DB2 9 for UNIX®: Integrating with a NetApp Storage System

Jawahar Lal and Roger Sanders, NetApp, Inc.

November 6, 2006,

TR 3531

Executive Summary

This document describes the steps carried out to configure and install a DB2 database in a NetApp UNIX environment. It particularly elaborates the configuration steps for the UNIX host as well as the NetApp storage system. The UNIX host types covered in this document includes Linux®, IBM AIX, Sun™ Solaris™, and HP-UX.

Table of Contents

1. Purpose and Scope

This document describes the steps necessary to integrate a DB2 9 for UNIX with a NetApp™ storage system. Specifically, we cover the following topics:

- Preparing an UNIX server for DB2 installation
- Preparing a storage system for DB2 database storage
- Installing DB2 on the UNIX server and creating a database on the storage system
- Separating database transaction log files from data files and storing on the storage system
- Migrating an existing database from a local disk to a storage system

2. Assumptions

This document describes the deployment of DB2 Enterprise Server Edition (ESE) V9 into an already functioning UNIX and storage system environment. Therefore, it is assumed that you are familiar with DB2 9 and the operation of storage systems. It is also assumed that you possess basic UNIX administration skills. The examples provided may require modifications before they can be run under your version of UNIX.

It is recommended that you read DB2 9 installation documentation for your particular operating system and follow its steps to install and configure the DB2 instance and database. Where the DB2 documentation and this technical report contradict, you should assume that the DB2 documentation is correct. Please inform NetApp of any such contradictions found so this document can be corrected.

The sample scripts in this technical report assume the following:

- The name of the storage system used is `netapp`
- The name of the UNIX host, henceforth known as a database server, is `db2srv1`
- The user name for DB2 instance owner is `db2inst1`
- The DB2 Instance on the UNIX server is `db2`
- The home directory for the DB2 Instance is `/home/db2inst1`
- Database data files are stored on FlexVol volume named `dbdata` storage system is netapp`:/vol/dbdata`
- The location of database transactions log files on the storage system is netapp`:/vol/dblogs`
- The mount point on the UNIX server for the volume is **`netapp:/vol/dbdata`** is /mnt**/**`dbdata`
- The mount point on the UNIX server for the volume is `netapp:/vol/dblogs` is `/mnt/dblogs`

You may need to make the appropriate changes to one or more of these settings in order to make these scripts work in your environment.

Through out this document term DB2 Enterprise Server Edition V9 and DB2 9 are used interchangeably and means same thing unless otherwise mentioned..

3. Infrastructure

In order to use DB2 9 for UNIX in conjunction with a NetApp storage system, the following elements are required:

- DB2 Enterprise Server Edition V9.
- One or more UNIX Host servers (database servers) with DB2 administrator and user accounts
- One or more NetApp storage systems (F800 or later series), henceforth known as storage system, running Data ONTAP® 7G (or later)
- Network connectivity between database server and storage system
- FCP or iSCSI HBA (not required when NFS or an iSCSI software initiator is used)

3.1. DB2 Enterprise Server Edition V9

In order to perform the steps outlined in this document, you need to have DB2 Enterprise Server Edition (ESE) V9 installed and running on a UNIX server. Before proceeding with installation of DB2 ESE, check the system requirements described in the DB2 installation documentation for your host platform.

3.2. Database Server with DB2 Administrator and User Accounts

In order to perform the steps outlined in this document, you will need to have both a DB2 administrator and one or more DB2 user accounts on the UNIX server. If you already have DB2 installed on your UNIX server, chances are that the appropriate DB2 user accounts already exist. If however, you are installing DB2 for the first time, a DB2 administrator and one or more user accounts can be created as part of the DB2 installation process. Refer to the DB2 documentation for detailed instructions on setting up the appropriate accounts during the installation process. Database server configuration is also covered in a later section of this document.

3.3. NetApp storage system

DB2 stores data in tablespace containers. A tablespace container, which can be a file or a raw device, can physically reside on a NetApp storage system. Any NetApp storage system (F800 or later series) running a supported version of Data ONTAP can be used to store a DB2 database and its transaction log files. In order to produce this document we used a NetApp FAS900 series storage system running Data ONTAP 7G.

3.4. Network

In order to store a DB2 database on a NetApp storage system, first you must establish a network connection between the database server and the storage system. The type of connection you establish is dependant upon the type of environment you are using. If your environment is SAN with Fibre Channel (FC) then you will need to establish a FC connection between the database server and the storage system; otherwise, an ethernet connection will work for both IP SAN and NAS. We used two Gigabit Ethernet connections between our AIX server and the storage system to develop this document (connections were made using two cross-over fiber optic cables). For optimal performance, it is strongly recommended that you use Gigabit Ethernet interfaces to support your DB2 and storage system database environment.

Some network performance notes:

- Flow control and full-duplex operation should be enabled on the Ethernet interfaces on *both* the UNIX server and the storage system.
- When using Gigabit Ethernet adapters, consider turning on jumbo frames.
- Consider using local locking for all mount points that communicate with the storage system.
- If possible, dedicate at least one private network for the connection between the DB2 server and the storage system. This can be done using a switch or a crossover cable on most networking topologies (we used Gigabit Ethernet and cross over cable to develop this document).
- To increase the occurrence of parallel I/O between the database server and the storage system, create separate volumes on the storage system for database data and transaction logs and use separate mount points to mount them to the database server**.**
- Dedicated or private network connections between the DB2 server and the storage system are recommended for the following reasons:
    - Any issues of contention or latency are eliminated if the DB2 server and the storage system are the only two nodes in the network.
    - Creating a private network connection ensures security. There is no need to protect DB2-specific files from tampering, as would be the case if a shared network were used.
    - As additional storage systems are added to the network to increase storage capacity, they will adhere to the same security levels as the database server and the existing storage system do.
- Each dedicated network connection, using a high-end networking protocol such as Gigabit Ethernet, provides a large bandwidth between the storage system and the database server. These connections are identical to SCSI or FC-AL I/O connections, except they use the NFS service instead of local I/O services.

4. Configuration Overview

Traditional installation of DB2 on a UNIX platform calls for the storage of databases data and corresponding transaction log files on local disk(s). When a NetApp storage system is used, a Database Administrator can take full advantage of NetApp technologies such as Snapshot™, SnapMirror®, RAID-DP™, FlexVol, and

FlexClone™ by placing the database's overhead files, data, and transaction log files on a NetApp storage system.

4.1. Designing a database with recovery in mind

When designing a new database, it is important to think about recovery needs from the start. As part of the database design process, you should identify any relationships that exist between the various database objects. These relationships can be at an application level, where transactions can work with one or more tables, as well as at a database level, where referential integrity constraints can exist between tables, or where events against one table can activate triggers that affect other tables. With these relationships in mind, you can group related database objects together on the same logical FlexVol volume or group of volumes. Placing database objects with similar backup requirements or functions on the same logical volume(s) will simplify the use of Snapshot copies and typically make recovery that much easier. In addition, it goes without saying that the archive logs should be stored on a separate volume from that on which the data is stored. A documented model of your database design can be a tremendous benefit when making these kinds of decisions.

4.2. Aggregate and Volume Configuration

NetApp Data ONTAP 7G supports a new virtual storage layer called flexible volumes, hereafter known as FlexVol™ volume or volume. A FlexVol volume is created within an aggregate and is loosely coupled with its containing aggregate. The FlexVol volume provides greater performance than traditional volumes and can grow and shrink as needed by executing one simple command. Data ONTAP makes it easy to control the placement of related database objects on a FlexVol volume. For more information on creating and managing FlexVol volumes, refer to the vol command in the appropriate NetApp documentation available online at the NetApp NOW™ Web site.

A key to ease of use and manageability of a DB2 database stored on a NetApp storage system is the notion that the entire database can be stored on one or more FlexVol volumes. This configuration requires minimal attention by DBAs and System Administrators – allowing the storage system to manage the physical data storage ensures high performance and availability. However, there are a few storage system physical design considerations that need to be taken into consideration to ensure that these benefits are not compromised:

- The root volume should be its own volume. The root volume can be a traditional volume or a FlexVol volume.
- It is strongly recommended that all the database object files be stored on a volume on the NetApp storage system that is not the root volume.
- It is strongly recommended that the database data and transaction logs (archive as well as active) be kept on separate FlexVol volumes. This allows the database to be backed up on a different schedule from the logs. This helps decrease the loss of data after a database failure, due to the availability of more current logs. If the transaction logs and the database reside on the same FlexVol volume, the recovery of the volume would cause in the active and archive logs to be overwritten, resulting in a loss of all log data that was generated after the Snapshot copy was created. As a result, no transaction log data would exist for a roll-forward recovery operation.
- A large aggregate delivers better performance, therefore, used large aggregate whenever possible. Multiple FlexVol volumes required for a database can be created within a single large aggregate.
- Over committed aggregates is not recommended for the database storage.

NOTE: Throughout this document, the terms volume and FlexVol volume are interchangeable unless otherwise mentioned.

5. Advantages of Storing DB2 Databases on a NetApp Storage system

Running a DB2 database with data and transaction log files stored on a NetApp storage system has several advantages:

- **Extremely Fast Backup:** Backup performance can be significantly improved using Snapshot copies in conjunction with DB2's `set write suspend` and `set write resume` commands. Snapshot copies can be created in a matter of seconds, regardless of the size of the database or the level of activity on the NetApp storage system used. This reduces the database backup window from hours to seconds and allows DBAs to take frequent full database backups without having to take the database offline.
- **Quick Recovery**: Using the Data ONTAP SnapRestore® command, an entire database can be restored in a matter of seconds. Since there is no data copying involved, an incredible amount of time is saved as the file system is put back to the original state it was in at the time the Snapshot was created. Data ONTAP 7G supports 255 Snapshot copies per WAFL® volume. The ability to store a large number of low-impact, frequently created Snapshot copies brings the time needed to perform a roll-forward recovery operation down to minutes or seconds and in many circumstances it allows the DBA to restore the database immediately without the necessity to restore from tape.
- **High Availability:** The need for 24x7availability is fast becoming a reality for organizations of all sizes. Organizations cannot tolerate scheduled downtime nor can they afford extended periods of slow system response caused by traditional database backup methods. NetApp Snapshot technology that can create database backup in matter of seconds without bringing a database down can be used as complementary technology to ensure higher system uptime.
- **High Reliability:** The RAID architecture used for NetApp storage systems is unique and provides greater reliability than many traditional RAID implementations. If a disk in a NetApp RAID group fails, it is reconstructed automatically without any user intervention. Additionally, NetApp supports RAID-DP architecture. RAID-DP is considered approximately 4000 times more reliable than traditional RAID. For more details on RAID-DP please read NetApp Data Protection: Double Parity RAID for Enhanced Data Protection with RAID-DP.
- **No Impact on System Response Time:** Because a Snapshot copy is simply a picture of the file system at a specific point-in-time and creating a database backup using Snapshot doesn't involve actual data I/O, the process of backing up a database virtually have no performance impact on system response time.
- **Minimum Storage requirement**: Two Snapshot copies created in sequence differ from each other by the blocks added or changed in the time interval between the two. This block-incremental behavior limits associated storage capacity consumption.
- **Tablespace container Load Balance:** Many of the tasks associated with balancing the load between multiple tablespace containers can be eliminated. Because of the high performance of the NetApp storage system Data ONTAP operating system, only one container needs to be defined for each tablespace used.

6. Configuring the Database Server and the Storage System

This section describes the steps involved in installing the DB2 9 software on a database server and creating a DB2 database on a NetApp storage system. (*Note*: The items described in section 3. Infrastructure must exist in order for this information to be valid.) Before you begin the installation process, you must ensure that the appropriate operating system kernel updates have been applied, and all the kernel parameters have been updated as per the installation documentation. Additional information on the DB2 installation procedure can be found at the IBM DB2 Web site in the manual*, "Quick Beginnings for DB2 Servers."*

Before you create a database on a storage system, you need to complete a series of steps to configure your environment. These steps can be broken down into two sets: one set of steps must be performed on the NetApp storage system while the other needs to be performed on the database server.

6.1. Configuring the NetApp storage system:

(1). Setup and install Data ONTAP 7.0

Install Data ONTAP 7G on your NetApp storage system if it is not already installed. For Data ONTAP installation and setup instructions please refer to "Software Setup guide" on the NOW Web site.

(2). Activate the appropriate license keys

On the NetApp storage system file access protocols FCP, iSCSI, NFS, and CIFS are licensed services. You need to enable the appropriate service by activating license keys for the protocol you intend to use. The License keys can be activated by executing the following command from the NetApp storage system:

license add [*LicenseCode*]

Where

- `LicenseCode` identifies the product license key you obtained from NetApp.

Note: Parameters shown in angle brackets (`< >`) are optional; parameters or options shown in square brackets (`[ ]`) are required; option shown in curly brackets (`{}`) are mutually exclusive and one value must be selected; a comma followed by ellipses (…) indicates that the preceding parameter can be repeated multiple times

For example, to activate the license code `123XYZABCD` obtained from NetApp for NFS; you would execute the following command on the storage system:

```
license add 123XYZABCD
```

(3). Update the `/etc/host` file on the storage system used

The storage system must be able to communicate with the database server and vice versa. The storage system can communicate to the database server if there is an entry in its `/etc/hosts` file for the database server or alternatively, if it uses some other host name resolution techniques like NIS or DNS. By default, the `/etc/hosts` file is checked first for host name resolution. The easiest way to update the `/etc/hosts` file on the storage system is by using FilerView®. Entries made in the `/etc/hosts` file should look similar to the following:

[*HostIP*] [*HostName*]

Where

- `HostIP` identifies the IP address assigned to the database server.
- `HostName` identifies the network name of the database server.

For example, to add an entry for a database server named `db2srv1` and that has IP address `172.17.32.112`, you would add the following line to the `/etc/hosts` file on the storage system:

172.17.32.112 db2srvr1

(4). Enable '`rsh`' access for the database server

In order to use the '`rsh`' (remote shell) command from the database server you need to perform two steps. First, enable the '`rsh`' option on the storage system by executing the following command:

```
options rsh.enable on
```

Then, add the database host and user name entry to the `/etc/hosts.equiv` file found on the storage system. The entry to this file looks somewhat similar to the following:

[*HostIP*] [*UserName*]

Where

- `HostIP` identifies the IP address assigned to the database server.
- `UserName` identifies the name of the user who has been granted `rsh` access to the storage system

For example, to allow '`rsh`' command execution from a database server named `db2srv1` by a user named `db2inst1`, you would add the following line to the `/etc/hosts.equiv` file on the storage system:

```
db2srvr1 db2inst1
```

(5). Create space to store data and transaction log files

In order to create a database on the NetApp storage system, you need to create aggregates, flexible volumes and, in some cases, LUNs. An aggregate is a physical pool of storage at RAID level and is created using the following command:

aggr create [*AggrName*] –f –t {raid4 | raid_dp} –r [*RaidSize*] [*nDisk*]@[*DiskSize*] | –d <disk1, disk2,..,diskn>

Where

- `AggrName` identifies the name assigned to the aggregate.
- `RaidSize` identifies the size of the raid group that will be created by the command implicitly.
- `nDisk` identifies the number of disks that are going to be used for the aggregate.
- `DiskSize` identifies the size of the disks that are being used for the aggregate.

For example, to create an aggregate named `db2aggr1` that has eight 72 Gig disks in it and that uses RAID-DP, you would execute the following command on the storage system:

```
aggr create db2aggr1 –r 8 8@72g –t raid_dp
```

A FlexVol volume is logical storage container inside an aggregate. A FlexVol volume can be as small as a few megabytes and as large as the aggregate itself. A FlexVol volume can be created using the following command:

vol create [*VolName*] [*AggrName*] [*VolSize*]

Where

- `VolName` identifies the name assigned to the FlexVol volume that is being created.
- `AggrName` identifies the name assigned to the aggregate.
- `VolSize` identifies the size of the volume in terms of MB,GB,TB etc

For example, to create a flexible volume named `dbdata` that is 100GB in size and that resides in the aggregate named `db2agg1`, you would execute the following command on the storage system:

```
vol create dbdata  db2agg1  100GB
```

LUN is an acronym for Logical Unit Number. One or more LUNs can be created within a volume. LUNs are used to map storage system in SAN environments. A LUN is created by executing the following command on a storage system:

lun create –s [*LunSize*] –t [*OSType*] [*LunPath*]

Where

- `LunSize` identifies the size of the LUN that is being created.
- `OSType` identifies the type of operating system the LUN is created for.
- `LunPath` identifies the path for LUN.

For example, to create a LUN named `/vol/dbdata/lun1` that is 50GB in size and resides in a FlexVol volume named `dbdata`, you would execute the following command on the storage system:

```
lun create –s 50G –t linux  /vol/dbdata/lun1
```

It is important to note that the LUN created in the example above will be accessed by a Linux® host.

(6). Specify the volume security settings to use

You must ensure that the volume(s) on the storage system that are to be used for DB2 database support either the UNIX or the mixed security style. The security setting of a volume can be set using the following command:

qtree security [*VolName*] { unix | mixed | NTFS }

Where

- `VolName` identifies the name assigned to the volume.

For example, to change the security style to UNIX of a FlexVol volume named `dbdata,` you would execute the following command on the storage system:

```
qtree security /vol/dbdata  unix
```

Repeat this step and change the security style for all the volumes to be used for the database.

(7). Disable the automatic Snapshot feature

Normally, a database is backed up based on a user defined schedule. Therefore, it is recommended that you turn off the automatic Snapshot feature for any volume to be used for the database and its transaction log files. The automatic Snapshot feature can be turned off using the following command:

vol options [*VolName*] nosnap on

Where

- `VolName` identifies the name assigned to the volume.

For example, to turn the automatic Snapshot feature off for a volume named dbdata, you would execute the following command on the storage system:

```
vol options dbdata nosnap on
```

Repeat this step and turn the auto Snapshot off for all the volumes that are to be used for the database.

(8). Export the volumes if NFS is used (Skip this step if FCP or iSCSI is used)

In order to access the volumes on a storage system using the NFS protocol, you must export them. A volume can be exported by adding an entry to the `/etc/exports` file found on the storage system and giving the database server appropriate access to it. You can use FilerView or the `exportfs` command for this purpose. NFS exports are managed using the following command:

exportfs –p rw=[*HostName*],root=[*HostName*] [*PathName*]

Where

- *`HostName`* identifies the name assigned to the database server.
- *`PathName`* identifies the path for the volume

For example, to create an export entry for a volume named `dbdata` and allow `root` access to it from a database server named `db2srv1`, you would execute the following command on the storage system:

```
exportfs –p rw=db2srv1,root=db2srv1 /vol/dbdata
```

Repeat this step and create an export entry for all the volumes to be used for the database. After creating the export entries use following command to refresh the exports on the storage system:

```
exportfs –all
```

(9). Set up Initiators and LUNs if FCP or iSCSI is used (Skip this step if NFS is used)

In a SAN or IP SAN environment, UNIX hosts serve as initiators and the storage systems serve as targets. The logical storage layer created on a storage system is referred as a LUN. Fibre Channel SAN and IP SAN (iSCSI) are block based access protocols that access LUNs on a storage system using a hardware or software initiator installed on the database server. The initiator has a unique network identification know as World Wide Port Number (`WWPN`) or World Wide Node Name (`WWNN`).

a). Create an Initiator Group (igroup) for the database server

Before attempting to create an igroup, you need to find the `WWPN` or `WWNN` for the initiator. In order to determine `WWPN`/`WWNN`, refer to *section 6.2 Configuring Database Host Server in SAN environment* of this document. Use the `WWPN` or `WWNN` to create an igroup on the storage system by executing the following command:

igroup create –{ f | I } –t { linux | aix |solaris | hpux } [*iGroupName*] [*NodeIdentifier*]

Where

- `iGroupName` identifies the name assigned to the initiator group being created.
- `NodeIdentifier` identifies the initiator's `WWPN` or `WWNN`.

For example, to create an igroup named `db2srv1_igp` for an initiator on a Linux host that has a `WWPN` value of `10000000c93a069b`, you would execute the following command on the storage system:

```
igroup create –f –t linux db2srv1_igp 10000000c93a069b
```

b). Create appropriate LUNs and LUN Mappings

LUNs are used by initiators to identify SCSI devices on the target storage system. A LUN is created within a volume or qtree by executing the following command:

lun create –s size { k|m|g|t } –t { linux | aix | solaris | hpux} [*LunPath*]

Where

- ***LunPath*** identifies the path assigned to the lun being created.

For example, to create a 75GB LUN within a FlexVol volume named dbdata, you would execute the following command on the storage system:

```
lun create –s 75G –t linux /vol/dbdata/data1
```

In order to make LUNs accessible to a host server you must map them to an igroup, which is associated to an initiator installed on the host. A LUN mapping can be created by executing the following command:

lun map [*LunPath*] [*iGroupName*] [*LunID*]

Where

- `LunPath` identifies the path assigned to the lun being created.
- `iGroupName` identifies the name assigned to the initiator group.
- `LunID` identifies a numeric id using that LUN is mapped to an igroup.

For example, to create a LUN mapping for a LUN named `/vol/dbdata/data1` to an igroup named `db2srv1_fcp_igp`, you would execute the following command on the storage system:

```
lun map /vol/dbdata/data1  db2srv1_fcp_igp 0
```

c). Obtain a node name for the storage system used

Target storage systems are identified by a unique node name that is used for creating persistent binding for the storage devices on the host. You can find out the node name assigned to a storage system by executing the following command:

{ fcp | iscsi } NodeName

For example, to find the node name assigned to an iSCSI target, you would execute the following command on the storage system:

```
iscsi nodename
```

6.2. Configuring Database Server in NAS environment

After creating volumes on the storage system and updating the `/etc/exports/` file, the database server must be configured to make use of these volumes. The steps carried out to configure the database server to use the storage system volumes over NFS depend on the UNIX flavor. Following are the basic steps needed to configure the database server for most popular UNIX flavors:

(1). Log in to the database server as the `root` user.

(2). Add the IP address and name of the storage system to the `/etc/hosts` file found on the database server. The entry in the `/etc/hosts` file should look somewhat similar to the following:

[*StorageSystemIP*] [*StorageSystemName*]

Where
- `StorageSystemIP` identifies the network IP address assigned to the storage system.
- `StorageSystemName` identifies the name assigned to the storage system.

For example, to add a storage system named `netapp` that has the IP Address `10.60.175.57` to the `/etc/hosts` file found on a database server, you would enter the following line:

10.60.175.57 netapp

(3). In order to install and work with DB2 database you need to create database user groups and user accounts. User and group authentication is managed in a facility external to DB2, such as the operating system, a domain controller, or a Kerberos security system.

To produce this document, we used the operating system's authentication mechanism and we created a group named `db2adm` and a user account named `db2inst1`. This was done by executing the following commands on the database server:

```
groupadd -g 700 db2adm          # group of users to be granted SYSDBA authority
useradd -c "DB2 9 Instance Owner" -u 701 -g db2adm -G db2adm -d
/home/db2inst1 db2inst1 -p db2inst1
```

Additional groups and user accounts can be created using the `groupadd` and `useradd` commands.

(4). Create mount points to mount storage system volumes. In order to create mount points execute the following command on the database server:

mkdir -p [*MountPoint*]

Where
- `MountPoint` identifies a directory path on the database server that is used for mounting a FlexVol volume.

For example, to create a mount point named `/mnt/dbdata`, you would execute the following command on the database server:

```
mkdir -p /mnt/dbdata
```

Repeat this step and create a mount point for each FlexVol volume to be used for the database.

6.2.1. Linux

(1). Append an entry for each FlexVol volume to the `/etc/fstab` file found on the database server. The entry should look somewhat similar to the following one:

[*StorageSystemName*]:[*VolName*] [*MountPoint*] nfs hard,
rw,nointr,rsize=32768,wsize=32768,bg,vers=3,tcp 0 0

Where
- `StorageSystemName` identifies the name assigned to the storage system.
- `VolName` identifies the name assigned to a FlexVol volume on the storage system.
- `MountPoint` identifies a directory path on the database server that used to mount a FlexVol volume

For example, in order to mount a FlexVol volume named `dbdata` on a mount point named `/mnt/dbdata`, you would add the following line to the `/etc/fstab` file found on the database server:

```
netapp:/vol/dbdata /mnt/dbdata nfs hard,rw,nointr,rsize=32768,
wsize=32768,bg,vers=3,tcp 0 0
```

(2). After adding the entries to the `/etc/fstab` file you need to mount and make the storage system volumes available by executing the following command on the database server:

mount [*MountPoint*]

Where

- `MountPoint` identifies a directory path on the database server that is used to mount a FlexVol volume

For example, to mount a NetApp storage volume named dbdata on a mount point named /mnt/dbdata and make it available, you would execute the following command on the database server:

```
mount /mnt/dbdata
```

Repeat this step for each FlexVol volume to be used for the DB2 database.

(3). In order to create a database on the storage system volumes the database instance owner must have ownership of the mounted file systems. You need to change ownership for each mounted FlexVol volume that is used for the database by executing the following command on the database server:

chown –R db2inst1:db2adm [*MountPoint*]

Where

- `MountPoint` identifies a directory path that is used to mount the FlexVol volume on the database server.

For example, to grant ownership of a file system mounted on the mount point named `/mnt/dbdata` to the user named `db2inst1`, you would execute the following command on the database server:

```
chown –R db2inst1:db2adm /mnt/dbdata
```

6.2.2. IBM AIX

(1). The storage system volumes to be used for the database need to be mounted on the database server. You need to modify the `/etc/filesystems` file found on the database server and add an entry for each storage system volume. Open the `/etc/filesystems` file with a text editor and add entries for each storage system volume. The entry should look similar to the following one:

```
[MountPoint]:
              dev      = [VolPath]
              mount    = true
              vfs      = nfs
              nodename = netapp
              options  = bg,nointr,rw
              type     = nfs_mount
              account  = false
```

Where

- `MountPoint` identifies a directory path that is used to mount a FlexVol volume on a database server.
- `VolPath` identifies the path assigned to the FlexVol volume.

For example, to add an entry for a volume named `dbdata` that is to be mounted on a mount point named `/mnt/dbdata`, you need to append the following lines to the `/etc/filesystem` file on the database server:

```
/mnt/dbdata:
              dev      = /vol/dbdata
              mount    = true
              vfs      = nfs
```

```
nodename = netapp
options  = bg,nointr,rw
type     = nfs_mount
account  = false
```

In the above entry `mount = true` makes a particular file system persistent across the system reboots.

After adding the entries to the `/etc/filesystems` file, you need to mount the FlexVol volumes by executing the following command on the database server:

mount [*MountPoint*]

Where

- `MountPoint` identifies a directory path that is used to mount a FlexVol volume on a database server.

For example, to mount a FlexVol volume on a mount point named `/mnt/dbdata`, you would execute the following command on the database server:

```
mount /mnt/dbdata
```

Repeat this step for each FlexVol volume to be used for the database.

(2). In order to create a database on the storage system volumes the database instance owner must have ownership of the mounted file systems. You need to change ownership for each mounted FlexVol volume by executing the following command on the database server:

chown –R db2inst1:db2adm [*MountPoint*]

Where

- `MountPoint` identifies a directory path that is used to mount a FlexVol volume on a database server.

For example, to change the ownership of a file system mounted on the mount point named `/mnt/dbdata` to the DB2 instance owner named `db2inst1`, you would execute the following command on the database server:

```
chown –R db2inst1:db2adm /mnt/dbdata
```

Repeat this step for each mount point to be used for the DB2 database.

NOTE: You can also use SMITTY or SMIT to mount the storage system volumes.

6.2.3. Sun Solaris

(1). Append an entry for each FlexVol volume to be used for the database to the `/etc/vfstab` file on the database server. The entry should look somewhat similar to the following:

[*StorageSystemName*]:[*VolName*] [*MountPoint*] nfs – yes rw,bg,hard,intr,rsize=32768,wsize=32768,{forcedirectio | llock},tcp,vers=3

Where

- `StorageSystemName` identifies the name assigned to the storage system.
- `VolName` identifies the name assigned to a FlexVol volume on the storage system.
- `MountPoint` identifies a directory path on the database server that used to mount a FlexVol volume

For example, in order to add an entry for a FlexVol volume named `dbdata` that is to be mounted on a mount point named `/mnt/dbdata`, you would add the following line to the `/etc/vfstab` file on the database server:

netapp:/vol/dbdata /mnt/dbdata nfs – yes rw,bg,hard,intr,rsize=32768, wsize=32768,forcedirectio,tcp,vers=3

(2). After appending the entries to the /etc/fstab file you need to mount and make the storage system volumes available by executing the following command on the database server:

mount [*MountPoint*]

Where

- MountPoint identifies a directory path that is used to mount a FlexVol volume on a database server

For example, to mount a FlexVol volume named dbdata on a mount point named /mnt/dbdata, you would execute the following command on the database server:

```
mount /mnt/dbdata
```

Repeat this step for each FlexVol volume to be used for the DB2 database.

(3). In order to create a database on the FlexVol volumes the database instance owner should have ownership of the mounted volumes. You need to change ownership for each mounted file system to be used for the database, by executing the following command on the database server:

chown –R db2inst1:db2adm [*MountPoint*]

Where

- MountPoint identifies a directory path that is used to mount a FlexVol volume on a database server

For example, to change the ownership of a FlexVol volume mounted on a mount point named /mnt/dbdata to the DB2 instance owner named db2inst1, you would execute the following command on the database server:

```
Chown –R db2inst1:db2adm /mnt/dbdata
```

6.2.4. HP-UX

(1). Append an entry for each FlexVol volume to be used for the database, to the /etc/fstab file on the database server. The entry should look somewhat similar to the following one:

[*StorageSystemName*]:[*VolName*] [*MountPoint*] nfs rsize=32768,wsize=32768,hard,nointr,rw,bg,vers=3,proto=tcp,noac,forcedirectio,timeo=600 0 0

Where

- StorageSystemName identifies the name assigned to the storage system.
- VolName identifies the name assigned to a FlexVol volume on the storage system.
- MountPoint identifies a directory path on the database server that used to mount a FlexVol volume

For example, in order to add an entry for a FlexVol volume named dbdata that is to be mounted on a mount point named /mnt/dbdata, you would append the following line to the /etc/fstab file on the database server:

netapp:/vol/dbdata /mnt/dbdata nfs rsize=32768,wsize=32768,hard,nointr,rw,bg,vers=3,proto=tcp,noac,forcedirectio,timeo=600 0 0

(2). After appending the entries to the /etc/fstab file you need to mount the FlexVol volumes available by executing the following command on the database server:

mount [*MountPoint*]

Where

- MountPoint identifies a directory path that is used to mount a FlexVol volume on a database server

For example, to mount a FlexVol volume named `dbdata` on a mount point named `/mnt/dbdata` and make it available, you would execute the following command on the database server:

```
mount /mnt/dbdata
```

Repeat this step for each FlexVol volume to be used for the DB2 database.

(3). In order to create a database on the storage system volumes the database instance owner must have ownership of the mounted volumes. You need to change ownership for each mounted FlexVol volume to be used for the database by executing the following command on the database server:

chown –R db2inst1:db2adm [*MountPoint*]

Where

- `MountPoint` identifies a directory path that is used to mount a FlexVol volume on a database server

For example, to change the ownership of the FlexVol volume mounted on the mount point named `/mnt/dbdata` to the DB2 instance owner named `db2inst1`, you would execute the following command on the database server:

```
chown –R db2inst1:db2adm /mnt/dbdata
```

6.3. Configuring Database Server in SAN environment

After creating initiator group (igroup) and LUN mappings, the database server must be configured to utilize the space available on the storage system LUNs. The steps carried out to configure the database server depend on the flavor of UNIX being used. The following section describes the basic steps used to configure the most common UNIX hosts.

6.3.1. Linux

(1). Set up kernel parameters

The default kernel parameters value may not be good enough to run DB2 successfully on your Linux server. You may be required to update some of the kernel parameters' default values. Some of the recommended Linux Kernel parameter values for DB2 are:

- `kernel.shmmax=268435456 (for 32-bit)`
- `kernel.shmmax=1073741824 (for 64-bit) is should be 90% of total memory`
- `kernel.msgmni=1024`
- `fs.file-max=8192`
- `kernel.sem=250 32000 32 1024`

`NOTE: SHMALL parameter can be calculated using the following formula:`

$$Kernel.\ shmall = ceil\left(\frac{shm\max}{4096}\right)$$

DB2 9 has a new feature that checks the values of the `kernel.semmni`, `kernel.msgmni`, and `kernel.shmmax` parameters when you enter the `db2start` command, and changes them to the recommended value if the current values are not optimal. However, for optimal setting you need to set these parameters along some of the other parameters may need to be changed manually.

To check your current shared memory segment, semaphore array, and message queue limits, enters the following command on your database server:

```
ipcs –l
```

In order to make the kernel parameter changes permanent over system reboot, complete the following steps:

(i). Open up `/etc/sysctl.conf` in a text editor and add entries:

```
kernel.shmmax=268435456
kernel.msgmni=1024
kernel.sem="250 32000 32 1024"
```

(ii). Enter the `sysctl -p` command to load in `sysctl` settings from `/etc/sysctl.conf`.

(iii). Enter `ipcs -l` to view the updated kernel parameters in `sysctl.conf` file.

The entries from the `sysctl.conf` file are read by the network initialization script during startup. On some distributions you may be required to add `sysctl -p` in one of the system initialization files (for example, `rc.local`) so that kernel parameters are set after each reboot. For a detailed list of recommended kernel parameters please refer to the DB2 *documentation "Quick beginning for DB2 server"* chapter *"Preinstallation Task."*

(2). Configuring a Linux Host for FCP

A). Install NetApp host Attach kit.

It is recommended that you download and install NetApp provided "SAN (FCP) host attach kit for Linux." This product simplifies the configuration of a Linux machine as the host component of a NetApp SAN environment. In order to install this product you need to complete the following basic steps:

1) Log in to the NOW (http://now.netapp.com/) Web site using your username and password. Navigate to the "SERVICES & SUPPORT" page and select "Download Software."

2) On the Download Software page, go to the SAN (FCP) Host Attach Kit product row of the table, select Linux from the Select Platform drop-down list, and click on the 'GO' button.

3) Follow the prompts to reach the Software Download page and download the NetApp software file to a local directory.

4) Copy the downloaded attach kit tar file to a directory (`/tmp/netapp`) on the database server and extract the software by executing the following command:

```
tar -xvzf /tmp/netapp netapp_linux_tools_1_0.tar.gz
```

For example, in order to extract the tar file named **`ntap_linux_tools_1_0.tar.gz`**, you would execute the following command on the database server:

```
tar -xvzf /tmp/netapp/netapp_linux_tools_1_0.tar.gz
```

The extracted software is placed in the `/tmp/netapp/netapp_linux_tools_1_0` directory.

5) Change the directory to `netapp_linux_tools_1_0`, where you have extracted attach kit software and execute the following command on the database server:

```
./install
```

6) Add the following lines to `<home>/.bash_profile` file found in the user's home directory:

```
export PATH=$PATH:/opt/NetApp/santools/bin
export MANPATH=/usr/share/man:/opt/NetApp/santools/man
```

For detailed installation steps please refer to ***"Installation and Setup Guide 1.0 for Fibre Channel Protocol on Linux"*** on the NOW Web site.

B). Install the HBA and driver.

Before you install the HBA on the database server, you need to check product compatibility on the NOW Web site. The compatibility matrix is available in on "*Compatibility and Configuration Guide for NetApp*

FCP and iSCSI Products" page. After confirming compatibility, install HBA and its driver on the database server. For the HBA and its driver installation steps, please refer to Product's Installation and configuration guide on NOW Web site (http://now.netapp.com/NOW/knowledge/docs/hba/fcp_linux/fcp_linux10/pdfs/install.pdf).

C). Obtain WWPN for the initiator

Each initiator installed on the database server is uniquely identified by WWPN or WWNN. The WWPN is required to create an igroup for FC on the storage system. The WWPN for the HBA installed on the database server can be obtained by completing the following steps:

1) After installing the NetApp attach kit, the HBA, and the required driver, you would execute the following command on the database server to obtain the `WWPN`:

```
sanlun fcp show adapter -c
```

Upon execution with '-c' option the `sanlun` command generates *"igroup create"* command. The output of the above command looks somewhat similar to the one below:

```
igroup create -f -t linux db2srv1 10000000c93a069b
```

2) The WWPN for adapter in the above example is `10000000c93a069b`. Execute the above generated command to create igroup on the storage system.

3) Refresh FC adapter by executing the following commands on your database server:

```
modprobe -r qla2300
modprobe -v qla2300
```

(3). Configuring a Linux Host for iSCSI

A). Installing NetApp iSCSI Support Kit

It is recommended that you download and install NetApp provided "iSCSI Host Support Kit" for Linux. This product simplifies the configuration of a Linux host in NetApp IP SAN environment. In order to install this product you need to complete the following basic steps:

1) Log in to NOW (http://now.netapp.com/) Web site using your username and password. Navigate to the "SERVICES & SUPPORT" page and select "Download Software."

2) On the Download Software page, go to the *iSCSI Host Support Kit* product row of the table, select Linux from the select platform drop-down list, and click on the 'GO' button.

3) Follow the prompts to reach the *"iSCSI Red Hat Enterprise Linux Initiator Support Kit 1.3 Download Page"* and download the software tar file to a directory on your local machine.

4) Copy the downloaded support kit file to a directory (`/tmp/netapp`) on the database server and extract it using the following command:

```
tar -xvzf [FileName]
```

Where

- `FileName` identifies the name assigned to the downloaded tar file.

For example, to extract the tar file named **`ntap_linux_tools_1_3.tar`**, you would execute the following command on the database server:

```
tar -xvf /tmp/netapp/ntap_linux_tools_1_3.tar
```

5) Change the directory to `ntap_linux_tools_1_3`, where the tar file is extracted and install the product by executing the following command on the database server:

```
./install
```

6) Add the following lines to your <home>/.bash_profile file found in the user's home directory:

```
export PATH=$PATH:/opt/NetApp/santools/bin
export MANPATH=/usr/share/man:/opt/NetApp/santools/man
```

B). Install the iSCSI initiator.

NetApp supports iSCSI software initiator for Linux. In order to install the iSCSI initiator, complete the following steps:

1) If you are using RHEL 3.0 Update 4 or higher then the iSCSI Software initiator is contained in Red Hat Linux distribution; otherwise download from http://sourceforge.net/projects/linux-iscsi/ Web site.

2) After download, create a work directory /tmp/netapp on the database server and ftp the downloaded initiator tar file to the work directory on the database server. Uncompress the tar file by executing the following command:

```
tar –xvzf [FileName]
```

Where

- FileName identifies the name assigned to the downloaded tar file.

For example, to uncompress and extract the initiator tar file named linux-iscsi-4.0.2.gz, you would execute the following command on the database server:

```
tar xvzf /tmp/netapp/linux-iscsi-4.0.2.gz
```

3) Change the directory to linux-iscsi-4.0.2, where you have extracted the software and compile the iSCSI driver by executing the following command on the database server:

```
Make
```

If your kernel sources are not in the usual place, add 'TOPDIR=/path/to/kernel' or edit the definition of TOPDIR in the makefile file.

If your kernel configuration file is not in the usual place, add KERNEL_CONFIG=/path/to/.config' or edit the definition of KERNEL_CONFIG in the makefile file.

4) Install the driver by executing the following command on the database server:

```
make install
```

5) In order to make the initiator discover the LUN devices on the storage system you need to add the following lines to the iSCSI configuration file named /etc/iscsi.conf that is found on the database server:

```
continuous=no
headerDigest=never
dataDigest=never
discoveryAddress=[StorageSystemIP]
```

Where

- StorageSystemIP identifies the network IP address assigned to the storage system.

For example, to make LUNs discovered on a storage system that has IP address 10.32.90.100, you would add following lines to the /etc/iscsi.conf file that is found on the database server:

```
continuous=no
headerDigest=never
dataDigest=never
discoveryAddress=10.32.90.100
```

6) After updating the `/etc/iscsi.conf` file you need to start the iSCSI service and load the driver by executing the following command on the database server:

```
/etc/init.d/iscsi start
```

Every time you make any LUN mapping change the iSCSI driver needs to be refreshed. You can do so by executing the following command on the database server:

```
/etc/init.d/iscsi restart
```

For additional information and troubleshooting please read the README file in the directory where you extracted the initiator software.

C). Obtain WWNN for the initiator.

The iSCSI initiator has a unique identification known as `Word Wide Node Name` or `WWNN`. To access LUNs on the storage system it is required to associate the initiator on the database server with an igroup on the storage system. The `WWNN` for an initiator on the database server is required to create an igroup. The Linux software initiator stores its `WWNN` information in `/etc/initiatorname.iscsi` file. You can obtain the `WWNN` for the initiator by executing the following command on the database server:

```
cat /etc/initiatorname.iscsi
```

(4). Making LUNs Accessible on the database server

DB2 supports two types of tablespace containers, file system containers and raw devices containers. The following sections describe the steps required to configuration both types of containers.

a). Accessing LUNs as regular file system tablespace container:

Once the FCP or iSCSI driver is refreshed, the database server should be able to discover target LUNs from the storage system as new devices. In order to use these devices as file system tablespace containers complete the following steps:

1) Get the assigned names for the newly added devices by executing the following command on database server:

```
sanlun lun show
```

2) Before you use the newly added devices as the tablespace containers, you need to create partition tables and format them by executing the following command on the database server:

fdisk [*DeviceName*]

Where

- `DeviceName` identifies the name assigned to LUN device on the database server.

For example, to create partition tables and format a device named `/dev/sdb`, you would execute the following command on the database server:

```
fdisk  /dev/sdb
```

The `fdisk` command invokes format wizard and you need to answer series of questions in order to complete the formatting.

3) After formatting, you need to create file system on each device by executing the following command on the database server:

mkfs [*DeviceName*]

Where

- `DeviceName` identifies the name assigned to LUN device on the database server

For example, to create file system on the device named `/dev/sdb`, you would execute the following command on the database server:

```
mkfs /dev/sdb
```

4) Add an ext3 journal to the file system, if it was not created as a Journal file system, by execute the following command on the database server:

tune2fs –j [*DeviceName*]

Where

- `DeviceName` identifies the name assigned to LUN device on the database server.

For example, to added ext3 journal to a device named `/dev/sdb`, you would execute the following command on the database server:

```
tune2fs –j /dev/sdb
```

5) After adding the file system, the LUNs devices can be mounted as a shared file system on the database server by executing the following command:

mkdir –p [*MountPoint*]

mount [*MountPoint*] [*DeviceName*]

Where

- `DeviceName` identifies the name assigned to LUN device on the database server
- `MountPoint` identifies a directory path that is used to mount a FlexVol volume on a database server

In order to mount a device named `/dev/sdb` on a mount point named `/mnt/dbdata`, you would execute the following command on the database server:

```
mount /mnt/dbdata /dev/sdb
```

6) In order to create a database on the storage system volumes the database instance owner must have appropriate permissions on all file systems to be used for the database. You can do so by changing the ownership of each mount file system to the database instance owner. The ownership can be changed by executing the following command on the database server:

chown –R db2inst1:db2adm [*MountPoint*]

Where

- `MountPoint` identifies a directory path that is used to mount a FlexVol volume on a database server

For example, to change the ownership of the FlexVol volume mounted on the mount point named `/mnt/dbdata`, you would execute the following command on the database server:

```
chown –R db2inst1:db2adm /mnt/dbdata
```

7) In order to make the file system mount persistent over system reboot you need to add an entry for each file system to the `/etc/fstab` file, which is found on the database server. The entry in `/etc/fstab.iscsi` file should appear similar to the one indicated below:

[*DeviceName*] [*MountPoint*] [*FSType*] [*MountOption*] [*BkupFreq*] [*FsckPass*]

Where

- `DeviceName` identifies the name assigned to LUN device on the database server.
- `MountPoint` identifies a directory path that is used to mount a FlexVol volume on a database server.
- `FSType` identifies the file system type for the device.

- `MountOption` identifies the mount options used for the device.
- `BkupFreq` identifies the frequency of backup. This is used by dump to backup the file system. If the number is zero then dump will ignore the backup for the file system.
- `FsckPass` identifies the order in which the file system integrity should be checked. If the value is set to `0` fsck will not check the filesystem.

For example, to make mount point persistent for a device named `/dev/sdb` that is mounted on `/mnt/dbdata` you would add the following lines to the `/etc/fstab.iscsi` file found on the database server:

```
**************************************************************
#device      mount        FS      mount      backup       fsck
#to mount    point        type    options    frequency    pass

/dev/sdb    /mnt/dbdata  ext3    –netdev      0          0
```

b). *Accessing LUNs as raw devices*

If raw devices are desired as tablespace containers then each LUN device used as raw device needs to be converted to a character device. In order to use newly added LUN devices as raw device tablespace containers, complete the following steps on the database server:

1) Convert newly added LUN devices to character devices by executing the following command on the database server:

   raw /dev/raw/raw[Number] [*DeviceName*]

   Where
   - `DeviceName` identifies the name assigned to LUN device on the database server.

   For example, to convert a block device named `/dev/sdd` to a character device you would execute the following command on the database server:

   ```
   raw /dev/raw/raw1 /dev/sdd
   ```

2) Add an entry for each raw device to the `/etc/sysconfig/rawdevices` file found on the database server. The entry should look somewhat similar to the following:

   [*RawDevName*] [*BlockDeviceName*]

   Where
   - `RawDeviceName` identifies the character device name assigned to LUN device.
   - `BlockDeviceName` identifies the block device name assigned to LUN device.

   For example, to add a raw device named `/dev/raw/raw1,` you would add the following lines to the `/etc/sysconfig/rawdevices` file on the database server:

   /dev/raw/raw1 /dev/sdd

3) After updating the raw device configuration file you need to start raw device daemon and reload the raw devices by executing the following command on the database server:

   ```
   /sbin/service rawdevices restart
   /etc/init.d/rawdevices reload
   ```

4) Set permission to raw devices using the following command:

   chmod 777 [*RawDevName*]

   Where
   - `RawDeviceName` identifies the character device name assigned to LUN device.

   For example, to set permission to 777 for a raw device named `/dev/raw/raw1`, you would execute the following command on the database server:

```
chmod 777 /dev/raw/raw1
```

Sometimes permissions for raw device controller named `/dev/rawctl`, which acts as the central repository of raw to block device binding information, are not set appropriately. You can change the permission for controller by executing the following command on the database server:

```
chmod 777 /dev/rawctl
```

5) Verify if you can write to the raw devices by executing the following command on the database server:

dd if=/dev/zero of=[raw device] bs=4096 count=10

For example, to perform write test on the raw device named /dev/raw/raw1, you would execute the following command on the database server:

```
dd if=/dev/zero of=/dev/raw/raw1 bs=4096 count=10
```

6.3.2. IBM - AIX

(1). Configuring AIX Host for FCP

A). Install NetApp host Attach Kit

It is recommended that you download and install NetApp provided "SAN (FCP) Host Attach Kit" for AIX on the database server. This product simplifies the configuration and management of the AIX host in a NetApp SAN environment. In order to install this product you need to complete the following basic steps

1) Log in to NOW (http://now.netapp.com/) Web site using your username and password. Navigate to the "SERVICES & SUPPORT" page and select "Download Software."

2) On the Download Software page, go to the ***SAN (FCP) Host Attach Kit*** product row of the table, select AIX from the select platform drop-down list, and click on the 'GO' button.

3) Follow the prompts to reach the *"FCP IBM AIX Attach Kit 1.2 Download p*age" and download the Software tar file to a local directory.

4) Copy the downloaded software tar file to a work directory `(/tmp/netapp)` on the database server and uncompress the file using the following command:

```
uncompress /tmp/netapp/ntap_aix_fcp_1.2.tar.Z
```

5) Extract the file by executing the following command on the database server:

tar –xvf [*FileName*]

Where

- `FileName` identifies the tar file name to be extracted.

For example, to extract a tar file named `ntap_aix_fcp_1.2.tar`, you would execute the following command on the database server:

```
tar –xvf /tmp/netapp/ntap_aix_fcp_1.2.tar
```

The above command extracts the attach kit software in `ntap_aix_fcp_1.2` directory.

6) Change the directory to `ntap_aix_fcp_1.2`, where you have extracted the attach kit software and execute the following command:

```
./install
```

Answer the prompt and complete the installation.

7) Add the following lines to the `<home>/.profile` file found on the in the user's home directory on the database server:

```
export PATH=$PATH:/opt/NetApp/santools/bin
export MANPATH=/usr/share/man:/opt/NetApp/santools/man
```

8) After installation is complete, reboot the system using the following command:

```
shutdown -F -r now
```

For detailed installation steps, please refer to the "*FCP IBM AIX Attach Kit 1.2 Installation and Setup Guide*" on NOW Web site.

B). Install the HBA and driver.

Before you install HBA on the database server, you need to check product compatibility on NOW Web site. The compatibility matrix is available on "*Compatibility and Configuration Guide for NetApp FCP and iSCSI Products*" page. After confirming compatibility, install HBA and driver on the database server. For HBA and driver installation, please refer to Product's Installation and configuration guide.

C). Obtain `WWPN` for the HBA

Each FC HBA attached to the database server is uniquely identified by a `WWPN`. In order to create an igroup on the storage system the `WWPN` for the HBA is required. In order to obtain the `WWPN` complete the steps as described below:

1). After installing the NetApp host Attach kit for AIX, HBA, and drivers on the database server, you can find `WWPN` by executing the following command:

```
sanlun fcp show adapter -c
```

Upon execution with '-c' option the `sanlun` command generates '`igroup create`' command which can be used to create an igroup on the storage system. The output from the above command looks similar to the following:

```
igroup create -f -t aix db2srv1 10000000c93a069b
```

2). The `WWPN` for adapter in the above example is `10000000c93a069b`. Use the generated command to create igroup.

3). Load the FC driver by executing the following command on the database server:

```
cfgmgr -l fcs0
```

Once the driver is loaded, the database server should able to see the target LUNs on the storage system as new devices.

(2). Configuring an AIX Host for iSCSI

A). Install NetApp iSCSI host Attach Kit

It is recommended that you install NetApp provided "iSCSI Host Support Kit" for AIX. This product simplifies the configuration and management of NetApp IP SAN environment and it can be installed by few simple steps as described below:

1) Log in to NOW (http://now.netapp.com/) Web site using your username and password. Navigate to the "SERVICES & SUPPORT" page and select "Download Software."

2) On the Download Software page, go to the ***iSCSI Host Support Kit*** product row of the table and select AIX from the select platform drop-down list and click on the 'GO' button.

3) Follow the prompts to reach the *"iSCSI IBM AIX Initiator Support Kit 1.1 Download Page"* and download the Software tar file to a local directory.

4) Copy the downloaded tar file to a work directory (/tmp/netapp) on the database server and uncompress it using the following command:

```
uncompress /tmp/netapp/netapp_aix_SAN_kit_1.1.tar.Z
```

5) Extract the file by executing the following command at the database server:

tar -xvf [*FileName*]

Where

- FileName identifies the tar file name to be extracted.

For example, to extract a file named ntap_aix_SAN_kit_1.1.tar, you would execute the following command on the database server:

```
tar -xvf /tmp/netapp/netapp_aix_SAN_kit_1.1.tar
```

The above command will extract the support kit software in the netapp_aix_SAN_kit_1.1 directory.

6) Change the directory to netapp_aix_SAN_kit_1.1, where you have extracted the support kit software and install the support kit by executing the following command:

```
./install -f -a /tmp/netapp/netapp_aix_SAN_kit_1.1
```

Answer the prompts and complete the installation.

7) After installing the support kit software, add the following lines to the user's <home>/.profile file found in the home directory for the user on the database server:

```
export PATH=$PATH:/opt/NetApp/santools/bin
export MANPATH=/usr/share/man:/opt/NetApp/santools/man
```

8) Reboot the system by executing the following command on the database server:

```
shutdown -F -r now
```

For detailed installation, steps please refer to "*iSCSI IBM AIX Support Kit 1.1 Installation and Setup Guide*" on NOW web site.

B). Configure AIX iSCSI software initiator

NetApp supports iSCSI software initiator for AIX. The iSCSI initiator is included with the IBM AIX 5.2 operating system in the maintenance level 3 (ML3) release. If you are using a lower version of DB2, then download and install iSCSI initiator from IBM Web site. The iSCSI software initiator for AIX can be configured using SMIT as described below:

1) Start SMITTY by executing the following command on the database server:

```
Smit
```

2) Select Devices -> iSCSI -> iSCSI Protocol Device.

3) Select Change / Show Characteristics of an iSCSI Protocol Device.

4) Verify if the Initiator's WWNN is correct; it has to be in a specific format as indicated in the one below:

```
iqn.yyyy-mm.backward_naming_authority:unique_device_name
i.e. - iqn.1992-08.ibmp650.hostid.0xa3cafcd
```

The Initiator's WWNN is required to create an igroup on the storage system.

5) Obtain iSCSI target node name by executing the following command on the storage system:

```
iscsi nodename
```

6) Now define the storage system as an iSCSI target by adding a line to the `/etc/iscsi/targets` file found on the database server. The entry in the `/etc/iscsi/targets` file should look like the following:

[*StorageSystemIP*] 3260 [*StorageSystemNodename*]

Where

- `StorageSystemIP` identifies the network IP address assigned to the storage system.
- `StorageSystemNodename` identifies the nodename assigned to the target adapter on the storage system.

For example, to add a storage system that has an IP `10.32.90.100`, as an iSCSI target for the initiator, you would add the following line to the `/etc/iscsi/targets` file on the database server:

10.32.90.100 3260 iqn.1992-08.com.netapp:sn.33606694

C). Obtain `WWNN` for the initiator

1) The iSCSI initiator is identified by a unique World Wide Node Name or WWNN. You can obtain WWNN for an initiator by executing the following command on the database server:

```
lsattr -l iscsi0 -a initiator_name -E
```

2) Verify if the initiator's `WWNN` is correct. The Initiator's `WWNN` is required for creating an igroup and it has to adhere to a naming format somewhat similar to the following:

iqn.yyyy-mm.backward_naming_authority:unique_device_name
***i.e.* – iqn.1992-08.ibmp650.hostid.0xa3cafcd**

If the initiator name is not in the correct format, it can be changed by executing the following command on the database server:

chdev -l 'iscsi0' -a initiator_name=['NewWWNN']

Where

- `NewWWNN` identifies the new `WWNN` assigned to the initiator.

For example, to change the initiator name, you would execute the following command on the database server:

```
chdev -l 'iscsi0' -a initiator_name='iqn.1992-
08.ibmp650.hostid.0xa384509'
```

3) Load the iSCSI driver by executing the following command on the database server:

```
cfgmgr -l iscsi0
```

D). Making LUNs Accessible on the database server

1) Once the initiator is refreshed, the database server should be able to discover the LUNs as new devices: You can see these newly added devices by executing the following command on database server:

```
lsdev -Cc disk
```

2) Get device names by executing the following command:

```
sanlun lun show
```

Upon execution, the "*sanlun lun* show" command shows the LUN devices along with assigned names. These devices can be used as tablespace containers for the DB2 database.

3) Verify the attributes of the new disks by entering the following command:

lsattr –El [*DeviceName*]

Where
- DeviceName identifies the name assigned to LUN device on the database server.

For example, to verify the attribute of newly added device named hdisk3, you would execute the following command on the database server:

```
lsattr –El hdisk3
```

4) Now create a volume group for each LUN device by issuing the following command on database server:

mkvg –f –y '[VgName]' [*DeviceName*]

Where
- DeviceName identifies the name assigned to LUN device on the database server.

For example to create a volume group named datavg1 for the device named hdisk3, you would execute the following command on the database server:

```
mkvg –f –y 'datavg1' hdisk3
```

Repeat this process until all necessary volume groups are created.

5) You can verify the volume groups by executing the following command on the database server:

```
Lspv
```

If you assigned a wrong name to a Volume Group, it can be changed by executing the following commands:

varyoffvg [*VgName*]
exportvg [*VgName*]
mkvg –q –f –y '[*NewVgName*]' [*DeviceName*]

Where
- VgName identifies the volume group name that is to be changed.
- NewVgName identifies the new volume group name.
- DeviceName identifies the name assigned to LUN device on the database server.

For example, to change a volume group's name from dblogsvg1 to logsvg1 you would execute the following commands on the database server:

```
varyoffvg dblogsvg1
exportvg dblogsvg1
mkvg –q –f –y 'logsvg1' hdisk4
```

6) Create mount points on the database server and assign appropriate permission to the database instance by executing the following commands:

mkdir –p [*MountPoint*]
chown –R [*UserName*]:[*UserGroupName*] [*MountPoint*]

Where
- MountPoint identifies the mount point name that is to be used to mount file system.
- UserName identifies the name assigned to the database instance owner.

- `UserGroupName` identifies the name assigned to the db instance owner group

For example, to create a mount point named `/mnt/dbdata` and change ownership to user name `db2inst1`, you would execute the following command on the database server:

```
mkdir –p /mnt/dbdata
chown =R db2inst1:db2adm /mnt/dbdata
```

7) Create new file system on the volume group by executing the following command:

crfs –v jfs –a bf=true –g `[*VgName*]' –a size=[*FSSize*] { M|G } –m `[*MountPoint*] –p `rw' –a nbpi=4096 –a ag=64

Where

- `VgName` identifies volume group name to be changed.
- `MountPoint` identifies the mount point name to be used to mount file system.
- `FSSize` identifies the size of the file system to be created on the volume group.

For example, to create a file system on the volume group named `data1vg` that is mounted on a mount point named `/mnt/dbdata,` you would execute the following command on the database server:

```
crfs –v jfs –a bf=true –g 'data1vg' –a size=6000M –m '/mnt/dbdata1/' –p
'rw' –a nbpi=4096 –a ag=64
```

Note: The value for `nbpi` must be set to 8192, 16384, 32768, 65546 or 131072 if you need to create a file system that is greater than 64 GB in size.

8) To make the file system mount persistent on system reboots, you need to update the file `/etc/filesystems`. Open the file `/etc/filesystems` and locate the entries that correspond to the file systems you want to make persistent and set the option `mount = true.` For example, to make a file system named `/dev/lvl1` persistent on database server reboot, you would modify the corresponding entry in the `/etc/filesystems` file and after update the entry should look somewhat similar to the following one:

```
/mnt/dbdata:
              dev             = /dev/lv11
              vfs             = jfs
              log             = /dev/loglv10
              mount           = true
              options         = rw
              account         = false
```

Repeat this step for each mounted file system to be used for the database.

9) The tablespace containers that use raw devices reference the character device name assigned to a block device. The character device name corresponding to a block device can be obtained by prefixing the device name with the letter 'r'. For example, the character device name for a block device named `dblogs` shown in the following example should be /dev/rlv12. This is the name that would be used to specify a raw device container for a tablespace.

```
/mnt/dblogs:
              dev             = /dev/lv12
              vfs             = jfs
              log             = /dev/loglv10
              mount           = true
              options         = rw
              account         = false
```

10) Make sure that you can write to the raw device. You can perform this test by executing the following on the database server:

```
dd if=/dev/zero of=/dev/raw/raw1 bs=4096 count=10
```

6.3.3. Sun Solaris

(1). Update Kernel Parameter

The default values for the kernel configuration parameters might not be good enough to run DB2 efficiently. For example, the default values for `semsys:seminfo_semume` and `shmsys:shminfo_shmseg` kernel parameters are not good enough for most of the multi-threaded applications with a fair number of connections. DB2 provides a utility named `db2osconf` to make recommendations for the kernel parameter values based on the size of a system. Complete the following steps to set up appropriate values for the kernel parameters:

1) Use `db2osconf` command with `-t` option to get recommended values:

   db2osconf –t [*NumThreads*]

   Where
   - `NumThreads` identifies the count of threads to be run on the database server.

   For example, you would execute the following command on the database server to get recommended kernel parameter values for 500 threads:

   ```
   db2osconf –t 500
   ```

2) In order to set a kernel parameter value other than default one, you need to add a line to the `/etc/system` file:

   set [*ParameterName*]=[*NewKernalValue*]

   Where
   - `ParameterName` identifies the parameter name to be updated.
   - `NewKernalValue` identifies the new kernel parameter value.

   For example, to set the value of the parameter `msgsys:msginfo_msgmax` to `65535`, you would add the following line to the `/etc/system` file found on the database server:

   set msgsys:msginfo_msgmax=65535

3) After updating the `/etc/system` file, you must restart your system.

   For detailed list of recommended kernel parameters please refer to DB2 manual *"Quick beginning for DB2 server,"* chapter *"Preinstallation Task."*

(2). Configure Sun Solaris Host for FCP

A). Install NetApp SAN (FCP) host Attach kit

It is recommended that you install NetApp provided "SAN (FCP) host attach kit" for Solaris on the database server. This product simplifies the configuration of Solaris host in the NetApp SAN environment. In order to install this product you need to complete the following steps:

1) Log in to NOW (http://now.netapp.com/) Web site using your username and password. Navigate to the "SERVICES & SUPPORT" page and select "Download Software."
2) On the Download Software page, go to the ***SAN (FCP) Host Attach Kit*** product row of the table, select Solaris from the select platform drop-down list, and click on the 'GO' button.
3) Follow the prompts to reach the FCP Solaris Attach Kit 2.0 Download Page and download the software tar file to a local directory.
4) Copy the downloaded software tar file to a work directory `(/tmp/netapp)` on the database server and uncompress the file by executing the following command on the database server:

   ```
   uncompress /tmp/netapp/ntap_solaris_fcp_2_0.tar.Z
   ```

5) After uncompressing, you need to extract the file by executing the following command on the database server:

tar –xvf [*FileName*]

Where
- `FileName` identifies the tar file name to be extracted.

For example, you would execute the following command on the database server to extract software tar file named `ntap_solaris_fcp_2_0.tar`:

```
tar –xvf /tmp/netapp/ntap_solaris_fcp_2_0.tar
```

6) To install the attach kit software you need to change the directory to `ntap_solaris_fcp_2_0`, where you have extracted the software, and execute the following command on the database server:

```
./install
```

Answer the prompt and complete the installation.

For detailed installation steps, please refer to *"FCP Solaris Attach Kit 2.0 Installation and Setup Guide"* on NOW web site.

B). Install the HBA and driver.

Log in to NOW Web site and check the Compatibility and Configuration Guide for NetApp FCP and iSCSI Products for your database server product compatibility. NetApp supports Emulex FC HBA for Sun Solaris. Install the FC HBA and its driver on the database server. For installation, steps please refer to *"Host Bus Adapter Installation and Setup Guide 1.1 for Fibre Channel Protocol on Solaris"* on NOW Web site (http://now.netapp.com/NOW/knowledge/docs/hba/sun/relhbas11/pdfs/setup.pdf).

1) After installing initiator, create an igroup for it and map the LUNs to the igroup. The database server must be configured to make use of the new resources. In order for LUN mappings to take effect you need to augment the "`/kernel/drv/sd.conf`" file with additional entries. The entry should look somewhat similar to the following:

name="sd" parent="lpfc" target=[*TargetID*] lun=[*LunID*];

Where
- `TargetID` identifies the target group ID for the target adapter.
- `LunID` identifies an ID on the target group assigned for the LUN device.

For example, to make LUN ID '`0`' and '`1`' accessible on the target ID '0', you would add the following lines to the "`/kernel/drv/sd.conf`" file on the database server:

```
name="sd" parent="lpfc" target=0 lun=0;
name="sd" parent="lpfc" target=0 lun=1;
```

2) After any modification is made to the `kernel/drv/sd.conf` file, the system must be rebooted before it will take effect. A reconfiguration reboot will scan for and create the new device entries. Execute the following command at the database server:

```
reboot -- -r
```

3) After adding LUN devices you can execute the following command on the database server to refresh the driver:

```
Devfsadm
```

C). Obtain WWPN for the Initiator

Each FC HBA attached to your database server is uniquely identified by WWPN. In order to create igroup on the storage system you will require WWPN for the HBA. After installing FC adapter and driver, you can obtain the WWPN by executing the following sanlun command on the database server:

```
sanlun fcp show adapter –c
```

Upon execution with '–c' option the `sanlun` command generates 'igroup create' command which can be used to create igroup on the storage system. The output from the above command should look like the one noted below:

```
igroup create –f –t solaris db2srv1 10000000c93a069b
```

The WWPN for initiator in the above example is `10000000c93a069b`.

(3). Sun Solaris Host Configuration for iSCSI

A). Install NetApp iSCSI host Support Kit.

It is recommended that you install NetApp provided "iSCSI Host Support Kit" for Solaris. This product simplifies the configuration and management of the SAN environment. In order to install this product you need to complete the following basic steps.

1) Log in to NOW (http://now.netapp.com/) Web site using your username and password. Navigate to the "SERVICES & SUPPORT" page and select "Download Software."

2) On the Download Software page, go to the ***iSCSI Host Support Kit*** product row of the table and select **Solaris** from the Select Platform drop-down list and click on the 'GO' button.

3) Follow the prompts to reach the "iSCSI QLogic Initiator Support Kit 1.2 Download Page" and download the software tar file to a local directory.

4) Copy the downloaded software tar file to a work directory `(/tmp/netapp)` on the database server and uncompress the file using the following command:

```
gunzip /tmp/netapp/ntap_basic_iscsi_solaris_tools_1.0.tar.gz
```

5) After uncompressing, you need to extract the file by executing the following command on the database server:

tar –xvzf [*FileName*]

Where

- `FileName` identifies the tar file name that is to be uncompressed and extracted.

For example, you would execute the following command on the database server to extract tar file named **ntap_basic_iscsi_solaris_tools_1.0.tar:**

```
tar –xvf /tmp/netapp/ntap_basic_iscsi_solaris_tools_1.0.tar
```

6) Change to the `ntap_solaris_openiscsi_1.0N20040713_1731` directory where you extracted the software and execute the following command on the database server:

```
./install
```

This will complete the installation of Host Support Kit.

B). Install the HBA and driver.

NetApp supports Qlogic iSCSI HBA for iSCSI on Solaris. You need to download the iSCSI driver for the initiator and SanSurfer iSCSI Command line utility from Qlogic Web site (*http://www.qlogic.com/products/iscsi_products_hba.asp*). We have used QLogic SANsurfer iSCSI Command Line tools for configuring initiator. Log in as the user root to your database server and perform the basic installation steps as described below:

1) Download the HBA driver and SanSurfer CLI tools from the Qlogic web site to a local directory.

2) Copy the downloaded driver and SanSurfer CLI tools file to a work directory `(/tmp/netapp)` on the database server and uncompress the files by executing the following command on the database server:

```
uncompress /tmp/netapp/qla4010.z
uncompress /tmp/netapp/iscli-1.0.36-2_solaris_sparc_x86.Z
```

3) If you already have previous installation of the QLogic driver then remove it by executing the following command on the database server:

```
pkgrm QLA4010-0
```

Make sure you reboot the system after removing the old driver.

4) Install the driver by executing the following command on the database server:

```
pkgadd -d /tmp/netapp/qla4010
```

5) After starting `'pkgadd'` command follow and answer the prompts displayed to finish the installation.

6) Install the Qlogic SANsurfer iSCSI CLI tools by executing the following command on the database server:

```
pkgadd -d /tmp/netapp/iscli-1.0.36-2_solaris_sparc_x86
```

This completes the Installation for an add-on HBA Driver and SANsurfer CLI tools.

7) Now shut down, power off the system, insert the QLogic HBA into an empty PCI slot on the database server, and connect the adapter to your iSCSI target storage system.

8) Update the `/kernel/drv/sd.conf` file found on your database server by adding the following line for each target LUN as indicated below:

name="sd" class="scsi" class_prop="atapi" target=[*TargetID*] lun=[*LunID*];

Where

- `TargetID` identifies the target ID.
- `LunID` identifies ID on the target for a LUN device.

For example, to add an entry for LUN id 0 that is accessible on target id 0, you would add the following line to the `/kernel/drv/sd.conf` file on the database server:

name="sd" class="scsi" class_prop="atapi" target=0 lun=0;

For detailed installation steps, please download and read ***"readme.txt"*** file from the Qlogic Web site for the HBA you have installed.

9) Now, configuration reboot the system by executing the following command at the database server:

```
reboot -- -r
```

C). Obtain `WWNN` for the initiator

Each iSCSI HBA attached to your database server is uniquely identified with a `WWNN`. In order to create igroup on the storage system you will need `WWNN` for the HBA. You can find the `WWNN` by executing the following Qlogic SANsurfer iSCSI CLI command on the database server:

```
iscli -i
```

The output from the above command should look somewhat similar to the following one:

iqn.1992-08.sunv20z.svl12.0xa3cafcd

To manage and configure Qlogic HBA on Sun Solaris host in NetApp IP-SAN environment you can also use Qlogic SANsurfer iSCSI HBA Manager.

(4). Making LUNs Accessible on the database server

1) After LUNs are discovered, execute the "`format`" command to create a partition table on the new devices on the database server:

```
Format
```

Upon execution of the `format` command, a numbered list of available disks is displayed.

2) Type the number for the disk from the list that you want to repartition.

Specify disk (enter its number): *disk-number*

For example, you would specify 7 in order to create partition and format the disk number 7:

```
Specify disk (enter its number): 7
```

3) This is first time the LUN is used as device so you will be prompted to create a label for the disk. Enter 'no' at the prompt.

```
Disk not labeled. Label it now? No
```

4) Select partition from the menu and type print to display the current partition table.

```
format>part

Partition>print
```

5) Modify the predefined partition table, reserve 1 cylinder at the head of the disk for the partition table and using the "`all free hog`" option for slice 6 as follows:

```
partition> modify
Select partitioning base:
        0. Current partition table (default)
        1. All Free Hog
Choose base (enter number) [0]? 1

Do you wish to continue creating a new partition
table based on above table[yes]? Yes
Free Hog partition [6]? 6

Enter size of partition '0' [0b, 0c, 0.00mb, 0.00gb]: 1c
Enter size of partition '1' [0b, 0c, 0.00mb, 0.00gb]:
Enter size of partition '3' [0b, 0c, 0.00mb, 0.00gb]:
Enter size of partition '4' [0b, 0c, 0.00mb, 0.00gb]:
Enter size of partition '5' [0b, 0c, 0.00mb, 0.00gb]:
Enter size of partition '7' [0b, 0c, 0.00mb, 0.00gb]:
Part      Tag    Flag     Cylinders        Size            Blocks
  0       root    wm       0 -    0        1.00MB    (1/0/0)           2048
  1       swap    wu       0               0         (0/0/0)              0
  2     backup    wu       0 - 6141        6.00GB    (6142/0/0)    12578816
  3 unassigned    wm       0               0         (0/0/0)              0
  4 unassigned    wm       0               0         (0/0/0)              0
  5 unassigned    wm       0               0         (0/0/0)              0
  6        usr    wm       1 - 6141        6.00GB    (6141/0/0)    12576768

  7 unassigned    wm       0               0         (0/0/0)              0
```

6) Create the new partition table by typing '`Yes`' on the prompt and entering table name:

```
Okay to make this the current partition table[yes]? Yes
Enter table name (remember quotes): dbdata1
```

7) Create label and complete the formatting operation on the database server:

```
Ready to label disk, continue? Y
partition> quit
format> quit
```

8) Create file system on the disk by executing the following command at the database server:

newfs –b [*BlockSize*] –i [*NBytes*] [*DeviceName*]

Where

- `BlockSize` identifies the block size for the file system, either 4096 or 8192 bytes per block.
- `NBytes` identifies the number of bytes per inode. The default varies depending on the disk size. This value should reflect the expected average size of files in the file system. If fewer inodes are desired, a larger number should be used. To create more inodes, a smaller number should be given. The default for nbpi is as follows:

Disk size	Density
Less than 1GB	2048
Less than 2GB	4096
Less than 3GB	6144
3GB to 1 Tbyte	8192
Greater than 1 Tbyte	1048576

  The number of inodes can increase if the file system is expanded with the `growfs` command.

- `DeviceName` identifies the name assigned to LUN device on the database server

For example, to create a file system on a device identified as c3t0d0s6, you would execute the following command on the database server:

```
newfs –i 12000 –b 8192  /dev/dsk/c3t0d0s6
```

6.3.4. HP-UX

(1). Set up Kernel Parameters:

Before installing DB2, it is recommended that you update your system's kernel configuration parameters. IBM provides a utility `db2osconf` for DB2, which makes recommendations for kernel parameter values based on the size of a system. Complete the following setups to modify kernel parameters values:

1) Use `db2osconf` command with –t option to get recommended kernel parameter values:

db2osconf –t [*NumThreads*]

Where

- `NumThreads` identifies the count of threads to be run on the database server.

For example, you would execute the following command on the database server to get recommended kernel parameter values for 500 threads:

```
db2osconf –t 500
```

2) Log in to the database server as root user and start the System Administration Manager (SAM) by executing the following command at the database server:

```
Sam
```

3) Select Kernel Configuration parameter from the list.

4) Select the Configurable Parameters from the list.

5) Select the parameter that you want to change and press tab.

6) Select the Action -> Modify Configuration Parameter from the main menu and enter the new value in the Formula/Value field.

7) Press tab and go to OK.

8) Repeat these steps for each of the kernel configuration parameters that you want to change.

9) When you are finished setting all of the kernel configuration parameters, select Action --> Process New Kernel from the action menu bar.

10) On next panel, confirm your modification by selecting `yes` option. This will update kernel parameter and reboot your server automatically.

For detailed list of recommended kernel parameters please refer to DB2 manual *"Quick beginning for DB2 server"* chapter *"Preinstallation Task."*

(2). Configure HP-UX host system for FCP

A). Install NetApp SAN (FCP) host Attach kit

It is recommended that you install NetApp provided "SAN (FCP) host attach kit" for HP-UX. This product simplifies the configuration and management the HP-UX host component of the NetApp SAN environment. In order to install this product you need to complete the following basic steps on the database server

1) Log in to NOW (http://now.netapp.com/) Web site using your username and password and select "Download Software" from "SERVICES & SUPPORT" page.

2) On the Download Software page, go to the ***SAN (FCP) Host Attach Kit*** product row of the table, select HP-UX from the select platform drop-down list, and click on the 'GO' button.

3) Follow the prompts to reach the FCP HP-UX Attach Kit 1.1 and ASL 3.5 Download page and download the software tar file to a local directory.

4) Copy the downloaded software tar file to a work directory (/tmp/netapp) on the database server and uncompress the file using the following command:

```
gunzip /tmp/netapp/ntap_hpux_fcp_1_1.depot.gz
```

5) Install NetApp Host Attach kit by executing the following command on the database server:

```
/usr/sbin/swinstall -s /tmp/netapp/ntap_hpux_fcp_1_1.depot
```

6) Add the following lines to the `<home>/.profile` file found in the user's home directory on the database server:

```
export PATH=$PATH:/opt/NetApp/santools/bin
export MANPATH=/usr/share/man:/opt/NetApp/santools/man
```

For detailed installation steps, refer to the installation guide *"FCP HP-UX Attach Kit 1.1 Installation and Setup Guide"* available on NOW Web site.

B). Install the HBA and driver

In order to install the HBA complete the following steps on your database server:

1) Log in to NOW Web site and go to *"Compatibility and Configuration Guide for NetApp FCP and iSCSI Products"* page (http://now.netapp.com/NOW/knowledge/docs/san/fcp_iscsi_config).
2) Check *FCP Host Compatibility Matrix* and verify the product compatibility for your database server products.

3) Download the HBA driver from the HP Web site (http://software.hp.com). You should also download the HBA installation guide from hp Web site (http://docs.hp.com/en/netcom.html).
4) Follow instructions described in HBA "Installation and configuration guide" and install the card and driver on the database server.

C). Configure the initiator and the WWPN

1) Each FC HBA attached to your database server has a unique identification known as WWPN. In order to create igroup on the storage system you will need WWPN for the HBA. After installing the HBA and NetApp host Attach kit for HP-UX, you can obtain WWPN by executing the following command on the database server:

```
sanlun fcp show adapter –c
```

Upon execution with '-c' option sanlun command generates the 'igroup create' command. The output from the above command looks similar to the following:

```
igroup create –f –t hpux db2srv1 50060b00002cf7c8
```

The WWPN for initiator in the above example is 50060b00002cf7c8.

You can use the above generated 'igroup create' to create an igroup on the storage system.

2) Enter the following command on the database server to create device nodes for the storage system LUNs:

```
ioinit –i
```

Use the insf –e command if the ioinit –i command does not create device nodes for all of the paths to the LUN.

3) The LUN devices are not automatically visible to the database server. To perform LUN discovery, you need to execute the following command on the database server:

```
ioscan –fn –C disk
```

(3). Configure HP-UX host system for iSCSI

A). Install NetApp iSCSI Host Support kit

It is recommended that you install NetApp provided "iSCSI Host Support Kit" for HP-UX. This product simplifies the configuration and management of HP-UX host component of NetApp IP SAN environment. In order to install this product you need to complete the following basic steps on the database server:

1) Log in to NOW (http://now.netapp.com/) Web site using your username and password and select "Download Software" from "SERVICES & SUPPORT" page.
2) On the Download Software page, go to the ***iSCSI Host Support Kit*** product row of the table and select HP-UX from the select platform drop-down list and click on the 'GO' button.
3) Follow the prompts to reach the "iSCSI HP-UX Initiator Support Kit 1.2 Download Page" and download the software tar file to a local directory.
4) Copy the downloaded support kit file to a work directory (/tmp/netapp) on the database server and uncompress the file using the following command:

```
gunzip /tmp/netapp ntap_hpux_iscsi_1.2.depot.gz
```

5) Install the support kit by executing the following command on the database server:

```
swinstall –s /tmp/netapp/ntap_hpux_iscsi_1.2.depot iscsitools
```

The diagnostic scripts are installed to the `/opt/NetApp/iscsitools/bin` directory.

6) Add the following lines to the `.profile` file found on the database server in the user's home directory:

```
export PATH=$PATH:/opt/NetApp/santools/bin
export MANPATH=/usr/share/man:/opt/NetApp/santools/man
```

B). Install the iSCSI initiator

NetApp supports software initiator for HP-UX. In order to install HP-UP iSCSI Software Initiator complete the following steps.

1) Log in to NOW Web site and go to *"Compatibility and Configuration Guide for NetApp FCP and iSCSI Products"* page (http://now.netapp.com/NOW/knowledge/docs/san/fcp_iscsi_config).

   Check *iSCSI Host Compatibility Matrix* and find the supported initiator details for your database server products.

2) Download the iSCSI software initiator from HP Web site (http://software.hp.com).

   Enter iSCSI Software Initiator in the search box. When the search results show *iSCSI Software Initiator* click on Receive for Free button.

3) Follow the prompts to reach the Software download page and download the software initiator file and installation instructions to a local directory.

4) Copy the downloaded initiator file to a work directory `(/tmp/netapp)` on the database server.

5) Install the initiator by executing the following command on the database server:

```
swinstall -x autoreboot=true -s /tmp/netapp/iSCSI-00_B.11.23.03e_HP-
   UX_B.11.23_IA+PA.depot iSCSI-00
```

   The "`autoreboot=true`" option will cause a system reboot after installation is complete.

6) Add the following line to ./`profile` file found in the home directory of the user on the database server:

```
export PATH=$PATH:/opt/iscsi/bin
```

7) Download the *"HP-UX iSCSI Software Initiator Support Guide"* for http://docs.hp.com and refer for detailed installation and configuration steps.

C). Configure the initiator and Obtain WWNN

1) After installing iSCSI initiator, you would execute the following command on the database server to obtain its `WWNN`:

```
/opt/iscsi/bin/iscsiutil -l

Initiator Name :iqn.1980-03.com.hp:cx2600-1.d25ca127-6ec1-11d7-a3a1-
9759e748a354
```

   Use this value of `WWNN` to create igroup on the storage system. The initiator's `WWNN` has to be in a specific format as indicated below:

```
iqn.yyyy-mm.backward_naming_authority:unique_device_name
```

   If the `WWNN` is not in the required format then you can change the `WWNN` value by executing the following command on the database server:

```
iscsiutil /dev/iscsi -i -N [iSCSIInitiatorName]
```

   Where

   - `iSCSIInitiatorName` identifies the name assigned to iSCSI initiator on the database server.

For example, to change initiator's WWNN from `iqn.1980-03.com.hp:cx2600-1.d25ca127-6ec1-11d7-a3a1-9759e748a354` to `iqn.1986-03.com.hp:hpfcs214.2000853943`, you would execute the following command on the database server:

```
iscsiutil /dev/iscsi -i -N iqn.1986-03.com.hp:hpfcs214.2000853943
```

2) After installing the initiator and its driver, the iSCSI service needs to be started on the database server. You would execute the following command on the database server to start iSCSI service:

```
/sbin/init.d/iscsi start
```

3) Each iSCSI target used for the database need to be identified by the database server persistently over the system reboots. In order to make the target's storage system identification persistent, you need to add the name or IP address of the storage system to the kernel configurations of the database server by executing the following command:

iscsiutil /dev/iscsi -a -I [*StorageSystemIPORName*] -P <TCPPort> -M <PortalGrpTag>

Where

- ***StorageSystemIPORName*** identifies the name or IP address assigned for the storage system used.
- **TCPPort** identifies the TCP port number for the target. Default value is 3260
- **PortalGrpTag** identifies the target portal group ID. Default value is 1.

For example, to make static identification for the storage system named `netapp`, you would execute the following command on the database server:

```
iscsiutil /dev/iscsi -a -I netapp
```

4) To make database server discover storage system LUNs, you would execute the following command on the database server:

```
ioscan -H 255
```

You can check the newly discovered devices by executing the following command on the database server:

```
ioscan -funC disk
```

5) To create device nodes for a storage system LUN device execute the following command:

```
ioinit -i
```

Use the `insf -e` command if the `ioinit -i` command does not create device nodes for all of the paths to the LUN.

(4). Making LUNs Accessible on the database server

In order to create a database on the storage system LUN devices you need to create physical volumes, volume groups, and logical volumes on them. You need to complete the following steps to create file systems and configure the newly added devices:

1) The database server assigns a device name to each LUN it discovers. Obtain the assigned device name for each LUN by executing the following sanlun command:

sanlun lun show -p [*StorageSystemIPORName*]:[*LunPath*]

Where

- `StorageSystemIPORName` identifies the name or IP address assigned for the storage system.

- LunPath identifies the assigned LUN path.

For example, to find a device name for the LUN named /vol/dbdata/data1 on a storage system named netapp, you would execute the following command on the database server:

```
sanlun lun show -p netapp:/vol/dbdata/data1
netapp:/vol/dbdata/data1 (LUN 0)
     75g (80530636800)     lun state: GOOD
Storage system_CF_State: Cluster Enabled  Multipath_Policy: None
Multipath-provider: None
--------  --------- ---------------  -------- ---------- ----------
host path filer path      /dev/dsk   host      primary    partner
state     type            filename   HBA       filer port filer port
--------  --------- ---------------- --------  ---------- --------
up         FCP       /dev/dsk/c4t0d0 lan1     10.32.90.31
```

The block device name in the output from the above command is /dev/dsk/c4t0d0. The corresponding character device name will be /dev/rdsk/c4t0d0.

2) You can create a physical volume on the storage system LUN by specifying its character device name as indicated below:

pvcreate [*CharDeviceName*]

Where
- CharDeviceName identifies the character device name assigned to LUN.

For example, to create a physical volume on the character device named /dev/rdsk/c4t0d0, you would execute the following command on the database server:

```
pvcreate /dev/rdsk/c4t0d0
```

3) Execute the following command on the database server to display a list of existing volume groups:

```
ls -l /dev/*/group
crw-r-----  1 root sys   64 0x000000 Mar 28 10:59 /dev/vg00/group
```

Pick an unused minor device number for the new volume group. For example, in the output of the previous command the used minor device number is '0' so next unused minor device number is '1'. Set a shell variable containing the next unused minor device number.

```
VG=1
```

4) Before a volume group is created, a device node directory for the volume group must exist, the minor device number needs to be converted to hexadecimal base, and a device node needs to be created. Enter the following command on the database server to create a device node directory:

mkdir -p /dev/[*VgName*]

Where
- VgName identifies volume group name the directory is created for.

For example, to create a device node directory for the volume group named vg01_dbdata, you would execute the following command on the database server:

```
mkdir -p /dev/vg01_dbdata
```

Enter the following command on the database server to convert the minor device number to hexadecimal:

```
VGH=$(echo 16o $VG p|dc)
```

Execute the following command to create device node for the volume group:

mknod /dev/[*VgName*]/group c 64 ox[*HexMinorDeviceNumber*]0000

Where

- VgName identifies volume group name.
- HexMinorDeviceNumber identifies major device number in hexadecimal format.

For example, to create a device node for the volume group vg01_dbdata you would execute the following command on the database server:

```
mknod /dev/vg01_dbdata/group c 64 0x${VGH}0000
```

Repeat steps 1 through 4 for each LUN you want to use for the database.

5) Now create a volume group by executing the following command on database server:

vgcreate [*VgName*] /dev/dsk/[device], …

Where

- VgName identifies volume group name.

For example, to create a volume group named vg01_dbdata, you would execute the following command on the database server:

```
vgcreate vg01_dbdata /dev/dsk/c4t0d0, /dev/dsk/c4t0d1
```

A single volume group can have multiple devices (Note- The device name for a LUN is obtained from step '2' of this section). Depending on the size of volume group you are creating, you need to specify an extent size to the vgcreate command. The total storage in the volume group can be divided into maximum of 65535 extents. The extent size can be power of 2 between 1 and 256 MB.

6) Create logical volume on the volume group by executing the following command on the database server:

lvcreate –L size –n [*VgName*] [L*VgName*]

Where

- VgName identifies logical volume group name that is to be created.
- LVgName identifies volume group name the logical volume group is created on.

For example, to create a 1000MB logical volume named dbdata within vg01_dbdata volume group, you would execute the following command on the database server:

```
lvcreate –L 1000 –n dbdata vg01_dbdata
```

Repeat step '6' until all necessary logical volumes have been created. You can create up to 255 logical volumes in one volume group.

7) To see the volume group detail execute the following command on the database server:

```
vgdisplay –v vg01_dbdata
```

8) If database tablespace containers use raw devices then skip this step and proceed with step '9.'

(i). Create a new file system on each logical volume by executing the following command on the database server:

newfs –F vxfs –o largefiles /dev/[*VgName*]/[*LVgName*]

Where

- LVgName identifies logical volume group name.
- VgName identifies volume group name the logical volume group is created on.

For example, to create a file system on the logical volume named dbdata, you would execute the following command on the database server:

```
newfs -F vxfs -o largefiles /dev/vg01_dbdata/dbdata
```

Repeat this step until file systems have been created on all required logical volumes.

(ii). Create the mount points and change the ownership to the database instance owner by executing the following commands on the database server:

mkdir -p /mnt/[*MountPoint*]
chown -R db2inst1:db2adm [*MountPoint*]

Where
- `MountPoint` identifies the mount point name to be used to mount file system.

For example, to create mount point named `/mnt/dbdata` and change its ownership to the database instance owner named `db2inst1`, you would execute the following command on the database server:

```
mkdir -p /mnt/dbdata
chown -R db2inst1:db2adm /mnt/dbdata
```

Repeat this step and create all required mount points for your environment.

(iii). Now mount the file system using newly created mount points by executing the following command on the database server:

mount -F vxfs /dev/[*VgName*]/[*LVgName*] [*MountPoint*]

Where
- `MountPoint` identifies the mount point name to be used to mount file system.
- `LVgName` identifies logical volume group name.
- `VgName` identifies volume group name the logical volume group is created on.

For example, to mount a file system to a mount point named `/mnt/dbdata`, you would execute the following command on the database server:

```
mount -F vxfs /dev/vg01_dbdata/dbdata /mnt/dbdata
```

(iv). In order to make the file system mounting persistent on the database server reboot, you need to add an entry for each file system you have created to the `/etc/fstab` file found on the database server. The entry in the `/etc/fstab` file should look somewhat similar to the following:

/dev/[*VgName*]/[*LVgName*] [*MountPoint*] vsfx delaylog 0 2

Where
- `MountPoint` identifies the mount point name to be used to mount file system.
- `LVgName` identifies logical volume group name.
- `VgName` identifies volume group name the logical volume group is created on.

For example, you would add the following line to the `/etc/fstab` file on the database server to make mount persistent for the logical volume named `dbdata` on the mount point named `/mnt/dbdata`:

/dev/vg01_dbdata/dbdata1 /mnt/dbdata vsfx delaylog 0 2

9) If raw devices are desired as the tablespace containers then reference the logical volume name prefixed with 'r' as the storage container name. For example, the raw device name for the logical volume `/dev/vg01_dbdata/dbdata1` would be `/dev/vg01_dbdata/rdbdata1`. This is the name that would be used to specify the container for a raw DMS tablespace.

7. Creating a New Database on a NetApp Storage System

This section describes the steps needed to create a new DB2 database whose data and transaction log files reside on the NetApp storage system volume(s)/LUN(s). The creation of DB2 database is a very straightforward process that utilizes standard DB2 commands. Complete the following steps to create DB2 database:

1) Log in as the instance owner and Install the DB2 on the database server. For installation instructions, refer to DB2 manual *"Quick Beginnings for DB2 Servers*."

2) You can create a new database on a FlexVol volume that is mounted to a mount point on the database server. You can do so by executing the following command on the database server:

   db2 "CREATE DATABASE [*DatabaseName*] on [*MountPoint*]"

   Where
   - `MountPoint` identifies the name assigned to the mount point to be used for the database.
   - `DatabaseName` identifies the name of the database to be created

   The mount point name has to be explicitly specified in CREATE DATABASE command.

   For example, to create a database named `MYDB` on a FlexVol volume named `dbdata` that is mounted on a mount point named `/mnt/dbdata`, you would execute the following command on the database server:

   ```
   db2 "CREATE DATABASE MYDB ON /mnt/dbdata"
   ```

3) By default, when a database is first created, circular logging is used and primary log files are created in the subdirectory "`[`*DbPath*`]/[`*InstanceName*`]/[`*NodeName*`]/`*SQLnnnn*`/SQLOGDIR/`."

   Where

   - `DbPath` identifies the path where the database is created on.
   - `InstanceName` identifies the name of the database instance.
   - `NodeName` identifies the number of the database node.
   - `SQLnnnn` identifies the number referring to the database in the instance. If you have only one database in the instance then this value will be SQL0001.

   It is recommended that you move transaction log files to another volume on the storage system. You can do so by updating the database registry parameter named `NEWLOGPATH`. In order to update this parameter, you would execute the following command on the database server:

   db2 "UPDATE DB CFG FOR MYDB USING NEWLOGPATH [new log location]"

   For example, to move transaction logs for a database named `mydb` to another volume named `dblogs` that is mounted on a mount point named `/mnt/dblogs`, you would execute the following command on the database server:

   ```
   db2 "UPDATE DB CFG FOR MYDB USING NEWLOGPATH /mnt/dblogs"
   ```

   Keep in mind that, in most cases, changes made to a database configuration file do not take effect until the database has been restarted after terminating all the database connections.

4) You can verify If the database is created correctly and exists on the storage system by executing the following command on the database server:

   ```
   db2 "LIST DB DIRECTORY"
   ```

   If the database was created successfully, its name should appear in this list, along with information about where the overhead files for this database are stored.

5) Assuming the database was created successfully, establish a connection to it by issuing the following command:

db2 "CONNECT TO [*DatabaseName*]"

Where

- `DatabaseName` is the alias assigned to the database that was created (if no alias was specified, the alias will be the same as the name assigned to the database).

For example, you would execute the following command on the database server to connect to the database named mydb:

```
db2 "CONNECT TO mydb"
```

6) List the tablespaces that were created with the database by executing the following command on the database server:

```
db2 "LIST TABLESPACES"
```

7) Verify that the tablespace container(s) associated with each tablespace refer to a file or directory located on a FlexVol volume by executing the following command:

db2 "LIST TABLESPACE CONTAINER [*TBSpaceID*] SHOW DETAIL"

Where

- `TBSpaceID` is the unique identifier (0, 1, 2, etc.) that has been assigned to the tablespace that container information is to be retrieved for.

If the database was created correctly, the container information returned by this command should identify the file or directory that was assigned to the tablespace specified when the **CREATE DATABASE** command was executed.

8) For example, to verify the tablespace container for tablespace ID 2, you would execute the following command on the database server:

```
db2 "LIST TABLESPACE CONTAINER 2 SHOW DETAIL"
```

9) Additional tablespaces can be created on FlexVol volume using “CREATE TABLESPACE” command. For details on the command syntax, refer to IBM DB2 SQL Reference Vol. 2.

8. Migrating Existing DB2 Databases to a NetApp Storage System

The migration of existing DB2 databases to a FlexVol volume is also a fairly straightforward process that utilizes standard DB2 commands. This section describes the steps needed to convert an existing DB2 database whose data and transaction log files reside on local disk to a new DB2 database whose data and transaction log files reside on a storage system volume.

Note: The items described in section 3, Infrastructure, are required in order for this process to work correctly.

**** IMPORTANT:*
Always ensure that you have fully recoverable backup copies of your database and related files before attempting to migrate an existing database to a storage system volume.

8.1. Migrating a DB2 database using redirected restore operation

The easiest way to move a DB2 database from one storage location to another is by backing up the database with DB2's **BACKUP DATABASE** command and then performing what is known as a "**redirected restore**" operation to copy the database to its new location. Redirected restore operations are performed by executing a combination of DB2's **RESTORE DATABASE** and **SET TABLESPACE CONTAINERS** commands once a backup image of the database has been obtained. The steps needed to perform this type of DB2 database migration are:

(1). Establish a connection to the database to be migrated by issuing the following command:

db2 "CONNECT TO [*DatabaseName*]"

where –

- DatabaseName identifies the database alias or name.

For example, you would execute the following command on the database server to connect to a database named mydb:

```
db2 "CONNECT TO mydb"
```

(2). List the tablespaces that have been created for the database to be migrated by executing the following command:

```
db2 "LIST TABLESPACES" > tablespace.out
```

(3). Stop and restart the DB2 Database Manager instance (to make the change take effect) by executing the following commands:

```
db2 "FORCE APPLICATIONS ALL"
db2stop
db2start
```

(4). Create a full backup image of the database to be migrated by issuing the following command on the database server:

db2 "BACKUP DATABASE [*DatabaseName*] TO [*BKUPLocation*]"

Where

- DatabaseName identifies the database alias or name
- BKUPLocation is the location where the database backup image is to reside.

For example, to back up a database named mydb on a location named /dbbkup/mydb you would execute the following command on the database server:

```
db2 "BACKUP DATABASE mydb TO /dbbkup/mydb"
```

(5). Create a script file that contains the commands needed to perform a redirected restore. This script file should contain the following commands:

db2 "restore database [*OldDatabaseName*] from [*BKUPLocation*] to [*NewLocation*] into [*NewDatabaseName*] redirect"
db2 "set tablespace containers for [*TBSpaceID*] using ('[Directory]')"
or
db2 "set tablespace containers for [*TBSpaceID*] using (FILE '[*FileName*]' [Size])"
or
db2 "set tablespace containers for [*TBSpaceID*] using (DEVICE '[*DeviceName*]' [SIZE])"
db2 "restore database [*OldDatabaseName*] continue"

Where-

- OldDatabaseName identifies the alias assigned to the database that was backed up,
- BKUPLocation identifies the location where the database backup image resides,
- NewLocation is the location where the overhead files associated with the database are to reside,
- NewDatabaseName is the name that is to be assigned to the new database that will be created,
- TBSpaceID is the unique identifier (0, 1, 2, etc.) that has been assigned to the tablespace that new container information is to be assigned for,
- Directory is the new directory SMS tablespace data is to be stored in,
- FileName is the name of the file that DMS tablespace data is to be stored in,
- DeviceName is the name of the device that is to be used to store tablespace data, and

- `Size` is the size, in 4, 8, 16, or 32 k pages, of the file or device specified.

For example, to restore a database named mydb database from a backup location named `/dbbkup/mydb` and move the tablespace containers to FlexVol volume your script should look somewhat similar to the following:

```
DumpDir=/dbbkup/mydb
NewDBDIR=/mnt/dbdata/mydb
OLD_DB_NAME=mydb
NEW_DB_NAME=mydb
db2 "restore database $OLD_DB_NAME from $DumpDir TO $NewDBDIR INTO
$NEW_DB_NAME redirect without prompting" > restore.out
db2 "SET TABLESPACE CONTAINERS FOR 0 USING (PATH $NewDBDIR/catalog)"
db2 "SET TABLESPACE CONTAINERS FOR 1 USING (PATH $NewDBDIR/temp)"
db2 "restore db $OLD_DB_NAME continue"
```

Note: Using the information collected in Step 3, you will need to create “set tablespace container for” commands for each tablespace found in the database.

Execute the script file just created to perform a redirected restore operation.

(6). Verify that the database was successfully migrated by performing a few simple tests (as outlined in step 4 of *Section 7. Verifying a DB2 Database was created on a NetApp storage system*).

8.2. Migrating a DB2 database using db2look and db2move

Earlier, it was mentioned that the easiest way to move a DB2 database from one storage location to another is by backing up the database and then performing a redirected restore operation to copy the database to a new location. However, because a redirected restore operation allows you to change the storage location for one or more tablespaces but not the tablespace type (SMS or DMS), this process cannot be used for migrating a DB2 database if you wish to change the tablespace type (for example, to switch from DMS tablespaces to SMS tablespaces).

Instead, to perform this type of migration, a new DB2 database that uses the desired tablespace types must be created and data from the existing database must be copied to the new database. Data can be copied on a table-by-table basis using DB2's `EXPORT` and `IMPORT` commands, but a more efficient way to copy an entire DB2 database is by using DB2's `db2move` utility. This utility queries the system catalog tables for the specified database and compiles a list of all user tables found. It then exports the contents and table structure of each table found to a PC/IXF formatted file. The set of files produced can be used to populate a new a DB2 database that uses tablespaces that are associated with files stored on a storage system.

The `db2move` utility can be run in one of three modes: EXPORT, IMPORT, or LOAD. When run in EXPORT mode, the `db2move` utility invokes DB2's EXPORT utility to extract data from one or more tables and externalize it to PC/IXF formatted files. It also creates a file named `db2move.lst` that contains the names of all tables processed, along with the names of the files that the table's data was written to. In addition, the `db2move` utility may produce one or more message files that contain warning or error messages that were generated as a result of an export operation.

When run in IMPORT mode, the `db2move` utility invokes DB2's Import utility to recreate a table, and its indexes from data stored in PC/IXF formatted files. When run in this mode, the file `db2move.lst` is used to establish a link between the PC/IXF formatted files needed and the tables into which data will be imported.

When run in LOAD mode, the `db2move` utility invokes DB2's LOAD utility to populate tables that already exist with data stored in PC/IXF formatted files. Again, the file `db2move.lst` is used to establish a link between the PC/IXF formatted files needed and the tables into which data will be loaded. For the purpose of migrating a DB2 database, the `db2move` utility should never be run in LOAD mode.

Unfortunately, the `db2move` utility only works with table and index data objects. If the database to be migrated contains other data objects such as aliases, views, triggers, user-defined data types (UDTs), user-defined functions (UDFs), etc., you must find another way to migrate those objects as well. This is where the

DB2 utility `db2look` comes in. This utility analyzes an existing database and produces a set of Data Definition Language (DDL) SQL statements that can then be used to recreate all of the objects found in the database analyzed.

Now that we have seen how the tools that are needed work, let's take a look at how this type of migration is performed. The steps needed to perform this type of DB2 database migration are:

(1). Stop and restart the DB2 Database Manager instance by executing the following commands:

```
$ db2 "FORCE APPLICATIONS ALL"
$ db2stop
$ db2start
```

(2). Generate DDL that can be used to recreate the data objects found in the DB2 database to be migrated by executing the following command:

db2look -d [*DatabaseName*] -e -o [*OutFile*]

Where

- `DatabaseName` identifies the database alias or name that is to be migrated.
- `OutFile` identifies the name and location of the file that the DDL information produced is to be written.

For example, to create DDL for the object database named `mydb`, you would execute the following command on the database server:

```
db2look -d mydb -e -o /export/home/db2ins81/mydb.ddl
```

When executed, the command will place the `mydb.ddl` file in the directory `/export/home/db2ins81.`

(3). Edit the file produced in Step 2 and delete the first CONNECT statement found. Save the file when finished.

(4). Use `db2move` command to query the system catalog tables of the DB2 database to be migrated and export the contents and table structure of each table found to a PC/IXF formatted

db2move [*DatabaseName*] EXPORT

Where

- `DatabaseName` is the alias assigned to the database that is to be migrated.

For example, to migrate a database named `mydb`, you would execute the following command on the database server:

```
db2move mydb export
```

(5). Create a new database on the storage system volumes as outlined in section 7.

Note: If you want to assign the name of the database being migrated to the database being created, you must first drop or uncatalog the database being migrated. If you select to uncatalog the database being migrated, you can always recatalog it.

(6). Establish a connection to the database created in Step 5 and create the database objects using script created `db2look` utility in step 2:

```
db2 "connect to mydb1"
db2 -t -f /export/home/mydb.ddl
```

(7). Import the contents and table structure of each table found in the DB2 database being migrated to the database created in Step 5 by executing the following command:

```
db2move mydb IMPORT
```

(8). Verify that the database was successfully migrated by performing simple tests as outlined in step 4 of section 7.

9. Performance Considerations

DB2 provides two sets of configuration parameters (one for the DB2 Database Manager instance and one for the database itself) along with several registry and environment variables that can be used to tune a system for optimum performance. Two of these registry variables, ***DB2_PARALLEL_IO*** and ***DB2_STRIPED_CONTAINERS***, should always be set when DB2 is used in conjunction with a storage system. The system command `db2set` is used to display, set, or remove values for DB2 registry/environment variables; after setting any registry variable, the DB2 Database Manager must be stopped and restarted before the changes will take effect.

9.1. The `DB2_PARALLEL_IO` registry variable

When reading data from or writing data to tablespace containers, DB2 can use parallel I/O if the number of containers in the tablespace is greater than 1. However, there are situations when it would be beneficial to have parallel I/O enabled for single container tablespaces. For example, if the container is created on a FlexVol volume or qtree, performance may be improved if read and write calls are issued in parallel.

To force DB2 to use parallel I/O for a tablespace that has only one container, you use the **DB2_PARALLEL_IO** registry variable. This variable can be set to asterisk (*), meaning every tablespace in every database for the database instance is to use parallel I/O, or it can be set to a list of tablespace IDs that are separated by commas. Execute one of the following commands on your database server:

```
db2set DB2_PARALLEL_IO=*
or
db2set DB2_PARALLEL_IO=1,2,4,8
```

In the above example, first command will turn parallel I/O on for all tablespaces, whereas the second command will turn parallel I/O on only for tablespaces 1, 2, 4, and 8.

9.2. The DB2 configuration advisor

When a database is created using CREATE DATABASE command it is created with a set of a default database configuration parameter values. These configuration parameter values may not be appropriate for your environment. To obtain a good performance you may need to tweak these parameter values. If your experience with DB2 is limited, you can use DB2 configuration Advisor to tune these configuration parameters.

The Configuration Advisor is a tool that asks you a series of questions about your database environment and, using the answers provided, recommends configuration parameter values that will improve overall database performance. The Configuration Advisor is invoked from the DB2 Control Center by highlighting the database you want to tune, from the list of databases presented, pressing the right mouse button to display the database action menu, and selecting **Configuration Advisor.**

9.3. Additional factors that affect performance

Finally, when running a DB2 database whose data and transaction files reside on a storage system, you should take the following into consideration:

In order to achieve point-of-failure recovery, you must ensure that the database's active and archive log files are always accessible and up-to-date. The first step towards doing this is to store a database's transaction log files on a volume that is separate from the volume where the database's object files reside. (In our test environment, the database object files were stored on the FlexVol volume named **`dbdata`** and the database's transaction log files were stored on the FlexVol volume named **`dblogs`**.)

Each tablespace should be assigned a prefetch size that is a multiple of the extent size used multiplied by the number of data disks used in the Storage System RAID group the tablespace resides on.

If you are using SMS tablespaces, tell DB2 to allocate new SMS space by the extent rather than the page. This is done by enabling multipage file allocation for a database by executing the DB2 command on the database server:

db2empfa [*DatabaseName*]

Where

- `DatabaseName` identifies the database alias or name.

For example, to enable multipage file allocation for the database named `mydb`, you would execute the following command on the database server:

```
db2empfa mydb
```

10. Enabling NVFAIL on the Storage System for Additional Data Protection

When storing database data on a storage system, it is a good idea to enable the `nvfail` feature for the volumes used for the database. The `nvfail` is a Data ONTAP operating system software feature and provides support for special error processing, and it is enabled by entering the following command on the storage system:

vol options [*VolName*] nvfail on

Where –

- ***VolName*** is the name of the FlexVol volume the `nvfail` feature is to be enabled for.

For example, to enable `nvfail` feature for the volume named `dbdata,` you would execute the following command on the storage system:

```
vol options dbdata nvfail on
```

If an NVRAM failure occurs on a volume, Data ONTAP detects the failure at boot up time. If you enabled the vol options `nvfail` option for the volume and it contains the LUNs, Data ONTAP performs the following actions:

- Put the volumes and LUNs within volumes offline that had the NVRAM failure enabled.
- Stop exporting LUNs over FCP.
- Send error messages to the console stating that Data ONTAP took the LUNs offline or that NFS file handles are stale (This is also useful if the LUN is accessed over NAS protocols.)

If desired, a second feature that renames certain files that the system administrator or DBA may not want to be made accessible to the network until after they have been carefully examined can be used to provide additional protection. This feature is controlled by the presence or absence of the file `/etc/nvfail_rename` (stored on the root volume of the storage system). The format of the `/etc/nvfail_rename` file is simply the name of a file found on the storage system, one file name per line; if this file exists, each file listed in it is renamed by having the string ".nvfail" appended to it.
For example, if the following files exist on the storage system:

db2filer:/vol/dbdata/db2inst1/NODE0000/SQL0001/SQLSPCS.1
db2filer:/vol/dbdata/db2inst1/NODE0000/SQL0001/SQLSPCS.2

If the file `/etc/nvfail_rename` exists and looks something like this:

/vol/dbdata/db2inst1/NODE0000/SQL0001/SQLSPCS.1
/vol/dbdata/db2inst1/NODE0000/SQL0001/SQLSPCS.2

If a NVRAM failure has occurred, the files listed in the file `/etc/nvfail_rename` will be renamed to:

db2filer:/vol/dbdata/db2inst1/NODE0000/SQL0001/SQLSPCS.1.nvfail
db2filer:/vol/dbdata/db2inst1/NODE0000/SQL0001/SQLSPCS.2.nvfail

Since this occurs before the storage system begins providing network service, these files will no longer have the same file names they had before the error occurred. As a result, the DB2 Database Manager will not be able to open the files needed (because they no longer exist) and an error will occur. Once the NVRAM problem has been corrected, you can restore use of the Storage System by stopping the DB2 Database Manager, renaming the affected files, and then restarting the DB2 Database Manager.

Note: To ensure that a DBA becomes aware of an NVRAM failure regardless of when any user attempts to access any database stored on a storage system, you may want to create a `/etc/nvfail_rename` file that contains the names of every file on the storage system that is being used by a particular DB2 database.

11. Caveats

The information presented in this document has been tested by NetApp using only a limited set of hardware and software options. Therefore, your experience may differ from that presented here. If you have any problems with the techniques shown in this document, please contact the author.

12. Important Links

1. Sourceforge page for Linux iSCSI driver download – *http://sourceforge.net/projects/linux-iscsi*
2. Qlogic HBA driver download page: *http://www.qlogic.com/products/iscsi_products_hba.asp*
3. Qlogic HBA documentation: *http://www.qlogic.com/support/product_resources.asp?id=341*
4. Emulex support Web site for driver/firmware download: *http://www.emulex.com/ts/indexemu.html*
5. DB2 Product Manual "Quick Beginning for DB2 Server": *(http://www-306.ibm.com/software/data/db2/udb/support/manualsv8.html)*
6. HP - Fibre Channel driver software downloaded page: (http://www.hp.com/go/softwaredepot)
7. HP Home page for Fibre Channel adapter documentation: (http://docs.hp.com/en/netcom.html)
8. HP-UX iSCSI Software Initiator Support Guide: HP-UX 11i v1 & 11i v2: (http://www.hp.com/go/softwaredepot)
9. NetApp Documentation
    a. Software download page on NOW: (http://now.netapp.com/NOW/cgi-bin/software
    b. FCP Solaris Attach Kit 2.0 Installation and Setup Guide (http://now.netapp.com/NOW/knowledge/docs/hba/sun/relhbas20/pdfs/setup.pdf)
    c. SAN Host Attach Kit 1.0 for Fibre Channel Protocol on HP-UX Installation and Setup Guide (*http://now.netapp.com/NOW/knowledge/docs/hba/fcp_hp-ux/relhp-ux10/html/index.shtml)*
    d. NetApp FCP Quick Reference Commands-Data ONTAP 6.5 & HP-UX 11i (*http://now.netapp.com/NOW/knowledge/docs/san/fcp_iscsi_config/QuickRef/FCPHP-UX65QuickRef.pdf)*
    e. NetApp Fibre Channel Configuration Guide (*http://now.netapp.com/NOW/knowledge/docs/san/fcp_iscsi_config/QuickRef/FCPConfigurationGuide.pdf)*
    f. Block Access Management Guide for FCP (*http://now.netapp.com/NOW/knowledge/docs/ontap/rel70rc/pdfs/ontap/bsagfcp.pdf)*
    g. The Compatibility and Configuration Guide for NetApp FCP and iSCSI Products (*http://now.netapp.com/NOW/knowledge/docs/san/fcp_iscsi_config/index.shtml)*
    h. Configuring NetApp storage systems with iSCSI Initiators: http://now.netapp.com/knowledge/docs/hba/iscsi/setup.pdf
    i. *iSCSI IBM AIX Support Kit 1.1 Installation and Setup Guide*

http://now.netapp.com/NOW/knowledge/docs/hba/iscsi/aix/iscsi_aix_1.1/reliscsiaix11/pdfs/setup.pdf