXT operand was encountered for a SCANDIR request. That is, the possible dumping of load module text blocks was specified in a request which does not have access to text blocks.

**System action:** The DUMPTEXT operand is ignored and processing continues.

**Operator response:** Remove the DUMPTEXT operand to avoid this message. The DUMPTEXT operand should only be specified at the request of IBM support.

**System programmer response:** None.

**Module:** HSIPINQ

**HSIP031I BAD SELECTION CRITERIA FOR *dsn***

**Explanation:** Processing of a data set was specified but attributes did not match other selection criteria also specified in the request. This message is followed by HSIP038I which details the cause.

In the message text:

*dsn*
name of the data set being processed.

**System action:** Processing of this data set is terminated.

**Operator response:** If this data set is a program library which should be processed by the Inquisitor then modify or remove the conflicting selection criteria.

**System programmer response:** None.

**Module:** HSIPINQ

**HSIP032I OBTAIN FAILED RC=*rc* VOLUME *vol***

**Explanation:** The system could not read the VTOC entry for the data set named in the HSIP033I message which follows this message. This message is only issued when a program parameter of "DSNMSG" or "ALLMSG" is specified.

In the message text:

*rc* hexadecimal return code of the OBTAIN macro.

*vol*
volume serial number being processed.

**System action:** Processing of this data set is terminated.

**Operator response:** Ensure the relevant catalog entry is correct. Ensure the relevant volume is online and available to the system. Ensure there is no I/O error in the relevant volume's VTOC. If necessary, gather appropriate diagnostic materials and contact IBM support.

**System programmer response:** None.

**Module:** HSIPINQ

---

**HSIP033I OBTAIN FAILED FOR DATA SET** *dsn*

**Explanation:** The system could not read the VTOC entry for the data set on the volume named in the previous HSIP032I message. This message is only issued when a program parameter of "DSNMSG" or "ALLMSG" is specified.

In the message text:

*dsn*
name of the data set being processed.

**System action:** Processing of this data set is terminated.

**Operator response:** Ensure the relevant catalog entry is correct. Ensure the relevant volume is online and available to the system. Ensure there is no I/O error in the relevant volumes VTOC. If necessary, gather appropriate diagnostic materials and contact IBM support.

**System programmer response:** None.

**Module:** HSIPINQ

---

**HSIP034I REFER DATE WAS** *date* **FOR** *dsn*

**Explanation:** A load library was opened. The reference date of the data set before the OPEN is reported in this message. This message is only issued when a program parameter of "DSNMSG" or "ALLMSG" is specified.

In the message text:

*date*
the Julian reference date from the VTOC entry.

*dsn*
name of the data set being processed.

**System action:** Processing of this data set continues.

**Operator response:** None required.

**System programmer response:** None.

**Module:** HSIPINQ

---

**HSIP036E OPEN ERROR ENCOUNTERED READING VTOC OF VOLUME** *vol* **ON DEVICE** *dev*

**Explanation:** The VTOC of the volume shown could not be opened.

In the message text:

*vol*
volume serial number being processed.

*dev*
device number of the volume.

**System action:** Processing of this track of the VTOC is terminated.

**Operator response:** None required, but you may wish to exclude the volume from processing, or correct the cause of the I/O error.

**System programmer response:** None.

**Module:** HSIPINQ

---

**HSIP037E SECURITY ACCESS DENIED FOR** *dsn*

**Explanation:** A RACROUTE macro determined the program had insufficient security access to read the data set.

In the message text:

*dsn*
name of the data set being processed.

**System action:** Processing of this data set is terminated.

**Operator response:** Contact Security Administration to obtain sufficient security access to read the data set or exclude the data set from processing.

**System programmer response:** None.

**Module:** HSIPINQ

---

**HSIP038I BAD SELECTION CRITERIA WAS** *dsn*

**Explanation:** Processing of a data set was specified but attributes did not match other selection criteria also specified in the request. This message follows HSIP031I which shows the data set name.

In the message text:

*dsn*
cause of the data set processing failure.

**System action:** Processing of this data set is terminated.

**Operator response:** If this data set is a program library which should be processed by the Inquisitor then modify or remove the conflicting selection criteria.

**System programmer response:** None.

**Module:** HSIPINQ

---

**HSIP039S ALL POSSIBLE DATA SETS ARE EXCLUDED**

**Explanation:** An exclusion mask has been specified which excludes all possible data sets included by a selection mask. Both masks are shown after this message.

**System action:** Terminates with a condition code of 12.

**Operator response:** Modify or remove the conflicting selection criteria.

**System programmer response:** None.

**Module:** HSIPINQ

---

**HSIP040S ALL POSSIBLE DASD VOLUMES ARE EXCLUDED**

**Explanation:** An exclusion mask has been specified which excludes all possible DASD volumes included by a selection mask. Both masks are shown after this message.

**System action:** Terminates with a condition code of 12.

**Operator response:** Modify or remove the conflicting selection criteria.

**System programmer response:** None.

**Module:** HSIPINQ

---

**HSIP041S ALL POSSIBLE PROGRAMS ARE EXCLUDED**

**Explanation:** An exclusion mask has been specified which excludes all possible programs included by a selection mask. Both masks are shown after this message.

**System action:** Terminates with a condition code of 12.

**Operator response:** Modify or remove the conflicting selection criteria.

**System programmer response:** None.

**Module:** HSIPINQ

---

**HSIP042S ALL POSSIBLE MODULES ARE EXCLUDED**

**Explanation:** An exclusion mask has been specified which excludes all possible modules included by a selection mask.

**System action:** Terminates with a condition code of 12.

**Operator response:** Modify or remove the conflicting selection criteria. If no CSECT-level records are required then omit both MODULE and XMODULE operands.

**System programmer response:** None.

**Module:** HSIPINQ

---

**HSIP043I "MODULE"/"CSECT" OPERAND IGNORED FOR "SCANDIR" VERB**

**Explanation:** A MODULE operand was encountered for a SCANDIR request. That is, the output of program structure data was requested in a request which does not have access to this data.

**System action:** The MODULE operand is ignored and processing continues.

**Operator response:** Remove the MODULE operand to avoid this message.

**System programmer response:** None.

**Module:** HSIPINQ

---

**HSIP044I "XMODULE"/"XCSECT" OPERAND IGNORED FOR "SCANDIR" VERB**

**Explanation:** An XMODULE operand was encountered for a SCANDIR request. That is, the output of program structure data was implied in a request which does not have access to this data.

**System action:** The XMODULE operand is ignored and processing continues.

**Operator response:** Remove the XMODULE operand to avoid this message.

**System programmer response:** None.

**Module:** HSIPINQ

---

**HSIP045I THE "XDSNAME" MASK IS NOT A SUBSET OF ANY "DSNAME" MASK**

**Explanation:** The mask specified in the XDSNAME operand excludes possible values not included in the DSNAME operand. This message is issued to highlight possible inconsistencies in a request.

**System action:** Processing continues.

**Operator response:** Specify the XDSNAME operand as a further qualification of the DSNAME operand to avoid this message.

**System programmer response:** None.

**Module:** HSIPINQ

---

**HSIP046I THE "XVOLUME" MASK IS NOT A SUBSET OF ANY "VOLUME" MASK**

**Explanation:** The mask specified in the XVOLUME operand excludes possible values not included in the VOLUME operand. This message is issued to highlight possible inconsistencies in a request.

**System action:** Processing continues.

**Operator response:** Specify the XVOLUME operand as a further qualification of the VOLUME operand to avoid this message.

**System programmer response:** None.

**Module:** HSIPINQ

---

**HSIP047I THE "XPROGRAM" MASK IS NOT A SUBSET OF ANY "PROGRAM" MASK**

**Explanation:** The mask specified in the XPROGRAM operand excludes possible values not included in the PROGRAM operand. This message is issued to

highlight possible inconsistencies in a request.

**System action:** Processing continues.

**Operator response:** Specify the XPROGRAM operand as a further qualification of the PROGRAM operand to avoid this message.

**System programmer response:** None.

**Module:** HSIPINQ

---

**HSIP048I THE "XMODULE" MASK IS NOT A SUBSET OF ANY "MODULE" MASK**

**Explanation:** The mask specified in the XMODULE operand excludes possible values not included in the MODULE operand. This message is issued to highlight possible inconsistencies in a request.

**System action:** Processing continues.

**Operator response:** Specify the XMODULE operand as a further qualification of the MODULE operand to avoid this message.

**System programmer response:** None.

**Module:** HSIPINQ

---

**HSIP049I THE "XSTOGROUP" MASK IS NOT A SUBSET OF ANY "STOGROUP" MASK**

**Explanation:** The mask specified in the XSTOGROUP operand excludes possible values not included in the STOGROUP operand. This message is issued to highlight possible inconsistencies in a request.

**System action:** Processing continues.

**Operator response:** Specify the XSTOGROUP operand as a further qualification of the STOGROUP operand to avoid this message.

**System programmer response:** None.

**Module:** HSIPINQ

---

**HSIP050I MODULES=*cnt***

**Explanation:** Processing of a request with MODULE specified has completed. This message follows HSIP015I.

In the message text:

*cnt*
count of CSECTs processed in this request.

**System action:** Processing continues.

**Operator response:** None required.

**System programmer response:** None.

**Module:** HSIPINQ

---

**HSIP051I ******** PARSE ONLY REQUEST PROCESSED - NO ACTION TAKEN ********

**Explanation:** Processing of a SCANCMD request has completed.

**System action:** Processing continues.

**Operator response:** None required.

**System programmer response:** None.

**Module:** HSIPINQ

---

**HSIP052U MISSING HSIPOUT AND HSIPZIP FILES**

**Explanation:** Neither an HSIPOUT nor an HSIPZIP file is allocated. At least one output file is required.

**System action:** Terminates with a condition code of 16.

**Operator response:** Specify an output file and rerun the job.

**System programmer response:** None.

**Module:** HSIPINQ

---

**HSIP053U COMPRESSION SUBROUTINE ERROR**

**Explanation:** While processing the HSIPZIP file the compression subroutine encountered an error. The error message from the compression subroutine immediately follows this message.

**System action:** Terminates with a condition code of 16.

**Operator response:** Correct the error described in the message from the compression subroutine. If necessary, gather appropriate diagnostic materials and contact IBM support

**System programmer response:** None.

**Module:** HSIPINQ

---

**HSIP054I "FULLDIR" OPERAND IGNORED FOR "SCANDIR" VERB**

**Explanation:** A FULLDIR operand was encountered for a SCANDIR request. That is, the processing of load module member data was specified in a request which does not have access to this data.

**System action:** The FULLDIR operand is ignored and processing continues.

**Operator response:** Remove the FULLDIR operand to avoid this message.

**System programmer response:** None.

**Module:** HSIPINQ

**HSIP056I** *date time* **COMMENCING SCAN OF VOLUME** *vol* **ON UNIT** *unit*

**Explanation:** A request without the CATALOG keyword began processing a DASD volume. This message provides feedback on the progress of long-running Inquisitor requests.

In the message text:

*date*
date of message.

*time*
time of message.

*vol*
serial number of volume.

*unit*
device number of volume.

**System action:** Processing continues.

**Operator response:** None required.

**System programmer response:** None.

**Module:** HSIPINQ

**HSIP057E** **ABEND** *abend* **PROCESSING VTOC OF VOLUME** *vol* **ON UNIT** *unit*

**Explanation:** A request without the CATALOG keyword attempted to process a DASD volume VTOC and the OPEN or CLOSE abended. The volume may not not usable.

In the message text:

*abend*
hexadecimal system abend and reason codes.

*vol*
serial number of volume.

*unit*
device number of volume being processed.

**System action:** Processing of this volume is terminated.

**Operator response:** Vary the volume offline, and/or reformat the volume. Institute any appropriate volume recovery procedures.

**System programmer response:** None.

**Module:** HSIPINQ

**HSIP058S** **DUPLICATE OPERAND ENCOUNTERED:** *op*

**Explanation:** An input request was found to have the indicated operand specified more than once.

In the message text:

*op* name of the duplicate operand

**System action:** Terminates with a condition code of 12.

**Operator response:** Correct the SYSIN file contents and rerun the program.

**System programmer response:** None.

**Module:** HSIPINQ

**HSIP059W** **BINDER FAILURE FOR MEMBER** *mbr* **RC=***rc* **RS=***rs*

**Explanation:** The Binder could not successfully process a member of a PDSE.

In the message text:

*mbr*
name of the member being processed.

*rc* hexadecimal Binder FDA API return code.

*rs* hexadecimal Binder FDA API reason code.

**System action:** Terminates data collection for this member, writes out data already collected and continues processing the next member.

**Operator response:** None required.

**System programmer response:** The Binder Fast Data Access API return and reason codes provide more detailed indication of the cause.

**Module:** HSIPINQ

**HSIP060S** **SYMBOL SUBSTITUTION FAILURE - ASASYMBP RC=***rc*

**Explanation:** The system symbol substitution routine could not successfully perform symbol substitution. Data before and after substitution is shown in the SYSPRINT file.

In the message text:

*rc* hexadecimal return code.

**System action:** Terminates with a condition code of 12.

**Operator response:** Correct or remove the symbols in control statement input.

**System programmer response:** None.

**Module:** HSIPINQ

**HSIP061I** *pgm* **NON-REEDITABLE IN** *dsn*

**Explanation:** A program object in a PDSE was encountered which cannot be processed by the Program Binder. The program was bound with the NE or OVLY attribute. This message is only issued when a program parameter of "PGMMSG" or "ALLMSG" is specified.

In the message text:

*pgm*
name of program which cannot be processed.

*dsn*
name of the data set being processed.

**System action:** Further data collection for this member is terminated.

**Operator response:** None required.

**System programmer response:** None.

**Module:** HSIPINQ

---

**HSIP062S THE CATALOG REQUEST NEEDS EXACTLY ONE DSNAME MASK**

**Explanation:** A request with the CATALOG operand either omitted the DSNAME operand or specified more than one DSNAME mask.

**System action:** Terminates with a condition code of 12.

**Operator response:** Correct the SYSIN file contents and rerun the program. To process multiple data set name masks via the CATALOG specify a separate Inquisitor request for each mask. There is no programmed limit to the number of requests which can be processed in a single Inquisitor run.

**System programmer response:** None.

**Module:** HSIPINQ

---

**HSIP063S ALL POSSIBLE STORAGE GROUPS ARE EXCLUDED**

**Explanation:** An exclusion mask has been specified which excludes all possible storage groups included by the selection mask. Both masks are shown after this message.

**System action:** Terminates with a condition code of 12.

**Operator response:** Modify or remove the conflicting selection criteria.

**System programmer response:** None.

**Module:** HSIPINQ

---

**HSIP064W ABEND** *abend* **FOR** *mbr* **IN** *dsn*

**Explanation:** A subtask processing a program object from a PDSE has abended. The abend probably occurred in the Program Binder API.

In the message text:

*abend*
hexadecimal system abend code.

*mbr*
name of the member being processed.

*dsn*
name of the data set being processed.

**System action:** Data collected for this member so far is retained. Other Data Management abends may follow, especially in CLOSE processing, which are unrecoverable and may abend the main Inquisitor task.

**Operator response:** Exclude the programs causing the failure and rerun the Job.

**System programmer response:** None.

**Module:** HSIPINQ

---

**HSIP065S MCDS FILE FAILED VERIFICAION**

**Explanation:** The MCDS data definition (DD) was found to be unusable by the Inquisitor. One or more of the following is true: 1) The MCDS file could not be opened. Message HSIP066E follows. 2) The MCDS file is not a VSAM key-sequenced data set (KSDS). 3) The KSDS relative key position (RKP) is not zero (0). 4) The KSDS key length is not forty-four (44).

**System action:** Terminates with a condition code of 12.

**Operator response:** Either ensure that the Inquisitor has read access to DFHSM's MCDS, or change the Inquisitor request(s) so that the MCDS is not required. MCDS access is required if either or both of the REMIGRATE and NOML2 keywords are specified.

**System programmer response:** None.

**Module:** HSIPINQ

---

**HSIP066E MCDS OPEN ERROR - RC=***rc* **RS=***rs*

**Explanation:** The OPEN of the MCDS data definition (DD) was not successful.

In the message text:

*rc* VSAM OPEN hexadecimal return code.

*rs* VSAM OPEN hexadecimal reason code.

**System action:** Issues message HSIP065S and terminates with a condition code of 12.

**Operator response:** Either ensure that the Inquisitor has read access to DFHSM's MCDS, or modify the Inquisitor request(s) so that the MCDS is not required. MCDS access is required if either or both of the REMIGRATE and NOML2 keywords are specified.

**System programmer response:** None.

**Module:** HSIPINQ

---

**HSIP067E MCDS READ RC=***rc* **RS=***rs* **FOR** *dsn*

**Explanation:** The MCDS record of a data set cataloged on volume MIGRAT could not be read. Either the record is missing or there was an I/O error.

In the message text:

*rc* VSAM GET hexadecimal return code.

*rs* VSAM GET hexadecimal reason code.

*dsn*
name of data set cataloged on volume MIGRAT.

**System action:** Processing of this data set is terminated.

**Operator response:** If the data set is not really migrated then correct the catalog entry. If the MCDS is corrupt then begin recovery procedures.

**System programmer response:** None.

**Module:** HSIPINQ

---

**HSIP068W CATALOG RC=***rc* **RS=***rs*,*modid cat*

**Explanation:** The Catalog Search Interface returned an entry which is flagged as being in error by Catalog Management.

In the message text:

*rc* Catalog Management decimal return code.

*rs* Catalog Management decimal reason code.

*modid*
Catalog Management module identifier.

*cat*
name of catalog entry in error.

**System action:** Processing of this data set is terminated.

**Operator response:** Correct the catalog entry. Refer to the System Messages manual for message IDC3009I to find out the meaning of the Catalog Management error codes.

**System programmer response:** None.

**Module:** HSIPINQ

---

**HSIP069U PROGRAM IS NOT APF AUTHORIZED**

**Explanation:** The Inquisitor has determined that it is not running in an APF authorized environment, and PARM=NOAPF was not specified.

**System action:** Terminates with a condition code of 20.

**Operator response:** Ensure that the HSIPINQ program is run in an APF authorized environment, or specify PARM=NOAPF in the JCL.

**System programmer response:** None.

**Module:** HSIPINQ

---

**HSIP070E BAD BLKSIZE AFTER OPEN FOR** *dsn*

**Explanation:** A BPAM DCB was opened for the named PDS, but despite the VTOC entry indicating a suitable blocksize, the blocksize in the DCB after the OPEN was not positive.

In the message text:

*dsn*
name of the data set being processed.

**System action:** Processing of member contents for this data set is terminated to avoid an S002-30 abend.

**Operator response:** The PDS is probably corrupt and should be deleted. Recreate it from a backup if appropriate.

**System programmer response:** None.

**Module:** HSIPINQ

---

**HSIP071W IGNORING INVALID DSNAME IN** *dsn*

**Explanation:** The Catalog Search Interface (CSI) returned a data set name with invalid characters. Although VTOC entries can contain keys that are not valid data set names, such entries cannot be cataloged. Therefore the entry returned from the CSI does not represent an actual data set.

In the message text:

*dsn*
name of the catalog being processed.

**System action:** The returned catalog entry is discarded.

**Operator response:** Ensure that the named catalog is not corrupt and contains no invalid entries.

**System programmer response:** None.

**Module:** HSIPINQ

---

**HSIP072I BYPASS PROCESSING DATA SET** *dsn*

**Explanation:** The name of the data set indicated that it does not contain programs which would normally be executed, and therefore the Inquisitor skipped processing it. This message is only issued when a program parameter of "DSNMSG" or "ALLMSG" is specified.

In the message text:

*dsn*
name of the data set being bypassed.

**System action:** The data set is not opened, and no data from it is collected.

**Operator response:** None required, but if the data set must be processed then specify its name in an inclusion mask without any generic masking characters, either by adding this mask to the existing request, or by adding

an additional request to the same Inquisitor run.

**System programmer response:** None.

**Module:** HSIPINQ

---

**HSIP073I NO DATA WAS EXTRACTED FROM** *dsn*

**Explanation:** The data set contained no members eligible for selection. This message is only issued when a program parameter of "DSNMSG" or "ALLMSG" is specified.

In the message text:

*dsn*
name of the processed data set.

**System action:** The data set was opened, but no data from it is collected.

**Operator response:** None required.

**System programmer response:** None.

**Module:** HSIPINQ

---

**HSIP074S ABRIN OR ABRPRINT FILES NOT ALLOCATED**

**Explanation:** A request had ABRMIG and/or ABRARC specified but at least one of the required ABRIN and ABRPRINT files was not defined in the JCL.

**System action:** Terminates with a condition code of 12.

**Operator response:** Ensure the required files are pre-allocated for the Inquisitor.

**System programmer response:** None.

**Module:** HSIPINQ

---

**HSIP075W FDRABR ABEND** *abend* **CHECKING** *dsn*

**Explanation:** An abend occurred during ABR processing while checking a data set which may have been archived.

In the message text:

*abend*
hexadecimal system abend code.

*dsn*
name of the data set being processed.

**System action:** Processing of this data set is terminated.

**Operator response:** Ensure the catalog entry for the data set is correct.

**System programmer response:** None.

**Module:** HSIPINQ

---

**HSIP076E BAD LOAD** *abend-rs*: *mbr dsn*

**Explanation:** The Inquisitor attempted to load a product tag data module from the named data set, but LOAD issued the displayed abend code.

In the message text:

*abend*
abend code returned by LOAD.

*rs* abend reason code returned by LOAD.

*mbr*
name of the member containing the tag data.

*dsn*
name of the data set containing the tag data module.

**System action:** Processing continues with the next member in the data set.

**Operator response:** Verify that the named data set contains no unusable modules. If necessary, delete any modules that are of no further use.

**System programmer response:** None.

**Module:** HSIPINQ

---

**HSIP077W ISITMGD RC=***rc* **RS=***rs* **FOR** *dsn*

**Explanation:** The Inquisitor executed an ISITMGD macro for the named data set, but ISITMGD issued a non-zero return code.

In the message text:

*rc* decimal return code issued by ISITMGD.

*rs* hexadecimal reason code issued by ISITMGD.

*dsn*
name of the data set being processed.

**System action:** Processing continues with the next data set.

**Operator response:** Consult the applicable DFSMS Macro Instructions for Data Sets manual to determine the meaning of the ISITMGD return and reason codes. Ensure that the named data set is a valid and accessible partitioned data set. If necessary, gather appropriate diagnostic materials and contact IBM support.

**System programmer response:** None.

**Module:** HSIPINQ

---

**HSIP078W DESERV RC=***rc* **RS=***rs* **FOR** *dsn*

**Explanation:** The Inquisitor executed a DESERV FUNC=GET_ALL macro for the named data set, but DESERV issued a non-zero return code.

In the message text:

*rc* decimal return code issued by DESERV.

*rs* decimal reason code issued by DESERV.

*dsn*
name of the data set being processed.

**System action:** Processing continues with the next data set.

**Operator response:** Consult the applicable DFSMS Macro Instructions for Data Sets manual to determine the meaning of the DESERV return and reason codes. Ensure that the named data set is a valid and accessible partitioned data set. If necessary, gather appropriate diagnostic materials and contact IBM support.

**System programmer response:** None.

**Module:** HSIPINQ

---

**HSIP080I DYNALLOC FAILURE: DSN=***dsn*

**Explanation:** A data set could not be dynamically allocated.

In the message text:

*dsn*
name of the data set being processed.

**System action:** Depends upon other messages associated with this message.

**Operator response:** None required.

**System programmer response:** None.

**Module:** HSIPINQ

---

**HSIP081S ALL POSSIBLE DEVICE NUMBERS ARE EXCLUDED**

**Explanation:** An exclusion mask has been specified which excludes all possible device numbers included by a selection mask. Both masks are shown after this message.

**System action:** Terminates with a condition code of 12.

**Operator response:** Modify or remove the conflicting selection criteria.

**System programmer response:** None.

**Module:** HSIPINQ

---

**HSIP082I THE "XDEVICE" MASK IS NOT A SUBSET OF ANY "DEVICE" MASK**

**Explanation:** The mask specified in the XDEVICE operand excludes possible values not included in the DEVICE operand. This message is issued to highlight possible inconsistencies in a request.

**System action:** Processing continues.

**Operator response:** Specify the XDEVICE operand as a further qualification of the DEVICE operand to avoid this message.

**System programmer response:** None.

**Module:** HSIPINQ

---

**HSIP083E RENAME FAILED FOR DATA SET** *dsn*

**Explanation:** The rename operation to add one or more extra low-level qualifiers to a data set name as specified by the LLQ program parameter setting did not succeed. The named data set is allocated to either the HSIPZIP or HSIPOUT file. This message is preceded by either an associated explanatory message, or by messages from IDCAMS detailing the results of the rename attempt.

In the message text:

*dsn*
name of the HSIPZIP or HSIPOUT data set.

**System action:** The output data set retains its original name.

**Operator response:** Ensure that the specified LLQ string length does not exceed 44 bytes, that any symbols used are valid for this system, and that resultant data set names are not longer than 44 bytes. Examine associated messages to determine the reason for the rename failure.

**System programmer response:** None.

**Module:** HSIPINQ

---

**HSIP084I ABEND** *abend* **OPENING DSN** *dsn*

**Explanation:** An abnormal end occurred while opening a data set.

In the message text:

*abend*
hexadecimal system abend and reason

*dsn*
name of the data set being processed.

**System action:** Processing of this data set is terminated.

**Operator response:** None required, but you may wish to correct the cause of the abend.

**System programmer response:** None.

**Module:** HSIPINQ

---

**HSIP085I ABEND** *abend* **CLOSING DSN** *dsn*

**Explanation:** An abnormal end occurred while closing a data set.

In the message text:

*abend*
hexadecimal system abend and reason

*dsn*
name of the data set being processed.

**System action:** Processing of this data set is terminated.

**Operator response:** None required, but you may wish to correct the cause of the abend.

**System programmer response:** None.

**Module:** HSIPINQ

---

**HSIP086S NO PROGRAMS OR TAG DATA FOUND - NO DATA FOR IMPORT WAS PRODUCED**

**Explanation:** All scanning operations failed to find any executable programs or program tag data, so no data suitable for subsequent processing was created.

**System action:** Terminates with a condition code of 12.

**Operator response:** Correct any selection criteria errors and rerun the job.

**System programmer response:** None.

**Module:** HSIPINQ

---

**HSIP087I SKIP INACCESSIBLE DATA SET** *dsn*

**Explanation:** The named data set was encountered on an SMS-managed volume, but no matching catalog entry could be located, which means that the data set cannot be successfully allocated by any job.

In the message text:

*dsn*
name of the data set being skipped.

**System action:** The data set is bypassed and processing continues.

**Operator response:** None required, but you may wish to either catalog the data set to make it accessible, or delete it to reclaim the disk space.

**System programmer response:** None.

**Module:** HSIPINQ

---

**HSIP088I DEVICES=***vol* **CU-GROUPS=***ds* **USED-CHPIDS=***dsbad*

**Explanation:** Processing of a SCANDEV request has been completed. Counts of online I/O devices, device groups and used channel paths are shown. A group of devices with the same device type, control unit, and CHPID connectivity generates one CU record. A CP record is generated for each online CHPID connected to at least one online I/O device.

In the message text:

*vol*
count of online I/O devices discovered.

*ds* count of CU records written.

*dsbad*
count of CP records written.

**System action:** Processing continues.

**Operator response:** None.

**System programmer response:** None.

**Module:** HSIPINQ

---

**HSIP089E INSUFFICIENT AUTHORIZATION TO PROCESS "SCANDEV" REQUEST**

**Explanation:** To process a SCANDEV request either the Inquisitor must be APF authorized or the user must have UPDATE access to the IOSCDR entity in the FACILITY security class.

**System action:** Processing continues with the next request.

**Operator response:** Get the appropriate authorization, or omit the SCANDEV request.

**System programmer response:** None.

**Module:** HSIPINQ

---

**HSIP090I EXCESSIVE DESERV DELAY FOR** *dsn*

**Explanation:** This message is issued when a DESERV macro has not returned control to the Inquisitor after 15 minutes. This might indicate that the data set being processed is corrupt and unusable.

In the message text:

*dsn*
name of the data set being scanned.

**System action:** Processing continues, possibly without making any further progress.

**Operator response:** If the scan does not progress, and the delay is not caused by contention with other work, cancel the job. Either delete the data set, or add an exclusion to the Inquisitor control statement before rerunning the job.

**System programmer response:** None.

**Module:** HSIPINQ

---

**HSIP097E CATALOG SEARCH INTERFACE ERROR RC=***csirc*

**Explanation:** A request with the CATALOG keyword was specified, and the Catalog Search Interface encountered an error.

In the message text:

*csirc*
return code from the Catalog Search Interface.

**System action:** Processing catalog entries for the request is terminated.

**Operator response:** Correct any related catalog errors.

**System programmer response:** None.

**Module:** HSIPINQ

---

**HSIP098E CATALOG SEARCH INTERFACE ERROR RC=*csirc* CATALOG RC=*rc* CATALOG RS=*rs***

**Explanation:** A request with the CATALOG keyword was specified, and the Catalog Search Interface encountered an error.

In the message text:

*csirc*
return code from the Catalog Search Interface.

*rc* return code from Catalog Management.

*rs* reason code from Catalog Management.

**System action:** Processing catalog entries for the request is terminated.

**Operator response:** Correct any related catalog errors.

**System programmer response:** None.

**Module:** HSIPINQ

---

**HSIP099E CATALOG SEARCH INTERFACE ERROR RC=*csirc* CATALOG RC=*rc* CATALOG RS=*rs* MODULE=*modid***

**Explanation:** A request with the CATALOG keyword was specified, and the Catalog Search Interface encountered an error.

In the message text:

*csirc*
return code from the Catalog Search Interface.

*rc* return code from Catalog Management.

*rs* reason code from Catalog Management.

*modid*
module identifier.

**System action:** Processing catalog entries for the request is terminated.

**Operator response:** Correct any related catalog errors.

**System programmer response:** None.

**Module:** HSIPINQ

---

**HSIP999U MODULE HSIPMSG FAILED - MSGID=*msgid* RC=*rc* RS=*rs***

**Explanation:** HSIMSG was called to produce a message text, but the call failed.

In the message text:

*msgid*
identifier of the failing message.

*rc* HSIMSG return code.

*rs* HSIMSG reason code.

**System action:** Terminates with a condition code of 20.

**Operator response:** Inform the system programmer.

**System programmer response:** Contact IBM Support.

**Module:** HSIPINQ

# HSIT - Product tagging messages

### Return codes

*Table 97. Return codes and their meaning*

| Return code | Description |
|---|---|
| 0 | No errors encountered. All requests processed successfully. |
| 4 | Warning issued. Processing continues. Input/Output error in one or more program libraries. |
| 8 | Error - Incomplete data. Processing continues. OPEN or other system service error. |
| 12 | Syntax error. Processing terminates. Utility failure or syntax error. |
| 16 | Unrecoverable error. No requests processed. SYSIN file cannot be used. |
| 20 | Disastrous error. No requests processed. SYSPRINT file cannot be used. |

### Message suffix codes

*Table 98. Message suffix codes and associated condition codes*

| Suffix | Meaning | Raises minimum condition code to: |
|---|---|---|
| I | Information message | 0 |
| W | Warning message | 4 |
| E | Error message | 8 |
| S | Severe error message | 12 |
| U | Unrecoverable error message | 16 |

**Message texts and explanations**

All numeric completion codes of system services reported in these messages are in hexadecimal unless otherwise stated.

**HSIT001U HSITAGP COULD NOT OPEN THE INPUT FILE** *file*

**Explanation:** A required file could not be opened successfully.

In the message text:

*file*
name of file.

**System action:** Processing terminates with condition code 16.

**Operator response:** Correct the file definition and rerun the job.

**System programmer response:** None.

**Module:** HSITAGP

**HSIT002S UNRECOGNIZED STATEMENT TYPE:** *stattyp*

**Explanation:** Input text was encountered which does not match any known statement type.

In the message text:

*stattyp*
encountered input data.

**System action:** Processing terminates with condition code 12.

**Operator response:** Correct the input and rerun the job.

**System programmer response:** None.

**Module:** HSITAGP

**HSIT003S DUPLICATE VALUE SUPPLIED FOR** *stattyp*

**Explanation:** More than one occurrence of the named statement type was encountered, but only one value can be accepted.

In the message text:

*stattyp*
name of the statement verb.

**System action:** Processing terminates with condition code 12.

**Operator response:** Remove the redundant statement and rerun the job.

**System programmer response:** None.

**Module:** HSITAGP

**HSIT004S VALUE MISSING IN** *stattyp* **STATEMENT**

**Explanation:** An input statement of the type indicated was encountered, but no non-blanks followed the statement type name.

In the message text:

*stattyp*
name of the statement verb.

**System action:** Processing terminates with condition code 12.

**Operator response:** Supply an appropriate value after the statement type name.

**System programmer response:** None.

**Module:** HSITAGP

**HSIT005S VALUE SPECIFIED FOR LICENSED WAS NEITHER "YES" NOR "NO"**

**Explanation:** A LICENSED statement was processed which had a value specified other than one of the valid values.

**System action:** Processing terminates with condition code 12.

**Operator response:** Correct the value and rerun the job.

**System programmer response:** None.

**Module:** HSITAGP

---

**HSIT006S THE *parm* PARAMETER HAD NO SUBPARAMETER VALUE SPECIFIED**

**Explanation:** A statement parameter or operand was specified, but the required subparameter, or value of the parameter, was not specified. One cause for this condition is the omission of a parenthesis.

In the message text:

*parm*
name of the parameter or operand being processed when the error is detected.

**System action:** Processing terminates with condition code 12.

**Operator response:** Correct the input and rerun the job.

**System programmer response:** None.

**Module:** HSITAGP

---

**HSIT007I A CLOSING PARENTHESIS ASSUMED FOR *parm***

**Explanation:** End-of-file was raised when processing input statements before an expected close parenthesis was encountered.

In the message text:

*parm*
name of the parameter or operand being processed when the error is detected.

**System action:** Processing continues as if the expected close parenthesis had been specified.

**Operator response:** Check that the resulting processing is as expected. Correct the input file for future use, and rerun the job if the desired processing was not performed.

**System programmer response:** None.

**Module:** HSITAGP

---

**HSIT008S UNEXPECTED OPEN PARENTHESIS ENCOUNTERED AFTER *parm***

**Explanation:** An open parenthesis was encountered when one was not expected. If this occurred while a parameter or operand was being processed, then it is named in the message.

In the message text:

*parm*
name of the parameter or operand being processed when the error is detected.

**System action:** Processing terminates with condition code 12.

**Operator response:** Correct the input file and rerun the job.

**System programmer response:** None.

**Module:** HSITAGP

---

**HSIT009S UNEXPECTED CLOSE PARENTHESIS ENCOUNTERED AFTER *parm***

**Explanation:** A close parenthesis was encountered when one was not expected. If this occurred while a parameter or operand was being processed, then it is named in the message.

In the message text:

*parm*
name of the parameter or operand being processed when the error is detected.

**System action:** Processing terminates with condition code 12.

**Operator response:** Correct the input file and rerun the job.

**System programmer response:** None.

**Module:** HSITAGP

---

**HSIT010S *parm* IS AN UNKNOWN SELECT PARAMETER**

**Explanation:** Input data was encountered which is not a recognized parameter, or operand, of the SELECT statement.

In the message text:

*parm*
the encountered input data.

**System action:** Processing terminates with condition code 12.

**Operator response:** Correct the input file and rerun the job.

**System programmer response:** None.

**Module:** HSITAGP

---

**HSIT011S MEMBER NAME *parm* HAS EMBEDDED BLANK(S)**

**Explanation:** The value specified on the TAGMEM statement was not a valid partitioned data set member name, a blank was found within the eight character member name.

In the message text:

*parm*
the input value specified on the TAGMEM statement.

**System action:** Processing terminates with condition code 12.

**Operator response:** Correct the input file and rerun the job.

**System programmer response:** None.

**Module:** HSITAGP

---

**HSIT012S MISSING OPEN PARENTHESIS AFTER** *parm*

**Explanation:** Whilst parsing the SELECT statement looking for a subparameter, or value, in parentheses specified for the parameter or operand named in the message, text was encountered which was not enclosed in parentheses.

In the message text:

*parm*
name of the parameter or operand being processed when the error is detected.

**System action:** Processing terminates with condition code 12.

**Operator response:** Correct the input and rerun the job.

**System programmer response:** None.

**Module:** HSITAGP

---

**HSIT013S VALUE** *data* **TOO LONG FOR PARAMETER** *parm*

**Explanation:** The length of a subparameter or value was found to exceed the maximum length allowed. The maximum length allowed depends on the specific parameter or operand being processed. For example, a data set name mask exceeding 44 characters in length causes this condition, as will a volume mask exceeding six characters in length.

In the message text:

*data*
encountered input data.

*parm*
name of the parameter or operand being processed when the error is detected.

**System action:** Processing terminates with condition code 12.

**Operator response:** Correct the input and rerun the job.

**System programmer response:** None.

**Module:** HSITAGP

---

**HSIT014S END OF INPUT REACHED, EXPECTED CONTINUATION IS MISSING**

**Explanation:** End-of-file was raised on the input (SYSIN) file, but the SELECT statement currently being processed was expected to continue on the next record.

**System action:** Processing terminates with condition code 12.

**Operator response:** Either supply the missing input data, or remove the continuation character from the last input record. Rerun the job.

**System programmer response:** None.

**Module:** HSITAGP

---

**HSIT017S NO VALUE FOR** *stattyp* **WAS SPECIFIED**

**Explanation:** A value for a statement of the type named in the message is required, but was not found in the input file.

In the message text:

*stattyp*
the type of input statement required to specify the missing value.

**System action:** Processing terminates with condition code 12.

**Operator response:** Supply a statement of the named type which specifies a value.

**System programmer response:** None.

**Module:** HSITAGP

---

**HSIT019E DESERV FAILED - RC=***rc* **RS=***rs* **FOR DATA SET** *dsn*

**Explanation:** A DESERV FUNC=GET_ALL macro was issued to acquire the member list for a data set, but DESERV issued a non-zero return code.

In the message text:

*rc* the decimal return code issued by DESERV.

*rs* the hexadecimal reason code issued by DESERV.

*dsn*
the name of the data set being processed by DESERV.

**System action:** The named data set is not processed, and processing continues with the next relevant data set.

**Operator response:** Consult the applicable DFSMS Macro Instructions for Data Sets manual to determine the meaning of the DESERV return and reason codes. Ensure that the named data set is a valid and accessible program library. If necessary, gather appropriate diagnostic materials and contact IBM support.

**System programmer response:** None.

**Module:** HSITAGP

---

**HSIT020S DYNAMIC ALLOCATION FAILURE - BPXWDYN RC=*rc***

**Explanation:** BPXWDYN was called to dynamically allocate a required work file, but BPXWDYN issued a non-zero return code. As a result, processing cannot proceed.

In the message text:

*rc* the hexadecimal return code issued by BPXWDYN.

**System action:** Processing terminates with condition code 12.

**Operator response:** Consult the applicable Using REXX and z/OS UNIX System Services manual to determine the meaning of the return code. Examine the job log and messages to see any associated dynamic allocation error message.

**System programmer response:** None.

**Module:** HSITAGP

---

**HSIT022S RC=*rc* WAS RETURNED BY PROGRAM *pgm***

**Explanation:** Either the High Level Assembler (program ASMA90) or the Program Binder (program IEWL) was dynamically started to assist with creating the output data, but the named program issued a non-zero return code.

In the message text:

*rc* the decimal return code issued by the named program.

*pgm*
the name of the program that was started.

**System action:** Processing terminates with condition code 12.

**Operator response:** Examine all associated job output to determine if the problem is caused by a correctable environmental error. If so, make the correction and rerun the job. If not, gather all relevant diagnostic materials and contact IBM support.

**System programmer response:** None.

**Module:** HSITAGP

---

**HSIT023I PROCESSING TERMINATED DUE TO ENCOUNTERED ERROR CONDITION**

**Explanation:** Because of a previously reported error, the Product Tagging Utility is terminating unilaterally, without processing all of the specified program library data sets, and without generating all of the requested program product tagging data.

**System action:** Processing terminates.

**Operator response:** Investigate any previously reported error conditions.

**System programmer response:** None.

**Module:** HSITAGP

---

**HSIT024E ISITMGD FAILED - RC=*rc* RS=*rs* FOR FILE *file* AND DATA SET *dsn***

**Explanation:** An ISITMGD macro was issued against a program library, but ISITMGD issued a non-zero return code.

In the message text:

*rc* the decimal return code issued by ISITMGD.

*rs* the decimal reason code issued by ISITMGD.

*file*
the name of the file being processed by ISITMGD.

*dsn*
the name of the data set being processed by ISITMGD.

**System action:** The named data set is not processed, and processing continues with the next relevant data set.

**Operator response:** Consult the applicable DFSMS Macro Instructions for Data Sets manual to determine the meaning of the ISITMGD return and reason codes. Ensure that the named data set is a valid and accessible partitioned data set. If necessary, gather the appropriate diagnostic materials and contact IBM support.

**System programmer response:** None.

**Module:** HSITAGP

---

**HSIT025I *pgmcnt* PROGRAMS FOUND TO TAG FROM DATA SET *dsn***

**Explanation:** Input processing of the named data set has completed, resulting in data from the reported number of programs being accumulated for subsequent output.

In the message text:

*pgmcnt*
the number of programs processed.

*dsn*
the data set name containing the processed programs.

**System action:** Processing continues.

**Operator response:** None required.

**System programmer response:** None.

**Module:** HSITAGP

**HSIT026I PROCESSING COMPLETE - RC=*rc* AND *pgmcnt* PROGRAMS TAGGED IN TOTAL**

**Explanation:** The Product Tagging Utility program HSITAGP has completed processing. This message reports the return code issued by HSITAGP, and the number of programs from which data has been collected during this run.

In the message text:

*rc* the return code issued by the HSITAGP upon termination.

*pgmcnt*
the number of programs processed in this run of HSITAGP.

**System action:** Processing is completed with the displayed return code.

**Operator response:** None required.

**System programmer response:** None.

**Module:** HSITAGP

**HSIT028W UNABLE TO ACQUIRE ANY PRODUCT MAINTENANCE LEVEL DATE**

**Explanation:** After having processed all of the relevant programs, HSITAGP was unable to acquire any date stamp for use as a maintenance level indicator.

**System action:** Blanks are placed in the maintenance level field and processing continues.

**Operator response:** None required.

**System programmer response:** None.

**Module:** HSITAGP

**HSIT029S *stattyp* STATEMENT VALUE LENGTH EXCEEDS THE ALLOWED MAXIMUM OF *max* BYTES**

**Explanation:** The value specified for the named statement type was found to be longer than the maximum allowed. The maximum byte count allowed for a value of this statement type is shown in the message.

In the message text:

*stattyp*
the type of input statement being processed.

*max*
number of bytes.

**System action:** Processing terminates.

**Operator response:** Correct the input and rerun the job.

**System programmer response:** None.

**Module:** HSITAGP

**HSIT030S INVALID TEXT CHARACTER X"*char*" FOUND IN *stattyp* STATEMENT**

**Explanation:** The displayed data byte was encountered when processing the value specified for the statement type indicated. The value specified on the statement is expected to be a string. Valid byte values for text data are in the range from X'40' to X'FE' inclusive. The control code encountered is either not valid input, or not valid input in this location. The only control codes that can be used in the input value are SO (X'0E') and SI(X'0F'), when they are used to encapsulate DBCS data.

In the message text:

*char*
the hexadecimal value of the invalid text code point.

*stattyp*
the type of input statement being processed.

**System action:** Processing terminates.

**Operator response:** Remove the undisplayable characters from the input value. If using DBCS, ensure that SO precedes DBCS text and SI terminates DBCS text, and that the DBCS text is an even number of valid text bytes.

**System programmer response:** None.

**Module:** HSITAGP

**HSIT999U HSIMSG/HSITMSG FAILURE - MSGID=*msgid* RC=*rc* RS=*rs***

**Explanation:** HSIMSG was called to produce a message text, but the call failed.

In the message text:

*msgid*
identifier of the failing message.

*rc* HSIMSG return code.

*rs* HSIMSG reason code.

**System action:** Terminates with a condition code of 20.

**Operator response:** Inform the system programmer.

**System programmer response:** Ensure Joblib/Steplib contains the library where the HSITMSG message module resides. If you cannot resolve this issue then contact IBM support.

**Module:** HSITAGP

# HSIX - Inquisitor for z/OS UNIX messages and codes

### Return codes

*Table 99. Return codes and their meaning*

| Return code | Description |
|---|---|
| 0 | No errors encountered. All requests processed successfully. |
| 4 | Input/Output error in one or more program libraries. |
| 8 | Error - Incomplete data. Processing continues. OPEN or other system service error. |
| 12 | Syntax error. |
| 16 | Unrecoverable error. No requests processed. SYSIN or HSIPZIP or HSIPOUT File cannot be used, or unsupported Operating System. |
| 20 | Disastrous error. No requests processed. SYSPRINT file cannot be used. |

### Message suffix codes

*Table 100. Message suffix codes and associated condition codes*

| Suffix | Meaning | Raises minimum condition code to: |
|---|---|---|
| I | Information message | 0 |
| W | Warning message | 4 |
| E | Error message | 8 |
| S | Severe error message | 12 |
| U | Unrecoverable error message | 16 |

### Message texts and explanations

All numeric completion codes of system services reported in these messages are in hexadecimal unless otherwise stated.

**HSIX002I THE SPECIFIED DIRECTORY NAME DOES NOT START WITH A SLASH**

**Explanation:** A record from file HSIXROOT was read and was found to start with a non-blank that is not a slash. It is reported in case processing errors result from the non-standard directory name.

**System action:** Processing continues.

**Operator response:** Correct the input if it is incorrect.

**System programmer response:** None.

**Module:** HSIXINQ

**HSIX003I PROGRAM PARAMETER "*parm*" DISCARDED**

**Explanation:** The program parameter contained some unrecognized data.

In the message text:

*parm*
parameter in error.

**System action:** The displayed part of the program parameter is ignored.

**Operator response:** Correct the program parameter.

**System programmer response:** None.

**Module:** HSIXINQ

**HSIX004I FUNCTION *func* COMPLETED WITH RC=*rc* AND REASON=*rs***

**Explanation:** The named z/OS UNIX system service issued a negative return value.

In the message text:

*func*
function name.

*rc* hexadecimal return code.

*rs* hexadecial reason code.

**System action:** Processing continues.

**Operator response:** Determine the meaning of the return and reason codes, and correct the problem if appropriate. Information relating to the failing UNIX function can be found in the UNIX System Services Assembler Callable Services manual. Information relating to the Return Code and Reason Code of the failing UNIX function can be found in the UNIX System Services Messages and Codes manual.

**System programmer response:** None.

**Module:** HSIXINQ

---

**HSIX006E RENAME FAILED FOR DATA SET** *dsn*

**Explanation:** The rename operation to add one or more extra low-level qualifiers to a data set name as specified by the LLQ program parameter setting did not succeed. The named data set is allocated to either the HSIXZIP or HSIXOUT file. If this message is not followed by an associated explanatory message then an IDCAMS report detailing the results of the rename attempt will have been written to SYSPRINT.

In the message text:

*dsn*
name of the HSIXZIP or HSIXOUT data set.

**System action:** The output data set retains its original name.

**Operator response:** Ensure that the specified LLQ string length does not exceed 44 bytes, that any symbols used are valid for this system, and that resultant data set names are not longer than 44 bytes. Examine associated messages to determine the reason for the rename failure.

**System programmer response:** None.

**Module:** HSIXINQ

---

**HSIX007E FUNCTION** *func* **FAILED, RC=***rc*, **REASON=***rs*, **FOR PATH** *pth*

**Explanation:** The named z/OS UNIX system service issued a negative return value.

In the message text:

*func*
function name.

*rc* hexadecimal return code.

*rs* hexadecimal reason code.

*pth*
path in error.

**System action:** Processing continues.

**Operator response:** Determine the meaning of the return and reason codes, and correct the problem if appropriate. Information relating to the failing UNIX function can be found in the UNIX System Services Assembler Callable Services manual. Information relating to the Return Code and Reason Code of the failing UNIX function can be found in the UNIX System Services Messages and Codes manual.

**System programmer response:** None.

**Module:** HSIXINQ

---

**HSIX008E FUNCTION** *func* **WAS DENIED ACCESS TO PATH** *pth*

**Explanation:** The named z/OS UNIX system service issued a return code of hexadecimal 6F which indicates that access was denied.

In the message text:

*func*
function name.

*pth*
path in error.

**System action:** Processing continues.

**Operator response:** Grant the user access to the parts of the UNIX file system to be scanned.

**System programmer response:** None.

**Module:** HSIXINQ

---

**HSIX009S NO EXECUTABLE SOFTWARE FOUND - NO DATA FOR IMPORT WAS PRODUCED**

**Explanation:** All scanning operations failed to find any programs or other executable software, so no data suitable for subsequent processing was created.

**System action:** Terminates with a condition code of 12.

**Operator response:** Correct any selection criteria errors and rerun the job.

**System programmer response:** None.

**Module:** HSIPINQ

---

**HSIX999U HSIMSG/HSIXMSG FAILURE - MSGID=***msgid* **RC=***rc* **RS=***rs*

**Explanation:** HSIMSG was called to produce a message text, but the call failed.

In the message text:

*msgid*
identifier of the failing message.

*rc* HSIMSG return code.

*rs* HSIMSG reason code.

**System action:** Terminates with a condition code of 20.

**Operator response:** Inform the system programmer.

**System programmer response:** Ensure Joblib/Steplib contains the library where the HSIXMSG message module resides. If you cannot resolve this issue then contact IBM support.

**Module:** HSIXINQ

# HSIZ - Usage Monitor messages

### Return codes

*Table 101. Return codes and their meaning*

| Return code | Description |
|---|---|
| 0 | Normal termination. |
| 16 | Initialization failed. |

### Message suffix codes

*Table 102. Message suffix codes and associated condition codes*

| Suffix | Meaning | Raises minimum condition code to: |
|---|---|---|
| I | Information message | 0 |
| W | Warning message | 4 |
| E | Error message | 8 |
| S | Severe error message | 12 |
| U | Unrecoverable error message | 16 |

### Message texts and explanations

All numeric completion codes of system services reported in these messages are in hexadecimal unless otherwise stated.

**HSIZ001I USAGE MONITOR INITIALIZING**

**Explanation:** The Usage Monitor has been started.

**System action:** Processing continues.

**Operator response:** None required.

**System programmer response:** None.

**Module:** HSIZMON

**HSIZ002I** *csid* **DETECTED UNSUPPORTED OPERATING SYSTEM**

**Explanation:** The Usage Monitor may not run on an unsupported operating system.

In the message text:

*csid*
current system identifier.

**System action:** Processing terminates.

**Operator response:** None required.

**System programmer response:** None.

**Module:** HSIZMON

**HSIZ003I** *csid* **USAGE MONITOR NOT APF AUTHORIZED**

**Explanation:** The Usage Monitor needs to be executed in an APF authorized environment.

In the message text:

*csid*
current system identifier.

**System action:** Processing terminates.

**Operator response:** See System Programmer to correct the error.

**System programmer response:** APF authorize the load libraries that the Usage monitor runs from.

**Module:** HSIZMON

**HSIZ005I** *csid* **USAGE MONITOR ALREADY ACTIVE**

**Explanation:** The Usage Monitor is already running.

Only one concurrent copy can run in an operating system image.

In the message text:

*csid*
current system identifier.

**System action:** Processing terminates. The established Usage Monitor task continues.

**Operator response:** None required.

**System programmer response:** None.

**Module:** HSIZMON

---

**HSIZ006I** *csid* **USAGE MONITOR QEDIT BUFFER SET FAILED**

**Explanation:** A QEDIT issued to set up MODIFY command processing has failed.

In the message text:

*csid*
current system identifier.

**System action:** Processing terminates.

**Operator response:** Notify the system programmer.

**System programmer response:** Gather appropriate diagnostic materials and contact IBM support.

**Module:** HSIZMON

---

**HSIZ007I** *csid* **USAGE MONITOR MODULE** *mod* **FAILED - RC=***rc*

**Explanation:** A Usage Monitor subroutine has failed.

In the message text:

*csid*
current system identifier.

*mod*
failing module name.

*rc* decimal return code.

**System action:** Processing terminates.

**Operator response:** Notify the system programmer.

**System programmer response:** Gather appropriate diagnostic materials and contact IBM support.

**Module:** HSIZMON

---

**HSIZ008I** *csid* **USAGE MONITOR INITIALIZED - ASID** *asid* **SET IN AVT** *avt*

**Explanation:** An Anchor Vector Table (AVT) has been acquired or reacquired, and has been updated for the current server address space, which has completed initialization.

In the message text:

*csid*
current system identifier.

*asid*
ASID number.

*avt*
AVT Address.

**System action:** Processing continues.

**Operator response:** None required.

**System programmer response:** None.

**Module:** HSIZMON

---

**HSIZ009I** **DATA WRITTEN TO DSN=***dsn*

**Explanation:** Usage Monitor data has been written to the named data set.

In the message text:

*dsn*
data set name of the created output.

**System action:** Processing continues.

**Operator response:** Transfer the named data set to the system where the database resides so it can be processed.

**System programmer response:** None.

**Module:** HSIZMON

---

**HSIZ010E** *csid* **USAGE MONITOR - WRITER TASK ENDED - RC=***rc*

**Explanation:** A writer task has ended with a non-zero return code.

In the message text:

*csid*
current system identifier.

*rc* return code of writer task.

**System action:** Processing continues.

**Operator response:** Notify the system programmer.

**System programmer response:** Gather appropriate diagnostic materials and contact IBM support.

**Module:** HSIZMON

---

**HSIZ011E** *csid* **USAGE MONITOR - WRITER TASK ABENDED - S***abend*

**Explanation:** A writer task has ended abnormally.

In the message text:

*csid*
current system identifier.

*abend*
abend code from writer task.

**System action:** Processing continues.

**Operator response:** Notify the system programmer.

**System programmer response:** Local reasons for system abends should be investigated. If necessary, gather appropriate diagnostic materials and contact IBM support.

**Module:** HSIZMON

---

**HSIZ012I** ****DATA LOSS** UNUSABLE DSN=***dsn*

**Explanation:** It is likely that Usage Monitor data has been lost because of unexpected behaviour by a writer task. Any compressed output data that has been written will probably be unusable.

In the message text:

*dsn*
data set name of the created output file.

**System action:** Processing continues.

**Operator response:** Examine any preceding messages to determine the likely cause of the writer task error. If the output data set is complete it can be used, otherwise if the data is compressed it is unusable. If the data set is empty then this fact can be noted and the data set can be deleted. Unless retaining an unusable data set for diagnosis reasons it can be deleted.

**System programmer response:** Investigate any writer task abends.

**Module:** HSIZMON

---

**HSIZ013I** *csid* **USAGE MONITOR - UNRECOGNISED PROGRAM PARAMETER IGNORED**

**Explanation:** An unrecognised program parameter was specified.

In the message text:

*csid*
current system identifier.

**System action:** Processing continues.

**Operator response:** Remove or correct the program parameter.

**System programmer response:** None.

**Module:** HSIZMON

---

**HSIZ014I** *csid* **USAGE MONITOR - COULD NOT OPEN FILE HSIZIN**

**Explanation:** The HSIZIN file could not be opened by the Usage Monitor.

In the message text:

*csid*
current system identifier.

**System action:** Processing terminates.

**Operator response:** Supply or correct the HSIZIN DD statement in the JCL.

**System programmer response:** None.

**Module:** HSIZMON

---

**HSIZ015I** *csid* **USAGE MONITOR - COULD NOT OPEN FILE HSIZMSG**

**Explanation:** The HSIZMSG file could not be opened by the Usage Monitor.

In the message text:

*csid*
current system identifier.

**System action:** Processing terminates.

**Operator response:** Supply or correct the HSIZMSG DD statement in the JCL.

**System programmer response:** None.

**Module:** HSIZMON

---

**HSIZ016I** *csid* **USAGE MONITOR TERMINATING - INVALID OR MISSING HSIZIN DATA**

**Explanation:** At least one HSIZIN input statement was invalid, or input required to be present in the HSIZIN file was missing.

In the message text:

*csid*
current system identifier.

**System action:** Processing terminates.

**Operator response:** Examine the HSIZMSG output report. Correct any invalid statements. Ensure a valid data set name prefix was specified.

**System programmer response:** None.

**Module:** HSIZMON

---

**HSIZ017I** *csid* **USAGE MONITOR TERMINATING - NOW WRITING CAPTURED DATA**

**Explanation:** A STOP command has been encountered. The current repository contents are written before the Usage Monitor terminates.

In the message text:

*csid*
current system identifier.

**System action:** The Usage Monitor starts a writer task and waits for its completion before terminating.

**Operator response:** None required.

**System programmer response:** None.

**Module:** HSIZMON

---

**HSIZ018I** *csid* **USAGE MONITOR HAS NOW TERMINATED**

**Explanation:** The Usage Monitor has now freed resources and is about to terminate.

In the message text:

*csid*
current system identifier.

**System action:** Processing terminates.

**Operator response:** None required.

**System programmer response:** None.

**Module:** HSIZMON

---

**HSIZ019I** *csid* **USAGE MONITOR REPOSITORY FULL - NOW SWITCHING**

**Explanation:** The current Usage Monitor data collection repository is full.

In the message text:

*csid*
current system identifier.

**System action:** A new repository is created and used for subsequent data collection. A writer task is initiated for the full repository.

**Operator response:** None required.

**System programmer response:** None.

**Module:** HSIZMON

---

**HSIZ020I** *csid* **THE SPECIFIED NUMBER WAS TOO SMALL**

**Explanation:** The numeric value of a command subparameter was too small to be valid in the command context.

In the message text:

*csid*
current system identifier.

**System action:** The command is discarded.

**Operator response:** Correct the numeric value and reissue the command.

**System programmer response:** None.

**Module:** HSIZMON

---

**HSIZ021I** *csid* **THE SPECIFIED NUMBER WAS TOO LARGE**

**Explanation:** The numeric value of a command subparameter was too large to be valid in the command context.

In the message text:

*csid*
current system identifier.

**System action:** The command is discarded.

**Operator response:** Correct the numeric value and reissue the command.

**System programmer response:** None.

**Module:** HSIZMON

---

**HSIZ022I** *csid* **PASSIVE MODE SET FROM PROGRAM PARAMETER**

**Explanation:** PASSIVE was specified in the program parameter.

In the message text:

*csid*
current system identifier.

**System action:** The Usage Monitor starts in passive mode unless overridden by input from the HSIZIN file.

**Operator response:** Set the Usage Monitor into collection mode to start data collection.

**System programmer response:** None.

**Module:** HSIZMON

---

**HSIZ023I** *csid* **PROGRAM NAME MASK** *mask* **NOT ADDED - ALREADY IN TABLE**

**Explanation:** A command to add a program name mask to a program mask table was issued, but the mask was already present in the table.

In the message text:

*csid*
current system identifier.

*mask*
program mask specified in command.

**System action:** Processing continues.

**Operator response:** None required.

**System programmer response:** None.

**Module:** HSIZMON

---

**HSIZ024I** *csid* **PROGRAM NAME MASK** *mask* **ADDED TO TABLE**

**Explanation:** A command to add a program name mask to a program mask table was issued, and the mask was added successfully.

In the message text:

*csid*
current system identifier.

*mask*
program mask specified in command.

**System action:** Processing continues.

**Operator response:** None required.

**System programmer response:** None.

**Module:** HSIZMON

**HSIZ025I** *csid* **PROGRAM NAME MASK** *mask* **NOT DELETED - NOT FOUND IN TABLE**

**Explanation:** A command to delete a program name mask from a program mask table was issued, but the mask was not present in the table.

In the message text:

*csid*
current system identifier.

*mask*
program mask specified in command.

**System action:** Processing continues.

**Operator response:** None required.

**System programmer response:** None.

**Module:** HSIZMON

**HSIZ026I** *csid* **PROGRAM NAME MASK** *mask* **DELETED FROM TABLE**

**Explanation:** A command to delete a program name mask to a program mask table was issued, and the mask was deleted successfully.

In the message text:

*csid*
current system identifier.

*mask*
program mask specified in command.

**System action:** Processing continues.

**Operator response:** None required.

**System programmer response:** None.

**Module:** HSIZMON

**HSIZ027I** **ECSA APPEARS TO BE EXHAUSTED - INCREASE SIZE FOR NEXT IPL**

**Explanation:** The Usage Monitor has attempted to acquire storage from ECSA, but was given CSA storage by the system. This indicates that there is insufficient ECSA for the current workloads, and that it should be increased for the next IPL.

**System action:** Processing continues.

**Operator response:** Notify the system programmer.

**System programmer response:** Add around 50 to 100 megabytes to the ECSA size in the system IPL parameters. Check the capacity of the COMMON page data set.

**Module:** HSIZMON

**HSIZ028I** **ECSA AND CSA APPEAR TO BE EXHAUSTED - INCREASE ECSA NEXT IPL**

**Explanation:** The Usage Monitor has attempted to acquire some common storage, but the requested amount was unavailable. This indicates that there is insufficient ECSA for the current workloads, and that it should be increased for the next IPL.

**System action:** Processing continues.

**Operator response:** Notify the system programmer.

**System programmer response:** Add around 50 to 100 megabytes to the ECSA size in the system IPL parameters. Close some applications using CSA. If necessary, commence orderly shutdown and reIPL before the system crashes. Check the capacity of the COMMON page data set.

**Module:** HSIZMON

**HSIZ029I** *csid* **THERE IS CURRENTLY NO EXCLUDE TABLE**

**Explanation:** A request was made to change or display the program name mask exclude table, but there is currently no exclude table.

In the message text:

*csid*
current system identifier.

**System action:** Processing continues.

**Operator response:** None required. The EXC command may be used to create a table.

**System programmer response:** None.

**Module:** HSIZMON

**HSIZ030I** *csid* **USAGE MONITOR - NO DATA COLLECTED SO SKIPPING WRITE**

**Explanation:** Before a writer task was initiated to output the contents of a Usage Monitor repository, it was found that the repository contained no data, and that therefore data output processing could be omitted.

In the message text:

*csid*
current system identifier.

**System action:** Processing continues.

**Operator response:** None required.

**System programmer response:** None.

**Module:** HSIZMON

**HSIZ031I** *csid* **INITIATING REPOSITORY SWITCH**

**Explanation:** A switch (SWI) command was issued and the requested action is being initiated.

In the message text:

*csid*
current system identifier.

**System action:** Processing continues.

**Operator response:** None required.

**System programmer response:** None.

**Module:** HSIZMON

**HSIZ032I** *csid cmd* **COMMAND UNKNOWN**

**Explanation:** A command was issued but was not recognised.

In the message text:

*csid*
current system identifier.

*cmd*
name of the issued command.

**System action:** The command is ignored. Processing continues.

**Operator response:** If necessary, correct and reissue the command.

**System programmer response:** None.

**Module:** HSIZMON

**HSIZ033I** *csid cmd* **COMMAND PROCESSED**

**Explanation:** A command was issued and has been processed successfully.

In the message text:

*csid*
current system identifier.

*cmd*
name of the issued command.

**System action:** Processing continues.

**Operator response:** None required.

**System programmer response:** None.

**Module:** HSIZMON

**HSIZ034I** *csid cmd* **COMMAND HAS INVALID OPERAND**

**Explanation:** A command was issued but an invalid operand was encountered.

In the message text:

*csid*
current system identifier.

*cmd*
name of the issued command.

**System action:** The command is ignored. Processing continues.

**Operator response:** If necessary, correct and reissue the command.

**System programmer response:** None.

**Module:** HSIZMON

**HSIZ035I** *csid cmd* **COMMAND FAILED**

**Explanation:** A command was issued but insufficient resources were available to execute it successfully.

In the message text:

*csid*
current system identifier.

*cmd*
name of the issued command.

**System action:** The command is ignored. Processing continues.

**Operator response:** Try again after more resources become available.

**System programmer response:** None.

**Module:** HSIZMON

**HSIZ036I** *csid cmd* **COMMAND CAUSED NO CHANGE**

**Explanation:** A command was issued but the state to be set by the command was found to already exist.

In the message text:

*csid*
current system identifier.

*cmd*
name of the issued command.

**System action:** Processing continues.

**Operator response:** None required.

**System programmer response:** None.

**Module:** HSIZMON

---

**HSIZ037I** *csid cmd* **COMMAND REJECTED**

**Explanation:** A recognised command was issued at a time when the Usage Monitor is unable to process the command.

In the message text:

*csid*
current system identifier.

*cmd*
name of the issued command.

**System action:** The command is ignored. Processing continues.

**Operator response:** Try again after the Usage Monitor has freed the resources.

**System programmer response:** None.

**Module:** HSIZMON

---

**HSIZ038I** *csid* **CURRENT USAGE MONITOR PROGRAM EXCLUDE LIST:**

**Explanation:** A D-X command was issued to display the program name exclude table contents. The active entries are shown after this message.

In the message text:

*csid*
current system identifier.

**System action:** The data is displayed and processing continues.

**Operator response:** None required.

**System programmer response:** None.

**Module:** HSIZMON

---

**HSIZ039I** *csid* **REPOSITORY SWITCH HAS BEEN QUEUED**

**Explanation:** A repository switch was triggered by a SWI or STOP command, or by the current repository becoming full, but a writer task is already active. This message is followed by message HSIZ040I which shows the creation timestamp of the active writer task.

In the message text:

*csid*
current system identifier.

**System action:** Data collection is suspended. Wait for the current writer task to complete whereupon a new writer task is created, and a new repository is created, and data collection is resumed.

**Operator response:** Check that there are sufficient resources to dispatch the Usage Monitor address space. Check that there are no serialization problems with system components such as device allocation which could be inhibiting writer task processing.

**System programmer response:** None.

**Module:** HSIZMON

---

**HSIZ040I** *csid* **WAITING FOR WRITER TASK ATTACHED** *ts*

**Explanation:** A repository switch was triggered by a SWI or STOP command, or by the current repository becoming full, but a writer task is already active. This message follows message HSIZ039I and shows the creation timestamp of the active writer task.

In the message text:

*csid*
current system identifier.

*ts* Time stamp of write task.

**System action:** Data collection is suspended. Wait for the current writer task to complete whereupon a new writer task is created, and a new repository is created, and data collection is resumed.

**Operator response:** Check that there are sufficient resources to dispatch the Usage Monitor address space. Check that there are no serialization problems with system components such as device allocation which could be inhibiting writer task processing.

**System programmer response:** None.

**Module:** HSIZMON

---

**HSIZ041I** *csid* **CURRENT USAGE MONITOR OUTPUT DYNALLOC PARMS:**

**Explanation:** A D-A command was issued to display the current output dynamic allocation parameters, which are shown after this message.

In the message text:

*csid*
current system identifier.

**System action:** The data is displayed and processing continues.

**Operator response:** None required.

**System programmer response:** None.

**Module:** HSIZMON

**HSIZ042I CURRENT USAGE MONITOR OUTPUT SYSTEM ID IS "*csid*"**

**Explanation:** A D-I command was issued to display the current system identifier which is to be contained in output header records.

In the message text:

*csid*
current system identifier.

**System action:** Processing continues.

**Operator response:** None required.

**System programmer response:** None.

**Module:** HSIZMON

**HSIZ043I** *csid* **DATA DISCARDED DUE TO ECSA STORAGE LIMIT**

**Explanation:** The Usage Monitor has detected for the first time in the life of the repository or since a CSA setting change that program usage event data has been discarded due to the ECSA storage usage limit being reached. This limit was set with the CSA command.

In the message text:

*csid*
current system identifier.

**System action:** Processing continues.

**Operator response:** Adjust the Usage Monitor CSA limit as appropriate for the particular system. Ensure that the ECSA size has been generously defined for the system, and that the common page data set size is adequate. Ensure that the Usage Monitor address space is running at a higher priority than all CPU-bound workloads. Generally, monitors need to run at a higher priority than the workloads being monitored.

**System programmer response:** None.

**Module:** HSIZMON

**HSIZ044I** *csid* **SWITCH-AND-WRITE TIME-OF-DAY IS SET TO** *hh*:*mm*

**Explanation:** A D-T command was issued to display the switch-and-write time-of-day setting for this system.

In the message text:

*csid*
current system identifier.

*hh* Hour of the day.

*mm* minute of the hour.

**System action:** Processing continues.

**Operator response:** None required.

**System programmer response:** None.

**Module:** HSIZMON

**HSIZ045I** *csid* **CREATED** *area token-addr*

**Explanation:** A memory object required to hold data was created.

In the message text:

*csid*
current system identifier.

*area*
purpose of the memory object.

*token*
storage token of the memory object.

*addr*
storage address of the memory object.

**System action:** Processing continues.

**Operator response:** None required.

**System programmer response:** None.

**Module:** HSIZMON

**HSIZ046I** *csid* **DELETED** *area token-addr*

**Explanation:** A memory object which was no longer needed was deleted

In the message text:

*csid*
current system identifier.

*area*
purpose of the memory object.

*token*
storage token of the memory object.

*addr*
storage address of the memory object.

**System action:** Processing continues.

**Operator response:** None required.

**System programmer response:** None.

**Module:** HSIZMON

**HSIZ047I** *csid* **USAGE MONITOR - ATTACHING WRITER SEQ-NO-***seqnbr*

**Explanation:** A writer task is being attached to write out repository contents. The writer task sequence number is also reported. The first writer task to run after the Usage Monitor starts has a sequence number of 1.

In the message text:

*csid*
current system identifier.

*seqnbr*
sequence number of writer task this run.

**System action:** Processing continues.

**Operator response:** None required.

**System programmer response:** None.

**Module:** HSIZMON

---

**HSIZ048I** *csid* **USAGE MONITOR - IDENTIFY FAILED HEX RC=***rc*

**Explanation:** The Usage Monitor executed an IDENTIFY macro which failed.

In the message text:

*csid*
current system identifier.

*rc* hexadecimal return code of the IDENTIFY macro.

**System action:** Processing terminates.

**Operator response:** Notify the system programmer.

**System programmer response:** Investigate why an IDENTIFY macro would fail with that return code.

**Module:** HSIZMON

---

**HSIZ049I** *csid* **DATA SET NAME MASK NOT DEACTIVATED, NOT FOUND IN LIST**

**Explanation:** A command to delete a data set name mask from a data set name mask list was issued, but the mask was not present in the list.

In the message text:

*csid*
current system identifier.

**System action:** Processing continues.

**Operator response:** None required.

**System programmer response:** None.

**Module:** HSIZMON

---

**HSIZ050I** *csid* **DATA SET NAME MASK** *mask* **LIST** *list*

**Explanation:** A D-D command was issued to display the data set name mask include and exclude lists. These header and trailer lines mark the start and end of the lists.

In the message text:

*csid*
current system identifier.

*mask*
INCLUDE or EXCLUDE.

*list*
START or END.

**System action:** Processing continues.

**Operator response:** None required.

**System programmer response:** None.

**Module:** HSIZMON

---

**HSIZ053I** *csid* **MONITORING UNIX PROGRAMS?** *ans*

**Explanation:** Either a USS command was issued to change the UNIX program monitoring status or a D-S command was issued. When the answer is YES the usage of programs fetched from UNIX files is monitored. When the answer is NO only the usage of programs from PDS and PDSE libraries is monitored.

In the message text:

*csid*
current system identifier.

*ans*
YES or NO.

**System action:** Processing continues.

**Operator response:** None required.

**System programmer response:** None.

**Module:** HSIZMON

---

**HSIZ054I** *csid* **MONITORING LINK PACK AREA PROGRAMS?** *ans*

**Explanation:** Either an LPA command was issued to change the LPA program monitoring status or a D-S command was issued. When the answer is YES the usage of programs residing in the Link Pack Area is monitored. When answer is NO only the usage of programs loaded into address space regions (and sometimes into CSA) is monitored.

In the message text:

*csid*
current system identifier.

*ans*
YES or NO.

**System action:** Processing continues.

**Operator response:** None required.

**System programmer response:** None.

**Module:** HSIZMON

---

**HSIZ056I** *csid* **PREFER VOLUME SYMBOL OVER SERIAL?** *ans*

**Explanation:** Either a SYM command was issued to change the volume symbol status or a D-S command was issued. When the answer is YES a matching system static symbol which evaluates to the volume serial is collected instead of the volume serial if such a symbol exists, otherwise the actual volume serial is collected. When the answer is NO the captured volume serial number is always output. A YES setting may be useful to improve data matching when system software platform volume switches take place.

In the message text:

*csid*
current system identifier.

*ans*
YES or NO.

**System action:** Processing continues.

**Operator response:** None required.

**System programmer response:** None.

**Module:** HSIZMON

**HSIZ058I** *csid* **FILE HSIZIN IS NOT ALLOCATED - CANNOT PERFORM REFRESH**

**Explanation:** A REF command was issued to refresh settings from commands in the HSIZIN file, but the HSIZIN file had been freed, and was no longer allocated to the Usage Monitor.

In the message text:

*csid*
current system identifier.

**System action:** The refresh operation is suppressed and processing continues.

**Operator response:** Ensure FREE=CLOSE is not specified in the HSIZIN JCL DD statement. Recycle the Usage Monitor to refresh the settings if necessary.

**System programmer response:** None.

**Module:** HSIZMON

**HSIZ059I** *csid* **REFRESH PERFORMED WITH NO ERRORS**

**Explanation:** A REF command was issued to refresh settings from commands in the HSIZIN file. All commands in the HSIZIN file were completed successfully.

In the message text:

*csid*
current system identifier.

**System action:** Processing continues.

**Operator response:** None required.

**System programmer response:** None.

**Module:** HSIZMON

**HSIZ060I** *csid* **REFRESH PERFORMED BUT ERROR(S) FOUND**

**Explanation:** A REF command was issued to refresh settings from commands in the HSIZIN file. At least one command in the HSIZIN file resulted in an error.

In the message text:

*csid*
current system identifier.

**System action:** Processing terminates.

**Operator response:** Examine the output in the HSIZMSG file to determine the problem(s).

**System programmer response:** None.

**Module:** HSIZMON

**HSIZ063I** *csid* **COLLECTING "UNKNOWN" EVENTS?** *ans*

**Explanation:** Either a UNK command was issued or a D-S command was issued. When the answer is YES this message indicates that the Usage Monitor logs events with incomplete data which would not normally be collected. Data base content is not affected.

In the message text:

*csid*
current system identifier.

*ans*
YES or NO.

**System action:** Processing continues.

**Operator response:** None required.

**System programmer response:** None.

**Module:** HSIZMON

**HSIZ064I** *csid* **WILL WRITER TASK COMPRESS THE DATA?** *ans*

**Explanation:** Either a ZIP command was issued to change the output compression setting or a D-S command was issued. When the answer is YES the writer task writes compressed data to reduce I/O volumes.

In the message text:

*csid*
current system identifier.

*ans*
YES or NO.

**System action:** Processing continues.

**Operator response:** None required.

**System programmer response:** None.

**Module:** HSIZMON

---

**HSIZ065I** *csid* **WILL WRITER TASK CORRECT LINKLIST DSN?** *ans*

**Explanation:** Either an LLC command was issued or a D-S command was issued. When the answer is YES the writer task will perform a BLDL for programs known to have been fetched from the link list, and each output record for such programs will be altered to reflect the link list data set name that the writer task found the program in.

In the message text:

*csid*
current system identifier.

*ans*
YES or NO.

**System action:** Processing continues.

**Operator response:** None required.

**System programmer response:** None.

**Module:** HSIZMON

---

**HSIZ066I** *csid nbr* **IDLE ELEMENT(S) "LOST" DUE TO ZERO POINTER**

**Explanation:** The Usage Monitor was terminating normally when a storage accounting discrepancy was discovered. The storage for the idle element chain was being freed when it was found to be terminated by a zero pointer before the expected number of elements had been processed. The most probable cause is a storage overlay. This may or may not represent a Usage Monitor logic error. The size of common storage which may be unusable until the next IPL can be calculated by multiplying the element count by the size of an element.

In the message text:

*csid*
current system identifier.

*nbr*
the number of elements being reported.

**System action:** Termination continues.

**Operator response:** Determine if the size of the potential loss of common storage is likely to impact upon system stability, and take the appropriate action. Ensure that all appropriate maintenance has been applied.

**System programmer response:** None.

**Module:** HSIZMON

---

**HSIZ068I** *csid* **COLLECTING JOB ACCOUNTS NOW?** *ans*

**Explanation:** A D-S command was issued. When the answer is YES job account data is currently being collected as program usage events are recorded. When the answer is NO job account data is not being collected currently.

In the message text:

*csid*
current system identifier.

*ans*
YES or NO.

**System action:** Processing continues.

**Operator response:** None required.

**System programmer response:** None.

**Module:** HSIZMON

---

**HSIZ069I** *csid* **COLLECTING JOB ACCOUNTS LATER?** *ans*

**Explanation:** Either a JAC command was issued or a D-S command was issued. When the answer is YES job account data will be collected after the next Usage Monitor collection repository switch. If the answer is NO job account data will not be collected from that time onwards.

In the message text:

*csid*
current system identifier.

*ans*
YES or NO.

**System action:** Processing continues.

**Operator response:** None.

**System programmer response:** None.

**Module:** HSIZMON

---

**HSIZ070I** *csid* **COLLECTING REGISTERED PRODUCT DATA NOW?** *ans*

**Explanation:** A D-S command was issued. When the answer is YES registered software product data from SMF is currently being collected by the Usage Monitor. When the answer is NO then this SMF data is not being currently collected.

In the message text:

*csid*
current system identifier.

*ans*
YES or NO.

**System action:** Processing continues.

**Operator response:** None.

**System programmer response:** None.

**Module:** HSIZMON

---

**HSIZ071I** *csid* **COLLECTING REGISTERED PRODUCT DATA LATER?** *ans*

**Explanation:** Either a PRS command was issued or a D-S command was issued. When the answer is YES registered software product data from SMF will be collected after the next Usage Monitor collection repository switch. When the answer is NO this SMF data will not be collected after the next switch.

In the message text:

*csid*
current system identifier.

*ans*
YES or NO.

**System action:** Processing continues.

**Operator response:** None.

**System programmer response:** None.

**Module:** HSIZMON

---

**HSIZ072I** *csid* **COLLECTING DYNAMIC CAPACITY DATA NOW?** *ans*

**Explanation:** A D-S command was issued. When the answer is YES hardware capacity information is currently being collected by the Usage Monitor. When the answer is NO hardware capacity information is not being currently collected.

In the message text:

*csid*
current system identifier.

*ans*
YES or NO.

**System action:** Processing continues.

**Operator response:** None.

**System programmer response:** None.

**Module:** HSIZMON

---

**HSIZ073I** *csid* **COLLECTING DYNAMIC CAPACITY DATA LATER?** *ans*

**Explanation:** Either a CAP command was issued or a D-S command was issued. When the answer is YES the Usage Monitor will collect hardware capacity information after the next Usage Monitor collection repository switch. When the answer is NO the hardware capacity information will not be collected after the next switch.

In the message text:

*csid*
current system identifier.

*ans*
YES or NO.

**System action:** Processing continues.

**Operator response:** None.

**System programmer response:** None.

**Module:** HSIZMON

---

**HSIZ074I** *csid* **OUTPUT NAMES OF COLLECTED USERS?** *ans*

**Explanation:** Either a UNM command was issued or a D-S command was issued. When the answer is YES collected user names will be included in the data output by the Usage Monitor writer task. When the answer is NO user names will not be written to the output data set. Even if the answer is YES, no user names will be output if no user information was collected.

In the message text:

*csid*
current system identifier.

*ans*
YES or NO.

**System action:** Processing continues.

**Operator response:** None.

**System programmer response:** None.

**Module:** HSIZMON

---

**HSIZ075I** *csid* **COLLECTING USER INFORMATION NOW?** *ans*

**Explanation:** A D-S command was issued. When the answer is YES the identifier and name of each program user is currently being collected by the Usage Monitor. When the answer is NO these user details are not being currently collected.

In the message text:

*csid*
current system identifier.

*ans*
YES or NO.

**System action:** Processing continues.

**Operator response:** None.

**System programmer response:** None.

**Module:** HSIZMON

---

**HSIZ076I** *csid* **COLLECTING USER INFORMATION LATER?** *ans*

**Explanation:** Either a UID command was issued or a D-S command was issued. When the answer is YES the identifier and name of each program user will be collected after the next Usage Monitor collection repository switch. When the answer is NO these userid details will not be collected after the next switch.

In the message text:

*csid*
current system identifier.

*ans*
YES or NO.

**System action:** Processing continues.

**Operator response:** None.

**System programmer response:** None.

**Module:** HSIZMON

**HSIZ077I** *csid* **COLLECTING JOB NAMES NOW?** *ans*

**Explanation:** A D-S command was issued. When the answer is YES the names of jobs using programs are currently being collected by the Usage Monitor. When the answer is NO only generic address space type data such as JOB, STC and TSU is currently being collected instead of individual job names.

In the message text:

*csid*
current system identifier.

*ans*
YES or NO.

**System action:** Processing continues.

**Operator response:** None.

**System programmer response:** None.

**Module:** HSIZMON

**HSIZ078I** *csid* **COLLECTING JOB NAMES LATER?** *ans*

**Explanation:** Either a JNM command was issued or a D-S command was issued. When the answer is YES the names of jobs using programs will be collected after the next Usage Monitor collection repository switch. When the answer is NO only generic address space type data such as JOB, STC and TSU will be collected after the next switch instead of individual job names.

In the message text:

*csid*
current system identifier.

*ans*
YES or NO.

**System action:** Processing continues.

**Operator response:** None.

**System programmer response:** None.

**Module:** HSIZMON

**HSIZ079I** *csid* **ALLOWING CICS EVENT COLLECTION?** *ans*

**Explanation:** A D-S command was issued. When the answer is YES this message indicates that CICS event collection from suitably customized CICS regions is enabled. When the answer is NO CICS generated program usage information will not be collected.

In the message text:

*csid*
current system identifier.

*ans*
YES or NO.

**System action:** Processing continues.

**Operator response:** None required.

**System programmer response:** None.

**Module:** HSIZMON

**HSIZ080I** *csid dsn*

**Explanation:** Displays the dataset name mask for a D-D command.

In the message text:

*csid*
current system identifier.

*dsn*
data set name.

**System action:** Processing continues.

**Operator response:** None required.

**System programmer response:** None.

**Module:** HSIZMON

**HSIZ081I** *csid* **MONITORING PREVIOUSLY RUNNING PROGRAMS?** *ans*

**Explanation:** A D-S command was issued. When the answer is YES this message indicates usage data for programs resident in the regions of jobs which are older than the current Usage Monitor repository and which have SMF interval recording active will be collected. When the answer is NO there will be no usage data collected for programs which were running before the current repository was created and do not terminate before the repository collection period ends.

In the message text:

*csid*
current system identifier.

*ans*
YES or NO.

**System action:** Processing continues.

**Operator response:** None required.

**System programmer response:** None.

**Module:** HSIZMON

---

**HSIZ082I** *csid* **RETAINING ALL BATCH JOB IDENTIFIERS?** *ans*

**Explanation:** Either a JID or a D-S command was issued. When the answer is YES the message indicates that data for batch jobs with different JES job identifiers will not be aggregated together but will be reported as usage by separate jobs. When the answer is NO data for usage of programs will be aggregated only by userid and job name and only the most recent job identifier will be retained. Aggregation corresponding to the NO answer is always used for started tasks and TSO sessions.

In the message text:

*csid*
current system identifier.

*ans*
YES or NO.

**System action:** Processing continues.

**Operator response:** None required.

**System programmer response:** None.

**Module:** HSIZMON

---

**HSIZ083I** *csid***ADJUSTING FOR HYPERVISOR TIME OFFSET?** *ans*

**Explanation:** Either a HOF or a D-S command was issued. When the answer is YES the message indicates that timestamps in collected data will be adjusted by subtracting the hypervisor time offset from the z/OS data and time. When the answer is NO the unaltered local z/OS data and time will be used in the collected data.

In the message text:

*csid*
current system identifier.

*ans*
YES or NO.

**System action:** Processing continues.

**Operator response:** None required.

**System programmer response:** None.

**Module:** HSIZMON

---

**HSIZ85I** *csid cmd* **NO LONGER SUPPORTED**

**Explanation:** A command was issued that is no longer a functional Usage Monitor command.

In the message text:

*csid*
current system identifier.

*cmd*
name of the issued command.

**System action:** The command is ignored. Processing continues.

**Operator response:** Do not issue this command.

**System programmer response:** None.

**Module:** HSIZMON

---

**HSIZ201I** **DYNALLOC FAILURE RC=***rc* **ERROR=***s99error* **INFO=***s99info* **DSN=***dsn*

**Explanation:** The writer task could not dynamically allocate a new output data set.

In the message text:

*rc* DYNALLOC return code.

*s99error*
dynamic allocation reason code (DARC).

*s99info*
dynamic allocation information code.

*dsn*
name of the data set being allocated.

**System action:** Processing of the repository is terminated, and the data lost.

**Operator response:** Correct the cause of the allocation failure. If necessary, use the DSN, PRI, SEC and UNT commands to customize the allocation request for your installation. Note: The meanings of most DARC values are usually available in Appendix A of the ISPF Tutorial.

**System programmer response:** None.

**Module:** HSIZ0203

---

**HSIZ202I** **USAGE MONITOR - COMPRESSION SUBROUTINE ERROR**

**Explanation:** While processing repository data the compression subroutine encountered an error. The error message from the compression subroutine immediately follows this message.

**System action:** Processing of the repository is terminated, and the data lost.

**Operator response:** Correct the error described in the

message from the compression subroutine. If you cannot resolve this issue then gather appropriate diagnostic materials and contact IBM support.

**System programmer response:** None.

**Module:** HSIZ0203

---

**HSIZ203I USAGE MONITOR - SORT FAILED - RC=*rc***

**Explanation:** While sorting repository data the SORT task ended with a non-zero condition code which is taken to mean that the sort was not successful. This message is followed by message HSIZ205I.

In the message text:

*rc* decimal return code of the sort subtask.

**System action:** The output data set is closed, and the writing of unsorted data to the same data set is attempted.

**Operator response:** Consult the documentation of the SORT utility. The contents of the SORT report file (DDNAME=SYSOUT) may be helpful.

**System programmer response:** None.

**Module:** HSIZ0203

---

**HSIZ204I USAGE MONITOR - SORT ABENDED - ABEND CODE=*abend***

**Explanation:** While sorting repository data the SORT task ended abnormally. This message is followed by message HSIZ205I.

In the message text:

*abend*
the abend code of the sort subtask.

**System action:** The output data set is closed, and the writing of unsorted data to the same data set is attempted.

**Operator response:** Investigate why such an abend could occur. The contents of the SORT report file (DDNAME=SYSOUT) may be helpful.

**System programmer response:** None.

**Module:** HSIZ0203

---

**HSIZ205I USAGE MONITOR - UNSORTED DATA WILL BE WRITTEN**

**Explanation:** The sorting of output data has failed so the data is now written unsorted.

**System action:** The message is preceded by either HSIZ203I or HSIZ204I. After the SORT task ended the output data set has been closed and reopened. Repository data is about to be written to the output data set.

**Operator response:** Investigate why the sort failed.

**System programmer response:** None.

**Module:** HSIZ0203

---

**HSIZ206I** *errmsg*

**Explanation:** The HSISHRNK compression routine issued an error message which is displayed.

In the message text:

*errmsg*
error message from HSISHRNK.

**System action:** The message is preceded by message HSIZ202I.

**Operator response:** Examine the message for further information.

**System programmer response:** None.

**Module:** HSIZ0203

---

**HSIZ207I** *mb* **MB NOT AVAILABLE - IARV64 RC=*rc* REASON=*rs***

**Explanation:** The writer task tried to acquire extra storage to assist with sorting data, but the requested storage was not available.

In the message text:

*mb* megabytes requested.

*rc* return code from IARV64.

*rs* reason code from IARV64.

**System action:** Processing continues. Some output records may not be in sort order.

**Operator response:** Investigate the IARV64 feedback. If appropriate, increase MEMLIMIT for the Usage Monitor.

**System programmer response:** None.

**Module:** HSIZ0203

---

**HSIZ301I DESERV FUNC=EXIT RC=*rc* REASON=*rs***

**Explanation:** DESERV FUNC=EXIT issued a non-zero return code.

In the message text:

*rc* return code from DESERV.

*rs* reason code from DESERV.

**System action:** The DESERV exit is not installed.

**Operator response:** Notify the system programmer.

**System programmer response:** Research the DESERV feedback to determine why the exit could not be installed.

**Module:** HSIZ0303

---

**HSIZ302I CSVDYNEX ADD (*excd*) RC=*rc* REASON=*rs***

**Explanation:** CSVDYNEX ADD issued a non-zero return code. An exit could not be dynamically defined for the named exit point.

In the message text:

*excd*
dynamic exit point name.

*rc* return code from CSVDYNEX.

*rs* reason code from CSVDYNEX.

**System action:** The SMF exit is not installed.

**Operator response:** Notify the system programmer.

**System programmer response:** Research the CSVDYNEX feedback to determine why the exit could not be installed. Ensure that IEFU84 is an active SMF exit for the system or subsystem. If you cannot resolve this issue then gather appropriate diagnostic materials and contact IBM support.

**Module:** HSIZ0303

---

**HSIZ303I ATTRIBUTE MISMATCH - *mod* NOT INSTALLED**

**Explanation:** The examined SVC table entry did not have the expected attributes.

In the message text:

*mod*
module name.

**System action:** The SVC intercept is not installed.

**Operator response:** Notify the system programmer.

**System programmer response:** Gather appropriate diagnostic materials and contact IBM support.

**Module:** HSIZ0303

---

**HSIZ306I BAD *statnm* ENTRY PGM=*pgm* JOB=*jbn* USER=*user* ID=*id* DATE=*date* REJECTED**

**Explanation:** An invalid work element has been detected and some of its contents are displayed.

In the message text:

*statnm*
status name.

*pgm*
program name.

*jbn*
job name.

*user*
user name.

*id* id name.

*date*
date.

**System action:** Attempted to dump some data to HSIZSNAP if the file is allocated, and will then try to free the work element without processing its contents.

**Operator response:** Notify the system programmer.

**System programmer response:** The problem is indicative of a storage overlay. Gather appropriate diagnostic materials and contact IBM support.

**Module:** HSIZ3060

---

**HSIZ307I *csid* SCAN OF *volume* VTOC ABENDED -S *cde***

**Explanation:** A scan of the named volume abended with the reported abend code. The scan was performed to find the data set name of a program library that is in use.

In the message text:

*csid*
current system identifier.

*volume*
volume where VTOC resides.

*cde*
abend code from VTOC scan.

**System action:** Processing continues.

**Operator response:** Notify the system programmer.

**System programmer response:** Check the named volume for error conditions. If necessary, gather appropriate diagnostic materials and contact IBM support.

**Module:** HSIZMON

---

**HSIZ310I MODULE *mod* INSTALLED AT ADDRESS *loadpt* SIZE *size***

**Explanation:** The Usage Monitor has dynamically loaded a module into common storage and will now register it in DLPA.

In the message text:

*mod*
module name.

*loadpt*
module load point.

*size*
module size.

**System action:** Processing continues.

**Operator response:** None.

**System programmer response:** None.

**Module:** HSIZ0303

**HSIZ311I CSVDYLPA RC=*rc* RS=*rs* FOR *mod***

**Explanation:** The Usage Monitor attempted to register a newly installed module in DLPA, but CSVDYLPA issued a non-zero return code.

In the message text:

*rc* decimal return code issued by CSVDYLPA.

*rs* hexadecimal reason code issued by CSVDYLPA.

*mod*
name of the module being registered.

**System action:** Processing continues.

**Operator response:** Notify the system programmer.

**System programmer response:** Investigate why the named module could not be registered in the current DLPA configuration.

**Module:** HSIZ0303

**HSIZ999U HSIMSG/HSIZMSG FAILURE - MSGID=*msgid* RC=*rc* RS=*rs***

**Explanation:** HSIMSG was called to produce a message text, but the call failed.

In the message text:

*msgid*
identifier of the failing message.

*rc* HSIMSG return code.

*rs* HSIMSG reason code.

**System action:** Terminates with a condition code of 20.

**Operator response:** Inform the system programmer.

**System programmer response:** Ensure Joblib/Steplib contains the library where the HSIZMSG message module resides. If you cannot resolve this issue then gather appropriate diagnostic materials and contact IBM support.

**Module:** HSIZMON

# HSIC - Operation messages

### Return codes

*Table 103. Return codes and their meaning*

| Return code | Description |
|---|---|
| 0 | No errors encountered. All requests processed successfully. |
| 16 | Unrecoverable error. No requests processed. SYSIN or HSIPZIP or INQSOUT File cannot be used, or unsupported operating system. |

### Message suffix codes

*Table 104. Message suffix codes and associated condition codes*

| Suffix | Meaning | Raises minimum condition code to: |
|---|---|---|
| I | Information message | 0 |
| W | Warning message | 4 |
| E | Error message | 8 |
| S | Severe error message | 12 |
| U | Unrecoverable error message | 16 |

### Message texts and explanations

All numeric completion codes of system services reported in these messages are in hexadecimal unless otherwise stated.

**HSIC002E A message is missing from the internal repository**

**Explanation:** A message is missing from the internal message repository. When the default language is not English, it could simply mean that no translation of the

given message exists. If the default language is English, that would indicate an error in the given application.

**System action:** The application would normally continue ignoring the given message number , but the specific action depends on the code attempting to issue the message which could also terminate the application.

**User response:** Contact IBM support.

---

**HSIC003U** **The internal message repository is corrupted**

**Explanation:** When attempting to issue a message, the internal message repository layout did not follow the expected format.

**System action:** The application terminates.

**User response:** Contact IBM support.

---

**HSIC020E** *application-name* **encountered errors. Error code =** *errorcode*

**Explanation:** The Application has encountered errors during processing. This is a general message on completion indicating that an error has occurred.

**System action:** Completes with given error code .

**User response:** Refer to additional message, or to the section "Return codes" on page 225, and to the log for more details on the specific error. Contact IBM support.

---

**HSIC021S** *application-name* **encountered fatal errors. Error code =** *error-code*

**Explanation:** The Application has encountered fatal errors during processing.

**System action:** Terminates with given error code

**User response:** Refer to additional message, or to the section "Return codes" on page 225, and to the log for more details on the specific error. Contact IBM support.

---

**HSIC023E** **Inquisitor Import error occurred in opening:** *filename*

**Explanation:** The Inquisitor import could not open the given file.

**System action:** Terminates without processing any records.

**User response:** Check that the file exists, and if it does, check for any additional log message identifying the error. Contact IBM support.

---

**HSIC024E** **Inquisitor Import input file is in error. It looks like a usage data file**

**Explanation:** The inquisitor import has encountered an invalid input file.

**System action:** Terminates without processing any records.

**User response:** Check that the input file is a valid file. Contact IBM support.

---

**HSIC025E** **Inquisitor Import input file is in error. It looks like a hardware data file**

**Explanation:** The Inquisitor Import has encountered an invalid input file.

**System action:** Terminates without processing any records.

**User response:** Check that the input file is a valid file. Contact IBM support.

---

**HSIC026E** **Inquisitor Import detected that table** *tablename* **is missing or invalid**

**Explanation:** The expected table is missing from the database or has invalid format. This suggests a mismatch between the database and this version of the product.

**System action:** Terminates without processing any records.

**User response:** Check for a version mismatch between the database and the version of the product. Contact IBM support.

---

**HSIC027S** **Inquisitor Import table** *tablename* **is missing a column**

**Explanation:** The given table is missing an expected column. This suggests a mismatch between the database and this version of the product.

**System action:** The application terminates without processing any records.

**User response:** Check for a version mismatch between the database and the version of the product. Contact IBM support.

---

**HSIC028S** **Inquisitor Import table** *tablename* **appears to be an old version**

**Explanation:** The given table in the database does not have the expected format.

**System action:** The application terminates without processing any records.

**User response:** Check for a version mismatch between the database and the version of the product. Contact IBM support.

---

**HSIC029S** **Inquisitor Import error when writing to table** *tablename*

**Explanation:** An SQL error occurred when attempting to write to the given table.

**System action:** The application terminates.

**User response:** Check the log for additional details about the given error. Contact IBM support.

**HSIC030S** **The Inquisitor Import did not find a valid system header record in the input file**

**Explanation:** The input file does not follow the expected format.

**System action:** The application terminates.

**User response:** Check that the correct input file is supplied, and that there is no version mismatch. Contact IBM support.

**HSIC034S** **Error reading Repository TPARAM table**

**Explanation:** An error occurred while reading the TPARAM Repository table.

**System action:** The application terminates.

**User response:** Check the log for any additional messages indicating the cause of the error. Contact IBM support.

**HSIC035E** **The Repository is in use by the** *application-name*

**Explanation:** The application cannot run because the Repository is already in use by another application. Wait until *application-name* completes before running the current application. If the Repository is not in use by *application-name*, then the cause could be that it was previously run, but did not run to completion. To correct the problem, either rerun the *application-name* identified in this message, or alternatively, run the HSISTPRM supplied job to reset FVALUE to 0 where FKEY = PROCRUN in the TPARAM table.

**System action:** The application terminates.

**User response:** Check the application is not already in use, before running this application.

**HSIC036E** **Syntax error scanning TPARAMS file on line** *linenumber*

**Explanation:** The TPARAM file does not conform to the required syntax on the given line.

**System action:** The specified option or value is ignored, and its default value is used where applicable.

**User response:** Check that valid options/values are supplied as specified in the documentation of the application that you are running.

**HSIC037E** **Schema** *schemavalue* **is too long in param** *param*

**Explanation:** A schema id that is too long has been specified.

**System action:** The application terminates.

**User response:** Check that the schema id does not exceed 8 characters in length.

**HSIC038E** **Unbalanced quote for value:** *value* **in param:** *param*

**Explanation:** A starting quote was found for the given parameter that has no matching end quote.

**System action:** The application terminates.

**User response:** Check that the given parameter has matching quotes

**HSIC039E** **Illegal character in value***:value* **of param:***param*

**Explanation:** An invalid character was found in the given value.

**System action:** The application terminates.

**User response:** Check that the given parameter value is valid for its type.

**HSIC040E** **Reserved word:** *reservedword* **in param:** *param*

**Explanation:** A reserved word or system value schema ID was chosen as a parameter value.

**System action:** The application terminates.

**User response:** Specify a different parameter value

**HSIC041W** **value:***value* **in param:***param* **is not a recommended schema ID**

**Explanation:** The value is not recommended because of possible conflicts with existing values.

**System action:** The application continues.

**User response:** Please choose a different value to avoid any conflicts

**HSIC042E** **TPARAM file: param***:param* **has an invalid proposed value:** *value*

**Explanation:** The parameter cannot be set to the given value, because the value is not valid..

**System action:** The value is ignored, and the application continues.

**User response:** Please choose a valid value as per the

documentation of the given application

**HSIC043E The application has failed to open the TPARAM file. Error:** *errordescription*

**Explanation:** The application could not open the TPARAM file. The error description contains more details regarding the reason for the error.

**System action:** The application terminates.

**User response:** Check that the TPARAM file exists and is valid.

**HSIC045E String** *string* **cannot exceed** *numberchars* **in length**

**Explanation:** A parameter length limit has been exceeded.

**System action:** The application terminates.

**User response:** Ensure that the specified parameter length is not exceeded.

**HSIC050E The** *program-name* **program has detected an invalid date parameter**

**Explanation:** A date parameter was found to be invalid.

**System action:** The application terminates.

**User response:** Ensure that the date format is valid, and start dates do not overlap end dates.

**HSIC051S Error adding record**

**Explanation:** An SQL error occurred when adding a record to a table.

**System action:** The application terminates.

**User response:** Check the log for additional information about the error. Contact IBM support.

**HSIC052S Error updating record**

**Explanation:** An SQL error occurred when updating a record in a table.

**System action:** The application terminates.

**User response:** Check the log for additional information about the error. Contact IBM support.

**HSIC053S Error deleting record**

**Explanation:** An SQL error occurred when deleting a record from a table.

**System action:** The application terminates.

**User response:** Check the log for additional information about the error. Contact IBM support.

**HSIC054E Usage Summary detected an invalid SUMBY value**

**Explanation:** The Usage Summary detected an invalid SUMBY value.

**System action:** The specified value is ignored. The application continues using the default SUMBY value.

**User response:** Refer to the documentation of the Usage Summary parameter for valid SUMBY values.

**HSIC055S Table initialization failure during Repository Merge**

**Explanation:** At least one table initialization failed when merging repositories.

**System action:** The application terminates.

**User response:** Check the log for any additional details about this error. Contact IBM support.

**HSIC056S Some table destination fields are smaller than source**

**Explanation:** Some fields in the target repository are not large enough to fit the contents of fields in the source repository.

**System action:** The application terminates, and the repositories are not merged.

**User response:** Check that the destination repository is not an older version than the source repository. You can recreate the destination repository using the latest version of the product. If the problem persists, contact IBM support.

**HSIC057E A value for parameter:** *parameter-name* **must be specified**

**Explanation:** A mandatory parameter for this application has not been specified.

**System action:** The application terminates during the syntax checking of input parameters.

**User response:** Ensure that a value for the given parameter is specified. Refer to the documentation of the failing application for an explanation of the given parameter and/or valid parameter values.

**HSIC058E Could not open** *filename*

**Explanation:** File could not be opened.

**System action:** The application terminates.

**User response:** Check the log for additional information about the error. Contact IBM support.

**HSIC059E Could not read** *filename*

**Explanation:** File could not be read.

**System action:** The application terminates.

**User response:** Check the log for additional information about the error. Contact IBM support.

**HSIC060E IQDATA DD does not contain unzipped IQ data**

**Explanation:** The input IQDATA dataset does not contain unzipped IQ data.

**System action:** The application terminates.

**User response:** Check the log for additional information about the error. Contact IBM support.

**HSIC061E Internal error hcreate***(number)* **phase1a failed**

**Explanation:** An internal error has occurred.

**System action:** The application terminates

**User response:** Check the log for additional information about the error. Contact IBM support.

**HSIC062E No SMF 30-2 or 30-4 data matched IQ data**

**Explanation:** No match was found for the SMF data and IQ data.

**System action:** The application terminates.

**User response:** Check that the correct data sets have been used. Contact IBM support

**HSIC063E Internal error hsearch***(key)* **table add failed**

**Explanation:** An error occurred when inserting data into a table.

**System action:** The application terminates.

**User response:** Check the log for additional information about the error. Contact IBM support.

**HSIC064E Could not write** *type* **to FMOUT**

**Explanation:** Could not write to file FMOUT.

**System action:** The application terminates.

**User response:** Check the log for additional information about the error. Contact IBM support.

**HSIC065E SYSUT1 data Is not IQ text or UM text**

**Explanation:** The SYSUT1 dataset does not contain the expected data.

**System action:** The application terminates.

**User response:** Check that the SYSUT1 dataset is correct. Contact IBM support.

**HSIC066E Internal error hsearch***(key)* **table failed**

**Explanation:** An error occurred when retrieving data from a table.

**System action:** The application terminates.

**User response:** Check the log for additional information about the error. Contact IBM support.

**HSIC067E Unable to acquire storage**

**Explanation:** An error has occurred when attempting to acquire storage.

**System action:** The application terminates.

**User response:** Try increasing the region size specified in the region parameter on the JOB or EXEC statement in the JCL for the job. Contact IBM support.

**HSIC068E IBMMOD Internal error**

**Explanation:** An internal error has occurred.

**System action:** The application terminates.

**User response:** Check the log for additional information about the error. Contact IBM support.

**HSIC069E IBMMOD_INIT internal error**

**Explanation:** An error occurred when retrieving data from a table.

**System action:** The application terminates.

**User response:** Check the log for additional information about the error. Contact IBM support.

**HSIC070I A full rematch will be performed**

**Explanation:** A full import and rematch will be performed, which will not try to exclude modules of unchanged libraries. The default behaviour is to exclude such libraries from matching, which would normally lead to faster processing.

The program performs a full rematch, if any of the following is true :

- If requested by the FULLREMATCH option.
- When the specified inventory is not found, for example on the first run when the inventory has not yet been created, and no previous match was done.

- If it is safer to perform a full rematch, as when a GKB change is detected or the REPLACEFULL option is in effect.

More specific details on why a full rematch is being performed, can be found in the log.

**System action:** A full rematch of the data is performed. All libraries are processed.

**User response:** Ensure that a FULLREMATCH and the REPLACEFULL options are not in effect for better performance, unless a full rematch is desired.

If this is the first run of the Inquisitor Import, or there has been recent a change to the GKB, then no action is necessary; the program will try on subsequent runs (subsequent to loading the current data into the repository) to exclude unchanged libraries.

---

**HSIC071I** *&number_modules* **modules in** *&number_libraries* **unchanged libraries were ignored**

**Explanation:** This is a report of the number of the modules and libraries that are ignored when the FULLREMATCH option is not in effect. Details of these ignored libraries are in the log.

**System action:** None.

**User response:** None.

---

**HSIC073E** **Usage Import for system** *SID* **on** *time* **is a duplicate and will be ignored.**

**Explanation:** This input file has already been processed.

**System action:** The file is not processed. Processing is terminated.

**User response:** Provide an input file that is the output of a more recent Usage Monitor output.

---

**HSIC074E** **IQ input file dated** *time1* **is earlier than the latest** *os_type* **scan for SID** *sid* **dated** *time2***.**

**Explanation:** The input file is earlier than an an input file that was already processed for this system.

**System action:** The file is not processed. Processing is terminated.

**User response:** Provide an input file that is the output of a more recent Inquisitor scan.

---

**HSIC075I** *num_libs* **libraries containing** *num_modules* **modules are mirrors.**

**Explanation:** The program encountered mirror libraries, which are libraries that are identical in name and volume to libraries that the program previously processed on different SIDs.

**System action:** The program records the names and locations of such libraries but does not process their contents of modules. The program displays the names of such libraries in the log, as well as their current and base SIDs.

**User response:** This operation is normal as long as the repository is set up correctly to receive SIDs containing libraries that are unique in library and volume name except when they are copies or shared, and no other SIDs that do not conform to these rules were mistakenly placed into the repository. Refer to the Inquisitor Import description for more information. Refer to the log for a list of mirror libraries and their base SIDs. You can run the System Deletion Job to remove any SIDs that have been placed recently but incorrectly into this repository.

---

**HSIC076I** **IQ input file dated** *time* **is a duplicate of the last** *OS_TYPE* **scan for SID** *SID_NAME***.**

**Explanation:** This input file has already been processed.

**System action:** The program continues and the file is processed.

**User response:** None.

---

**HSIC077I** **Analyzer initialization complete.**

**Explanation:** The Analyzer has started.

**System action:** Processing continues.

**User response:** None.

---

**HSIC078I** **Analyzer has now terminated.**

**Explanation:** The Analyzer has stopped.

**System action:** Processing continues.

**User response:** None.

---

**HSIC079I** **Analyzer is unable to display URL because gethostname returned a null string for the host name.**

**Explanation:** The analyzer attempted to get the host name of the system it is running on using the z/OS C/C++ gethostname library function and a null string was returned. This may happen if the TCP/IP TCPIP.DATA configuration file does not have a HOSTNAME statement.

**System action:** The analyzer is unable to display the url that clients can use to access it in its log. The analyzer continues processing and listening for connections from clients.

**User response:** Add a HOSTNAME statement to the TCP/IP TCPIP.DATA configuration file. In order for this change to have an effect, the TCP/IP address space

would need to be stopped and restarted. Refer to the z/OS Communications Server: IP Configuration Reference for more information on the HOSTNAME statement, on the TCPIP.DATA configuration file and how to modify them.

---

**HSIC080I** **Analyzer is unable to display URL because getaddrinfo for host hostName failed with errno errnoNumericValue, errnoDescriptionString.**

**Explanation:** The analyzer attempted to get the fully qualified domain name and IP address for the host it is running on which is named hostName. The z/OS C/C++ getaddrinfo library function was used and it returned the errno shown.

**System action:** The analyzer is unable to display the url that clients can use to access it in its log. The analyzer continues processing and listening for connections from clients.

**User response:** Refer to the z/OS Communications Manager: IP and SNA Codes manual for a description of Resolver return codes. It is possible that the host name could not be resolved due to a Resolver configuration problem or a Domain Name System (DNS) configuration problem. Refer to z/OS Communications Server: IP Configuration Guide and the z/OS Communications Server: IP Configuration Reference for information about how to configure Resolver and the BIND 9-based Domain Name System.

---

**HSIC081E** **Incompatible GKB level for gkb schema:***schema***. Expected level:***expected_level_number***, received::***received_level_number***.**

**Explanation:** A mismatch between the GKB level and the level expected by the program was found. This indicates that the GKB and the code are at different maintenance levels. If the *received_level_number* is greater than the *expected_level_number*, it indicates that a later level of the GKB is used than can be handled by the current maintenance level of the code.

If the *received_level_number* is less than the *expected_level_number*, it indicates that an old level of the GKB, that can no longer be handled by the current maintenance level of the code, is used.

**System action:** The application terminates.

**User response:** Apply GKB and code maintenance to the indicated GKB schema as required to ensure code and GKB are compatible. Contact IBM support for further assistance.

---

**HSIC082E** **Incompatible repository level for repository schema:** *schema***. Expected level:***expected_level_number***, received::***received_level_number***.**

**Explanation:** A mismatch between the Database repository level and the level expected by the program was found. This indicates that the repository and the code are at different maintenance levels. If the *received_level_number* is greater than the *expected_level_number*, it indicates that a later level of the repository is used than can be handled by the current maintenance level of the code.

If the *received_level_number* is less than the *expected_level_number*, it indicates that an old level of the repository, that can no longer be handled by the current maintenance level of the code, is used.

**System action:** The application terminates.

**User response:** Apply Database repository and code maintenance to the indicated repository schema as required to ensure code and repository are compatible. Contact IBM support for further assistance.

---

**HSIC083I** **GKB schema:** *schema* **of version date:***gkb_date* **is out of date. It is more than** *number of months* **old.**

**Explanation:** The used GKB of the given schema name is likely to be superseded and can be replaced by a more recent GKB containing more product updates. The GKB that is used is older than the default value of the *number_of_months* from the current date which means the latest identification changes may be missed.

**System action:** The application continues.

**User response:** Apply GKB maintenance as soon as possible.

---

**HSIC084E** **GKB schema:** *schema* **is missing configuration parameters.**

**Explanation:** The used GKB of the given schema name is missing configuration parameters that the program needs. This is most likely to occur if the GKB has not been loaded successfully.

**System action:** The application terminates.

**User response:** Ensure GKB is loaded successfully. Contact IBM support if problem persists.

---

**HSIC085E** **GKB schema:** *schema* **does not have any scorecards.**

**Explanation:** The used GKB of the given schema name does not have any scorecards that can be used to match discovered modules. This is most likely to occur if the GKB has not been loaded successfully.

**System action:** The application terminates.

**User response:** Ensure GKB is loaded successfully. Contact IBM support if problem persists.

## Return codes

| | |
|---|---|
| **6016** | **Input text file open error** |
| **6060** | **Input Parameter error** |
| **6061** | **Database open error** |
| **6062** | **Database commit error** |
| **6063** | **Error reading repository TPARAM table** |
| **6065** | **Repository is in use** |
| **6066** | **Unknown SID parameter value** |
| **6067** | **SQL error** |
| **6068** | **Expected parameter missing from the TPARAM table** |
| **6069** | **Specified SID is not found** |
| **6070** | **Invalid data was encountered** |
| **6071** | **Usage Import file is duplicate** |
| **6072** | **IQ Import file is duplicate or of earlier date.** |
| **6073** | **File read error.** |
| **6074** | **GKB level is unsupported.** |
| **6075** | **Repository level is unsupported.** |
| **6076** | **GKB is missing configuration parameters.** |
| **6077** | **GKB is missing scorecards.** |
| **6203** | **Inquisitor Import table open fail** |
| **6204** | **MVS system header record not found in input file** |
| **6205** | **Unix System Services header record not found in input file** |
| **6206** | **No system header record found in input file** |
| **6208** | **Error writing to TPARAM table** |
| **6209** | **Error opening input file** |
| **6211** | **Fatal error writing system record** |
| **6212** | **Fatal error writing library record** |
| **6213** | **Fatal error writing module record** |
| **6218** | **Input file looks like a usage data file** |
| **6219** | **Input file looks like a hardware data file** |
| **6220** | **Index missing error** |
| **6221** | **Vendor product version table processing error** |
| **6222** | **Tagged module key table processing error** |
| **6223** | **Error encountered when retrieving the inventory ID** |
| **6224** | **Error encountered when retrieving the current GKB version** |
| **6225** | **Error encountered when retrieving the inventory GKB version** |
| **6237** | **Inquisitor Import table does not exist or is a missing a column** |
| **6238** | **Inquisitor Import table does not exist** |
| **6239** | **Inquisitor Import table appears to be an old version** |
| **6240** | **Error updating fGPassLibID record** |
| **6241** | **Error deleting empty libraries** |

| | |
|---|---|
| 6244 | Error assigning package information to TMODULE records |
| 6260 | Nothing to import, as no module records were found in IQ file |
| 6400 | Knowledge Base type is incorrect |
| 6402 | Failure in initializing IQ tables |
| 6403 | IQ TMODULE open error |
| 6404 | IQ TMODULE index error |
| 6405 | IQ database is empty |
| 6409 | TDECISION table open error |
| 6413 | Error creating scorecard tables for Match Engine |
| 6417 | GKB table is empty |
| 6428 | Local KB TRULES table open error |
| 6434 | Failure to open archive file |
| 6435 | Error creating index |
| 6436 | Error setting current index |
| 6437 | Search KB phase error |
| 6438 | Volume serial library phase error |
| 6439 | Inter Library phase error |
| 6440 | Rules processing phase error |
| 6444 | LPA phase error |
| 6448 | Error while clearing LMOD count |
| 6449 | TDECISION Table is missing FDECRPTION and/or FCATEGORY fields |
| 6450 | GKB TPRODUCT record seek error |
| 6451 | LKB TPRODUCT record seek error |
| 6452 | TDECISION record edit error |
| 6453 | KB TVERSION record access error |
| 6454 | KB TPRODUCT record access error |
| 6455 | KB TVENDOR record access error |
| 6600 | Match Engine tables TDECISION and/or TMIGREPORT are missing |
| 6619 | Error opening TPACKAGE table |
| 6620 | Repository table initialization failed |
| 6621 | Failure opening IQ table |
| 6622 | Unable to access GKB TVERSION table |
| 6623 | IQ TMODULE table is empty |
| 6624 | Predecessor inventory ID key does not exist |
| 6625 | Repository is not enabled for Unix System Services |
| 6626 | Repository must be enabled for Unix System Services, when the REPLACE option is in effect |
| 6627 | SYSPLEX ID mismatch in inventory record |
| 6628 | SMFID mismatch in inventory record |
| 6629 | Inventory ID key of zero is not valid |
| 6630 | Error in deleting library record |
| 6632 | Error transferring TLIBRARY information from IQ to Repository |

| | |
|---|---|
| 6633 | Error accessing TINVCTL table |
| 6634 | Mismatch found between the TINVCTL record flag and the REPLACE option |
| 6635 | Error updating FMODCNT field in TLIBRARY and TPOVLIB tables |
| 6636 | Product version key error |
| 6637 | Module key error |
| 6639 | Error updating FINVID18 fields in TUIMPORTCTRL table |
| 6640 | Error updating FINVID field in TINVREG table |
| 6641 | Error updating FINVID field in TINVREG table |
| 6642 | Error updating summary tables |
| 6643 | Error querying table in FMODID order |
| 6645 | Error marking TLIBRARY, TMODULETPOVLIB and TPOVINV records as deleted |
| 6647 | Repository type does not match IQ type |
| 6648 | When using a Continuous Inventory, an Inventory Name must be specified |
| 6666 | Error when accessing the TLIBSYS table |
| 6800 | At least one repository failed during initialization |
| 6802 | No matching LPAR found in table |
| 6803 | Primary Inventory ID set to 0 for LPAR |
| 6804 | Error trying to find FMODID or FLIBID |
| 6805 | Inventory ID does not exist |
| 6806 | Unable to find or create TLPAR record for LPAR |
| 6807 | Error trying to find or create Job or User entry |
| 6808 | Error writing MTD record |
| 6809 | Error updating summary tables |
| 6810 | Error adding TUSELIBRARY record |
| 6811 | TLIBRARY update error |
| 6812 | Summary table error |
| 6813 | Error reading import control record |
| 6814 | User initiated stop |
| 7000 | At least one table failed initialization |
| 7002 | Invalid usage summary parameters |
| 7003 | Invalid month in usage summary parameter |
| 7004 | Date order error |
| 7005 | TMODULE record seek error |
| 7011 | Error inserting record into TMODULE table |
| 7013 | TJOBDATA record seek error |
| 7014 | TJOBDATA record add error |
| 7015 | TUSERDATA record seek error |
| 7016 | TUSERDATA record add error |
| 7017 | TUSEMTD record seek error |
| 7018 | TUSEMTD record add error |

| | |
|---|---|
| 7019 | TUSEMTD record edit error |
| 7020 | TUSEMTD record delete error |
| 7021 | TPOVINV record seek error |
| 7022 | TPERIODS record seek error |
| 7023 | TPERIODS record add error |
| 7024 | TPERIODS record edit error |
| 7025 | TUSEPOVLIB record seek error |
| 7026 | TUSEPOVLIB record add error |
| 7027 | TUSEPOVLIB record edit error |
| 7028 | TUSEPOV record seek error |
| 7029 | TUSEPOV record add error |
| 7030 | TUSEPOV record edit error |
| 7034 | TUSEMTD critical failure |
| 7035 | TUSEMTD error updating record with zero FMTDID |
| 7036 | TVERSION record seek error |
| 7037 | TUSEPO record seek error |
| 7038 | TUSEPO record seek error |
| 7039 | TUSEPO record edit error |
| 7040 | TUSEPO record delete error |
| 7043 | TMODULE record edit error |
| 7044 | TUSEPOVLIB record delete error |
| 7045 | TUSEPOV record delete error |
| 7046 | TPERIODS record delete error |
| 7051 | TUSELIB record delete error |
| 7052 | IDS_USUM_TUSELIB_ AUTONUM_ERROR |
| 7055 | TLPAR record edit error |
| 7056 | TUSELIB record seek error |
| 7057 | TUSELIB record add error |
| 7058 | TPOVLIB record seek error |
| 7060 | TLPAR record seek error |
| 7061 | Join record seek error |
| 7062 | TLIBRARY record edit error |
| 7063 | TLIBRARY record seek error |
| 7065 | Invalid SUMBY value |
| 7066 | Date formatting error |
| 7067 | Usage Summary schema is empty |
| 7068 | PRODUCT_USE delete error |
| 7069 | PRODUCT_USE_DETAIL delete error |
| 7201 | Inventory to be deleted does not exist in repository |
| 7203 | TLIBRARY record delete failure |
| 7204 | TPOVINV record delete failure |
| 7205 | TPERIODS record delete failure |
| 7206 | TLPAR record delete failure |
| 7207 | TUIMPCTRL record delete failure |

| | |
|---|---|
| **7208** | **Failure updating Delete Inventory ID record** |
| **7209** | **Failure deleting TINVCTL records of deleted inventory** |
| **7210** | **Error scanning product version** |
| **7211** | **Error reassigning predecessor links in successor InvCTL records** |
| **7600** | **Table initialization failure** |
| **7601** | **Destination repository column size failure** |
| **7602** | **TINVCTL record seek error** |
| **7603** | **TINVCTL record edit error** |
| **7604** | **TINVCTL record add error** |
| **7605** | **TINVCTL record delete error** |
| **7606** | **TLIBRARY record seek error** |
| **7607** | **TLIBRARY record edit error** |
| **7608** | **TLIBRARY record add error** |
| **7609** | **TLIBRARY record delete error** |
| **7610** | **Transfer product version join seek error** |
| **7611** | **TPOVLIB record seek error** |
| **7612** | **TPOVLIB record edit error** |
| **7613** | **TPOVLIB record add error** |
| **7614** | **TPOVLIB record delete error** |
| **7615** | **TPOVINV record seek error** |
| **7616** | **TPOVINV record edit error** |
| **7617** | **TPOVINV record add error** |
| **7618** | **TPOVINV record delete error** |
| **7619** | **Table TINVPOV failed in initialization** |
| **7620** | **TVERSION record seek error** |
| **7621** | **TVERSION record edit error** |
| **7622** | **TVERSION record add error** |
| **7623** | **TVERSION record delete error** |
| **7624** | **Table TVERSION open failed** |
| **7625** | **TPRODUCT record seek error** |
| **7626** | **TPRODUCT record edit error** |
| **7627** | **TPRODUCT record add error** |
| **7628** | **TPRODUCT record delete error** |
| **7629** | **TPRODUCT open error** |
| **7630** | **TVENDOR record seek error** |
| **7631** | **TVENDOR record edit error** |
| **7632** | **TVENDOR record add error** |
| **7633** | **TVENDOR record delete error** |
| **7634** | **TVENDOR open error** |
| **7635** | **TMODULE record seek error** |
| **7636** | **TMODULE record edit error** |
| **7637** | **TMODULE record add error** |
| **7638** | **TMODULE record delete error** |

| | |
|---|---|
| 7639 | TREGCLASS record seek error |
| 7640 | TREGCLASS record edit error |
| 7641 | TREGCLASS record add error |
| 7642 | TREGCLASS record delete error |
| 7643 | TREGION record seek error |
| 7644 | TREGION record edit error |
| 7645 | TREGION record add error |
| 7646 | TREGION record delete error |
| 7647 | TREGLEAF record seek error |
| 7648 | TREGLEAF record edit error |
| 7649 | TREGLEAF record add error |
| 7650 | TREGLEAF record delete error |
| 7651 | TINVREG record seek error |
| 7652 | TINVREG record edit error |
| 7653 | TINVREG record add error |
| 7654 | TINVREG record delete error |
| 7655 | TJOBDATA record seek error |
| 7656 | TJOBDATA record edit error |
| 7657 | TJOBDATA record add error |
| 7658 | TJOBDATA record delete error |
| 7659 | TUSERDATA record seek error |
| 7660 | TUSERDATA record edit error |
| 7661 | TUSERDATA record add error |
| 7662 | TUSERDATA record delete error |
| 7663 | TLPAR record seek error |
| 7664 | TLPAR record edit error |
| 7665 | TLPAR record add error |
| 7666 | TLPAR record delete error |
| 7667 | TUSEMTD record seek error |
| 7668 | TUSEMTD record edit error |
| 7669 | TUSEMTD record add error |
| 7670 | TUSEMTD record delete error |
| 7671 | TUSELIB record seek error |
| 7672 | TUSELIB record edit error |
| 7673 | TUSELIB record add error |
| 7674 | TUSELIB record delete error |
| 7675 | TPERIODS record seek error |
| 7676 | TPERIODS record edit error |
| 7677 | TPERIODS record add error |
| 7678 | TPERIODS record delete error |
| 7679 | TUSEPOVLIB record seek error |
| 7680 | TUSEPOVLIB record edit error |
| 7681 | TUSEPOVLIB record add error |
| 7682 | TUSEPOVLIB record delete error |

| | |
|---|---|
| **7683** | **TUSEPOVLIB open error** |
| **7684** | **TUSEPOV record seek error** |
| **7685** | **TUSEPOV record edit error** |
| **7686** | **TUSEPOV record add error** |
| **7687** | **TUSEPOV record delete error** |
| **7688** | **TUSEPOV open error** |
| **7689** | **TUSEPO record seek error** |
| **7690** | **IDS_MRGE_TUSEPO_EDIT_ERROR** |
| **7691** | **TUSEPO record add error** |
| **7692** | **TUSEPO record delete error** |
| **7693** | **TUSEPO open error** |
| **7694** | **TUIMPORTCTRL record seek error** |
| **7695** | **TUIMPORTCTRL record edit error** |
| **7696** | **TUIMPORTCTRL record add error 7697** |
| **7697** | **TUIMPORTCTRL record delete error** |
| **7698** | **Source and destination repositories are not the same type** |
| **7699** | **Source and/or Destination Repositories are not the correct category database** |

# Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan, Ltd.
19-21, Nihonbashi-Hakozakicho, Chuo-ku
Tokyo 103-8510, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those websites. The materials at those websites are not part of the materials for this IBM product and use of those websites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
2Z4A/101
11400 Burnet Road
Austin, TX 78758 U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. The sample programs are provided "AS IS", without warranty of any kind. IBM shall not be liable for any damages arising out of your use of the sample programs.

## Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at “Copyright and trademark information” at www.ibm.com/legal/copytrade.shtml.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

Microsoft, Windows, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Printed in USA

SC22-5474-00