HPSS Conversion Guide

High Performance Storage System, version 9.2.0.0.0, 10 May 2021

HPSS Conversion Guide

High Performance Storage System, version 9.2.0.0.0, 10 May 2021

Table of Contents

List of Figures

List of Tables

About this Book. The HPSS Conversion Guide is for use when upgrading HPSS. It will guide system administrators through the planning and execution of upgrading an HPSS system from their current version to their target version.

Chapter 1 gives an overview of the conversion process.

Chapter 2 provides a process for converting from HPSS 6.2 to HPSS 7.1.0.

Chapter 3 provides a process for converting from HPSS 7.1.0 to HPSS 7.1.2.

Chapter 4 provides a process for converting from HPSS 7.1.2 to HPSS 7.3.1.

Chapter 5 provides a process for converting from HPSS 7.3.1 to HPSS 7.3.2.

Chapter 6 provides a process for converting from HPSS 7.3.2 to HPSS 7.3.3.

Chapter 7 provides a process for converting from HPSS 7.3.3 to HPSS 7.4.1.

Chapter 8 provides a process for converting from HPSS 7.4.1 to HPSS 7.4.2.

Chapter 9 provides a process for converting HPSS from 7.4.2 to HPSS 7.4.3.

Chapter 10 provides a process for converting HPSS between patches 1 and 2 within HPSS 7.4.3.

Chapter 11 provides a process for converting HPSS between patches 2 and 3 within HPSS 7.4.3.

Chapter 12 provides a process for converting HPSS from a big-endian to little-endian system.

Chapter 13 gives HPSS-specific instructions on upgrading DB2, AIX, and Linux, and current recommendations for the metadata layout of HPSS.

Chapter 14 provides troubleshooting procedures for problems administrators may encounter.

In addition, a glossary of terminology and abbreviations is provided.

Each conversion process is broken up into three sections:

1. The first section provides general information about the conversion process.

2. The second section provides information about which metadata changes are required and guidance.

3. The third section provides the conversion steps.

Conventions Used in This Book. Example commands that should be typed at a command line will be proceeded by a percent sign ("%") if the command may be executed as a user other than the local superuser root and be presented in a courier font:

```
% sample command
```

or with a pound sign ("#") if the command must be executed as the local superuser **root**:

```
# sample command
```

Example command output and example contents of ASCII files will be presented in a courier font without a prompt:

```
sample file line 1

sample file line 2
```

In contexts other than providing command execution instruction, any text preceded by a pound sign ("#") should be considered a comment:

```
# This is a comment
```

Chapter 1. Conversion Overview

1.1. Introduction

This conversion guide provides background and the necessary upgrade steps to upgrade HPSS from any *supported* version up to the latest HPSS version. The functionality to perform each of the metadata upgrades/conversions is included in the target version of HPSS.

As HPSS software versions become unsupported, they will be removed from this guide so long as they are not required for any supported conversion path.

In order to upgrade from a version several releases back, it is **necessary** to perform each metadata upgrade/conversion one level at a time. For example, to upgrade from 6.2 to 7.4.2, one **must** execute the metadata upgrade/conversions from:

- 6.2 to 7.1.0
- 7.1.0 to 7.1.2
- 7.1.2 to 7.3.1
- 7.3.1 to 7.3.2
- 7.3.2 to 7.3.3
- 7.3.3 to 7.4.1
- 7.4.1 to 7.4.2
- 7.4.2 to 7.4.3
- 7.4.3 (7.4.3p1 to subsequent 7.4.3 patches).

This ensures that metadata changes are applied in the proper order, enabling compatibility with the installed HPSS 7.4.2 version. The conversion guides use a variety of scripts and tools which are included with HPSS. Most may be found either in `bin/`, `tools/convert71`, or `tools/metadata/db2`.

1.2. Contents of This Guide

The following HPSS conversions are provided in this guide:

- 6.2 to 7.1.0
- 7.1.0 to 7.1.2
- 7.1.2 to 7.3.1
- 7.3.1 to 7.3.2

- 7.3.2 to 7.3.3
- 7.3.3 to 7.4.1
- 7.4.1 to 7.4.2
- 7.4.2 to 7.4.3
- 7.4.3 (7.4.3p1 to subsequent 7.4.3 patches)

1.3. Preparation

The following steps are recommended prior to performing any HPSS upgrade:

1.3.1. Gather Data On Existing System

Gather information on the existing system using the **lshpss**, **dump_sspvs**, and **lsdb2** tools, and save the results. These can be used later to validate the conversion.

```
% lshpss -all
% dump_sspvs -i#         (for each subsystem)
% dump_sspvs -i# -d      (for each subsystem)
% lsdb2             (run as hpssdb)
```

1.3.2. Benchmark Existing System

If desired, it may be beneficial to benchmark the existing system prior to upgrading so that it may be compared with the performance of the upgraded system. This allows a site to determine whether any performance degradation has taken place and provides support representatives with information to help them determine the cause.

1.3.3. Backup Existing System

Back up the existing database tables using a full backup.

```
% db2 backup db hcfg to <backup_dir>
% db2 backup db hsubsys1 to <backup_dir>
```

Note: All subsystem databases should be backed up if multiple subsystems are configured.

Now **cd** to the backup dir and run:

```
% db2ckbkpd
```

All images should be successfully verified.

1.4. hpss_managetables

This program is used by **mkhpss** to create database tables and indexes when new systems are created. It is designed to be operated by both **mkhpss** and by an HPSS administrator performing conversions. It is used in a number of conversions within this guide.

It is vitally important to understand that this program, when used from a command line by an administrator, has the ability to destroy an HPSS system. It can create and delete any database table in an HPSS system. It can create and delete any index, view or referential integrity constraint on any database table. It should also be understood that this program alone is the source of the HPSS database schema. Therefore, it will be necessary to use this program from time to time as changes to the schema are required.

hpss_managetables is a keyboard interactive program that operates in a "manual commit" mode. This means that database transactions are committed manually from the keyboard, not automatically. The administrator uses the program to make a connection to the database, then issues commands that make changes to the database. When the connection is made, the program creates an empty database transaction. When any command is issued that makes a change to the database, the change is contained in the transaction and remains uncommitted. Additional commands can be issued which are added to the transaction. The transaction remains uncommitted until a "commit" command is entered at which time the changes are committed to the database. If an "abort" command is entered instead, the changes are rolled back and no net change takes place. If an uncommitted transaction exists when the "quit" command is entered, the program reports the existence of the transaction and will not quit.

When performing conversions, even through multiple releases, use the **hpss_managetables** for the final target release. For example, if upgrading from 7.1.0 to 7.3.2, use the **hpss_managetables** for 7.3.2. By doing this you’ll ensure that the metadata updates you are making are as current as possible.

If you follow the instructions carefully and refrain from experimenting with **hpss_managetables**, you won’t do any damage to your system. If something unexpected happens, use the "abort" command and "quit" and seek advice from HPSS support.

hpss_managetables must be run by a user ID that has DBADM authority on the databases. We recommend running as "hpssdb". The program can be found in the HPSS `bin` directory.

hpss_managetables recognizes two environment variables, **HPSS_MM_SCHEMA_NAME** and **HPSS_GRP_NAME_SERVER**. Both of these variables have appropriate defaults set by HPSS. **HPSS_MM_SCHEMA_NAME** is set to "HPSS" and **HPSS_GRP_NAME_SERVER** is set to "hpsssrvr". If for some reason these values don’t apply to your system, adjust them in the environment in which **hpss_managetables** will be run, or in the HPSS `env.conf` file before starting the program.

More information on **hpss_managetables** can be found in the **hpss_managetables** man page (`man/hpss_managetables.7`).

Chapter 2. HPSS 6.2 to HPSS 7.1.0 Conversion

This chapter summarizes HPSS conversion from HPSS 6.2 to HPSS 7.1.0.

2.1. General Information

The HPSS 6.2 to 7.1 upgrade allows sites to migrate from either version 6.2 or version 6.2.2 to version 7.1.0. No special steps are taken to differentiate between 6.2 or 6.2.2 except when upgrading the version of DB2 after completing the metadata conversion.

The metadata conversion creates and uses new HPSS tables so that the version 6.2 tables remain unperturbed. The conversion programs use DB2 replication to keep the HPSS 6.2 and 7.1.0 tables synchronized. This will enable a site to continue to run the HPSS 6.2 system during metadata transformations and allow reversion back to HPSS 6.2 quickly.

Administrators should read this entire section before taking any action. The HPSS 6.2 to 7.1.0 conversion is complex, even though the majority of the steps are automated.

A metadata conversion program **hpss_convert_71** automates the conversion process.

2.2. DB2 Replication Overview

DB2 replication capabilities are numerous, but this section will focus on the specific capabilities that pertain to the 6.2 to 7.1 conversion process.

The HPSS configuration database, normally known as HCFG or CFG, does not use replication to make any metadata transformations from HPSS 6.2 to 7.1.

The HPSS subsystem databases use replication to transform metadata into the new HPSS 7.1 format and keep metadata synchronized from the HPSS 6.2 to 7.1 tables.

The replication process starts with configuring the replication environment. The first step is to create the capture and apply control tables. These tables remain small throughout the replication process and enable DB2 to maintain consistent state during replication while updates continue to occur to the replicated tables. The conversion scripts create new tablespaces using files as database managed (DMS) containers that reside in `/var/hpss/convert/7.1`. Replication control tables are created in each HPSS subsystem database in a new schema called ASN.

The conversion scripts create a 7.1 table identical to the 6.2 table being replicated. The HPSS 6.2 table is known to replication as the source table and the HPSS 7.1 table is known as the target table. Specific to the conversion, the target table is created in the same tablespace as the source table resides when running the conversion programs. This allows a site to end up with an identical database configuration with respect to tables and which tablespaces they reside in and the containers and buffer pools they use.

The source tables are then registered to DB2 as sources by altering the table definition to start capturing changes. Replication also requires a change data (CD) table for each table being replicated. The conversion creates the CD table for each source table in `/var/hpss/convert/7.1` using a file as a database managed (DMS) container separate from other containers or tablespaces previously mentioned. It is important to note that the size of these tables will increase based on the number of updates occurring to the source table within the selected 5 minute replication interval. The tablespace for each CD table will automatically increase by 10% as space allows.

Replication subscription sets allow DB2 to understand how source tables map to target tables. The conversions create subscription sets and add the target tables as members of each subscription set. At this point, DB2 knows that the site wants to replicate table A to table B and that table A will record changes that occur once the subscription set is activated.

When the subscription set is activated and the capture program is started the first time, the capture control tables record the current log sequence number (LSN) of the last committed transaction. Replication and the conversion scripts use the **ASNLOAD** utility to perform a LOAD operation reading from the source table and loading into the target table; this is called a full refresh. During this operation, HPSS 6.2 may continue to run normally but should expect significant performance degradation. Initial testing showed up to 40% degradation in metadata heavy operations. Database tuning and customizing options to the **ASNLOAD** program can improve performance of replication. Sites should also be aware that the **LOAD** and **ASNLOAD** utilities compete for database resources (memory, CPU and disk) and heavily impact database performance for both normal operation and specifically during backups. To limit the backup and load utility’s impact on database operations, set the **UTIL_IMPACT_LIM** environment variable to 50 or some number lower than its default 100 setting. **UTIL_IMPACT_LIM** is a maximum percentage of database resources that should be used by load and backup utilities. Set it as you would any other environment variable, for example:

```
% export UTIL_IMPACT_LIM=50
```

Once the full refresh completes, the capture program will perform work on a designated interval of 5 minutes. The capture program will read DB2 subsystem database logs starting with the LSN of the last committed transaction that it recorded from completing the last processing cycle and insert records into the change data table for the source table describing the transaction (an update, delete, insert). At some point, the capture program will again record the LSN of the transaction it stopped processing and rest for 5 minutes. This process continues until either the capture program experiences a problem, is disabled (i.e. deactivating the subscription set), or is shutdown. An important point here is that the capture program requires DB2 logs to work and log archiving could interfere with its operation in the event that it cannot keep pace with the amount of updates occurring to the database given its 5 minute interval. A utility is provided with the conversions to identify the oldest DB2 log with the LSN that the capture program requires to begin processing. The site may need to adjust the capture program’s interval as well updating it to "continuous" if necessary (contact IBM HPSS support).

The apply program is intended to simply read the change data table and perform work on its designated interval of 5 minutes. Apply performs each transaction (update, delete or insert) on the target table. After some period of time, it rests for 5 minutes and then continues processing where it left off. The apply process continues until the program is disabled or shutdown.

Once HPSS 6.2 is shutdown and the site desires the target table to be synchronized with the source table, the site should run the programs provided to verify synchronization. One of the programs uses the **ASNTDIFF** utility to identify the exact differences between the tables. It is possible to then use the **ASNTREP** utility to synchronize the tables, but the preferred method would be to simply startup the capture and apply programs with HPSS 6.2 down and allow the programs to process at least two capture and apply cycles (10 minutes) and then rerun the synchronization verification utilities provided with the conversions.

2.3. Planning

2.3.1. Estimating Disk Space Required

Tablespaces should be at least 50% free prior to starting the conversion.

2.3.2. Estimating Down Time Required

In considering the amount of time required to complete an upgrade from beginning to end, the single largest factors are the number of records in the ABSADDR, BITFILE, STORAGESEGTAPE, STORAGESEGDISK, and NSOBJECT tables and the capabilities of the hardware available. The downtime window where the HPSS 6.2 system is unavailable during the upgrade may be reduced to the time it takes to:

1. (optional) Run HPSS benchmarks - time the speed of 6.2 system prior to upgrade.

2. Shut down HPSS 6.2 - to prevent further changes to the 6.2 tables.

3. Stop the replication programs - stops the capture and apply programs for DB2 replication. Changes to HPSS 6.2 are no longer propagated to the 7.1.0 tables.

4. Ensure replication to target tables completed - run DB2’s RUNSTATS utilities on new tables, run the provided replication verification program to compare 6.2 to 7.1.0 tables. If rows are identified as missing with the quick running verify program, then run the diff program provided to save details of differences and for possible resolution.

5. Perform the post replication upgrade steps - copy data from 6.2 to 7.1.0 table for the following tables in the configuration database: CORE, COS, GLOBAL, PVLACTIVITY, PVLDRIVE, PVLPV. Replication does not replicate the configuration database.

6. Perform the pre-7.1.0 upgrade steps - swap the 6.2 tables with 7.1.0 tables.

7. It is important to run the provided resolve program at this point to rectify any metadata inconsistencies that may exist between the tape virtual volume, storage segment, and absolute address tables. This is essential to have a working system because of new SQL join logic that the Core Server relies on in the metadata manager library for consistent metadata queries.

8. Complete any other steps to save HPSS 6.2 configuration

9. Startup HPSS 7.1.0

10. (optional) Run HPSS benchmarks - to time speed of new system.

2.4. Preparation

2.4.1. Installing Perl DBI

The programs make use of the Perl Database Interface (DBI) and Database Driver (DBD) for DB2. The DBI software can be obtained from:

http://dbi.perl.org/

The easiest way to install DBI is by using CPAN. CPAN will obtain necessary prerequisite packages. Users without experience using CPAN may want to stick with the manual installation of the DBI package and its prerequisites.

http://cpan.perl.org/

Software is available for Linux and AIX.

2.4.2. Installing DB2 DBD

The DBD software can be obtained by following instructions at:

http://www-306.ibm.com/software/data/db2/perl/

After downloading and installing the DBI and DBD software, the conversion scripts need to be executed in a given order and must run without error before proceeding to the next step. The scripts should not be run individually unless directed by IBM Support to do so. Follow the process described in the *Upgrading Section 2.6, “Upgrading to HPSS Release 7.1.0”* section.

After installing DBI and DBD, it may be necessary to set the **LD_LIBRARY_PATH** environment variable to point at the DB2 library. The DB2 library (libDB2) will be located in your DB2 installation path.

2.5. Metadata Changes

This chapter summarizes HPSS metadata changes for Release 7.1.0.

2.5.1. New Features Affecting Conversion

This section describes the new HPSS features that result in metadata transformations from HPSS Release 6.2 and 6.2.2.

2.5.1.1. Multiple Streams of Class of Service (COS) Changes

HPSS Classes of Service (COS) can now be configured to perform multiple change operations simultaneously. This is an advantage for any site that was previously limited by the single COS change thread in the Core Server. For sites upgrading, the number of COS change threads is set to one. Sites desiring multiple streams of COS changes need to use SSM to alter the COS configuration by updating the number of threads to the desired amount.

2.5.1.2. Aggregation When Migrating to Tape

The Migration and Purge Server is now capable of aggregating files during migration from disk to tape. The option is not enabled by default during the conversion. To enable the feature, administrators should alter existing Migration Policies.

2.5.1.3. Variable Length Storage Segments

Previous to HPSS 7.1.0, only fixed length storage segments were allowed. The new variable length storage segments feature enables HPSS to use the minimum and maximum storage segment size within a COS to continue doubling the storage segment size until the maximum storage segment size is reached when the size of the file is unknown to HPSS when it is created. Existing HPSS COS’s will not be modified to variable length allocations; instead, they will remain "Classic Style". However, all existing HPSS COS’s will have a new flag set to truncate the final storage segment so that only the space that HPSS requires to store the file is actually used. The truncate final segment storage class flag will be enabled by default because it has the desired side effect of providing a new allocation scheme that should help reduce fragmentation on disk. If desired, sites may override this by altering their COS’s with SSM and disabling the truncate final segment flag. There is a new configuration option for HPSS COS’s called Allocation Method. Again, all existing COS’s remain the classic fixed length storage segment allocation upon conversion. A site may choose to alter this with SSM after the conversion.

2.5.1.4. Tape Drive Pools

This feature is already in HPSS 6.2.2. It allows tape drives to be configured into various pools for altered allocation behavior. The conversion process will either convert existing drive pool metadata or initialize drive pool metadata depending upon the version being upgraded from. The conversions will not enable the feature.

2.5.1.5. New Tables, Views and Constraints

The replication programs create new tables that are target replication tables beginning with `V71_`. These tables are created in the same table spaces as the original source tables so that once they are swapped with the original (HPSS 6.2) tables, the database configuration (table to tablespace mappings) will remain the same.

The following tables are new to HPSS 7.1.0:

Table Name	**Purpose**
STORAGESEGTAPEABSADDR	Holds records from ABSADDR that previously applied to STORAGESEGTAPE records.
VVTAPEABSADDR	Holds records from ABSADDR that previously applied to VVTAPE records.
STORAGESEGDISKEXTENTS	Holds information to keep track of disk storage segment allocation.
STORAGESEGAUX	Holds additional information about storage segments that may not always be needed.

The following constraints are new to HPSS 7.1.0:

Constraint Name	**Effect**
BFTAPESEGCON1	Prevents rows in STORAGESEGTAPE from being deleted if they are referenced by any row in BFTAPESEG.
DISKSEGEXTENTSCON1	Causes rows in STORAGESEGDISKEXTENTS that are dependent on rows in STORAGESEGDISK to be deleted when the parent rows are deleted.
DISKVVMAPCON1	Causes rows in STORAGEMAPDISK that are dependent on rows in VVDISK to be deleted when the parent rows are deleted.
DISKVVPVCON1	Causes rows in SSPVDISK that are dependent on rows in VVDISK to be deleted when the parent rows are deleted.
DISKVVSEGSCON1	Prevents rows in VVDISK from being deleted if they are referenced by any row in STORAGESEGDISK.
TAPESEGABSADDRCON1	Causes rows in STORAGESEGTAPEABSADDR that are dependent on rows in STORAGESEGTAPE to be deleted when the parent rows are deleted.
TAPESEGAUXCON1	Causes rows in STORAGESEGAUX that are dependent on rows in STORAGESEGTAPE to be deleted when the parent rows are deleted.
TAPEVVAACON1	Causes rows in VVTAPEABSADDR that are dependent on rows in VVTAPE to be deleted when the parent rows are deleted.
TAPEVVMAPCON1	Causes rows in STORAGEMAPTAPE that are dependent on rows in VVTAPE to be deleted when the parent rows are deleted.
TAPEVVPVCON1	Causes rows in SSPVTAPE that are dependent on rows in VVTAPE to be deleted when the parent rows are deleted.
TAPEVVSEGSCON1	Prevents rows in VVTAPE from being deleted if they are referenced by any row in STORAGESEGTAPE.

The following are new views in HPSS 7.1.0:

View	**Purpose**
VVTAPEMAPVIEW	To provide selected metadata from the VVTAPE and STORAGEMAPTAPE tables.
STORAGESEGDISKVIEW	To provide selected metadata from the STORAGESEGDISK, STORAGEMAPDISK, and STORAGESEGDISKEXTENTS tables.
STORAGESEGTAPEVIEW	To provide selected metadata from the STORAGESEGTAPE and STORAGESEGTAPEABSADDR tables.

View	Purpose
VVDISKVIEW	To provide selected metadata from the VVDISK and SSPVDISK tables.
VVTAPEVIEW	To provide selected metadata from the VVTAPE, VVTAPEABSADDR, and SSPVTAPE tables.

2.5.2. Metadata Change Detail

For new columns, the values the conversion will set metadata to is provided in parenthesis following the new column name. In all cases, the value may be altered by navigating to the appropriate screen in SSM and updating the setting.

2.5.2.1. HPSS Configuration Database Changes (HCFG/CFG)

Table Name	Change
CORE	Add new TAPE_DISMOUNT_DELAY (15 secs), TAPE_HANDOFF_DELAY (15 secs), PVL_MAX_CONN_WAIT (300 secs), FRAGMENT_TRIM_LIMIT (64), FRAGMENT_SMALLEST_BLOCK (8) columns.
COS	Add new ALLOC_METHOD (0 - Classic) column, set flag for truncating final segment.
GLOBAL	Add new DB_LOG_MONITOR_INTERVAL (1800 secs), COS_CHANGE_THREAD_COUNT (1) columns.
MIGPOL	Add new MIN_FILES_TO_AGGR (10), MAX_FILES_TO_AGGR (500), MAX_FILESIZE_TO_AGGR (1MB), MIGRATION_OPTION_FLAGS (0 - none), MIGRATION_STREAMS_PER_FAMILY (1) columns.
PVLACTIVITY	Add DRIVE_POOL_ID (0) column for HPSS 6.2 systems. HPSS 6.2.2 systems already have this change.
PVLDRIVE	Add DRIVE_POOL_ID (0) and TEXT (empty value) columns for HPSS 6.2 systems. HPSS 6.2.2 systems already have these changes the DRIVE_POOL_ID, but not the TEXT.
PVLPV	Add TEXT (empty value) column.
STORSUBSYS	Add new DB_LOG_MONITER_INTERVAL (1800 secs), COS_CHANGE_THREAD_COUNT (0 - by default setting in GLOBAL of 1 is in effect for all subsystems) columns.
SCLASS	For existing SCLASS’s if MAX_SSEG_SIZE is not a power of 2, update MAX_SSEG_SIZE to next higher power of 2 from current setting.
FILESYSTEMS	New table no initialization needed.

2.5.2.2. HPSS Subsystem Database Changes

Table Name	Change
BFCOSCHANGE	Add STREAM_ID (0) column.
BFMIGRREC	Add MIGRATION_FAILURE_COUNT (0) column.

Table Name	**Change**
BFDISKALLOCREC	Normalize table from one record/row containing up to 8 regions to a single row/record per region.
BITFILE	Add ALLOC_METHOD (0), MIN_SEG_SIZE (0) columns.
BFSSUNLINK	New table definition, but will ensure table is empty so no metadata transformation.
NSOBJECT	Remove obsolete columns DM_HANDLE_LENGTH, VERSION, MAC_LABEL, TIME_BILLED, TIME_DELETED, TIME_EXPIRED, TIME_TRASHED, ENTRY_COUNT, FAMILY_ID, DM_HANDLE. Drop primary key (forward scan index) and define new index to allow reverse scan as primary key.
SSPVDISK	Remove obsolete columns CREATION, UPDATE, LAST_MAINT, IN_SERVICE columns.
SSPVTAPE	Remove obsolete columns MOUNT_COUNT_SINCE_SERVICE, MOUNT_COUNT_SINCE_MAINT, CREATION, UPDATE, LAST_WRITE, LAST_READ, LAST_MAINT, IN_SERVICE, FAMILY_ID columns.
STORAGESEGDISK	Remove columns SCLASS_ID, NUM_READS, NUM_WRITES, ADMIN_STATE, SS_STATE, BLOCK_SIZE, WRITTEN_LENGTH, LAST_READ, LAST_WRITE.
STORAGESEGTAPE	Remove columns ADMIN_STATE, ALLOCATED_LENGTH, OWNER.
STORAGEMAPTAPE	Remove EST_SIZE column, and add NUMBER_OF_FILES (0), NUMBER_OF_AGGREGATES (0), BYTES_IN_AGGREGATE (0) columns.

2.5.2.3. Database Configuration Changes

There are several new features of DB2 v9.5 that are enabled in the conversions. Any default buffer pools that are named IBMDEFAULTBP are altered to AUTOMATIC memory allocation/sizing. All eligible tablespaces are modified to LARGE tablespaces. The conversions enable AUTOMATIC memory management. For details on these changes, see the *DB2 v9.5 Administration Guide*.

2.6. Upgrading to HPSS Release 7.1.0

This chapter is only intended for sites upgrading HPSS from version 6.2 to version 7.1.0. Sites wishing to upgrade from prior HPSS releases must first upgrade to HPSS 6.2. Sites that are going to install and configure HPSS 7.1.0 from scratch will not need to perform the upgrade. The upgrade procedures in this chapter are intended only for upgrading HPSS systems on the same operating system.

⚠ *Due to possible risk of losing HPSS metadata, we strongly recommend that the upgrade procedure be planned and performed with the help of your HPSS customer support representative.*

⚠ *There are special considerations not documented here in upgrading a HPSS 6.2 system across operating system architectures. Contact your HPSS customer support representative.*

The upgrade to HPSS 7.1 begins with completing the metadata conversion process. The programs listed in this section perform work necessary to complete the metadata conversion process.

2.6.1. Conversion File System Space and Output

All output is directed to either stdout or stderr. Any detailed output for upgrade steps can be found in `/var/hpss/convert/7.1` on the machine running the upgrade programs.

The `/var/hpss/convert/7.1` file system should not be allowed to fill up as it will cause the conversions significant problems and may require a complete redo of the conversion process by throwing away the replicated tables and starting over. The file system is known to DB2 replication as the apply_path and the capture_path.

Prior to starting replication, ensure the provided `asnload.ini` file is placed in `/var/hpss/convert/7.1` with permissions set allowing the HPSS instance owner to read it. This file disables the recoverable option of the conversion load operation. When enabled, an unnecessary copy of the loaded data is made requiring many GBs of extra space in `/var/hpss/convert/7.1`.

It is highly advisable that each site make the `/var/hpss/convert/7.1` path its own file system to ensure that `/var/hpss`, which is normally vital to a running HPSS system, does not fill up.

With the `asnload.ini` in place, the conversions will require significantly less free space in `/var/hpss/convert/7.1`.

Finally, all change data and capture control tables are created in `/var/hpss/convert/7.1`. These tables will initially require about 25MB of space as DMS file containers. They are set to resize automatically increasing 10% each time more space is needed. On a production sized system, these tables can be expected to grow to require several GBs of free space during replication. With the recoverable option disabled, a system with a minimal amount of metadata completed replication in about 2 hours and used about 100MB of space; a system with production metadata (12 million bitfiles) completed in 7 hours and used about 200MB of space.

2.6.2. Conversion Requirements and Limitations

The conversion steps in the **hpss_convert_71** program need to be run in order. Each step must complete successfully for the subsequent step to work correctly. If a problem occurs, see the Troubleshooting section later in this document.

2.6.2.1. BFSSUNLINK Table Empty

Sites converting must ensure that the BFSSUNLINK table is empty prior to shutting down HPSS 6.2 and starting the post replication steps of the conversion menu. In order to check if the table is empty prior, login as HPSS DB2 instance owner (e.g. hpssdb) and execute:

```
% db2 connect to hsubsys1
% db2 select count(*) from hpss.bfssunlink
```

Output like such would indicate that there are no rows in the table:

```
1
-----------
          0

1 record(s) selected.
```

If there are records in the table, sites need to ensure that users cannot access the HPSS system but leave the HPSS 6.2 Core Server running until the number reaches 0.

2.6.2.2. Converting With HPSS 6.2 Running

In order to run the conversions successfully while the HPSS 6.2 system is running, the site will install the conversion programs somewhere on the system and then need to set **HPSS_ROOT** in the shell they plan to run the conversions from. For example, if the HPSS 6.2 code is located in and running out of `/opt/hpss` and the site installs the HPSS 7.1.0 code in `/usr/lpp/hpss71`, they would want to set **HPSS_ROOT** prior to running the conversions as such:

```
% export HPSS_ROOT=/usr/lpp/hpss71
```

Note, the conversion tools would be expected to be located and run from `/usr/lpp/hpss71/bin/convert71` in the example provided above. So if the site chooses something other than `/usr/lpp/hpss71` or just wants to install the conversion programs rather than the entire HPSS code tree, they would want to ensure that at least the conversions are located in the `bin/convert71` subdirectory of some root directory that they set **HPSS_ROOT** to. In addition, in order for the **-h** option of the **hpss_convert_71_subsys_monitor** program to work, the site will need to set a new temporary environment variable called **HPSS_ROOT62** to the root directory of the HPSS 6.2 code. The monitor program will be looking for a `bin` subdirectory under that **HPSS_ROOT62** path for the **hpssmsg** program.

2.6.2.3. Conversion Programs

The main upgrade program is **hpss_convert_71** which presents a menu to the user. The menu options are sorted in the order that the upgrade steps should be executed.

The "undo" steps are listed last and should not be executed if replication information or data is to be retained. The undo steps correspond to menu items towards the top of the menu. Careful attention should be paid to only undo steps that have been successfully completed. For instance, a site should not execute the "Undo pre-7.1 startup steps for the cfg db" if the "Perform pre-7.1 startup steps for the cfg db" menu option has not completed successfully.

Each step may be run more than once and depending on the problem may help correct the problem. If a failure occurs during a particular step, no harm will be done by rerunning the same step.

The HPSS configuration database steps only need to be executed once for each HPSS system being upgraded. The HPSS subsystem database upgrade steps need to be executed once for each subsystem on the HPSS system.

A discussion of each step and the program(s) normally needed follows.

2.6.2.4. hpss_convert_71

This is a menu driven program that provides and executes programs to complete the steps of upgrading a system from HPSS 6.2 to HPSS 7.1.0. It will help create the new HPSS 7.1.0 tables, setup replication to copy data from HPSS 6.2 to 7.1 for the subsystem metadata conversion steps, directly copy the data from HPSS 6.2 to 7.1 for the configuration metadata, and make the final switch of HPSS 6.2 to 7.1.0 tables to prepare the system to run HPSS 7.1.0.

2.6.2.5. hpss_convert_71_log_needed

This is a program that discovers the earliest DB2 log filename required for replication to continue after replication is stopped and before it is restarted. If replication is stopped and restarted after a period of time has elapsed, depending on the amount of database activity (i.e. high DB2 log activity) and the site specific log archive method, a database log file may be archived that is required by replication once restarted. This utility will help identify the log file name of the earliest log required by the replication programs for the site to place the log in the archive log directory for access by the replication programs.

2.6.2.6. hpss_convert_71_subsys_monitor

This is a program that parses the logs produced by the various replication programs. The log files are all located in `/var/hpss/convert/7.1` and are for the replication monitor, alert notifications produced by the monitor, and the apply and capture programs. Note that most logs are truncated each time a program is run. This is due to the verbose amount of logging information. The utility should be run periodically and after reading the log, will produce a summary of the number of info, warning or error messages contained within the monitor log. The utility may be helpful in determining whether there is a need to look at the various logs in detail or not. It also has an option to submit the summary line to **hpssmsg** utility so that replication status may be viewed on SSM Alarms & Events.

2.6.2.7. hpss_convert_71_subsys_diff

A program that uses the **asntdiff** utility in DB2 to inform you when replication source and target tables match or not for HPSS subsystem database tables. This program parses the output and informs you whether the HPSS 7.1.0 tables in the subsystem database that were replicated match their HPSS 6.2 counterparts or not. The program is able to check the following replicated tables:

- V71_BFCOSCHANGE
- V71_BFDISKALLOCREC
- V71_BITFILE
- V71_NSOBJECT
- V71_SSPVDISK
- V71_SSPVTAPE
- V71_SSEGDISKEXTENTS

- V71_STORAGESEGTAPE
- V71_STORAGESEGDISK

The utility does not check the V71_SSEGTAPEABSADDR or V71_VVTAPEABSADDR tables. The utility should only be run when directed by IBM customer support for detailed problem diagnosis. The program may be run when replication has been stopped and before the post replication or pre-7.1 startup conversion steps are performed in **hpss_convert_71**. The program takes significant disk space and time to complete its checks. On a system with 11.8 million BITFILE records, the program took over 3 hours to complete. Running RUNSTATS has no effect on the time this takes to complete.

2.6.2.8. hpss_convert_71_subsys_verify

This program uses SQL to verify that every row in HPSS 6.2 tables exist in their HPSS 7.1.0 counterparts. When rows are identified in a HPSS 6.2 table that does not exist in HPSS 7.1.0, the primary key of the HPSS 6.2 record is outputted to the screen. The **hpss_convert_71_subsys_diff** program should be run if this program identifies missing rows after replication has been stopped and before any post replication or pre-7.1 startup conversion steps are performed with the hpss_convert_71 program. Running RUNSTATS does affect how quickly this can complete. On a system with 11.8 million BITFILE records and having completed RUNSTATS on the tables listed below, this program took a little over 3 minutes to complete. Prior to running the utility, the RUNSTATS program should be run on the following tables to improve query performance:

- V71_BFCOSCHANGE
- V71_BITFILE
- V71_NSOBJECT
- V71_SSPVDISK
- V71_SSPVTAPE
- V71_STORAGESEGTAPE
- V71_STORAGESEGDISK
- V71_BFDISKALLOCREC
- V71_SSEGDISKEXTENTS
- V71_SSEGTAPEABSADDR
- V71_VVTAPEABSADDR

The program is able to check the above tables, which are all the tables replicated between HPSS 6.2 and 7.1.0.

2.6.2.9. hpss_convert_71_subsys_resolve

This program uses SQL to verify that rows in HPSS 7.1.0 VVTAPE and STORAGESEGTAPE tables have corresponding rows in STORAGESEGTAPEABSADDR and VVTAPEABSADDR. It also checks and corrects inconsistencies created by the replication process for any orphaned HPSS 6.2 ABSADDR metadata. It provides sites with the option of restoring the original HPSS 6.2 values for orphaned ABSADDR metadata or deleting the useless HPSS 6.2 orphaned ABSADDR metadata. Either choice is necessary and sufficient to create consistent metadata between the HPSS 7.1.0 tables to allow the Core Server to access metadata for valid HPSS files as it could in HPSS 6.2. The program can be terminated and rerun as often as desired. The program should be run until it reports "all ok" for all conditions that it checks. Often this takes several runs. If you choose to not delete the orphaned segments, then you will not get an "all ok" for that step of the program and you do not need to rerun the program more than twice.

The **hpss_convert_71_subsys_resolve** program needs to be run after step 4 after verifying that all metadata has been converted (e.g. after running **hpss_convert_71_subsys_verify**). Once the site runs step 5 or 6, the **hpss_convert_71_subsys_resolve** program will not run properly because the tables it requires have been renamed and constraints have been put in-place necessary for HPSS 7.1.0 but that prevent the utility from correcting metadata.

2.6.3. Conversion Steps

The conversion steps intended for a subsystem database conversion should be executed once for each subsystem the HPSS system contains.

Sites should also be aware that although replication can run indefinitely, special consideration to maintaining the replication environment is not taken into account. It is assumed that each site will only replicate for a short period of time to complete the metadata transformations in immediate preparation for running HPSS 7.1.0.

If a site chooses to run for extended periods of time (several weeks to a month or more) with replication configured, they should consult the DB2 replication documentation about maintaining the replication environment. The replication environment should be treated as the production environment such that RUNSTATS, REORGS, and other database maintenance will be needed to keep performance of replication targets satisfactory as replicated tables change over time. In addition, replication change data tables and monitor tables will require pruning over time to maintain proper performance and preserve disk space.

In preparation for upgrading to HPSS 7.1.0, the site needs to consider additional disk space requirements for using DB2 replication and the output generated by the conversion programs. The amount of new disk required for replication will be more than twice the amount of disk currently required for data in the HPSS 6.2 tables listed below:

- CORE
- COS
- GLOBAL
- MIGPOL
- PVLACTIVITY
- PVLDRIVE

- PVLPV
- BFCOSCHANGE
- BFDISKALLOCREC
- BITFILE
- NSOBJECT
- SSPVDISK
- SSPVTAPE
- STORAGEMAPTAPE
- STORAGESEGDISK (new STORAGESEGDISKEXTENTS)
- STORAGESEGTAPE ABSADDR (new SSEGTAPEABSADDR)
- ABSADDR (new VVTAPEABSADDR)

All of the tables listed above are copied during the upgrade to preserve the original table format and metadata to ensure that any change may be reversed and that a site could return to HPSS 6.2 if needed. For disk space estimations for HPSS 7.1.0, determine the current size of each of the tables above and ensure that you allow for at least 50% free space within the tablespace that the tables belong to prior to conversion. Ensure that containers are expanded or that additional containers are added to the tablespace(s) prior to starting replication. Approximately 200MB of the free space needed will be consumed during the setup of replication step of the conversion as well. If the site plans to take backups of the HPSS configuration or subsystem databases during replication or conversion to HPSS 7.1.0, then additional disk space will be required to accommodate larger backup files that will result from having new tables and duplicate data in each of the databases. The backup files should increase in size by the same amount as what the tables listed above currently use. New 7.1.0 tables will be created in the same tablespace as the HPSS 6.2 table being replicated. In addition, `/var/hpss/convert/7.1` will require 50MB of free disk space while replication runs to hold alert monitor logs, and logs from capture and apply programs.

A backup of HPSS 6.2 databases prior to starting the upgrade is not absolutely necessary, but definitely advised. The HPSS 6.2 databases are not touched prior to the pre-7.1 startup steps. To reiterate, copies of the original tables are made and any alterations or conversions are made/applied to the table copies. Sites should ensure that the new 7.1.0 tables are fully synchronized with HPSS 6.2 prior to running the post-replication steps. To ensure tables are synchronized, the site should ensure HPSS 6.2 servers are shutdown and run RUNSTATS on the subsystem databases to be verified. This will ensure statistics for the new 7.1.0 HPSS tables are updated and allow the verification program to run optimally. To see that all HPSS 6.2 records are in HPSS 7.1.0 tables, run the **hpss_convert_71_subsys_verify** program. The verify program should be sufficient for a site to know that the tables are synchronized. Sites will likely want to run the **hpss_convert_71_subsys_verify** program with a random sampling option. On a site with 70 million HPSS files, the random sampling option checked 1 million rows and took 2 hrs and 15 minutes to complete. Another program is provided to perform a DB2 diff between the HPSS 6.2

and 7.1.0 tables, but is only intended to aid in problem determination or diagnosis. Execute the **hpss_convert_71_subsys_diff** program if disk, memory and time resources are not an issue. Note that the diff program is resource intensive and can take abnormally long amounts of time to complete. On a system with 70 million HPSS files, the **hpss_convert_71_subsys_diff** program took several days to complete. This is the point at which sites must run the **hpss_convert_71_subsys_resolve** program to correct any metadata issues. The **hpss_convert_71_subsys_resolve** program will not function properly once the post-replication steps are run because 7.1.0 constraints will be in-place that prevent the program from resolving metadata issues. The post-replication steps will partially unconfigure the replication environment so that table renames may occur in subsequent steps. After executing the post-replication steps, replication will not be able to be restarted (step 3). The pre-7.1 startup steps are the first time the HPSS databases are modified such that simply restarting HPSS 6.2 is not possible. HPSS 6.2 must be down and previous steps should be completed in sequence without error before executing the pre-7.1 startup steps. The pre-7.1 startup steps of the upgrade program are when HPSS 6.2 tables are renamed and the HPSS 7.1.0 tables are put in their place and the replication environment is completely unconfigured.

2.6.3.1. Capturing the Metadata Conversion Output (Step 1)

The conversion output for the **hpss_convert_71** program should be captured and retained using **script** for problem diagnosis and verification purposes. Sites need to investigate errors before continuing with conversion steps. Warnings are intended to call attention to a step that did not succeed and should be looked into but generally don’t prevent a site from moving to the next step.

Should any of the conversion programs fail, they will exit with an error. To correct the error and continue with the conversion, refer to the Troubleshooting chapter.

2.6.3.2. Setup HPSS 7.1.0 Configuration Tables (Step 1)

Set the following environment variables **HPSS_ROOT** and **HPSS_ROOT62**, then execute the **hpss_convert_71** program as the HPSS instance owner (e.g. hpssdb). **HPSS_ROOT** should be set to the root directory of the 7.1.0 HPSS code and **HPSS_ROOT62** should be set to the root directory of the 6.2 HPSS code.

To begin the upgrade process, ensure you have the provided `asnload.ini` file in `/var/hpss/convert/7.1`.

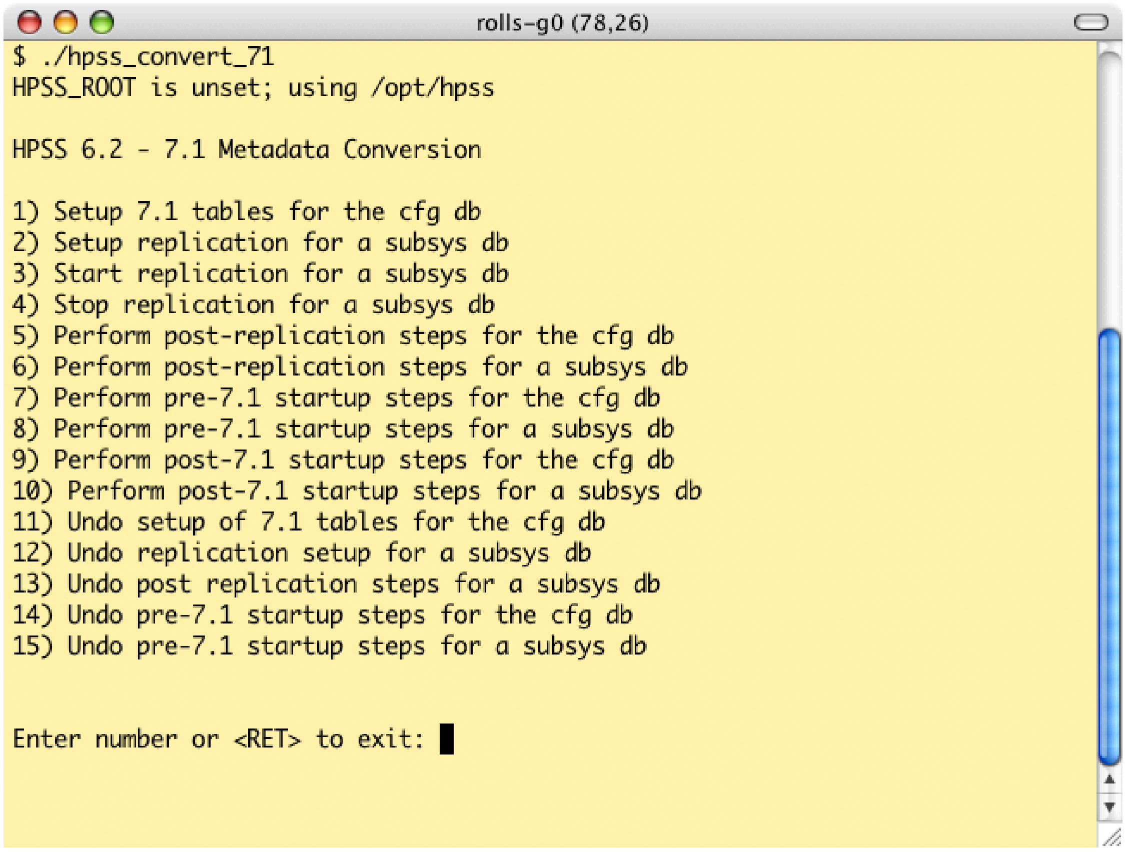
```
% cp ${HPSS_ROOT}/bin/convert71/asnload.ini /var/hpss/convert/7.1/.
```

The conversion program **hpss_convert_71** needs to be run as the DB2 instance owner (e.g. hpssdb). Login as this user and execute the following to begin:

```
% export PATH=${HPSS_ROOT}/bin/convert71:$PATH
% hpss_convert_71
```

A menu is presented to assist a site in performing most conversion steps necessary to complete the upgrade. See the figure below:

Figure 2.1. hpss_convert_71 menu

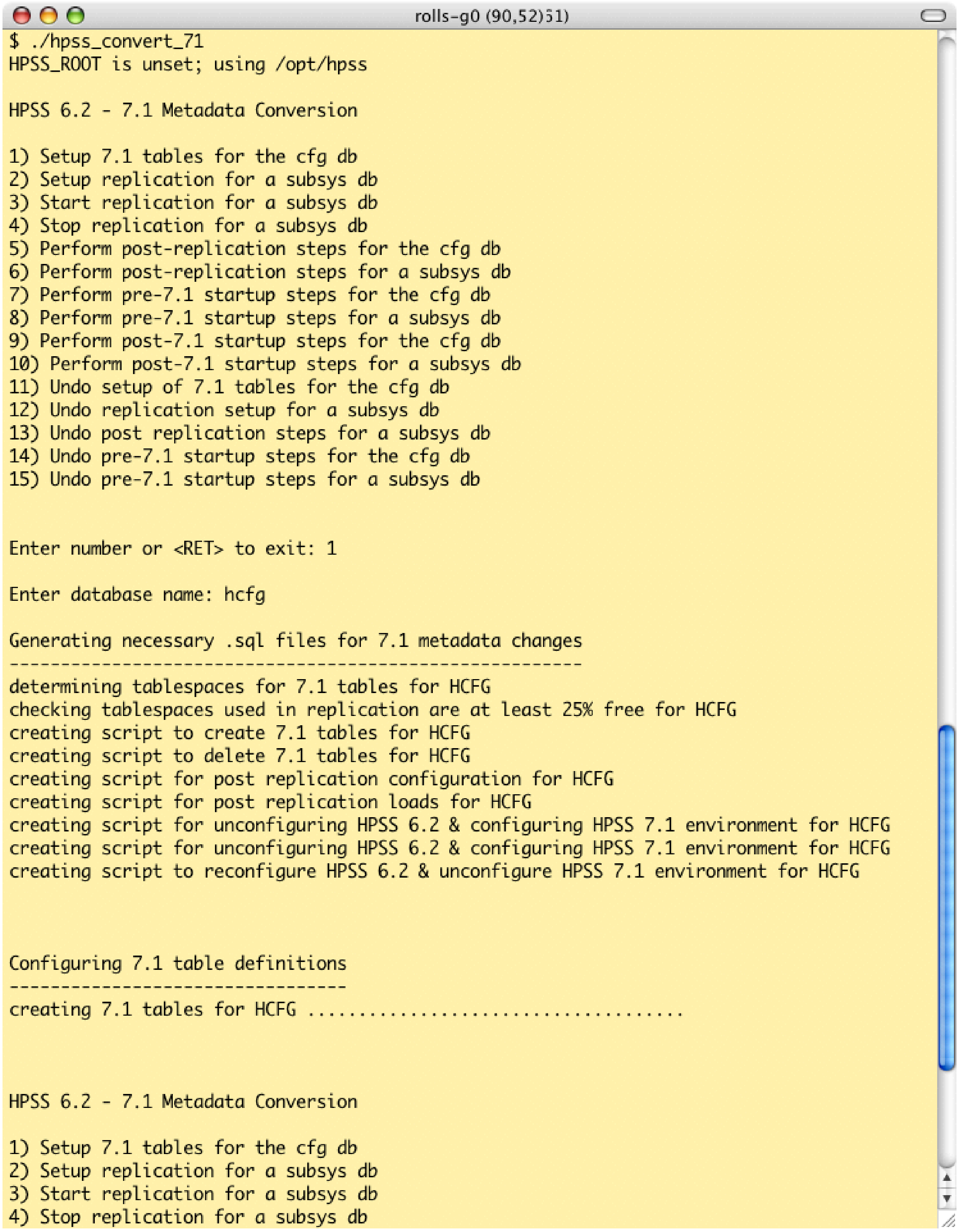


```
$ ./hpss_convert_71
HPSS_ROOT is unset; using /opt/hpss

HPSS 6.2 - 7.1 Metadata Conversion

1) Setup 7.1 tables for the cfg db
2) Setup replication for a subsys db
3) Start replication for a subsys db
4) Stop replication for a subsys db
5) Perform post-replication steps for the cfg db
6) Perform post-replication steps for a subsys db
7) Perform pre-7.1 startup steps for the cfg db
8) Perform pre-7.1 startup steps for a subsys db
9) Perform post-7.1 startup steps for the cfg db
10) Perform post-7.1 startup steps for a subsys db
11) Undo setup of 7.1 tables for the cfg db
12) Undo replication setup for a subsys db
13) Undo post replication steps for a subsys db
14) Undo pre-7.1 startup steps for the cfg db
15) Undo pre-7.1 startup steps for a subsys db


Enter number or <RET> to exit: 
```

The site should execute the steps in the order provided except for the "Undo" steps starting at step 11. Those steps should only be executed in the event a site desires to revert to HPSS 6.2.

Running step 1 will execute the **hpss_convert_71_config_prep** and **hpss_convert_71_config** programs. These will setup the HPSS 7.1.0 tables for the HPSS configuration database by creating the CORE, COS, GLOBAL, MIGPOL, PVLACTIVITY and PVLDRIVE tables in the same tablespace of each HPSS 6.2 table. Successful completion of step 1 will look like the figure below:

Figure 2.2. Successful completion of step 1

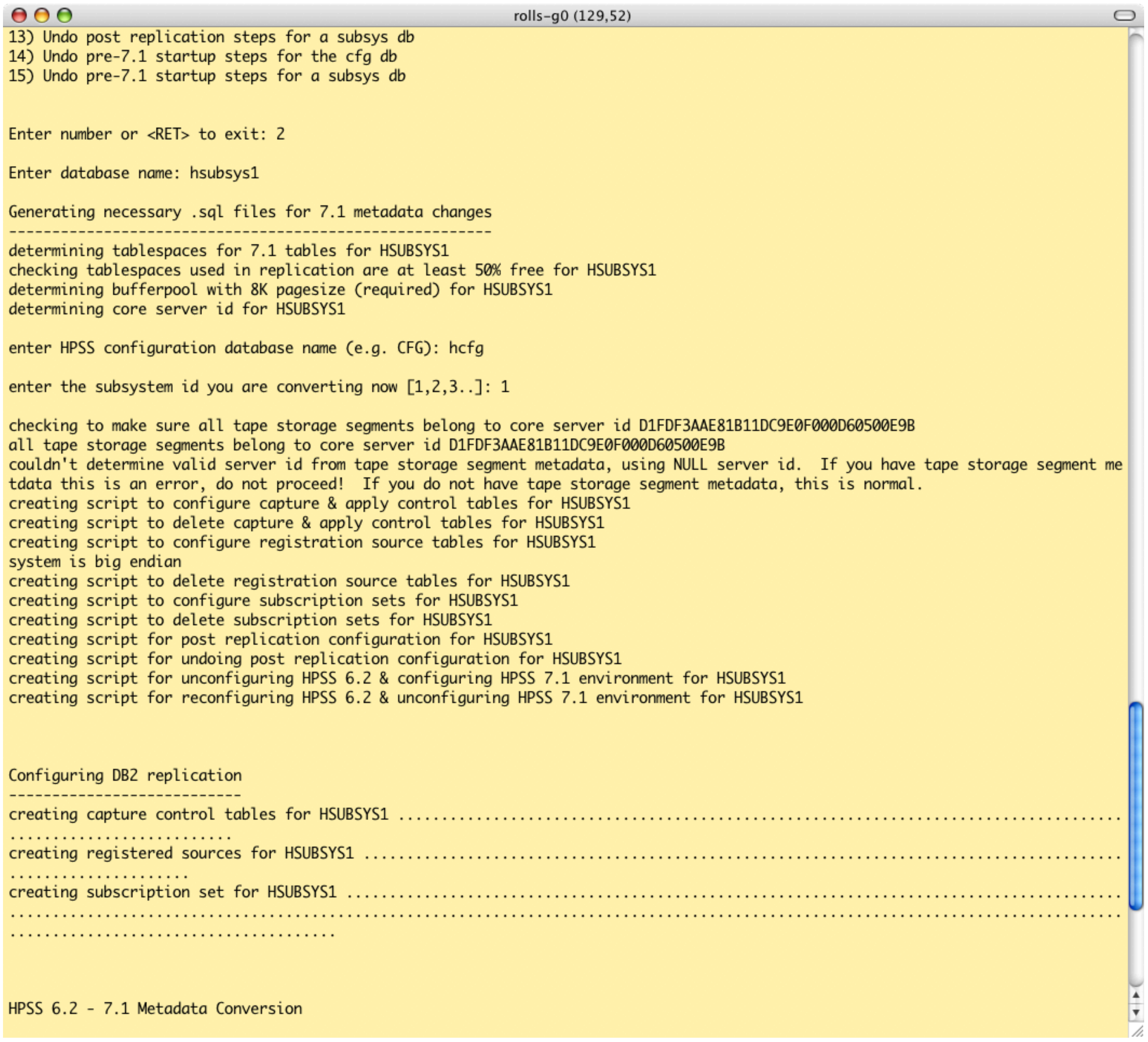


```
$ ./hpss_convert_71
HPSS_ROOT is unset; using /opt/hpss

HPSS 6.2 - 7.1 Metadata Conversion

1) Setup 7.1 tables for the cfg db
2) Setup replication for a subsys db
3) Start replication for a subsys db
4) Stop replication for a subsys db
5) Perform post-replication steps for the cfg db
6) Perform post-replication steps for a subsys db
7) Perform pre-7.1 startup steps for the cfg db
8) Perform pre-7.1 startup steps for a subsys db
9) Perform post-7.1 startup steps for the cfg db
10) Perform post-7.1 startup steps for a subsys db
11) Undo setup of 7.1 tables for the cfg db
12) Undo replication setup for a subsys db
13) Undo post replication steps for a subsys db
14) Undo pre-7.1 startup steps for the cfg db
15) Undo pre-7.1 startup steps for a subsys db


Enter number or <RET> to exit: 1

Enter database name: hcfg

Generating necessary .sql files for 7.1 metadata changes
--------------------------------------------------------
determining tablespaces for 7.1 tables for HCFG
checking tablespaces used in replication are at least 25% free for HCFG
creating script to create 7.1 tables for HCFG
creating script to delete 7.1 tables for HCFG
creating script for post replication configuration for HCFG
creating script for post replication loads for HCFG
creating script for unconfiguring HPSS 6.2 & configuring HPSS 7.1 environment for HCFG
creating script for unconfiguring HPSS 6.2 & configuring HPSS 7.1 environment for HCFG
creating script to reconfigure HPSS 6.2 & unconfigure HPSS 7.1 environment for HCFG



Configuring 7.1 table definitions
---------------------------------
creating 7.1 tables for HCFG ....................................



HPSS 6.2 - 7.1 Metadata Conversion

1) Setup 7.1 tables for the cfg db
2) Setup replication for a subsys db
3) Start replication for a subsys db
4) Stop replication for a subsys db
```

2.6.3.3. Setup HPSS 7.1.0 Subsystem Tables (Step 2)

A reminder to execute this and all subsequent steps as the DB2 instance owner (e.g. hpssdb). Ensure you login as this user before executing the step. Step 2 sets up replication for the subsystem database entered when prompted. Programs executed are **hpss_convert_71_subsys_prep** and **hpss_convert_71_subsys**. The programs will setup tables necessary for performing replication and create the new 7.1.0 tables. Some tables necessary for replication are placed in new tablespaces setup as DMS containers that are files located in `/var/hpss/convert/7.1`. About 200 MB of space is required for the replication tables. However, systems with significantly large replicated tables or that have heavy update activity during replication will require more free disk space in `/var/hpss/convert/7.1` to capture and apply changes to the replicated tables between the replication interval of 5 minutes.

Figure 2.3. Creating capture control



```
13) Undo post replication steps for a subsys db
14) Undo pre-7.1 startup steps for the cfg db
15) Undo pre-7.1 startup steps for a subsys db


Enter number or <RET> to exit: 2

Enter database name: hsubsys1

Generating necessary .sql files for 7.1 metadata changes
-------------------------------------------------------
determining tablespaces for 7.1 tables for HSUBSYS1
checking tablespaces used in replication are at least 50% free for HSUBSYS1
determining bufferpool with 8K pagesize (required) for HSUBSYS1
determining core server id for HSUBSYS1

enter HPSS configuration database name (e.g. CFG): hcfg

enter the subsystem id you are converting now [1,2,3..]: 1

checking to make sure all tape storage segments belong to core server id D1FDF3AAE81B11DC9E0F000D60500E9B
all tape storage segments belong to core server id D1FDF3AAE81B11DC9E0F000D60500E9B
couldn't determine valid server id from tape storage segment metadata, using NULL server id.  If you have tape storage segment me
tdata this is an error, do not proceed!  If you do not have tape storage segment metadata, this is normal.
creating script to configure capture & apply control tables for HSUBSYS1
creating script to delete capture & apply control tables for HSUBSYS1
creating script to configure registration source tables for HSUBSYS1
system is big endian
creating script to delete registration source tables for HSUBSYS1
creating script to configure subscription sets for HSUBSYS1
creating script to delete subscription sets for HSUBSYS1
creating script for post replication configuration for HSUBSYS1
creating script for undoing post replication configuration for HSUBSYS1
creating script for unconfiguring HPSS 6.2 & configuring HPSS 7.1 environment for HSUBSYS1
creating script for reconfiguring HPSS 6.2 & unconfiguring HPSS 7.1 environment for HSUBSYS1



Configuring DB2 replication
--------------------------
creating capture control tables for HSUBSYS1 ..................................................................................
..........................
creating registered sources for HSUBSYS1 ......................................................................................
.....................
creating subscription set for HSUBSYS1 ........................................................................................
...............................................................................................................................
......................................



HPSS 6.2 - 7.1 Metadata Conversion
```

2.6.3.4. Disabling DB2 Backups for HPSS 6.2 (optional)

This step is optional and largely depends on the site’s individual circumstances. However, if HPSS 6.2 is running and replication is active while backups are performed, HPSS performance will be severely degraded. This is largely due to the impact of the initial LOAD for replication and the backup utilities on the database memory and disk resources, and on system CPU load. If backups are still performed during replication, consider disabling the backups until at least the initial LOAD (full refresh) completes during replication startup. At larger HPSS sites, the initial LOAD process may take several days to complete. Disabling the backups while the LOAD runs for all subsystem tables will significantly ease the load on the system and on HPSS 6.2 if it is running.

2.6.3.5. Starting Replication for a HPSS 7.1.0 Subsystem (Step 3)

Step 3 executes **hpss_convert_71_subsys_start** which will start the capture and apply servers for replicating subsystem databases. This step should not be run against an HPSS configuration database (e.g. HCFG), but should be executed against each subsystem database (e.g. hsubsys1, hsubsys2…). The step may be initiated for all subsystem databases in parallel. The capture and apply servers started will initiate replication for the subsystem databases designated when running the utility. Replication is only intended to be run for as long as it takes to complete replication of the HPSS 6.2 tables. If a site chooses to run the replication environment longer than that, they need to consult DB2 documentation about maintaining a replication environment. The program will also start up a replication monitor. The replication monitor will monitor for certain alert conditions that have been preconfigured with the HPSS conversions.

The first time a site starts the capture and apply programs, they will perform what is called a "full refresh" operation from replication source to target tables. The conversion programs in previous steps have already setup the replication source and target tables and configured capture and apply to run. The first time started, capture and apply will perform a DB2 LOAD operation on each replication source table. It is important to note that depending on each site’s various DB2 configuration settings and hardware, load may not run at full parallelism. This means that DB2 may restrict a system to being able to perform only one load or full refresh at a time. If a site desires the LOAD operations to run with maximum parallelism, it should consider increasing the size of the utility heap, UTIL_HEAP_SZ for each subsystem database.

While replication is running and before it completes fully replicating the HPSS 6.2 environment, a site may run the **hpss_convert_71_subsys_monitor** program to read and summarize the replication alert monitor log. This will provide the number of informational, warning or error messages in the log to know whether there are any concerns or not in terms of replication progress or accuracy. In addition, a site may see the full refresh operation progress by issuing a load query command to DB2 on the target HPSS 7.1.0 tables, such as:

```
% db2 connect to <subsystem dbname>
% db2 load query table HPSS.V71_BFCOSCHANGE
```

Where <subsystem dbname> is the database name for the HPSS subsystem desired. If the table state provided is "Normal" no load is in progress. If a load is in progress, the table state is normally "Load Pending" or "Load".

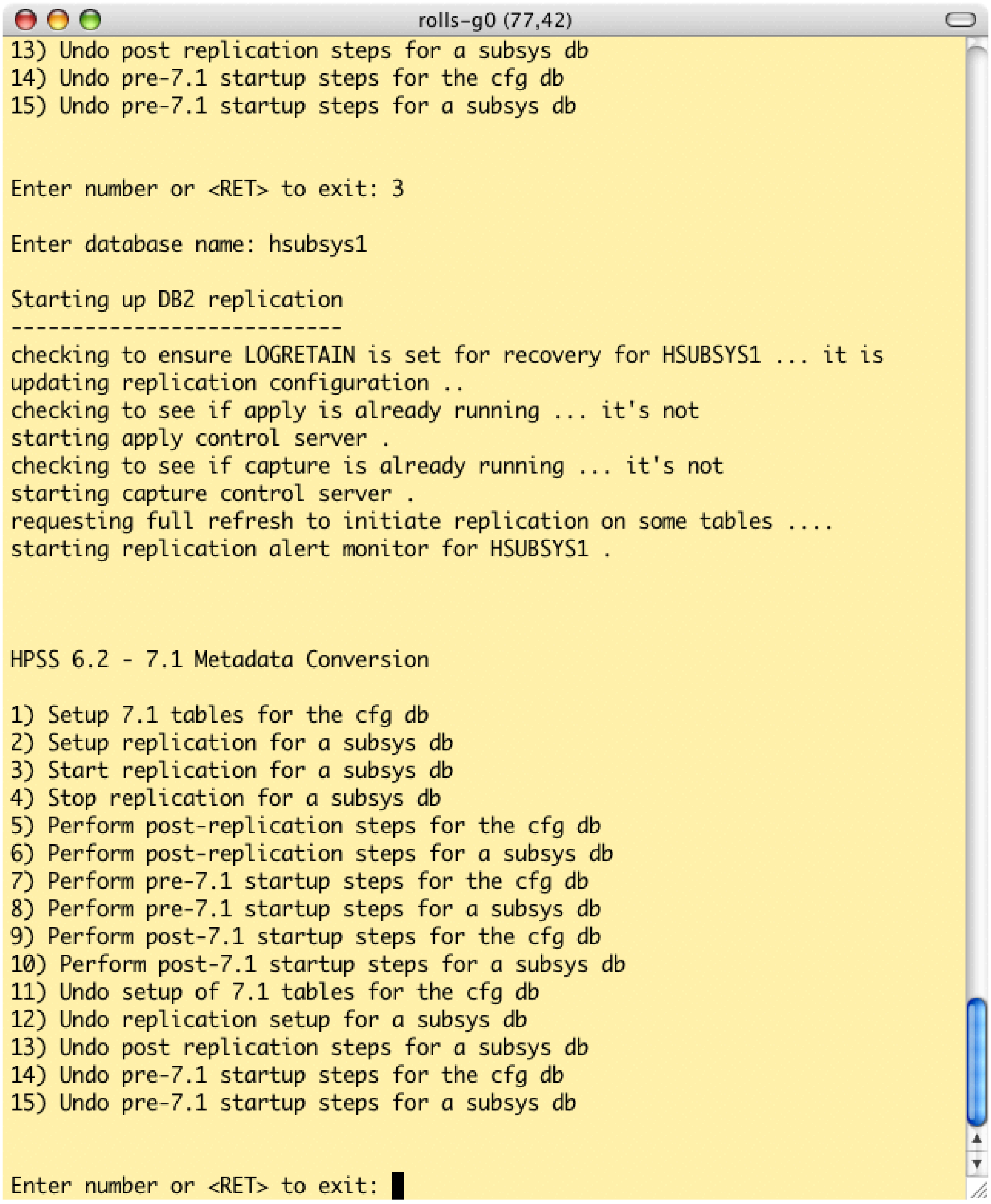
The replication target tables are:

- V71_BFCOSCHANGE
- V71_BFDISKALLOCREC
- V71_BFMIGRREC
- V71_BITFILE
- V71_NSOBJECT
- V71_SSEGDISKEXTENTS
- V71_SSEGTAPEABSADDR
- V71_SSPVDISK
- V71_SSPVTAPE
- V71_STORAGEMAPTAPE
- V71_STORAGESEGDISK
- V71_STORAGESEGTAPE
- V71_VVTAPEABSADDR

Note: there is also a new V71_STORAGESEGAUX table that post conversion will be renamed to STORAGESEGAUX, but the table is not replicated and is intended to be empty with a newly converted 7.1.0 HPSS system. Also, there is a new V71_BFSSUNLINK table that post conversion will rename to BFSSUNLINK, but the table is not replicated and will be empty with the newly converted 7.1.0 HPSS system.

See below for output of a successful replication startup.

Figure 2.4. Successful replication startup



```
13) Undo post replication steps for a subsys db
14) Undo pre-7.1 startup steps for the cfg db
15) Undo pre-7.1 startup steps for a subsys db


Enter number or <RET> to exit: 3

Enter database name: hsubsys1

Starting up DB2 replication
---------------------------
checking to ensure LOGRETAIN is set for recovery for HSUBSYS1 ... it is
updating replication configuration ..
checking to see if apply is already running ... it's not
starting apply control server .
checking to see if capture is already running ... it's not
starting capture control server .
requesting full refresh to initiate replication on some tables ....
starting replication alert monitor for HSUBSYS1 .



HPSS 6.2 - 7.1 Metadata Conversion

1) Setup 7.1 tables for the cfg db
2) Setup replication for a subsys db
3) Start replication for a subsys db
4) Stop replication for a subsys db
5) Perform post-replication steps for the cfg db
6) Perform post-replication steps for a subsys db
7) Perform pre-7.1 startup steps for the cfg db
8) Perform pre-7.1 startup steps for a subsys db
9) Perform post-7.1 startup steps for the cfg db
10) Perform post-7.1 startup steps for a subsys db
11) Undo setup of 7.1 tables for the cfg db
12) Undo replication setup for a subsys db
13) Undo post replication steps for a subsys db
14) Undo pre-7.1 startup steps for the cfg db
15) Undo pre-7.1 startup steps for a subsys db


Enter number or <RET> to exit:
```

2.6.3.6. Checking Status and Monitoring Replication Progress

Sites may run **hpss_convert_71_subsys_verify** while HPSS 6.2 is running or not when replication is underway to see if all rows in the HPSS 6.2 tables are in the HPSS 7.1.0 tables. During the first startup of replication, the HPSS 7.1.0 tables will be unavailable for query due to the LOAD operation that occurs during the full refresh or initial part of the replication process. Running the verification program shortly after initiating replication may fail for tables that are being loaded by the apply program.

Remember to execute these programs once for each subsystem database in the HPSS system. There is a -d option provided to specify database name.

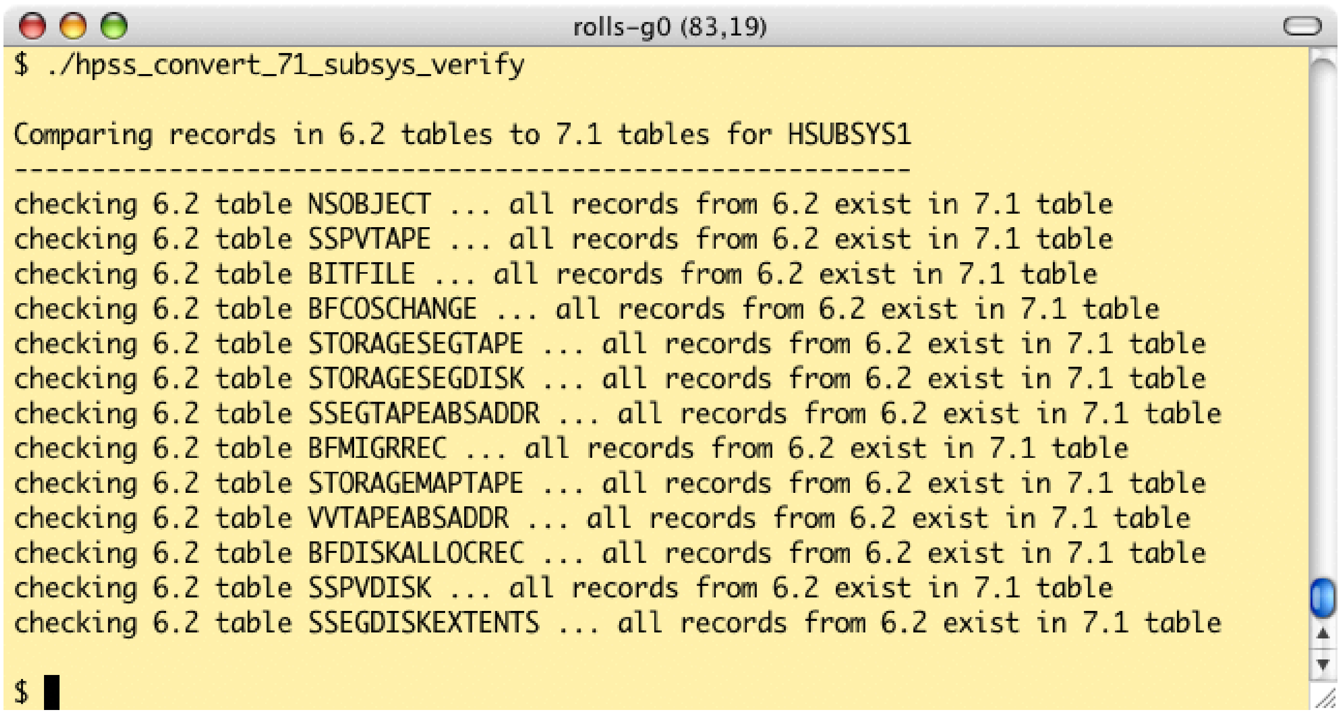
```
% ./hpss_convert_71_subsys_verify -d hsubsys2
% ./hpss_convert_71_subsys_monitor -d hsubsys2
```

To avoid scanning the whole table for every mismatch that may occur, sites may also use the -n option to specify the number of random rows to check. On a HPSS system with 140 million records in a table, the verify program ran for over 10 hours checking an entire table. This is why large HPSS sites are recommended to only check a sampling of metadata during replication. For example:

```
% ./hpss_convert_71_subsys_verify -n 100
```

Will show 100 randomly selected rows from the HPSS 6.2 table and tell you whether they are in the HPSS 7.1.0 table or not. Output for a database called HSUBSYS1 (the default) that has been fully replicated and for which replication may be stopped looks like the output in the figure below.

Figure 2.5. Subsystem verify output

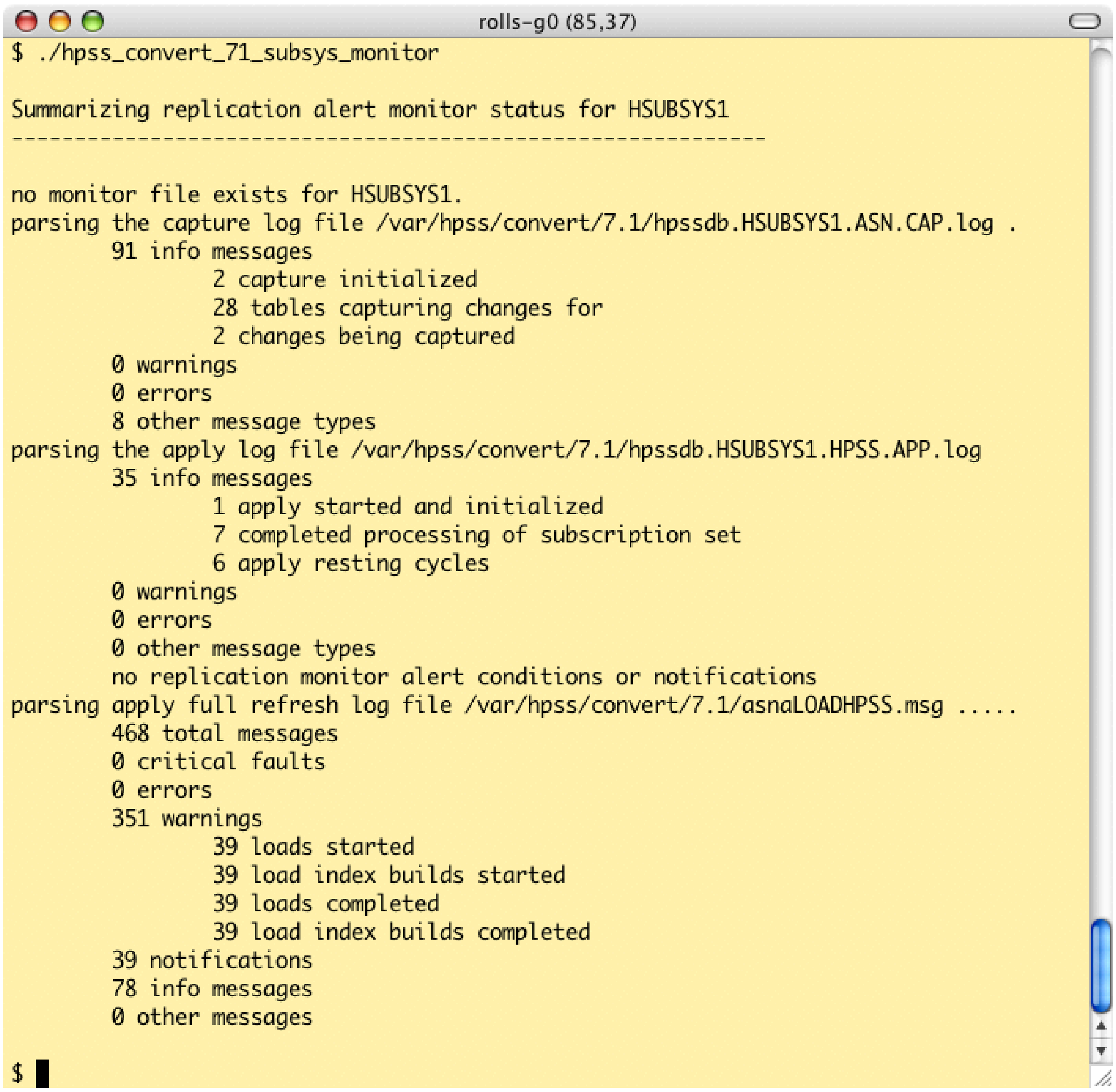


```
$ ./hpss_convert_71_subsys_verify

Comparing records in 6.2 tables to 7.1 tables for HSUBSYS1
----------------------------------------------------------
checking 6.2 table NSOBJECT ... all records from 6.2 exist in 7.1 table
checking 6.2 table SSPVTAPE ... all records from 6.2 exist in 7.1 table
checking 6.2 table BITFILE ... all records from 6.2 exist in 7.1 table
checking 6.2 table BFCOSCHANGE ... all records from 6.2 exist in 7.1 table
checking 6.2 table STORAGESEGTAPE ... all records from 6.2 exist in 7.1 table
checking 6.2 table STORAGESEGDISK ... all records from 6.2 exist in 7.1 table
checking 6.2 table SSEGTAPEABSADDR ... all records from 6.2 exist in 7.1 table
checking 6.2 table BFMIGRREC ... all records from 6.2 exist in 7.1 table
checking 6.2 table STORAGEMAPTAPE ... all records from 6.2 exist in 7.1 table
checking 6.2 table VVTAPEABSADDR ... all records from 6.2 exist in 7.1 table
checking 6.2 table BFDISKALLOCREC ... all records from 6.2 exist in 7.1 table
checking 6.2 table SSPVDISK ... all records from 6.2 exist in 7.1 table
checking 6.2 table SSEGDISKEXTENTS ... all records from 6.2 exist in 7.1 table

$
```

If HPSS 6.2 is still running you should expect to have metadata that has not been replicated to the 7.1.0 tables. The replicated metadata should not be expected to be synchronized until HPSS 6.2 has been shutdown and at least 10 minutes has elapsed for several capture and apply cycles to complete.

The **hpss_convert_71_subsys_monitor** program helps with parsing the various logs that are available in `/var/hpss/convert/7.1` to identify the number of warnings (generally can be ignored), errors (should not be ignored, replication may not be working), and critical problems (replication is definitely not working) that exist in the logs. If errors or critical problems exist, the logs will need to be analyzed to determine the exact problem. Monitor program output for replication progressing successfully would look similar to the figure below.

Figure 2.6. Replication monitoring output



```
$ ./hpss_convert_71_subsys_monitor

Summarizing replication alert monitor status for HSUBSYS1
------------------------------------------------------------

no monitor file exists for HSUBSYS1.
parsing the capture log file /var/hpss/convert/7.1/hpssdb.HSUBSYS1.ASN.CAP.log .
        91 info messages
                2 capture initialized
                28 tables capturing changes for
                2 changes being captured
        0 warnings
        0 errors
        8 other message types
parsing the apply log file /var/hpss/convert/7.1/hpssdb.HSUBSYS1.HPSS.APP.log
        35 info messages
                1 apply started and initialized
                7 completed processing of subscription set
                6 apply resting cycles
        0 warnings
        0 errors
        0 other message types
        no replication monitor alert conditions or notifications
parsing apply full refresh log file /var/hpss/convert/7.1/asnaLOADHPSS.msg .....
        468 total messages
        0 critical faults
        0 errors
        351 warnings
                39 loads started
                39 load index builds started
                39 loads completed
                39 load index builds completed
        39 notifications
        78 info messages
        0 other messages

$
```

Do not run the **hpss_convert_71_subsys_diff** program until the HPSS 6.2 system is stopped and significant disk and time are available to spend allowing the program to complete. This program is intended for detailed problem diagnosis rather than for validating metadata during conversion.

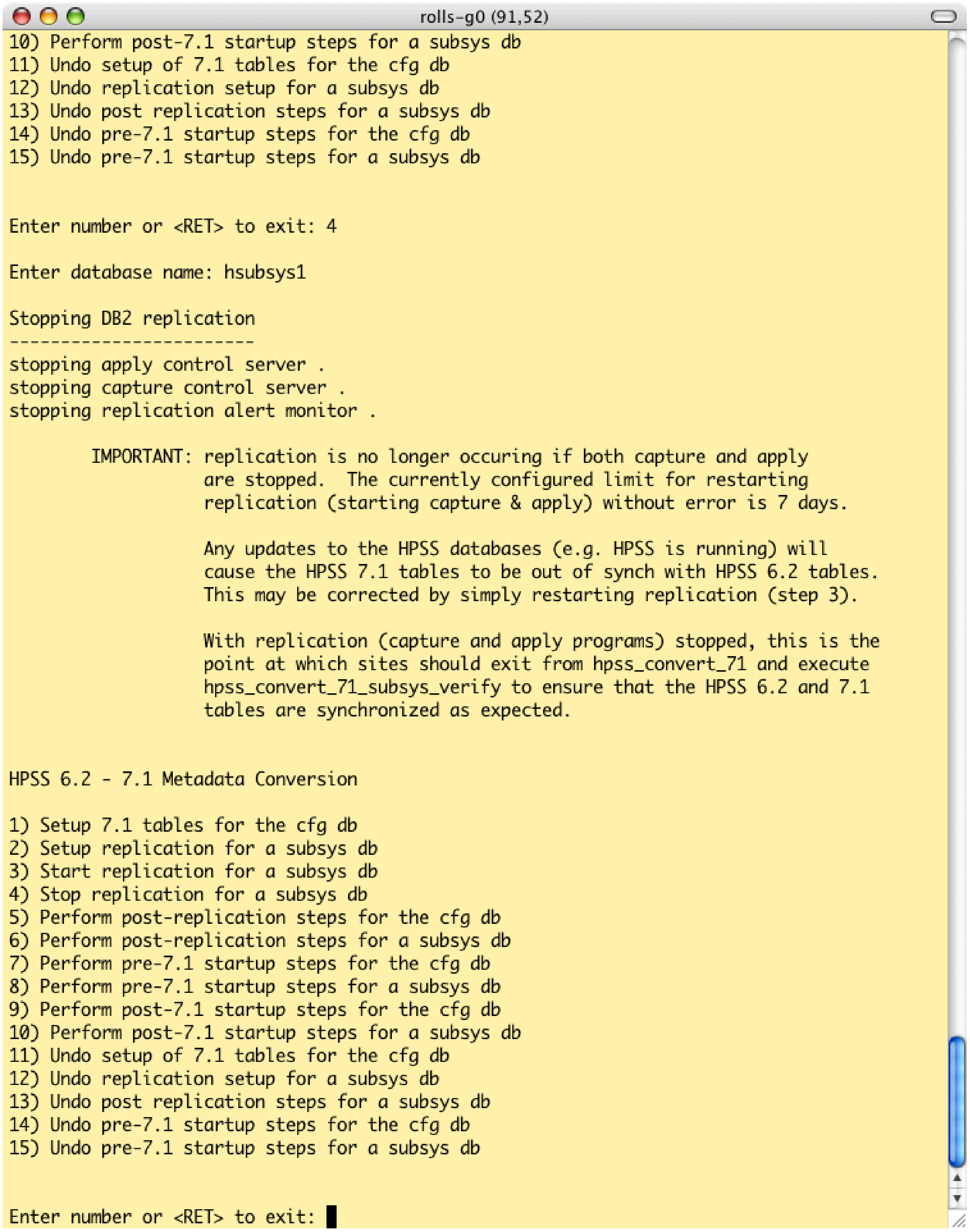
2.6.3.7. Stopping Replication for a HPSS Subsystem (Step 4)

Step 4 will run the **hpss_convert_71_subsys_stop** program for the subsystem database entered when prompted. Remember to execute this step once for each subsystem database in the HPSS system. This will stop the capture program from reading database logs and inserting records into the change data tables. It will also stop the apply program from reading the change data tables and applying those changes to the target (HPSS 7.1.0) tables.

Stopping the replication programs has consequences if changes to the replicated tables still occur. The major issue is that the capture program requires database log files that have not been processed to be available. In the event that log files are archived and archived logs are removed from the archive log directory, the capture program will not be able to begin processing once restarted. A program **hpss_convert_71_log_needed** is provided to query a capture control table to identify the log sequence number (LSN) of the last committed transaction handled by the capture program. The log file name is provided that contains this LSN. This log file and any others since will need to be made available so that the capture program can restart.

Prior to executing step 4 (stopping replication) and prior to stopping the HPSS 6.2 system, ensure that the BFSSUNLINK table is empty. For details on checking the BFSSUNLINK table, see section 5.2.4 of this guide. When deciding to stop replication in preparation for completing the remaining upgrade steps, the HPSS 6.2 system should be stopped. To stop replication, execute step 4 of the **hpss_convert_71** program. See the figure below for output of a successful replication stop.

Figure 2.7. Stop replication output



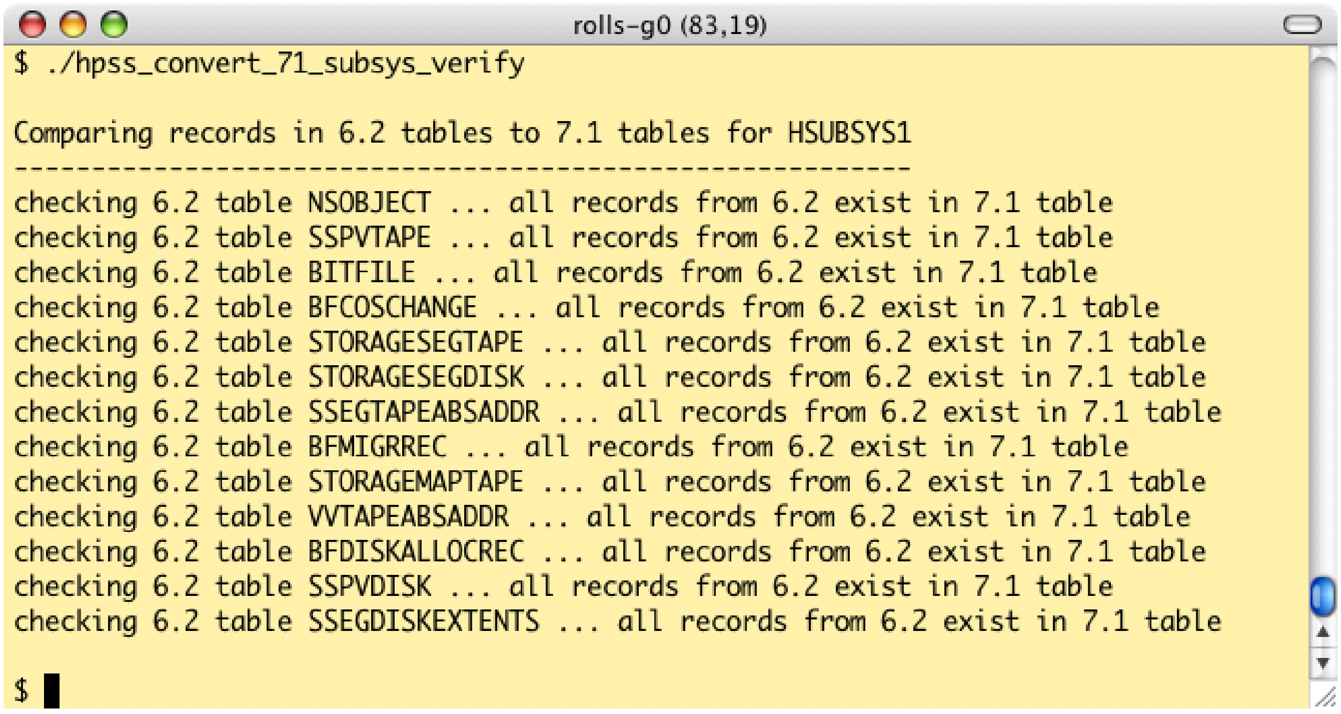
```
10) Perform post-7.1 startup steps for a subsys db
11) Undo setup of 7.1 tables for the cfg db
12) Undo replication setup for a subsys db
13) Undo post replication steps for a subsys db
14) Undo pre-7.1 startup steps for the cfg db
15) Undo pre-7.1 startup steps for a subsys db


Enter number or <RET> to exit: 4

Enter database name: hsubsys1

Stopping DB2 replication
------------------------
stopping apply control server .
stopping capture control server .
stopping replication alert monitor .

        IMPORTANT: replication is no longer occuring if both capture and apply
                   are stopped.  The currently configured limit for restarting
                   replication (starting capture & apply) without error is 7 days.

                   Any updates to the HPSS databases (e.g. HPSS is running) will
                   cause the HPSS 7.1 tables to be out of synch with HPSS 6.2 tables.
                   This may be corrected by simply restarting replication (step 3).

                   With replication (capture and apply programs) stopped, this is the
                   point at which sites should exit from hpss_convert_71 and execute
                   hpss_convert_71_subsys_verify to ensure that the HPSS 6.2 and 7.1
                   tables are synchronized as expected.


HPSS 6.2 - 7.1 Metadata Conversion

1) Setup 7.1 tables for the cfg db
2) Setup replication for a subsys db
3) Start replication for a subsys db
4) Stop replication for a subsys db
5) Perform post-replication steps for the cfg db
6) Perform post-replication steps for a subsys db
7) Perform pre-7.1 startup steps for the cfg db
8) Perform pre-7.1 startup steps for a subsys db
9) Perform post-7.1 startup steps for the cfg db
10) Perform post-7.1 startup steps for a subsys db
11) Undo setup of 7.1 tables for the cfg db
12) Undo replication setup for a subsys db
13) Undo post replication steps for a subsys db
14) Undo pre-7.1 startup steps for the cfg db
15) Undo pre-7.1 startup steps for a subsys db


Enter number or <RET> to exit: █
```

2.6.3.8. Verifying Completion of Metadata Transformations

With HPSS down, the site should ensure that replication completes processing of all remaining transactions by leaving the capture and apply programs running through several capture and apply intervals. The programs are configured to run every 5 minutes.

The site needs to run the **hpss_convert_71_subsys_verify** program. This program will perform SQL to ensure that all rows present in each HPSS 6.2 table are present in the HPSS 7.1.0 tables. It will not check if there are additional rows in the HPSS 7.1.0 tables. A **-n** option is provided to check a random subset of rows in the HPSS 6.2 tables for sites that do not have downtime to allow the verify program to complete. At a larger HPSS installation with 140 million records in some DB2 tables, the verify program took over 10 hours on a single table. Output of a successful verification are shown in the figure below.

Figure 2.8. Subsystem conversion verification



```
$ ./hpss_convert_71_subsys_verify

Comparing records in 6.2 tables to 7.1 tables for HSUBSYS1
----------------------------------------------------------
checking 6.2 table NSOBJECT ... all records from 6.2 exist in 7.1 table
checking 6.2 table SSPVTAPE ... all records from 6.2 exist in 7.1 table
checking 6.2 table BITFILE ... all records from 6.2 exist in 7.1 table
checking 6.2 table BFCOSCHANGE ... all records from 6.2 exist in 7.1 table
checking 6.2 table STORAGESEGTAPE ... all records from 6.2 exist in 7.1 table
checking 6.2 table STORAGESEGDISK ... all records from 6.2 exist in 7.1 table
checking 6.2 table SSEGTAPEABSADDR ... all records from 6.2 exist in 7.1 table
checking 6.2 table BFMIGRREC ... all records from 6.2 exist in 7.1 table
checking 6.2 table STORAGEMAPTAPE ... all records from 6.2 exist in 7.1 table
checking 6.2 table VVTAPEABSADDR ... all records from 6.2 exist in 7.1 table
checking 6.2 table BFDISKALLOCREC ... all records from 6.2 exist in 7.1 table
checking 6.2 table SSPVDISK ... all records from 6.2 exist in 7.1 table
checking 6.2 table SSEGDISKEXTENTS ... all records from 6.2 exist in 7.1 table

$
```

Should mismatches occur, ensure that capture and apply have finished processing through several cycles (5 minutes each cycle). Use the **hpss_convert_71** program to startup replication and allow it to process through several capture and apply cycles (5 minutes each). Stop replication with the **hpss_convert_71** program and rerun the provided verification or diff programs to see if the mismatches still exist. If the mismatch is still present, the site should contact IBM HPSS Customer Support for assistance. The site could consult DB2 documentation to use the asntrep utility to attempt to repair the target tables with output generated by the **asntdiff** utility (used by the **hpss_convert_71_subsys_diff** program).

2.6.3.9. Resolving Potential Metadata Inconsistencies

After step 4, sites should run the **hpss_convert_71_subsys_resolve** program to correct any metadata inconsistencies between the absolute address and tape storage segment and virtual volume metadata.

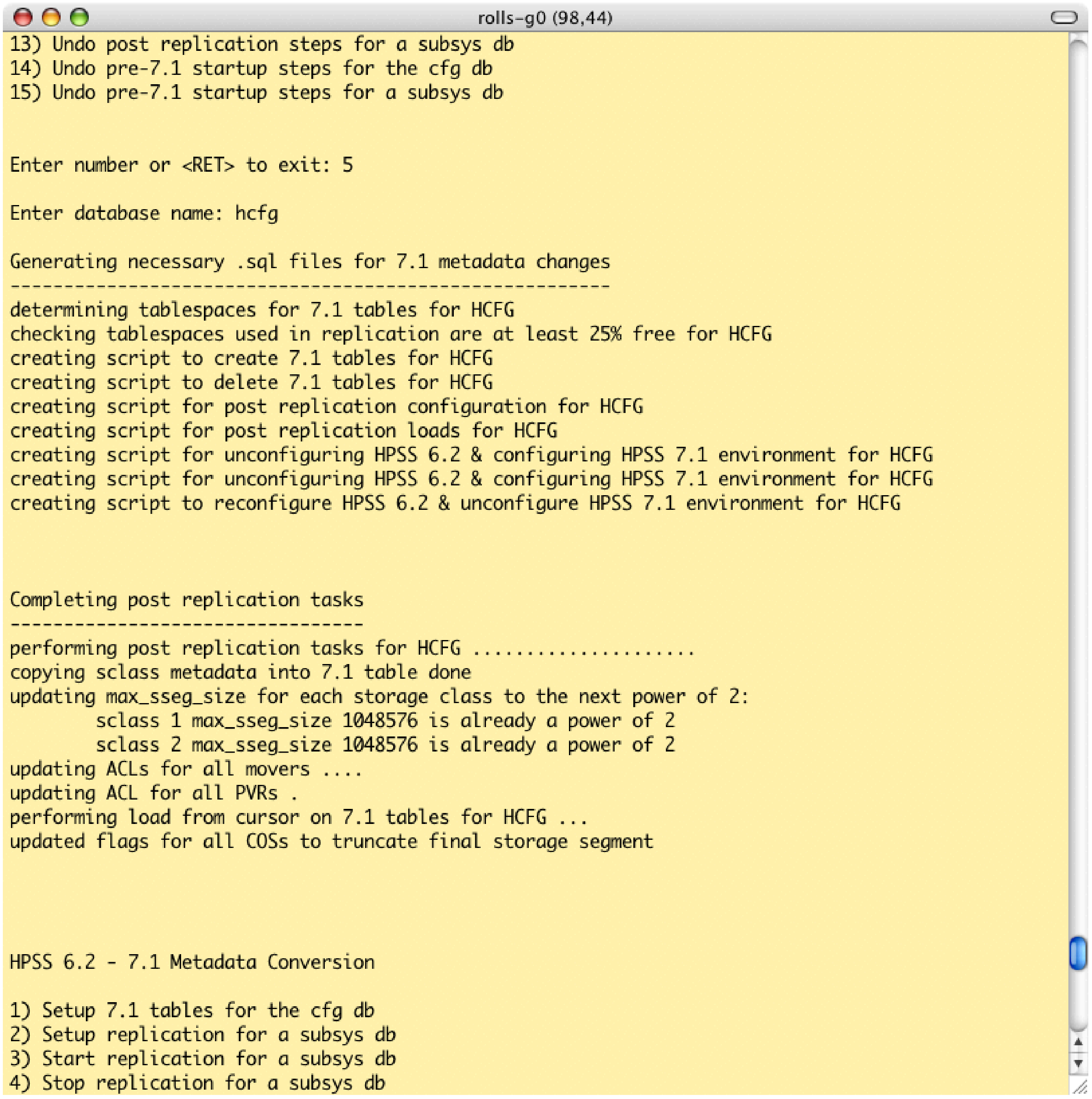
This is necessary so that new JOIN logic in the metadata manager library that the Core Server relies on will function properly. Sites will be prompted for whether they wish to restore any identified 7.1.0 orphaned metadata to its 6.2 orphaned values, or delete the orphaned metadata. Either option is valid and will resolve the metadata inconsistencies to allow proper operation of the metadata manager library and Core Server. The program should be run several times and is only complete when the utility reports "all ok" for each problem that it checks for. Note, if orphaned segments are not deleted when prompted to do so, that step will not report "all ok" and provided that is the only part that reports something other than "all ok", the utility doesn't need to be rerun.

2.6.3.10. Post Replication for HPSS 7.1.0 Configuration Tables (Step 5)

*Before proceeding, be aware that the **hpss_convert_71_subsys_verify** and **hpss_convert_71_subsys_diff** programs will no longer work once this step is complete!*

Once the replication tables match, the site should run step 5. The post replication step for the configuration database is the main step for performing the metadata transformations for the configuration database. The **hpss_convert_71_config_postrepl** program will define a cursor to select all records from the HPSS 6.2 tables being transformed or altered and perform a load from cursor into the HPSS 7.1.0 tables. The program will then perform the transformations on the HPSS 7.1.0 tables (e.g. adding columns and setting default values for new metadata). Successful completion of the post replication for HPSS configuration tables is shown below.

Figure 2.9. Post conversion replication changes

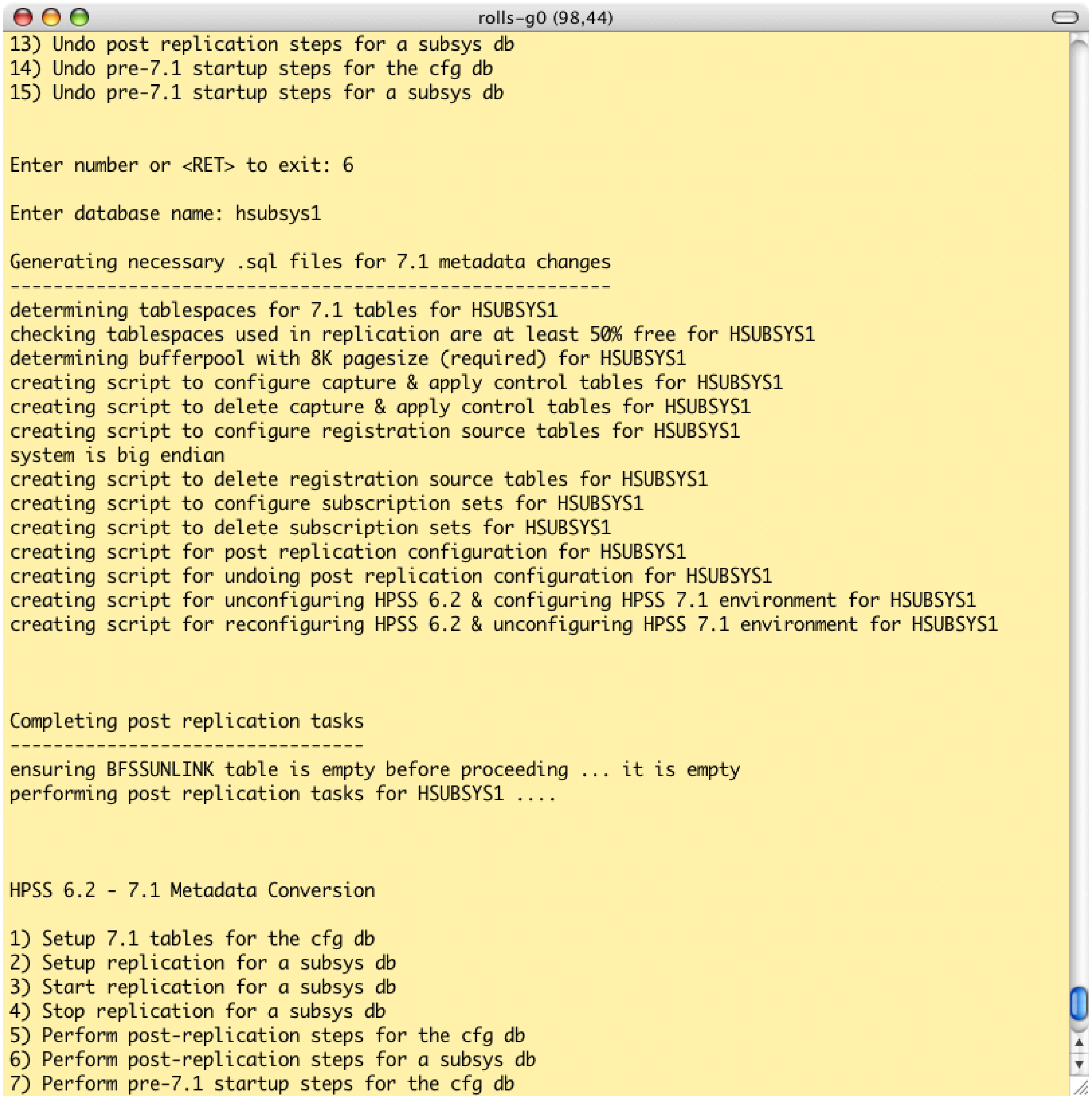


```
13) Undo post replication steps for a subsys db
14) Undo pre-7.1 startup steps for the cfg db
15) Undo pre-7.1 startup steps for a subsys db


Enter number or <RET> to exit: 5

Enter database name: hcfg

Generating necessary .sql files for 7.1 metadata changes
--------------------------------------------------------
determining tablespaces for 7.1 tables for HCFG
checking tablespaces used in replication are at least 25% free for HCFG
creating script to create 7.1 tables for HCFG
creating script to delete 7.1 tables for HCFG
creating script for post replication configuration for HCFG
creating script for post replication loads for HCFG
creating script for unconfiguring HPSS 6.2 & configuring HPSS 7.1 environment for HCFG
creating script for unconfiguring HPSS 6.2 & configuring HPSS 7.1 environment for HCFG
creating script to reconfigure HPSS 6.2 & unconfigure HPSS 7.1 environment for HCFG



Completing post replication tasks
---------------------------------
performing post replication tasks for HCFG .....................
copying sclass metadata into 7.1 table done
updating max_sseg_size for each storage class to the next power of 2:
        sclass 1 max_sseg_size 1048576 is already a power of 2
        sclass 2 max_sseg_size 1048576 is already a power of 2
updating ACLs for all movers ....
updating ACL for all PVRs .
performing load from cursor on 7.1 tables for HCFG ...
updated flags for all COSs to truncate final storage segment




HPSS 6.2 - 7.1 Metadata Conversion

1) Setup 7.1 tables for the cfg db
2) Setup replication for a subsys db
3) Start replication for a subsys db
4) Stop replication for a subsys db
```

2.6.3.11. Post Replication for HPSS 7.1.0 Subsystem Tables (Step 6)

Step 6 is the post replication step for subsystem databases in HPSS. Remember to execute this step once for each subsystem database in the HPSS system. The post replication step performs the metadata transformations to HPSS 7.1.0 tables in the subsystem database designated when prompted. The **hpss_convert_71_subsys_postrepl** program will alter the HPSS 7.1.0 tables. Output for completion of this step is shown in the figure below.

Figure 2.10. Post replication output



```
13) Undo post replication steps for a subsys db
14) Undo pre-7.1 startup steps for the cfg db
15) Undo pre-7.1 startup steps for a subsys db


Enter number or <RET> to exit: 6

Enter database name: hsubsys1

Generating necessary .sql files for 7.1 metadata changes
--------------------------------------------------------
determining tablespaces for 7.1 tables for HSUBSYS1
checking tablespaces used in replication are at least 50% free for HSUBSYS1
determining bufferpool with 8K pagesize (required) for HSUBSYS1
creating script to configure capture & apply control tables for HSUBSYS1
creating script to delete capture & apply control tables for HSUBSYS1
creating script to configure registration source tables for HSUBSYS1
system is big endian
creating script to delete registration source tables for HSUBSYS1
creating script to configure subscription sets for HSUBSYS1
creating script to delete subscription sets for HSUBSYS1
creating script for post replication configuration for HSUBSYS1
creating script for undoing post replication configuration for HSUBSYS1
creating script for unconfiguring HPSS 6.2 & configuring HPSS 7.1 environment for HSUBSYS1
creating script for reconfiguring HPSS 6.2 & unconfiguring HPSS 7.1 environment for HSUBSYS1



Completing post replication tasks
---------------------------------
ensuring BFSSUNLINK table is empty before proceeding ... it is empty
performing post replication tasks for HSUBSYS1 ....



HPSS 6.2 - 7.1 Metadata Conversion

1) Setup 7.1 tables for the cfg db
2) Setup replication for a subsys db
3) Start replication for a subsys db
4) Stop replication for a subsys db
5) Perform post-replication steps for the cfg db
6) Perform post-replication steps for a subsys db
7) Perform pre-7.1 startup steps for the cfg db
```

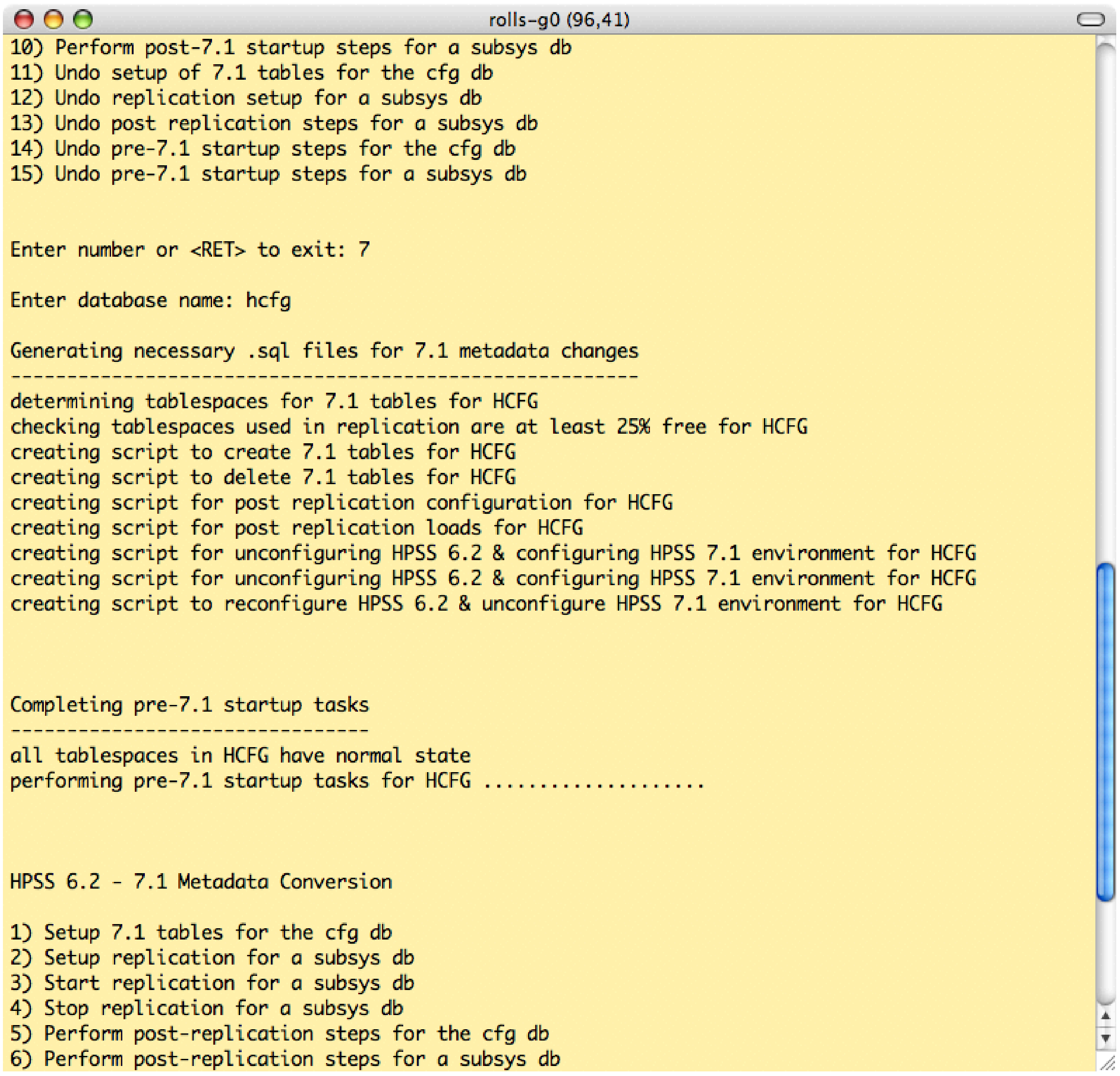
2.6.3.12. Enabling DB2 Backups for HPSS 7.1.0 (optional)

Upon completion of steps 5 and 6, the site should take a full backup of each database. This is the last point at which HPSS 6.2 may be run and is just prior to the point at which HPSS 7.1.0 may be started. Alternatively, if sites do not intend on reverting to HPSS 6.2 they could wait to enable backups of DB2 once the conversion is complete (after step 10).

2.6.3.13. Pre-7.1.0 Startup HPSS Configuration Table Conversion (Step 7)

Once the backups are complete, the site will be ready to run step 7 which simply renames the HPSS 6.2 tables by appending _62 to the table name and renames the HPSS 7.1.0 tables to the original table name. The **hpss_convert_71_config_pre71start** program accomplishes these tasks. See below for output showing successful completion.

Figure 2.11. Configuration table setup tasks


rolls-g0 (96,41)

```
10) Perform post-7.1 startup steps for a subsys db
11) Undo setup of 7.1 tables for the cfg db
12) Undo replication setup for a subsys db
13) Undo post replication steps for a subsys db
14) Undo pre-7.1 startup steps for the cfg db
15) Undo pre-7.1 startup steps for a subsys db


Enter number or <RET> to exit: 7

Enter database name: hcfg

Generating necessary .sql files for 7.1 metadata changes
--------------------------------------------------------
determining tablespaces for 7.1 tables for HCFG
checking tablespaces used in replication are at least 25% free for HCFG
creating script to create 7.1 tables for HCFG
creating script to delete 7.1 tables for HCFG
creating script for post replication configuration for HCFG
creating script for post replication loads for HCFG
creating script for unconfiguring HPSS 6.2 & configuring HPSS 7.1 environment for HCFG
creating script for unconfiguring HPSS 6.2 & configuring HPSS 7.1 environment for HCFG
creating script to reconfigure HPSS 6.2 & unconfigure HPSS 7.1 environment for HCFG



Completing pre-7.1 startup tasks
--------------------------------
all tablespaces in HCFG have normal state
performing pre-7.1 startup tasks for HCFG ....................



HPSS 6.2 - 7.1 Metadata Conversion

1) Setup 7.1 tables for the cfg db
2) Setup replication for a subsys db
3) Start replication for a subsys db
4) Stop replication for a subsys db
5) Perform post-replication steps for the cfg db
6) Perform post-replication steps for a subsys db
```

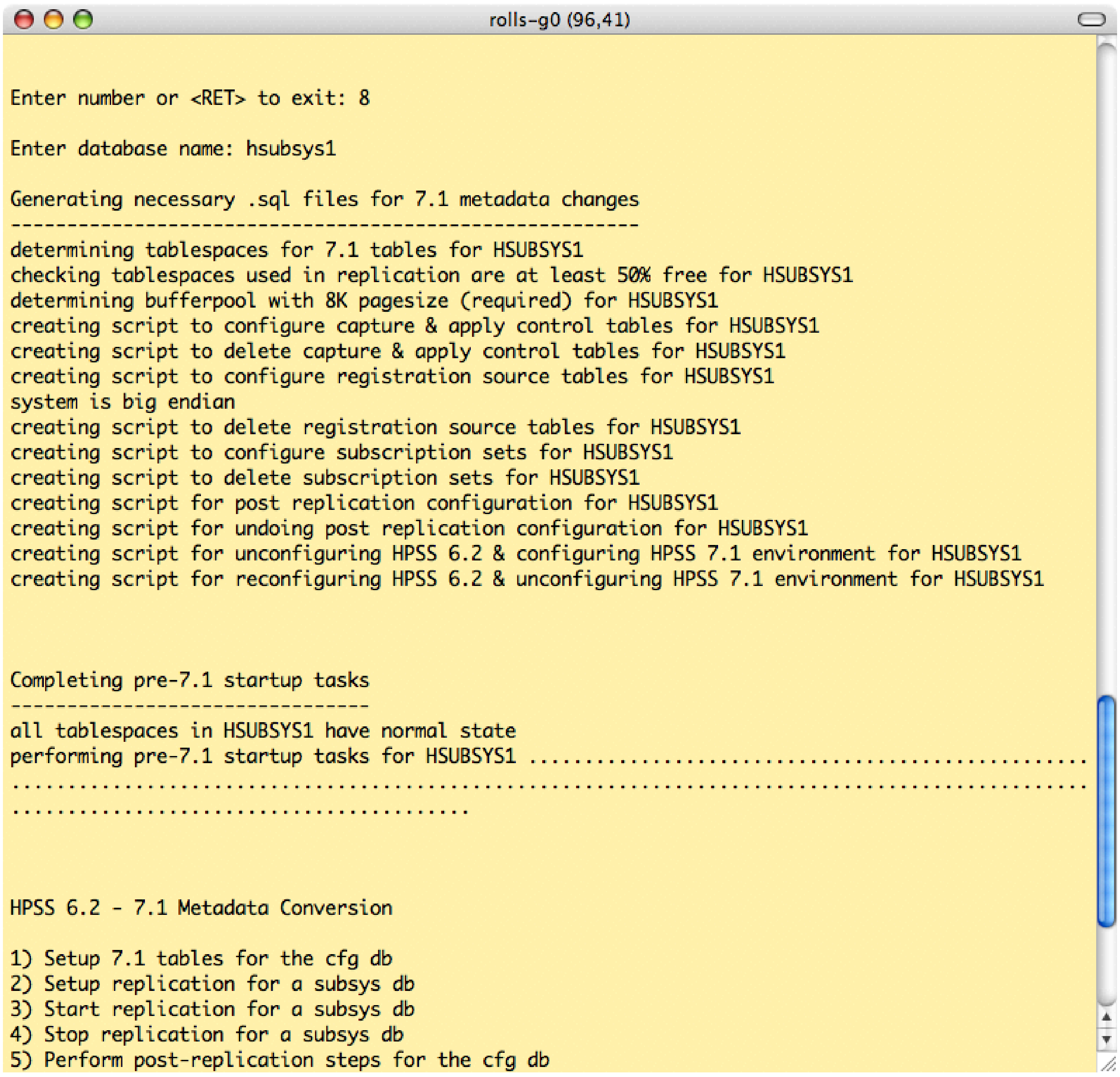
2.6.3.14. Pre-7.1.0 Startup HPSS Subsystem Table Conversion (Step 8)

Step 8 will run the **hpss_convert_71_subsys_pre71start** program to rename the HPSS 6.2 tables by appending _62 to the table name and rename the HPSS 7.1.0 tables to the original table name. Remember to execute this step once for each subsystem database in the HPSS system.

This is the point at which sites can still revert to HPSS 6.2. To do so, sites would run the **hpss_convert_71** program and execute all "undo" steps. These steps remove any HPSS 7.1.0 settings or changes and replace the HPSS 6.2 tables so that HPSS 6.2 may be started after the undo steps are completed. After this step, sites will have a more difficult time reverting to HPSS 6.2.

For output of successful completion of this step, see below.

Figure 2.12. Start subsystem table conversion



```


Enter number or <RET> to exit: 8

Enter database name: hsubsys1

Generating necessary .sql files for 7.1 metadata changes
--------------------------------------------------------
determining tablespaces for 7.1 tables for HSUBSYS1
checking tablespaces used in replication are at least 50% free for HSUBSYS1
determining bufferpool with 8K pagesize (required) for HSUBSYS1
creating script to configure capture & apply control tables for HSUBSYS1
creating script to delete capture & apply control tables for HSUBSYS1
creating script to configure registration source tables for HSUBSYS1
system is big endian
creating script to delete registration source tables for HSUBSYS1
creating script to configure subscription sets for HSUBSYS1
creating script to delete subscription sets for HSUBSYS1
creating script for post replication configuration for HSUBSYS1
creating script for undoing post replication configuration for HSUBSYS1
creating script for unconfiguring HPSS 6.2 & configuring HPSS 7.1 environment for HSUBSYS1
creating script for reconfiguring HPSS 6.2 & unconfiguring HPSS 7.1 environment for HSUBSYS1



Completing pre-7.1 startup tasks
--------------------------------
all tablespaces in HSUBSYS1 have normal state
performing pre-7.1 startup tasks for HSUBSYS1 ...................................................
................................................................................................
.........................................



HPSS 6.2 - 7.1 Metadata Conversion

1) Setup 7.1 tables for the cfg db
2) Setup replication for a subsys db
3) Start replication for a subsys db
4) Stop replication for a subsys db
5) Perform post-replication steps for the cfg db
```

2.6.3.15. Upgrade to DB2 v9.5 (optional)

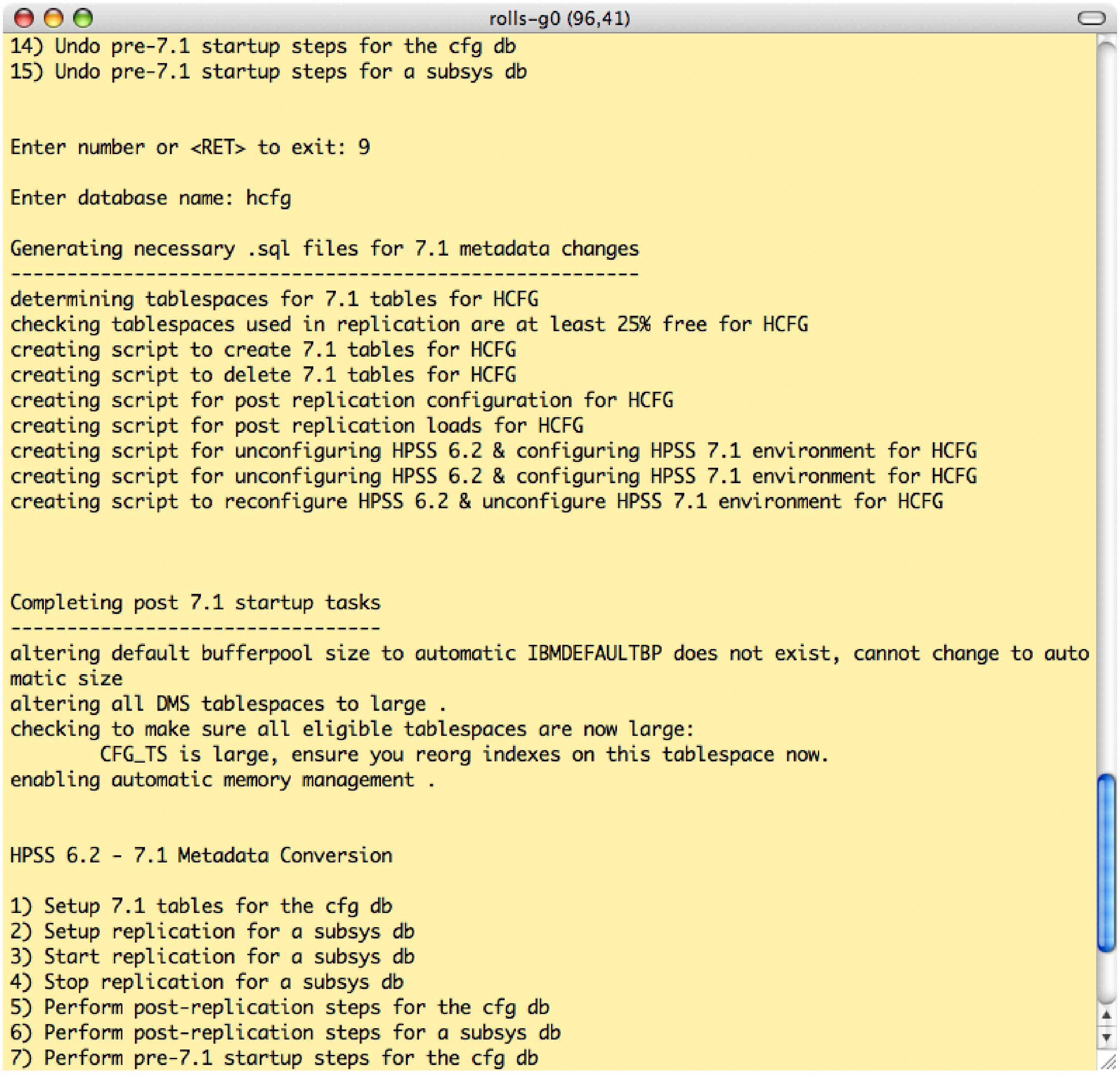
After step 8, the site should migrate from DB2 v8 to v9 to meet HPSS prerequisite software requirements for HPSS 7.1. See DB2 documentation for details in accomplishing this step, or Chapter 10 in this guide. It is possible to run HPSS 7.1.0 on DB2 v8.2 or 9.1 if the site chooses to simplify or isolate the conversion process from problems that may occur with a specific version of DB2. In addition, the conversions also work on a system that is currently running DB2 v9.5, so it is possible to upgrade DB2 before starting the conversions, but not recommended as this would prevent a site from returning to HPSS 6.2 if they are running DB2 v8.2 with HPSS 6.2; DB2 does not support reverting a migrated v9 instance back to v8. If a site is already running DB2 v9.5 with HPSS 6.2 they can revert back without any special considerations.

2.6.3.16. Post-7.1.0 Startup HPSS Configuration Database Changes (Step 9)

Step 9 must be run after the site upgrade to DB2 v9.5 or the attempted modifications will not succeed. This step executes the **hpss_convert_71_config_post71start** program which enables automatic memory management, adjusts the default buffer pool to automatic memory management and changes all DMS tablespaces to large. See the figure below for output of successful completion of this step.

Sites should also refer to *HPSS Configuration Database Changes Section 2.5.2.1, “HPSS Configuration Database Changes (HCFG/CFG)”* and consider site specific values that differ from the default values that the conversion automatically selects. Changes do not need to be made unless a site prefers to take advantage of a new HPSS feature.

Figure 2.13. Subsystem conversion verification

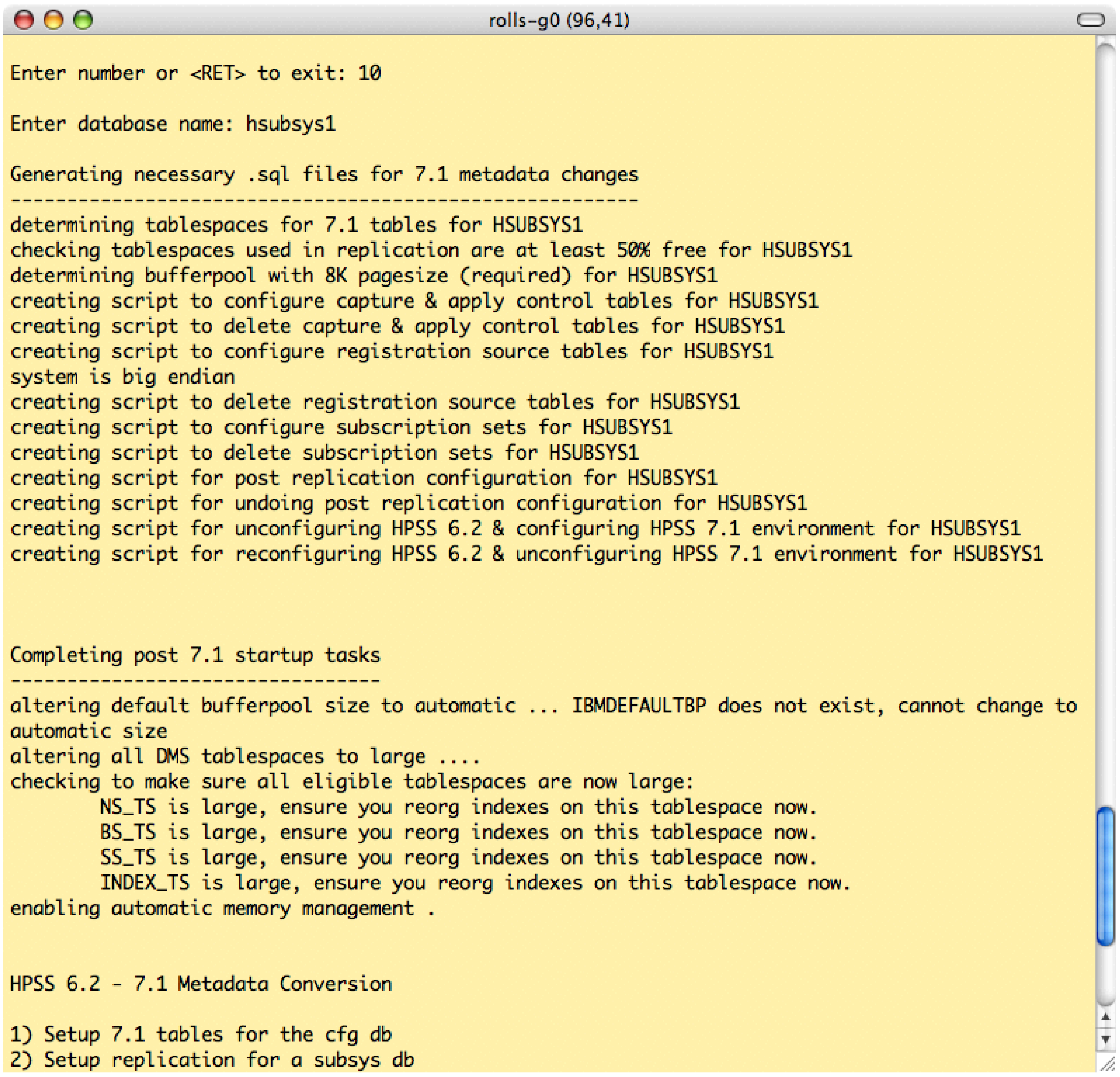


```
14) Undo pre-7.1 startup steps for the cfg db
15) Undo pre-7.1 startup steps for a subsys db


Enter number or <RET> to exit: 9

Enter database name: hcfg

Generating necessary .sql files for 7.1 metadata changes
-------------------------------------------------------
determining tablespaces for 7.1 tables for HCFG
checking tablespaces used in replication are at least 25% free for HCFG
creating script to create 7.1 tables for HCFG
creating script to delete 7.1 tables for HCFG
creating script for post replication configuration for HCFG
creating script for post replication loads for HCFG
creating script for unconfiguring HPSS 6.2 & configuring HPSS 7.1 environment for HCFG
creating script for unconfiguring HPSS 6.2 & configuring HPSS 7.1 environment for HCFG
creating script to reconfigure HPSS 6.2 & unconfigure HPSS 7.1 environment for HCFG



Completing post 7.1 startup tasks
---------------------------------
altering default bufferpool size to automatic IBMDEFAULTBP does not exist, cannot change to auto
matic size
altering all DMS tablespaces to large .
checking to make sure all eligible tablespaces are now large:
        CFG_TS is large, ensure you reorg indexes on this tablespace now.
enabling automatic memory management .


HPSS 6.2 - 7.1 Metadata Conversion

1) Setup 7.1 tables for the cfg db
2) Setup replication for a subsys db
3) Start replication for a subsys db
4) Stop replication for a subsys db
5) Perform post-replication steps for the cfg db
6) Perform post-replication steps for a subsys db
7) Perform pre-7.1 startup steps for the cfg db
```

2.6.3.17. Post-7.1.0 Startup HPSS Subsystem Database Changes (Step 10)

Step 10 must be run after the site upgrade to DB2 v9.5 or the attempted modifications will not succeed. This step executes the **hpss_convert_71_subsys_post71start** program which enables automatic memory management, adjusts the default buffer pool to automatic memory management and changes all DMS tablespaces to large. The figure below shows output for successful completion of this step.

Figure 2.14. Subsystem table changes



```
Enter number or <RET> to exit: 10

Enter database name: hsubsys1

Generating necessary .sql files for 7.1 metadata changes
--------------------------------------------------------
determining tablespaces for 7.1 tables for HSUBSYS1
checking tablespaces used in replication are at least 50% free for HSUBSYS1
determining bufferpool with 8K pagesize (required) for HSUBSYS1
creating script to configure capture & apply control tables for HSUBSYS1
creating script to delete capture & apply control tables for HSUBSYS1
creating script to configure registration source tables for HSUBSYS1
system is big endian
creating script to delete registration source tables for HSUBSYS1
creating script to configure subscription sets for HSUBSYS1
creating script to delete subscription sets for HSUBSYS1
creating script for post replication configuration for HSUBSYS1
creating script for undoing post replication configuration for HSUBSYS1
creating script for unconfiguring HPSS 6.2 & configuring HPSS 7.1 environment for HSUBSYS1
creating script for reconfiguring HPSS 6.2 & unconfiguring HPSS 7.1 environment for HSUBSYS1



Completing post 7.1 startup tasks
---------------------------------
altering default bufferpool size to automatic ... IBMDEFAULTBP does not exist, cannot change to
automatic size
altering all DMS tablespaces to large ....
checking to make sure all eligible tablespaces are now large:
        NS_TS is large, ensure you reorg indexes on this tablespace now.
        BS_TS is large, ensure you reorg indexes on this tablespace now.
        SS_TS is large, ensure you reorg indexes on this tablespace now.
        INDEX_TS is large, ensure you reorg indexes on this tablespace now.
enabling automatic memory management .


HPSS 6.2 - 7.1 Metadata Conversion

1) Setup 7.1 tables for the cfg db
2) Setup replication for a subsys db
```

Sites should also refer to the configuration changes section (Metadata Change Detail) and consider site specific values that differ from the default values that the conversion automatically selects. Changes do not need to be made unless a site prefers to take advantage of a new HPSS feature.

2.6.4. Installing Java5 for HPSS 7.1.0 SSM

The HPSS 7.1.0 SSM requires the 32-bit Java5 SDK to function. After downloading and installing Java5 on the system where HPSS 7.1.0 SSM will run, the site should use the new SSM configuration file template provided with HPSS 7.1. Copy the template into the SSM configuration directory, normally located in `/var/hpss/ssm`. The file should be named `ssm.conf` and have at least read permissions for the user that HPSS SSM server runs as. For example, the HPSS 7.1.0 code is installed in `/usr/lpp/hpss71`:

```
% cp /usr/lpp/hpss71/config/templates/ssm.conf.template \
    /var/hpss/ssm/ssm.conf
```

Edit the contents to support the site specific settings desired. The file can be world readable, but should be read accessible by the user the SSM server runs under.

At this point, the SSM should function under HPSS 7.1.0. Note that some SSM configuration files are no longer used and may be deleted. Specifically, the SSM `login.conf` file normally located in `/var/hpss/ssm` is no longer necessary.

2.6.5. Updating the HPSS.conf for ftp and pftp

The HPSS pftp daemon requires some changes to the format of the `HPSS.conf` file in `/var/hpss/etc`. Sites are encouraged to use the new default `HPSS.conf` in `$HPSS_ROOT/config/templates/HPSS.conf.tmpl` and merge original `HPSS.conf` settings into the default template.

2.6.6. Updating the HPSS server ACLs for New Dynamic Device Config Feature

Sites will need to update all mover server ACLs and PVR server ACLs by adding entries for the PVL server to them in order to create, update or delete devices in HPSS 7.1.0 after the conversion. To accomplish this, sites should use the **hpss_server_acl** program. Specific commands are included below:

```
% hpss_server_acl
hsa> acl -n "<mover server name 1>" -d "Administrative Interface"
hsa> add user hpsspvl rtc

hsa> acl -n "<pvr server name>"
hsa> add user hpsspvl rwcdt
```

Execute the above commands for each mover and PVR in the HPSS system. To discover the name of your mover and PVR servers (the argument to -n) sites can execute:

```
% lshpss -svr
```

The descriptive names of servers is in the first column.

2.6.7. HPSS 7.1.0 System Startup and Testing

After successful installation of Java, sites should startup HPSS 7.1.0 servers and ensure they can both read and write to the system.

If the HPSS 7.1.0 system does not function properly, sites should contact their IBM HPSS support representative.

Upon certifying that the HPSS 7.1.0 system functions properly sites should take database backups to ensure that they can recover to this point-in-time.

This is the point at which sites should upgrade to AIX 6.1. See the AIX documentation to accomplish this. HPSS 7.1.0 will run on AIX 5.3 as well, but AIX 6.1 is the official software prerequisite for HPSS 7.1.0.

2.6.8. Tuning for Replication of Large HPSS Systems

These steps are primarily intended for sites with 100 million records or more in a table or when overall conversion time is critical to minimize. Or for sites with extremely high HPSS 6.2 database activity that intend to keep the HPSS 6.2 system active with backups and other normal production activities during the replication or conversion process.

Note that database tuning for LOAD performance will affect the performance of the database under normal HPSS operation and may cause significantly poorer performance in the running HPSS 6.2 system.

Some of the changes provided below may change how DB2 accesses HPSS metadata in the source (HPSS 6.2) tables and cause HPSS servers significant problems like deadlock which will cause user operations to retry or fail. At the least, as soon as the LOAD utilities (full refresh) operation is complete the site should return to previous database settings.

2.6.8.1. Optimizing Replication LOAD and DB2 Backups

The first database configuration parameter to tune for each subsystem database is the UTIL_HEAP_SZ. The LOAD utility will use memory from the utility heap, up to 50%, to operate. The BACKUP and RESTORE utilities also use the utility heap. A site with regular intensive backups should increase this parameter significantly to accommodate the load utilities during replication until the full refresh is complete.

The SORTHEAP database configuration parameter specifies the number of pages per index being rebuilt during the second phase of a LOAD operation. If the SORTHEAP is set too low, then the the LOAD will utilize the default buffer pool for the system temporary tablespace (normally IBMDEFAULTBP) and if this value is still to small for the size of the index, then it will resort to disk I/O to the system temporary tablespace which will drastically slowdown the LOAD process.

Although the above database configuration parameters may be changed dynamically, the LOAD utility will not pick up the change until it disconnects from the database and reconnects.

The site will want to consider increasing the size of the buffer pools for all system temporary tablespaces to prevent overflow to the system temporary tablespace containers for index rebuilding during LOAD.

Ensure that the file `/var/hpss/convert/7.1/asnload.ini` exists. This is where the LOAD utility will look for options other than defaults for performing replication with the conversions.

All changes to this file should be made under the [COMMON] stanza to allow all ASNLOADs for all databases (subsystems) to pickup the settings unless specifically intended to tune one subsystem/ database over another.

Do not remove the "copy=no" option as it prevents the LOAD utility from maintaining costly roll forward recovery and load restart capabilities. This is unnecessary as sites can easily restart or redo the conversion process with the given utilities with HPSS 6.2 in operation.

Other options to consider adding to the `/var/hpss/convert/7.1/asnload.ini` file are:

Option	**Details**
disk_parallelism=n	Where n is the number of threads the LOAD uses. By default this is set to the number of containers the tablespace has for the table being loaded. Increasing this up to 4x the number of CPUs or 50 total is the maximum setting and can drastically improve LOAD performance.
data_buffer_size=n	Where n is the number of extents the LOAD will use to load rows into the target table. This can have drastic effects on LOAD performance but should not be set above 50% of the UTIL_HEAP_SZ for the database.

2.6.8.2. Tuning Replication for High HPSS 6.2 Activity

For sites with intense HPSS 6.2 database activity during the conversion, one issue that may arise is the default 5-minute capture and apply cycles not keeping pace with the rate of change in the database. Sites may want to change to continuous replication intervals for the capture and apply programs. This will increase overall activity with the database and is only advised if the replication monitor (**hpss_convert_71_subsys_monitor**) reports alert notifications for various thresholds being reached for the capture or apply programs. To change the intervals to continuous: Stop the capture and apply programs by using **hpss_convert_71**, and select menu option 4 "Stop replication for a subsys db". Then manually update the intervals in the database like such:

```
% db2 connect to <DBNAME>
% db2 update ASN.IBMSNAP_CAPPARMS set COMMIT_INTERVAL=1,
PRUNE_INTERVAL=1, MONITOR_INTERVAL=1
% db2 update ASN.IBMSNAP_APPPARMS set DELAY=1, ERRWAIT=1
% db2 update ASN.IBMSNAP_MONPARMS set MONITOR_INTERVAL=5
% db2 update ASN.IBMSNAP_CONDITIONS set PARM_INT=1
```

Then restart the capture and apply programs by using **hpss_convert_71**, and select menu option 3 "Start replication for a subsys db".

Chapter 3. HPSS 7.1.0 to HPSS 7.1.2 Conversion

This chapter summarizes HPSS conversion from HPSS 7.1.0 to HPSS 7.1.2. These steps also apply to 7.1.1, but for simplicity only 7.1.0 will be referred to in this section.

3.1. General Information

The metadata conversion from HPSS 7.1.0 to 7.1.2 is a minor one. Several indexes will be recreated with the intent of improving performance of some operations. There are no structural changes associated with this conversion, and it should be easily reversible.

3.2. Metadata Changes

This chapter summarizes HPSS changes for Release 7.1.2 only as they relate to metadata alterations.

3.2.1. New Features Affecting Conversion

This section describes the new HPSS features that result in metadata transformations from HPSS Release 7.1.0.

No new features.

3.2.1.1. New Tables, Views and Constraints

No new tables, views, or constraints.

3.2.2. Metadata Change Detail

3.2.2.1. HPSS Subsystem Database Changes

Table Name	**Change**
STORAGESEGTAPEABSADDR	Remove SSID from Index.
VVTAPEABSADDR	Remove VVID from Index.

3.3. Introduction

The following chapter describes the database schema changes that must be made to convert an HPSS 7.1.0 system to an HPSS 7.1.2 system.

Please read the section hpss_managetables Section 1.4, "hpss_managetables" for a warning and guidance concerning the usage of the ***hpss_managetables*** *tool before continuing.*

3.3.1. Drop SSID from STORAGESEGTAPEABSADDR Index

```
% hpss_managetables
hmt>
```

Connect to the database to be modified. In this example we will use "subsys1":

```
hmt> db subsys1
```

The program will connect to the database and show you information about the connection. Take this opportunity to verify that the database and schema names are correct.

Drop the indexes from the STORAGESEGTAPEABSADDR table:

```
hmt> del indexes storagesegtapeabsaddr
```

The program will remove the existing index from STORAGESEGTAPEABSADDR.

```
hmt> add indexes storagesegtapeabsaddr
```

The program will add indexes to STORAGESEGTAPEABSADDR and show you the SQL commands executed. The indexes will be added to the same tablespace as was defined originally by the table.

Commit your work and terminate the program:

```
hmt> commit
....
hmt> quit
```

Repeat these steps for each subsystem database in your HPSS system, altering the "db" command to the appropriate subsystem database name.

3.3.2. Drop VVID from VVTAPEABSADDR Index

```
% hpss_managetables
hmt>
```

Connect to the database to be modified. In this example we will use "subsys1":

```
hmt> db subsys1
```

The program will connect to the database and show you information about the connection. Take this opportunity to verify that the database and schema names are correct.

Drop the indexes from the VVTAPEABSADDR table:

```
hmt> del indexes vvtapeabsaddr
```

The program will remove the existing index from VVTAPEABSADDR.

```
hmt> add indexes vvtapeabsaddr
```

The program will add indexes to VVTAPEABSADDR and show you the SQL commands executed. The indexes will be added to the same tablespace as was defined originally by the table.

Commit your work and terminate the program:

```
hmt> commit
....
hmt> quit
```

Repeat these steps for each subsystem database in your HPSS system, altering the "db" command to the appropriate subsystem database name.

Chapter 4. HPSS 7.1.2 to HPSS 7.3.1 Conversion

This chapter summarizes HPSS conversion from HPSS 7.1.2 to HPSS 7.3.1.

4.1. General Information

The conversion from 7.1.2 to 7.3.1 includes table changes, new tables, new constraints, and new indexes. Additionally, there are changes to the storage segment layouts which will require a conversion of every storage segment in the system. This conversion uses several scripts to make the process simpler.

4.2. Metadata Changes

This chapter summarizes HPSS changes for Release 7.1.2 only as they relate to metadata alterations.

4.2.1. New Features Affecting Conversion

This section describes the new HPSS features that result in metadata transformations when converting from HPSS Release 7.1.0 or 7.1.1. All of these instructions apply to both 7.1.0 and 7.1.1, but for simplicity we will only refer to 7.1.0 for the remainder of this section.

4.2.1.1. User-Defined Attributes

The user-defined attributes (UDA) feature can be used to associate multiple pieces of metadata with a namespace object in a flexible way. The UDAs are stored in the database as XML. UDA supports a simple set of APIs which enables users to set, get, delete, and list the UDAs for a particular namespace object without any knowledge of XML. UDA also allows XML savvy users to leverage XML/XQuery with the advanced API set. The advanced APIs allow users to craft their own XQueries to retrieve or update their user-defined attributes the way they want to.

4.2.1.2. Multi-Homed Passive Mover

Enhancements have been made to the HPSS mover and configuration data to support the use of multiple IP addresses for passive side IP endpoints. This removes the restriction that all data movement for a passive mover use a single network adapter. By utilizing this feature, a single mover should be able to manage multiple tape drives whose aggregate data rate exceeds the bandwidth of a single network adapter.

4.2.1.3. New Tables, Views and Constraints

The following tables are new to HPSS 7.3.0:

Table Name	Purpose
USERATTRS	Holds records for user-defined metadata associated with namespace objects.

The following constraints are new to HPSS 7.3.0:

Constraint Name	Purpose
NSOBJECTUSERATTRS1	Links USERATTRS with NSOBJECT, providing a foreign key for OBJECT_ID and cascading deletion of UDAs.

4.2.2. Metadata Change Detail

4.2.2.1. HPSS Configuration Database Changes

Table Name	Change
MOVER	Add support for multi-homed passive movers; change Mover specific configuration to allow multiple host names.

4.2.2.2. HPSS Subsystem Database Changes

Table Name	Change
STORAGESEGDISKEXTENTS	Reformatted to use STARTING_OFFSET and ENDING_OFFSET rather than OFFSET and LENGTH.
USERATTRS	Added.
NSOBJECTUSERATTRS1	Added.

4.3. Upgrading from 7.1.2 to 7.3.1

The following chapter describes the database schema changes that must be made to convert an HPSS 7.1.2 system to an HPSS 7.3.1 system.

Please read the section hpss_managetables Section 1.4, "hpss_managetables" for a warning and guidance concerning the usage of the ***hpss_managetables*** *tool before continuing.*

4.4. User-Defined Attributes

4.4.1. Conversion Procedure

The USERATTRS table provides metadata storage space for the User Defined Attributes feature which was added to HPSS in version 7.3.0.

Adding the USERATTRS table is done by performing the following steps:

1. Add the USERATTRS table to all subsystem databases
2. Adjust schema settings associated with the NSOBJECT table in all subsystem databases
3. Add Relational Integrity constraints associated with the USERATTRS table to all subsystem database schemata.

The schema changes for the user-defined attributes feature are backward compatible with HPSS 7.1.X, so these changes can be made well in advance of moving a production system to 7.3.1. Changes to a subsystem database schema must be performed while HPSS is not running. All HPSS system servers including the System Manager and the Startup Daemon must be halted before making changes.

This conversion is performed on the subsystem database. These instructions should be repeated for each subsystem database in your HPSS system. The global (or "cfg") database is not changed in this conversion. These are the steps:

Start **hpss_managetables**:

```
$ hpss_managetables
hmt>
```

Connect to the database to be modified. In this example we will use "subsys1":

```
hmt> db subsys1
```

The program will connect to the database and show you information about the connection. Take this opportunity to verify that the database and schema names are correct.

Create the USERATTRS table:

```
hmt> add userattrs:<userattrs_tablespace>:<userattrs_index_tablespace>
```

The program will create the USERATTRS table and show you the SQL commands it executed. The correct syntax is the tablename, the tablespace for the table, and the tablespace for the indexes with colons between. Also, the **ts** command may be used to set the default tablespace for tables and indexes. If there are no errors, commit this change to the database:

```
hmt> commit
```

Next, we want to add a Relational Integrity constraint to the schema, binding the NSOBJECT and USERATTRS tables together. But to to do this, it may be necessary to make a small change to the settings on the NSOBJECT table. Most systems that were created before HPSS 7.1 will need this change. Make this change with this command:

```
hmt> add indexes nsobject
```

This command will run four SQL statements on the NSOBJECT table. Some of these statements will attempt to create indexes that already exist. The program intercepts this error and displays a message telling you that the index was not created and that this situation is not considered to be an error. Any real errors are clearly described.

One of the four SQL commands adds a "primary key" to the NSOBJECT table. Adding the primary key to the table is a change we need to make in order to take the next step. In most cases this change will occur with no errors. If there is a problem, it will be clearly spelled out by **hpss_managetables**. If an error occurs, "abort" the transaction, "quit" the program and get help.

Now we are ready to add the Relational Integrity constraint:

```
hmt> add subsys constraints
```

Again, a number of SQL commands will be issued. Many will be followed by a comment from **hpss_managetables** indicating that the constraint already existed. The last constraint, NSOBJECTUSERATTRS1, should be applied without an error or comment.

Commit your work and terminate the program:

```
hmt> commit
....
hmt> quit
```

Repeat these steps for each subsystem database in your HPSS system, altering the "db" command to the appropriate subsystem database name.

4.5. Mover Metadata Conversion

4.5.1. Conversion Procedure

The Multi-Homed Passive Mover Support feature requires a conversion of the Mover specific configuration table. This conversion must be performed before starting any Movers.

The procedure for running this conversion has been simplified into a single script.

To run this conversion you must first compile the 7.3.1 source, ensure that HPSS is not running, and that DB2 is running.

Next, run the conversion script as root by simply doing the following:

```
% /opt/hpss/tools/convert71/hpss_convert_73_mvr_config
```

Run the script and note any errors. If something unexpected happens and you get an error, you will have to seek advice. Or, if you are unsure how to interpret the output, seek advice from HPSS support.

4.6. Disk Storage Segment Extents Metadata Conversion

4.6.1. Conversion Preparation

PTR 7786 introduced a change to the format of the STORAGESEGDISKEXTENTS table. This PTR is a part of the 7.1.2 and 7.3.1 HPSS releases. To run these systems, the STORAGESEGDISKEXTENTS table must be converted to a new format. This conversion must be performed on all existing systems on which the 7.3.1 or following releases are installed, even if the system in question does not have a disk cache.

The basics of the conversion are simple. The existing STORAGESEGDISKEXTENTS table is renamed, a new table is created in the new format, the contents of the old table are copied to the new table, and the 7.3.1 system is started. If all goes well, the old table is removed.

The STORAGESEGDISKEXTENTS table can be a large table. If the installation in question has a large disk cache, this table will have at least one row for each file in the cache. Installations with a small or non-existent disk cache will have short or empty tables. Before performing this conversion, consider the amount of free space in the tablespace that contains the table. It may be necessary to expand that tablespace. Since expanding a tablespace can be a task that is highly specific to the installation, this document does not describe that process.

The conversion process is designed so that if a problem should occur, the conversion can be aborted and the metadata put back the way it was. The original metadata table is retained. This requires that the system source code be put back to the previous version as well. Should this become necessary, you will have to seek advice - the exact procedure to put things back as they were depends on what failed.

4.6.2. Conversion Procedure

4.6.2.1. Verify Prerequisite Space

The first step is to validate the free space in the impacted tablespaces. At a minimum you need at least 50% free space in STORAGESEGDISK and INDEXES tablespaces. The temporary tablespace (usually TEMPSPACE1) should have sufficient space allocated to construct the indexes for the new table (this will vary by site). The script to verify this is named **check_tablespaces.sh** (located in (`HPSS_ROOT)/bin/check_tablespaces.sh`) and it has three arguments:

```
check_tablespaces.sh [-d subsystem database name] [-s schema name] [-t
Temporary tablespace name]
```

By default the script will use "HSUBSYS1" as the database name, "HPSS" as the schema name, and "TEMPSPACE1" as the temporary tablespace name. Typical command lines are:

```
check_tablespaces.sh -d hsubsys1
```

Checks the tablespaces in database "HSUBSYS1", uses the schema "HPSS" to acquire the tablespaces used for the STORAGESEGDISKEXTENTS table data and indexes, and will gather the file system space for the container assigned to tablespace "TEMPSPACE1".

```
check_tablespaces.sh -d hsubsys2 -t tempspace2
```

Checks the tablespaces in database "HSUBSYS2", uses the schema "HPSS" to acquire the tablespaces used for the STORAGESEGDISKEXTENTS table, and will gather the filesystem space for the container assigned to tablespace "TEMPSPACE2".

```
check_tablespaces.sh -s hpss
```

Checks the tablespaces in database "HSUBSYS1", uses the schema "HPSS" to acquire the tablespaces used for the STORAGESEGDISKEXTENTS table, and will gather the filesystem space for the container assigned to tablespace "TEMPSPACE1".

Example output:

```
$ /opt/hpss/tools/metadata/db2/check_tablspaces.sh

02/15/2010 15:47:18 connect to HSUBSYS1
02/15/2010 15:47:19 DB20000I The SQL command completed successfully.

02/15/2010 15:47:19 select tbspace from syscat.tables where
tabschema='HPSS' and tabname='STORAGESEGDISKEXTENTS'
02/15/2010 15:47:19 DB20000I The SQL command completed successfully.

```

```
02/15/2010 15:47:19 select TBSP_UTILIZATION_PERCENT from
sysibmadm.tbsp_utilization where TBSP_NAME='STORAGESEGDISK '
02/15/2010 15:47:19 DB20000I The SQL command completed successfully.

02/15/2010 15:47:20 Data tablespace --> STORAGESEGDISK
02/15/2010 15:47:20 utilization percent = 6.13

02/15/2010 15:47:20 select distinct substr(b.tbspace,1,20) from
syscat.indexes a, syscat.tablespaces b where a.tabschema='HPSS' and
a.tabname='STORAGESEGDISKEXTENTS' and a.tbspaceid=b.tbspaceid
02/15/2010 15:47:20 DB20000I The SQL command completed successfully.

02/15/2010 15:47:20 select TBSP_UTILIZATION_PERCENT from
sysibmadm.tbsp_utilization where TBSP_NAME='STORAGESEGDISKIDX   '
02/15/2010 15:47:20 DB20000I The SQL command completed successfully.

02/15/2010 15:47:20 Index tablespace --> INDEXES
02/15/2010 15:47:20 utilization percent = 4.59

02/15/2010 15:47:20
02/15/2010 15:47:20 tempspace tablespace --> TEMPSPACE1
02/15/2010 15:47:20
tempspace_cont=/var/hpss/hpssdb/hpssdb/NODE0000/HSUBSYS1/T0000001/C0000000.TMP
02/15/2010 15:47:20 df -khP
/var/hpss/hpssdb/hpssdb/NODE0000/HSUBSYS1/T0000001/C0000000.TMP
Filesystem         1024-blocks      Used Available Capacity Mounted on
/dev/mapper/VolGroup00-LogVol00 234476168 36858064 185515244       17%
DB20000I The SQL command completed successfully.

connect reset
DB20000I The SQL command completed successfully.

terminate
DB20000I The TERMINATE command completed successfully.
```

This script is provided as a guide to assist in evaluating the current utilization of the tablespaces being used by the affected table. The script shows you the commands as they are run and will stop if any command gets an error. Run the script and note any errors. If something unexpected happens and you get an error, you will have to seek advice. Or, if you are unsure as to how to interpret the output seek advice from HPSS support.

4.6.2.2. Perform the Conversion

Before proceeding with the conversion, completely stop HPSS. Stop all HPSS processes and servers. Leave DB2 running. The conversion script will try to connect in "EXCLUSIVE MODE". The connect step will fail if there are any other connection to the target database. Run the script and note any errors. If something unexpected happens and you get an error, you will have to seek advice. Keep in mind that the original data is preserved and can be set back to its original state.

The second step constructs the new table and loads it with the data from the original table and makes the table ready for use. This step is performed by a script that runs the DB2 CLP. This script is called **cnvrt_storagesegdiskextents.sh** (located in `(HPSS_ROOT)/bin/cnvrt_storagesegdiskextents.sh`) and it has four arguments:

```
cnvrt_storagesegdiskextents.sh [-d subsystem database name (default
'HSUBSYS1')] [-s schema name (default 'HPSS')] [-c provide before and
after row counts] [-i flag to drop original table indexes]
```

By default, the script converts subsystem 1 in schema "hpss" and will not provide before/after row counts and will not drop the indexes from the original storagesegdiskextents table. The -c option, if specified, will execute "select count(*) from storagesegdiskextents" before and after the conversion. If you have several million rows this may take several minutes each time it is executed. The -i option will drop all of the indexes from the original table. If selected, the recovery time to "undo" the conversion will be increased as the original indexes (along with the foreign key constraint) will have to be recreated. It may be necessary to use the "-i" option if there is not enough additional space in the indexes tablespace to have 2 copies of indexes: one for the original table and one for the new table.

Typical command lines are:

```
cnvrt_storagesegdiskextents.sh
```

Converts subsystem 1 (database HSUBSYS1) using schema "HPSS" and does not provide before/after row counts and does not drop the indexes from the original table.

```
cnvrt_storagesegdiskextents.sh -d hsubsys2
```

Converts subsystem 2 (database HSUBSYS2) using schema "HPSS" and **does not** provide before/after row counts and **does not** drop the indexes from the original table.

```
cnvrt_storagesegdiskextents.sh -i -c
```

Converts subsystem 1 (database HSUBSYS1) using schema "HPSS" and **does** provide before/after row counts and **does** drop the indexes from the original table.

The script shows you the commands as they are run and will stop if any command gets an error. The "load from cursor", "alter table…add foreign key" and "runstats" commands may take some time to complete in large systems.

When the script completes normally, the metadata tables are ready for HPSS to be started.

Example output from a successful conversion:

```
$ /opt/hpss/tools/metadata/db2/cnvrt_storagesegdiskextents.sh -i -c

02/15/2010 16:23:19 Step --> connect

02/15/2010 16:23:17 connect to HSUBSYS1 in exclusive mode
02/15/2010 16:23:19 DB20000I The SQL command completed successfully.

02/15/2010 16:23:19 Step --> pre_conversion_check

02/15/2010 16:23:19 select count(*) from syscat.columns where
colname='ENDING_OFFSET' and tabschema='HPSS' and
tabname='STORAGESEGDISKEXTENTS'
02/15/2010 16:23:19 DB20000I The SQL command completed successfully.
02/15/2010 16:23:19 Column ENDING_OFFSET does not exist in table
HPSS.STORAGESEGDISKEXTENTS, conversion may proceed.

02/15/2010 16:23:19 Step --> before_conversion

02/15/2010 16:23:19 Step --> describe_table
02/15/2010 16:23:19 db2 describe table HPSS.STORAGESEGDISKEXTENTS

                           Data type                     Column
Column name                schema    Data type name      Length     Scale Nulls
-------------------------- --------- ------------------- ---------- ----- ------
```

```
SSID                           SYSIBM    CHARACTER          32             0 No
VVID                           SYSIBM    CHARACTER          32             0 Yes
ORDINAL                        SYSIBM    INTEGER             4             0 No
OFFSET                         SYSIBM    INTEGER             4             0 Yes
LENGTH                         SYSIBM    INTEGER             4             0 Yes

  5 record(s) selected.


02/15/2010 16:23:19 Step --> list_indexes
02/15/2010 16:23:20 db2 select substr(rtrim(indschema) || '.' ||
rtrim(indname),1,50) as index_name, substr(colnames,1,60) as
columns,uniquerule from syscat.indexes where tabschema='HPSS' and
tabname='STORAGESEGDISKEXTENTS'

INDEX_NAME                         COLUMNS                              UNIQUERULE
---------------------------------- ------------------------------------ ----------
HPSS.STORAGESEGDISKEXTENTS_PKEY    +SSID+ORDINAL                        P
HPSS.VVID_EXTENTS                  +VVID+OFFSET+LENGTH                  D

  2 record(s) selected.


02/15/2010 16:23:20 Step --> list_constraints
02/15/2010 16:23:20 db2 select substr(a.constname,1,30) as
constname,a.type,substr(rtrim(b.tabschema) || '.' ||
rtrim(b.tabname),1,35) as child_table_name, substr(rtrim(b.reftabschema)
|| '.' || rtrim(b.reftabname),1,30) as prnt_table_name from
syscat.tabconst a left outer join syscat.references b on
a.constname=b.constname and a.tabname=b.tabname where a.tabschema='HPSS'
and a.tabname='STORAGESEGDISKEXTENTS'

CONSTNAME                  TYPE CHILD_TABLE_NAME           PRNT_TABLE_NAME
-------------------------- ---- -------------------------- -------------------
DISKSEGEXTENTSCON1         F    HPSS.STORAGESEGDISKEXTENTS  HPSS.STORAGESEGDISK
STORAGESEGDISKEXTENTS_PKEY P    -                          -

  2 record(s) selected.


02/15/2010 16:23:20 Step --> list_row_counts
02/15/2010 16:23:20 db2 select count(*) as rowcount from
HPSS.STORAGESEGDISKEXTENTS

ROWCOUNT
-----------
       2991

  1 record(s) selected.


02/15/2010 16:23:20 Step --> tblspace_data

02/15/2010 16:23:20 select tbspace from syscat.tables where
tabschema='HPSS' and tabname='STORAGESEGDISKEXTENTS'
02/15/2010 16:23:20 DB20000I The SQL command completed successfully.
02/15/2010 16:23:20 Data tablespace --> STORAGESEGDISK

02/15/2010 16:23:20 Step --> tblspace_index
```

```
02/15/2010 16:23:20 select distinct substr(b.tbspace,1,20) from
syscat.indexes a, syscat.tablespaces b where a.tabschema='HPSS' and
a.tabname='STORAGESEGDISKEXTENTS' and a.tbspaceid=b.tbspaceid
02/15/2010 16:23:20 DB20000I The SQL command completed successfully.
02/15/2010 16:23:20 Index tablespace --> STORAGESEGDISKIDX

02/15/2010 16:23:21 Step --> get_foreign_key_name

02/15/2010 16:23:21 select substr(constname,1,30) from syscat.tabconst
where tabschema='HPSS' and tabname='STORAGESEGDISKEXTENTS' and type='F'
02/15/2010 16:23:21 DB20000I The SQL command completed successfully.
02/15/2010 16:23:21 Foreign Key constraint name --> DISKSEGEXTENTSCON1

02/15/2010 16:23:21 Step --> drop_foreign_key

02/15/2010 16:23:21 alter table HPSS.storagesegdiskextents drop foreign
key DISKSEGEXTENTSCON1
02/15/2010 16:23:21 DB20000I The SQL command completed successfully.

02/15/2010 16:23:21 Step --> drop_view

02/15/2010 16:23:21 drop view HPSS.STORAGESEGDISKVIEW
02/15/2010 16:23:21 DB20000I The SQL command completed successfully.

02/15/2010 16:23:21 Step --> rename_orig_table

02/15/2010 16:23:21 rename table HPSS.storagesegdiskextents to
storagesegdiskextents_orig
02/15/2010 16:23:21 DB20000I The SQL command completed successfully.

02/15/2010 16:23:21 Step --> manage_orig_indexes

02/15/2010 16:23:21 alter table HPSS.storagesegdiskextents_orig drop
primary key
02/15/2010 16:23:21 DB20000I The SQL command completed successfully.

02/15/2010 16:23:21 select 'drop index ' || substr(rtrim(indschema) ||
'.' || rtrim(indname),1,50) || ';' from syscat.indexes where
tabschema='HPSS' and tabname='STORAGESEGDISKEXTENTS_ORIG'
02/15/2010 16:23:21 DB20000I The SQL command completed successfully.

02/15/2010 16:23:21 drop index HPSS.VVID_EXTENTS
02/15/2010 16:23:22 DB20000I The SQL command completed successfully.

02/15/2010 16:23:22 Step --> create_new_table

02/15/2010 16:23:22 CREATE TABLE HPSS.STORAGESEGDISKEXTENTS_NEW (SSID
CHAR(32) FOR BIT DATA NOT NULL , VVID CHAR(32) FOR BIT DATA , ORDINAL
INTEGER NOT NULL , STARTING_OFFSET INTEGER , ENDING_OFFSET INTEGER ) IN
STORAGESEGDISK INDEX IN STORAGESEGDISKIDX
02/15/2010 16:23:22 DB20000I The SQL command completed successfully.

02/15/2010 16:23:22 Step --> create_new_indexes

02/15/2010 16:23:22 CREATE UNIQUE INDEX HPSS.SSID_EXTENTS ON
HPSS.STORAGESEGDISKEXTENTS_NEW (SSID ASC, ORDINAL ASC) ALLOW REVERSE
SCANS
02/15/2010 16:23:22 DB20000I The SQL command completed successfully.
```

```
02/15/2010 16:23:22 CREATE INDEX HPSS.VVID_EXTENTS_OFFSETS ON
HPSS.STORAGESEGDISKEXTENTS_NEW (VVID ASC, STARTING_OFFSET ASC,
ENDING_OFFSET ASC) ALLOW REVERSE SCANS
02/15/2010 16:23:22 DB20000I The SQL command completed successfully.

02/15/2010 16:23:22 Step --> add_new_primary_key

02/15/2010 16:23:22 ALTER TABLE HPSS.STORAGESEGDISKEXTENTS_NEW ADD
CONSTRAINT STORAGESEGDISKEXTENTS_PKEY PRIMARY KEY (SSID,ORDINAL)
02/15/2010 16:23:22 SQL0598W Existing index "HPSS.SSID_EXTENTS" is used
as the index for the primary key or a unique key. SQLSTATE=01550

02/15/2010 16:23:22 Step --> declare_cursor

02/15/2010 16:23:22 declare segcurs cursor for select
ssid,vvid,ordinal,offset,offset+length-1 from
HPSS.storagesegdiskextents_orig
02/15/2010 16:23:22 DB20000I The SQL command completed successfully.

02/15/2010 16:23:22 Step --> load_new_table

02/15/2010 16:23:22 load from segcurs of cursor insert into
HPSS.storagesegdiskextents_new nonrecoverable
02/15/2010 16:23:23 DB20000I The LOAD command completed successfully.

02/15/2010 16:23:23 Step --> rename_new_table

02/15/2010 16:23:23 rename table HPSS.storagesegdiskextents_new to
storagesegdiskextents
02/15/2010 16:23:23 DB20000I The SQL command completed successfully.

02/15/2010 16:23:23 Step --> add_foreign_key

02/15/2010 16:23:23 ALTER TABLE HPSS.STORAGESEGDISKEXTENTS ADD
CONSTRAINT DISKSEGEXTENTSCON1 FOREIGN KEY (SSID) REFERENCES
HPSS.STORAGESEGDISK (SSID) ON DELETE CASCADE ON UPDATE NO ACTION
ENFORCED ENABLE QUERY OPTIMIZATION
02/15/2010 16:23:23 DB20000I The SQL command completed successfully.

02/15/2010 16:23:23 Step --> create_insert_trigger

02/15/2010 16:23:23 CREATE TRIGGER HPSS.CHECK_OVERLAP_BEFORE_INSERT NO
CASCADE BEFORE INSERT ON HPSS.STORAGESEGDISKEXTENTS REFERENCING NEW AS N
FOR EACH ROW MODE DB2SQL WHEN ( 0 < (select count(*) from
HPSS.storagesegdiskextents where vvid=N.vvid and N.starting_offset <=
ending_offset and N.ending_offset >= starting_offset) ) BEGIN ATOMIC
SIGNAL SQLSTATE '99001' set MESSAGE_TEXT = 'segment overlap fault'; END

02/15/2010 16:23:23 DB20000I The SQL command completed successfully.

02/15/2010 16:23:23 Step --> create_update_trigger

02/15/2010 16:23:23 CREATE TRIGGER HPSS.CHECK_OVERLAP_BEFORE_UPDATE NO
CASCADE BEFORE UPDATE OF SSID,VVID,ORDINAL,starting_OFFSET,ENDing_OFFSET
ON HPSS.STORAGESEGDISKEXTENTS REFERENCING NEW AS N FOR EACH ROW MODE
DB2SQL WHEN ( 0 < (select count(*) from HPSS.storagesegdiskextents where
vvid=N.vvid and N.starting_offset <= ending_offset and N.ending_offset
>= starting_offset) ) BEGIN ATOMIC SIGNAL SQLSTATE '99001' set
```

```
MESSAGE_TEXT = 'segment overlap fault'; END

02/15/2010 16:23:23 DB20000I The SQL command completed successfully.

02/15/2010 16:23:23 Step --> recreate_view

02/15/2010 16:23:24 CREATE VIEW HPSS.STORAGESEGDISKVIEW AS SELECT
S.SSID, S.VVID, S.ALLOCATED_LENGTH, S.OWNER, S.CREATION, S.UPDATE,
M.SCLASS_ID, S.FILE_SYSTEM_ID, M.BLOCK_SIZE, M.CLUSTER_LENGTH,
E.ORDINAL, E.starting_OFFSET, E.ENDing_OFFSET FROM (HPSS.STORAGESEGDISK
AS S INNER JOIN HPSS.STORAGEMAPDISK AS M ON S.VVID = M.VVID) INNER JOIN
HPSS.STORAGESEGDISKEXTENTS AS E ON S.SSID = E.SSID
02/15/2010 16:23:24 DB20000I The SQL command completed successfully.

02/15/2010 16:23:24 Step --> rename_orig_table_complete

02/15/2010 16:23:24 rename table HPSS.storagesegdiskextents_orig to
storagesegdiskextents_orig_complete
02/15/2010 16:23:24 DB20000I The SQL command completed successfully.

02/15/2010 16:23:24 Step --> runstats

02/15/2010 16:23:24 runstats on table HPSS.storagesegdiskextents with
distribution and detailed indexes all
02/15/2010 16:23:24 DB20000I The RUNSTATS command completed
successfully.
02/15/2010 16:23:24 Step --> after_conversion

02/15/2010 16:23:24 Step --> describe_table
02/15/2010 16:23:24 db2 describe table HPSS.STORAGESEGDISKEXTENTS
                           Data type                     Column
Column name                schema    Data type name      Length     Scale Nulls
-------------------------- --------- ------------------- ---------- ----- ------
SSID                       SYSIBM    CHARACTER                   32     0 No
VVID                       SYSIBM    CHARACTER                   32     0 Yes
ORDINAL                    SYSIBM    INTEGER                      4     0 No
STARTING_OFFSET            SYSIBM    INTEGER                      4     0 Yes
ENDING_OFFSET              SYSIBM    INTEGER                      4     0 Yes

  5 record(s) selected.


02/15/2010 16:23:24 Step --> list_indexes
02/15/2010 16:23:24 db2 select substr(rtrim(indschema) || '.' ||
rtrim(indname),1,50) as index_name, substr(colnames,1,60) as
columns,uniquerule from syscat.indexes where tabschema='HPSS' and
tabname='STORAGESEGDISKEXTENTS'

INDEX_NAME                        COLUMNS                              UNIQUERULE
--------------------------------- ------------------------------------ ----------
HPSS.SSID_EXTENTS                 +SSID+ORDINAL                        P
HPSS.VVID_EXTENTS_OFFSETS         +VVID+STARTING_OFFSET+ENDING_OFFSET  D

  2 record(s) selected.


02/15/2010 16:23:24 Step --> list_constraints
02/15/2010 16:23:24 db2 select substr(a.constname,1,30) as
constname,a.type,substr(rtrim(b.tabschema) || '.' ||
rtrim(b.tabname),1,35) as child_table_name, substr(rtrim(b.reftabschema)
|| '.' || rtrim(b.reftabname),1,30) as prnt_table_name from
syscat.tabconst a left outer join syscat.references b on
a.constname=b.constname and a.tabname=b.tabname where a.tabschema='HPSS'
and a.tabname='STORAGESEGDISKEXTENTS'

CONSTNAME                  TYPE CHILD_TABLE_NAME            PRNT_TABLE_NAME
-------------------------- ---- --------------------------- --------------------
DISKSEGEXTENTSCON1         F    HPSS.STORAGESEGDISKEXTENTS  HPSS.STORAGESEGDISK
STORAGESEGDISKEXTENTS_PKEY P    -                           -

  2 record(s) selected.


02/15/2010 16:23:24 Step --> list_row_counts
02/15/2010 16:23:25 db2 select count(*) as rowcount from
HPSS.STORAGESEGDISKEXTENTS

ROWCOUNT
-----------
       2991

  1 record(s) selected.



exiting
commit
DB20000I The SQL command completed successfully.

connect reset
DB20000I The SQL command completed successfully.

terminate
DB20000I The TERMINATE command completed successfully.
```

Example output with errors:

```
$ /opt/hpss/tools/metadata/db2/cnvrt_storagesegdiskextents.sh -i -c
02/15/2010 16:28:06 connect to HSUBSYS1 in exclusive mode
02/15/2010 16:28:06 SQL1035N The database is currently in use.  SQLSTATE=57019
ERROR: There was at least one unexpected SQLCODE
```

One way to validate the conversion prior to attempting to use the table is to look at the describe table descriptions in DB2. These are shown in the above example output. The table should contain fields for "STARTING_OFFSET" and "ENDING_OFFSET" rather than "OFFSET" and "LENGTH".

4.6.3. Subsystems

If the installation has multiple subsystems, you must convert each of the STORAGESEGDISKEXTENTS tables in each of the subsystems. Repeat the appropriate procedure above modifying the "-d hsubsys1" commands parameter on the invocations of the **check_tablespaces.sh** and **cnvrt_storagesegdiskextents.sh** scripts appropriately.

4.6.4. Using the New Table

Once you have performed the conversion described above, the next step should be to verify the conversion is correct. You can do this by starting the HPSS system and noting if any errors are reported by the Core Server. Avoid starting MPS until you have verified all is well.

The Core Server checks the elements of the database schema as it is starting. If it detects any errors, it will not start. If the installation has any files in its disk cache, the Core Server will build the in-memory space maps associated with those files. If there are errors associated with this process, the server will send appropriate alarm messages. Plan on not opening the system to its users until the Core Server has informed you that all of the disk space maps have been built without error.

If anything is amiss, stop the system and seek advice. Do not attempt to correct errors associated with this conversion yourself. This conversion process has been carefully tested and we don’t expect problems with it, but don’t attempt to correct problems yourself.

When you have established that the system is running properly, you will need to decide when to remove the old metadata table, "storagesegdiskextents_orig_complete". Once disk files are created or destroyed using the new metadata table, the old table cannot be used as a "fall back". Installations with a large number of disk files will probably want to recover the space occupied by the old table.

Remove the old table by using the DB2 CLP. Give the following commands:

```
connect to subsysx
drop table hpss.storagesegdiskextents_orig_complete
terminate
```

Since the name "storagesegdiskextents_orig_complete" is not a recognized HPSS metadata table name, you can’t harm the system by dropping it, even when the system is running.

Chapter 5. HPSS 7.3.1 to HPSS 7.3.2 Conversion

This chapter summarizes HPSS conversion from HPSS 7.3.1 to HPSS 7.3.2.

5.1. General Information

The conversion from 7.3.1 to 7.3.2 involves deleting and adding tables.

5.2. Metadata Changes

This chapter summarizes HPSS changes for Release 7.3.2 only as they relate to metadata alterations.

5.2.1. New Features Affecting Conversion

This section describes the new HPSS features that result in metadata transformations from HPSS Release 7.3.1 and 7.3.2.

5.2.1.1. Separate Disk and Tape Unlink Tables

In order to improve the performance for some systems, the segment unlink table has been split into disk and tape segment unlink tables.

5.2.1.2. New Tables, Views and Constraints

The following tables are new to HPSS 7.3.2:

Table Name	**Purpose**
DISKSEGUNLINK	Store disk segment records which are waiting to be unlinked.
TAPESEGUNLINK	Store tape segment records which are waiting to be unlinked.

5.2.2. Metadata Change Detail

5.2.2.1. HPSS Subsystem Database Changes

Table Name	**Change**
BFSSUNLINK	Removed.
DISKSEGUNLINK	Added.
TAPESEGUNLINK	Added.

5.3. Introduction

The following chapter describes the database schema changes that must be made to convert an HPSS 7.3.1 system to a HPSS 7.3.2 system.

*Please read the section hpss_managetables Section 1.4, “hpss_managetables” for a warning and guidance concerning the usage of the **hpss_managetables** tool before continuing.*

5.4. Convert Unlink Tables

5.4.1. Introduction

The new unlink tables improve the delete performance of HPSS.

Conversion of an HPSS 7.3.1 system to version 7.3.2 is done by performing the following steps:

1. Validate that the existing unlink tables are empty
2. Drop the BFSSUNLINK table
3. Add the new TAPESEGUNLINK and DISKSEGUNLINK tables

To revert the conversion, drop the new (empty) TAPESEGUNLINK and DISKSEGUNLINK tables and add the BFSSUNLINK table. The program that makes the changes is **hpss_managetables**.

5.4.2. Conversion Procedure

This conversion is performed on the subsystem database. These instructions should be repeated for each subsystem database in your HPSS system. The global (or "cfg") database is not changed in this conversion. These are the steps:

First, the BFSSUNLINK table must be empty. To validate this, connect to the database and run:

```
connect to subsysx
select count(*) from BFSSUNLINK
     1
-----------
          0

  1 record(s) selected.
quit
```

Start **hpss_managetables**:

```
$ hpss_managetables
hmt>
```

Connect to the database to be modified. In this example we will use "subsys1":

```
hmt> db subsys1
```

The program will connect to the database and show you information about the connection. Take this opportunity to verify that the database and schema names are correct.

Drop the BFSSUNLINK table:

```
hmt> del bfssunlink
```

Create the TAPESEGUNLINK table:

```
hmt> add
tapesegunlink:<tapesegunlink_tablespace>:<tapesegunlink_index_tablespace>
```

Create the DISKSEGUNLINK table:

```
hmt> add
disksegunlink:<disksegunlink_tablespace>:<disksegunlink_index_tablespace>
```

The program will create the new tables and show you the SQL commands it executed. The correct syntax is the tablename, the tablespace for the table, and the tablespace for the indexes with colons between. Also, the **ts** command may be used to set the default tablespace for tables and indexes. If there are no errors, commit these changes to the database:

```
hmt> commit

hmt> quit
```

Repeat these steps for each subsystem database in your HPSS system, altering the "db" command to the appropriate subsystem database name.

Chapter 6. HPSS 7.3.2 to HPSS 7.3.3 Conversion

This chapter summarizes HPSS conversion from HPSS 7.3.2 to HPSS 7.3.3.

6.1. General Information

The conversion from 7.3.2 to 7.3.3 includes index changes, table changes, and ACL modifications. This conversion uses several scripts to make the process simpler.

6.2. Metadata Changes

The following sections describe the HPSS metadata changes for Release 7.3.3.

6.2.1. New Features Affecting Conversion

This section describes the new HPSS features that result in metadata transformations from HPSS Release 7.3.2 to 7.3.3.

6.2.1.1. Improve Free Space Map Build Performance

The layout of the HPSS.SSDE_VVID_SO index on the HPSS.STORAGESEGDISKEXTENTS table will be changed to include the ENDING_OFFSET column. This change provides both offsets in a single index allowing the free space map building step to run much faster.

6.2.1.2. Unique Constraint for RPC Program Numbers

The EXECUTE_HOSTNAME column has been removed from index HPSS.SERVER_RPC_ID on the HPSS.SERVER table. This is needed to guarantee that RPC Program Numbers are unique throughout the HPSS realm.

6.2.1.3. VFS Checksum

Index HPSS.HPSS_VFS_USER (XML pattern index) on table HPSS.USERATTRS has been added to improve query performance.

6.2.1.4. PVR Device Conversion

To support SCSI Passthrough, thirty additional PVR Device fields have been added.

6.2.1.5. New Tables, Views and Constraints

There are no new tables.

6.2.2. Metadata Change Detail

6.2.2.1. HPSS Cfg Database Changes

Table Name	Change
PVR	Renamed column DEVICE_SPEC_A to DEVICE0. Renamed column DEVICE_SPEC_B to DEVICE1. Added thirty new columns DEVICE2 - DEVICE31.
SERVER	Removed column EXECUTE_HOSTNAME from index HPSS.SERVER_RPC_ID on table HPSS.SERVER.

6.2.2.2. HPSS Subsystem Database Changes

Table Name	Change
STORAGESEGDISKEXTENTS	Added column ENDING_OFFSET to index HPSS.SSDE_VVID_SO on table HPSS.STORAGESEGDISKEXTENTS.
USERATTRS	Added index HPSS.HPSS_VFS_USER on table HPSS.USERATTRS.

6.3. Upgrade Preparation

The following sections document the procedures that must be completed to convert an HPSS 7.3.2 system to an HPSS 7.3.3 system. Before attempting any HPSS upgrades, follow the preparation procedure in *Gather Data On Existing System Section 1.3.1, “Gather Data On Existing System”*, *Benchmark Existing System Section 1.3.2, “Benchmark Existing System”* and *Backup Existing System Section 1.3.3, “Backup Existing System”* in this guide.

Please read the section hpss_managetables Section 1.4, “hpss_managetables” for a warning and guidance concerning the usage of the ***hpss_managetables*** *tool before continuing.*

6.4. Setup Temporary Tablespace

6.4.1. Introduction

For the HPSS config database (cfg), HPSS 7.3.3 requires a database temporary table space with 8K pages to match the page size for the USERSPACE1 table space. The page size for the default temporary table space (TEMPSPACE1) in existing systems is typically 4K. In this step, you will determine the page size of the available temporary table spaces and create one with an 8K page size if none exists. To complete the following, HPSS should be down and DB2 should be running.

6.4.2. Conversion Procedure

1. Become the database instance owner, then list the table spaces for the config database.

   ```
   % su - hpssdb
   % db2 connect to hcfg
   % db2 list tablespaces show detail
   ```

   The sample output below shows a typical HPSS cfg database setup. It has been edited to remove extraneous table space definitions. Note the **Page size** for TEMPSPACE1 is 4096 bytes, while the **Page size** is 8192 bytes for USERSPACE1. Additionally, this database has no other TEMPSPACE table space definitions. If your system does not have a System Temporary tablespace with 8K pages, continue with step 2. Otherwise, skip to the next section.

   ```
             Tablespaces for Current Database

   Tablespace ID                        = 1
   Name                                 = TEMPSPACE1
   Type                                 = System managed space
   Contents                             = System Temporary data
   State                                = 0x0000
     Detailed explanation:
       Normal
   Total pages                          = 1
   Useable pages                        = 1
   Used pages                           = 1
   Free pages                           = Not applicable
   High water mark (pages)              = Not applicable
   Page size (bytes)                    = 4096
   Extent size (pages)                  = 32
   Prefetch size (pages)                = 32
   Number of containers                 = 1
   Tablespace ID                        = 2
   Name                                 = USERSPACE1
   Type                                 = System managed space
   Contents                             = All permanent data. Regular table space.
   State                                = 0x0000
     Detailed explanation:
       Normal
   Total pages                          = 206
   Useable pages                        = 206
   Used pages                           = 206
   Free pages                           = Not applicable
   High water mark (pages)              = Not applicable
   Page size (bytes)                    = 8192
   Extent size (pages)                  = 32
   Prefetch size (pages)                = 32
   Number of containers                 = 1
   ```

2. Determine the storage location to use for TEMPSPACE2 by listing the container for TEMPSPACE1. Use the **Tablespace ID** from the previous output. (Note that the output lines below have wrapped.)

   ```
   % db2 list tablespace containers for 1

               Tablespace Containers for Tablespace 1
   ```

```
Container ID                          = 0
Name                                  =
/var/hpss/hpssdb/hpssdb/NODE0000/SQL00001/SQLT0001.0
Type                                  = Path
```

3. Create a second temporary table space (TEMPSPACE2) with 8K page size. Use the directory portion of the **Name** from the previous output to place the temporary table space beneath the same directory as TEMPSPACE1. (Note that the command is a single line and has wrapped in the description below.)

```
% db2 "create temporary tablespace tempspace2 pagesize 8k managed by
system
using('/var/hpss/hpssdb/hpssdb/NODE0000/SQL00001/SQLT0001/tempspace2/')
bufferpool userspace1"

DB20000I The SQL command completed successfully.
```

4. Verify that TEMPSPACE2 exists and has been created as a "System Temporary Tablespace". System temporary tablespaces are identified by "Contents" that are set to "System Temporary Data". Edited output showing just information for TEMPSPACE2 is below.

```
% db2 list tablespaces show detail

Tablespace ID                         = 5
Name                                  = TEMPSPACE2
Type                                  = System managed space
Contents                              = System Temporary data
State                                 = 0x0000
  Detailed explanation:
    Normal
Total pages                           = 1
Useable pages                         = 1
Used pages                            = 1
Free pages                            = Not applicable
High water mark (pages)               = Not applicable
Page size (bytes)                     = 8192
Extent size (pages)                   = 32
Prefetch size (pages)                 = 32
Number of containers                  = 1
```

5. Disconnect from the database

```
% db2 connect reset
```

6.5. Update StorageSegDiskExtents Index

6.5.1. Introduction

Adding information into the SSDE index improves the performance of building the Core Server free space map, which in turn allows the Core Server to initialize more quickly. To complete the following, HPSS should be down and DB2 should be running.

6.5.2. Conversion Procedure

To add the index, the following steps (1 - 4) need to be repeated for each subsystem:

1. Become the HPSS database instance owner, and connect to the subsystem database

```
% su - hpssdb
% db2 connect to hsubsys1
```

2. The commands to run to modify the index will depend on your current metadata configuration. Executing the following will generate the correct commands (note the command line has wrapped in this document):

```
% db2 "select substr('grant control on INDEX ' || rtrim(indschema) ||
'.' || rtrim(indname) || ' to USER hpss ',1,100) as grant_command from
syscat.indexes where tabname='STORAGESEGDISKEXTENTS'"
```

   This will result in one of the two sets of commands below:

```
GRANT_COMMAND
---------------------------------------------------------------------------
grant control on INDEX HPSS.SSDE_PKEY to USER hpss
grant control on INDEX HPSS.SSDE_VVID_SO to USER hpss
grant control on INDEX HPSS.SSDE_VVID_EO to USER hpss
3 record(s) selected.
```

   Or

```
GRANT_COMMAND
----------------------------------------------------------------------------
grant control on INDEX HPSS.SSID_EXTENTS to USER hpss
grant control on INDEX HPSS.SSDE_VVID_EXTENTS_OFFSETS to USER hpss
2 record(s) selected.
```

3. Use the result as input to the DB2 Command Line Processor. For example, if the output matches the first set of grant commands, then execute the following:

```
% db2 "grant control on INDEX HPSS.SSDE_PKEY to USER hpss"
DB20000I The SQL command completed successfully.
% db2 "grant control on INDEX HPSS.SSDE_VVID_SO to USER hpss"
DB20000I The SQL command completed successfully.
% db2 "grant control on INDEX HPSS.SSDE_VVID_EO to USER hpss"
DB20000I The SQL command completed successfully.
% db2 connect reset
DB20000I The SQL command completed successfully.
```

4. As root, complete the index setup using **hpss_managetables**:

```
#/opt/hpss/bin/hpss_managetables
hmt> db subsys1
hmt> convert storagesegdiskextents 717
hmt> quit
```

   Sample output from a run is below.

```
# /opt/hpss/bin/hpss_managetables
hmt> db subsys1

Available tablespaces:
                                          Owner Data
ID  Tablespace Name           Owner       Type  Type

2   USERSPACE1                HPSSDB      U     L
```

```
3   TABLES                    HPSSDB      U     L
4   BFDALLOC                  HPSSDB      U     L
5   NSOBJECT                  HPSSDB      U     L
6   STSGTAPE                  HPSSDB      U     L
7   BFTAPESEG                 HPSSDB      U     L
8   BITFILE                   HPSSDB      U     L
9   STSGDISK                  HPSSDB      U     L
10  USERATTRS                 HPSSDB      U     L
11  INDEXES                   HPSSDB      U     L
12  NSOBJIDX                  HPSSDB      U     L
13  USERATTRSIDX              HPSSDB      U     L

Database: subsys1
Schema: HPSS
Default tablespace: NOT SELECTED
Default index tablespace: NOT SELECTED
hmt> convert storagesegdiskextents 717
Removed the primary key on table STORAGESEGDISKEXTENTS
Removed index SSDE_VVID_SO on table STORAGESEGDISKEXTENTS
Removed index SSDE_VVID_EO on table STORAGESEGDISKEXTENTS
Removed trigger CHECK_DISK_EXTENTS_OVERLAP_I
Removed trigger CHECK_DISK_EXTENTS_OVERLAP_U
Removed procedure CHECK_DISK_EXTENTS_OVERLAP_P
Transaction committed

ALTER TABLE STORAGESEGDISKEXTENTS ADD CONSTRAINT SSDE_PKEY PRIMARY KEY (
SSID, ORDINAL)

CREATE UNIQUE INDEX SSDE_VVID_SO ON STORAGESEGDISKEXTENTS ( VVID,
STARTING_OFFSET) INCLUDE ( ENDING_OFFSET) ALLOW REVERSE SCANS
```

6.6. Unique RPC Program Numbers

6.6.1. Introduction

The following steps are necessary to guarantee the system is using unique RPC Program Numbers for each server interface. Execute the procedure with HPSS down and DB2 running.

6.6.2. Conversion Procedure

1. Become root

```
% su -
```

2. Use **hpss_managetables** to modify the SERVER table index

```
# /opt/hpss/bin/hpss_managetables
hmt> db cfg
hmt> convert server 183
hmt> quit
```

Note: There is only one config database, so this need only be run once.

6.7. VFS UDA Index

6.7.1. Introduction

Prior to enabling VFS Checksum for the first time, the administrator should install an XML Index on the UDA table. This will allow VFS to efficiently query the namespace for locked files if lock cleanup is required. Execute the procedure with HPSS down and DB2 running.

6.7.2. Conversion Procedure

1. Become the HPSS instance owner:

   ```
   % su - hpssdb
   ```

2. Use the **xmladdindex** script to add the new XML index:

   ```
   /opt/hpss/config/xmladdindex hpss_vfs_user '/hpss/fs/user/*' 'varchar
   hashed'
   ```

   The command should report the following output:

   ```
   DB20000I The SQL command completed successfully.
   ```

6.8. VFS User ACL

6.8.1. Introduction

The VFS Checksum feature previously required that the VFS User (default hpssfs) be given additional permission when connecting to the HPSS Core Server. This was accomplished by adding the VFS user to the ACL for the core server’s Client Interface. This permission must now be removed. The **hpss_server_acl** command is used to modify the ACL. The procedure below will remove "write" permission for VFS. Complete the procedure with HPSS down and DB2 running.

6.8.2. Conversion Procedure

1. Become root

   ```
   % su -
   ```

2. **hpss_server_acl** is an interactive tool. At the prompt, request a menu of servers to choose from:

   ```
   # /opt/hpss/bin/hpss_server_acl

   hsa> acl -m
   ```

   A list similar to the following will be displayed:

   ```
   1) PVL
   2) STK PVR
   3) Mover tcp
   4) Gatekeeper
   ```

```
5) Core Server
6) SSM System Manager
Select a server
Choose an item by number (RET to cancel):
>
```

Select the *Core Server* from the list (*5* in this case), and the list of RPC interfaces the Core Server provides will be displayed. Enter the number for the *Client Interface*. For example, you would enter (2) if presented with the list of interfaces below:

```
1) PVL Mount Notification Interface (v1) 007ff347-e533-1cc6-b22d-02608c2cedf4
2) Client Interface (v1) 32ba9692-4667-11d6-aa3a-0004ac49692b
3) Account Validation Interface (v1) 647f22a8-a1e9-11d3-a739-000001341966
4) Realtime Monitor Interface (v1) 80c9a256-2f13-11d3-a0c8-000001341966
Select an interface
Choose an item by number (RET to cancel):
> 2
```

3. Enter the following at the prompt to grant write permission to the hpssfs user

```
hsa> del user hpssfs w
```

4. Display the ACL

```
hsa> show
```

Output will be similar to the following:

```
perms - type - ID (name) - realm ID (realm)
==========================================
rw-c-d- - user - 304 (hpssssm) - 10000 (SKAGWAY.CLEARLAKE.IBM.COM)
rw-c-dt - user - 308 (hpssmps) - 10000 (SKAGWAY.CLEARLAKE.IBM.COM)
r--c--- - user - 309 (hpssfs) - 10000 (SKAGWAY.CLEARLAKE.IBM.COM)
r--c--- - user - 315 (hpssftp) - 10000 (SKAGWAY.CLEARLAKE.IBM.COM)
------t - any_other

hsa>
```

5. Exit the utility

```
hsa> quit
```

6.9. Convert PVR Devices

6.9.1. Introduction

New PVR devices are required to support the SCSI passthrough feature. Conversion of the devices for HPSS 7.3.3 is completed by running the **convert_pvr_devices.sh** script (located in `tools/metadata/db2`). This script will create the new PVR table and convert the existing metadata into the new format. This information should then be verified using either **lshpss** or the SSM windows.

To revert the conversion, drop the new PVR table, rename the original table (which will be either PVR_ORIG or PVR_COMPLETE, depending upon how far along the conversion was) to PVR, and then add the primary key back on to the table PVR table (if necessary). If reverting the conversion is necessary, please contact HPSS support for assistance.

6.9.2. Conversion Procedure

This conversion is performed on the global config (**cfg**) database. The subsystem databases are not changed in this conversion. The conversion should be completed with HPSS down and DB2 running. Final configuration is performed with SSM. To perform the conversion, execute the following steps:

1. Become the database instance owner

   ```
   % su - hpssdb
   ```

2. Run the **convert_pvr_devices.sh** script

   ```
   % /opt/hpss/tools/metadata/db2/convert_pvr_devices.sh 2>&1| tee <output_file>
   ```

3. **Carefully examine the output in <output_file> for any errors. Sample output from a successful run is at the end of this section. If the script did not complete successfully, contact your HPSS support representative before continuing.**

4. If the script finishes without error, then verify PVR metadata. This can be completed using **lshpss** (as root)

   ```
   # lshpss -pvr
   ```

   and by using the PVR configuration windows in SSM. The device field values should be unchanged, however for the SCSI PVR you may see a **/dev/smc** device in the Device 1 field. This value was previously hidden and unused, and should be removed.

5. For a SCSI PVR, the "Drive Address" field of the Devices and Drives window must be changed from a "Drive Element Address" (e.g. 257) to a "Drive Serial Number". This change is required in order to guarantee a unique Drive Address across multiple libraries. These are similar in format to the Library Serial number, and can be obtained by running

   ```
   # device_scan -i
   ```

6.9.3. Sample Output

Sample output from running the **convert_pvr_devices.sh** script is below.

```
04/11/2011 16:05:13 Step --> connect

04/11/2011 16:05:12 connect to HCFG in exclusive mode

04/11/2011 16:05:13 Step --> pre_conversion_check

04/11/2011 16:05:13 select
                                count(*)
                        from
                                syscat.columns
                        where
                                colname='DEVICES31' and
                                tabschema='HPSS' and
                                tabname='PVR'
04/11/2011 16:05:13 DB20000I  The SQL command completed successfully.
04/11/2011 16:05:13 Column DEVICES31 does not exist in table HPSS.PVR,
conversion may proceed.
```

```
04/11/2011 16:05:13 Step --> before_conversion

04/11/2011 16:05:13 Step --> describe_table
04/11/2011 16:05:13 db2 describe table HPSS.PVR

                             Data type                     Column
Column name                  schema    Data type name      Length    Scale Nulls
---------------------------- --------- ------------------- --------- ----- -----
PVR_ID                       SYSIBM    CHARACTER                  16     0 No
VERSION                      SYSIBM    INTEGER                     4     0 Yes
DRIVE_ERR_LIMIT              SYSIBM    INTEGER                     4     0 Yes
TOTAL_CARTRIDGES             SYSIBM    INTEGER                     4     0 Yes
CARTS_ALARM_THRESHOLD        SYSIBM    INTEGER                     4     0 Yes
CARTS_CAPACITY               SYSIBM    INTEGER                     4     0 Yes
SAME_JOB_ON_CONTROLLER       SYSIBM    INTEGER                     4     0 Yes
OTHER_JOB_ON_CONTROLLER      SYSIBM    INTEGER                     4     0 Yes
DISTANCE_TO_DRIVE            SYSIBM    INTEGER                     4     0 Yes
CHAR_BITMAP                  SYSIBM    BIGINT                      8     0 Yes
TAPE_CHECKIN_ALARM           SYSIBM    INTEGER                     4     0 Yes
TAPE_CHECKIN_RETRY           SYSIBM    INTEGER                     4     0 Yes
DELAY_DISMOUNT_TIME          SYSIBM    INTEGER                     4     0 Yes
RETRY_MOUNT_TIME_LIMIT       SYSIBM    INTEGER                     4     0 Yes
RAIT_PHYS_COUNT              SYSIBM    INTEGER                     4     0 Yes
DEVICE_SPEC_A                SYSIBM    VARCHAR                   127     0 Yes
DEVICE_SPEC_B                SYSIBM    VARCHAR                   127     0 Yes
  17 record(s) selected.


04/11/2011 16:05:13 Step --> list_indexes
04/11/2011 16:05:13 db2 select substr(rtrim(indschema) || '.' || rtrim(indname),
1,50) as index_name, substr(colnames,1,60) as columns,uniquerule from syscat.ind
exes where tabschema='HPSS' and tabname='PVR'

INDEX_NAME                                     COLUMNS                 UNIQUERULE
---------------------------------------------- ----------------------- ----------
HPSS.PVR_PKEY                                  +PVR_ID                 P

  1 record(s) selected.

04/11/2011 16:05:13 Step --> list_constraints
04/11/2011 16:05:13 db2 select substr(a.constname,1,30) as constname,a.type,subs
tr(rtrim(b.tabschema) || '.' || rtrim(b.tabname),1,35) as child_table_name,subst
r(rtrim(b.reftabschema) || '.' || rtrim(b.reftabname),1,30) as prnt_table_name
from syscat.tabconst a left outer join  syscat.references b on a.constname=b.con
stname and a.tabname=b.tabname where a.tabschema='HPSS' and a.tabname='PVR'

CONSTNAME                  TYPE CHILD_TABLE_NAME                PRNT_TABLE_NAME
-------------------------- ---- ------------------------------- ---------------
PVR_PKEY                   P    -                               -

  1 record(s) selected.

04/11/2011 16:05:14 Step --> tblspace_data

04/11/2011 16:05:14 select
                                tbspace
                        from
                                syscat.tables
                        where
```

```
                                tabschema='HPSS' and
                                tabname='PVR'
04/11/2011 16:05:14 DB20000I  The SQL command completed successfully.
04/11/2011 16:05:14 Data tablespace --> USERSPACE1

04/11/2011 16:05:14 Step --> tblspace_index

04/11/2011 16:05:14 select
                                distinct substr(b.tbspace,1,20)
                        from
                                syscat.indexes a,
                                syscat.tablespaces b
                        where
                                a.tabschema='HPSS' and
                                a.tabname='PVR' and
                                a.tbspaceid=b.tbspaceid
04/11/2011 16:05:14 DB20000I  The SQL command completed successfully.
04/11/2011 16:05:14 Index tablespace --> USERSPACE1

04/11/2011 16:05:14 Step --> rename_orig_table

04/11/2011 16:05:14 rename table HPSS.PVR to PVR_orig
04/11/2011 16:05:14 DB20000I  The SQL command completed successfully.

04/11/2011 16:05:14 Step --> manage_orig_indexes

04/11/2011 16:05:14 alter table HPSS.PVR_orig drop primary key
04/11/2011 16:05:16 DB20000I  The SQL command completed successfully.

04/11/2011 16:05:16 Step --> create_new_table

04/11/2011 16:05:16 CREATE TABLE HPSS.PVR_new (
  PVR_ID CHAR (16) FOR BIT DATA NOT NULL,
  VERSION INTEGER,
  DRIVE_ERR_LIMIT INTEGER,
  TOTAL_CARTRIDGES INTEGER,
  CARTS_ALARM_THRESHOLD INTEGER,
  CARTS_CAPACITY INTEGER,
  SAME_JOB_ON_CONTROLLER INTEGER,
  OTHER_JOB_ON_CONTROLLER INTEGER,
  DISTANCE_TO_DRIVE INTEGER,
  CHAR_BITMAP BIGINT,
  TAPE_CHECKIN_ALARM INTEGER,
  TAPE_CHECKIN_RETRY INTEGER,
  DELAY_DISMOUNT_TIME INTEGER,
  RETRY_MOUNT_TIME_LIMIT INTEGER,
  RAIT_PHYS_COUNT INTEGER,
  DEVICES0 VARCHAR (127),
  DEVICES1 VARCHAR (127),
  DEVICES2 VARCHAR (127),
  DEVICES3 VARCHAR (127),
  DEVICES4 VARCHAR (127),
  DEVICES5 VARCHAR (127),
  DEVICES6 VARCHAR (127),
  DEVICES7 VARCHAR (127),
  DEVICES8 VARCHAR (127),
  DEVICES9 VARCHAR (127),
  DEVICES10 VARCHAR (127),
  DEVICES11 VARCHAR (127),
  DEVICES12 VARCHAR (127),
  DEVICES13 VARCHAR (127),
  DEVICES14 VARCHAR (127),
  DEVICES15 VARCHAR (127),
  DEVICES16 VARCHAR (127),
  DEVICES17 VARCHAR (127),
  DEVICES18 VARCHAR (127),
  DEVICES19 VARCHAR (127),
  DEVICES20 VARCHAR (127),
  DEVICES21 VARCHAR (127),
  DEVICES22 VARCHAR (127),
  DEVICES23 VARCHAR (127),
  DEVICES24 VARCHAR (127),
  DEVICES25 VARCHAR (127),
  DEVICES26 VARCHAR (127),
  DEVICES27 VARCHAR (127),
  DEVICES28 VARCHAR (127),
  DEVICES29 VARCHAR (127),
  DEVICES30 VARCHAR (127),
  DEVICES31 VARCHAR (127) ) IN USERSPACE1 INDEX IN USERSPACE1
04/11/2011 16:05:17 DB20000I  The SQL command completed successfully.

04/11/2011 16:05:17 Step --> create_new_indexes

04/11/2011 16:05:17 ALTER TABLE HPSS.PVR_new ADD CONSTRAINT PVR_PKEY
PRIMARY KEY (PVR_ID)
04/11/2011 16:05:18 DB20000I  The SQL command completed successfully.

04/11/2011 16:05:19 Step --> declare_cursor

04/11/2011 16:05:19 declare segcurs cursor for select pvr_id,version, drive_err_
limit, total_cartridges, carts_alarm_threshold, carts_capacity, same_job_on_cont
roller, other_job_on_controller, distance_to_drive, char_bitmap, tape_checkin_al
arm, tape_checkin_retry, delay_dismount_time, retry_mount_time_limit, rait_phys_
count, device_spec_a, device_spec_b, '', '', '', '', '', '', '', '', '', '', '',
 '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '' from
 HPSS.PVR_orig
04/11/2011 16:05:19 DB20000I  The SQL command completed successfully.

04/11/2011 16:05:19 Step --> load_new_table

04/11/2011 16:05:19 load from segcurs of cursor insert into HPSS.PVR_new
nonrecoverable
04/11/2011 16:05:19 DB20000I  The LOAD command completed successfully.

04/11/2011 16:05:19 Step --> rename_new_table

04/11/2011 16:05:19 rename table HPSS.PVR_new to PVR
04/11/2011 16:05:19 DB20000I  The SQL command completed successfully.

04/11/2011 16:05:19 Step --> rename_orig_table_complete

04/11/2011 16:05:19 rename table HPSS.PVR_orig to PVR_complete
04/11/2011 16:05:19 DB20000I  The SQL command completed successfully.

04/11/2011 16:05:19 Step --> runstats

04/11/2011 16:05:19 runstats on table HPSS.PVR with distribution and
detailed indexes all
```

```
04/11/2011 16:05:20 DB20000I  The RUNSTATS command completed
successfully.

04/11/2011 16:05:20 Step --> after_conversion
04/11/2011 16:05:20 Step --> describe_table
04/11/2011 16:05:20 db2 describe table HPSS.PVR

                             Data type                     Column
Column name                  schema    Data type name      Length    Scale Nulls
---------------------------- --------- ------------------- --------- ----- -----
PVR_ID                          SYSIBM    CHARACTER                   16     0 No
VERSION                         SYSIBM    INTEGER                      4     0 Yes
DRIVE_ERR_LIMIT                 SYSIBM    INTEGER                      4     0 Yes
TOTAL_CARTRIDGES                SYSIBM    INTEGER                      4     0 Yes
CARTS_ALARM_THRESHOLD           SYSIBM    INTEGER                      4     0 Yes
CARTS_CAPACITY                  SYSIBM    INTEGER                      4     0 Yes
SAME_JOB_ON_CONTROLLER          SYSIBM    INTEGER                      4     0 Yes
OTHER_JOB_ON_CONTROLLER         SYSIBM    INTEGER                      4     0 Yes
DISTANCE_TO_DRIVE               SYSIBM    INTEGER                      4     0 Yes
CHAR_BITMAP                     SYSIBM    BIGINT                       8     0 Yes
TAPE_CHECKIN_ALARM              SYSIBM    INTEGER                      4     0 Yes
TAPE_CHECKIN_RETRY              SYSIBM    INTEGER                      4     0 Yes
DELAY_DISMOUNT_TIME             SYSIBM    INTEGER                      4     0 Yes
RETRY_MOUNT_TIME_LIMIT          SYSIBM    INTEGER                      4     0 Yes
RAIT_PHYS_COUNT                 SYSIBM    INTEGER                      4     0 Yes
DEVICES0                        SYSIBM    VARCHAR                    127     0 Yes
DEVICES1                        SYSIBM    VARCHAR                    127     0 Yes
DEVICES2                        SYSIBM    VARCHAR                    127     0 Yes
DEVICES3                        SYSIBM    VARCHAR                    127     0 Yes
DEVICES4                        SYSIBM    VARCHAR                    127     0 Yes
DEVICES5                        SYSIBM    VARCHAR                    127     0 Yes
DEVICES6                        SYSIBM    VARCHAR                    127     0 Yes
DEVICES7                        SYSIBM    VARCHAR                    127     0 Yes
DEVICES8                        SYSIBM    VARCHAR                    127     0 Yes
DEVICES9                        SYSIBM    VARCHAR                    127     0 Yes
DEVICES10                       SYSIBM    VARCHAR                    127     0 Yes
DEVICES11                       SYSIBM    VARCHAR                    127     0 Yes
DEVICES12                       SYSIBM    VARCHAR                    127     0 Yes
DEVICES13                       SYSIBM    VARCHAR                    127     0 Yes
DEVICES14                       SYSIBM    VARCHAR                    127     0 Yes
DEVICES15                       SYSIBM    VARCHAR                    127     0 Yes
DEVICES16                       SYSIBM    VARCHAR                    127     0 Yes
DEVICES17                       SYSIBM    VARCHAR                    127     0 Yes
DEVICES18                       SYSIBM    VARCHAR                    127     0 Yes
DEVICES19                       SYSIBM    VARCHAR                    127     0 Yes
DEVICES20                       SYSIBM    VARCHAR                    127     0 Yes
DEVICES21                       SYSIBM    VARCHAR                    127     0 Yes
DEVICES22                       SYSIBM    VARCHAR                    127     0 Yes
DEVICES23                       SYSIBM    VARCHAR                    127     0 Yes
DEVICES24                       SYSIBM    VARCHAR                    127     0 Yes
DEVICES25                       SYSIBM    VARCHAR                    127     0 Yes
DEVICES26                       SYSIBM    VARCHAR                    127     0 Yes
DEVICES27                       SYSIBM    VARCHAR                    127     0 Yes
DEVICES28                       SYSIBM    VARCHAR                    127     0 Yes
DEVICES29                       SYSIBM    VARCHAR                    127     0 Yes
DEVICES30                       SYSIBM    VARCHAR                    127     0 Yes
DEVICES31                       SYSIBM    VARCHAR                    127     0 Yes

  47 record(s) selected.
```

```

04/11/2011 16:05:20 Step --> list_indexes
04/11/2011 16:05:20 db2 select substr(rtrim(indschema) || '.' || rtrim(indname),
1,50) as index_name, substr(colnames,1,60) as columns,uniquerule from syscat.ind
exes where tabschema='HPSS' and tabname='PVR'

INDEX_NAME                     COLUMNS                                  UNIQUERULE
------------------------------ ---------------------------------------- ----------
HPSS.PVR_PKEY                  +PVR_ID                                  P

  1 record(s) selected.

04/11/2011 16:05:20 Step --> list_constraints
04/11/2011 16:05:20 db2 select substr(a.constname,1,30) as constname,a.type,subs
tr(rtrim(b.tabschema) || '.' || rtrim(b.tabname),1,35) as child_table_name, subs
tr(rtrim(b.reftabschema) || '.' || rtrim(b.reftabname),1,30) as prnt_table_name
from syscat.tabconst a left outer join  syscat.references b on a.constname=b.con
stname and a.tabname=b.tabname where a.tabschema='HPSS' and a.tabname='PVR'

CONSTNAME                   TYPE CHILD_TABLE_NAME                 PRNT_TABLE_NAME
--------------------------- ---- -------------------------------- ---------------
PVR_PKEY                    P    -                                -

  1 record(s) selected.


exiting
commit
DB20000I  The SQL command completed successfully.

connect reset
DB20000I  The SQL command completed successfully.

terminate
DB20000I  The TERMINATE command completed successfully.
```

Chapter 7. HPSS 7.3.3 to HPSS 7.4.1 Conversion

This section summarizes the features introduced in HPSS 7.4.1 which require a metadata conversion. This is not a complete list of all new features. Refer to the HPSS 7.4.1 *Differences* section in the *HPSS Installation Guide* for more information.

7.1. General Information

The conversion from 7.3.3 to 7.4.1 involves configuration and metadata changes to support the following changes in HPSS:

- Migration Purge Server
  - Faster generation of migration candidate list.
  - Additional MPS configuration option to allow an administrator to set an active tape volume timeout. This is used to ensure that for tape to tape migration, migration candidates on busy volumes are migrated in a timely manner.
- Physical Volume Repository
  - Remove support for 3494 PVR
- Logging
  - Additional Core Server configuration option to control logging of disk and tape I/O requests.
  - New features in the Log Client and Log Daemon to control logging behavior.
- Support for HPSS redundant array of independent tapes (RAIT). For release 7.4.1, RAIT is a limited offering. Contact your support representative for further information.

7.2. Linux Kernel Parameter Changes

ASLR or Address Space Layout Randomization is a feature that is activated by default on some of the newer linux distributions. It is designed to load shared memory objects in random addresses.

In DB2, multiple processes map a shared memory object at the same address across the processes. It was found that DB2 cannot guarantee the availability of address for the shared memory object when ASLR is turned on.

Turn off the randomization by setting the following kernel parameter in /etc/sysctl.conf:

```
kernel.randomize_va_space=0.
```

The system must be rebooted to recognize the change.

7.3. Metadata Changes

This section provides additional information about the HPSS metadata changes in Release 7.4.1. The first section relates the new features described in the previous section to their specific metadata changes in DB2. The second section enumerates all DB2 metadata changes made to the global and subsystem databases.

7.3.1. New Features Affecting Conversion

7.3.1.1. Improved Tape to Tape Migration for Busy Volumes

A MAX_ACTIVE_FILE_DELAY column has been added to the MIGPOL table. This allows sites to customize how long an active tape volume should delay migrating candidate files.

7.3.1.2. Improved Performance Building Migration Candidate List

To reduce the time needed to build a migration candidate list, a new view named BFMIGRRECBITFILEVIEW has been added over the BITFILE and BFMIGREC tables.

7.3.1.3. Better Logging of HPSS I/O Requests

A LOGGING_OPTIONS column has been added to the CORE table in the global database. This is used to control I/O request logging.

7.3.1.4. Better Control of Disk Resident Log Files

The LOGC_PRINT_FILENAME column in the LOGCLIENT table has been renamed to LOGC_DIRECTORY to better reflect its use in 7.4.1. A LOGC_RETENTION_COUNT column has been added to track the number of log files to retain on local disk. The LOGDAEMON table has similar changes. See the table in the following section for more detail.

7.3.1.5. RAIT

In the global database, the PVLACTIVITY, PVR and SCLASS tables have new fields for RAIT support. There is a new RAITENGINE table to hold the specific configuration for the RAIT engine server.

In the subsystem database, the VVTAPE table has been modified to store RAIT metadata, and the SSPVHISTORY, used to store events affecting physical volumes, has been added.

7.3.1.6. Additional Changes

HPSS 7.4.1 includes metadata changes to the global and subsystem databases which are not tied to a specific feature, but instead are part of various internal improvements to the software. Obsolete columns have been removed from the PVR, SSPVTAPE, STORAGEMAPDISK, STORAGESEGTAPE, VVDISK and VVTAPE tables. The VVTAPEABSADDR table has been removed. Views in the subsystem database have been updated to reflect the underlying table changes.

7.3.2. Metadata Change Detail

The following tables list all the DB2 metadata changes in version 7.4.1 for the global and subsystem databases.

7.3.2.1. HPSS Global (CFG) Database Changes

Table	Change
CORE	Add LOGGING_OPTIONS column to control logging of I/O requests.
LOGCLIENT	Rename the LOGC_PRINT_FILENAME column to LOGC_DIRECTORY. This column will contain the name of the directory that will hold the log files. Add LOGC_RETENTION_COUNT column to track the number of log files to retain on local disk.
LOGDAEMON	Rename the LOGD_CHECK_SWITCH to LOGD_RETENTION_COUNT. This column will hold the number of log files to retain on local disk.
MIGPOL	Add MAX_ACTIVE_FILE_DELAY column to improve tape to tape migration responsiveness.
PVLACTIVITY	Add TAG, TOLERATE and MOUNTMODE columns to support RAIT.
PVR	Remove the RAIT_PHYS_COUNT column. This existed to support an STK RAIT offering which is now obsolete.
RAITENGINE	Add table for RAIT Server configuration.
SCLASS	Rename STRIPE_WIDTH column to DATA_STRIPE_WIDTH. Add PARITY_STRIPE_WIDTH, MINIMUM_WRITE_PARITY and NUMBER_RAIT_ENGINES columns to support RAIT.

7.3.2.2. HPSS Subsystem Database Changes

Table	Change
SSPVHISTORY	Add SSPVHISTORY table for RAIT support.
SSPVTAPE	Remove NEXT_WRITE_ADDR_SECTION, NEXT_WRITE_ADDR_OFFSET,

Table	Change
	NEXT_ABS_WRITE_ADDR_TYPE and NEXT_ABS_WRITE_ADDR_ADDR columns.
STORAGEMAPDISK	Remove NUM_ACTIVE_SEGMENTS column.
STORAGESEGTAPE	Remove SS_STATE, REL_NEXT_BYTE_ADDR_SECTION and REL_NEXT_BYTE_ADDR_OFFSET columns.
VVDISK	Remove UPDATE column.
VVTAPE	Rename STRIPE_WIDTH column to DATA_STRIPE_WIDTH. Add PARITY_STRIPE_WIDTH, MIN_WRITE_PARITY and NUM_RAIT_ENGINES columns to support RAIT. Remove UPDATE column.
VVTAPEABSADDR	Remove the VVTAPEABSADDR table.

View	Purpose
BFMIGRRECBITFILEVIEW	Add view over the BITFILE and BFMIGREC tables in order to speed up generation of migration candidate list.
STORAGESEGTAPEVIEW	Remove SS_STATE, REL_NEXT_BYTE_ADDR_SECTION and REL_NEXT_BYTE_ADDR_OFFSET columns.
VVDISKVIEW	Remove UPDATE column.
VVTAPEVIEW	Rename STRIPE_WIDTH column to DATA_STRIPE_WIDTH. Add PARITY_STRIPE_WIDTH, MIN_WRITE_PARITY and NUM_RAIT_ENGINES columns. Remove UPDATE, ORDINAL, ADDR and TYPE columns.
VVTAPEMAPVIEW	From the VVTAPE table: Rename STRIPE_WIDTH to DATA_STRIPE_WIDTH. Rename ACTUAL_SIZE to TOTAL_WRITTEN. Add PARITY_STRIPE_WIDTH, MIN_WRITE_PARITY, NUM_RAIT_ENGINES, and VV_FLAGS columns.

View	Purpose
	From the STORAGEMAPETAPE table: Rename TAPE_SIZE to ACTIVE_WRITTEN. Rename NUM_ACTIVE_SEGMENTS to NUMBER_OF_SEGMENTS. Rename FLAGS to MAP_FLAGS.

Constraint	Purpose
TAPEVVAACON1	As part of removing the VVTAPEABSADDR table, remove related constraint.
SSPVHISTORYCON1	Add constraint to keep the SSPVHISTORY and SSPVTAPE tables consistent when deleting entries from SSPVTAPE.

7.4. Conversion Overview

This section provides instructions for upgrading HPSS 7.3.3 DB2 metadata to release 7.4.1. To ensure that the conversion is successful, all steps must be followed in the order specified in the documentation. Prior to starting the conversion process, consult the HPSS 7.4.1 Release Notes for any special instructions.

7.4.1. Preparing to Upgrade

Before attempting any HPSS upgrade, follow the preparation procedure in *Gather Data On Existing System Section 1.3.1, "Gather Data On Existing System"*, *Benchmark Existing System Section 1.3.2, "Benchmark Existing System"* and *Backup Existing System Section 1.3.3, "Backup Existing System"* in this guide.

7.4.2. Software Prerequisites

The HPSS 7.4.1 prerequisite software is documented in the *HPSS 7.4.1 Installation Guide*. Depending on your current HPSS configuration and prerequisite software levels, there may be software components which must be upgraded prior to running the 7.4.1 metadata conversion. Follow the upgrade instructions provided by the prerequisite software to bring any software up to the level required by HPSS 7.4.1. Consult with your support representative for additional guidance.

The conversion process itself uses the Perl Database Interface (DBI) to interact with DB2. DBI provides a uniform interface to a number of Database engines, including DB2. DBI is available as part of the operating system on all supported Linux distributions. For sites running DB2 on AIX, a precompiled DBI RPM is available through your support representative. Alternatively, DBI in source form is available for download from http://www.cpan.org.

Because of licensing restrictions, the Database Driver (DBD) needed to interact with DB2 is not included in standard Linux distributions. The DB2 DBD was developed by and is copyrighted to IBM. Precompiled RPMs for AIX and LINUX are available from your support representative. DB2 DBD sources are also available for download from http://www.cpan.org.

The 7.4.1 conversion process was developed and tested with DBD version 1.84 and DBI version 1.609.

7.4.3. Special Considerations for 3494 PVR

In HPSS release 7.3.0, support for the IBM 3494 Tape Library and the 3494 PVR was discontinued. However, customers running HPSS 7.3 with existing 3494 PVR configurations were allowed to continue to use the library with no customer support. As of 7.4.1, metadata for the 3494 PVR has been removed, and HPSS no longer allows any 3494 PVR configurations. Any existing 3494 PVR configurations need to be deleted prior to installing the HPSS 7.4.1 software.

7.4.4. Conversion Planning

Careful planning for conversion is important in order to ensure a successful conversion. This section describes the planning steps that need to be completed before running conversion.

7.4.4.1. Table Layout Considerations

The SSPVHISTORY table in the subsystem database(s) is new for 7.4.1. The number of rows in this table will be within an order of magnitude of the number of rows in the SSPVTAPE table. Sites using automatic storage for table spaces will want to consider placing SSPVHISTORY and its index in their own table space. A standard database layout created by **mkhpss** in version 7.4.1 configures the following table spaces for SSPVHISTORY:

```
CREATE LARGE TABLESPACE "SSPVHISTORY" IN DATABASE PARTITION GROUP
IBMDEFAULTGROUP PAGESIZE 4096 MANAGED BY AUTOMATIC STORAGE

         EXTENTSIZE 128
         PREFETCHSIZE AUTOMATIC
         BUFFERPOOL SMALLTABLES
         OVERHEAD 7.500000
         TRANSFERRATE 0.060000
         AUTORESIZE YES
         MAXSIZE NONE
         NO FILE SYSTEM CACHING
         DROPPED TABLE RECOVERY ON

CREATE LARGE TABLESPACE "SSPVHISTORYIX" IN DATABASE PARTITION GROUP
IBMDEFAULTGROUP PAGESIZE 4096 MANAGED BY AUTOMATIC STORAGE

         EXTENTSIZE 128
         PREFETCHSIZE AUTOMATIC
         BUFFERPOOL SMALLTABLESIX
         OVERHEAD 7.500000
         TRANSFERRATE 0.060000
         AUTORESIZE YES
         MAXSIZE NONE
         NO FILE SYSTEM CACHING
         DROPPED TABLE RECOVERY ON
```

While this layout will not be appropriate for all existing installations, all sites must identify the table space to use for the SSPVHISTORY table, and the table space must exist before beginning the conversion. Sites using DMS storage will want to consider adding the table and index to an existing table space. Contact your support representative if additional guidance is needed.

7.4.4.2. Free Space Considerations

The conversion process saves a backup copy of all tables that it changes. Prior to starting the conversion, ensure there is adequate free space for making a backup copy of the tables that will be converted. The *Metadata Change Detail Section 7.3.2, "Metadata Change Detail"* section in this chapter lists the tables that will be converted. Pay particular attention to the space requirements of the STORAGESEGTAPE table. This will typically be the largest of the tables to be converted.

7.4.4.3. The Conversion Process

After the prerequisite steps have been completed, and any 3494 PVR configurations have been removed, the conversion process itself consists of the following steps:

- Install HPSS 7.4.1
- For each subsystem database, use **hpss_managetables** to remove database views and constraints.
- Use **hpss_convertdb** to perform the metadata conversion on the global (CFG) database and each subsystem database.
- For each subsystem database, use **hpss_managetables** to add database views and constraints.

7.4.4.4. Tools Used by the Conversion Process

Conversion uses **hpss_managetables** and **hpss_convertdb** to convert the HPSS metadata. Please read the section *hpss_managetables Section 1.4, "hpss_managetables"* for a warning and guidance concerning the usage of the **hpss_managetables** tool before continuing.

hpss_convertdb is used to perform the actual metadata conversion. The tool is only used for the conversion process, and is not installed in the standard *bin* location. Instead, it is located in `tools/convert/hpss` beneath the root of the HPSS installation. **hpss_convertdb** must be run as the root user, and requires the user to enter the password for the HPSS instance owner (typically hpssdb) in order to execute. Database operations are executed as the instance owner. This is required in order to set the proper permissions on database tables and indexes created during the conversion process. The following example shows invocation of the tool:

```
# /opt/hpss/tools/convert/hpss/hpss_convertdb
HPSS_DB_INSTANCE_OWNER is set - Using 'hpssdb' for database connections.
Password for hpssdb:
cnvt>
```

hpss_convertdb recognizes the following commands:

```
cnvt> help
Available commands:
   cleanup Cleanup after conversion
   close Disconnect from a database
   convert Run the conversion
   count Count the number of rows in a table
   db Connect to database
   env Dump configuration parameters
   help Display available commands
   revert Revert the conversion
   rsql Show the SQL that will be run to revert a table
   showts Show tablespace info for tables that will be converted
   sql Show the SQL that will be run to convert a table
   state Show the current state of conversion
   ts Set the tablespace to use
cnvt>
```

Additional description of the **hpss_convertdb** commands are below.

cleanup <all | table_name> - After a successful conversion, delete the backup copy of **table_name**. The operation requires a database connection. Once run, the conversion cannot be reverted.

close - close a database connection.

count - <all | table_name> - Count the number of rows in **table_name**. This operation requires a database connection.

convert <all | table_name> - Convert the indicated table. The operation requires a database connection. If **table_name** isn’t specified, then convert all tables in the connected database.

db <db_name> - Build a database connection to **db_name**. Note that **hpss_convertdb** expects the database alias name (CFG, or SUBSYS1 for example) to be provided, rather than the database name. This is so it can verify the name of subsystem databases in the global metadata.

env - Display database information and configuration parameters.

revert <all | table_name> - Revert the specified table. The operation requires a database connection. This option is meant to be used only by HPSS support since there may be additional steps needed to completely revert to the previous HPSS level.

rsql <table_name> - Show the SQL that will be used to revert the specified table. This requires a database connection.

showts - Show the tablespaces that will be used for the conversion. The operation requires a database connection. Before a table can be converted, it must have a table space associated with it to hold the table and indexes.

sql <table_name> - Show the SQL that will be used to convert the specified table. This operation requires a database connection.

state - Show the conversion state for the table. This requires a database connection. Each table will be in one of the following states: START, CREATE, FILL, DEPLOY, CLEANUP or DONE.

ts <table_name> - Set the tablespaces that will be used for table **table_name**. This requires a database connection.

The first time **hpss_convertdb** connects to a database, it creates a table to track the conversion process. For the 7.4.1 conversion, the table is called CONVERT_741. The table contains a row for each of the HPSS tables that will be converted. As a table is being worked on, the state maintained in the row will change to reflect the conversion’s progress. It is important not to delete or modify this tracking table. Doing so will cause the conversion process to fail and may leave the metadata in an inconsistent state.

7.5. Conversion - Detailed Instructions

This section provides detailed instructions for executing the 7.4.1 conversion. Depending on your configuration, (number of subsystems, table space layout, subsystem database names, etc.), there may be some differences in the steps you need to run to complete the conversion at your site. If you have questions about the conversion process, contact your support representative before attempting the conversion. This section assumes that the prerequisite steps have been completed, the HPSS 7.4.1 software has been installed, and you are ready to use **hpss_managetables** and **hpss_convertdb** to run the conversion.

7.5.1. Delete Subsystem Views and Constraints

The first step in the conversion process is to remove views and constraints from the subsystem database(s). For this and the remaining conversion steps, HPSS must be down and DB2 must be running.

1. Become root

   ```
   % su -
   ```

2. Use **hpss_managetables** to delete views and constraints

   ```
   # /opt/hpss/bin/hpss_managetables
   hmt> db subsys1
   hmt> del subsys views
   hmt> del subsys constraints
   hmt> commit
   hmt> quit
   ```

Using the 7.4.1 **hpss_managetables** to delete HPSS 7.3.3 views and constraints will result in a couple of messages about missing views and constraints. These will not affect the 7.4.1 conversion process and can be ignored. Output from the command is below.

```
hmt> del subsys views
Deleting BFMIGRRECBITFILEVIEW
View BFMIGRRECBITFILEVIEW doesn't exist.
Deleting STORAGESEGDISKVIEW
Deleting STORAGESEGTAPEVIEW
Deleting VVDISKVIEW
Deleting VVTAPEVIEW
Deleting VVTAPEMAPVIEW
hmt> del subsys constraints
Deleting BFTAPESEGCON1
Deleting DISKSEGEXTENTSCON1
Deleting DISKVVMAPCON1
Deleting DISKVVPVCON1
Deleting DISKVVSEGSCON1
Deleting TAPESEGABSADDRCON1
Deleting TAPESEGAUXCON1
Deleting TAPEVVMAPCON1
Deleting SSPVHISTORYCON1
Constraint SSPVHISTORYCON1 doesn't exist.
Deleting TAPEVVPVCON1
Deleting TAPEVVSEGSCON1
```

```
Deleting NSOBJECTUSERATTRS1
hmt> commit
Transaction committed
hmt>
```

Repeat these steps for each subsystem database in your HPSS, altering the **db** command to use the appropriate subsystem database name.

7.5.2. Convert the Global Database

The next step in the conversion process is to use **hpss_convertdb** to convert the global (CFG) database.

1. Become root

   ```
   % su -
   ```

2. Use **hpss_convertdb** to convert the CFG database. The commands needed to convert the tables in the CFG database are shown next, followed by a transcript from a run, and discussion of the commands.

   ```
   # /opt/hpss/tools/convert/hpss/hpss_convertdb
   cnvt> db cfg
   CFG> ts raitengine
   CFG> state
   CFG> count all
   CFG> convert all
   CFG> state
   CFG> count all
   CFG> close
   cnvt>

   # /opt/hpss/tools/convert/hpss/hpss_convertdb
   HPSS_DB_INSTANCE_OWNER is set - Using 'hpssdb' for database connections.
   Password for hpssdb:
   cnvt> db cfg
   Creating table 'CONVERT_741' to track conversion progress...
   Making entry for table CORE
   Making entry for table PVLACTIVITY
   Making entry for table MIGPOL
   Making entry for table PVR
   Making entry for table RAITENGINE
   Making entry for table SCLASS
   Making entry for table LOGCLIENT
   Making entry for table LOGDAEMON

   Table Name      Conversion State Table/Index Space
   --------------- --------------- -------------------------
   CORE            START           USERSPACE1/USERSPACE1
   PVLACTIVITY     START           USERSPACE1/USERSPACE1
   MIGPOL          START           USERSPACE1/USERSPACE1
   PVR             START           USERSPACE1/USERSPACE1
   RAITENGINE      START           <UNDEF>/<UNDEF>
   SCLASS          START           USERSPACE1/USERSPACE1
   LOGCLIENT       START           USERSPACE1/USERSPACE1
   LOGDAEMON       START           USERSPACE1/USERSPACE1
   ```

The first time a connection is built to the database, the table used to track conversion is created, an entry is made for each table that needs to be converted, and information for each table is displayed. In the output above, Conversion State refers to the state the table is at in the conversion process. Table/Index Space displays the table space that the converted table will reside in. The RAITENGINE table is new for HPSS 7.4.1 and has no table spaces assigned yet. One will need to be chosen before running the conversion process.

```
CFG> ts raitengine
Choose a tablespace to use for table RAITENGINE
  0. USERSPACE1

Enter the number of the item to choose, or q[uit] > 0
Choose a tablespace to use for RAITENGINE indexes
  0. USERSPACE1

Enter the number of the item to choose, or q[uit] > 0
Setting tablespace for table RAITENGINE to USERSPACE1
Setting tablespace for table RAITENGINE indexes to USERSPACE1
```

The **ts** command is used to choose space for a table and its associated indexes. There is typically only one table space to choose from in the CFG database.

```
CFG> state

Table Name      Conversion State Table/Index Space
--------------- --------------- ------------------------
CORE            START           USERSPACE1/USERSPACE1
PVLACTIVITY     START           USERSPACE1/USERSPACE1
MIGPOL          START           USERSPACE1/USERSPACE1
PVR             START           USERSPACE1/USERSPACE1
RAITENGINE      START           USERSPACE1/USERSPACE1
SCLASS          START           USERSPACE1/USERSPACE1
LOGCLIENT       START           USERSPACE1/USERSPACE1
LOGDAEMON       START           USERSPACE1/USERSPACE1
```

Now **state** shows the selected tablespace for RAITENGINE.

```
CFG> count all

Table Name      Number of Rows
--------------- ---------------
CORE                          1
PVLACTIVITY                   1
MIGPOL                        1
PVR                           0
RAITENGINE              <undef>
SCLASS                        2
LOGCLIENT                     1
LOGDAEMON                     1
```

Make a note of the number of rows in each of the tables prior to running conversion. This should be compared against the number of rows in the converted tables to verify that conversion completed successfully.

```
CFG> convert all
Converting CORE...
```

```
Executing operations for state: START
Executing operations for state: CREATE
Executing operations for state: FILL
Executing operations for state: DEPLOY
Done
Converting PVLACTIVITY...
Executing operations for state: START
Executing operations for state: CREATE
Executing operations for state: FILL
Executing operations for state: DEPLOY
Done
Converting MIGPOL...
Executing operations for state: START
Executing operations for state: CREATE
Executing operations for state: FILL
Executing operations for state: DEPLOY
Done
Converting PVR...
Executing operations for state: START
Executing operations for state: CREATE
Executing operations for state: FILL
Executing operations for state: DEPLOY
Done
Converting RAITENGINE...
Executing operations for state: START
Executing operations for state: CREATE
Executing operations for state: FILL
Executing operations for state: DEPLOY
Done
Converting SCLASS...
Executing operations for state: START
Executing operations for state: CREATE
Executing operations for state: FILL
Executing operations for state: DEPLOY
Done
Converting LOGCLIENT...
Executing operations for state: START
Executing operations for state: CREATE
Executing operations for state: FILL
Executing operations for state: DEPLOY
Done
Executing operations for state: START
Executing operations for state: CREATE
Executing operations for state: FILL
Executing operations for state: DEPLOY
Done
```

Run conversion for all the tables in this database. A message is displayed as the process transitions through each state. Once the operations in the DEPLOY state complete, the original table will have been renamed, and the converted table will have been swapped in place of the original.

```
CFG> state

Table Name      Conversion State Table/Index Space
--------------- ---------------- ------------------------
CORE            CLEANUP          USERSPACE1/USERSPACE1
PVLACTIVITY     CLEANUP          USERSPACE1/USERSPACE1
MIGPOL          CLEANUP          USERSPACE1/USERSPACE1
PVR             CLEANUP          USERSPACE1/USERSPACE1
```

```
RAITENGINE      CLEANUP         USERSPACE1/USERSPACE1
SCLASS          CLEANUP         USERSPACE1/USERSPACE1
LOGCLIENT       CLEANUP         USERSPACE1/USERSPACE1
LOGDAEMON       CLEANUP         USERSPACE1/USERSPACE1
```

Running **state** again shows that all tables are now awaiting cleanup. This means that the converted tables have been deployed, but a backup copy of the originals still exists.

```
CFG> count all

Table Name      Number of Rows
--------------- ---------------
CORE                          1
PVLACTIVITY                   1
MIGPOL                        1
PVR                           0
RAITENGINE                    0
SCLASS                        2
LOGCLIENT                     1
LOGDAEMON                     1
```

Compare the number of rows in the converted table to the number of rows in the original. If the counts don't match, do not continue the conversion.

```
CFG> close
```

Close the database connection.

7.5.3. Convert the Subsystem Databases

For each subsystem database, use **hpss_convertdb** to run the conversion process.

1. Become root

```
% su -
```

2. Use **hpss_convertdb** to convert the subsystem databases. The commands needed to convert the tables in the subsystem database are shown next, followed by a transcript and discussion of the commands. The steps are nearly identical to the steps used to convert the CFG database. Differences are pointed out in the discussion below.

```
cnvt> db subsys1
SUBSYS1> ts sspvhistory
SUBSYS1> state
SUBSYS1> count all
SUBSYS1> convert all
SUBSYS1> state
SUBSYS1> count all
SUBSYS1> close
cnvt>

cnvt> db subsys1
Creating table 'CONVERT_741' to track conversion progress...
Making entry for table SSPVTAPE
Making entry for table STORAGEMAPDISK
Making entry for table STORAGESEGTAPE
Making entry for table VVDISK
Making entry for table SSPVHISTORY
Making entry for table VVTAPE
Making entry for table VVTAPEABSADDR

Table Name      Conversion State Table/Index Space
--------------- --------------- -------------------------
SSPVTAPE        START SSPVTAPE/SSPVTAPEIX
STORAGEMAPDISK  START STORAGEMAPDISK/STORAGEMAPDISKIX
STORAGESEGTAPE  START STORAGESEGTAPE/STORAGESEGTAPEIX
VVDISK          START VVDISK/VVDISKIX
SSPVHISTORY     START <UNDEF>/<UNDEF>
VVTAPE          START VVTAPE/VVTAPEIX
VVTAPEABSADDR   START VVTAPEABSADDR/VVTAPEABSADDRIX
SUBSYS1> ts sspvhistory
Choose a tablespace to use for table SSPVHISTORY
  0. ACCOUNTING
  1. ACCOUNTINGIX
  2. BFCOSCHANGE
  3. BFCOSCHANGEIX
  4. BFDISKALLOCREC
  5. BFDISKALLOCRECIX
  6. BFDISKSEG
...
 34. SSPVHISTORY
 35. SSPVHISTORYIX
 36. SSPVTAPE
 37. SSPVTAPEIX
...
 60. VVTAPEABSADDR
 61. VVTAPEABSADDRIX
 62. VVTAPEIX

Enter the number of the item to choose, or q[uit] > 34
Choose a tablespace to use for SSPVHISTORY indexes
  0. ACCOUNTING
  1. ACCOUNTINGIX
  2. BFCOSCHANGE
  3. BFCOSCHANGEIX
  4. BFDISKALLOCREC
  5. BFDISKALLOCRECIX
  6. BFDISKSEG
...
 34. SSPVHISTORY
 35. SSPVHISTORYIX
 36. SSPVTAPE
 37. SSPVTAPEIX
...
 60. VVTAPEABSADDR
 61. VVTAPEABSADDRIX
 62. VVTAPEIX

Enter the number of the item to choose, or q[uit] > 35
Setting tablespace for table SSPVHISTORY to SSPVHISTORY
Setting tablespace for table SSPVHISTORY indexes to SSPVHISTORYIX
```

The pick list has been edited to save space. The list includes SSPVHISTORY and SSPVHISTORYIX table spaces, which were created prior to running the conversion. Select the table spaces that are appropriate for your site.

```
SUBSYS1> state

Table Name      Conversion State Table/Index Space
--------------- --------------- --------------------------
SSPVTAPE        START           SSPVTAPE/SSPVTAPEIX
STORAGEMAPDISK  START           STORAGEMAPDISK/STORAGEMAPDISKIX
STORAGESEGTAPE  START           STORAGESEGTAPE/STORAGESEGTAPEIX
VVDISK          START           VVDISK/VVDISKIX
SSPVHISTORY     START           SSPVHISTORY/SSPVHISTORYIX
VVTAPE          START           VVTAPE/VVTAPEIX
VVTAPEABSADDR   START           VVTAPEABSADDR/VVTAPEABSADDRIX
SUBSYS1> count all

Table Name      Number of Rows
--------------- ---------------
SSPVTAPE                  10000
STORAGEMAPDISK               52
STORAGESEGTAPE         10000001
VVDISK                       52
SSPVHISTORY             <undef>
VVTAPE                    10000
VVTAPEABSADDR             10000
SUBSYS1> convert all
Converting SSPVTAPE...
Executing operations for state: START
Executing operations for state: CREATE
Executing operations for state: FILL
Executing operations for state: DEPLOY
Done
Converting STORAGEMAPDISK...
Executing operations for state: START
Executing operations for state: CREATE
Executing operations for state: FILL
Executing operations for state: DEPLOY
Done
Converting STORAGESEGTAPE...
Executing operations for state: START
Executing operations for state: CREATE
Executing operations for state: FILL
Executing operations for state: DEPLOY
Done
Converting VVDISK...
Executing operations for state: START
Executing operations for state: CREATE
Executing operations for state: FILL
Executing operations for state: DEPLOY
Done
Converting SSPVHISTORY...
Executing operations for state: START
Executing operations for state: CREATE
Executing operations for state: FILL
Executing operations for state: DEPLOY
Done
Converting VVTAPE...
Executing operations for state: START
Executing operations for state: CREATE
Executing operations for state: FILL
Executing operations for state: DEPLOY
```

```
Done
Converting VVTAPEABSADDR...
Executing operations for state: START
Executing operations for state: CREATE
Executing operations for state: FILL
Executing operations for state: DEPLOY
Done
SUBSYS1> state

Table Name      Conversion State Table/Index Space
--------------- --------------- ------------------------
SSPVTAPE        CLEANUP         SSPVTAPE/SSPVTAPEIX
STORAGEMAPDISK  CLEANUP         STORAGEMAPDISK/STORAGEMAPDISKIX
STORAGESEGTAPE  CLEANUP         STORAGESEGTAPE/STORAGESEGTAPEIX
VVDISK          CLEANUP         VVDISK/VVDISKIX
SSPVHISTORY     CLEANUP         SSPVHISTORY/SSPVHISTORYIX
VVTAPE          CLEANUP         VVTAPE/VVTAPEIX
VVTAPEABSADDR   CLEANUP         VVTAPEABSADDR/VVTAPEABSADDRIX
SUBSYS1> count all

Table Name      Number of Rows
--------------- ---------------
SSPVTAPE                  10000
STORAGEMAPDISK               52
STORAGESEGTAPE         10000001
VVDISK                       52
SSPVHISTORY                   0
VVTAPE                    10000
VVTAPEABSADDR           <undef>
```

Compare the number of rows in the converted table to the number of rows in the original. If the counts don't match, do not continue the conversion. The `VVTAPEABSADDR` now shows `<undef>` for row count because the table was removed as part of the conversion process.

```
SUBSYS1> close
```

7.5.4. Create Views and Constraints

The 7.4.1 views and constraints need to be created for each subsystem database. Use **hpss_managetables** to do this.

1. Become root

```
% su -
```

2. Use **hpss_managetables** to add the database views and constraints

```
# /opt/hpss/bin/hpss_managetables
hmt> db subsys1
hmt> add subsys views
hmt> add subsys constraints
hmt> commit
hmt> quit
```

After completion of this step, the metadata conversion is complete. A transcript of using **hpss_managetables** to add 7.4.1 views and constraints follows.

```
hmt> add subsys views
```

```
Adding BFMIGRRECBITFILEVIEW
Adding STORAGESEGDISKVIEW
Adding STORAGESEGTAPEVIEW
Adding VVDISKVIEW
Adding VVTAPEVIEW
Adding VVTAPEMAPVIEW
hmt> add subsys constraints
Adding BFTAPESEGCON1
Adding DISKSEGEXTENTSCON1
Adding DISKVVMAPCON1
Adding DISKVVPVCON1
Adding DISKVVSEGSCON1
Adding TAPESEGABSADDRCON1
Adding TAPESEGAUXCON1
Adding TAPEVVMAPCON1
Adding SSPVHISTORYCON1
Adding TAPEVVPVCON1
Adding TAPEVVSEGSCON1
Adding NSOBJECTUSERATTRS1
hmt> commit
Transaction committed
hmt> quit
```

7.6. Post Conversion

This section describes that steps that need to be taken after the metadata conversion has completed.

7.6.1. Run Full Database Backups

A full backup of all databases should be run after conversion has completed. It is especially important to complete a full backup of the table space containing the STORAGESEGTAPE table. Conversion of the STORAGESEGTAPE table uses the db2 load facility. Load bypasses the normal transaction logging process, and the table will not be roll-forward recoverable until the full backup completes. Failing to complete the backup will result in data loss if a roll-forward recovery is needed.

7.6.2. Complete HPSS Configuration

After completing all 7.3.3 to 7.4.1 conversion steps and restarting HPSS, there are new logging, Core Server and Migration Purge Server options that can be customized for your site. Refer to the *New Features* section in the *HPSS Installation Guide*, and the relevant sections in the *HPSS Management Guide* for further information.

7.6.3. Cleanup

Once satisfied with conversion to the 7.4.1 system, the backup copies of the 7.3.3 tables and the table used to track the conversion process should be removed to free up space. After this step is completed, the conversion can not be reverted.

7.6.3.1. Cleanup the Global Database

1. Become root

```
% su -
```

2. Use **hpss_convertdb** to cleanup

```
# /opt/hpss/tools/convert/hpss/hpss_convertdb
HPSS_DB_INSTANCE_OWNER is set - Using 'hpssdb' for database connections.
Password for hpssdb:
cnvt> db cfg
Using existing table 'CONVERT_741' to track conversion progress.

Table Name      Conversion State Table/Index Space
--------------- --------------- -------------------------
CORE            CLEANUP         USERSPACE1/USERSPACE1
PVLACTIVITY     CLEANUP         USERSPACE1/USERSPACE1
MIGPOL          CLEANUP         USERSPACE1/USERSPACE1
PVR             CLEANUP         USERSPACE1/USERSPACE1
RAITENGINE      CLEANUP         USERSPACE1/USERSPACE1
SCLASS          CLEANUP         USERSPACE1/USERSPACE1
LOGCLIENT       CLEANUP         USERSPACE1/USERSPACE1
LOGDAEMON       CLEANUP         USERSPACE1/USERSPACE1
CFG> cleanup all
Cleaning up for CORE...
Executing operations for state: CLEANUP
Done
Cleaning up for PVLACTIVITY...
Executing operations for state: CLEANUP
Done
Cleaning up for MIGPOL...
Executing operations for state: CLEANUP
Done
Cleaning up for PVR...
Executing operations for state: CLEANUP
Done
Cleaning up for RAITENGINE...
Executing operations for state: CLEANUP
Done
Cleaning up for SCLASS...
Executing operations for state: CLEANUP
Done
Cleaning up for LOGCLIENT...
Executing operations for state: CLEANUP
Done
Cleaning up for LOGDAEMON...
Executing operations for state: CLEANUP
Done
Cleaning up table used to track conversion... Done
Finished converting this database. Closing database connection.
cnvt> quit
```

When **cleanup all** is run, the backup and tracking tables are removed, and the database connection is closed. Note that connecting to the database again will recreate the conversion tracking table, however initialization will detect that the HPSS tables are already converted.

```
cnvt> db cfg
Creating table 'CONVERT_741' to track conversion progress...
Making entry for table CORE
Table CORE has already been converted
Making entry for table PVLACTIVITY
```

```
Table PVLACTIVITY has already been converted
Making entry for table MIGPOL
Table MIGPOL has already been converted
Making entry for table PVR
Table PVR has already been converted
Making entry for table RAITENGINE
Table RAITENGINE has already been converted
Making entry for table SCLASS
Table SCLASS has already been converted
Making entry for table LOGCLIENT
Table LOGCLIENT has already been converted
Making entry for table LOGDAEMON
Table LOGDAEMON has already been converted

Table Name      Conversion State Table/Index Space
--------------- --------------- -------------------------
CORE            DONE            USERSPACE1/USERSPACE1
PVLACTIVITY     DONE            USERSPACE1/USERSPACE1
MIGPOL          DONE            USERSPACE1/USERSPACE1
PVR             DONE            USERSPACE1/USERSPACE1
RAITENGINE      DONE            USERSPACE1/USERSPACE1
SCLASS          DONE            USERSPACE1/USERSPACE1
LOGCLIENT       DONE            USERSPACE1/USERSPACE1
LOGDAEMON       DONE            USERSPACE1/USERSPACE1
```

Since connecting to a converted database recreates the conversion tracking table even if all HPSS tables have been converted, **cleanup all** should be run again to remove the tracking table.

7.6.3.2. Cleanup the Subsystem Databases

The same steps used to clean up the CFG database should be run to clean up each subsystem database.

1. Become root

```
% su -
```

2. Use **hpss_convertdb** to cleanup

```
# /opt/hpss/tools/convert/hpss/hpss_convertdb
cnvt> db subsys1
CFG> cleanup all
cnvt> quit

# /opt/hpss/tools/convert/hpss/hpss_convertdb
HPSS_DB_INSTANCE_OWNER is set - Using 'hpssdb' for database connections.
Password for hpssdb:
cnvt> db subsys1
Using existing table 'CONVERT_741' to track conversion progress.

Table Name      Conversion State Table/Index Space
--------------- --------------- -------------------------
SSPVTAPE        CLEANUP         SSPVTAPE/SSPVTAPEIX
STORAGEMAPDISK  CLEANUP         STORAGEMAPDISK/STORAGEMAPDISKIX
STORAGESEGTAPE  CLEANUP         STORAGESEGTAPE/STORAGESEGTAPEIX
VVDISK          CLEANUP         VVDISK/VVDISKIX
SSPVHISTORY     CLEANUP         SSPVHISTORY/SSPVHISTORYIX
VVTAPE          CLEANUP         VVTAPE/VVTAPEIX
VVTAPEABSADDR   CLEANUP         <UNDEF>/<UNDEF>
```

```
SUBSYS1> cleanup all
Cleaning up for SSPVTAPE...
Executing operations for state: CLEANUP
Done
Cleaning up for STORAGEMAPDISK...
Executing operations for state: CLEANUP
Done
Cleaning up for STORAGESEGTAPE...
Executing operations for state: CLEANUP
Done
Cleaning up for VVDISK...
Executing operations for state: CLEANUP
Done
Cleaning up for SSPVHISTORY...
Executing operations for state: CLEANUP
Done
Cleaning up for VVTAPE...
Executing operations for state: CLEANUP
Done
Cleaning up for VVTAPEABSADDR...
Executing operations for state: CLEANUP
Done
Cleaning up table used to track conversion... Done
Finished converting this database. Closing database connection.
```

Chapter 8. HPSS 7.4.1 to HPSS 7.4.2 Conversion

This section summarizes the features and/or fixes introduced in HPSS 7.4.2 which require a metadata conversion. This is not a complete list of all new features and/or fixes. Refer to the HPSS 7.4.2 *Differences* section in the *HPSS Installation Guide* for more information.

8.1. General Information

The conversion from 7.4.1 to 7.4.2 involves configuration changes to support the following changes in HPSS:

- Conversion of PVR and PVLDRIVE table needed due to introduction of bug 1238 (CR262) described as "excessive pass through of cartridges".

8.2. Metadata Changes

This section provides additional information about the HPSS metadata changes in Release 7.4.2. The first section relates the new features and /or fixes described in the previous section to their specific metadata changes in DB2. The second section enumerates all DB2 metadata changes made to the global and subsystem databases.

8.2.1. New Features Affecting Conversion

Conversion of PVR and PVLDRIVE tables needed due to introduction of bug 1238 (CR262).

Change scheduling of defer dismount operations to avoid excessive pass through of cartridges and preserve performance.

8.2.2. Metadata Change Detail

The following table lists all the DB2 metadata changes in version 7.4.2 for the global (CFG) database.

8.2.2.1. HPSS Global (CFG) Database Changes

Table	**Change**
PVR	Add DD_EXEMPT_COUNT column.
PVLDRIVE	Add LIB_UNIT_ID column.

8.3. Conversion Overview

This section provides instructions for upgrading HPSS 7.4.1 DB2 metadata to release 7.4.2. To ensure that the conversion is successful, all steps must be followed in the order specified in the documentation. Prior to starting the conversion process, consult the HPSS 7.4.2 Release Notes for any special instructions.

8.3.1. Preparing to Upgrade

Before attempting any HPSS upgrade, follow the preparation procedure in *Gather Data On Existing System Section 1.3.1, "Gather Data On Existing System"*, *Benchmark Existing System Section 1.3.2, "Benchmark Existing System"* and *Backup Existing System Section 1.3.3, "Backup Existing System"* in this guide.

8.3.2. Software Prerequisites

The HPSS 7.4.2 prerequisite software is documented in the *HPSS 7.4.2 Installation Guide*. Depending on your current HPSS configuration and prerequisite software levels, there may be software components which must be upgraded prior to running the 7.4.2 metadata conversion. Follow the upgrade instructions provided by the prerequisite software to bring any software up to the level required by HPSS 7.4.2. Consult with your support representative for additional guidance.

8.3.3. Special Considerations for LTO PVR

In HPSS release 7.3.0, support for the LTO PVR was discontinued. However, customers running HPSS 7.3 with existing LTO PVR configurations were allowed to continue to use the library with no customer support. As of 7.4.2, metadata support for the LTO PVR has been removed and HPSS no longer allows any LTO PVR configurations. Any existing LTO PVR configurations need to be deleted prior to installing the HPSS 7.4.2 software. The HPSS SCSI PVR should be used instead. Use the following procedures before upgrading to remove the LTO PVR metadata:

- Shut down the LTO PVR
- Lock all LTO PVR drives
- Create a SCSI PVR (described in management guide) which uses the LTO PVR library
- Update the drives to use the SCSI PVR
- Move all cartridges from the LTO PVR to the SCSI PVR using the move cartridges operation
- Verify that all drives and cartridges have been removed from the LTO PVR
- Remove the LTO PVR

8.3.4. Conversion Planning

Careful planning for conversion is important in order to ensure a successful conversion. This section describes the planning steps that need to be completed before running conversion.

Shut down HPSS, put 7.4.2 code in place, then perform conversion.

Perform Configuration (CFG) database backup.

8.3.4.1. The Conversion Process

This script is a reworked version of **hpss_tape_pool** conversion script used in 6.2 conversion.

In summary, the utility performs the following:

- *connects to config (CFG) database*
- *checks if conversion has already taken place*
- *creates new copies of targeted tables and populates with original data*
- *adds DD_EXEMPT_COUNT column to the PVR table*
- *adds LIB_UNIT_ID column to the PVLDRIVE table*
- *performs reorg and verifies number of rows in each table*
- *verifies columns have been added and prints results*

8.3.4.2. Conversion - Detailed Instructions

`HPSS_INSTALL_PATH/tools/convert742/hpss_exempt_defer_dismount` is intended to be run as root or as instance owner "hpssdb".

```
[root@]#./hpss_exempt_defer_dismount -db cfg -s hpss

LIB_UNIT_ID column does not exist in hpss.pvldrive table.
Table hpss.PVLDRIVE_PRE_DD_EXEMPT created
The hpss.PVLDRIVE_PRE_DD_EXEMPT table now has the following number of
records:
          4
dd_exempt_count column does not exist in hpss.pvr table.
Table hpss.PVR_PRE_DD_EXEMPT created
The hpss.PVR_PRE_DD_EXEMPT table now has the following number of records:
          1
Successfully completed. You now have:
        1.) a LIB_UNIT_ID column in the hpss.pvldrive table initialized to -1
for all rows.
        2.) a DD_EXEMPT_COUNT column in the hpss.pvr table initialized to -1 for
all rows.
        3.) an hpss.PVLDRIVE_PRE_DD_EXEMPT table in case you need to revert to
previous table definition or metadata
        4.) an hpss.PVR_PRE_DD_EXEMPT table in case you need to revert to
previous table definition or metadata.
```

8.3.4.3. Reversion - Detailed Instructions

If you have to revert to pre-conversion state, perform either one of the following methods:

- Restore to the last pre-conversion configuration (CFG) database backup
- Drop the added columns from the new tables using the following commands:

  1. `db2 alter table hpss.pvr drop column DD_EXEMPT_COUNT`

2. `db2 alter table hpss.pvldrive drop column LIB_UNIT_ID`

3. `db2 reorg table hpss.pvr`

4. `db2 reorg table hpss.pvldrive`

The **hpss_exempt_defer_dismount** utility does NOT remove (drop) the original renamed tables "PVR_PRE_DD_EXEMPT" and "PVR_PRE_DD_EXEMPT".

Chapter 9. HPSS 7.4.2 to HPSS 7.4.3 Conversion

This section summarizes the change introduced in HPSS 7.4.3 which require a metadata conversion. Refer to the HPSS 7.4.3 to the *HPSS Installation Guide* for more information.

9.1. General Information

The conversion from 7.4.2 to 7.4.3 involves metadata and other HPSS system administrative configuration changes to support the following change in HPSS:

- Migration Purge Server
  - Under the auspices of BZ2218 the MPS’s non-client-facing interface from version 3 to version 4. removal of long-deprecated (and long-unused) function from the interface as part of this work. Since the interface is changing, the version number is incremented.
  - Addressing Bug 5020 - Conversion from 743p1 to 743p2 requires grants statements to be applied to MpsForceMigrate table.
  - Addressing Bug 5342 - 'Migration issues after upgrade to 742p1 (CR194, Part2) requires metadata change affecting MIGPOL table
- Storage Server/Core Server
  - Addressing Bug 3291 - "Error recovery from failed tape/disk mover is incomplete" resulted in a need to set the "control" bit permission to the PVL ACL entry for the core server. Otherwise PVL will fail to start.
  - Addressing Bug 205 - "Tapes that contain no files should be reclaimed automatically" would necessitate a new index creation on table SSPVHISTORY.
  - Addressing Bug 4312 - FragmentTrimLimit and FragmentSmallestBlock no longer work correctly affecting CORE table
  - Addressing Bug 4277 - large number of NSTRASH table entries causes the core server to take too long to startup
  - Addressing Bug 5123: HPSS has a 64K limit on subdirectories within a directory affecting NSOBJECT table
  - Addressing Bug 4848 - SSPVHISTORY: A unique index cannot be created because the table contains data that would result in duplicate index entries.
- Improved Recover Utility
  - New metadata table MPSFORCEMIGRATE to support the revised recover utility

- HPSS Trashcan new feature
  - New metadata table and changes to support the HPSS Trashcan new feature
- HPSS Unix Authentication and "lspvhist" issue
  - Bug 3310 modifies HPSS to use PAM. Common functions such as password file manipulation and password hash generation have been consolidated into a single library. Both the user tools and the UNIX auth/authn libraries are refactored to use the common functions.
  - Bug 3938 addresses the fact that HPSS storage server doesn’t have insert rights to "SSPVHISTORY" table by making sure to add hpss to the hpsssrvr group.

9.2. Metadata Changes

This section provides additional information about the HPSS metadata change in Release 7.4.3.

9.2.1. Metadata Change to Global (CFG) Database

The following table list DB2 metadata changes in version 7.4.3 for the global database (CFG).

The MPS (Migration Purge Server) non-client-facing interface is incremented from version 3 to version 4 since the interface is changing. The code change will require an update to the SERVERINTERFACES metadata table.

Addressing Bug 3291 "Error recovery from failed tape/disk mover is incomplete" resulted in a need to set the "control" bit permission to the PVL ACL entry for the core server. Otherwise, PVL will fail to start. This change will require an update to the AUTHZACL metadata table via the **hpss_acl_server** utility.

So, in 7.4.3 and beyond the PVL ACL’s would look like this:

```
perms - type - ID (name) - realm ID (realm)
============================================
rw-c-dt - user - 302 (hpssssm) - 120500 (lanl.gov)
rw-c-dt - user - 311 (hpsscore) - 120500 (lanl.gov)
rw---dt - user - 312 (hpsspvr) - 120500 (lanl.gov)
------t - any_other
```

Addressing CR 241: HPSS Trashcan new feature

Addressing Bug 4312 FragmentTrimLimit and FragmentSmallestBlock no longer work correctly

Addressing Bug 5342 - Migration issues after upgrade to 742p1 (CR194, Part2) requires metadata change affecting MIGPOL table

Table	Change
SERVERINTERFACES	UPDATE hpss.serverinterfaces SET version=4 WHERE server_id IN (SELECT server_id FROM hpss.server WHERE server_type=14)
AUTHZACL	set the "control" bit permission to the PVL’s ACL entry for the core server via hpss_server_acl utility.

Table	Change
GLOBAL	Alter and update GLOBAL table to support Trashcan
CORE	Drop FragmentTrimLimit column from CORE table
MIGPOL	Add new column min_minutes_disk_migr_io to MIGPOL table

9.2.2. Metadata Change to Subsystems (SUBSYS) Databases

The following table list DB2 metadata changes in version 7.4.3 for the subsystem(s) databases (SUBSYS).

Addressing Bug 205 - "Tapes that contain no files should be reclaimed automatically" would necessitate a new index creation on table SSPVHISTORY.

Addressing CR 249 - Improved "recover" utility

Addressing CR 241 - HPSS Trashcan new feature

Addressing Bug 4277 - core server takes longer to start due to large number of NSTRASH table entries

Addressing Bug 4848 - SSPVHISTORY: A unique index cannot be created because the table contains data

Addressing Bug 5123: HPSS has a 64K limit on subdirectories within a directory NSOBJECT table

Table	Change
SSPVHISTORY	CREATE INDEX hpss.sspvhistory_last_empty ON hpss.sspvhistory (vvid asc, operation asc, timestamp desc, comment asc)
MPSFORCEMIGRATE	Create new table spaces (MPSFORCEMIGRATE, MPSFORCEMIGRATEIX), add MPSFORCEMIGRATE table and new triggers on the VVDisk and VVTape tables
NSTRASH	Create new table spaces (NSTRASH, NSTRASHIX), add NSTRASH table and NSTRASHVIEW
BITFILE	CREATE UNIQUE INDEX bitfile _pkey ON hpss.bitfile (bfid asc) INCLUDE (bfattr_data_len)
SSPVHISTORY	Drop the use of unique from SSPVHISTORY INDEX. Uniqueness not needed and reverts the change from Bug 205.

Table	Change
NSOBJECT	ALTER table hpss.nsobject ALTER COLUMN link_count SET DATA TYPE BIGINT

9.3. HPSS Unix Authentication

Bug 3310 modifies HPSS to use PAM. Common functions such as password file manipulation and password hash generation have been consolidated into a single library. Both the user tools and the UNIX auth/authn libraries are refactored to use the common functions.

9.4. LSPVHIST Is Broken When Converting from 7.4.2 to 7.4.3

Bug 3938 addresses the fact that HPSS storage server doesn’t have insert rights to "SSPVHISTORY" table by making sure to add hpss to the hpsssrvr group.

9.5. Conversion Overview

This section provides instructions for upgrading HPSS 7.4.2 DB2 metadata to release 7.4.3. To ensure that the conversion is successful, all steps must be followed in the order specified in the documentation. Prior to starting the conversion process, consult the HPSS 7.4.3 Release Notes for any special instructions.

9.5.1. Software Prerequisites

The HPSS 7.4.3 prerequisite software is documented in the *HPSS 7.4.3 Installation Guide*. Depending on your current HPSS configuration and prerequisite software levels, there may be software components which must be upgraded prior to running the 7.4.3 metadata conversion. Follow the upgrade instructions provided by the prerequisite software to bring any software up to the level required by HPSS 7.4.3. Consult with your support representative for additional guidance.

The conversion process itself uses the Perl Database Interface (DBI) to interact with DB2. DBI provides a uniform interface to a number of Database engines, including DB2. DBI is available as part of the operating system on all supported Linux distributions. For sites running DB2 on AIX, a precompiled DBI RPM is available through your support representative. Alternatively, DBI in source form is available for download from http://www.cpan.org.

Because of licensing restrictions, the Database Driver (DBD) needed to interact with DB2 is not included in standard Linux distributions. The DB2 DBD was developed by and is copyrighted to IBM. Precompiled RPMs for AIX and LINUX are available from your support representative. DB2 DBD sources are also available for download from http://www.cpan.org.

On Linux version 2.6.32-358.el6.x86_64, the 7.4.3 conversion process was developed and tested with DBD version 1.84, DBI version 1.609, DB2 version 9.7.0.9 Fix Pack 9a and 10.5.0.3 Fix Pack 3a.

On AIX version 4.2.3.4, the 7.4.3 conversion process was developed and tested with perl-DBD-DB2-1-1, DBI version 1.611-2, DB2 version 9.7.0.9 Fix Pack 9a and 10.5.0.3 Fix Pack 3a.

9.5.2. Conversion Planning

Careful planning for conversion is important in order to ensure a successful conversion. This section describes the planning steps that need to be completed before running conversion.

Perform an online or offline backup of global (CFG) database.

Shutdown down HPSS, but keep DB2 up and running; otherwise, start DB2 using *db2start*. Put 7.4.3 code in place, then perform conversion.

Use **hpss_convertdb743** to make HPSS administrative changes.

Use **hpss_convertdb743** to perform the metadata conversion on:

- CFG global database listed tables
  - SERVERINTERFACES
  - GLOBAL
  - CORE
  - MIGPOL
- SUBSYS databases listed tables
  - SSPVHISTORY
  - NSOBJECT
  - MPSFORCEMIGRATE
  - NSTRASH
  - BITFILE

Use **hpss_acl_server** utility to set the "control" bit permission to the PVL’s ACL entry for the core server on the CFG global database affecting AUTHZACL table.

Use hpss_managetables to drop/add views, constraints, and indexes.

9.5.3. Tools Used by the Conversion Process

Conversion uses **hpss_convertdb743** tool, **hpss_managetables** and **hpss_acl_server** utilities to perform the actual metadata conversions and HPSS administrative configuration changes.

The first tool **hpss_convertdb743** is only used for the conversion process, and is not installed in the standard `bin` location. Instead, it is located in `HPSS_INSTALL_PATH/tools/convert743/hpss` beneath the root of the HPSS installation. **hpss_convertdb743** must be run as the root user, and requires the user to enter the password for the HPSS instance owner (typically hpssdb. db2hpss in this example)) in order to execute. Database operations are executed as the instance owner (example "db2hpss"). This is required in order to set the proper permissions on database table created during the conversion process.

```
# /opt/hpss/tools/convert743/hpss/hpss_convertdb743

HPSS_DB_INSTANCE_OWNER is set - Using 'db2hpss' for database connections.
Password for db2hpss:
cnvt>
```

hpss_convertdb743 recognizes the following commands:

```
cnvt> help

Available commands:
   cleanup              Cleanup after conversion
   close                Disconnect from a database
   convert              Run the conversion
   count                Count the number of rows in a table
   db                   Connect to database
   env                  Dump configuration parameters
   help                 Display available commands
   pam_hpsssrv          Perform unix authentication, call PAM setup, and modify authz.con
   revert               Revert the conversion
   rsql                 Show the SQL that will be run to revert a table
   showts               Show table space info for tables that will be converted
   sql                  Show the SQL that will be run to convert a table
   state                Show the current state of conversion
   ts                   Set the table space to use
cnvt>
```

Additional description of the **hpss_convertdb743** commands are below.

cleanup <all | table_name> - After a successful conversion, delete the backup copy of **table_name**. The operation requires a database connection. Once run, the conversion cannot be reverted.

close - close a database connection.

count - <all | table_name> - Count the number of rows in **table_name** specific to the site. This operation requires a database connection.

convert <all | table_name> - Convert the indicated table. The operation requires a database connection. If **table_name** isn't specified, then convert all tables in the connected database.

db <db_name> - Build a database connection to **db_name**. Note that **hpss_convertdb** expects the database alias name (CFG) to be provided, rather than the database name. This is so it can verify the name of subsystem databases in the global metadata.

env - Display database information and configuration parameters.

pam_hpsssrv - Add user HPSS to HPSSSRV group file, run PAM setup, and edit libhpssunixauth.so to libhpssunixauthz.so in the authz.conf file.

revert <all | table_name> - Revert the specified table. The operation requires a database connection. This option is meant to be used only by HPSS support since there may be additional steps needed to completely revert to the previous HPSS level.

rsql <table_name> - Show the SQL that will be used to revert the specified table. This requires a database connection.

showts - Show the table spaces that will be used for the conversion. The operation requires a database connection. Before a table can be converted, it must have a table space associated with it to hold the table and indexes.

sql <table_name> - Show the SQL that will be used to convert the specified table. This operation requires a database connection.

state - Show the conversion state for the table. This requires a database connection. Each table will be in one of the following states: START, CREATE, FILL, DEPLOY, CLEANUP or DONE.

ts <table_name> - Set the table spaces that will be used for table **table_name**. This requires a database connection.

The first time **hpss_convertdb743** connects to a database, it creates a table to track the conversion process. For the 7.4.3 conversion, the table is called CONVERT_743. The table contains a row for each of the HPSS tables that will be converted. As a table is being worked on, the state maintained in the row will change to reflect the conversion's progress. It is important not to delete or modify this tracking table. Doing so will cause the conversion process to fail and may leave the metadata in an inconsistent state.

The second tool **hpss_managetables** is an hpss utility included in the HPSS packaged software and located in standard `bin` location by default and must be invoked as root also.

```
# /opt/hpss/bin/hpss_managetables

hmt> help
Available commands:
> db <database>
  select a database

> ts <table table space> <index table space>
  select default table spaces for creates

> add [{allglobal|allsubsys}] [<table>[:[<table_tbsp>][:<index_tbsp>]]] ...
 create table(s)
     allglobal - all global tables
     allsubsys - all subsys tables
     <table>[:[<tbsp1>][:<tbsp2>]] - use tbsp1 for table and tbsp2 for index
     (omit either table space to use default)

> add indexes <table> [list of index, constraint or trigger names]
  adds the indexes, table constraints and triggers for the named table
     if the list of names is omitted, all indexes, constraints and triggers
are added
     if a list of names is provided, only those indexes, constraints or
triggers are added

> add indexes <table> notriggers
adds all of the indexes to a table, but leaves out triggers and procedures

> add {global|subsys} {constraints|views} [list of names]
  add global or subsys RI constraints or views
     adds all global or subsys RI constraints or views if the names are
omitted

> del {allglobal|allsubsys|<table>}
```

```
  delete table(s)
   allglobal - delete all global tables
     allsubsys - delete all subsys tables
     <table> - delete the named table

> del indexes <table> [index, constraint or trigger name]
deletes all indexes, table constraints and triggers on the named table
     when the index, constraint or trigger name is provided,
     only that index, constraint or trigger is deleted

> del {global|subsys} {constraints|views} [list of names]
  delete global or subsys RI constraints or views
     deletes all global or subsys RI constraints or views if the names are
omitted
```

The third and last utility **hpss_acl_server** is an hpss utility included in the HPSS packaged software and located in standard `bin` location by default and must be invoked as root also.

```
# /opt/hpss/bin/hpss_server_acl

hsa> help
Available commands:

  To select an ACL:

  (by server options)
> acl -n "server name" [<interface>]
> acl -u <server uuid> [<interface>]
> acl -t <server type> [<interface>] (see below)

  Interface selection options (for use with the above)
  -U <if uuid> [-v <version>]
  -d "if desc" [-v <version>]
  -T <standard if type> (see below)

  (from a menu)
> acl -m

  Show current server/interface:
> acl

  Show current ACL:
> show

  Set all this server's ACLs to defaults for this server type:*
> default

  Empty ACL:
> clear

  Add an entry/perms:
> add <type> <args> <perms>
```

Conversion - Detailed Instructions

This section provides detailed instructions for executing the 7.4.3 conversion. If you have questions about the conversion process, contact your support representative before attempting the conversion. This section assumes that the prerequisite steps have been completed, the ASLR feature has been turned off on Linux systems, the HPSS 7.4.3 software has been installed, and you are ready to use **hpss_server_acl**, **hpss_convertdb743**, and **hpss_managetables** to run the conversion.

9.5.4. HPSS UNIX Authentication Using PAM, modifying authz.conf and adding hpss to hpsssvr group

The first step in the conversion process is to use **hpss_convertdb743** to reflect changes to UNIX authentication, add hpss to hpsssrv group and make changes to authz.conf.

Become root

```
% su -
# source ~db2hpss/sqllib/db2profile  (db2hpss is db2 instance in this example)

#/opt/hpss/tools/convert743/hpss/hpss_convertdb743

convt> pam_hpsssrv

'hpss' user is already a member of 'hpsssrvr' in '/var/hpss/etc/group'
[ setting up PAM config ]
[ using local HPSS passwords ]
/var/hpss/etc/authz.conf has already been modified.
```

The above command performs the following:

- add user hpss to hpsssrv group
- makes an `/var/hpss/etc/authz.conf.orig` and modifies `/var/hpss/etc/authz.conf` so `libhpssunixauth.so` should now be `libhpssunixauthz.so`
- executes `/opt/hpss/config/setup_pam.pm`

Note that the new /var/hpss/etc/authz.config will need to be copied to all the movers and clients, otherwise you will encounter mover’s errors.

If you need to revert to previous version of HPSS, there is no action to take to revert the changes introduced during the `/opt/hpss/config/setup_pam.pm` due to the way PAM works. The "hpss" stack added to PAM will just sit there inert if not being used by HPSS.

However `/var/hpss/etc/authz.conf` changes need to be reverted to `/var/hpss/etc/authz.conf` so `libhpssunixauthz.so` should now be `libhpssunixauth.so` by performing the following:

```
mv /var/hpss/etc/authz.conf /var/hpss/etc/authz.743
```

and

```
mv /var/hpss/etc/authz.conf.orig /var/hpss/etc/authz.conf
```

9.5.5. hpss_server_acl Utility

The second step in the conversion process is to use **hpss_server_acl** HPSS utility to set the "control" bit permission to the PVL’s ACL entry for the core server on the CFG global database affecting AUTHZACL table.

1. Become root

   ```
   % su -
   ```

2. Use **hpss_server_acl** utility to convert the CFG database. The commands needed to convert the table in the CFG database are shown next.

   ```
   #/opt/hpss/bin/hpss_server_acl
   hsa> acl -m
   1) Core Server
   2) Mover - Tape (hpss-linux)
   3) Startup Daemon (dev03-svr)
   4) Migration/Purge Server
   5) Gatekeeper
   6) STK PVR
   7) Log Daemon
   8) SSM System Manager
   9) Mover - Disk (dev03-svr)
   10) Log Client (dev03-svr)
   11) Location Server
   12) PVL
   Select a server
   Choose an item by number (RET to cancel):
   > 12   (Make sure you select PVL)
   hsa> show
   perms - type - ID (name) - realm ID (realm)
   ===========================================
   rw-c-dt - user - 302 (hpssssm) - 120500 (lanl.gov)
   rw---dt - user - 311 (hpsscore) - 120500 (lanl.gov)
   rw---dt - user - 312 (hpsspvr) - 120500 (lanl.gov)
   ------t - any_other

   hsa> add user hpsscore c
   hsa> show
   perms - type - ID (name) - realm ID (realm)
   ===========================================
   rw-c-dt - user - 302 (hpssssm) - 120500 (lanl.gov)
   rw-c-dt - user - 311 (hpsscore) - 120500 (lanl.gov)
   rw---dt - user - 312 (hpsspvr) - 120500 (lanl.gov)
   ------t - any_other
   hsa> quit
   ```

If you need to revert to the original AUTHZACL table content, follow the same instructions listed above using **hpss_server_acl** utility and issue the following:

```
hsa> del user hpsscore c
```

9.5.6. Convert the Global Database (CFG)

The third step in the conversion process is to use **hpss_convertdb743** to convert the global (CFG) database SERVERINTERFACES, GLOBAL, MIGPOL and CORE tables.

1. Become root

   ```
   % su -
   # source ~db2hpss/sqllib/db2profile  (db2hpss is db2 instance in this example)
   ```

2. Use **hpss_convertdb743** to convert the CFG database. The commands needed to convert the table in the CFG database are shown next, followed by a transcript from a run of the commands.

```
#/opt/hpss/tools/convert743/hpss/hpss_convertdb743

HPSS_DB_INSTANCE_OWNER is set - Using "db2hpss" for database connections.
Password for db2hpss:     <enter your instance password here>
cnvt> db cfg
Creating table 'CONVERT_743' to track conversion progress...
Making entry for table SERVERINTERFACES
Making entry for table GLOBAL
Making entry for table CORE
Making entry for table MIGPOL
Table Name      Conversion State Table/Index Space
--------------- --------------- -------------------------
SERVERINTERFACES START           USERSPACE1/USERSPACE1
GLOBAL           START           USERSPACE1/USERSPACE1
CORE             START           USERSPACE1/USERSPACE1
MIGPOL           START               USERSPACE1/USERSPACE1

CFG> count all  (result is site specific)

Table Name      Number of Rows
--------------- ---------------
SERVERINTERFACES             24
GLOBAL                       1
CORE                         1
MIGPOL                                      2

CFG> convert <all><tablename>
CFG> convert all
Converting SERVERINTERFACES...
Executing operations for state: START
Executing operations for state: CREATE
Executing operations for state: FILL
Executing operations for state: DEPLOY
Conversion Time: 0 minutes 1 seconds
Done
Converting GLOBAL...
Executing operations for state: START
Executing operations for state: CREATE
Executing operations for state: FILL
Executing operations for state: DEPLOY
DB20000I  The REORG command completed successfully.
Conversion Time: 0 minutes 1 seconds
Done
Converting CORE...
Executing operations for state: START
Executing operations for state: CREATE
Executing operations for state: FILL
Executing operations for state: DEPLOY
DB20000I  The REORG command completed successfully.
Conversion Time: 0 minutes 1 seconds
Done
Converting MIGPOL...
Executing operations for state: DEPLOY
DB20000I  The REORG command completed successfully.
Conversion Time: 0 minutes 0 seconds
Done
```

```
CFG> count all

Table Name      Number of Rows
--------------- ---------------
SERVERINTERFACES             24
GLOBAL                       1
CORE                         1
MIGPOL                                        2
```

If you need to revert to the original table(s) is to right stage to issue the following command most importantly before issuing **cleanup all** to rid of conversion tables. Make sure you are in "cleanup" state of conversion by issuing **state** command.

```
CFG> state
Table Name      Conversion State Table/Index Space
--------------- --------------- ------------------------
SERVERINTERFACES CLEANUP         USERSPACE1/USERSPACE1
GLOBAL           CLEANUP         USERSPACE1/USERSPACE1
CORE             CLEANUP         USERSPACE1/USERSPACE1
MIGPOL           CLEANUP         USERSPACE1/USERSPACE1

CFG> revert all
Reverting SERVERINTERFACES...
Running revert operations
Done
Reverting GLOBAL...
Running revert operations
Done
Reverting CORE...
Running revert operations
B20000I  The REORG command completed successfully.
Done
Reverting MIGPOL...
Running revert operations
DB20000I  The REORG command completed successfully.
Done
```

This action put back the "CFG" database target table(s) in the state of "Deploy" as shown below and conversion can be resumed from this point by issuing the command **convert all** at this stage if need to be.

```
CFG> state
Table Name      Conversion State Table/Index Space
--------------- --------------- ------------------------
SERVERINTERFACES DEPLOY          USERSPACE1/USERSPACE1
GLOBAL           DEPLOY          USERSPACE1/USERSPACE1
CORE             DEPLOY          USERSPACE1/USERSPACE1
MIGPOL           DEPLOY          USERSPACE1/USERSPACE1
```

Otherwise after CFG conversion is completed then close connection, quit, and resume SUBSYS conversion.

```
CFG> close
cnvt> quite
```

9.5.7. Convert the SUBSYS Databases

The fourth step in the conversion process is to use **hpss_managetables** and **hpss_convertdb743** to convert the SUBSYS database(s).

9.5.7.1. Use hpss_managetables to Drop SUBSYS Views and only SSPVHISTORY table constraint named *SSPVHISTORYCON1*

The commands needed to drop SUBSYS views and the single constraint are shown next, followed by a transcript from a run of the commands.

If you run into the following error while trying to drop either "Views" or "Constraints":

```
SQL0551N  The statement failed because the authorization ID does not have the required
authorization or privilege to perform the operation.  Authorization ID: "HPSS".

Operation: "DROP VIEW". Object:
```

you need to grant "hpss" user DBADM authority and also grant the needed privileges on the offending table or view to user "hpss".

Make sure you exit or quit out of hpss_managetables tool. Connect to SUBSYS database(s) and run *grant* command for these common tables as shown in the example below:

```
db2 grant control on table hpss.STORAGESEGDISKVIEW to user hpss
db2 grant control on table hpss.STORAGESEGTAPEVIEW to user hpss
db2 grant control on table hpss.VVDISKVIEW to user hpss
db2 grant control on table hpss.VVTAPEVIEW to user hpss
db2 grant control on table hpss.VVTAPEMAPVIEW to user hpss
db2 grant control on table hpss.BFMIGRRECBITFILEVIEW to user hpss
db2 grant control on table HPSS.SSPVHISTORY to user hpss
```

Become root

```
% su -
```

and run **hpss_managetables** in the following order:

```
#/opt/hpss/bin/hpss_managetables
hmt> db subsys1

Available tablespaces:
                                         Owner Data
ID Tablespace Name            Owner      Type  Type
2  USERSPACE1                 SYSIBM     S     L
3  ACCOUNTING                 DB2HPSS    U     L
4  ACCOUNTINGIX               DB2HPSS    U     L
```

and many more tables listed here…..

```
Database: subsys1 [Subsystem database]
Schema: HPSS
Default tablespace: NOT SELECTED
Default index tablespace: NOT SELECTED

hmt> del subsys views
```

```
Deleting BFMIGRRECBITFILEVIEW
Deleting STORAGESEGDISKVIEW
Deleting STORAGESEGTAPEVIEW
Deleting VVDISKVIEW
Deleting VVTAPEVIEW
Deleting VVTAPEMAPVIEW

hmt> del subsys constraints SSPVHISTORYCON1

Deleting SSPVHISTORYCON1

hmt> commit
Transaction committed

hmt> quit
```

9.5.7.2. Use hpss_convertdb743 to Convert the SUBSYS Database(s) BITFILE, SSPVHISTORY, NSTRASH and MPSFORCEMIGRATE Tables

The commands needed to convert the table in the SUBSYS databases are shown next, followed by a transcript from a run of the commands.

```
% su -
# source ~db2hpss/sqllib/db2profile (db2hpss is db2 instance in this example)
#/opt/hpss/tools/convert743/hpss/hpss_convertdb743
cnvt> db subsys1

Creating table 'CONVERT_743' to track conversion progress...
Making entry for table BITFILE
Making entry for table NSOBJECT
Making entry for table SSPVHISTORY
Making entry for table NSTRASH
Making entry for table MPSFORCEMIGRATE

Table Name      Conversion State Table/Index Space
--------------- --------------- ------------------------
BITFILE         START           BITFILE/BITFILEIX
NSOBJECT        START           NSOBJECT/NSOBJECTIX
SSPVHISTORY     START           SSPVTAPE/SSPVTAPEIX
NSTRASH         START           <UNDEF>/<UNDEF>
MPSFORCEMIGRATE START           <UNDEF>/<UNDEF>

SUBSYS1> count all (Site specifici result)

Table Name      Number of Rows
--------------- ---------------
BITFILE                       7
NSOBJECT                     14
SSPVHISTORY                   0
NSTRASH                 <undef>
MPSFORCEMIGRATE         <undef>

SUBSYS1> state

Table Name      Conversion State Table/Index Space
```

```
--------------- -------------- -------------------------
BITFILE         START          BITFILE/BITFILEIX
NSOBJECT        START          NSOBJECT/NSOBJECTIX
SSPVHISTORY     START          SSPVTAPE/SSPVTAPEIX
NSTRASH         START          <UNDEF>/<UNDEF>
MPSFORCEMIGRATE START          <UNDEF>/<UNDEF>

SUBSYS1> convert <all> <tablename>

SUSSYS1> convert all
Converting BITFILE...
Executing operations for state: START
Executing operations for state: CREATE
Executing operations for state: FILL
Executing operations for state: DEPLOY

Database Connection Information

Database server        = DB2/LINUXX8664 10.5.0
SQL authorization ID   = DB2HPSS
Local database alias   = SUBSYS1

SQL0598W Existing index "HPSS.BITFILE_PKEY" is used as the index for the
primary key or a unique key.  SQLSTATE=01550
Conversion Time: 0 minutes 2 seconds
Done
Converting NSOBJECT...
Executing operations for state: START
Executing operations for state: CREATE
Executing operations for state: FILL
Executing operations for state: DEPLOY
DB20000I  The REORG command completed successfully.
SQL0598W  Existing index "HPSS.NSOBJECT_PKEY" is used as the index for the
primary key or a unique key.  SQLSTATE=01550
Conversion Time: 0 minutes 0 seconds
Done
Converting SSPVHISTORY...
Executing operations for state: START
Executing operations for state: CREATE
Executing operations for state: FILL
Executing operations for state: DEPLOY
Conversion Time: 0 minutes 0 seconds
Done
Converting NSTRASH...
Executing operations for state: START
Executing operations for state: CREATE
Executing operations for state: FILL
Executing operations for state: DEPLOY

Database Connection Information

Database server        = DB2/LINUXX8664 10.5.0
SQL authorization ID   = DB2HPSS
Local database alias   = SUBSYS1

SQL0598W Existing index "HPSS.NSTRASH_PKEY" is used as the index for the
primary key or a unique key.  SQLSTATE=01550
Conversion Time: 0 minutes 2 seconds
Done
```

```
Converting MPSFORCEMIGRATE...
Executing operations for state: START
Executing operations for state: CREATE
Executing operations for state: FILL
Executing operations for state: DEPLOY
Conversion Time: 0 minutes 0 seconds
Done

SUBSYS1> count all     (site specific result)

SUBSYS1> count all

Table Name Number of Rows
--------------- ---------------
BITFILE                       7
NSOBJECT                     14
SSPVHISTORY                   0
NSTRASH                       0
MPSFORCEMIGRATE               0

SUBSYS1> state

Table Name      Conversion State Table/Index Space
--------------- --------------- -------------------------
BITFILE         CLEANUP          BITFILE/BITFILEIX
NSOBJECT        CLEANUP          NSOBJECT/NSOBJECTIX
SSPVHISTORY     CLEANUP          SSPVTAPE/SSPVTAPEIX
NSTRASH         CLEANUP          NSTRASH/NSTRASHIX
MPSFORCEMIGRATE CLEANUP          MPSFORCEMIGRATE/MPSFORCEMIGRATEIX
```

If you need to revert to the original table(s), this is the right stage to issue the following command most importantly before issuing **cleanup all** to rid of conversion tables. Make sure you are in "Cleanup" state of conversion by issuing **state** command.

```
SUBSYS1> revert all or revert <tablename>

Reverting BITFILE...
Running revert operations
Done
Reverting NSOBJECT...
Running revert operations
Done
Reverting SSPVHISTORY...
Running revert operations
Done
Reverting NSTRASH...
Running revert operations
Done
Reverting MPSFORCEMIGRATE...
Running revert operations
Done
```

This action put back the "SUBSYS" databases target table(s) in the state of "Deploy" as shown below and conversion can be resumed from this point by issuing the command **convert all** or **convert <tablename>** at this stage if need to be.

```
SUBSYS1> state

Table Name      Conversion State Table/Index Space
```

```
--------------- --------------- -------------------------
BITFILE         DEPLOY          BITFILE/BITFILEIX
NSOBJECT        DEPLOY          NSOBJECT/NSOBJECTIX
SSPVHISTORY     DEPLOY          SSPVTAPE/SSPVTAPEIX
NSTRASH         DEPLOY          NSTRASH/NSTRASHIX
MPSFORCEMIGRATE DEPLOY          MPSFORCEMIGRATE/MPSFORCEMIGRATEIX
```

9.5.7.3. Use hpss_managetables to Add SUBSYS Views, *SSPVHISTORYCON1* Constraint and Triggers

The commands needed to add SUBSYS views, *SSPVHISTORYCON1* constraint and triggers on VVDISK and VVTAPE tables are shown next, followed by a transcript from a run of the commands.

Become root

```
% su -
```

and run **hpss_managetables** in the following order:

```
#/opt/hpss/bin/hpss_managetables
hmt> db subsys1
Available tablespaces:
                                          Owner Data
ID  Tablespace Name           Owner      Type  Type
2   USERSPACE1                SYSIBM     S     L
3   ACCOUNTING                DB2HPSS    U     L
4   ACCOUNTINGIX              DB2HPSS    U     L
```

and many more tables listed here…..

```
Database: subsys1 [Subsystem database]
Schema: HPSS
Default tablespace: NOT SELECTED
Default index tablespace: NOT SELECTED

hmt> add subsys views

Adding BFMIGRRECBITFILEVIEW
Adding NSTRASHVIEW
Adding STORAGESEGDISKVIEW
Adding STORAGESEGTAPEVIEW
Adding VVDISKVIEW
Adding VVTAPEVIEW
Adding VVTAPEMAPVIEW

hmt> add subsys constraints SSPVHISTORYCON1

Adding SSPVHISTORYCON1

hmt> add indexes vvdisk
Creating indexes for table VVDISK ALTER TABLE
VVDISK ADD CONSTRAINT VVDISK_PKEY PRIMARY KEY ( VVID) The primary key was
not created because VVDISK already has a primary key. This is assumed not
to be an error.  If this is an error, you must remove the primary key
constraint and then run this command again. Use 'del indexes VVDISK' to
remove the indexes and primary key constraint. CREATE OR REPLACE TRIGGER
VVDISK_D_CHK_MPSFORCEMIGRATE BEFORE DELETE ON VVDISK REFERENCING OLD AS O
FOR EACH ROW WHEN (EXISTS(SELECT ORIG_VVID FROM MPSFORCEMIGRATE WHERE
ORIG_VVID = O.VVID)) BEGIN ATOMIC SIGNAL SQLSTATE 'Z0003' ('Foreign Key
Violation, child rows exist in table MPSFORCEMIGRATE ') ; END

hmt> add indexes vvtape
Creating indexes for table VVTAPE ALTER TABLE VVTAPE ADD CONSTRAINT
VVTAPE_PKEY PRIMARY KEY ( VVID) The primary key was not created
because VVTAPE already has a primary key. This is assumed not to be an
error.  If this is an error, you must remove the primary key
constraint and then run this command again. Use 'del indexes VVTAPE'
to remove the indexes and primary key constraint. CREATE OR REPLACE
TRIGGER VVTAPE_D_CHK_MPSFORCEMIGRATE BEFORE DELETE ON VVTAPE
REFERENCING OLD AS O FOR EACH ROW WHEN (EXISTS(SELECT ORIG_VVID FROM
MPSFORCEMIGRATE WHERE ORIG_VVID = O.VVID)) BEGIN ATOMIC SIGNAL
SQLSTATE 'Z0003' ('Foreign Key Violation, child rows exist in table
MPSFORCEMIGRATE ') ; END

hmt> commit
Transaction committed

hmt> quit
```

Post-Conversion Detailed Instructions

This section describes steps that need to be taken after the metadata conversion is completed.

9.5.8. Run Full Database Backups

A full backup of CFG database and SUBSYS(s) after conversion is desirable.

9.5.8.1. Cleanup

Once satisfied with conversion to the 7.4.3 system, the table used to track the conversion process as well as the renamed pre-conversion affected tables should be removed. After this step is completed, the conversion **CANNOT** be reverted.

9.5.8.2. Cleanup the Global Database

When **cleanup all** is run, tracking tables as well as renamed pre-conversion affected tables are dropped, and the database connection is closed.

1. Become root

   ```
   % su -
   ```

2. Use **hpss_convertdb743**

   ```
   # source ~db2hpss/sqllib/db2profile   (db2hpss is db2 instance owner in this example)
   # /opt/hpss/tools/convert743/hpss/hpss_convertdb743

   HPSS_DB_INSTANCE_OWNER is set - Using 'db2hpss' for database connections.
   Password for db2hpss:
   cnvt> db cfg
   Using existing table 'CONVERT_743' to track conversion progress.

   Table Name      Conversion State Table/Index Space
   --------------- --------------- -------------------------
   SERVERINTERFACES CLEANUP         USERSPACE1/USERSPACE1
   ```

```
GLOBAL           CLEANUP         USERSPACE1/USERSPACE1
CORE             CLEANUP         USERSPACE1/USERSPACE1
MIGPOL           CLEANUP         USERSPACE1/USERSPACE1

CFG> cleanup all
Cleaning up for SERVERINTERFACES...
Executing operations for state: CLEANUP
Done
Cleaning up for GLOBAL...
Executing operations for state: CLEANUP
Done
Cleaning up for CORE...
Executing operations for state: CLEANUP
Done
Cleaning up for MIGPOL...
Executing operations for state: CLEANUP
Done

CFG> state

Table Name      Conversion State Table/Index Space
--------------- --------------- -------------------------
SERVERINTERFACES DONE            USERSPACE1/USERSPACE1
GLOBAL           DONE            USERSPACE1/USERSPACE1
CORE             DONE            USERSPACE1/USERSPACE1
MIGPOL           DONE            USERSPACE1/USERSPACE1
```

Switch to SUBSYS database(s) and resume cleanup process

```
CFG> db subsys1
Using existing table 'CONVERT_743' to track conversion progress.

Table Name      Conversion State Table/Index Space
--------------- --------------- -------------------------
BITFILE         CLEANUP          BITFILE/BITFILEIX
NSOBJECT        CLEANUP         NSOBJECT/NSOBJECTIX
SSPVHISTORY     CLEANUP          SSPVTAPE/SSPVTAPEIX
NSTRASH         CLEANUP          NSTRASH/NSTRASHIX
MPSFORCEMIGRATE CLEANUP          MPSFORCEMIGRATE/MPSFORCEMIGRATEIX

SUBSYS1> cleanup all

Cleaning up for BITFILE...
Executing operations for state: CLEANUP
Done
Cleaning up for NSOBJECT...
Executing operations for state: CLEANUP
Done
Cleaning up for SSPVHISTORY...
Executing operations for state: CLEANUP
Done
Cleaning up for NSTRASH...
Executing operations for state: CLEANUP
Done
Cleaning up for MPSFORCEMIGRATE...
Executing operations for state: CLEANUP
Done

SUBSYS1> state
```

```
Table Name      Conversion State Table/Index Space
--------------- --------------- -------------------------
BITFILE         DONE             BITFILE/BITFILEIX
NSOBJECT        DONE             NSOBJECT/NSOBJECTIX
SSPVHISTORY     DONE             SSPVTAPE/SSPVTAPEIX
NSTRASH         DONE             NSTRASH/NSTRASHIX
MPSFORCEMIGRATE DONE             MPSFORCEMIGRATE/MPSFORCEMIGRATEIX

SUBSYS1> quit

Cleaning up table used to track conversion... Done
Finished converting this database. Closing database connection.
```

It is NOT recommended at this stage of the conversion to open another connection to either CFG or SUBSYS using `cnvt> db cfg or subsys(n)`: this will create a new conversion table "hpss_convert743" and might set some tables in undesirable state. Instead, issue a quit.

```
You are DONE!
```

Chapter 10. HPSS 7.4.3 Conversion

This section summarizes the change introduced in HPSS 7.4.3 which require a metadata conversion. Refer to the HPSS 7.4.3 to the *HPSS Installation Guide* for more information.

10.1. General Information

The conversion from 7.4.3 patch 1 to all subsequent 7.4.3 patches require a conversion to grant users HPSS and HPSSSRVR authorization to perform operations on the MPSFORCEMIGRATE table. If you currently have HPSS 7.4.3 patch 1 that was installed from scratch and plan to upgrade to a later 7.4.3 patch, this conversion will be needed. Once the conversion has been applied you will not need to run this again for further HPSS 7.4.3 patch upgrades. If you currently have HPSS 7.4.3 patch 1 by upgrading from HPSS 7.4.2 version then you will not need this conversion. The 7.4.2 to 7.4.3 upgrades includes the conversion described in Chapter 9 which cover the grants needed for the MpsForceMigrate table. Customers who plan to install fresh new systems with HPSS 7.4.3 patch 2 and beyond will not need the conversion because grant statements for MpsForceMigrate will be applied through mkhpss(Bug 4682). Verify that your system needs additional permissions to the MpsForceMigrate table by running:

```
$ db2 connect to hsubsys1
$ db2 "select * from syscat.tabauth where tabschema = 'HPSS' and tabname =
'MPSFORCEMIGRATE'"
```

Your system needs to have the minimum following permissions; otherwise, you will need to run the conversion described in this chapter:

```
GRANTOR GRANTORTYPE GRANTEE GRANTEETYPE TABSCHEMA TABNAME CONTROLAUTH ALTERAUTH
   DELETEAUTH INDEXAUTH INSERTAUTH REFAUTH SELECTAUTH UPDATEAUTH

HPSSDB  U  HPSSSRVR  G  HPSS  MPSFORCEMIGRATE  N  N  Y  N  Y  N  Y  Y
HPSSDB  U  HPSS      U  HPSS  MPSFORCEMIGRATE  N  Y  Y  Y  Y  Y  Y  Y
```

- Addressing Bug 5020 - 743p1 to subsequent 743 patches require additional grants for MPSFORCEMIGRATE table.

10.2. Metadata Changes

This section provides additional information about the HPSS metadata change in Release 7.4.3.

10.2.1. Metadata Change to Subsystems (SUBSYS) Databases

The following table lists DB2 metadata changes in version 7.4.3 for the subsystem(s) databases (SUBSYS).

This addresses bug 5020 - 743p1 to subsequent 743 patches require additional grants for MPSForceMigrate table.

Table	Change
MPSFORCEMIGRATE	Granting authorization for HPSSSRVR and HPSS to perform operations on the MPSFORCEMIGRATE table

10.3. Conversion Overview

This section provides instructions for upgrading HPSS 7.4.3 patch 1 DB2 metadata to release 7.4.3. To ensure that the conversion is successful, all steps must be followed in the order specified in the documentation. Prior to starting the conversion process, consult the HPSS 7.4.3 Release Notes for any special instructions.

10.3.1. Software Prerequisites

The HPSS 7.4.3 prerequisite software is documented in the *HPSS 7.4.3 Installation Guide*. Depending on your current HPSS configuration and prerequisite software levels, there may be software components which must be upgraded prior to running the 7.4.3 metadata conversion. Follow the upgrade instructions provided by the prerequisite software to bring any software up to the level required by HPSS 7.4.3. Consult with your support representative for additional guidance.

The conversion process itself uses the Perl Database Interface (DBI) to interact with DB2. DBI provides a uniform interface to a number of Database engines, including DB2. DBI is available as part of the operating system on all supported Linux distributions. For sites running DB2 on AIX, a precompiled DBI RPM is available through your support representative. Alternatively, DBI in source form is available for download from http://www.cpan.org.

Because of licensing restrictions, the Database Driver (DBD) needed to interact with DB2 is not included in standard Linux distributions. The DB2 DBD was developed by and is copyrighted to IBM. Precompiled RPMs for AIX and LINUX are available from your support representative. DB2 DBD sources are also available for download from http://www.cpan.org.

If you have recently upgraded DB2, check to make sure DBI/DBD is compatible to the version of DB2 that you are using. You may need to recompile or download newer versions of DBI/DBD against the upgraded DB2 version. A mismatch in compatibility will result in hpss_convertdb743 not working with errors such as:

```
 'perl: symbol lookup error: /opt/ibm/db2/V9.7/lib64/libdb2.so.1: undefined
symbol: _Z19ossOpenInstanceListPcPPvb'

or
'install_driver(DB2) failed: Can't locate DBD/DB2.pm in @INC... Perhaps the
DBD::DB2 perl module hasn't been fully installed, or perhaps the
capitalisation of 'DB2' isn't right.'
```

On Linux version 2.6.32-431.el6.x86_64, the 7.4.3 conversion process was developed and tested with DBD version 1.85, DBI version 1.609, DB2 version 10.5.0.5 Fix Pack 5.

On AIX version 6100-09-03-1415, the 7.4.3 conversion process was developed and tested with perl-DBD-DB2-1-1, DBI version 1.611-2, DB2 version 10.5.0.5 Fix Pack 5.

10.3.2. Conversion Planning

Careful planning for conversion is important in order to ensure a successful conversion. This section describes the planning steps that need to be completed before running conversion.

Perform an online or offline backup of global (CFG) database.

Shutdown down HPSS, but keep DB2 up and running; otherwise, start DB2 using *db2start*. Put 7.4.3 code in place, then perform conversion.

10.3.3. Tool Used by the Conversion Process

Conversion uses the **hpss_convertdb743** tool to perform from the actual metadata conversion. This section provides information and usage options for this tool.

The first tool **hpss_convertdb743** is only used for the conversion process, and is not installed in the standard `bin` location. Instead, it is located in `HPSS_INSTALL_PATH/tools/convert743/hpss` beneath the root of the HPSS installation. **hpss_convertdb743** must be run as the root user, and requires the user to enter the password for the HPSS instance owner (typically hpssdb. db2hpss in this example)) in order to execute. Database operations are executed as the instance owner (example "db2hpss"). This is required in order to set the proper permissions on database table created during the conversion process.

```
# /opt/hpss/tools/convert743/hpss/hpss_convertdb743

HPSS_DB_INSTANCE_OWNER is set - Using 'db2hpss' for database connections.
Password for db2hpss:
cnvt>
```

hpss_convertdb743 recognizes the following commands:

```
cnvt> help

Available commands:
   cleanup              Cleanup after conversion
   close                Disconnect from a database
   convert              Run the conversion
   count                Count the number of rows in a table
   db                   Connect to database
   env                  Dump configuration parameters
   help                 Display available commands
   mpsforcemigrategrant Perform grants on MPSFORCEMIGRATE table
   pam_hpsssrv          Perform unix authentication, call PAM setup and modify authz.con
   revert               Revert the conversion
   rsql                 Show the SQL that will be run to revert a table
   showts               Show table space info for tables that will be converted
   sql                  Show the SQL that will be run to convert a table
   state                Show the current state of conversion
   ts                   Set the table space to use
cnvt>
```

Additional description of the **hpss_convertdb743** commands are below.

cleanup <all | table_name> - After a successful conversion, delete the backup copy of **table_name**. The operation requires a database connection. Once run, the conversion cannot be reverted.

close - close a database connection.

count - <all | table_name> - Count the number of rows in **table_name** specific to the site. This operation requires a database connection.

convert <all | table_name> - Convert the indicated table. The operation requires a database connection. If **table_name** isn't specified, then convert all tables in the connected database.

db <db_name> - Build a database connection to **db_name**. Note that **hpss_convertdb** expects the database alias name (CFG) to be provided, rather than the database name. This is so it can verify the name of subsystem databases in the global metadata.

env - Display database information and configuration parameters.

mpsforcemigrategrant - Connect to subsys1 database, grant delete, insert, select, update to group HPSSSRVR to MPSFORCEMIGRATE, and grant alter, delete, index, insert, select, update, references to user HPSS on table MPSFORCEMIGRATE.

pam_hpsssrv - Add user HPSS to HPSSSRV group file, run PAM setup, and edit to libhpssunixauth.so to libhpssunixauthz.so in the authz.conf file.

revert <all | table_name> - Revert the specified table. The operation requires a database connection. This option is meant to be used only by HPSS support since there may be additional steps needed to completely revert to the previous HPSS level.

rsql <table_name> - Show the SQL that will be used to revert the specified table. This requires a database connection.

showts - Show the table spaces that will be used for the conversion. The operation requires a database connection. Before a table can be converted, it must have a table space associated with it to hold the table and indexes.

sql <table_name> - Show the SQL that will be used to convert the specified table. This operation requires a database connection.

state - Show the conversion state for the table. This requires a database connection. Each table will be in one of the following states: START, CREATE, FILL, DEPLOY, CLEANUP or DONE.

ts <table_name> - Set the table spaces that will be used for table **table_name**. This requires a database connection.

10.4. Conversion - Detailed Instructions

This section provides detailed instructions for executing the 7.4.3 conversion. If you have questions about the conversion process, contact your support representative before attempting the conversion. This section assumes that the prerequisite steps have been completed, the ASLR feature has been turned off on Linux systems, the HPSS 7.4.3 software has been installed, and you are ready to use **hpss_convertdb743** to run the conversion.

The objective of this conversion is to address bug 5020 - 743p1 to subsequent 743 patches require additional grants for MpsForceMigrate table.

The conversion process is to use hpss_convertdb743 to *grant* several privileges to group *hpsssrvr* and to user *hpss* for sites converting from 7.4.3 patch 1 to subsequent 7.4.3 patches.

Become root

```
% su -
# source ~db2hpss/sqllib/db2profile  (db2hpss is db2 instance in this example)

#/opt/hpss/tools/convert743/hpss/hpss_convertdb743

HPSS_DB_INSTANCE_OWNER is set - Using 'db2hpss' for database connections.
Password for db2hpss: <enter [instance_owner] password>

type 'mpsforcemigrategrant' at cnvt> prompt:

convt> mpsforcemigrategrant

DB20000I  The SQL command completed successfully
DB20000I  The SQL command completed successfully

cnvt> quit
```

You are DONE!

Chapter 11. HPSS 7.4.3 Conversion

This section summarizes the change introduced in HPSS 7.4.3 which require a metadata conversion. Refer to the HPSS 7.4.3 to the *HPSS Installation Guide* for more information.

11.1. General Information

The conversion from 7.4.3 patch 2 to all subsequent 7.4.3 patches require a conversion to Modify MIGPOL and NSOBJECT tables. If you currently have HPSS 7.4.3 patch 2 that was installed from scratch and plan to upgrade to a later 7.4.3 patch, This conversion will be needed. Once this conversion has been applied you will not need to run this again for further HPSS 7.4.3 patch upgrades.

- Addressing Bug 5342 - 'Migration issues after upgrade to 742p1 (CR194, Part2) requires metadata change affecting MIGPOL table.
- Addressing Bug 5123: HPSS has a 64K limit on subdirectories within a directory affecting NSOBJECT table.

11.2. Metadata Changes

This section provides additional information about the HPSS metadata change in Release 7.4.3.

11.2.1. Metadata Change to Subsystems (SUBSYS) Databases

The following table lists DB2 metadata changes in version 7.4.3 for the cfg database (CFG).

Addressing Bug 5342 - Migration issues after upgrade to 742p1 (CR194, Part2) requires metadata change affecting MIGPOL table.

Table	Change
MIGPOL	Add new column min_minutes_disk_migr_io to MIGPOL table

The following table lists DB2 metadata changes in version 7.4.3 for the subsystem(s) databases (SUBSYS). Addressing Bug 5123 - HPSS has a 64K limit on subdirectories within a directory affecting NSOBJECT table.

Table	Change
NSOBJECT	ALTER table hpss.nsobject ALTER COLUMN link_count SET DATA TYPE BIGINT

11.3. Conversion Overview

This section provides instructions for upgrading HPSS 7.4.3 Patch 2 DB2 metadata to later 7.4.3 patch releases. To ensure that the conversion is successful, all steps must be followed in the order specified in the documentation. Prior to starting the conversion process, consult the HPSS 7.4.3 Release Notes for any special instructions.

11.3.1. Software Prerequisites

The HPSS 7.4.3 prerequisite software is documented in the *HPSS 7.4.3 Installation Guide*. Depending on your current HPSS configuration and prerequisite software levels, there may be software components which must be upgraded prior to running the 7.4.3 metadata conversion. Follow the upgrade instructions provided by the prerequisite software to bring any software up to the level required by HPSS 7.4.3. Consult with your support representative for additional guidance.

The conversion process itself uses the Perl Database Interface (DBI) to interact with DB2. DBI provides a uniform interface to a number of Database engines, including DB2. DBI is available as part of the operating system on all supported Linux distributions. For sites running DB2 on AIX, a precompiled DBI RPM is available through your support representative. Alternatively, DBI in source form is available for download from http://www.cpan.org.

Because of licensing restrictions, the Database Driver (DBD) needed to interact with DB2 is not included in standard Linux distributions. The DB2 DBD was developed by and is copyrighted to IBM. Precompiled RPMs for AIX and LINUX are available from your support representative. DB2 DBD sources are also available for download from http://www.cpan.org.

If you have recently upgraded DB2, check to make sure DBI/DBD is compatible to the version of DB2 that you are using. You may need to recompile or download newer versions of DBI/DBD against the upgraded DB2 version. A mismatch in compatibility will result in hpss_convertdb743 not working with errors such as:

```
 'perl: symbol lookup error: /opt/ibm/db2/V9.7/lib64/libdb2.so.1: undefined symbol: _Z19

or

'install_driver(DB2) failed: Can't locate DBD/DB2.pm in @INC... Perhaps the DBD::DB2 per
```

On Linux version 2.6.32-358.el6.x86_64, the 7.4.3 conversion process was developed and tested with DBD version 1.84, DBI version 1.609, DB2 version 9.7.0.9 Fix Pack 9a and 10.5.0.3 Fix Pack 3a.

On AIX version 4.2.3.4, the 7.4.3 conversion process was developed and tested with perl-DBD-DB2-1-1, DBI version 1.611-2, DB2 version 9.7.0.9 Fix Pack 9a and 10.5.0.3 Fix Pack 3a.

11.3.2. Conversion Planning

Careful planning for conversion is important in order to ensure a successful conversion. This section describes the planning steps that need to be completed before running conversion.

Perform an online or offline backup of global (CFG) database and subsystem(s) (SUBSYSx) databases.

Shutdown down HPSS, but keep DB2 up and running; otherwise, start DB2 using *db2start*. Put 7.4.3 code in place, then perform conversion.

11.3.3. Tools Used by the Conversion Process

Conversion uses **hpss_convertdb743** tool to perform the actual metadata conversions. This section provides information and usage options for this tool.

The **hpss_convertdb743** is only used for the conversion process, and is not installed in the standard `bin` location. Instead, it is located in `HPSS_INSTALL_PATH/tools/convert743/hpss` beneath the root of the HPSS installation. **hpss_convertdb743** must be run as the root user, and requires the user to enter the password for the HPSS instance owner (typically hpssdb. db2hpss in this example) in order to execute. Database operations are executed as the instance owner (typically hpssdb. "db2hpss" in this example). This is required in order to set the proper permissions on database table created during the conversion process.

```
# /opt/hpss/tools/convert743/hpss/hpss_convertdb743

HPSS_DB_INSTANCE_OWNER is set - Using 'db2hpss' for database connections.
Password for db2hpss:
cnvt>
```

hpss_convertdb743 recognizes the following commands:

```
cnvt> help

Available commands:
   cleanup              Cleanup after conversion
   close                Disconnect from a database
   convert              Run the conversion
   count                Count the number of rows in a table
   db                   Connect to database
   env                  Dump configuration parameters
   help                 Display available commands
   pam_hpsssrv          Perform unix authentication, call PAM setup, and modify authz.con
   revert               Revert the conversion
   rsql                 Show the SQL that will be run to revert a table
   showts               Show table space info for tables that will be converted
   sql                  Show the SQL that will be run to convert a table
   state                Show the current state of conversion
   ts                   Set the table space to use
cnvt>
```

Additional description of the **hpss_convertdb743** commands are below.

cleanup <all | table_name> - After a successful conversion, delete the backup copy of **table_name**. The operation requires a database connection. Once run, the conversion cannot be reverted.

close - close a database connection.

count - <all | table_name> - Count the number of rows in **table_name** specific to the site. This operation requires a database connection.

convert <all | table_name> - Convert the indicated table. The operation requires a database connection. If **table_name** isn't specified, then convert all tables in the connected database.

db <db_name> - Build a database connection to **db_name**. Note that **hpss_convertdb** expects the database alias name (CFG) to be provided, rather than the database name. This is so it can verify the name of subsystem databases in the global metadata.

env - Display database information and configuration parameters.

revert <all | table_name> - Revert the specified table. The operation requires a database connection. This option is meant to be used only by HPSS support since there may be additional steps needed to completely revert to the previous HPSS level.

rsql <table_name> - Show the SQL that will be used to revert the specified table. This requires a database connection.

showts - Show the table spaces that will be used for the conversion. The operation requires a database connection. Before a table can be converted, it must have a table space associated with it to hold the table and indexes.

sql <table_name> - Show the SQL that will be used to convert the specified table. This operation requires a database connection.

state - Show the conversion state for the table. This requires a database connection. Each table will be in one of the following states: START, CREATE, FILL, DEPLOY, CLEANUP or DONE.

ts <table_name> - Set the table spaces that will be used for table **table_name**. This requires a database connection.

The first time **hpss_convertdb743** connects to a database, it creates a table to track the conversion process. For the 7.4.3 conversion, the table is called CONVERT_743. The table contains a row for each of the HPSS tables that will be converted. As a table is being worked on, the state maintained in the row will change to reflect the conversion’s progress. It is important not to delete or modify this tracking table. Doing so will cause the conversion process to fail and may leave the metadata in an inconsistent state.

Conversion - Detailed Instructions

This section provides detailed instructions for executing the 7.4.3 conversion. If you have questions about the conversion process, contact your support representative before attempting the conversion. This section assumes that the prerequisite steps have been completed, the ASLR feature has been turned off on Linux systems, the HPSS 7.4.3 software has been installed, and you are ready to use **hpss_convertdb743** to run the conversion.

11.3.4. Convert the Global Database (CFG)

This step in the conversion process is to use **hpss_convertdb743** to convert the global (CFG) database MIGPOL table.

1. Become root

```
% su -
# source ~db2hpss/sqllib/db2profile  (db2hpss is db2 instance in this example)
```

2. Use **hpss_convertdb743** to convert the CFG database. The commands needed to convert the table in the CFG database are shown next, followed by a transcript from a run of the commands.

   ```
   #/opt/hpss/tools/convert743/hpss/hpss_convertdb743

   HPSS_DB_INSTANCE_OWNER is set - Using "db2hpss" for database connections.
   Password for db2hpss:     <enter your instance password here>
   cnvt> db cfg
   Creating table 'CONVERT_743' to track conversion progress...
   Making entry for table SERVERINTERFACES
   Making entry for table GLOBAL
   Table GLOBAL has already been converted
   Making entry for table CORE
   Table CORE has already been converted
   Making entry for table MIGPOL

   Table Name      Conversion State Table/Index Space
   --------------- --------------- ------------------------
   SERVERINTERFACES START          USERSPACE1/USERSPACE1
   GLOBAL           DONE           USERSPACE1/USERSPACE1
   CORE             DONE           USERSPACE1/USERSPACE1
   MIGPOL           START               USERSPACE1/USERSPACE1
   ```

CAUTION: Ignore all listed tables and issue conversion only for MIGPOL table required for this patch level.

```
CFG> convert migpol

Converting MIGPOL...
Executing operations for state: START
Executing operations for state: CREATE
Executing operations for state: FILL
Executing operations for state: DEPLOY
Conversion Time: 0 minutes 1 seconds
Done
```

If you need to revert to the original table(s) is to right stage to issue the following command most importantly before issuing **cleanup <tablename>**. Make sure you are in *cleanup* state of conversion by issuing **state** command.

```
CFG> state
Table Name      Conversion State Table/Index Space
--------------- --------------- ------------------------
SERVERINTERFACES START        USERSPACE1/USERSPACE1
GLOBAL           DONE         USERSPACE1/USERSPACE1
CORE             DONE         USERSPACE1/USERSPACE1
MIGPOL           CLEANUP      USERSPACE1/USERSPACE1

CFG> revert migpol

Reverting MIGPOL...
Running revert operations
DB20000I  The REORG command completed successfully.
Done
```

This action put back the "CFG" database target table(s) in the state of "Deploy" as shown below and conversion can be resumed from this point by issuing the command **convert <tablename>** at this stage if need to be.

```
CFG> state
Table Name      Conversion State Table/Index Space
--------------- --------------- ------------------------
SERVERINTERFACES START          USERSPACE1/USERSPACE1
GLOBAL           DONE           USERSPACE1/USERSPACE1
CORE             DONE           USERSPACE1/USERSPACE1
MIGPOL           DEPLOY         USERSPACE1/USERSPACE1
```

Otherwise after CFG database conversion is completed, you can proceed to SUBSYS databases conversion.

11.3.5. Convert the SUBSYS Databases

This step in the conversion process is to also use **hpss_convertdb743** to convert the SUBSYS database(s) NSOBJECT table.

11.3.5.1. Use hpss_convertdb743 to Convert the SUBSYS Database(s) NSOBJECT Table

The commands needed to convert the table in the SUBSYS databases are shown next, followed by a transcript from a run of the commands.

```
CFG> db subsys1

Creating table 'CONVERT_743' to track conversion progress...
Making entry for table BITFILE
Making entry for table NSOBJECT
Making entry for table SSPVHISTORY
Making entry for table NSTRASH
Table NSTRASH has already been converted
Making entry for table MPSFORCEMIGRATE
Table MPSFORCEMIGRATE has already been converted

Table Name      Conversion State Table/Index Space
--------------- --------------- ------------------------
BITFILE         START           BITFILE/BITFILEIX
NSOBJECT        START           NSOBJECT/NSOBJECTIX
SSPVHISTORY     START           SSPVTAPE/SSPVTAPEIX
NSTRASH         DONE            NSTRASH/NSTRASHIX
MPSFORCEMIGRATE DONE            MPSFORCEMIGRATE/MPSFORCEMIGRATEIX
```

CAUTION:Ignore all listed tables and issue conversion only for NSOBJECT table required for this patch level.

```
SUBSYS1> convert nsobject

Converting NSOBJECT...
        Executing operations for state: START
        Executing operations for state: CREATE
        Executing operations for state: FILL
        Executing operations for state: DEPLOY
        DB20000I  The REORG command completed successfully.
        SQL0598W  Existing index "HPSS.NSOBJECT_PKEY" is used as the index for the
        primary key or a unique key.  SQLSTATE=01550
        Conversion Time: 0 minutes 0 seconds
        Done
```

```
SUBSYS1> state

Table Name      Conversion State Table/Index Space
--------------- --------------- ------------------------
BITFILE         START          BITFILE/BITFILEIX
NSOBJECT        CLEANUP        NSOBJECT/NSOBJECTIX
SSPVHISTORY     START          SSPVTAPE/SSPVTAPEIX
NSTRASH         DONE           NSTRASH/NSTRASHIX
MPSFORCEMIGRATE DONE           MPSFORCEMIGRATE/MPSFORCEMIGRATEIX
```

If you need to revert to the original table(s) , this is to right stage to issue the following command most importantly before issuing *cleanup <tablename>*. Make sure you are in *cleanup* state of conversion by issuing *state* command.

```
SUBSYS1> revert nsobject

Reverting NSOBJECT...
Running revert operations
Done
```

This action put back the "SUBSYS" databases NSOBJECT target table in the state of "Deploy" as shown below and conversion can be resumed from this point by issuing the command **convert <tablename>** at this stage if need to be.

```
SUBSYS1> state

Table Name      Conversion State Table/Index Space
--------------- --------------- ------------------------
BITFILE         START         BITFILE/BITFILEIX
NSOBJECT        DEPLOY        NSOBJECT/NSOBJECTIX
SSPVHISTORY     START         SSPVTAPE/SSPVTAPEIX
NSTRASH         DONE          NSTRASH/NSTRASHIX
MPSFORCEMIGRATE DONE          MPSFORCEMIGRATE/MPSFORCEMIGRATEIX
```

Post-Conversion Detailed Instructions

This section describes steps that need to be taken after the metadata conversion is completed.

11.3.6. Run Full Database Backups

A full backup of global CFG and SUBSYS(s) databases after conversion is desirable.

11.3.6.1. Cleanup

Once satisfied with conversion to the 7.4.3 patch level system, the renamed pre-conversion affected tables are dropped. After this step is completed, the conversion **CANNOT** be reverted.

11.3.6.2. Cleanup the Global CFG and SUBSYS(s) Databases

When **cleanup <tablename>** is run, tracking tables flags are updated and renamed pre-conversion affected tables are dropped.

```
SUBSYS> db cfg
```

```
Using existing table 'CONVERT_743' to track conversion progress.

Table Name      Conversion State Table/Index Space
--------------- --------------- ------------------------
SERVERINTERFACES START        USERSPACE1/USERSPACE1
GLOBAL           DONE         USERSPACE1/USERSPACE1
CORE             DONE         USERSPACE1/USERSPACE1
MIGPOL           CLEANUP      USERSPACE1/USERSPACE1

CFG> cleanup migpol

Cleaning up for MIGPOL...
Executing operations for state: CLEANUP
Done

CFG> state

Table Name      Conversion State Table/Index Space
--------------- --------------- ------------------------
SERVERINTERFACES START            USERSPACE1/USERSPACE1
GLOBAL           DONE            USERSPACE1/USERSPACE1
CORE             DONE            USERSPACE1/USERSPACE1
MIGPOL           DONE            USERSPACE1/USERSPACE1
```

Switch to SUBSYS database(s) and resume cleanup process.

```
CFG> db subsys1
Using existing table 'CONVERT_743' to track conversion progress.

Table Name      Conversion State Table/Index Space
--------------- --------------- ------------------------
BITFILE         START         BITFILE/BITFILEIX
NSOBJECT        CLEANUP       NSOBJECT/NSOBJECTIX
SSPVHISTORY     STAR          SSPVTAPE/SSPVTAPEIX
NSTRASH         DONE          NSTRASH/NSTRASHIX
MPSFORCEMIGRATE DONE          MPSFORCEMIGRATE/MPSFORCEMIGRATEIX

SUBSYS1> cleanup nsobject

Cleaning up for NSOBJECT...
Executing operations for state: CLEANUP
Done

SUBSYS1> state

Table Name      Conversion State Table/Index Space
--------------- --------------- ------------------------
BITFILE         START             BITFILE/BITFILEIX
NSOBJECT        DONE              NSOBJECT/NSOBJECTIX
SSPVHISTORY     START             SSPVTAPE/SSPVTAPEIX
NSTRASH         DONE              NSTRASH/NSTRASHIX
MPSFORCEMIGRATE DONE              MPSFORCEMIGRATE/MPSFORCEMIGRATEIX

SUBSYS1> quit
```

Chapter 12. Endian Conversion

This chapter summarizes HPSS metadata conversion from big-endian to little-endian. HPSS currently does not support little-endian to big-endian conversion.

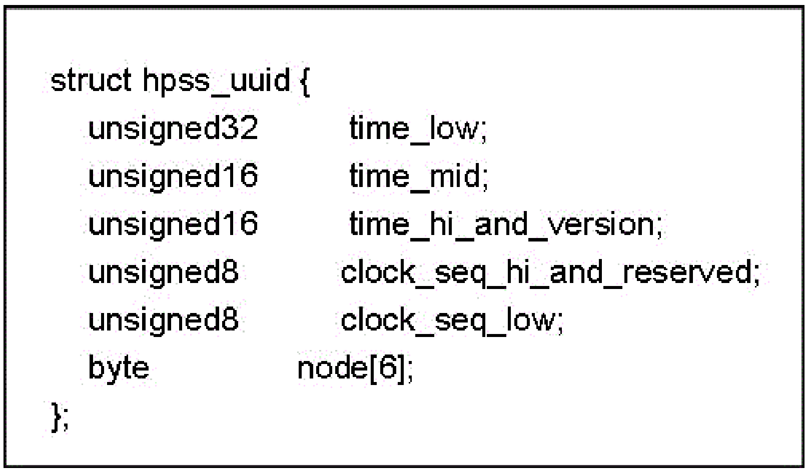
12.1. General Information

The conversion from big-endian to little-endian manipulates a number of metadata field which store data in a endian-dependent manner. The process is automated through the use of provided conversion scripts. In general, two types of fields must be converted, UUIDs and SOIDs:

12.1.1. UUIDs

For UUID types the following byte order swapping is necessary:

- Bytes 1-4 are reversed (4-3-2-1)
- Bytes 5-6 are reversed (6-5)
- Bytes 7-8 are reversed (8-7)
- Bytes 9-16 are not altered



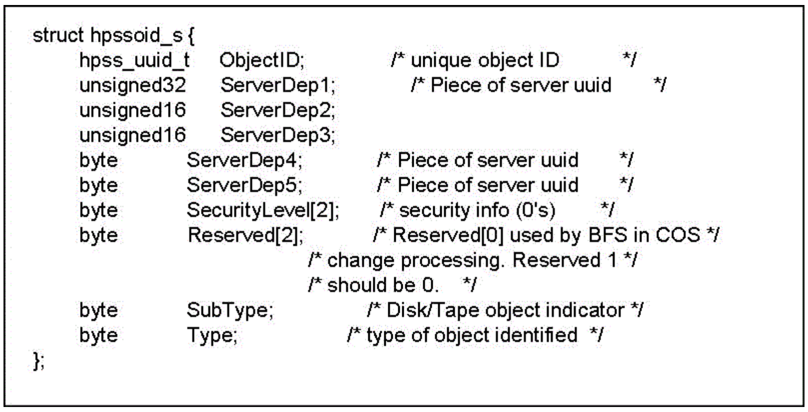
```
struct hpss_uuid {
    unsigned32      time_low;
    unsigned16      time_mid;
    unsigned16      time_hi_and_version;
    unsigned8       clock_seq_hi_and_reserved;
    unsigned8       clock_seq_low;
    byte            node[6];
};
```

12.1.2. SOIDs

For SOID types the following byte order swapping is necessary:

- Bytes 1-4 are reversed (4-3-2-1)
- Bytes 5-6 are reversed (6-5)
- Bytes 7-8 are reversed (8-7)
- Bytes 9-16 are not altered
- Bytes 17-20 are reversed (20-19-18-17)

- Bytes 21-22 are reversed (22-21)
- Bytes 23-24 are reversed (24-23)
- Bytes 25-32 are not altered



```
struct hpssoid_s {
      hpss_uuid_t    ObjectID;          /* unique object ID        */
      unsigned32     ServerDep1;          /* Piece of server uuid      */
      unsigned16     ServerDep2;
      unsigned16     ServerDep3;
      byte         ServerDep4;         /* Piece of server uuid      */
      byte         ServerDep5;         /* Piece of server uuid      */
      byte         SecurityLevel[2];     /* security info (0's)       */
      byte         Reserved[2];        /* Reserved[0] used by BFS in COS */
                              /* change processing. Reserved 1 */
                              /* should be 0.    */
      byte         SubType;            /* Disk/Tape object indicator */
      byte         Type;             /* type of object identified  */
};
```

12.1.3. Conversion Components

The conversion requires the execution of four scripts:

1. **hpss.hcfg.export.db2** - This script performs the necessary byte swaps for metadata stored in the HPSS configuration database. This script requires a connection to the source HPSS configuration database (big-endian).
2. **hpss.hsubsys.export.db2** - This script perform the necessary byte swaps for metadata stored in an HPSS subsystem database. It must be executed once for each subsystem database. This script requires a connection to the source HPSS subsystem database (big-endian).
3. **hpss.hcfg.load.db2** - This script loads the data generated by the **hpss.hcfg.export.db2** script. This script requires a connection to the target HPSS configuration database (little-endian).
4. **hpss.hsubsys.load.db2** - This script loads the data generated by the **hpss.hsubsys.export.db2** script. This script requires a connection to the target HPSS subsystem database (little-endian), and must be executed once for each subsystem.

12.2. Metadata Changes

This section describes what metadata will be changed in order to convert from big-endian to little-endian.

12.2.1. Metadata Change Detail

12.2.1.1. Configuration Tables

Table 12.1. Configuration Tables

Table Name	Change
AUTHZACL	Convert SERVER_ID, INTERFACE_ID.

Table Name	Change
CARTRIDGE	Convert PVR_ID.
CORE	Convert SERVER_ID.
DMG	Convert SERVER_ID.
GATEKEEPER	Convert GKID.
GLOBAL	Convert ROOT_CORE_SERVER_ID.
LOGCLIENT	Convert LOGC_SERVER_ID.
LOGDAEMON	Convert LOGD_SERVER_ID.
LSPOLICY	Convert HPSS_ID.
MOVERDEVICE	Convert MVR_ID.
MOVER	Convert MVR_ID.
MPS	Convert MPSID.
NFS	Convert NFS_ID.
NSGLOBALFILESET	Convert CORE_SERVER_UUID, GATEWAY_UUID.
PVLACTIVITY	Convert CLIENT.
PVLDRIVE	Convert MVR_ID, PVR.
PVLPV	Convert ALLOC_CLIENT_ID.
PVR	Convert PVR_ID.
SERVERINTERFACES	Convert INTERFACE_ID, SERVER_ID.
SERVER	Convert SERVER_ID.
SITE	Convert HPSS_ID.
STORAGESUBSYS	Convert GATEKEEPER_ID.

12.2.1.2. Subsystem Tables

Table 12.2. Subsystem Tables

Table Name	Change
BFCOSCHANGE	Convert BFID field.
BFDISKALLOCREC	Convert BFID and SSEG_ID fields.
BFMIGRREC	Convert BFID.
BFPURGEREC	Convert BFID.
BFSSEGCHKPT	Convert BFID, SSEG_ID.
BFTAPESEG	Convert BFID, SSEG_ID.
BITFILE	Convert BFID.
DISKSEGUNLINK	Convert SSEGID.
DMGFILESET	Convert MOUNT_PT_CORE_SERVER_ID, MOUNT_PT_HANDLE_CORE_UUID,

Table Name	Change
	RPC_ENDPOINT_INTERFACE_ID, SECURITY_SERVER_ID, SERVER_ID.
NSFILESETATTR	Convert FILESET_HANDLE_CORE_UUID.
NSOBJECT	Convert BITFILE_ID.
NSOBJECT	Convert SUB_TREE_HANDLE_CORE_UUID.
SSPVDISK	Convert VVID.
SSPVTAPE	Convert VVID.
STORAGEMAPDISK	Convert VVID.
STORAGEMAPTAPE	Convert CURRENT_SSID, VVID.
STORAGESEGAUX	Convert SSID.
STORAGESEGDISKEXTENTS	Convert SSID, VVID.
STORAGESEGDISK	Convert SSID, VVID.
STORAGESEGTAPEABSADDR	Convert SSID.
STORAGESEGTAPE	Convert SSID, VVID.
TAPESEGUNLINK	Convert SSEGID.
VVDISK	Convert VVID.
VVTAPEABSADDR	Convert VVID.
VVTAPE	Convert VVID.

12.3. Introduction

The following section will guide you through the process of converting the system from big-endian to little-endian. This process should only be attempted after discussions with IBM support. Though the process is fairly straightforward and automated, it is recommended that IBM support guide administrators through the process as mistakes can potentially be extremely dangerous and difficult to recover from.

12.3.1. Prepare Environment

- Log into the system as the HPSS DB2 instance owner.
- Create the following directory structure on the source system (big-endian) to hold the exported/converted metadata. This should be done on a file system with sufficient space. It will take approximately 1,200 bytes of storage per bitfile in the HPSS subsystem. This number will vary depending on HPSS configuration. Things like dual tape copies will require more space. This new directories should either be owned by the DB2 instance owner, or the instance owner should have sufficient privileges to create and write metadata export files.
  - `<path>/endian_conversion`
  - `<path>/endian_conversion/hcfg`

- For each subsystem:
  - `<path>/endian_conversion/<subsystem database name>`
- The target system will require a similar directory structure as was created on the source system.
- Follow the general guidelines for converting a system - shut down HPSS and take a full offline backup.

12.3.2. Export HCFG

- Execute the following statements:

```
% cd into <path>/endian_conversion/hcfg
% db2 connect to <HPSS configuration database name>
% db2 -tvf <path to export
scripts>/hpss.hcfg.export.db2 >
hpss.hcfg.export.db2.log
% db2 connect reset
```

Validate the number of rows exported matches the source.

12.3.3. Export Subsystems

- For each subsystem, execute the following statements:

```
% cd into <path>/endian_conversion/<subsystem database name>
% db2 connect to <subsystem database name>
 % db2 -tvf <path to export
scripts>/hpss.hsubsys.export.db2 >
hpss.hsubsys.export.db2.log
% db2 connect reset
```

Validate the number of rows exported matches the source table.

- Transfer .ixf files to the target system.
  - It is recommended that the ixf files be checksummed on source and destination machines to validate their integrity after the transfer.

12.3.4. Import HCFG

- Log in as the DB2 instance owner on the target system.
- Execute the following commands:

```
% cd into <path>/endian_conversion/hcfg
% db2 connect to <HPSS configuration database name>
% db2 -tvf <path to load scripts>/ hpss.hcfg.load.db2 >
hpss.hcfg.load.db2.log
% db2 connect reset
```

Validate the number of rows added to each table matches the source.

Perform an offline backup of the HPSS configuration database.

12.3.5. Import Subsystems

- For each subsystem, execute the following commands:

```
% cd <path>/endian_conversion/<HPSS subsystem database name>
% db2 connect to <HPSS subsystem database name>
% db2 -tvf <path to load scripts>/
hpss.hsubsys.load.db2 > hpss.hsubsys.load.db2.log
% db2 connect reset
```

Validate the number of rows added to each table matches the source.

Perform an offline backup of the HPSS subsystem database.

Chapter 13. Upgrading AIX, DB2, and Linux

13.1. Overview

This chapter provides guidelines and considerations to help the administrator complete software upgrades of AIX, DB2, and Linux to meet HPSS prerequisites.

13.2. Upgrading to DB2 v9.5

Note that although DB2 9.5 supports 32-bit operating systems, HPSS does not support 32-bit Core Servers beyond HPSS 7.3. 64-bit is required for 7.4 and beyond.

Minimum software requirements for DB2 v9.5 are:

AIX Version 5.3:

- 64-bit AIX kernel is required
- AIX 5.3 Technology Level (TL) 6 and Service Pack (SP) 2 plus APAR IZ03063
- Minimum C++ runtime level is xlC.rte 9.0.0.1 and xlC.aix50.rte 9.0.0.1. These files are included in the August 2007 IBM® C++ Runtime Environment Components for AIX package. AIX Version 6.12
- 64-bit AIX kernel is required
- Minimum C++ runtime level is xlC.rte 9.0.0.1 and xlC.aix61.rte 9.0.0.1. These files are included in the October 2007 IBM C++ Runtime Environment Components for AIX package.

Linux:

- Red Hat Enterprise Linux (RHEL) 4 Update 4
- Red Hat Enterprise Linux (RHEL) 5

Note: POWER requires a minimum of SLES 10 Service Pack 1 or RHEL 5

Minimum hardware requirements for DB2 v9.5 are:

AIX:

- 64-bit Common Hardware Reference Platform (CHRP) architecture
- All processors that are capable of running the supported AIX operating systems.

Linux:

- x86 (Intel Pentium®, Intel Xeon®, and AMD) 32-bit Intel and AMD processors
- x64 (64-bit AMD64 and Intel EM64T processors)
- POWER (pSeries)

To verify that it is a CHRP architecture system, issue the command **lscfg** and look for the following output: `Model Architecture: chrp`

In AIX 6.1 there are two types of Workload Partitions (WPARs): System WPARs and Application WPARs. DB2 installation is supported only on a System WPAR. AIX 6.1 also supports the ability to encrypt a JFS2 file system or set of files. This feature is not supported if you are using multiple partition instances.

To show kernel mode on AIX (32/64-bit), execute:

```
% bootinfo
```

To switch to a 64-bit kernel on AIX, execute:

```
% ln -sf /usr/lib/boot/unix_64 /unix
% ln -sf /usr/lib/boot/unix_64 /usr/lib/boot/unix
% bosboot -a
% shutdown -Fr
```

Once the software and hardware prerequisites are satisfied, the DB2 upgrade may begin. Note that sites upgrading from HPSS 6.2 will be upgrading from DB2 v8.2, while sites upgrading from HPSS 6.2.2 will be upgrading from DB2 v9.1. The first part of this section explains how to upgrade from v8.2, while the second part provides instructions for v9.1.

13.2.1. Upgrading DB2 from v8.2 or v8.1 Fix Pack 7(+)

Sites should familiarize themselves with the DB2 guide for performing this upgrade, the DB2 v9.5 Quick Beginnings Guide and DB2 v9.5 Migration Guide for Linux, Unix, and Windows. The document contains details not covered in this section. Some pre-migration requirements are:

On AIX:

- At least 100MB of free space in `/tmp`
- At least 1GB of free space in `/opt/IBM/db2/V9.5` (default install path)
- Java SDK 1.4.2 or later up through version 5.
- If Kerberos with DB2, then need NAS 1.4

On Linux:

- SWAP space twice the size of available memory
- 100MB of freespace in `/tmp`

- At least 1GB of free space in `/opt/ibm/db2/V9.5` (default install path)
- Java SDK 1.4.2 or later up through version 5.

The default installation location of DB2 v9.5 is `/opt/IBM/db2/V9.5` on AIX and `/opt/ibm/db2/V9.5` on Linux. Since multiple copies of DB2 are allowed on the same host now with DB2 v9, if the installation program detects DB2 in the default path it will automatically append _## to the default path, where ## is a monotonically increasing sequence number from 00-99, and install DB2 in this new location.

Before running any of the upgrade programs, unless otherwise specified, ensure all actions are performed as root user and without sourcing any existing DB2 profiles so that DB2 v9.5 has no knowledge of previous versions or instances of DB2 on the system. Otherwise problems will occur as DB2 v9.5 has a special process for allowing multiple instances of DB2 at different levels on the same system. HPSS sites should not attempt having multiple instances of DB2 at different levels on the same system.

Before upgrading, sites may want to use the provided db2_install program as an alternative to the db2setup steps documented below. Sites have reported that db2_install may be easier to use. See the DB2 v9.5 Migration Guide for details on this alternative and details on upgrading DB2 from a previous release to v9.5.

An English version of the guide can be found at:

ftp://ftp.software.ibm.com/ps/products/db2/info/vr95/pdf/en_US/db2mge951.pdf

Multi-language versions of the guide can be found at:

http://www-1.ibm.com/support/docview.wss?rs=71&uid=swg27009728

The DB2 Information Center also has good information on performing a migration to DB2 v9.5:

http://publib.boulder.ibm.com/infocenter/db2luw/v9r5/index.jsp?topic=/com.ibm.db2.luw.qb.migration.doc/doc/t0007200.html

Upgrading from v8 to v9.5 begins by installing DB2 v9.5 on the system by executing the db2setup Java based GUI included with the DB2 v9.5 software. This program should be used to install the DB2 v9.5 software on each DB2 v8 client and server for migration of all v8 DB2 clients and servers used with HPSS. Select Install a Product and then Install New under DB2 Enterprise Server Edition Version 9.5 to begin software installation. Select default settings on screens 1-5. On screen 6, select Do not install SA MP Base Component unless you run DB2 in a high availability environment. On screen 7, select existing user and enter your database administration server's (DAS) group (e.g. dasadm1) and user name (e.g. dasusr1) if applicable. If previously used **mkhpss** to install and configure DB2, the DAS will not be configured on the system and creating a new user is the only option. The DAS allows usage of the DB2 control center (db2cc) and is not essential to DB2 operation. On screen 8, select Do not create a DB2 instance as the HPSS instance already exists. On screen 9, select Do not configure automatic notification unless SMTP notifications of DB2 failures is necessary. Select Finish after reviewing the proposed installation options.

After installing the software, check to make sure each database is acceptable for migration by running:

```
% /opt/IBM/db2/V9.5/bin/db2ckmig <dbname> -l /tmp/ckmig.out
```

Where dbname is the database name of each HPSS database (e.g. HCFG, HSUBSYS1, HSUBSYS2, etc.). Any problems will need investigation and resolution through DB2 documentation or support. The program may hang after reporting db2ckmig was successful. Database(s) can be migrated and CTRL-C will exit properly. If this is the case, a temporary fix will be needed to successfully migrate the instance as it calls the db2ckmig program and will hang as well. A simple fix is to rename the program to something like `/opt/IBM/db2/V9.5/bin/db2ckmig.orig` and create a new script like such:

```
% cat /opt/IBM/db2/V9.5/bin/db2ckmig
#!/bin/ksh
exit 0
```

Ensure the db2ckmig script is executable by root.

After checking the databases for migration, start your previous version of DB2 (v8) if it isn't already running. Consider revoking PUBLIC group's permissions on HPSS databases with the db2undgp utility as this program will not longer work with DB2 v9.5. DB2 default installations grant permissions for a group called PUBLIC that sites don't need or want with their HPSS databases. For each HPSS database, execute:

```
% db2undgp -d <dbname> -r
```

Where dbname is the HPSS database name.

Take a full offline backup of each HPSS database at this point to allow for reversion if necessary. To complete the full offline backup, execute:

```
% db2 force applications all
% db2 backup database <dbname> to <outdir>
```

Check the validity of the backup images by invoking db2ckbkp with the backup file name as the input parameter. For example:

```
% db2ckbkp HSUBSYS1.0.hpssdb.NODE0000.CATN0000.20080709112544.001
```

If the image is valid, the final output should be:

```
Image Verification Complete - successful.
```

The following steps allow for restoration of the instance and databases if necessary and are optional but recommended. The following example is on a test system with 2 subsystems with databases HSUBSYS1 and HSUBSYS2 and a configuration database name HCFG.

Save database DDL:

```
% db2support <output-directory> -d <database-name> -cl 0
% db2support /tmp/hcfg -d hcfg -cl 0
% db2support /tmp/hsubsys1 -d hsubsys1 -cl 0
% db2support /tmp/hsubsys2 -d hsubsys2 -cl 0
```

Save package information:

```
% db2 connect to hcfg
% db2 list packages for schema hpss show detail > /tmp/hcfg/packages.out
% db2 terminate
% db2 connect to hsubsys1
% db2 list packages for schema hpss show detail >
/tmp/hsubsys1/packages.out
% db2 terminate
% db2 connect to hsubsys2
% db2 list packages for schema hpss show detail >
/tmp/hsubsys2/packages.out
% db2 terminate
```

If using DB2 auditing, backup audit configuration:

```
% db2audit describe > /tmp/db2audit.cfg.out
```

Backup any external routines that may be defined:

```
% cp -R $INSTHOME/sqllib/function $INSTHOME/routine_backup
```

Where $INSTHOME is your instance home directory (e.g. `/var/hpss/hpssdb`).

Save database manager and database configurations:

```
% db2 get dbm cfg > /tmp/db2.dbm.cfg.out
% db2 connect to hcfg; db2 get db cfg for hcfg show detail > \
/tmp/hcfg/db2.db.cfg.out
% db2 connect to hsubsys1; db2 get db cfg for hsubsys1 show detail > \
/tmp/hsubsys1/db2.db.cfg.out
% db2 connect to hsubsys2; db2 get db cfg for hsubsys2 show detail > \
/tmp/hsubsys2/db2.db.cfg.out
```

Save database information with db2look:

```
% db2look -d hcfg -e -o db2look.hcfg.out -l -x
% db2look -d hsubsys1 -e -o db2look.hsubsys1.out -l -x
% db2look -d hsubsys2 -e -o db2look.hsubsys2.out -l -x
```

Save DB2 registry settings:

```
% db2set -all > /tmp/db2set.out
```

At this point, enough information is retained to restore DB2 and the HPSS instance and databases to system with DB2 v8 software installed.

For most HPSS installations, databases already have adequate settings to support migration, but consider the following in preparation for migration. Consider increasing the number of primary logs available to each database (LOGPRIMARY in the database configuration settings). Consider increasing the size of the system temporary tablespaces for each database to allow for twice as much free pages as used pages. Consider increasing the size of the system catalog tablespace to have at least 50% free space prior to conversion.

On Linux, if sites are using raw devices for DB2 logging they must change to using block devices as this feature is no longer available with DB2 v9.5. See the DB2 Migration Guide for details on how to accomplish this.

The instance is now ready for migration to v9.5. To perform the instance migration for an HPSS instance called hpssdb perform the following as the instance owner/user:

```
% db2 force applications all
% db2stop
% slibclean
% ipclean
```

As root, execute:

```
% /opt/IBM/db2/V9.5/instance/db2imigr -u franky hpssdb
```

Where -u argument is the username of a user for DB2 fenced procedures. If you don't have one, just enter a user that won't do any useful work with DB2 (e.g. not root, or hpss or hpssdb).

To validate that the instance is at the 9.5 level as the instance owner execute:

```
% db2start
% db2level
```

If there is an existing DAS, now is the time to migrate it to v9.5. If migrating the DAS is problematic, sites may also simply drop the existing DAS and recreate a new one (see the DB2 Migration Guide). Migrating the DAS is optional and some sites may just want to skip to migrating the HPSS databases in the next step. To migrate the DAS:

As DAS owner (e.g. dasusr1):

```
% db2admin stop
```

As root:

```
% /opt/IBM/db2/V9.5/instance/dasmigr dasusr1
```

Now migrate each HPSS database. Prior to migrating the databases, rename the current DB2 diagnostic log to provide a clean separation between different versions of DB2.

```
% mv /var/hpss/hpssdb/sqllib/db2dump/db2diag.log \
/var/hpss/hpssdb/sqllib/db2dump/db2diag.log.db2v8
```

where `/var/hpss/hpssdb` is the path to the DB2 instance at your site.

To migrate each HPSS database as the instance owner (e.g. hpssdb):

```
% db2 migrate database <dbname> [ user xxxx using xxxx ]
```

where dbname is each HPSS database name and xxxx is the database user and password at your site if necessary for connection. For example:

```
% db2 migrate database hcfg
```

Now sites need to perform some post migration steps. Begin by activating each migrated database as the instance owner (hpssdb):

```
% db2 activate database <dbname>
```

Attempt to connect to each migrated database to ensure it's active and available:

```
% db2 connect to <dbname>
```

If auditing is enabled, grant the audit user the new SECADM authority on each database by connecting to it and issuing:

```
% db2 grant SECADM on database to user <user>
% db2 "select * from syscat.auditpolicies a where \
a.auditpolicyname=DB2AUDIT_CFG_MIGR"
```

Rebind any packages for each migrated database by executing the following as instance owner:

```
% db2rbind <dbname> -l /tmp/db2rbind.out
```

Where -l argument is the output file desired.

Congratulations, your instance and databases are now migrated to DB2 v9.5. Ensure you run the conversion program menu option Perform post-7.1 startup steps for the cfg db and Perform post-7.1 startup steps for a subsys db to make recommended modifications to the configuration to take advantage of new 9.5 features. Alternatively, sites can use the DB2 Migration Guide on post-migration steps to make configuration changes by hand.

13.2.2. Upgrading DB2 from v9.1

On AIX:

- At least 100MB of free space in `/tmp`
- At least 1GB of free space in `/opt/IBM/db2/V9.5` (default install path)
- Java SDK 1.4.2 or later up through version 5.
- If Kerberos with DB2, then need NAS 1.4

On Linux:

- SWAP space twice the size of available memory
- 100MB of freespace in `/tmp`
- At least 1GB of free space in `/opt/ibm/db2/V9.5` (default install path)
- Java SDK 1.4.2 or later up through version 5.

To begin, open a shell to the HPSS Core Server Machine. Shut down HPSS via hpssgui or hpssadm. Shut down the System Manager and Startup Daemon.

Verify the machine can run 64-bit applications as DB2 requires a 64-bit kernel.

Verify the kernel on AIX is running in 64-bit mode, as root execute:

```
% bootinfo K 64
```

If the machine is not 64-bit capable, DB2 V9 can not be installed!

If the kernel is not configured for 64-bit, perform the following steps:

```
% ln -sf /usr/lib/boot/unix_64 /unix
% ln -sf /usr/lib/boot/unix_64 /usr/lib/boot/unix
% bosboot -a
% shutdown -Fr
```

Save DB2 configuration information.

```
% . /var/hpss/hpssdb/sqllib/db2profile
% db2start
% cd /hpssdb_backups/db2info
% db2 get dbm cfg > dbm_cfg.v91
% db2 get db cfg for hcfg > hcfg_cfg.v91
% db2 get db cfg for hsubsys1 > hsubsys1_cfg.v91
% db2 get db cfg for hsubsys2 > hsubsys2_cfg.v91
% db2set -all > reg_hpssdb.v91
% set |grep DB2 > env_hpssdb.v91
```

Backup the databases.

```
% . /var/hpss/hpssdb/sqllib/db2profile
% db2 backup db hcfg to /hpssdb_backups/cfg

Backup successful. The timestamp for this backup image is :
YYYYMMDDhhmmss

% db2ckbkp HSUBSYS1.0.hpssdb.NODE0000.CATN0000.20081020162926.001
% db2 backup db hsubsys2 to /hpssdb_backups/subsys2

Backup successful. The timestamp for this backup image is :
YYYYMMDDhhmmss
```

Note: follow the above commands for backing up each subsystem database on the HPSS system.

Check the validity of the backup

```
% db2ckbkp HCFG.0.hpssdb.NODE0000.CATN0000.20081020162843.001
% db2 backup db hsubsys1 to /hpssdb_backups/subsys1

Backup successful. The timestamp for this backup image is :
YYYYMMDDhhmmss
```

Note: check the validity of each backup file generated.

Optional but recommended:

```
% db2support _output-directory_ -d _database-name_ -cl 0
% db2support /tmp/hcfg -d hcfg -cl 0
% db2support /tmp/hsubsys1 -d hsubsys1 -cl 0
% db2support /tmp/hsubsys2 -d hsubsys2 -cl 0
```

Save package information:

```
% db2 connect to hcfg
% db2 list packages for schema hpss show detail > /tmp/hcfg/packages.out
% db2 terminate
% db2 connect to hsubsys1
% db2 list packages for schema hpss show detail >
/tmp/hsubsys1/packages.out
% db2 terminate
% db2 connect to hsubsys2
% db2 list packages for schema hpss show detail >
/tmp/hsubsys2/packages.out
% db2 terminate
```

Ensure the db2profile is not sourced in the root profile or in the current environment. DB2 v9 (including 9.5) must not know of an existing DB2 installation to correctly upgrade the current instance and version of DB2.

```
% vi /.profile
```

Look for a line similar to the following:

```
. /var/hpss/hpssdb/sqllib/db2profile
```

Stop DB2.

```
% db2stop
```

Exit out of the shell system and open a new one. This will clear any db2 environmental variables.

Run **db2_install** from the `db2_91` directory:

```
% cd /scratch/db2_91_fixpk3/ese/disk1/
% db2_install
```

Default directory for installation of products is `/opt/IBM/db2/V9.5`. Input and output for using **db2_install** to perform the upgrade is provided below:

```
***********************************************************
Do you want to choose a different directory to install [yes/no] ?
no


Specify one or more of the following keywords,
separated by spaces, to install DB2 products.
  CLIENT
  RTCL
  ESE

Enter "help" to redisplay product names.
Enter "quit" to exit.
***********************************************************
ESE
DB2 installation is being initialized.

      Total number of tasks to be performed: 41
Total estimated time for all tasks to be performed: 1750

Task #1 start
Description: Checking license agreement acceptance
Estimated time 1 second(s)
Task #1 end

. . .

   Task #40 start
   Description: Initializing instance list
   Estimated time 5 second(s)
   Task #40 end

   Task #41 start
   Description: Updating global profile registry
   Estimated time 3 second(s)
   Task #41 end

   The execution completed successfully.

   For more information see the DB2 installation log at
"/tmp/db2_install.log.356414".
```

Set up DB2 license

```
% /opt/IBM/db2/V9.5/adm/db2licm -a /scratch/db2_95/db2ese_c.lic

LIC1402I License added successfully.
```

Run the pre-migration step on each HPSS database before migrating the instance.

```
% su - hpssdb
% /opt/IBM/db2/V9.5/instance/db2ckmig hcfg -l /tmp/db2ckmig.hcfg.log
db2ckmig was successful. Database(s) can be migrated.
% /opt/IBM/db2/V9.5/instance/db2ckmig hsubsys1 -l
/tmp/db2ckmig.hsubsys1.log
% /opt/IBM/db2/V9.5/instance/db2ckmig hsubsys2 -l
/tmp/db2ckmig.hsubsys2.log
```

Migrate the instance (as hpssdb):

```
% . /var/hpss/hpssdb/sqllib/db2profile
% db2stop
% slibclean
% ipclean
```

You must be root to run the instance migration:

```
% cd /opt/IBM/db2/V9.5/instance
% db2imigr -u hpssdb hpssdb

db2ckmig was successful. Database(s) can be migrated.

DBI1070I Program db2imigr completed successfully.
```

Verifying the upgrade:

For information, see:

http://publib.boulder.ibm.com/infocenter/db2luw/v9r5/index.jsp?topic=/
com.ibm.db2.luw.qb.migration.doc/doc/t0007200.html

Log on to the DB2 server as a user with sufficient authority to start your instance.

```
% db2start
10/21/2008 03:48:17 0 0 SQL1063N DB2START processing was successful.
SQL1063N DB2START processing was successful.

% db2level

DB21085I Instance "hpssdb" uses "64" bits and DB2 code release
"SQL09050" with level identifier "03010107".
Informational tokens are "DB2 v9.5.0.0", "s071001", "AIX6495", and Fix
Pack "0".
Product is installed at "/opt/IBM/db2/V9.5".
```

Prior to migrating the databases, rename the current DB2 diagnostic log to provide a clean separation between the different versions of DB2.

For information, see:

http://publib.boulder.ibm.com/infocenter/db2luw/v9r5/index.jsp?topic=/com.ibm.db2.luw.qb.migration.doc/doc/t0007200.html

```
% mv /var/hpss/hpssdb/sqllib/db2dump/db2diag.log \
/var/hpss/hpssdb/sqllib/db2dump/db2diag.log.db2v91
```

Migrate the databases. (su - hpssdb):

```
% db2 migrate database hcfg
DB20000I The MIGRATE DATABASE command completed successfully.

% db2 migrate database hsubsys1
DB20000I The MIGRATE DATABASE command completed successfully.
```

Note: repeat the database migration step for each HPSS subsystem database.

Verify your database migration is successful. Connect to the migrated databases and issue a small query:

```
% db2 connect to hcfg
% db2 "select * from syscat.dbauth"
% db2 disconnect current

% db2 connect to hsubsys1
% db2 "select * from syscat.dbauth"
% db2 disconnect current
% db2 terminate
```

Activate the databases:

```
% db2 activate database hcfg
DB20000I The ACTIVATE DATABASE command completed successfully.
% db2 activate database hsubsys1
DB20000I The ACTIVATE DATABASE command completed successfully.
% db2 activate database hsubsys2
DB20000I The ACTIVATE DATABASE command completed successfully.
```

Rebind packages. (as hpssdb)

```
% db2rbind hcfg -l /tmp/hcfg.log all
Rebind done successfully for database 'HCFG'.


% db2rbind hsubsys1 -l /tmp/hsubsys1.log all
Rebind done successfully for database 'HSUBSYS1'.

% db2rbind hsubsys2 -l /tmp/hsubsys1.log all
```

For other post migration task considerations check out:

http://publib.boulder.ibm.com/infocenter/db2luw/v9r5/index.jsp?topic=/com.ibm.db2.luw.qb.migration.doc/doc/t0007200.html

Compare DB2 configuration information.

```
% db2 get dbm cfg > dbm_cfg.v95
% diff dbm_cfg.v91 dbm_cfg.v95
% db2set -all > reg_hpssdb.v95
% view reg_hpssdb.v9
```

Note: Verify that the DB2INSTDEF is set to *hpssdb*

```
DB2INSTDEF=hpssdb

% set |grep DB2 > env_hpssdb.v95
% view env_hpssdb.v95
```

Note: Verify that DB2DIR is set to `/opt/IBM/db2/V9.5`

```
DB2DIR=/opt/IBM/db2/V9.5
```

Set the following parameter so HPSS will start. This is not be needed if it's already in reg_hpssdb.v9.

```
% db2set DB2COMM=tcpip
```

13.3. Reverting to DB2 v9.1 from v9.5

Backing out the upgrade will cause you to lose any changes to HPSS since the last backup was taken.

Shut down HPSS.

Drop the databases.

```
% db2start
% db2 drop database hcfg
DB20000I The DROP DATABASE command completed successfully.
% db2 drop database hsubsys1
DB20000I The DROP DATABASE command completed successfully.
```

Drop the instance.

```
% cd /opt/IBM/db2/V9.5/instance
% db2stop
% db2idrop hpssdb

DBI1070I Program db2idrop completed successfully.
```

Restore the V91 databases using **mkhpss** (the **mkhpss** version should be for the HPSS version you are restoring).

Run **mkhpss**. It may be necessary to set the DISPLAY before executing.

```
% export DISPLAY=xyz:0
% mkhpss
```

From the *Root Subsystem Machine* submenu in the left panel, click the *Configuration* expansion icon and then the *DB2 Services* icon. Everything should remain with its default value except for the following:

```
Password: (Enter hpss ID's current password)
Verify Password: (Enter hpss ID's current password)
Log File Directory: /hpssdb_logs/subsys1
```

Tablespace Config: Select each tablespace and click *Edit*. In the *Tablespace Config* window, select the "Tablespace Type" as listed in the table below. Then click *Add*, to assign the DB2 Container Path according to the table below. Select the container path in the list, and click *Open*. Then click *OK*, in the *Tablespace Config* window to complete the configuration of the tablespace.

As each Tablespace is configured, the *Config Status* field in the main *HPSS Installation & Configuration* window will change to "OK".

Below is a sample of a possible database configuration for HPSS.

Tablespace Name	**Tablespace Type**	**Container Path**
USERSPACE1	System Managed (filesystem)	/hpssdb_userspace1
TABLES	Database managed (raw devices)	/dev/rdbs1_tables
BFDALLOC	Database managed (raw devices)	/dev/rdbs1_bfdalloc
NSOBJECT	Database managed (raw devices)	/dev/rdbs1_nsobject
STSGTAPE	Database managed (raw devices)	/dev/rdbs1_stsgtape
BFTAPESEG	Database managed (raw devices)	/dev/rdbs1_bftapeseg
BITFILE	Database managed (raw devices)	/dev/rdbs1_bitfile
STSGDISK	Database managed (raw devices)	/dev/rdbs1_stsgdisk
INDEXES	Database managed (raw devices)	/dev/rdbs1_indexes
NSOBJIDX	Database managed (raw devices)	/dev/rdbs1_nsobjidx

When all tablespaces display a status of "OK", click *Configure DB2 Services* at the bottom of the window. Click *yes* to acknowledge the *'Are you sure you want to do this?'* window. Verify that the operation was successful by checking for the following status message in the *Command Output* window:

```
## run command exited with status 0
```

If the operation was not successful, resolve the reported error(s) and rerun the operation. Click *Done* on the *Command Output* window to close the window.

Restore the databases from backup.

- In this example, assuming that the backup images are in */hpssdb_backups+
- Exit and re-enter shell to clear old DB2 environmental variables.

```
% . /var/hpss/hpssdb/sqllib/db2profile
% cd /hpssdb_backups/cfg
% ls -ltr
total 209072
-rw-r----- 1 hpssdb hpssdb 53522432 Jul 23 11:02
HCFG.0.hpssdb.NODE0000.CATN0000.20070723110142.001
-rw-r----- 1 hpssdb hpssdb 53522432 Oct 25 10:52
HCFG.0.hpssdb.NODE0000.CATN0000.20071025105146.001
```

```
% db2 restore database hcfg from /hpssdb_backups/cfg taken at
20071025105146 without rolling forward

DB20000I The RESTORE DATABASE command completed successfully.

% cd ../subsys1
% ls -ltr
total 512568
-rw-r----- 1 hpssdb hpssdb 126693376 Jul 23 11:02
HSUBSYS1.0.hpssdb.NODE0000.CATN0000.20070723110213.001
-rw-r----- 1 hpssdb hpssdb 135741440 Oct 25 10:53
HSUBSYS1.0.hpssdb.NODE0000.CATN0000.20071025105257.001

% db2 restore database hsubsys1 from /hpssdb_backups/subsys1 taken at
20071025105257 without rolling forward
DB20000I The RESTORE DATABASE command completed successfully.
```

Note: If **mkhpss** was used to configure DB2, the following warning will be displayed:

```
SQL2523W Warning! Restoring to an existing database that is different
from the database on the backup image, but have matching names. The
target database will be overwritten by the backup version. The
Roll-forward recovery logs associated with the target database will be
deleted.
Do you want to continue ? (y/n) y
```

Type "y" to continue.

Catalog the databases (only do if restoring databases by hand).

```
% db2 catalog tcpip node LOCAL remote portland server 60000
% db2 catalog db hcfg as cfg at node LOCAL
% db2 catalog db hsubsys1 as subsys1 at node LOCAL
```

Uninstall DB2 V9.5 (optional).

Exit and re-enter shell to clear old db2 environmental variables.

```
% cd /opt/IBM/db2/V9.5/install
% db2_deinstall -a
DBI1016I Program db2_deinstall is performing uninstallation.
          Please wait.
The execution completed successfully.

For more information see the DB2 installation log at
"/tmp/db2_deinstall.log.655380".
```

13.4. DB2 v9.5 Configuration Changes

In order to take advantage of the large tablespaces, sites will need to reorganize tables and indexes for all DMS tablespaces. To take advantage of automatic memory management, sites should consider altering settings to AUTOMATIC for the following parameters:

- Buffer pools (controlled by the ALTER BUFFERPOOL and CREATE BUFFERPOOL statements)
- Package cache (controlled by the pckcachesz configuration parameter)

- Locking memory (controlled by the locklist and maxlocks configuration parameters)
- Sort memory (controlled by the sheapthres_shr and the sortheap configuration parameters)
- Total database shared memory (controlled by the database_memory configuration parameter)

The conversions automatically grant all permissions to user *hpss* and *root*. This is because HPSS servers typically run as these users and should have full access to the DB2 tables. Any custom permissions a site has on tables needs to be set by hand at this point. The new tables that would need custom permissions are:

HPSS Configuration Database

- CORE
- COS
- GLOBAL
- FILESYSTEMS
- FILESYSTEMSMOVERVIEW
- MIGPOL
- PVLACTIVITY
- PVLDRIVE
- PVLPV
- STORSUBSYS
- SCLASS
- AUTHZACL

HPSS Subsystem Databases

- BFCOSCHANGE
- BFMIGRREC
- BFDISKALLOCREC
- BFSSUNLINK
- BITFILE
- NSOBJECT
- STORAGESEGDISKEXTENTS
- STORAGESEGAUX

- STORAGESEGTAPEABSADDR
- SSPVDISK
- SSPVTAPE
- STORAGEMAPTAPE
- STORAGESEGDISK
- STORAGESEGTAPE
- VVTAPEABSADDR

The easiest way to add custom permissions to tables is to use the DB2 GRANT statement as the instance owner. Here are some examples:

```
% db2 grant all on hpss.BFCOSCHANGE to user hpss
% db2 grant all on hpss.BITFILE to group hpss
```

13.5. Upgrading to DB2 9.7

When upgrading to DB2 9.7, the same pre-upgrade steps that are described in the previous section *Upgrading from DB2 9.1 Section 13.2.2, "Upgrading DB2 from v9.1"*, such as backing up the databases, saving configuration information, performing pre-migration checks, etc. should be followed. Detailed information can be found at the DB2 9.7 Information Center (http://publib.boulder.ibm.com/infocenter/db2luw/v9r7/index.jsp?topic=/com.ibm.db2.luw.qb.upgrade.doc/doc/c0023662.html) or by contacting HPSS support.

13.6. Upgrading to DB2 10.5

When upgrading to DB2 10.5, the same pre-upgrade steps that are described in the previous section *Upgrading from DB2 9.1 Section 13.2.2, "Upgrading DB2 from v9.1"*, such as backing up the databases, saving configuration information, performing pre-migration checks, etc. should be followed. Detailed information can be found at the DB2 10.5 Information Center (http://pic.dhe.ibm.com/infocenter/db2luw/v10r5/index.jsp?topic=/com.ibm.db2.luw.qb.upgrade.doc/doc/c0023662.html) or by contacting HPSS support.

13.7. Upgrading AIX

Refer to documentation specific to AIX to perform the upgrade. There are no specific considerations or variances for HPSS in upgrading AIX. HPSS should be rebuilt following the upgrade.

13.8. Upgrading Linux

Refer to documentation specific to Linux to perform the upgrade. The only specific consideration for DB2 and HPSS on Linux is to be aware that after the Linux upgrade often the DB2 devices used for tablespaces used by HPSS have their permissions or ownership modified as a result of the operating system upgrade. The problem is often not noticed until HPSS servers are started and SSM Alarms &

Events shows errors accessing tables or metadata. The problem can be corrected by shutting down HPSS and DB2, correcting the permissions on the devices to allow access by the DB2 instance owner (e.g. hpssdb), and restarting DB2 and HPSS. HPSS should be rebuilt following the upgrade.

Chapter 14. Troubleshooting

This section covers warnings and errors that may occur during the conversion process and presents help in resolving the problem.

14.1. DBI and DBD

The following section provides information and troubleshooting when installing Perl's DBI module or when installing DB2's Perl DBD module. DBI was available for Linux for PowerPC platforms, but the DB2 DBD did not successfully install on Linux for PowerPC.

14.1.1. DBI and DBD - References

- DBI Mail List Archives

  http://groups.google.com/groups?group=perl.dbi.users

  http://www.xray.mpe.mpg.de/mailing-lists/dbi/

  http://www.mail-archive.com/dbi-users%40perl.org/

- Driver and Database Documentation (especially links at the bottom of the page):

  http://search.cpan.org/~timb/DBI/DBI.pm#Driver_and_Database_Documentation

- DBI Frequently Asked Questions:

  http://dbi.tiddlyspot.com/

14.1.2. Installation Problems

Problem:

Make step of DBD produces the following error:

```
        cc_r -c -I"/var/hpss/hpssdb/sqllib/include"
-I"/usr/opt/perl5/lib/site_perl/5.8.2/aix-thread-multi/auto/DBI"
-D_ALL_SOURCE -D_ANSI_C_SOURCE -D_POSIX_SOURCE -qmaxmem=16384
-qnoansialias -DUSE_NATIVE_DLOPEN -DNEED_PTHREAD_INIT -q32
-D_LARGE_FILES -qlonglong -O -DVERSION=\"1.1\" -DXS_VERSION=\"1.1\"
"-I/usr/opt/perl5/lib/5.8.2/aix-thread-multi/CORE" -DDB2_CACHE_FIX DB2.c
"DB2.xs", line 124.8: 1506-276 (S) Syntax error: possible missing ':'?
"DB2.xs", line 125.23: 1506-045 (S) Undeclared identifier svp.
"DB2.xs", line 125.9: 1506-045 (S) Undeclared identifier svp.
make: 1254-004 The error code from the last command is 1.
```

Description:

Failed to compile DB2.c on AIX 5.3 with DB2 v8.1 fix pack 15 and VAC compiler v5.

Resolution:

Upgrading to VAC compiler v7 resolved the problem.

Problem:

Make test step of DBD installation does not complete successfully.

Description:

The test phase of DBD installation on AIX fails with errors.

Resolution:

Ensure **LD_LIBRARY_PATH** is updated properly with the location of your DB2 libraries (specifically `libdb2.so`). For example: `LD_LIBRARY_PATH=/opt/IBM/db2/V8.1/lib64`

One site had a particular problem with `DynaLoader.pm` not finding the `libdb2.so` library even though the installer knew the location of DB2. This site set up a symbolic link as such as a workaround: `/usr/lib64/libdb2.so` -> `/opt/IBM/db2/V8.1/lib64/libdb2.so`

14.2. Pre-7.1 Startup Task, hpss_convert_71 Step 8

Problem:

Error executing **hpss_convert_71** step 8

```
alter table HPSS.storagemaptape add constraint tapevvmapcon1 foreign key
(vvid) references HPSS.vvtape (vvid) on delete cascade

[IBM][CLI Driver][DB2/6000] SQL0667N The FOREIGN KEY "TAPEVVMAPCON1"
cannot be created because the table contains rows with foreign key
values that cannot be found in the parent key of the parent table.
SQLSTATE=23520
```

Description:

The above means that there is orphaned metadata in the STORAGEMAPTAPE table. Specifically, there exists a record (by VVID) in the STORAGEMAPTAPE table that does not exist in the VVTAPE table.

Resolution:

Work with IBM HPSS Customer Support to identify the orphaned VVID and understand why the metadata is orphaned. If no HPSS data is at risk, removing the orphaned record from the STORAGEMAPTAPE table will resolve the problem.

Problem:

Error executing **hpss_convert_71** step 8

```
performing pre-7.1 startup tasks for HSUBSYS1
..........................DBD::DB2::db do failed: [IBM][CLI
Driver][DB2/LINUXX8664] SQL0750N The source table cannot be renamed
because it is referenced in a view, materialized query table, trigger,
SQL function, SQL method, check constraint, referential constraint, or
XSR object. SQLSTATE=42986

Error executing: rename table HPSS.nsobject to
pre71_nsobject[IBM][CLI Driver][DB2/LINUXX8664] SQL0750N The source
table cannot be renamed because it is referenced in a view, materialized
query table, trigger, SQL function, SQL method, check constraint,
referential constraint, or XSR object. SQLSTATE=42986

Error: /opt/hpss/bin/convert71/hpss_convert_71_subsys_pre71start.
```

Description:

The above means that a table could not be renamed. This is most likely due to a user-defined function specific to your site.

Resolution:

The user-defined function must be removed for the conversion to begin. To identify a user-defined function, execute:

```
% db2 "select funcname from syscat.funcdep where bname='NSOBJECT'"
```

Where the value NSOBJECT is substituted with the table name provided in the error message from the conversion program. Then execute the following to drop the user-defined function.

```
% db2 "drop specific function hpss.SQL071203153906500"
```

Where SQL071203153906500 is the result of the previous query, the *funcname*.

Now you will have to complete the previously incomplete conversion step and ignore some of the errors. You can do this by executing the conversion program that failed with the -f flag to force past error. You get the name of the underlying conversion program from the original error statement. The example above would be `/opt/hpss/bin/convert71/hpss_convert_71_subsys_pre71start`. So, you would execute that program as the DB2 instance owner (e.g. hpssdb) with the -f flag (no arguments). Since the program will continue executing past errors, check the error output carefully and ensure there aren't any errors beyond the expected errors stating that there are duplicates or that something already exists. Be very careful to return to using the **hpss_convert_71** program after executing the one program with the -f flag. Do not use the -f flag on any other underlying programs or for any other conversion steps or metadata may not be converted properly due to the extra steps and checks that the **hpss_convert_71** program performs.

14.3. Conversion Verification Problems

Problem:

Error executing the **hpss_convert_71_subsys_verify** program.

```
checking 6.2 table VVTAPEABSADDR ... DBD::DB2::db prepare failed:
```

```
[IBM][CLI Driver][DB2/6000] SQL0911N The current transaction has been
rolled back because of a deadlock or timeout. Reason code "2".
SQLSTATE=40001
DBD::DB2::st execute failed: [IBM][CLI Driver][DB2/6000] SQL0911N The
current transaction has been rolled back because of a deadlock or
timeout. Reason code "2". SQLSTATE=40001
Statement has no result columns to bind (perhaps you need to call
execute first) at ./hpss_convert_71_subsys_verify line 189.
```

Description:

The verification program is performing a left outer join between the source and target replication tables. If heavy load is placed on the source or target table, a deadlock may occur such as described above. This error simply means that the verification program was unable to check that particular table.

Resolution:

If verification of the table is required, consider either rerunning the program or wait until load on the source or target tables diminishes.

Problem:

Error executing the **hpss_convert_71_subsys_verify** program.

```
checking 6.2 table NSOBJECT ... the following record(s) are not in the
7.1 table:
      OBJECT_ID: 111
      OBJECT_ID: 123
```

Description:

The above message is not really an error, but provides OBJECT_IDs for NSOBJECT records that exist in the source table, but not the target table yet. During the initial replication startup, the records won’t appear until the full refresh (DB2 LOAD) completes. After the full refresh completes, these are records that should be processed upon the next capture and apply cycle completion (5 minutes).

Resolution:

Rerun after the full refresh completes or after the next capture and apply cycle completes to ensure that the same records no longer exist. If running the verification program while HPSS 6.2 is running, there is expected to be records that exist in HPSS 6.2 not in HPSS 7.1; however, this will be problematic if the number of records continue to increase over time as that would indicate that the replication programs are not able to keep pace with the volume of updates to the HPSS 6.2 database. In this case, you would want to consider seeing the tuning section in this guide for changing the capture and apply cycles to continuous.

Problem:

Error executing the **hpss_convert_71_subsys_verify** program.

```
checking 6.2 table BITFILE ... the following record(s) are not in the
```

```
7.1 table:
      BFID: 87f68ad0e82811dc8cf3000d60500e9bd1fdf3aae81b11dc9e0f000000000004
      BFID: 87f7b978e82811dc8cf3000d60500e9bd1fdf3aae81b11dc9e0f000000000004
```

Description:

The above message is not really an error, but provides BFIDs for BITFILE records that exist in the source table, but not the target table yet. During the initial replication startup, the records won't appear until the full refresh (DB2 LOAD) completes. After the full refresh completes, these are records that should be processed upon the next capture and apply cycle completion (5 minutes).

Resolution:

Rerun after the full refresh completes or after the next capture and apply cycle completes to ensure that the same records no longer exist. If running the verification program while HPSS 6.2 is running, there is expected to be records that exist in HPSS 6.2 not in HPSS 7.1; however, this will be problematic if the number of records continue to increase over time as that would indicate that the replication programs are not able to keep pace with the volume of updates to the HPSS 6.2 database. In this case, you would want to consider seeing the tuning section in this guide for changing the capture and apply cycles to continuous.

14.4. DB2 Upgrade Errors

Problem:

Various errors upon installing the DB2 software.

Description:

If the installation is not performed as the root user and without sourcing the db2profile of any existing instances of DB2, the software may produce different errors due to trying to allow you to have multiple instances of HPSS on the same system.

Resolution:

Uninstall the DB2 v9.5 software. Reinstall the DB2 v9.5 software and follow the migration guide or instructions included in this document as the root user and without sourcing the db2profile of any existing DB2 instance (e.g. ldap, hpssdb).

Problem:

Tablespace access not allowed upon completion of the migration under linux.

Description:

Sites have reported a common problem with migrating databases under Linux is that the software updates the permissions on the underlying devices for database containers to exclude the HPSS instance owner from accessing them with rwx permissions.

Resolution:

Update the permissions on the database container devices (e.g. `/dev`) to allow rwx permissions by the HPSS instance owner (e.g. hpssdb).

Problem:

The **db2ckmig** program hangs.

Description:

The **db2ckmig** program should exit after reporting whether the instance and databases are ready for migration or not. On AIX 5.3, the program hung and had to be terminated. This also caused the **db2imigr** program to hang as it calls **db2ckmig** initially.

Resolution:

CTRL-C the **db2ckmig** program after it reports whether the instance and databases are ready for migration or not. Also rename the program and create an empty (exit 0) **db2ckmig** in its place to proceed with the **db2imigr** program.

Appendix A. Glossary of Terms

5.3	Refers to version 5.3 of AIX
6.1	Refers to version 6.1 of AIX
6.2	Refers to version 6.2 of HPSS
7.1	Refers to version 7.1 of HPSS
7.3	Refers to version 7.3 of HPSS
7.3.2	Refers to version 7.3.2 of HPSS
7.3.3	Refers to version 7.3.3 of HPSS
7.4.1	Refers to version 7.4.1 of HPSS
7.4.2	Refers to version 7.4.2 of HPSS
8.2	Refers to version 8.2 of DB2
9.1	Refers to version 9.1 of DB2
9.5	Refers to version 9.5 of DB2
9.7	Refers to version 9.7 of DB2
10.5	Refers to version 10.5 of DB2
AIX	Advanced IBM Unix
ASN	Autonomous System Number, referring to a series of messages reserved for DB2 replication
BFID	HPSS bitfile identifier
DAS	DB2 Administration Server, used for remote administration of DB2 or for the GUI Control Center
DB2	Database 2
DBD	Database Driver, database specific implementation of DBI
DBI	Database Interface, Perl open source application programming interface for database interactions
DDL	Data Definition Language, used to define the table structure to DB2
DMS	Database Managed Storage, uses raw devices for tablespace containers
HPSS	High Performance Storage System software
Instance	Refers to a DB2 instance which is named according to the user that owns it
OBJECT_ID	HPSS namespace object identifier
RHEL	Red Hat Enterprise Linux
SLES	SUSE Linux Enterprise Server Instance
SMS	System Managed Space, uses filesystem facilities for tablespace containers
VVID	HPSS virtual volume identifier

Appendix B. References

1. **HPSS Installation Guide**
2. **HPSS Management Guide**
3. **DB2 9.1 Information Center (http://publib.boulder.ibm.com/infocenter/db2luw/v9/index.jsp)**
4. **DB2 9.5 Information Center (http://publib.boulder.ibm.com/infocenter/db2luw/v9r5/index.jsp)**
5. **DB2 9.7 Information Center (http://publib.boulder.ibm.com/infocenter/db2luw/v9r7/index.jsp)**
6. **DB2 10.5 Information Center (http://publib.boulder.ibm.com/infocenter/db2luw/v10r5/index.jsp)**
7. **HPSS Managetables Man Page (hpss_managetables.7)**

Appendix C. Developer Acknowledgments