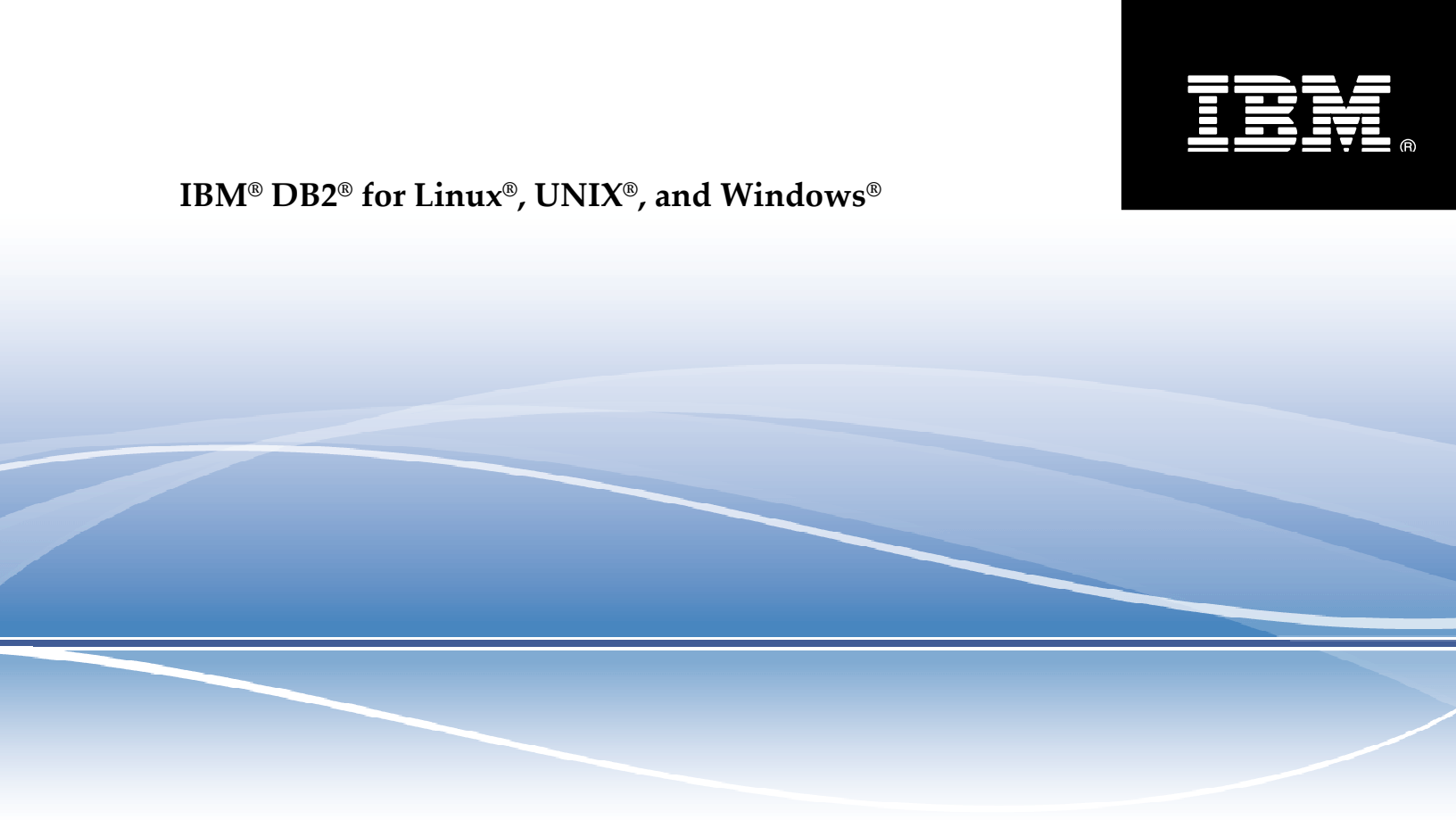

IBM® DB2® for Linux®, UNIX®, and Windows®

Upgrading to the DB2 pureScale™ Feature

Karen Pihowich
Manager, DB2 RDS, APM, Upgrade, and Catalog Services

Noureddine Brahimi
DB2 System Test

Last Updated: April 2011

Executive summary

In today's highly competitive marketplace, it is important to deploy a data-processing architecture that not only meets your immediate tactical needs but also can change to adapt to your future strategic requirements. In December 2009, IBM introduced the DB2 pureScale™ Feature for Enterprise Server Edition (the DB2 pureScale Feature). The DB2 pureScale Feature builds on familiar and proven design features from the IBM DB2 for z/OS® database software (DB2 for z/OS). IBM has brought the industry-leading technology and reliability of DB2 for z/OS to open systems.

The DB2 pureScale Feature provides the following key benefits

- **Virtually unlimited capacity** - The ability to scale out your system by easily adding servers to your cluster.
- **Application transparency** – The ability to leverage your existing applications without changes.
- **Continuous availability** - By providing an active-active architecture with inherent redundancy.
- **Reduced total cost of ownership (TCO)** - By providing simplified deployment and management of advanced technology.

This paper describes how to upgrade a DB2 V9.7 for Linux, UNIX, and Windows environment to the DB2 pureScale Feature (DB2 V9.8).

Introduction

The DB2 pureScale Feature leverages proven technology from the DB2 for z/OS data-sharing architecture to bring the active-active shared-disk technology to open systems.

The DB2 pureScale Feature offers you the following key benefits:

- **Virtually unlimited capacity.** The DB2 pureScale Feature provides virtually unlimited capacity by allowing the addition and removal of members on demand. The DB2 pureScale Feature can scale to 128 members and has a highly efficient centralized management facility that allows for very efficient scaling. The DB2 pureScale Feature also uses a technology called Remote Direct Memory Access (RDMA), which provides a highly efficient internode communication mechanism that also facilitates scaling.

- **Application transparency.** An application that runs in a DB2 pureScale environment does not have to know about the different members in the cluster or be concerned about partitioning data. The DB2 pureScale Feature automatically routes applications to the most appropriate members. The DB2 pureScale Feature also provides support for a great deal of the syntax that is used by database vendors other than IBM, allowing the applications that use that syntax to run in a DB2 pureScale environment with minimal or no changes. In many cases, you can gain the benefits of the DB2 pureScale Feature without having to modify your applications.

- **Continuous availability.** With the fully active-active configuration, if one member fails, processing can continue on the remaining active members. Only data that was being modified on the failed member is unavailable until database recovery is completed, which is performed for only that set of data and is very quick. In competing solutions, an entire system freeze might occur during database recovery.

- **Reduced total cost of ownership (TCO).** The DB2 pureScale Feature can help reduce TCO because it handles the deployment and maintenance of the components it includes. Integrated, simplified deployment and maintenance reduce possible steep learning curves that are associated with some of the competing technologies.

To understand better how the DB2 pureScale Feature offers the previously mentioned benefits, it is first important to understand a little more of the architecture. Figure 1 depicts the different components that are part of a DB2 pureScale configuration. Even though there are multiple advanced components, these components are largely transparent to you because the DB2 pureScale Feature deploys and manages them.

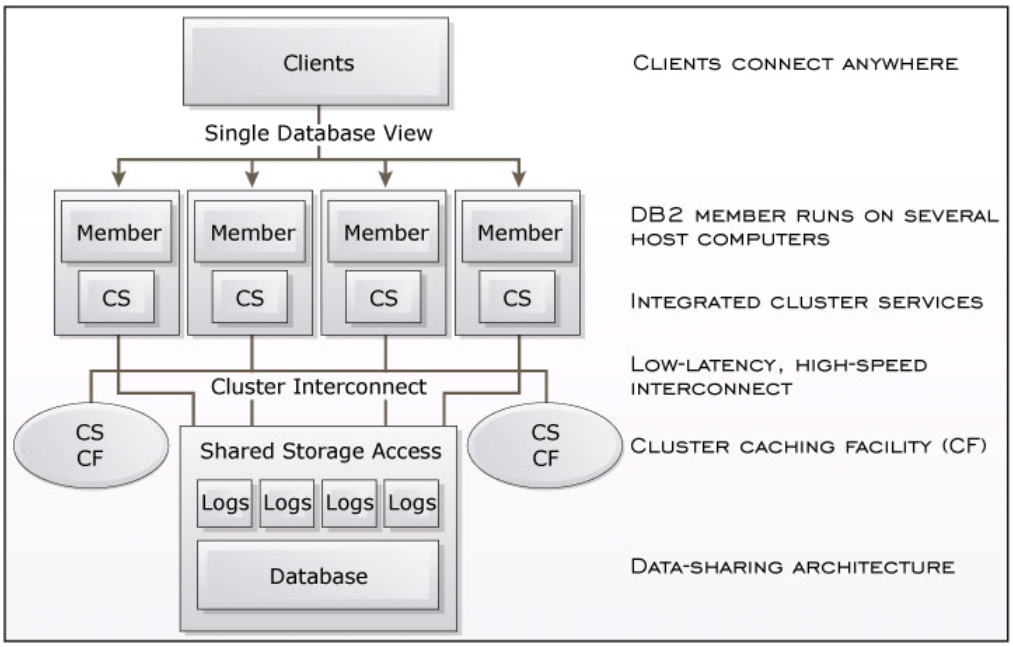

Figure 1, DB2 pureScale Feature topology overview

Each DB2 member represents a DB2 processing engine. The members cooperate with each other and the cluster caching facility (CF) to provide coherent access to the database from any member. You can add and remove members as processing demands change.

As shown in Figure 1, clients can connect to any member. You can modify the number of members without making any changes to the clients, because clients do not have to know how many members are active or which ones the clients connect to. The DB2 pureScale Feature can automatically load balance the clients across the different members based on the utilization of each member. If any host in the configuration fails, the DB2 pureScale Feature redirects clients among the active members on the remaining hosts.

Integrated with the DB2 pureScale Feature is a cluster services layer that provides failure detection, recovery automation, and a clustered file system. These technologies are integrated within the DB2 pureScale Feature and leverage IBM technologies that are optimized for DB2 software. These technologies include IBM Tivoli® Systems Automation for Multiplatforms (Tivoli® SA MP), IBM Reliable Scalable Cluster Technology (RSCT), and IBM General Parallel File System (GPFS™). The DB2 pureScale Feature automatically deploys and configures these technologies in accordance with a best-practice predefined configuration. These technologies are kept transparent to the end user; you do not have to configure them.

In the DB2 pureScale configuration, the members and the CFs must communicate. To make this communication as efficient as possible, the DB2 pureScale Feature leverages the RDMA technology. RDMA allows one server to read or write to the memory of another server without requiring any processor cycles on the target server. This mechanism, in conjunction with extremely high-speed networks such as InfiniBand,

allows for an extremely efficient transport layer, which enables the DB2 pureScale feature to scale efficiently.

The CFs provide a scalable and centralized locking mechanism to ensure data coherency. They also act as a fast cache for DB2 pages, leveraging RDMA technology to increase performance in situations where a physical disk operation might have been required. The CF and the efficient transport layer are two of the features that allow the DB2 pureScale Feature to scale so well, because each member does not have to negotiate with all other members when performing a task.

As mentioned previously, the DB2 pureScale Feature leverages a shared-disk technology. Any member can read or write to any portion of the database. If any member fails, the full set of data is still accessible from the other active members.

Upgrading to the DB2 pureScale Feature

Scenario overview

Objectives

This scenario shows how to upgrade a DB2 V9.7 Fix Pack 2 Enterprise Server Edition database to the DB2 pureScale Feature. The original database is a single-partition database on a POWER6 server. The upgraded database contains four logical partitions (LPARS) that are used as two members and two CFs.

Figure 2 shows the initial and final configurations for the database. Configuration details for each LPAR are in Appendix E.

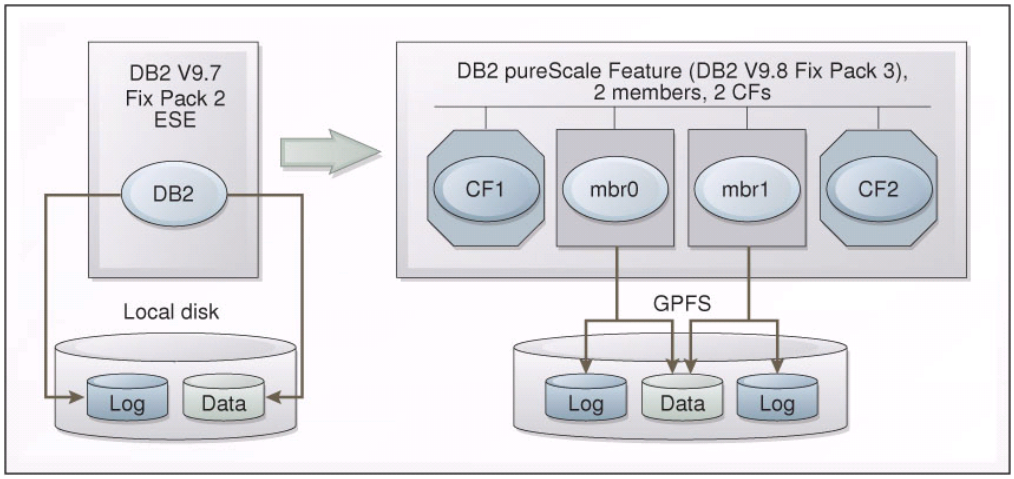

Figure 2. Initial and final database configurations

Upgrading to the DB2 pureScale Feature (DB2 V9.8) entails a few additional steps compared to upgrading to earlier DB2 releases, to ensure that the database meets DB2 pureScale Feature requirements. Two key requirements are as follows:

- All table spaces must be managed by automatic storage.
- All data and logs must be on a GPFS file system.

In this scenario, the database being upgraded is not on a GPFS file system and contains table spaces that are not managed by automatic storage. Therefore, the scenario includes steps to convert table spaces to be managed by automatic storage and to move the database to a GPFS file system. There are several ways to convert table spaces to be managed by automatic storage. This scenario shows one method. Information about additional methods and how to decide which one is appropriate for your database is provided in the appendixes.

High-level upgrade steps

In releases before DB2 V9.8, the steps to upgrade were as follows:

1. Install the code for the new DB2 release.
2. Issue the **db2ckupgrade** command (formerly known as the **db2ckmig** command) to verify that the database can be upgraded.
3. Upgrade the database server in one of two ways:
   - Upgrade the instance and then upgrade the database
   - Create a new instance at the new release level, and restore a full offline backup that you took in the previous release environment into the new instance.

As mentioned previously, upgrading to the DB2 pureScale Feature involves additional steps. This scenario requires the following steps:

1. Verify the instance type.
2. Install the DB2 pureScale Feature.
3. Set up the GPFS file system.
4. Check whether the database meets DB2 pureScale requirements.
5. Ensure that the database meets DB2 pureScale requirements.
6. Verify that the database is ready to upgrade.
7. Upgrade the instance.
8. Upgrade the database.
9. Convert the instance to a DB2 pureScale instance type.
10. Perform post-upgrade steps.
11. Add a member and a CF.

Figure 3 shows how the database environment changes as these steps are completed.

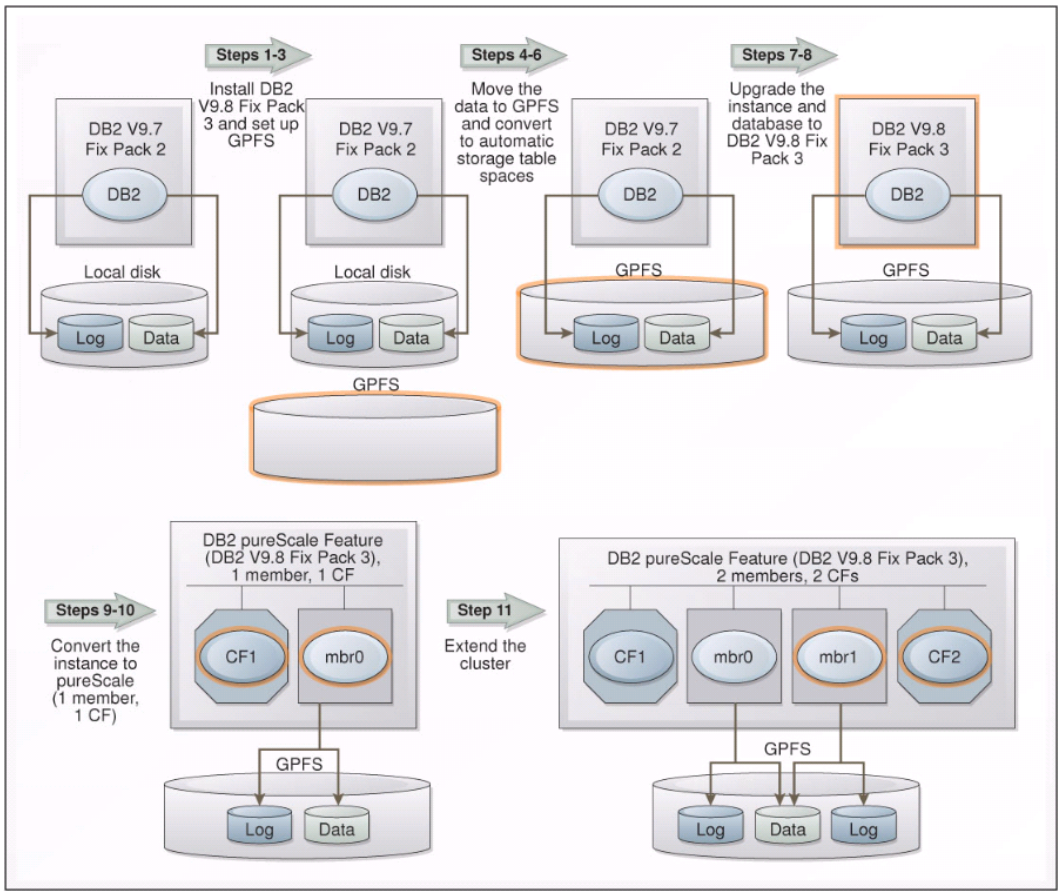

Figure 3. High-level upgrade steps

Scenario details

The source database, which was created using DB2 Version 9.7 Fix Pack 2, is named DBS. DBS was created on a journaled file system (JFS) and contains two DMS user table spaces (TBSP1 and TBSP2) and DMS default table spaces (catalog, temporary, and user). A table TAB0 was created in TBSP1 with indexes in TBSP2, and data was inserted into TAB0.

This scenario shows the steps to upgrade the database to a DB2 pureScale (V9.8) environment with two members and two CFs.

The rest of this section provides details about the previously outlined upgrade steps. Sample commands and output are provided.

Step 1 – Verify the instance type

If your instance type is not Enterprise Server Edition (ESE), you must convert the instance type to ESE before upgrading to a DB2 pureScale environment.

To ensure that the instance type is ESE, perform these steps:

1. Determine whether the instance type is ESE by issuing the **`get database manager configuration`** command:

```
db2 get dbm cfg | grep 'Node type'
```

If the instance type is ESE, the following output is displayed:

```
Node type = Enterprise Server Edition with local and remote clients
```

2. If the output is different, convert the instance type to ESE by issuing the **db2iupdt** command, as root:

```
cd DB297DIR/instance
./db2iupdt instance name
```

where *DB297DIR* is the location where DB2 V9.7 Enterprise Server Edition is installed.

Step 2 - Install the DB2 pureScale Feature

To install a DB2 pureScale Feature copy, issue the **db2_install** command. The **db2_install** command installs all components that the DB2 pureScale Feature requires, including Tivoli SA MP and GPFS. If these products are already installed, the **db2_install** command does not modify them. Therefore, if these products are already installed on your system, before you issue the **db2_install** command you must ensure that these products are at the level that the DB2 pureScale Feature requires. For the required product levels, see the Installation prerequisites for DB2 pureScale Feature (AIX)[1] or the Installation prerequisites for DB2 pureScale Feature (Linux)[2] topic in the IBM DB2 pureScale Feature Information Center.

In this scenario, Tivoli SA MP and GPFS were not installed previously and will be installed by the **db2_install** command.

The following output shows that after entering the **db2_install** command, you are prompted for only two pieces of information:

- The installation directory (in this case, the default, `/opt/IBM/db2/V9.8`, is used)
- The keyword that represents the DB2 product to be installed (in this case ESE_DSF)

The rest of the installation is automatic.

```
./db2_install

Default directory for installation of products - /opt/IBM/db2/V9.8
***********************************************************
Do you want to choose a different directory to install [yes/no] ?
no

Specify one of the following keywords to install DB2 products.
```

[1] http://publib.boulder.ibm.com/infocenter/db2luw/v9r8/index.jsp?topic=/com.ibm.db2.luw.sd.doc/doc/r0054850.html
[2] http://publib.boulder.ibm.com/infocenter/db2luw/v9r8/index.jsp?topic=/com.ibm.db2.luw.sd.doc/doc/r0057441.html

```
  ESE_DSF

Enter "help" to redisplay product names.
Enter "quit" to exit.
************************************************************
ESE_DSF

DB2 installation is being initialized.

 Total number of tasks to be performed: 46
Total estimated time for all tasks to be performed: 2776 second(s)

Task #1 start
Description: Checking license agreement acceptance
Estimated time 1 second(s)
Task #1 end
……..
Task #46 start
Description: Updating global profile registry
Estimated time 3 second(s)
Task #46 end

The execution completed successfully.

For more information see the DB2 installation log at
"/tmp/db2_install.log.1273936".
```

Step 3 - Set up the GPFS file system

DB2 V9.8 includes the **db2cluster_prepare** command, which you use to set up a DB2 managed GPFS file system. A DB2 managed file system is recommended for a DB2 pureScale environment because it simplifies cluster management. When the DB2 product manages the GPFS cluster and file system, the DB2 installer and instance utilities add or remove the host and issue the required mount or unmount commands when a host is added to or removed from the DB2 pureScale instance. If you manage the GPFS file system, you must perform these steps manually when a host is added to or removed from the instance. For details, see User-managed file system[3] in the DB2 pureScale Feature Information Center.

The first step in setting up the file system is to determine what disks are available and which ones are suitable for the GPFS file system. This section shows how to do this in an AIX operating system. For details on doing this step in the Linux operating system, see Preinstallation checklist for DB2 pureScale Feature (Linux)[4] in the DB2 pureScale Feature Information Center. This topic contains a preinstallation checklist and shows how to get more details about the disks.

[3] http://publib.boulder.ibm.com/infocenter/db2luw/v9r8/topic/com.ibm.db2.luw.sd.doc/doc/c0056461.html
[4] http://publib.boulder.ibm.com/infocenter/db2luw/v9r8/topic/com.ibm.db2.luw.sd.doc/doc/r0057204.html

To set up a GPFS file system on an AIX operating system, perform these steps:

1. Display information about the physical volumes on the system by issuing the **lspv** command as root:

```
> lspv
hdisk0          00cc14e225758eee                    rootvg          active
hdisk1          00cc14e225ae923d                    homevg          active
hdisk2          None                                None
hdisk3          None                                None
hdisk4          None                                None
hdisk5          00cc14e2de8ce942                    db2instvg       active
hdisk6          None                                None
```

   The second column shows the physical volume identifier (PVID) for the disk, and the third column shows the volume group to which the physical volume belongs. In this case, hdisk2, hdisk3, hdisk4, and hdisk6 are available for use by the DB2 instance.

2. Change each available disk (hdisk2, hdisk3, hdisk4, and hdisk6) to a physical volume and define a PVID by issuing the **chdev** command for each disk from one host in the cluster:

```
> chdev -l hdisk2 -a pv=yes
> chdev -l hdisk3 -a pv=yes
> chdev -l hdisk4 -a pv=yes
> chdev -l hdisk6 -a pv=yes
```

3. On each of the other hosts in the cluster, remove the definitions for hdisk2, hdisk3, hdisk4 and hdisk6 by issuing the **rmdev** command:

```
> rmdev -dl hdisk2
> rmdev -dl hdisk3
> rmdev -dl hdisk4
> rmdev -dl hdisk6
```

   After you issue the **chdev** and **rmdev** commands, the **lspv** command displays information for only hdisk0, hdisk1, and hdisk5.

4. Retrieve the updated definitions for all the hosts by issuing the **cfgmgr** command on each host in the cluster:

```
> cfgmgr
```

   The **lspv** command shows the same results on all hosts in the cluster:

```
> lspv
hdisk0          00cc14e225758eee                    rootvg          active
hdisk1          00cc14e225ae923d                    homevg          active
hdisk2          00cc14e2869f6770                    None
hdisk3          00cc14e2869f68c7                    None
hdisk4          00cc14e2356ef748                    None
hdisk5          00cc14e2de8ce942                    db2instvg       active
hdisk6          00cc14e2869f6977                    None
```

5. Determine which disk to use as a *tiebreaker disk*. You will define a tiebreaker disk when you issue the **db2iupgrade** command in a later step. DB2 software uses the tiebreaker disk if cluster recovery is needed. The tiebreaker disk needs a minimum of only 25 MB. To see how big the disks are, issue the **bootinfo** command as root:

```
> bootinfo -s hdisk2
204800
> bootinfo -s hdisk3
10240
> bootinfo -s hdisk4
128
> bootinfo -s hdisk6
204800
```

The hdisk4 disk is the smallest disk; use it as a tiebreaker. The hdisk3 disk is the next smallest disk, which you can use for the `sqllib_shared` directory. The DB2 server puts all shared files into this directory. Use hdisk2 for logs and hdisk6 for data.

6. In the *DB298DIR*`/instance` directory, where *DB298DIR* is the location where DB2 V9.8 is installed, create a DB2 managed GPFS file system using hdisk3 by issuing the **db2cluster_prepare** command:

```
> cd DB298DIR/instance
> db2cluster_prepare -instance_shared_dev /dev/hdisk3
DBI1446I  The db2cluster_prepare command is running, please wait.

DB2 installation is being initialized.

 Total number of tasks to be performed: 1
Total estimated time for all tasks to be performed: 60 second(s)

Task #1 start
Description: Creating GPFS Cluster and Filesystem
Estimated time 60 second(s)
Task #1 end

The execution completed successfully.

For more information see the DB2 installation log at
"/tmp/db2cluster_prepare.log".
DBI1070I  Program db2cluster_prepare completed successfully.
```

Get the name of the file system by issuing the **db2cluster** command:

```
> db2cluster -cfs -list -filesystem

FILE SYSTEM NAME                      MOUNT_POINT
--------------------------------      ------------------------
db2fs1                                /db2sd_20110201154154
```

The file system created by the **db2cluster_prepare** command (db2fs1) will host DB2 shared files after the upgrade.

7. In the *DB298DIR*/instance directory, create a file system named **log** for logs on hdisk2 and a file system called **data** for data on hdisk6 by issuing the **db2cluster** command:

```
> cd DB298DIR/bin

> db2cluster -cfs -create -filesystem log -disk /dev/hdisk2
File system 'log' has been successfully created.

> db2cluster -cfs -create -filesystem data -disk /dev/hdisk6
File system 'data' has been successfully created.
```

You can issue the **db2cluster** command again to list all shared file systems:

```
> db2cluster -cfs -list -filesystem

FILE SYSTEM NAME                      MOUNT_POINT
--------------------------------      -------------------------
data                                  /db2fs/data
log                                   /db2fs/log
db2fs1                                /db2sd_20110201154154
```

8. Issue the **chown** command to change the ownership on the shared directories so that the instance owner can create folders and files there from any member that will be created later. In the following example, the instance owner is user ID brahimi under group pdxdb2:

```
> chown brahimi:pdxdb2 /db2fs/data /db2fs/log /db2sd_20110201154154
```

Step 4 - Check whether the database meets DB2 pureScale requirements

There are certain requirements that a database must meet before you can upgrade it to the DB2 pureScale Feature. To check whether the database meets requirements, issue the **db2checkSD** command. The **db2checkSD** command is located in the *DB298DIR*/bin directory, where *DB298DIR* is the directory where DB2 V9.8 is installed.

In this scenario issuing the **db2checkSD** command against database DBS identifies errors:

```
> db2checkSD dbs -l checkSD_before.log
DBT5002N  The db2checkSD utility completed with errors. The database or databases
cannot be upgraded to a DB2 pureScale environment because the db2checkSD utility
found database objects or features that are not supported in a DB2 pureScale
environment. The output log file is named "checkSD_before.log".
```

The contents of the log file that is generated by the **db2checkSD** command are as follows:

```
> cat checkSD_before.log
```

```
Version of DB2CHECKSD being run: VERSION 9.8.

Database: DBS

DBT5024N  The db2checkSD utility found the following database objects or features
that are not supported in a DB2 pureScale environment: one or more table space
containers or database log or storage paths that are not on a General Parallel
File System (GPFS).

DBT5012N  The db2checkSD utility found the following database objects or features
that are not supported in a DB2 pureScale environment: table spaces that do not
use automatic storage. The db2checkSD utility generated a user script named
"db2checkSD_DBS.clp".

DBT5002N  The db2checkSD utility completed with errors. The database or databases
cannot be upgraded to a DB2 pureScale environment because the db2checkSD utility
found database objects or features that are not supported in a DB2 pureScale
environment. The output log file is named "checkSD_before.log".
```

The **log** indicates the following problems:

- There are table space containers, logs, or storage paths that are not on a GPFS file system.
- There are table spaces that do not use automatic storage. You can use the generated **db2checkSD_DBS.clp** script to get information about the table spaces.

You must address these items before converting the database to the DB2 pureScale Feature.

Step 5 - Ensure that the database meets DB2 pureScale requirements

To ensure that the database meets DB2 pureScale requirements, perform these steps:

1. Get some information about the table spaces by running the **db2checkSD_DBS.clp** script. The contents of the **db2checkSD_DBS.clp** script are shown here:

```
> cat db2checkSD_DBS.clp
--- DBT5012N  The db2checkSD utility found the following database objects or
features that are not supported in a DB2 pureScale environment: table spaces that
do not use automatic storage. The db2checkSD utility generated a user script named
"db2checkSD_DBS.clp".

select tbsp_name, tbsp_id, tbsp_type, AUTO_STORAGE_HYBRID, tbsp_using_auto_storage
from table( sysproc.mon_get_tablespace( '',0) ) where TBSP_USING_AUTO_STORAGE=0 OR
AUTO_STORAGE_HYBRID=1;
```

2. Run the query in the script, which produces the following result:

```
TBSP_NAME         TBSP_ID TBSP_TYPE  AUTO_STORAGE_HYBRID TBSP_USING_AUTO_STORAGE
----------------- ------- ---------- ------------------- -----------------------
SYSCATSPACE             0 DMS                         0                       0
TEMPSPACE1              1 DMS                         0                       0
```

```
USERSPACE1              2 DMS                       0                       0
TBSP1                   3 DMS                       0                       0
TBSP2                   4 DMS                       0                       0

  5 record(s) selected.
```

3. The query results show that none of the table spaces uses automatic storage, so verify whether the database is enabled for automatic storage. A database that is enabled for automatic storage must have at least one automatic storage path.

   To check whether the database has any automatic storage paths, connect to the database and issue the **`get snapshot`** command:

```
> db2 connect to dbs

 Database Connection Information

 Database server        = DB2/AIX64 9.7.2
 SQL authorization ID   = BRAHIMI
 Local database alias   = DBS

> db2 get snapshot for all databases | grep -E 'Database name|storage'
Database name                               = DBS
Number of automatic storage paths           = 0
```

4. Database DBS does not have any automatic storage paths, so it is not enabled for automatic storage. Enable it by issuing the **`alter database`** command with the **`add storage`** clause, specifying the file system created for data in substep 7 of Step 3 - Set up the GPFS file system on page 14. The **`add storage`** clause is supported in DB2 V9.7 and later releases:

```
> db2 "alter database dbs add storage on '/db2fs/data/dbs'"
DB20000I  The SQL command completed successfully.
```

   Because the database is enabled for automatic storage and all permanent table spaces are of type DMS, you can perform a redirected restore to move the database to the GPFS file system and convert these table spaces to use automatic storage, as shown in the next steps. For other ways to move a database, depending on the types of the table spaces, see Appendix A. Options for getting existing table spaces to be managed by automatic storage on page 31.

5. Limit access to the database by connecting to the database and issuing the **`quiesce database`** command with the **`force connections`** parameter. The **`quiesce database`** command prevents new connections from being accepted, and the **`force connections`** parameter stops existing connections cleanly.

   After issuing the **`quiesce database`** command, disconnect from the database by issuing the **`connect reset`** and command, and use the **`list application`** command to verify that no users are connected to the database.

   The following example shows the commands to use:

```
> db2 connect to dbs

 Database Connection Information

 Database server        = DB2/AIX64 9.7.2
 SQL authorization ID   = BRAHIMI
 Local database alias   = DBS

> db2 quiesce db immediate force connections
DB20000I  The QUIESCE DATABASE command completed successfully.

>Wait few seconds here<

> db2 connect reset
  DB20000I  The SQL command completed successfully.

> db2 list application
  SQL1611W  No data was returned by Database System Monitor.
```

6. Take an offline backup by issuing the **backup database** command:

```
> db2 "backup db dbs to /db1/dbs_bkupimg"
  Backup successful. The timestamp for this backup image is : 20110202102358
```

7. Drop the old database. This is required because the redirected restore does not modify the *dbpath* if it is restoring to an existing database. The *dbpath* is where the database manager stores various control files for the database, and must be on a GPFS file system for DB2 pureScale. In this scenario the original database is on the same host as the new one, and the original database must be dropped so that the redirected restore will move the *dbpath*.

   Dropping the database deletes any log files in the active log path. If you are using log retention or log archiving and want to be able to restore using an older backup image and roll forward to a point in time before the upgrade, you must copy or move any log files in the active log path before dropping the database.

   You must unquiesce the database before you can drop it. To prevent access to the database after you unquiesce it but before you drop it perform the following steps:

   a. Issue the **uncatalog database** command.

   b. Re-catalog the database with a different name by issuing the **catalog database** command.

   c. Connect to the database using the name you specified on the **catalog database** command.

   d. Issue the un**quiesce database** command.

   e. Drop the database.

      The following example shows the commands:

```
> db2 uncatalog db dbs
DB20000I  The UNCATALOG DATABASE command completed successfully.
DB21056W  Directory changes may not be effective until the directory cache is
refreshed.

> db2 catalog db dbs as newdbs
DB20000I  The CATALOG DATABASE command completed successfully.
DB21056W  Directory changes may not be effective until the directory cache is
refreshed.

> db2 connect to newdbs

 Database Connection Information

 Database server        = DB2/AIX64 9.7.2
 SQL authorization ID   = BRAHIMI
 Local database alias   = DBS

> db2 unquiesce db
DB20000I  The UNQUIESCE DATABASE command completed successfully.

> db2 connect reset
DB20000I  The SQL command completed successfully.

> db2 drop db newdbs
DB20000I  The DROP DATABASE command completed successfully.
```

8. Perform a redirected restore. A redirected restore consists of a two-step database restore process with an intervening step in which you redefine the table space containers:

   a. Issue the **`restore database`** command with the **`redirect`** option. The command uses the two GPFS file systems that you created for data and logs.

   b. Redefine the containers for the four permanent table spaces by issuing the **`set tablespace containers`** command. Notice that there is no **`set tablespace containers`** command for the temporary table space because you cannot convert temporary DMS table spaces to use automatic storage. Later, you will drop the temporary table space and create a new one that is managed by automatic storage.

   c. Issue the **`restore database`** command again, this time specifying the **`continue`** option.

      The following example shows the commands:

```
db2 “restore db dbs from /db1/dbs_bkupimg \
     on /db2fs/data/dbs                   \
     dbpath on /db2fs/data/dbs            \
     into dbs                             \
     newlogpath /db2fs/log                \
     redirect                             \
     without rolling forward              \
     without prompting”
SQL1277W  A redirected restore operation is being performed.  Table space
configuration can now be viewed and table spaces that do not use automatic
storage can have their containers reconfigured.
```

```
DB20000I  The RESTORE DATABASE command completed successfully.

db2 set tablespace containers for 0 using automatic storage
DB20000I  The SET TABLESPACE CONTAINERS command completed successfully.

db2 set tablespace containers for 2 using automatic storage
DB20000I  The SET TABLESPACE CONTAINERS command completed successfully.

db2 set tablespace containers for 3 using automatic storage
DB20000I  The SET TABLESPACE CONTAINERS command completed successfully.

db2 set tablespace containers for 4 using automatic storage
DB20000I  The SET TABLESPACE CONTAINERS command completed successfully.

db2 restore db dbs continue
DB20000I  The RESTORE DATABASE command completed successfully.
```

9. Verify that all of the table space containers are on the GPFS file system and are enabled for automatic storage by connecting to the database and issuing the **`get snapshot`** command:

```
> db2 connect to dbs

   Database Connection Information

 Database server        = DB2/AIX64 9.7.2
 SQL authorization ID   = BRAHIMI
 Local database alias   = DBS

> db2 get snapshot for tablespaces on dbs | grep -E "Tablespace name|Tablespace
ID|Using automatic storage|Container Name"

Tablespace name                            = SYSCATSPACE
  Tablespace ID                            = 0
  Using automatic storage                  = Yes
  Container Name                           =
/db2fs/data/dbs/brahimi/NODE0000/DBS/T0000000/C0000000.CAT

Tablespace name                            = TEMPSPACE1
  Tablespace ID                            = 1
  Using automatic storage                  = No
  Container Name                           = /db1/brahimi/dbs/temp

Tablespace name                            = USERSPACE1
  Tablespace ID                            = 2
  Using automatic storage                  = Yes
  Container Name                           =
/db2fs/data/dbs/brahimi/NODE0000/DBS/T0000002/C0000000.LRG

Tablespace name                            = TBSP1
  Tablespace ID                            = 3
  Using automatic storage                  = Yes
  Container Name                           =
/db2fs/data/dbs/brahimi/NODE0000/DBS/T0000003/C0000000.USR

Tablespace name                            = TBSP2
  Tablespace ID                            = 4
  Using automatic storage                  = Yes
  Container Name                           =
/db2fs/data/dbs/brahimi/NODE0000/DBS/T0000004/C0000000.USR
```

The output shows that all the containers are on the GPFS file system and all the table spaces are enabled for automatic storage, except the default temporary table space.

10. Get details about TEMPSPACE1 by issuing the **`get snapshot`** command:

```
> db2 get snapshot for tablespaces on dbs

Part of the result for temporary tablespace

Tablespace name                            = TEMPSPACE1
  Tablespace ID                            = 6
  Tablespace Type                          = System managed space
  Tablespace Content Type                  = System Temporary data
  Tablespace Page size (bytes)             = 4096
  Tablespace Extent size (pages)           = 32
  Automatic Prefetch size enabled          = Yes
  Buffer pool ID currently in use          = 1
  Buffer pool ID next startup              = 1
  Using automatic storage                  = Yes
  File system caching                      = Yes
  Tablespace State                         = 0x'00000000'
   Detailed explanation:
     Normal
  Tablespace Prefetch size (pages)         = 32
```

11. Using the details in the **`get snapshot`** command output, such as the prefetch size and page size information, create a new temporary table space with the same characteristics as TEMPSPACE1. Then drop the original temporary table space:

```
> db2 "create system temporary tablespace tempspace2 pagesize 4K managed by
automatic storage extentsize 32 prefetchsize automatic bufferpool ibmdefaultbp"
DB20000I  The SQL command completed successfully

> db2 drop tablespace TEMPSPACE1
DB20000I  The SQL command completed successfully
```

12. Check whether the logs and the log archive files are on a GPFS file system by querying the database configuration file. As shown in the output, one of the log archive paths is not on a GPFS file system.

```
> db2 get db cfg for dbs | grep -E "Path to log files|LOGARCHMETH"
 Path to log files                          = /db2fs/log
 First log archive method           (LOGARCHMETH1) = DISK:/db1/dblog/
 Second log archive method          (LOGARCHMETH2) = OFF
```

13. Change the setting of the **`logarchmeth1`** database configuration parameter to the file system that was created for the logs in substep 7 of Step 3 - Set up the GPFS file system on page 14:

```
> db2 update db cfg for dbs using logarchmeth1 disk:/db2fs/log
DB20000I  The UPDATE DATABASE CONFIGURATION command completed successfully.
```

14. Verify that all storage paths are on a GPFS file system by issuing the **get snapshot** command again:

```
> db2 get snapshot for all databases | grep -E 'Database name|storage'
Database name                               = DBS
Number of automatic storage paths           = 1
Automatic storage path                      = /db2fs/data/dbs
```

All storage paths are on a GPFS file system, so no action is needed.

15. The **db2checkSD** command checks whether the **auto_stats_prof** and **dyn_query_mgmt** database configuration parameters and the **federated** and **health_mon** database manager configuration parameters are turned off. In this example they are all off, but if they are on, the **db2checkSD** command writes warnings to the log file. You can ignore the warnings because the DB2 software turns these parameters off when it converts the instance to a DB2 pureScale type, or you can turn these parameters off manually to prevent the **db2checkSD** command from generating the warnings. The following example shows the commands to turn off these options. The database manager must be stopped and restarted to have the new value for **dyn_query_mgmt** take effect :

```
> db2 update db cfg for dbs using auto_stats_prof off
DB20000I  The UPDATE DATABASE CONFIGURATION command completed successfully.

> db2 update db cfg for dbs using dyn_query_mgmt disable
DB20000I  The UPDATE DATABASE CONFIGURATION command completed successfully.

> db2 update dbm cfg using health_mon off
DB20000I  The UPDATE DATABASE CONFIGURATION command completed successfully.

> db2 update dbm cfg using federated no
DB20000I  The UPDATE DATABASE CONFIGURATION command completed successfully.

> db2stop
SQL1064N  DB2STOP processing was successful.

> db2start
SQL1063N  DB2START processing was successful.
```

Step 6 - Verify that the database is ready to upgrade

Before upgrading you must run the **db2checkSD** command and the **db2ckupgrade** command. The **db2checkSD** command verifies that the database meets DB2 pureScale requirements, and the **db2ckupgrade** command checks to make sure the database can be upgraded to the new release. For a list of these checks, see db2ckupgrade - Check database for upgrade command[5] in the DB2 pureScale Feature Information Center.

To verify that the database is ready to upgrade, perform these steps:

1. Ensure that the instance is started.

[5] http://publib.boulder.ibm.com/infocenter/db2luw/v9r8/topic/com.ibm.db2.luw.admin.cmd.doc/doc/r0002028.html

2. From *DB298DIR*/bin, where *DB982DIR* is the directory where DB2 V9.8 is installed, issue the **db2ckupgrade** command:

```
> DB298DIR/bin/db2ckupgrade dbs -l ~/db2ckupgrade.log
db2ckupgrade was successful. Database(s) can be upgraded.
```

3. From the same directory, issue the **db2checkSD** command. Direct the output to a file that is different from the one that you used when you issued the command earlier.

```
> DB298DIR/bin/db2checkSD dbs -l ~/db2checkSD2.log
DBT5000I  The db2checkSD utility completed successfully. The specified database
can be upgraded to a DB2 pureScale environment. The output log file is named
"db2checkSD2.log".
```

4. It is recommended that you perform a full offline database backup before you upgrade the database in order to have a backup image that is as current as possible. If an error occurs during the upgrade process, you can use this backup for recovery purposes. Substeps 5 and 6 from Step 5 - Ensure that the database meets DB2 pureScale requirements on page 15 show the commands to use to quiesce the database and take a full backup.

5. Ensure that there is enough system temporary table space and log space. For more details, see Managing your disk space requirements before upgrading to a DB2 pureScale environment [6] in the DB2 pureScale Information Center.

   To verify that the amount of space available is sufficient, connect to the database and perform the following steps:

   a. Check the amount of catalog space (SYSCATSPACE) by issuing the following query:

```
db2 "SELECT SUBSTR(TBSP_NAME,1,15) NAME, TBSP_TYPE TYPE,
     TBSP_TOTAL_PAGES TOTAL_PGS, TBSP_USED_PAGES USED_PGS,
     TBSP_FREE_PAGES FREE_PGS,TBSP_MAX_SIZE MAX_SZ, TBSP_PAGE_SIZE PG_SZ
     FROM SYSIBMADM.TBSP_UTILIZATION
     WHERE TBSP_NAME='SYSCATSPACE')"

NAME            TYPE   TOTAL_PGS   USED_PGS     FREE_PGS     MAX_SZ     PG_SZ
--------------- ----   ---------   ----------   ----------   --------   ------
SYSCATSPACE     DMS       100000        40704        59264         -1     4096

  1 record(s) selected.
```

   Disk space requirements are as follows:

   - For SYSCATSPACE, the number of free pages, shown in the FREE_PGS column in the output, must be at least equal to the number of used pages in the USED_PGS column. Therefore, in this scenario, the minimum disk space that is required for SYSCATSPACE is 40704 four KB pages, or 163 MB. In this example the value of FREE_PGS is larger than the value of

[6] http://publib.boulder.ibm.com/infocenter/db2luw/v9r8/topic/com.ibm.db2.luw.sd.doc/doc/t0056893.html

USED_PAGES, so you just need to verify that there is enough disk space available.

- For TEMPSPACE2, the number of pages required could be twice the number of pages in the TOTAL_PGS column for SYSCATSPACE. Therefore, in this scenario, the minimum disk space that is required for TEMPSPACE2 is 2 x 100000 four KB pages, or 800 MB.

b. Determine how much disk space is available:

```
> df -k /db2fs/data
Filesystem    1024-blocks      Free %Used    Iused %Iused Mounted on
/dev/data     209715200 208236544    1%    4331     2% /db2fs/data
```

The output shows that there are 208,236,544 blocks of free space, each of which is 1024 bytes. This more than meets the disk space requirements.

c. Display the current value of the database configuration parameters that are related to log space (**LOGFILSZ**, **LOGPRIMARY**, and **LOGSECOND**):

```
> db2 get db cfg for dbs | grep '(LOG[FPS]'
 Log file size (4KB)                         (LOGFILSIZ) = 1024
 Number of primary log files                (LOGPRIMARY) = 13
 Number of secondary log files               (LOGSECOND) = 4
```

d. Optional: Increase the log space size. You might want to increase the amount of space available for log files to avoid running out of log space during the upgrade. If you have a large log space or if you are using infinite logging you can skip this step. Otherwise you can increase the number of secondary log files by issuing the following command:

```
> db2 update db cfg for dbname using logsecond value
```

In this scenario *value* is set to (the current value of the **logprimary** configuration parameter + the current value of the **logsecond** configuration parameter) x 2, or 34.

The following command sets *value* to 34:

```
> db2 update db cfg for dbs using logsecond 34
DB20000I  The UPDATE DATABASE CONFIGURATION command completed successfully.
```

You will set the value of **logsecond** back to its original value after you upgrade the database.

6. Perform the following steps to take the database server offline for the upgrade:

a. Stop the DB2 license server by issuing the **db2licd** command.

b. Disconnect all applications by issuing the **quiesce database** command with the **force connections** parameter.

c. Disconnect from the database by issuing the **connect reset** command.

d. Verify that no users are connected to the database by issuing the **list application** command.

e. Stop the database manager by issuing the **db2stop** command.

The following example shows the commands to use:

```
> db2licd -end

> db2 quiesce db immediate force connections
DB20000I  The QUIESCE DATABASE command completed successfully.

> db2 connect reset
DB20000I  The SQL command completed successfully.

> db2 list application
SQL1611W  No data was returned by Database System Monitor.

> db2stop
SQL1064N  DB2STOP processing was successful.
```

Step 7 - Upgrade the instance

To upgrade the instance, perform these steps:

1. Log in as root.

2. From the *DB298DIR*/instance directory, issue the **db2iupgrade** command. In the following example, brahimi is the instance owner:

```
./db2iupgrade -d -k brahimi
DB2 installation is being initialized.

 Total number of tasks to be performed: 7
Total estimated time for all tasks to be performed: 429 second(s)

Task #1 start
Description: Installing or updating DB2 HA scripts for Tivoli SA MP
Estimated time 40 second(s)
Task #1 end

Task #2 start
Description: Installing or updating DB2 Cluster Scripts for GPFS
Estimated time 40 second(s)
Task #2 end

Task #3 start
Description: Setting default global profile registry variables
Estimated time 1 second(s)
Task #3 end

Task #4 start
Description: Register NTP
```

```
Estimated time 40 second(s)
Task #4 end

Task #5 start
Description: Initializing instance list
Estimated time 5 second(s)
Task #5 end

Task #6 start
Description: Configuring DB2 instances
Estimated time 300 second(s)
Task #6 end

Task #7 start
Description: Updating global profile registry
Estimated time 3 second(s)
Task #7 end

The execution completed successfully.

For more information see the DB2 installation log at
"/tmp/db2iupgrade.log.667836".
DBI1446I  The db2iupgrade command is running, please wait.

DBI1070I  Program db2iupgrade completed successfully.
```

Step 8 - Upgrade the database

To upgrade the database, perform the following steps as an instance owner:

1. Issue the **db2start** command.
2. Issue the **upgrade database** command.
3. Issue the **db2stop** command.

The following examples show the results of issuing the commands:

```
> db2start
SQL1063N  DB2START processing was successful.

> db2 upgrade db dbs
DB20000I  The UPGRADE DATABASE command completed successfully.

> db2stop
SQL1064N  DB2STOP processing was successful.
```

Step 9 - Convert the instance to a DB2 pureScale instance type

To convert the instance to a DB2 pureScale instance, perform these steps:

1. Log in as root.
2. From the *DB2I98DIR*/instance directory, issue the **db2iupdt** command, as shown in the following example. The command specifies two hosts for the DB2 pureScale instance: a member (host coralpib127) and a CF (host coralpib129). The hdisk4 disk is used as the tiebreaker disk. The command also specifies the mount point of the GPFS

file system, which was set up when the **db2cluster_prepare** command was issued, in Step 3 - Set up the GPFS file system on page 11.

```
./db2iupdt -d -m coralpib127:coralpib127-ib0 -cf coralpib129:coralpib129-ib0 -
instance_shared_dir /db2sd_20110201154154 -tbdev /dev/hdisk4 brahimi
DBI1446I  The db2iupdt command is running, please wait.

DB2 installation is being initialized.
 Total number of tasks to be performed: 10
Total estimated time for all tasks to be performed: 1069 second(s)

Task #1 start
Description: Installing DB2 files on remote hosts
Estimated time 600 second(s)
Task #1 end

Task #2 start
Description: Installing or updating DB2 HA scripts for Tivoli SA MP
Estimated time 40 second(s)
Task #2 end

Task #3 start
Description: Installing or updating DB2 Cluster Scripts for GPFS
Estimated time 40 second(s)
Task #3 end

Task #4 start
Description: Registering licenses on remote hosts
Estimated time 40 second(s)
Task #4 end

Task #5 start
Description: Setting default global profile registry variables
Estimated time 1 second(s)
Task #5 end

Task #6 start
Description: Register NTP
Estimated time 40 second(s)
Task #6 end

Task #7 start
Description: Initializing instance list
Estimated time 5 second(s)
Task #7 end

Task #8 start
Description: Initiating the remote host list
Estimated time 0 second(s)
Task #8 end

Task #9 start
Description: Configuring DB2 instances
Estimated time 300 second(s)
Task #9 end

Task #10 start
Description: Updating global profile registry
Estimated time 3 second(s)
Task #10 end

The execution completed successfully.
```

```
For more information see the DB2 installation log at
"/tmp/db2iupdt.log.192602".
DBI1070I  Program db2iupdt completed successfully.
```

After you convert the instance to a DB2 pureScale type, the `db2nodes.cfg` file shows the new member and CF:

```
> cat sqllib/db2nodes.cfg
0 coralpib127.torolab.ibm.com 0 coralpib127-ib0 - MEMBER
128 coralpib129.torolab.ibm.com 0 coralpib129-ib0 - CF
```

3. Check the status of the instance by issuing the **`db2instance`** command as root or the instance owner. If you issue the command as root, as shown in the following example, you must specify the **`-instance`** parameter:

```
./db2instance -instance brahimi -list
ID  TYPE   STATE   HOME_HOST   CURRENT_HOST ALERT PARTITION_NUMBER LOGICAL_PORT
NETNAME
--  ----   -----   ---------   ------------ ----- ---------------- ------------ --
-
0   MEMBER STOPPED coralpib127 coralpib127  NO    0                0
coralpib127-ib0
128 CF     STOPPED coralpib129 coralpib129  NO    -                0
coralpib129-ib0

HOSTNAME          STATE       INSTANCE_STOPPED  ALERT
--------          -----       ----------------  -----
coralpib127       ACTIVE      NO                NO
coralpib129       ACTIVE      NO                NO
```

You can display only CFs by specifying the **`-cf`** parameter or only members by specifying the **`-member`** parameter.

4. Verify that the upgrade and conversion to a DB2 pureScale environment was successful by connecting to the database and issuing a simple query, as shown in the following example:

```
> db2 connect to dbs

 Database Connection Information
 Database server        = DB2/AIX64 9.8.3
 SQL authorization ID   = BRAHIMI
 Local database alias   = DBS

> db2 "select count(*) from tab0"

1
-----------
          3
  1 record(s) selected.
```

Step 10 - Perform post-upgrade steps

After the upgrade is complete, review the post-upgrade steps in the DB2 pureScale Feature Information Center and perform any of the steps that apply to your environment. If you changed the setting of the database configuration parameter logsecond in Step 6 - Verify that the database is ready to upgrade on page 21, you should review its value and reset if it necessary.

Refer to the topic Post-upgrade tasks for a DB2 pureScale environment[7] in the DB2 pureScale Feature Information Center for more information on these steps.

Step 11 - Add a member and a CF

To complete the environment in this scenario, add a new member and CF by performing these steps:

1. As the instance owner, stop the instance.

2. As root, add a CF on coralpib130 by issuing the **db2iupdt** command. In the following example, brahimi is the instance owner:

```
./db2iupdt -d -add -cf coralpib130:coralpib130-ib0 brahimi
DBI1446I  The db2iupdt command is running, please wait.
DB2 installation is being initialized.

 Total number of tasks to be performed: 10

Total estimated time for all tasks to be performed: 1069 second(s)
Task #1 start
Description: Installing DB2 files on remote hosts
Estimated time 600 second(s)
Task #1 end

……….

Task #10 start
Description: Updating global profile registry
Estimated time 3 second(s)
Task #10 end

The execution completed successfully.
For more information see the DB2 installation log at
"/tmp/db2iupdt.log.475306".
DBI1070I  Program db2iupdt completed successfully.
```

The `db2nodes.cfg` file shows the new CF:

```
> cat /home/brahimi/sqllib/db2nodes.cfg
0 coralpib127.torolab.ibm.com 0 coralpib127-ib0 - MEMBER
128 coralpib129.torolab.ibm.com 0 coralpib129-ib0 - CF
129 coralpib130.torolab.ibm.com 0 coralpib130-ib0 - CF
```

3. Add a member on coralpib128 by issuing the **db2iupdt** command:

[7] http://publib.boulder.ibm.com/infocenter/db2luw/v9r8/topic/com.ibm.db2.luw.sd.doc/doc/t0056771.html

```
./db2iupdt -d -add -m coralpib128:coralpib128-ib0 brahimi
DBI1446I  The db2iupdt command is running, please wait.

DB2 installation is being initialized.
```

```
 Total number of tasks to be performed: 10
Total estimated time for all tasks to be performed: 1069 second(s)

Task #1 start
Description: Installing DB2 files on remote hosts
Estimated time 600 second(s)
Task #1 end

……..

Task #10 start
Description: Updating global profile registry
Estimated time 3 second(s)
Task #10 end

The execution completed successfully.
For more information see the DB2 installation log at
"/tmp/db2iupdt.log.1253590".
DBI1070I  Program db2iupdt completed successfully.
```

The `db2nodes.cfg` file now shows two members and two CFs:

```
> cat /home/brahimi/sqllib/db2nodes.cfg
0 coralpib127.torolab.ibm.com 0 coralpib127-ib0 - MEMBER
1 coralpib128.torolab.ibm.com 0 coralpib128-ib0 - MEMBER
128 coralpib129.torolab.ibm.com 0 coralpib129-ib0 - CF
129 coralpib130.torolab.ibm.com 0 coralpib130-ib0 - CF
```

4. To see the state of the instance, issue the `db2instance` command:

```
./db2instance -instance brahimi -list
ID  TYPE   STATE   HOME_HOST   CURRENT_HOST ALERT PARTITION_NUMBER LOGICAL_PORT
NETNAME
--  ----   -----   ---------   ------------ ----- ---------------- ------------ --
-
0   MEMBER STOPPED coralpib127 coralpib127  NO    0              0
coralpib127-ib0
1   MEMBER STOPPED coralpib128 coralpib128  NO    0              0
coralpib128-ib0
128 CF     STOPPED coralpib129 coralpib129  NO    -              0
coralpib129-ib0
129 CF     STOPPED coralpib130 coralpib130  NO    -              0
coralpib130-ib0

HOSTNAME          STATE       INSTANCE_STOPPED  ALERT
--------          -----       ----------------  -----
coralpib127       ACTIVE      NO                NO
coralpib128       ACTIVE      NO                NO
coralpib129       ACTIVE      NO                NO
coralpib130       ACTIVE      NO                NO
```

After you add a member, the database is put in backup pending state. You must take a backup before using the database. You can take an offline backup by issuing the **`backup database`** command:

```
> db2 "backup db dbs to /db1/dbs_bkupimg"

  Backup successful. The timestamp for this backup image is : 20110202153821
```

Conclusion

The DB2 pureScale Feature for Enterprise Server Edition provides a database solution that meets the needs of the most demanding customers. By leveraging the cluster caching facility and RDMA technologies, the DB2 pureScale Feature can scale effectively to meet the growing and dynamic needs of different organizations. To meet the demands of peak processing times, you can add members to the DB2 pureScale environment without affecting existing applications. The DB2 pureScale Feature automatically balances the workload across all DB2 members in the cluster without requiring any changes to applications, taking full advantage of the additional processing capacity. If a DB2 member fails, applications are automatically routed among the active members. When the failed member host comes back online, applications are transparently routed to the restarted member.

The DB2 pureScale Feature can provide a lower total cost of ownership compared to that of other solutions, through a simplified deployment and maintenance model. The DB2 pureScale Feature installation process manages the deployment of all the bundled software components to all the hosts in the DB2 pureScale environment and manages the configuration of those components. You can easily monitor and maintain the environment from any of the active members.

Appendix A. Options for getting existing table spaces to be managed by automatic storage

As mentioned in Scenario overview on page 7, two requirements that a database must meet before you can upgrade it to the DB2 pureScale Feature are as follows:

- All table spaces must be managed by automatic storage
- All data and logs must be on General Parallel File System (GPFS).

There are several methods to get existing table spaces to be managed by automatic storage, and you can usually move the data and logs to a GPFS file system as part of this step as well.

The methods that are covered in this document are redirected restore, the **`restore database`** command with the **`transport`** parameter, and the **`db2move`** command. The key questions to answer to choose a method are as follows:

- Does my database contain mostly SMS table spaces? If yes, it might be most efficient to re-create the database regardless of whether the catalog is SMS, because you must re-create all the SMS table spaces. Appendix B on page 33 shows how to re-create the database and use the **`db2look`** and **`db2move`** commands to re-create the database objects and load the data.
- Is my catalog table space SMS? If yes, you must re-create the database from scratch. However, if you have primarily non-SMS user table spaces, creating a new database and using the **`restore database`** command with the **`transport`** parameter to move entire schemas into it might be a good option. Appendix C on page 36 shows how to use this method.
- Is my catalog table space non-SMS, and are my user table spaces mostly non-SMS? If yes, using a redirected restore to convert the table spaces and move to a GPFS file system is a good option. Substeps 5 through 8 in Step 5 - Ensure that the database meets DB2 pureScale requirements, on page 15 shows how to do a redirected restore using a full offline backup. Appendix D on page 40 shows an example of a redirected restore using an online backup.

The decision to use a redirected restore or a restore with the **`transport`** parameter depends primarily on the number of SMS user table spaces, because you will have to drop and recreate these table spaces individually after you restore the database. Figure 4 summarizes these alternatives.

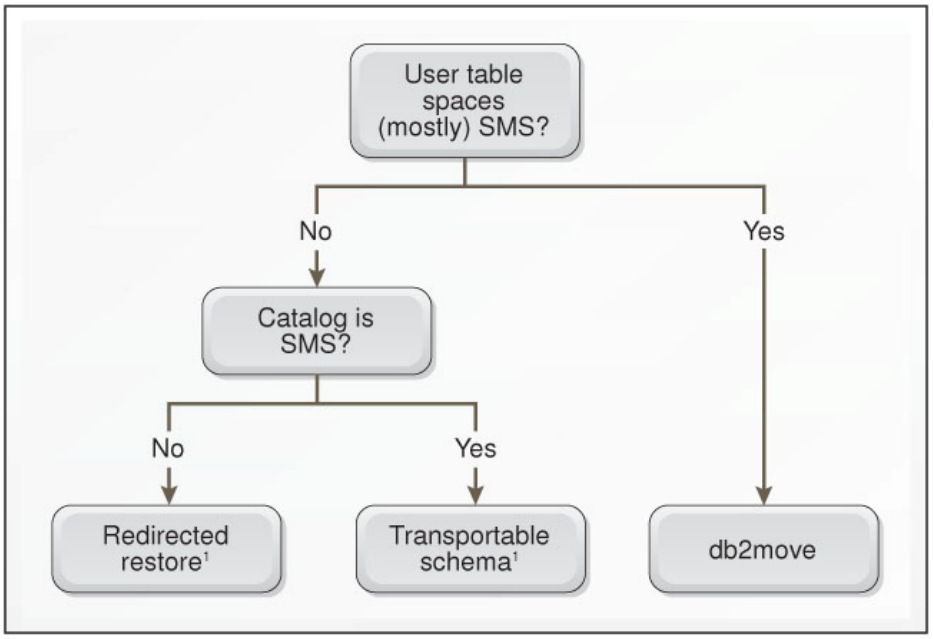

Figure 4. Methods for getting existing table spaces to use automatic storage

Sample commands and output are provided in the following appendixes.

Appendix B. Getting to automatic storage by using the db2move command

Step 5 - Ensure that the database meets DB2 pureScale requirements, on page 15, shows how to convert table spaces to be managed by automatic storage and move data and logs to a GPFS file system. However, you cannot convert SMS table spaces to be managed by automatic storage; you must re-create them. If the system catalog is also SMS, you must re-create the entire database. This appendix shows how to create a new database based on the structure of the old one, with table spaces that are on the GPFS file system and are managed by automatic storage, and how to move the data using the **db2move** command.

The process shown in this appendix replaces substeps 5 through 8 in Step 5 - Ensure that the database meets DB2 pureScale requirements on page 15.

Perform the following steps:

1. If necessary, create the SYSTOOLSPACE table space, which is used by the **db2move** command to store some temporary tables. This table space might not exist because it is not created automatically when you create a database. The table space is created when you use certain DB2 procedures or functions. To create the table space in the source database, which is DBS in this example, connect to the database and invoke the **get_dbsize_info** procedure:

```
> db2 "call GET_DBSIZE_INFO(?, ?, ?, -1)"

  Value of output parameters
  --------------------------
  Parameter Name  : SNAPSHOTTIMESTAMP
  Parameter Value : 2010-10-13-17.03.15.435679

  Parameter Name  : DATABASESIZE
  Parameter Value : 164061184

  Parameter Name  : DATABASECAPACITY
  Parameter Value : 2346512896

  Return Status = 0
```

2. Create the target database on the GPFS file system with default table spaces that are managed by automatic storage, and using a different name than that of the source database:

```
> db2 "create db dbt on /db2fs/data/dbt                    \
       catalog tablespace managed by automatic storage    \
       temporary tablespace managed by automatic storage  \
       user tablespace managed by automatic storage"
DB20000I  The CREATE DATABASE command completed successfully.
```

3. Recreate the objects from the source database in the target database:

a. Generate a file (**dbs_db2look.ddl** in this example) that contains the SQL statements to create the database objects from the source database by issuing the **db2look** command:

```
> db2look -d dbs -o dbs_dblook.ddl -l
-- No userid was specified, db2look tries to use Environment variable USER
-- USER is: BRAHIMI
-- Creating DDL for table(s)
-- Output is sent to file: dbs_dblook.ddl
-- Binding package automatically ...
-- Bind is successful
-- Binding package automatically ...
-- Bind is successful
```

b. In the **dbs_dblook.ddl** file, make the following changes:

- Change all occurrences of **connect to dbs** to **connect to dbt**.
- For each **create tablespace** statement, make the following changes:
  - Change the **managed by** clause (**managed by database** in this example) to **managed by automatic storage**.
  - Delete the **using** clause.

The following example shows the statements that need to be modified in the original file, with the elements to be changed in italics:

```
CONNECT TO DBS;
CREATE REGULAR TABLESPACE "TBSP1" IN DATABASE PARTITION GROUP IBMDEFAULTGROUP
PAGESIZE 8192 MANAGED BY DATABASE
         USING (FILE '/home/brahimi/brahimi/NODE0000/SQL00001/tbsp1_1' 300,
                FILE '/home/brahimi/brahimi/NODE0000/SQL00001/tbsp1_2' 300)
         EXTENTSIZE 10
         PREFETCHSIZE 15
         BUFFERPOOL BUF8
         OVERHEAD 7.500000
         TRANSFERRATE 0.060000
         AUTORESIZE YES
         MAXSIZE NONE
         NO FILE SYSTEM CACHING
         DROPPED TABLE RECOVERY ON;
```

The following example shows the changed elements:

```
CONNECT TO DBT;
CREATE REGULAR TABLESPACE "TBSP1" IN DATABASE PARTITION GROUP IBMDEFAULTGROUP
PAGESIZE 8192 MANAGED BY AUTOMATIC STORAGE
         EXTENTSIZE 10
         PREFETCHSIZE 15
         BUFFERPOOL BUF8
         OVERHEAD 7.500000
         TRANSFERRATE 0.060000
         AUTORESIZE YES
         MAXSIZE NONE
         NO FILE SYSTEM CACHING
         DROPPED TABLE RECOVERY ON;
```

c. Run the statements in the file to create the objects from the source database in the target database:

```
> db2 -tvf dbs_dblook.ddl | tee dbs_dblook.log
```

4. Move the data from DBS to DBT by issuing the **db2move** command:

```
> db2move dbs copy -sn userid -co target_db dbt user userid using password

Application code page not determined, using ANSI codepage 819

*****  DB2MOVE  *****

Action:  COPY

Start time:  Mon Feb 7 11:08:45 2011


All schema names matching:  BRAHIMI;

Connecting to database DBS ... successful!  Server : DB2 Common Server V9.7.2

Copy schema BRAHIMI to BRAHIMI on the target database DBT

Create DMT :  "SYSTOOLS"."DMT_4cb65820c8c6"

Start Load Phase :


db2move finished successfully


Files generated:
-----------------
COPYSCHEMA.20101013210845.msg
LOADTABLE.20101013210845.MSG

Please delete these files when they are no longer needed.


End time:  Mon Feb 7 11:08:57 2011
```

5. Drop the original database:

```
> db2 drop db dbs
DB20000I  The DROP DATABASE command completed successfully.
```

6. Catalog the new database with the name of the original database:

```
> db2 catalog database dbt as dbs
DB20000I  The CATALOG DATABASE command completed successfully.
DB21056W  Directory changes may not be effective until the directory cache is
refreshed.
```

Continue with substep 9 in Step 5 - Ensure that the database meets DB2 pureScale requirements on page 15, which is the step in which you verify that the logs are on a GPFS file system.

Appendix C. Getting to automatic storage by using the `restore database` command with the `transport` parameter

This appendix shows another alternative approach for getting table spaces to be managed by automatic storage and moving data to a GPFS file system. This method is a good option when the catalog table space is SMS but most user table spaces (or all of them, as in the case of the example in this appendix) are DMS.

In this example, you re-create the database on the GPFS file system and use the **`restore database`** command with the **`transport`** parameter to move database schemas to it. The **`transport`** parameter, which was introduced in DB2 V9.7 Fix Pack 2, restores a set of database schemas from a database backup image to a different, existing database. The database objects in the transported schema are re-created in the target database, and the data is restored into that database.

The process shown in this appendix replaces substeps 5 through 8 in Step 5 - Ensure that the database meets DB2 pureScale requirements on page 15.

Perform these steps:

1. Extract buffer pool information for later use. The **`transport`** parameter does not re-create the buffer pools that are associated with the table spaces, so you must create any non-default buffer pools in the target database. Also, you must ensure that the default buffer pool has the same characteristics in the source (original) and target databases. After connecting to the DBS database, issue a query to retrieve details about the buffer pools:

```
> db2 connect to dbs

 Database Connection Information

 Database server        = DB2/AIX64 9.7.2
 SQL authorization ID   = BRAHIMI
 Local database alias   = DBS

> db2 "select substr(b.bpname,1,15) as bp,              \
              substr(t.tbspace,1,15) as tbsp,           \
              b.NPAGES, b.PAGESIZE,                     \
       from   syscat.tablespaces t, syscat.bufferpools b \
              where t.bufferpoolid = b.bufferpoolid"

BP              TBSP            NPAGES      PAGESIZE
--------------- --------------- ----------- -----------
IBMDEFAULTBP    SYSCATSPACE              -2        4096
IBMDEFAULTBP    USERSPACE1               -2        4096
IBMDEFAULTBP    TEMPSPACE1               -2        4096
BUF             TBSP                     -2        8192
```

```
  4 record(s) selected.
```

2. List the schemas in the source database for later use, for example, by running the following query:

```
> db2 "select substr(SCHEMANAME,1,20) as schema, substr(definer,1,20) as definer
from syscat.SCHEMATA where SCHEMANAME not in ('NULLID', 'SQLJ', 'SYSTOOLS') and
DEFINER = 'BRAHIMI'"

SCHEMA               DEFINER
-------------------- --------------------
BRAHIMI2             BRAHIMI
BRAHIMI              BRAHIMI

  2 record(s) selected.
```

3. Back up the source database, as shown in substeps 5 and 6 from Step 5 - Ensure that the database meets DB2 pureScale requirements. This backup will be used as the source for the transport of the database schemas.

4. Optional: Copy or move any log files in the active log path. Dropping the database in the next step deletes any log files in the active log path. If you are using log retention or log archiving and want to be able to restore from an older backup image and roll forward to a point in time before the upgrade, you will require these logs.

5. Optional: Drop the source database so that you can create a target database with the same database name. If you do not want to drop the source database before creating the target database, you can drop the source database after creating the target one and then catalog the target database with the source database name, as shown in Step 6 in Appendix B.

```
> db2 drop db dbs
DB20000I  The DROP DATABASE command completed successfully
```

6. Create the target database on the GPFS file system with automatic storage default table spaces:

```
> db2 "create db dbs on /db2fs/data/dbs                    \
       catalog tablespace managed by automatic storage    \
       temporary tablespace managed by automatic storage  \
       user tablespace managed by automatic storage"
DB20000I  The CREATE DATABASE command completed successfully.
```

7. At this point the target database contains only one buffer pool, IBMDEFAULTBP. Create buffer pool BUF in the target database with the same characteristics as buffer pool BUF in the source database. To do this, connect to the target database and issue the **`create bufferpool`** statement, using the output from step 1:

```
> db2 “create bufferpool buf size automatic pagesize 8K”
```

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

8. Verify that the definition of IBMDEFAULTBP in the target database is the same as the definition of that buffer pool from the source database, using the output from step 1. In this scenario, the buffer pools are the same. If they were not, you could use the **alter bufferpool** statement to change the definition of IBMDEFAULTBP.

9. Ensure that the target database is recoverable. Typically you use the same type of logging on the target database as you did on the source database. However, if the source database is not recoverable, you must make the target database recoverable, at least temporarily, to do the transport.

   In this source database logs were archived to disk, so set up the same method on the target database:

   a. Check the current value of the **logarchmeth1** database configuration parameter:

```
> db2 get db cfg for dbs | grep LOGARCHMETH1
 First log archive method           (LOGARCHMETH1) = OFF
```

   b. Update the value of the **logarchmeth1** database configuration parameter using the **update database configuration command**, specifying that log files will be archived to disk and providing a fully qualified, existing path name:

```
> db2 update db cfg for dbs using logarchmeth1 disk:/db2fs/log
DB20000I  The UPDATE DATABASE CONFIGURATION command completed successfully.
```

   c. Back up the database:

```
> db2 “backup db dbs to /db1/dbs_bkupimg2“
Backup successful. The timestamp for this backup image is : 20110208101017
```

10. Restore the backup of the source database taken in Step 3 on page 37, by issuing the **restore** command and specifying the **tablespace, schema, transport,** and **redirect** parameters. By specifying the **redirect** parameter, you can use the **set tablespace containers** command to convert the DMS user table space to be managed by automatic storage. The example transports the schemas brahimi and brahimi2, which contain the table space TBSP:

```
> db2 “restore db dbs                \
  tablespace (tbsp)                  \
  schema (brahimi, brahimi2)                    \
  from /db1/dbs_bkupimg       \
  transport into dbs                 \
  redirect”
SQL1277W  A redirected restore operation is being performed. Table space
configuration can now be viewed and table spaces that do not use automatic
storage can have their containers reconfigured.
DB20000I  The RESTORE DATABASE command completed successfully.
```

```
> db2 “set tablespace containers for 3 using automatic storage”
DB20000I  The SET TABLESPACE CONTAINERS command completed successfully.

> db2 “restore db dbs continue”
DB20000I  The RESTORE DATABASE command completed successfully.
```

Continue with substep 9 in Step 5 - Ensure that the database meets DB2 pureScale requirements on page 15, which is the step in which you verify that the logs are on a GPFS file system.

Appendix D. Getting to automatic storage by using redirected restore from an online backup

This appendix shows the third alternative approach for getting table spaces to be managed by automatic storage and moving data to the GPFS file system. In this example, the catalog and user table spaces are DMS, but they are not managed by automatic storage. Like the example in Step 5 - Ensure that the database meets DB2 pureScale requirements on page 15, this approach uses a redirected restore, but in this example an online backup is used to minimize the outage on the source database. This method is applicable only if the database is recoverable, that is, only if the database is not using circular logging. Another difference in this example is that in the source database, the temporary table space TEMPSPACE1 is SMS. Therefore, you can use the **SET TABLESPACE CONTAINERS** command to convert TEMPSPACE1 to automatic storage during the redirected restore instead of dropping and re-creating it afterward.

The process shown in this appendix replaces substeps 5 through 8 in Step 5 - Ensure that the database meets DB2 pureScale requirements on page 15.

Perform these steps:

1. Take an online backup of the database, including the logs:

```
> db2 “backup db dbs online to /db1/dbs_bkupimg include logs”
  Backup successful. The timestamp for this backup image is : 20110209132805
```

2. When you are ready to move the database to the GPFS file system, perform the following steps:
   a. Limit access to the database by issuing the **quiesce database** command with the **force connections** parameter.
   b. Disconnect from the database by issuing the **connect reset** command.
   c. Use the **list application** command to verify that no users are connected to the database.

   The following example shows the commands to use:

```
> db2 quiesce db immediate force connections
DB20000I  The QUIESCE DATABASE command completed successfully.

>Wait few seconds here<

> db2 connect reset
  DB20000I  The SQL command completed successfully.

> db2 list application
  SQL1611W  No data was returned by Database System Monitor.
```

3.

4. Save the logs of the source database, for use when rolling forward the target database:

```
> cp /db1/dblog/DBS/NODE0000/C0000000/* /db1/save_dbs_logs
```

5. Drop the source database:

```
> db2 drop db newdbs
DB20000I  The DROP DATABASE command completed successfully.
```

6. Perform the redirected restore from the online backup:

```
> db2 “restore db dbs                  \
       from /db1/dbs_bkupimg           \
       on /db2fs/data/dbs              \
       dbpath /db2fs/data/dbs          \
       into dbs                        \
       newlogpath /db2fs/log           \
       redirect                        \
       without prompting”
SQL1277W  A redirected restore operation is being performed. Table space
configuration can now be viewed and table spaces that do not use automatic
storage can have their containers reconfigured.
DB20000I  The RESTORE DATABASE command completed successfully.

> db2 “set tablespace containers for 0 using automatic storage”
DB20000I  The SET TABLESPACE CONTAINERS command completed successfully.

> db2 “set tablespace containers for 1 using automatic storage”
DB20000I  The SET TABLESPACE CONTAINERS command completed successfully.

> db2 “set tablespace containers for 2 using automatic storage”
DB20000I  The SET TABLESPACE CONTAINERS command completed successfully.

> db2 “set tablespace containers for 3 using automatic storage”
DB20000I  The SET TABLESPACE CONTAINERS command completed successfully.

> db2 “restore db dbs continue”
DB20000I  The RESTORE DATABASE command completed successfully.
```

7. Apply the logs that were generated after the online backup was taken by issuing the **`rollforward database`** command, using the folder into which you saved the logs in a previous step:

```
> db2 “rollforward db dbs to end of logs and stop              \
       overflow log path ( /db1/save_dbs_logs )”
```

Continue with substep 9 in Step 5 - Ensure that the database meets DB2 pureScale requirements, which is the step in which you verify that the logs are on a GPFS file system.

Appendix E. LPAR configuration details

The following table shows the configurations used for the LPARs in this scenario.

Host name	coralpib127	coralpib129	coralpib128	coralpib130
Operating-system level	AIX® 6.1 TL3 SP3	AIX 6.1 TL3 SP3	AIX 6.1 TL3 SP3	AIX 6.1 TL3 SP3
Server type	Member 0	Primary CF	Member 1	Secondary CF
Cores				
RAM	8 GB	8 GB	8 GB	8 GB
Shared disks	hdisk2 – 204 GB used as GPFS file system hdisk3 – 10 GB used for the sqllib_shared directory hdisk4 – 128 MB used as tiebreaker hdisk6 – 204 GB used as GPFS file system			
Disk device driver	MPIO (6.1.4.2)	MPIO (6.1.4.2)	MPIO (6.1.4.2)	MPIO (6.1.4.2)
Ethernet interface	en0	en0	en0	en0
InfiniBand host name	coralpib127-ib0	coralpib129-ib0	coralpib128-ib0	coralpib130-ib0
InfiniBand interface	ib0	ib0	ib0	ib0
OpenSSH	4.5.0.5302	4.5.0.5302	4.5.0.5302	4.5.0.5302
Firmware	IBM,EL350_039		IBM,EL350_039	

Table 1, LPAR configuration details

Installation prerequisites for DB2 pureScale Feature (AIX)[8], in the DB2 pureScale Feature Information Center, details the requirements for the DB2 pureScale Feature for the AIX operating system.

Installation prerequisites for DB2 pureScale Feature (Linux)[9], in the DB2 pureScale Feature Information Center, details the requirements for the DB2 pureScale Feature for the Linux operating system.

[8] http://publib.boulder.ibm.com/infocenter/db2luw/v9r8/topic/com.ibm.db2.luw.sd.doc/doc/r0054850.html
[9] http://publib.boulder.ibm.com/infocenter/db2luw/v9r8/topic/com.ibm.db2.luw.sd.doc/doc/r0057441.html

Contributors

Paul Dermody
DB2 Information Development

Deepen Manek
Manager, DB2 System Test

Jessica Rockwood
DB2 Performance Benchmarks and Solutions Development

Kelly Schlamb
DB2 pureScale Specialist & Business Partner Outreach

Yvonne Chan
Principal, Information Management Technology Ecosystem, DB2 pureScale

Frank Ning
Manager, DB2 Install and Up & Running

Aimin Wu
IBM System Optimization Competency Center

Quentin Presley
DB2 Software Development

David Sciaraffa
DB2 Software Development

Wojciech Kuczynski
DB2 Software Development

Serge Boivin
DB2 Information Development

Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. The sample programs are provided "AS IS", without warranty of any kind. IBM shall not be liable for any damages arising out of your use of the sample programs.

Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. If these and other IBM trademarked terms are marked on their first occurrence in this information with a trademark symbol (® or ™), these symbols indicate U.S. registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. A current list of IBM trademarks is available on the Web at “Copyright and trademark information” at www.ibm.com/legal/copytrade.shtml

Windows is a trademark of Microsoft Corporation in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

Other company, product, or service names may be trademarks or service marks of others.