Technical Report

IBM DB2 for UNIX: Backup and Recovery Using NetApp Technologies

Bobby Oommen and Jawahar Lal, NetApp
Updated April 2011 | TR-3114

EXECUTIVE SUMMARY

This technical report describes how to use the NetApp storage system's Snapshot™ and SnapRestore® features to back up and restore a DB2 database in UNIX® environments and how to meet the challenges posed by shrinking backup and recovery windows for a mission-critical application. Regardless of the size of the database or the level of activity on the NetApp storage system, a Snapshot copy takes only a few seconds to create. The SnapRestore feature dramatically improves the system availability by providing the ability to recover in minutes, instead of hours or days.

TABLE OF CONTENTS

1 INTRODUCTION

Today, businesses have operations across the globe, and they are required to keep their mission-critical applications running 24/7. They also expect application performance to be maintained during routine maintenance, such as backup and recovery, regardless of the data growth rate, which can sometimes be very high. Backup windows are shrinking and the amount of data that needs to be backed up is increasing; therefore, defining a point in time at which a backup can be taken with minimal effect on system performance and availability can be a complex task.

The following factors make traditional backup and recovery methods very challenging:

- **System Performance Impact.** Database backup operations typically have a significant effect on the performance of a production system because they place a high load on the database server, the storage system, and the underlying network during the backup process.
- **Shrinking Backup Windows.** Enterprises expect maximum availability of their applications. Defining an appropriate window for creating backup images can present a challenge when the database needs to be accessible 24/7.
- **Rapid Data Growth.** Enterprise data is growing exponentially. Therefore, longer windows are needed to back up the data. Organizations that can no longer afford an extended window for backups are forced to invest heavily in the backup infrastructure to keep the backup window short. Growing databases also require more tape media or disk storage space for backup images. Incremental backups can address these issues, but longer restore times make them unacceptable.
- **Decreasing Mean Time to Recover (MTTR).** The mean time to recover is the time needed to recover from a database failure. The MTTR can be divided into two parts: the time that is necessary to restore the database from a backup image and the time that is necessary to perform a roll-forward recovery of the database. The roll-forward recovery time depends on the number of archive and active logs that need to be reapplied to the database after it has been restored. A NetApp Snapshot™ copy is a local image of the data.

NetApp Snapshot and SnapRestore technologies offer unique features to address the challenges presented by traditional backup and recovery methods.

2 PURPOSE AND SCOPE

This technical report describes how Snapshot and SnapRestore technologies can be used to back up and restore an IBM® DB2 database. Specifically, this report covers:

- Key advantages of NetApp Snapshot technology
- Infrastructure required to integrate the DB2 ESE with a NetApp storage system
- Overview of DB2 database backup and recovery terminologies
- Backup of a DB2 database using Snapshot technology
- Restoring a DB2 database using SnapRestore technology

3 OVERVIEW

3.1 SNAPSHOT AND SNAPRESTORE TECHNOLOGY

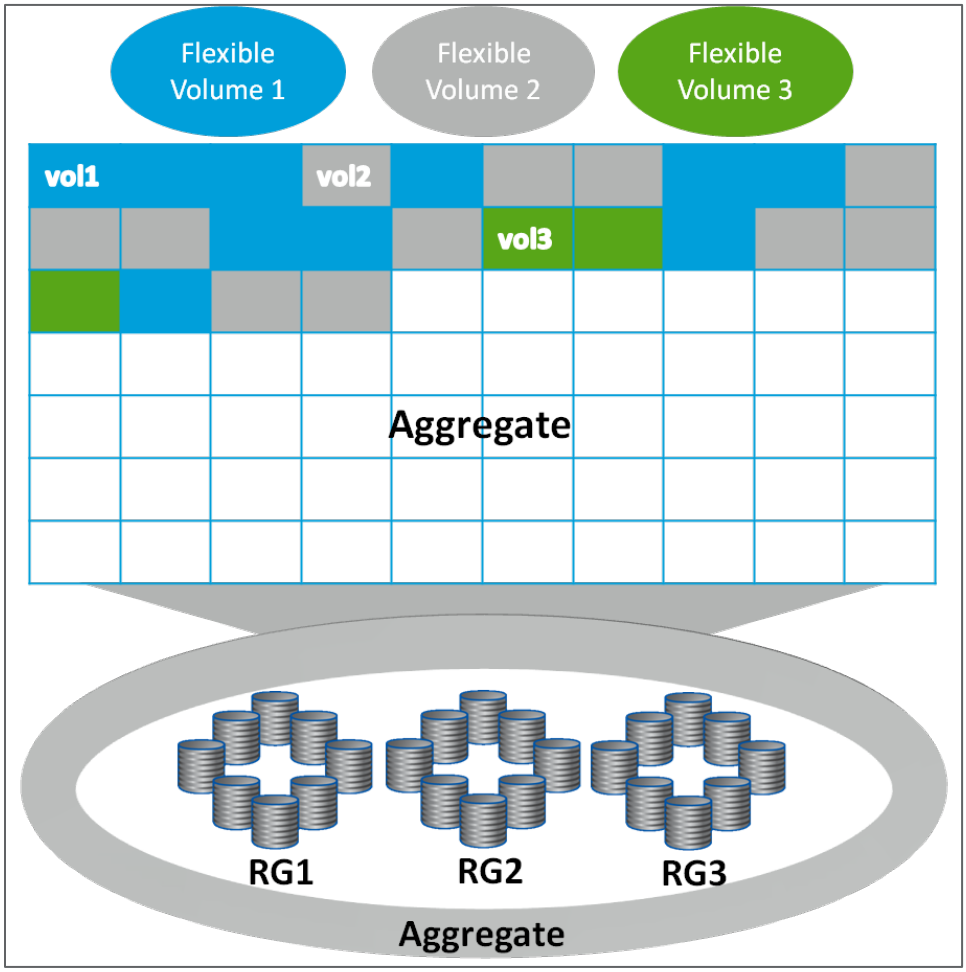
The core of the NetApp approach to database backup and recovery is the use of Snapshot technology. NetApp Snapshot technology is a feature of the NetApp WAFL® (Write Anywhere File Layout) storage virtualization technology that is part of NetApp Data ONTAP®, the microkernel that ships with every NetApp storage system. A Snapshot copy is a locally retained point-in-time “frozen” image of a WAFL volume that provides easy access to old versions of files, directory hierarchies, and/or logical unit numbers (LUNs). The high performance of NetApp Snapshot copies makes them highly scalable. A Snapshot copy takes only a

few seconds to create, regardless of the size of the volume or the level of activity on the NetApp storage system. After a Snapshot copy has been created, changes to data objects are reflected in updates to the current version of the objects, as if the Snapshot copy did not exist. Meanwhile, the Snapshot versions of the data remain completely stable. Therefore, Snapshot copies incur no performance overhead; users can comfortably store up to 255 Snapshot copies per WAFL volume, all of which can be accessible as read-only, online versions of the data.

Snapshot technology can be used to create an online or offline database backup in seconds. The time needed to create a Snapshot copy is independent of the size of the database because Snapshot technology does not move data blocks. Snapshot copies vastly improve the frequency and reliability of backups, because they incur virtually no performance overhead and can be safely created while a database is up and running. Customers running a DB2 database on a NetApp storage system typically take several Snapshot copies throughout the day.

A NetApp Snapshot copy also provides key advantages for the database recovery operation. The NetApp SnapRestore feature of Data ONTAP provides a way to restore the entire database to the state it was in at the point in time when any available Snapshot copy was taken. Copying data is not involved; therefore, a significant amount of time is saved as the file system is returned to its earlier state. The restore process can be done in a few minutes, regardless of the size of the database. In addition, when low-impact Snapshot backups are created frequently throughout the day, fewer transaction logs need to be reapplied as part of the recovery process, resulting in a dramatic reduction in recovery time. Figure 1 illustrates the logical relationship between aggregate, volumes, and disks.

Figure 1) Logical relationship between aggregate, volumes, and disks.



3.2 KEY ADVANTAGES

The Snapshot technology offers various features for DB2 customers including fast backup and restore, high availability and reliability, better performance, and maximum storage efficiency.

- **Fast Backup.** To meet the challenge posed by shrinking backup windows, the Snapshot feature of Data ONTAP is extremely useful. A Snapshot copy of a database 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. This dramatically reduces the database backup window from hours to seconds and helps DBAs to schedule frequent low-impact database backups.
- **Quick Recovery.** Using the Data ONTAP SnapRestore command, an entire database can be restored in a matter of seconds from a Snapshot backup. Copying data is not involved; therefore, an incredible amount of time is saved when a database is returned to the state it was in at the time the Snapshot copy was created. Additionally, because Snapshot copies can be taken quickly and a large number of Snapshot copies can be retained, the amount of time needed to perform a roll-forward recovery operation against a database can be greatly reduced.
- **High Availability.** The need for 24/7 availability is fast becoming a reality for organizations of all sizes. Companies cannot tolerate scheduled downtime, nor can they afford extended periods of slow system response that are often caused by traditional database backup methods. Snapshot copies, on the other hand, can be taken in a matter of seconds without any impact on system response time. This leads to high availability and uninterrupted system response.
- **High Reliability.** The RAID architecture used for NetApp storage systems is unique and provides greater reliability than direct-attached storage. If a RAID member disk fails, it is automatically reconstructed (using parity disk data) without any user intervention. NetApp supports single parity as well as NetApp RAID-DP® (double parity RAID). RAID-DP is considered approximately 10,000 times more reliable than traditional RAID. For more detail on RAID-DP, refer to the technical report "RAID-DP: Implementation of RAID Double Parity for Data Protection."
- **Uninterrupted System Response.** Because a Snapshot copy is just a frozen image of the file system at a specific point in time, the process of creating a database backup with a Snapshot copy does not require the actual copying of data; therefore, it has virtually no impact on system response time.
- **Minimum Storage Required for Backup Images.** Two Snapshot copies taken in sequence differ from one another by the blocks added or changed in the time interval between the two. This block-incremental behavior minimizes the amount of storage space consumed.
- **Storage Savings with NetApp FlexClone®.** NetApp FlexClone is another feature based on Snapshot technology that can offer DB2 customers large storage savings in testing, development, and virtualized environments. By using FlexClone technology, customers can create a full database clone in a matter of minutes without requiring additional storage. For more information on FlexClone, refer to "A Thorough Introduction to FlexClone Volumes" and "DB2: Cloning a Database Using NetApp FlexClone Technology."

Snapshot backups are stored on the same NetApp storage system as the database. Therefore, NetApp recommends using Snapshot backups as a supplement, not a replacement, for backups to a second location, whether backing up to disk or to tape. Although backups to a second location are still necessary, only a slight probability exists that these backups will be needed for restore and recovery. Most restore and recovery actions can be handled by using SnapRestore. Restores from a second location (disk or tape) are necessary only in situations in which the primary storage system holding the Snapshot copies is damaged or if a database needs to be restored from a backup that is no longer available in the form of a Snapshot copy—for instance, a two-month-old backup.

For further information on Snapshot and SnapRestore features, visit the NetApp Support (NOW™) Web site at now.netapp.com. Also refer to "File System Design for an NFS File Server Appliance" by Dave Hitz, James Lau, and Michael Malcolm and "Data Protection Online Backup and Recovery Guide."

Data ONTAP 7G and later versions support aggregates, flexible volumes, and LUNs as logical storage layers. An aggregate is a RAID-level physical pool of storage and possesses its own RAID configuration, plex structure, and set of assigned disks. Within each aggregate, you can create one or more flexible

volumes, the logical file systems that share the physical storage resources, the RAID configuration, and the plex structure of that common containing aggregate. An aggregate can be created by executing the following command on a NetApp storage system:

```
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 created by the command implicitly.
- `nDisk` identifies the number of disks used for the aggregate.
- `DiskSize` identifies the size of the disks being used for the aggregate.

Note: Parameters shown in angle brackets (< >) are optional; parameters or options shown in square brackets ([ ]) are required and must be provided; a comma followed by ellipses (…) indicates that the preceding parameter can be repeated multiple times.

For example, to create an aggregate named `dbaggr1` that has 10 disks, each 300GB in size, execute the following command on the NetApp storage system:

```
aggr create dbaggr1 –f –t raid_dp –r 10 10@300
```

A NetApp FlexVol® volume, also known as a flexible volume, is a logical storage container. A FlexVol volume resides inside an aggregate and delivers optimum performance by using all of the disk spindles available to the aggregate. A FlexVol volume can be as small as a few megabytes and as large as the aggregate itself. Create a FlexVol volume by executing the following command on a NetApp storage system:

```
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, and so on.

For example, to create a FlexVol volume named `dbdata` that is 10GB in size and resides in an aggregate named `dbaggr1`, execute the following command on the NetApp storage system:

```
vol create dbdata dbaggr1 10G
```

One or more logical unit numbers (LUNs) can be created within a volume. LUNs are used to map NetApp storage in SAN environments. A LUN is created by executing the following command on a NetApp 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 the LUN.

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

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

Note: The LUN created in this example is accessed by a Linux® host.

3.3 OVERVIEW OF DB2 ADVANCED COPY SERVICES

In a traditional backup or restore operation, the database manager copies data to or from a disk or a storage device using operating system calls. DB2 Advanced Copy Services (ACS) enable the use of Snapshot or fast copying technology of a storage device to perform the data copying part of backup and restore operations. Using the storage device to perform the data copying makes the backup and restore operations extremely fast. In addition to Snapshot backup and restore, DB2 ACS offers the following key features:

- Multipartition DB2 UDB database support
- Centralized configuration using the Configuration Wizard
- Multiple Snapshot backups
- Policy-based management of Snapshot backups

To perform Snapshot backup and restore operations, a DB2 ACS API driver is required. A DB2 ACS API driver for the NetApp storage system is integrated into the IBM Data Server. For information on supporting other storage hardware, refer to DB2 Advanced Copy Services (ACS) Supported Operating Systems and Hardware.

4 REQUIREMENTS AND ASSUMPTIONS

4.1 GENERAL ASSUMPTIONS

The following general assumptions about the audience were made in developing this technical report:

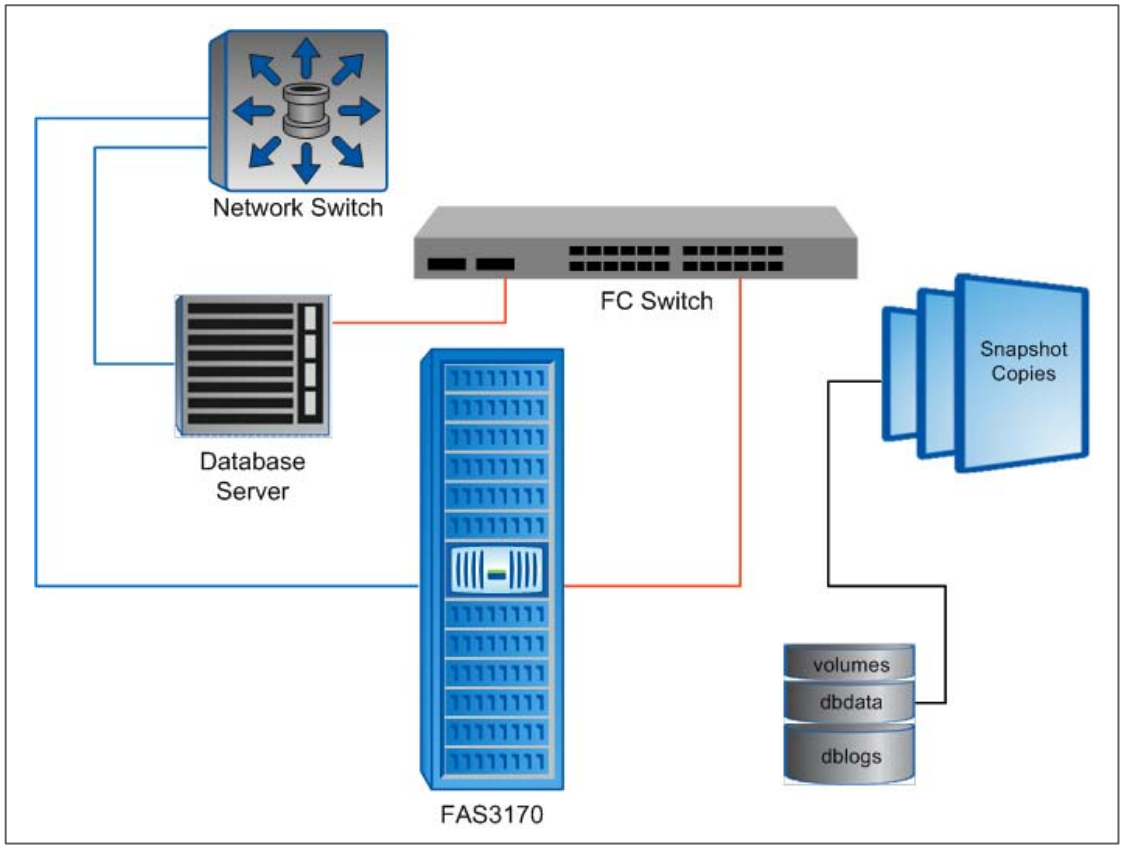
- Familiar with DB2 ESE version 9.x and its backup and recovery mechanisms
- Familiar with the operation of a NetApp storage system and its Data ONTAP microkernel
- Some knowledge of the flavor of UNIX used in the environment

To produce the data for this document, we installed the DB2 Enterprise Server Edition, version 9.5, on a UNIX host and created a database with its tablespace containers located on NetApp storage system FlexVol volumes. The following environment assumptions were made:

- The name of the UNIX host used for the DB2 database is `db2srv1`.
- The name of the DB2 administrator account is `db2inst1`.
- The name of the DB2 instance is `db2`.
- The name of the database is `myproddb`.
- The name of the NetApp storage system is `ntap1`.
- The name of the aggregate on the NetApp storage system is `dbaggr1`.
- The database's table data is stored on a FlexVol volume named `dbdata`.
- The database's transaction logs are stored on a FlexVol volume named `dblogs`.
- The NetApp storage system's root volume, named `/vol/vol0`, is mounted on the database server using a mount point named `/mnt/ntap1`.

The steps and scripts outlined in this document may require significant modifications to run under your database and host UNIX environment. Figure 2 illustrates a simple backup architecture for a DB2 database in a UNIX-NetApp storage system environment.

Figure 2) Backup and restore architecture for a DB2 database on a NetApp storage system.



4.2 PROVIDE REMOTE SHELL ACCESS TO THE NETAPP STORAGE SYSTEM

Remote shell (RSH) access allows secure access to the NetApp storage system from a remote host. If you plan to execute Data ONTAP commands from the database server, then you must enable RSH access to the NetApp storage system by completing the following steps.

1. Enable the RSH access feature by executing the following command on the NetApp storage system:

```
options rsh.enable on
```

2. Add an entry in the `/etc/hosts.equiv` file found on the NetApp storage system for the database host and the user. Entries in the `/etc/hosts.equiv` file should look similar to this:

   [`HostName`] [`UserName`]

   Where
   - `HostName` identifies the name of the database host from where the `rsh` command is executed.
   - `UserName` identifies the name of the user who wants to execute the `rsh` command.

3. For example, to create an entry for a database server named `db2srv1` and a user named `db2inst1`, add the following line to the `/etc/hosts.equiv` file on the NetApp storage system:

```
db2srv1 db2inst1
```

Once the appropriate entry has been added to the `/etc/hosts.equiv` file, a user can execute Data ONTAP commands directly from the host. The format used to execute Data ONTAP commands from the host looks similar to this:

```
rsh –l [UserName]:[Password] [StorageSystemName] [DOTCommand]
```

Where

- `UserName` identifies the name of the user who wants to execute the `rsh` command.
- `Password` identifies the password of the user who wants to execute the `rsh` command.
- `StorageSystemName` identifies the name of the storage system.
- `DOTCommand` identifies the Data ONTAP command.

For example, to obtain the RSH option setting on a NetApp storage system that has an entry for the database server in its `/etc/hosts.equiv` file, execute the following command on the database server:

```
rsh –l root:prodna1 ntap1 options rsh
```

Note: If a console password has not been set, RSH access is allowed, even if there is no entry in the `hosts.equiv` file.

4.3 SEPARATE DATA FROM TRANSACTION LOG FILES ONTO THE NETAPP STORAGE SYSTEM

NetApp Snapshot technology works at the volume level. Therefore, if you plan to incorporate Snapshot, SnapMirror®, or SnapRestore technology into your backup and recovery strategy, make sure that the database's data and transaction log files are stored on separate volumes on the NetApp storage system. If a database recovery operation becomes necessary, maintaining separate volumes for data and logs enables you to easily restore the database files from the appropriate Snapshot copy and roll forward using the database's transaction logs.

You can move transaction log files to another volume on the storage system by updating the database configuration parameter named `NEWLOGPATH`. To update this parameter, execute the following command on the database server:

```
db2 "UPDATE DB CFG FOR [DatabaseName] USING NEWLOGPATH [NewLogLocation]"
```

Where

- `DatabaseName` identifies the name of the database to be created.
- `NewLogLocation` identifies the name assigned to the mount point to be used for the database transaction logs.

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

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

5 OVERVIEW OF DB2 COMPONENTS USED FOR BACKUP AND RECOVERY

The main purpose of creating a database backup is to have a copy of mission-critical application data available in case the original data becomes corrupted or unavailable. By using a backup copy, application data can be returned to the state it was in at the point in time when the backup image was created. Before looking at how Snapshot copies can be used to create database backup images, it helps to be familiar with the components that DB2 uses for backup and recovery.

5.1 TRANSACTION LOG FILES

DB2 keeps track of all changes made to a database's data and its objects in the transaction log files. Thus, transaction log files can be used to recover a database. Depending on how it has been configured, a DB2 database can use two types of transaction log files:

- **Active/online log files.** Log files with contents that have not yet been applied to the database's data files are called active or online log files. The online log files are created in the active database log directory and are required to perform crash as well as roll-forward recovery.
- **Archive/offline log files.** Log files with contents that have been applied to the database's data files are called archive log files. They are not required for crash recovery but are crucial for roll-forward recovery. An active log file becomes a candidate for archiving as soon as its contents are written to the data files, and it may be moved automatically from the active log directory to an archive location.

5.2 LOGGING MODES

DB2 supports two different techniques for managing log files:

- **Circular logging.** Only online logs are retained, and log files are used in a round-robin fashion. The online logs are used only for crash recovery. The primary means of database recovery is version recovery from a full database backup copy. This type of database logging is called circular logging, and the database is called a nonrecoverable database. Roll-forward recovery is not possible. For circular logging, the `logarchmeth1` and `logarchmeth2` database configuration parameters are set to the value `OFF`.
- **Archive logging.** Both the online and the archive logs are retained. The online logs are used for crash recovery, and both the online and archive logs are used for roll-forward recovery. The database with this type of logging is called a recoverable database, and it not only supports roll-forward recovery but also supports tablespace-level backup and restores. Archival logging can be enabled by setting the `logarchmeth1` and `logarchmeth2` database configuration parameters to a value other than the value `OFF`.

For example, to enable archive logging for a database named `prod` and to store archive logs on another volume named `dbarchive` that is mounted on a mount point named `/mnt/dbarchive`, execute the following command on the database server:

```
db2 update db cfg for prod using logarchmeth1 disk:/dbarchive
```

5.3 RECOVERY HISTORY FILE

DB2 uses a special type of log file, called the recovery history file, as a repository for tracking various database activities such as backup, restore, roll forward, alter, rename, quiesce tablespace, load, drop, reorganization of table, and update of table statistics. View entries in this file by executing the `list history` command.

5.4 RECOVERY METHODS

DB2 supports the following database recovery methods.

- **Crash recovery.** Database transactions (or units of work) can be interrupted unexpectedly. If a failure occurs before all of the changes that are part of the unit of work are completed and committed, the database is left in an inconsistent and unusable state. Crash recovery is the process by which the database is returned to a consistent and usable state. This is done by rolling back incomplete transactions and completing committed transactions that were still in memory when the crash occurred. (Transaction records and corresponding commit records stored in transaction log files are used to return

the database to a consistent state.) The order in which statements are rolled back is the reverse of the order in which they were originally executed.

- **Version recovery.** This type of recovery allows the restoration of a previous version of a database using a backup copy of the database that was created by the DB2 backup utility. This method of recovery is normally used for restoring a nonrecoverable database. You can also use this method to restore a recoverable database if the DB2 restore utility is invoked with the `without rolling forward` option specified. A version recovery reconstructs an entire database using a previously created backup copy.
- **Roll-forward recovery.** Roll-forward recovery extends version recovery by using full database backups in conjunction with archive and active log files. When a roll-forward recovery operation is performed, the database must first be restored from a backup copy, and then transaction records stored in log files are applied to the restored database. This procedure returns a database or a tablespace to the state it was in at a particular point in time. The roll-forward recovery requires enabling of archival logging.

5.5 DB2 WRITE SUSPEND AND WRITE RESUME

DB2 Write Suspend and Write Resume commands allow users to suspend write operations temporarily. When the `set write suspend` command is executed, all writes to a database and its corresponding transaction log files are suspended. Read-only transactions are able to continue executing; however, some transactions may wait if they require disk I/O (for example, flushing dirty pages from the buffer pool or flushing records from the log buffer). These transactions proceed normally once the write operations on the database are resumed. When using Snapshot copies for database backup, NetApp recommends enabling database application consistency by suspending writes to the database just before taking the Snapshot copy.

To suspend write activity for a DB2 database, execute the following command on the database server:

```
db2 set write suspend for database
```

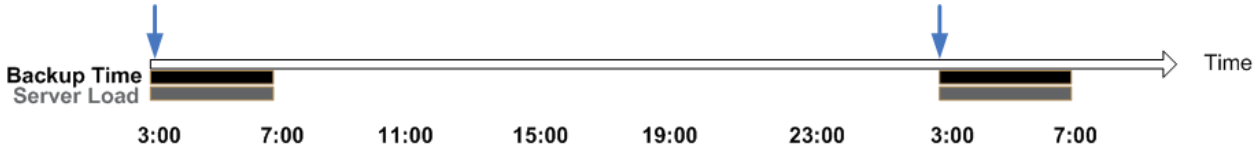
To resume write activity for a DB2 database for which I/O has been suspended, execute the following command on the database server:

```
db2 set write resume for database
```

6 BACK UP AND RESTORE USING SNAPSHOT TECHNOLOGY

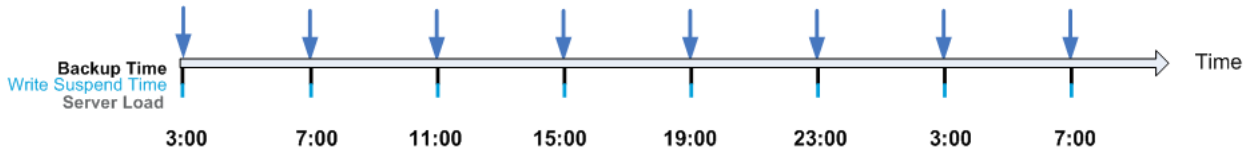
A conventional database backup to tape generates a significant load on both the database server and the storage system and takes several hours to complete. The system availability and performance are decreased while the backup is being performed. Figure 3 illustrates the backup time, the duration of write suspend, and the duration of the load generated by the backup process in a traditional backup environment. In these types of environments, backups are scheduled during off-peak hours, and frequent backups are not possible.

Figure 3) Conventional backup to tape.



On the other hand, a Snapshot-based backup copy can be created in a few seconds, and it has virtually no effect on the database server or on the NetApp storage system. This low impact on the system enables DBAs to take Snapshot-based backups much more frequently; for example, every n^{th} hour. Figure 4 illustrates the backup time, the duration of the write suspend, and the load duration for a NetApp Snapshot-based backup environment. The actual backup time is in seconds; therefore, frequent backups are possible.

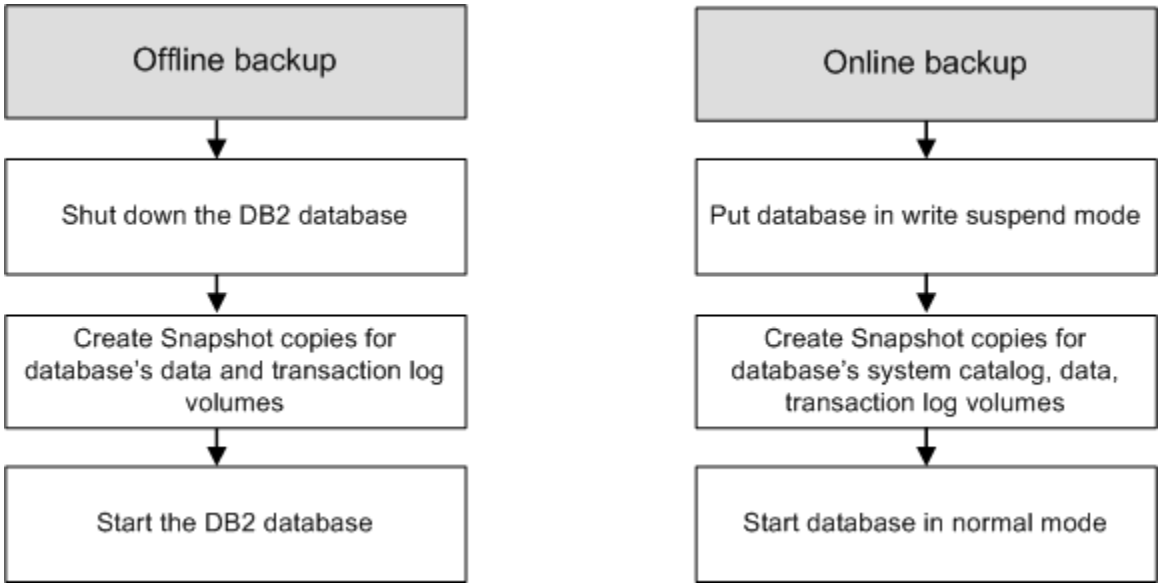
Figure 4) Database backup using Snapshot technology.



Depending on the retention policy used, a certain number of Snapshot copies are available to restore the database. A higher frequency of backups provides a more flexible restore strategy and a shorter database restore time. The shorter time frame between backups reduces the number of transaction logs that need to be applied during a roll-forward recovery operation.

Online and offline backups of a DB2 database can be created by using a Snapshot copy. If the database remains online, it must be placed in write-suspend mode before the Snapshot copies of the FlexVol volumes are created. For an offline backup, all application connections to the database need to be terminated, and the database should be inactive. Write-suspend mode is not applicable for offline backups. Figure 5 illustrates the basic steps required to create offline and online database backups.

Figure 5) DB2 database backup using NetApp Snapshot technology.



When you deploy a database on NetApp storage, you have the following three options available to back up the DB2 databases:

- Scripts or Command Line option
- NetApp SnapCreator framework option
- Integrated Snapshot feature for DB2 9.5 and higher

The following sections describe the steps required to create an offline and an online database backup using each option.

6.1 DATABASE BACKUP AND RESTORE USING SCRIPT OR COMMAND LINE OPTIONS

This section describes a very basic approach that allows customers to use the Data ONTAP commands to create a database backup. These commands can be scripted and used with existing backup and restore architecture.

DATABASE BACKUP

Creating an Offline Database Backup

A database is considered offline when all connections to it have been terminated and the database is no longer active. To take an offline backup of a DB2 database using Snapshot technology, complete the following steps:

1. To terminate all the connections to the database, execute the following command on the database server:

```
db2 force applications all
```

2. Create a database backup by creating a Snapshot copy for each NetApp storage system volume that is used by the database. To create a Snapshot copy of a FlexVol volume, execute the following command on the NetApp storage system:

   ```
   snap create [VolName] [SnapshotName]
   ```

   Where

   - `VolName` identifies the name of the volume the Snapshot copy is created for.
   - `SnapshotName` identifies the name of the Snapshot copy being created by this command.

   For example, to create a Snapshot copy for a FlexVol volume named `dbdata`, execute the following command on the NetApp storage system:

```
snap create dbdata dbdata_snp01
snap create dblogs dblogs_snp01
```

Note: You can also create a Snapshot copy for a FlexVol volume by executing an `rsh` command from the database server.

After completing these two easy steps, the database backup is finished, and you can start using the database. You can create scripts for these commands using scripting language such as shell script or Perl script. Refer to Appendix D for an example.

Creating an Online Database Backup

A database is considered active when one or more connections have been established to it. To conduct an online backup of a DB2 database, complete the following steps:

1. To temporarily suspend all write activity for the database, execute the following commands on the database server:

```
db2 connect to myproddb user db2inst1 using db2inst1
db2 set write suspend for database
```

Note: While a DB2 database is in the write suspended state, only the write activities for the database are suspended; read operations still continue uninterrupted.

2. When using a SAN or IP SAN (FCP, FCoE, or iSCSI) configuration, the host system has control of the file system on the NetApp storage system. Therefore, confirm the file system integrity before creating a Snapshot copy. The integrity is checked by executing the `Sync` system command on the server. When executed, the `Sync` command flushes all previously unwritten system buffers, including the modified super blocks, the modified inodes, and the delayed block I/O to the disk.

   For example, to flush all previously unwritten system buffers to disk before taking a Snapshot copy, open a telnet session to the database server, log in as the root user, and execute the following command:

```
Sync
```

3. Create a database backup by creating a Snapshot copy for each NetApp storage system volume that holds the data in the database. To create a Snapshot copy of a FlexVol volume, execute the following command on the NetApp storage system:

   ```
   snap create [VolName] [SnapshotName]
   ```

   Where

   - `VolName` identifies the name of the volume the Snapshot copy is created for.
   - `SnapshotName` identifies the name of the Snapshot copy being created by this command.

   For example, to create a Snapshot copy for a FlexVol volume named `dbdata`, execute the following command on the NetApp storage system:

```
snap create dbdata dbdata_snp01
snap create dblogs dblogs_snp01
```

4. Now that one or more Snapshot copies have been taken, execute the following command on the database server to resume writes to the database:

```
db2 set write resume for database
```

You can create scripts for these commands using scripting language such as shell script or Perl script. Refer to Appendix D for an example.

DATABASE RESTORE

As previously mentioned, you can easily integrate NetApp SnapRestore technology into your existing DB2 database backup and recovery architecture. Maintaining separate volumes on the NetApp storage system to hold the database's data and transaction log files provides more flexibility for the database restore operation. The recovery process using NetApp SnapRestore technology is similar to that of the recovery process using a database Snapshot option. However, it is much faster because the NetApp storage system only has to change a pointer on the file system rather than physically copy the files from the backup copy.

Restoring a Nonrecoverable Database From an Offline Backup Snapshot Copy

The NetApp Data ONTAP SnapRestore feature allows database recovery from Snapshot copies. You can restore a nonrecoverable DB2 database by using Snapshot copies of the FlexVol volumes that were created when the database was offline. This type of recovery is called version recovery. Complete the following steps to perform the version recovery of a database.

1. Disconnect all the applications and stop the database by executing the following commands:

```
db2 force applications all
db2stop
```

2. Remove all DB2 interprocess communication (IPC) resources by executing the following command on the database server:

```
~/sqllib/bin/db2_kill
~/sqllib/bin/ipclean db2inst1
```

3. Unmount the NetApp storage system volumes that are used for the database and that are mounted on the database server by executing the following command:

   ```
   umount [MountPoint]
   ```

   Where

   - `MountPoint` identifies the name of the mounted file system that needs to be unmounted.

For example, to unmount the NetApp storage system volume mounted on the mount point named `/mnt/dbdata`, execute the following command on the database server:

```
su – root
umount /mnt/dbdata
```

4. Recover the database by restoring the volumes that are used for the database's data and transaction logs using the Snapshot copies that are created as described in section 6.1. You can safely restore a database using Snapshot copies by executing the following command on the NetApp storage system:

   `snap restore –s [SnapshotName] [VolName]`

   Where

   – `SnapshotName` identifies the name of the Snapshot copy that is used to restore the volume.
   – `VolName` identifies the name of the volume that is being restored from its Snapshot copy.

   For example, to restore a volume named `dbdata` from a Snapshot copy named `dbdata_snp01`, execute the following command on the NetApp storage system:

```
snap restore -s dbdata_snap01 dbdata
snap restore -s dblogs_snap01 dblogs
```

On execution, the `snap restore` command reverts the FlexVol volume to the state it was in when the Snapshot copy was created.

5. After FlexVol volumes are restored, check the LUN mappings and status by executing the following commands on the NetApp storage system:

Note: Skip this step if your system configuration uses NFS.

```
lun show
lun show –m
```

The LUNs should be online and should have the same mapping as they had at the time the Snapshot copy was taken.

6. Refresh the HBA driver on the database server. For example, to refresh a QLogic FC HBA on a Linux host, execute the following commands:

```
modprobe –r qla2300
modprobe –v qla2300
```

For any other operating system (OS) and HBA, refer to the OS reference manual and the HBA installation guide.

7. After the restore process, mount the FlexVol volumes (or LUNs in the case of the FC/iSCSI) on the database server by executing the following command:

   `mount [MountPoint]`

   Where

   – `MountPoint` identifies the name of the file system that needs to be mounted.

   For example, to mount a FlexVol volume that has mount details specified in `/etc/fstab` of the Linux database host to a mount point named `/mnt/dbdata`, execute the following command:

```
mount /mnt/dbdata
```

After completing these steps, your database recovery is complete, and you can connect to the database and run the data verification scripts.

You can create scripts for these commands using scripting language such as shell script or Perl script. Refer to Appendix D for an example.

Restoring a Recoverable Database From an Online Backup Snapshot Copy

Roll-forward recovery for a recoverable database is a two-step process. In the first step, the database is restored from a full database backup. In the second step, transaction logs are reapplied. For roll-forward recovery, the database must run in archive mode, and active and archived logs must be available. To recover such a database using the SnapRestore feature, Snapshot copies for the volumes that hold the database's data and that are created as described in Section 6.2 are required. Complete the following steps to recover the database:

1. Disconnect all the applications and stop the database by executing the following commands:

```
db2 force applications all
db2stop
```

2. Remove all DB2 IPC resources by executing the following commands on the database server:

```
~/sqllib/bin/db2_kill
~/sqllib/bin/ipclean db2inst1
```

3. Unmount the NetApp storage system volumes that hold the database's data by executing the following command on the database server:

   `umount [MountPoint]`

   Where

   - `MountPoint` identifies the name of the mounted file system that needs to be unmounted.

   For example, to unmount a NetApp storage system volume mounted on a mount point named `/mnt/dbdata` on a Linux host, execute the following command on the database server:

```
umount /mnt/dbdata
```

4. As the first step in the roll-forward recovery, restore the volumes that are used for the database data from the Snapshot copies that are created as described in Section 7.2. Restore a volume by executing the following command on the NetApp storage system:

   `snap restore –s [SnapshotName] [VolName]`

   Where

   - `SnapshotName` identifies the name of the Snapshot copy that is used to restore the volume.
   - `VolName` identifies the name of the volume that is being restored from its Snapshot copy.

   For example, to restore a volume named `dbdata` from a Snapshot copy named `dbdata_snp01`, execute the following command on the NetApp storage system:

```
snap restore -s dbdata_snap01 dbdata
```

On execution, the `snap restore` command reverts the FlexVol volume to the state it was in when the Snapshot copy was created.

Note: For roll-forward recovery, only the volumes that hold the database's data are restored. Volumes that hold the database's transaction and archive logs are not restored.

5. After the FlexVol volumes are restored, check the LUN mappings and status by executing the following commands on the NetApp storage system:

Note: Skip this step if your system configuration uses NFS.

```
lun show
lun show –m
```

The LUNs should be online and should have the same mapping they had at the time that the Snapshot copy was taken.

6. Refresh the HBA driver on the database server. For example, to refresh a Qlogic FC HBA on a Linux host, execute the following commands:

```
modprobe –r qla2300
modprobe –v qla2300
```

For other operating systems and HBAs, refer to the OS reference manual and the HBA installation guide.

7. After restoring the database's data volumes, mount them on the database server by executing the following command:

   `mount [MountPoint]`

   Where

   – `MountPoint` identifies the name of the file system that needs to be mounted.

   For example, to mount a volume (or LUN device) that holds the database's data to a mount point named `/mnt/dbdata` on a Linux host, execute the following command:

```
mount /mnt/dbdata
```

This command assumes that you have specified mount details, including mount options for the volume (or LUN) in the `/etc/fstab` file on the database server.

8. The database is now restored to the state it was in before the backup Snapshot copies were created. To recover the database without any data loss, perform a roll-forward recovery by reapplying the archive and the active logs. Before you perform a roll-forward recovery, place the database in a roll-forward pending state by executing the following command on the database server:

`db2inidb [DBName] as mirror`

Where

– `DBName` identifies the name of the database that needs to be initialized.

For example, to put a database named `myproddb` in a roll-forward pending state, execute the following command on the database server:

```
db2inidb myproddb as mirror
```

9. Once the database is in a roll-forward pending state, reapply the logs and perform roll-forward recovery by executing the following command on the database server:

   `db2 "rollforward database [DBName] to end of logs and complete"`

   Where

   – `DBName` identifies the name of the database that needs to be initialized.

   For example, to perform a roll-forward recovery for a database named `myproddb`, execute the following command on the database server:

```
db2 "rollforward database myproddb to end of logs and complete"
```

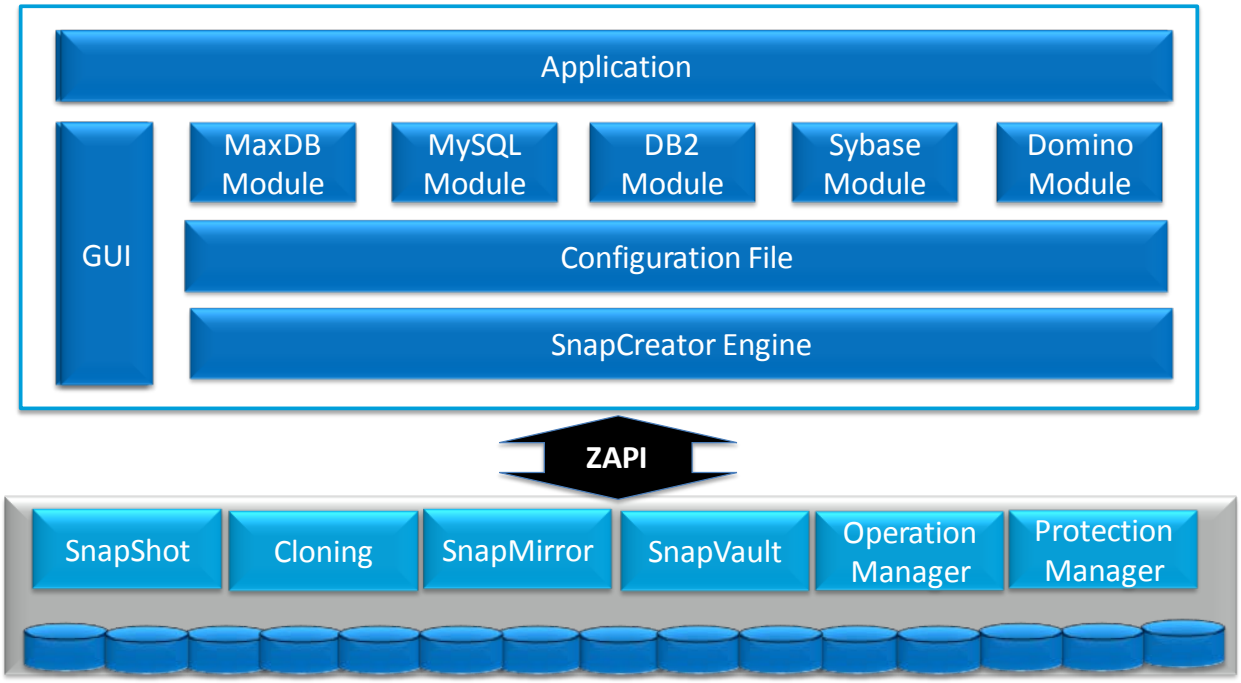
After completing all of these steps, the database is restored and ready to use. Now you can run some verification scripts to check your data.

You can create scripts for these commands using scripting language such as shell script or Perl script. Refer to Appendix D for an example.

6.2 AUTOMATING BACKUP AND RECOVERY OF A DB2 DATABASE USING SNAPCREATOR

The purpose of SnapCreator is to create a central framework that provides seamless integration with the DB2 application and NetApp storage Snapshot technology. Normally, using NetApp Snapshot technology to do the backup or recovery requires a hard-coded custom script that interfaces with the application and the NetApp storage. These custom scripts are written every day, over and over again, and are normally not reused, so the purpose of SnapCreator is to save time and to provide the most reliable solution possible. The application piece is usually unique. SnapCreator does not have a code to specifically handle application consistency. Instead, it has a framework (shown in Figure 6) in which you can integrate the DB2 module. This module handles the application consistency in SnapCreator. SnapCreator communicates with NetApp storage and performs various tasks, which include policy-based Snapshot management and integration with other NetApp products.

Figure 6) SnapCreator architecture diagram.



SnapCreator is made up of three components:

- **SnapCreator engine.** The SnapCreator engine is the main component of SnapCreator, and it runs on a central backup server or on the DB2 server where the database is installed. SnapCreator can also be integrated with an existing backup architecture. SnapCreator is easy to install, is flexible, and is also platform independent. SnapCreator communicates with the NetApp storage system by using ZAPI calls and does policy-based Snapshot management. SnapCreator has a robust error-handling process, and it can send alerts to an e-mail distribution or communicate with external monitoring systems.
- **SnapCreator configuration file.** The SnapCreator configuration file controls what SnapCreator does and can be customized based on your needs. SnapCreator distribution comes with a default configuration file and can be used as a template to create the configuration file for the database. More than one configuration file per database is supported, but only one file can be active at a time.
- **Application module (DB2 in this case).** The DB2 application module makes sure your Snapshot copy is crash consistent by putting the database in an I/O suspend mode during the Snapshot operation. Once the Snapshot copy is done, the DB2 module puts the database back into normal mode. Optionally, you

can use a customized script or commands to put the database on I/O suspend and resume modes. Refer to the SnapCreator Installation and Administration Guide for details.

INSTALLATION AND SETUP OF SNAPCREATOR FOR THE DB2 DATABASE

Refer to the SnapCreator Installation and Administration Guide for directions on the installation and basic setup for SnapCreator. The following section explains how to set up the DB2 database to use SnapCreator.

A SnapCreator configuration file must be created to back up or recover the DB2 database. The SnapCreator configuration file has different sections, and the following information shows the minimum required fields for each section.

Basic Configuration

- `SNAME=MYPRODDB.` This field defines the prefix for your Snapshot copy. The naming convention should be unique because Snapshot copies on the NetApp storage system are deleted according to the naming convention and retention policy used. As a best practice, NetApp recommends configuring the `SNAME` field as a database name, which, in this case, is `MYPRODDB`.
- `SNAP_TIMESTAMP_ONLY=Y`. When this field is set to `Y`, Snapshot copies end with a DB2 timestamp (`YYYYMMDDHHMMSS`).
- `VOLUMES=dbdata,dblogs`. These volumes are used by the database `MYPRODDB`. SnapCreator allows you to configure more than one volume or storage controller, and it enables a database-consistent Snapshot copy to be created across the volumes and storage controllers. The configuration for the volumes is `filer1:vol1,vol2,vol3;filer2:vol1;filer3:vol2,vol3`. The `MYPRODDB` database is on one storage controller and spans two volumes; therefore, the `VOLUMES` field can also be configured in the same way.
- `NTAP_USERS=ntap1:db2inst1/db2inst1`.To create a Snapshot copy or a SnapRestore operation on the NetApp storage system, create a user on the NetApp storage system with the necessary privileges. Refer to the SnapCreator 3.2 Installation and Administration Guide for directions on how to set up the user on the storage system. The `NTAP_USERS` field is configured as the list of appliances and their corresponding user names and passwords. For example, `filer1:joe/password1;filer2:bob/password2;filer3:ken/password3`. We only have one storage system for this setup, so it is configured in the same way. SnapCreator supports password encryption. For information on password encryption refer to the SnapCreator 3.2 Installation and Administration Guide.
- `TRANSPORT=HTTP`. SnapCreator supports both HTTP and HTTPS protocols for API communications.

Note: HTTPS requires `openssl-devel rpm` and, at this point in time, has only been tested with Linux and AIX.

- `PORT=80`. The port that SnapCreator uses to communicate with the NetApp storage controller(s) is normally 80 or 443. Because we are using HTTP for the TRANSPORT type, the port is configured as 80.

NetApp Options

- `NTAP_SNAPSHOT_RETENTIONS=daily:7,weekly:4,monthly:1`. This setting determines the number of NetApp Snapshot copies to retain for a given policy. Based on the setting, SnapCreator keeps at least seven daily backups, four weekly backups, and one monthly backup preserved on the NetApp storage system.
- `NTAP_SNAPSHOT_RETENTION_AGE=7`. This setting, which is in days, defines a retention age for Snapshot copies. If configured, Snapshot copies are deleted only if there are more than the number defined in `NTAP_SNAPSHOT_RETENTIONS` and if they are older than the retention age (in days). Even though this field is optional, NetApp recommends setting this field especially when you have an RPO requirement in a production environment.
- `NTAP_CONSISTENCY_GROUP_SNAPSHOT=Y`. This setting uses the Data ONTAP consistency group (CG) feature during the Snapshot operation and makes sure that all of the storage volumes of the database

have the same consistency point across the volumes and storage controllers. This setting is not required if you are creating Snapshot copies while the DB2 database is on write suspend mode. But NetApp recommends creating a CG Snapshot copy when you include files that are outside of the database (for example, stored procedures). Also, CG Snapshot copies are faster, and NetApp recommends using them when the database is on multiple data volumes and your application prohibits the database from being put on write suspend mode for more than seven seconds.

Note: Enabling this option requires `NTAP_CONSISTENCY_GROUP_TIMEOUT`.

- `NTAP_CONSISTENCY_GROUP_TIMEOUT=medium`. This setting controls how long the Snapshot creation process on the NetApp storage system waits for I/O fencing between volumes to finish. Before creating a CG Snapshot copy, all volumes must be quiesced (I/O fencing). The default setting and recommendation is medium (7 seconds); however, that may not be long enough depending on how many volumes you have and how many storage controllers are involved. The wait time in (seconds) for different modes is as follows: urgent is 2, medium is 7, and relaxed is 20. After the wait time has passed, SnapCreator times out and the database becomes available to the user or to the application.

Post Commands

- `POST_RESTORE_CMD1=db2inidb myproddb as mirror`
- `POST_RESTORE_CMD2=db2 "rollforward database myproddb to end of logs and complete"`

  These two fields are optional and can be configured if you want to execute a command after the database restore process. In this case, run the **`db2inidb`** command to restart the database. Configuring this field automates the database restart through SnapCreator. You can also restart the database by issuing the commands manually because of the inherent flexibility in SnapCreator.

Additional Modules

- `APP_NAME=db2`. This is the section that calls the DB2 module of SnapCreator to make sure the Snapshot copies are consistent with the database.

DB2 Options (UNIX Only)

- `DATABASES=myproddb:db2inst1`. This is the list of databases and the user name (`db1:user1;db2:user2`). NetApp recommends using the database instance ID as the user.
- `DB2_CMD=db2`. This is the path to the DB2 command, which is what we use to interact with the database.

BACKUP OF THE DB2 DATABASE USING SNAPCREATOR

Once the configuration file is created, you can create the Snapshot backup on the NetApp storage system using SnapCreator. When SnapCreator is executed with the Snap action, the following operations run in the background:

1. SnapCreator calls the DB2 module by using the SnapCreator configuration file and issues `write suspend` on the DB2 database `myproddb`.
2. SnapCreator does an inventory of all the configured volumes by using ZAPI calls on the NetApp storage system.
3. SnapCreator creates Snapshot copies for all of the configured volumes using the NetApp consistency group feature, which makes sure that all volumes are within the same I/O boundaries. It also makes sure that the Snapshot copies for all the volumes are done successfully.
4. SnapCreator calls the DB2 module again and issues a `write resume` operation, which puts the DB2 database into a normal mode.
5. SnapCreator applies the Snapshot management policies and deletes the expired Snapshot copies from the NetApp storage system.

6. Finally, SnapCreator sends alerts to the distribution list showing the status of the backup operation.

All of these operations are done in the background, and you don't have to script the process or manually issue commands on the database server or the underlying NetApp storage system. Use the following command to back up the database using SnapCreator:

```
./snapcreator --profile <profile name> --action <snap or restore> --policy <your policy> --verbose
```

Where

- `profile` is your profile name. In this case, the profile name is defined as the database name.
- `action` is the operation you are performing. In this case, the action is the Snap.
- `policy` is the backup policy. In this case, we select the policy as daily.
- `verbose` shows the SnapCreator operation on the console and is optional. SnapCreator saves the output to the logs regardless of whether or not you specify this option.

To create a database-consistent Snapshot copy for the `myproddb` database using SnapCreator with the `daily` policy, execute the following command:

```
./snapcreator --profile myproddb --action snap --policy daily --verbose
```

Refer to Appendix A for the output of the SnapCreator backup operation of the `myproddb` database.

Once the Snapshot copy is created on the NetApp storage system, you can list or view it by using SnapCreator. To list the Snapshot copy you just took on the NetApp storage system for the `myproddb` database, execute the following command:

```
./snapcreator --profile myproddb --action snaplist --policy daily
```

The SnapCreator Snapshot copy list shows which Snapshot copies are associated with the production database for the configured volumes. Refer to Appendix B for the sample output of the `snaplist` action.

DB2 DATABASE RESTORE USING SNAPCREATOR

Any production database may be subjected to data corruption either by a malicious process or by user error. To go back to a point in time before the data corruption, restore the database by using the latest available good Snapshot copy. SnapCreator can restore the DB2 database using the Snapshot copies from the NetApp storage system. The SnapCreator restore process is an interactive process that needs user input during the restore process. The restore process only lists the Snapshot copies for individual volumes that are taken through SnapCreator. Therefore, any other Snapshot copies either taken manually or through some other process on the NetApp storage system are ignored.

The following steps occur during a restore operation using SnapCreator:

1. The SnapCreator interactive restore menu prompts the restore of the configured volumes.
2. SnapCreator gives two views for the restore process: the Storage Admin view and the DBA view. The DBA view is recommended for users who come from a DBA background.
3. Once you are in the DBA view, SnapCreator lists the Snapshot copies on the NetApp storage system, and the user can select the appropriate Snapshot copy to restore to the configured volumes.
4. After all the volumes are restored, SnapCreator calls the POST commands to restart the database.

To restore the `myproddb` database using an earlier Snapshot copy taken through SnapCreator, execute the following command:

```
./snapcreator --profile myproddb --action restore --policy daily
```

Refer to Appendix C to see the output of the SnapCreator restore process.

The database is now restored to the state it was in before the backup Snapshot copies were created.

5. Perform a roll-forward recovery to recover the database without any data loss by reapplying the archive and the active logs. Before you perform a roll-forward recovery, place the database in a roll-forward pending state by executing the following command on the database server:

```
db2inidb [DBName] as mirror
```

Where

- `DBName` identifies the name of the database that needs to be initialized.

  For example, to put a database named `myproddb` in a roll-forward pending state, execute the following command on the database server:

```
db2inidb myproddb as mirror
```

6. After the database is in a roll-forward pending state, reapply the logs and perform roll-forward recovery by executing the following command on the database server:

   ```
   db2 "rollforward database [DBName] to end of logs and complete"
   ```

   Where

   - `DBName` identifies the name of the database that needs to be initialized.

   For example, to perform a roll-forward recovery for a database named `myproddb`, execute the following command on the database server:

```
db2 "rollforward database myproddb to end of logs and complete"
```

After completing all of these steps, the database is restored and ready to use. Now you can run some verification scripts to check your data.

You can create scripts for these commands using scripting language such as shell script or Perl script. Refer to Appendix D for an example.

The **POST** command in SnapCreator can be configured to restart the database, and this command is shown in the SnapCreator Configuration section of this document.

6.3 BACK UP AND RESTORE USING INTEGRATED SNAPSHOT OR DB2 ACS (DB 9.5 OR HIGHER)

DB2 Advance Copy Service (ACS) integration with DB2 9.5 or higher versions allows DB2 to take advantage of Snapshot capabilities of the underlying storage for its backup and restore operations. This feature allows the user to perform backup and restore operations based on the NetApp Snapshot copy from the native DB2 backup and restore commands. The integrated NetApp Data ONTAP SDK libraries enable DB2 to discover underlying storage volumes on the NetApp storage used by the database and establish volume-level relationships. During the backup process, the following operations are performed:

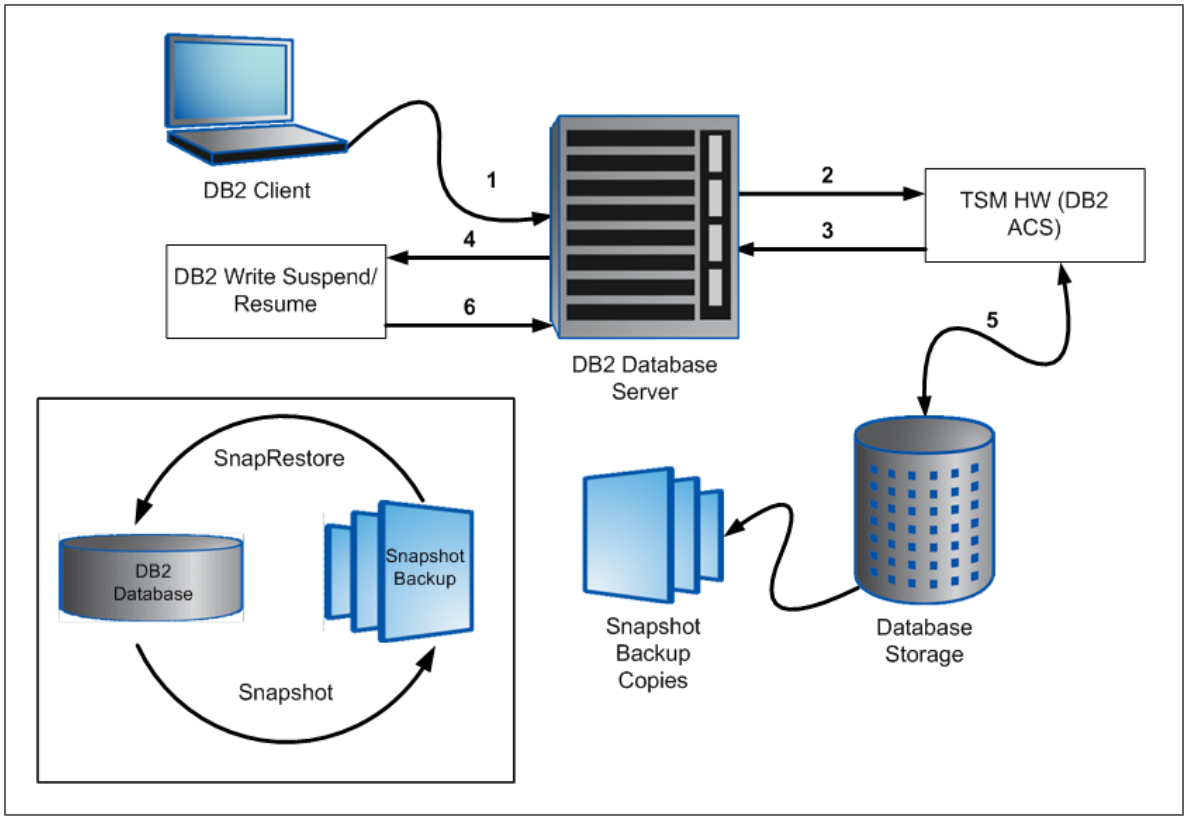
1. The user invokes the native DB2 backup command.
2. DB2 uses ACS for HW to identify volume relationships.
3. TSM provides the database container list.
4. DB2 suspends writes and requests a Snapshot copy.
5. TSM creates a Snapshot copy for HW.
6. DB2 resumes writes.

The Snapshot backups are allowed for single-node and multinode DB2 databases. For a multinode or Data Partitioning Feature (DPF) database, NetApp recommends using the `on ALL DBPARTITIONNUMS` option when invoking the backup command. The `on ALL DBPARTITIONNUMS` option allows DB2 to suspend and

resume I/O on the entire database, and the database backup is created across all of the database partitions at the same point in time.

Database backup steps vary based on the recoverability type of the created database. If archive logging is enabled, then the database is recoverable; otherwise, it is nonrecoverable. A nonrecoverable database backup can be created only after bringing the database offline. Roll-forward recovery is not permitted for a nonrecoverable database. For recoverable databases, backups may be done online or offline. After a database is restored, roll-forward recovery is required to recover a database to a given point in time. Figure 7 illustrates the Snapshot backup process flow for DB2 9.5 with the Snapshot options.

Figure 7) DB2 backup process flow with Snapshot backup options.



The following subsection describes the steps required for creating a backup for both nonrecoverable and recoverable databases.

Backup of a Nonrecoverable DB2 Database Using Integrated NetApp Snapshot Technology

To back up a single partition nonrecoverable database, execute the following commands on the database server with appropriate privilege:

```
db2 deactivate database [DatabaseName]
db2 backup db [DatabaseName] use snapshot
```

Where

- `DatabaseName` identifies the name of the database to be created.

For example, to create a Snapshot backup of a nonrecoverable database named `myproddb`, execute the following command on the database server:

```
db2 deactivate db myproddb
```

```
DB20000I The DEACTIVATE DATABASE command completed successfully.
db2 backup db myproddb use snapshot
Backup successful. The timestamp for this backup image is : 20100526175525
```

To back up a DPF database, execute the following command:

```
db2_all "db2 deactivate database [DatabaseName]"
db2 backup db [DatabaseName] on all nodes use snapshot
```

Where

- `DatabaseName` identifies the name of the database to be created.

For example, to create a Snapshot backup for a DPF database named `prod` and a second host named `db2srv2`, execute the following commands from the server:

```
db2_all "db2 deactivate db prod"
DB20000I The DEACTIVATE DATABASE command completed successfully.
db2srv1: db2 deactivate db ... completed ok
SQL1496W Deactivate database is successful, but the database was not
activated.
db2srv2: db2 deactivate db ... completed rc=2

db2 backup db prod on all nodes use snapshot
Part Result
---- ------------------------------------------------------------------------
0000 DB20000I The BACKUP DATABASE command completed successfully.
0001 DB20000I The BACKUP DATABASE command completed successfully.
Backup successful. The timestamp for this backup image is : 20100521091638
```

Backup of a Recoverable DB2 Database Using Integrated Snapshot Technology

Snapshot backups can be created for a recoverable database when the database is online or offline. Offline backups are performed in the same way for recoverable and nonrecoverable databases. To create an online Snapshot backup, execute the following command from the database server:

```
db2 backup db [DatabaseName] online use snapshot
```

Where

- `DatabaseName` identifies the name of the database to be created.

For example, to create an online Snapshot backup of the `myproddb` database, execute the following backup command on the database server:

```
db2 backup db myproddb online use snapshot
```

DB2 9.5 or higher provides a utility named `db2acsutil` to manage Snapshot backup copies. This utility can be used to list, to view the status, and to delete the backup copies. To view Snapshot backup copies for all databases and their status, execute the following command with the query option:

```
db2acsutil query status
```

For example, issue the following `db2acsutil` command:

```
db2acsutil query status
Instance  Database  Part Image Time   Status
=======================================
```

```
db2inst1   MYPRODDB          0 20100521092749 Repetitively restorable +
Destructively restorable

db2inst1   MYPRODDB          1 20100521092749 Repetitively restorable +
Destructively restorable
```

The online and offline backup process for a recoverable and nonrecoverable database can be summarized in a few easy steps, as illustrated in Table 1.

Table 1) Backup options for a recoverable and a nonrecoverable database (PROD).

Database Type	Offline	Online
Recoverable Database (Archive logging enabled)	db2 backup db prod use snapshot	db2 backup db myproddb online use snapshot
Nonrecoverable Database (Archive logging disabled)	db2 backup db myproddb use snapshot	N/A

Restore a Recoverable DB2 Database Using Integrated Snapshot Technology

Larger databases can take a significant amount of time to complete a conventional restore, which increases the database downtime. However, DB2 9.5 allows database restores from a Snapshot backup copy. This restore can be completed in seconds, regardless of the size of the database.

In DB2 9.5, Snapshot restores are invoked through DB2 ACS, and the Snapshot backup images are retrieved from their respective volumes. Then, these volumes get restored just like a traditional backup. However, the process used to perform this restore is slightly different.

The DB2 restore process invoked with the `USE SNAPSHOT` option performs the following internal operations:

1. Determine the volumes to restore.
2. Unmount the database file systems.
3. Perform volume-level restores on the NetApp storage system.
4. Remount the database file systems.
5. Place the database in a roll-forward pending state (for a recoverable database only).

The default behavior when restoring data from a Snapshot backup image is a `FULL DATABASE OFFLINE` restore of all paths that make up the database. These include containers in the database path (`DBPATH`) and the storage paths of the most recent Snapshot backup. If a timestamp is provided, then that Snapshot backup image is restored. `EXCLUDE LOGS` is the default for all Snapshot backup commands unless `INCLUDE LOGS` is explicitly stated. Therefore, if the `INCLUDE LOGS` option was not specified in the backup, then primary and mirror logs cannot be restored using that backup.

The Snapshot restore is a volume-level restore; hence, all data currently in the volumes is essentially destroyed and replaced with a Snapshot copy. For this reason, NetApp strongly recommends not sharing the volumes used by the database with other applications or users.

Prior to issuing a restore, make sure that all log paths are mounted and accessible. If a log path is not found, the restore uses the default log directory. This is the `SQLOGDIR` directory inside the database path.

The steps required to recover the database depend on its recoverability type. The following subsections describe the steps required to recover both recoverable and nonrecoverable databases.

Restore Based on Snapshot Technology for a Nonrecoverable Database

1. To perform a single partition database restore from a Snapshot backup copy, execute the following command:

```
db2 restore db [DatabaseName] use snapshot
```

Where

- `DatabaseName` identifies the name of the database to be created.

By default, DB2 takes the latest backup image from the storage system when the Snapshot option is used.

2. To restore the database from a specific Snapshot copy, specify the timestamp of the backup copy:

```
db2 restore db [DatabaseName]use snapshot taken at [BackupTimeStamp]
```

Where

- `DatabaseName` identifies the name of the database to be created.
- `BackupTimeStamp` identifies the time the database backup was created.

3. To obtain the timestamp for the intended Snapshot backup copy, execute the following command:

```
db2acsutil query db [DatabaseName]
```

Where

- `DatabaseName` identifies the name of the database to be created.

For example, to query the backup timestamp for a database named `myproddb`, execute the following `db2acsutil` command from the database server:

```
db2acsutil query db myproddb

Instance   Database   Part Image Time     Host       First Log
========== ========== ==== ============== ========== ==========
db2inst1   MYPRODDB          0 20100526121157 db2srv1          33
db2inst1   MYPRODDB          1 20100526121157 db2srv2          32
db2inst1   MYPRODDB          0 20100526121104 db2srv1          34
db2inst1   MYPRODDB          1 20100526121104 db2srv2          35
```

For example, to restore a database named `myproddb` using a backup copy based on a Snapshot copy created at 20100526121157, execute the following restore command:

```
export DB2NODE=0

db2 terminate

db2 DB20000I  The TERMINATE command completed successfully.

db2 restore db myproddb use snapshot taken at 20100526121157

SQL2539W  Warning!  Restoring to an existing database that is the same as the
backup image database.  The database files will be deleted.

Do you want to continue ? (y/n) y

DB20000I  The RESTORE DATABASE command completed successfully.

export DB2NODE=1

db2 terminate

DB20000I  The TERMINATE command completed successfully.

db2 restore db myproddb use snapshot taken at 20100526121157

SQL2539W  Warning!  Restoring to an existing database that is the same as the
backup image database.  The database files will be deleted.

Do you want to continue ? (y/n) y
```

```
DB20000I  The RESTORE DATABASE command completed successfully.
```

For a nonrecoverable database, the database recovery is complete after the Snapshot copy is restored.

Restore Based on Snapshot Technology for a Recoverable Nonpartitioned Database

1. To perform a restore for a recoverable database from a Snapshot backup copy, execute the following command:

   ```
   db2 restore db [DatabaseName] use snapshot
   ```

   Where

   - `DatabaseName` identifies the name of the database to be created.

For example, to restore a database named `myproddb` using the latest backup copy based on the Snapshot copy, execute the following command:

```
db2 restore db myproddb use snapshot

SQL2539W  Warning!  Restoring to an existing database that is the same as the
backup image database.  The database files will be deleted.

Do you want to continue ? (y/n) y

DB20000I  The RESTORE DATABASE command completed successfully.
```

After the restore is complete, the database is in a roll-forward pending state. To fully recover the database and bring the database out of the roll-forward pending state, a roll-forward command must be issued. Roll forward can be done to the end of the logs, to a point in time, or to the end of a backup. Prior to issuing a roll forward, make sure that all log files that are required to roll forward through are available.

2. To check which log files the backup image includes, issue a `db2acsutil query db` command:

   ```
   db2acsutil query db [DatabaseName]
   ```

   Where

   - `DatabaseName` identifies the name of the database to be created.

For example, to check the log file required for the restore of a database named `myproddb`, execute the following command:

```
db2acsutil query db myproddb
Instance   Database   Part Image Time     Host       First Log
========== ========== ==== ============== ========== ==========
svtdbm7    MYPRODDB          0 20100526180512 db2srv1.ibm.com         18
svtdbm7    MYPRODDB          0 20100526180410 db2srv1.ibm.com         10
```

3. To perform a roll-forward recovery to the end of the logs, execute the following command:

   ```
   db2 rollforward db [DatabaseName] to end of logs and complete
   ```

   Where

   - `DatabaseName` identifies the name of the database to be created.

For example, to perform a roll-forward recovery for a database named `myproddb`, execute the following command:

```
db2 rollforward db myproddb to end of logs and complete
```

You can also perform a point-in-time recovery by executing the following command:

```
db2 rollforward db [DatabaseName] to <timestamp> and complete
```

Where

- `DatabaseName` identifies the name of the database to be created.

For example, to perform a point-in-time recovery for a database name `myproddb`, execute the following command:

```
db2 rollforward db myproddb to 2010-05-26-22.14.12 and complete
```

As a second alternative, roll forward may be performed to the end of the backup. In this case, the database is only rolled forward to the minimum recovery time by executing the following command:

```
db2 rollforward db [DatabaseName] to end of backup and complete
```

Where

- `DatabaseName` identifies the name of the database to be created.

For example, to perform an end-of-backup roll forward, execute the following command:

```
db2 rollforward db myproddb to end of backup and complete
```

Snapshot Restore for a Recoverable Partitioned Database

A DPF database must be restored by partitions individually. In this case, first restore the catalog partition followed by all other partitions.

1. To restore a partition, execute the following commands on each partition:

   ```
   export DB2NODE=[DBPartitionNumber]
   ```

   Where

   - `DBPartitionNumber` identifies the database partition node.

   ```
   db2 restore db [DatabaseName] use snapshot
   ```

   Where

   - `DatabaseName` identifies the name of the database to be created.

For example, to restore a two-partition database named `myproddb`, execute the following commands from the database server:

```
export DB2NODE=0

db2 terminate

DB20000I  The TERMINATE command completed successfully.

db2 restore db myproddb use snapshot

SQL2539W  Warning!  Restoring to an existing database that is the same as the
backup image database.  The database files will be deleted.
Do you want to continue ? (y/n) y
DB20000I  The RESTORE DATABASE command completed successfully.

export DB2NODE=1

db2 terminate

DB20000I  The TERMINATE command completed successfully.

db2 restore db myproddb use snapshot

SQL2539W  Warning!  Restoring to an existing database that is the same as the
backup image database.  The database files will be deleted.
```

```
Do you want to continue ? (y/n) y

DB20000I  The RESTORE DATABASE command completed successfully.
```

Following the restore of all partitions, the database must be rolled forward to the end of the logs, to a point in time, or to the end of the backup, just as in the previous roll-forward case with a single-partition database.

Prior to issuing a roll forward, make sure that all log files that are required to roll forward through are available.

2. To check which log files the backup image includes, issue a `db2acsutil query db` command:

   `db2acsutil query db [DatabaseName]`

   Where

   - `DatabaseName` identifies the name of the database to be created.

For example, to check the log file required for the restore of a database named `myproddb`, execute the following command:

```
db2acsutil query db myproddb
Instance   Database   Part Image Time     Host       First Log
========== ========== ==== ============== ========== ==========
db2inst1   MYPRODDB          0 20100526162627 db2srv1.ibm.com          9
db2inst1   MYPRODDB          1 20100526162627 db2srv2.ibm.com          8
```

3. To complete the recovery, use one of the roll-forward recovery options: `to end of logs`, `to a point in time`, or `to the end of backup`. For example, to roll forward to the end of the logs, execute the following command:

```
db2 rollforward db myproddb to end of logs and complete
```

Note: By default, similar to a conventional restore, a DB2 Snapshot restore does not include the log directories from the backup image. The database manager returns an error when the restore is issued with the `logtarget include` option because the preexisting log directory on the disk conflicts with the log directory on the backup image.

4. To restore the database in this case, execute the restore command as follows:

`db2 restore db [DatabaseName] use snapshot logtarget include force`

Where

- `DatabaseName` identifies the name of the database to be created.

For example, to restore a database named `myproddb` and replace the log directory on disk with the log directory from the Snapshot backup image, execute the following command from the database server:

```
db2 restore db myproddb use snapshot logtarget include force

SQL2539W  Warning!  Restoring to an existing database that is the same as the
backup image database.  The database files will be deleted.

Do you want to continue ? (y/n) y

DB20000I  The RESTORE DATABASE command completed successfully.
```

The restore process for recoverable and nonrecoverable databases is summarized in Table 2.

Table 2) Restore options for a recoverable and a nonrecoverable database (MYPRODDB).

Database Type	Offline	Online
Recoverable Database (Archive logging enabled)	db2 restore db myproddb use snapshot without rolling forward	db2 restore db myproddb online use snapshot db2 rollforward db myproddb to end of logs and complete
Nonrecoverable Database (Archive logging disabled)	db2 backup db myproddb use snapshot without rolling forward	N/A

7 CONCLUSION

The NetApp storage system offers DBAs a compelling advantage in terms of database backup and recovery. By taking Snapshot copies frequently and keeping an appropriate number of Snapshot copies online, the DBA has the ability to restore a database without incurring the overhead normally associated with recovery from tape. Additionally, backup and recovery performance is dramatically improved—the process of creating or restoring from a Snapshot copy can be done in a fraction of the time needed for conventional methods.

8 REFERENCES

1. "Implementation of RAID Double Parity for Data Protection"
   http://www.netapp.com/library/tr/3298.pdf
2. "Using Integrated Snapshot Backup Feature of IBM DB2 9.5 with NetApp Storage System (Backup, Restore, Clone, and Tape Backup)"
   http://media.netapp.com/documents/tr-3668.pdf
3. "File System Design for an NFS File Server Appliance"
   http://www.netapp.com/tech_library/3002.html#i34
4. "Data Protection Online Backup and Recovery Guide"
   http://now.netapp.com/NOW/knowledge/docs/ontap/rel701r1/html/ontap/onlinebk/index.htm
5. "IBM DB2 on NetApp Storage: Deployment and Best Practices"
   http://www.netapp.com/library/tr/3272.pdf
6. "Data Recovery and High Availability Guide and Reference"
   ftp://ftp.software.ibm.com/ps/products/db2/info/vr82/pdf/en_US/db2hae81.pdf
7. "Quick Beginnings for DB2 Servers"
   ftp://ftp.software.ibm.com/ps/products/db2/info/vr82/pdf/en_US/db2ise81.pdf
8. "Data ONTAP 7.1 System Administration Guide"
   http://now.netapp.com/NOW/knowledge/docs/ontap/rel71/pdfs/ontap/sysadmin.pdf

9 REVISION HISTORY

Date	Author	Comments
March 2009	Jawahar Lal and Bobby Oommen	Original draft
December 2009	Bobby Oommen	Updated Data ONTAP, DB2, Controller Type, OS Version, and Template

Date	Author	Comments
August 2010	Bobby Oommen	SnapCreator integration and DB2 ACS integration
April 2011	Bobby Oommen	Minor updates

10 APPENDICES

10.1 APPENDIX A

```
./snapcreator --config myproddb  --action snap --policy daily --verbos

########## Application Backup Start #########
[Fri Apr 16 11:58:04 2010] Beggining backup start of myproddb
[Fri Apr 16 11:58:04 2010] INFO: Executing 'db2 connect to myproddb' for
Database: myproddb
[Fri Apr 16 11:58:06 2010] INFO: Command db2 connect to myproddb Completed
Successfully
[Fri Apr 16 11:58:06 2010] INFO: Executing 'db2 set write suspend for database'
for Database: myproddb
[Fri Apr 16 11:58:06 2010] INFO: Command db2 set write suspend for database
Completed Successfully
[Fri Apr 16 11:58:06 2010] INFO: Executing 'db2 connect reset' for Database:
myproddb
[Fri Apr 16 11:58:06 2010] INFO: Command db2 connect reset Completed
Successfully
[Fri Apr 16 11:58:06 2010] Backup Start of myproddb Completed Successfully
########## Generating Info ASUP on ntap1 #########
[Fri Apr 16 11:58:06 2010] INFO: NetApp ASUP create on ntap1 Completed
Successfully
########## Gathering Information for ntap1:myproddb_data #########
[Fri Apr 16 11:58:06 2010] INFO: Performing NetApp Snapshot Inventory for
myproddb_data on ntap1
[Fri Apr 16 11:58:07 2010] INFO: NetApp Inventory of myproddb_data on ntap1
completed Successfully
########## Gathering Information for ntap1:myproddb_logs #########
[Fri Apr 16 11:58:07 2010] INFO: Performing NetApp Snapshot Inventory for
myproddb_logs on ntap1
[Fri Apr 16 11:58:07 2010] INFO: NetApp Inventory of myproddb_logs on ntap1
completed Successfully
```

```
########## Running Netapp Snapshot Rename on Primary ntap1 #########

########## Starting Consistency Group Snapshot for ntap1 #########

[Fri Apr 16 11:58:07 2010] INFO: Starting NetApp Consistency Group Snapshot
VIPER_ONLINE-daily_20100416115804 for volumes myproddb_data,myproddb_logs on
ntap1

[Fri Apr 16 11:58:07 2010] INFO: NetApp Consistenc Group Start for ntap1 volumes
myproddb_data,myproddb_logs succeeded!

########## Committing Consistency Group Snapshot for all filers #########

[Fri Apr 16 11:58:07 2010] INFO: Commiting NetApp Consistency Group Snapshot for
ntap1 succeeded!

[Fri Apr 16 11:58:07 2010] INFO: Creation of NetApp Consistency Group Snapshot
VIPER_ONLINE-daily_20100416115804 on ntap1 Completed Successfully

########## Application Backup Stop #########

[Fri Apr 16 11:58:07 2010] Beggining backup stop of myproddb

[Fri Apr 16 11:58:07 2010] INFO: Executing 'db2 connect to myproddb' for
Database: myproddb

[Fri Apr 16 11:58:07 2010] INFO: Command db2 connect to myproddb Completed
Successfully

[Fri Apr 16 11:58:07 2010] INFO: Executing 'db2 set write resume for database'
for Database: myproddb

[Fri Apr 16 11:58:08 2010] INFO: Command db2 set write resume for database
Completed Successfully

[Fri Apr 16 11:58:08 2010] INFO: Executing 'db2 connect reset' for 1 Database:
myproddb

[Fri Apr 16 11:58:08 2010] INFO: Command db2 connect reset Completed
Successfully

[Fri Apr 16 11:58:08 2010] INFO: Backup Stop of myproddb Completed Successfully

########## Generating Info ASUP on ntap1 #########

[Fri Apr 16 11:58:08 2010] INFO: NetApp ASUP create on ntap1 Completed
Successfully

########## Running Netapp Snapshot Delete on Primary ntap1 #########

########## SnapCreator Completed Successfully #########

[Fri Apr 16 11:58:08 2010] INFO: SnapCreator Backup for VIPER_ONLINE ACTION:
snap POLICY: daily completed successfuly

[Fri Apr 16 11:58:08 2010] INFO: Creating OM Event (script:information-event) on
10.61.161.95

[Fri Apr 16 11:58:08 2010] INFO: OM Event (script:information-event) on
10.61.161.95 created successfully
```

10.2 APPENDIX B

```
./snapcreator --profile myproddb --action snaplist --policy daily
```

```
########## SnapCreator Snapshot (Primary) List for ntap1:myproddb_data #########
### Snapshot Name ###                          ### Snapshot Timestamp ###
MYPRODDB-daily_20100329222639                          Mar 29 2010 22:34:02
MYPRODDB-daily_20100329222939                          Mar 29 2010 22:37:02
MYPRODDB-daily_20100415115801                          Apr 15 2010 12:05:16
MYPRODDB-daily_20100415161507                          Apr 15 2010 16:22:21
MYPRODDB-daily_20100416143317                          Apr 16 2010 09:40:33
MYPRODDB-daily_20100416143429                          Apr 16 2010 09:41:43
MYPRODDB-daily_20100416143532                          Apr 16 2010 09:42:46
MYPRODDB-daily_20100416102025                          Apr 16 2010 10:27:42
MYPRODDB-daily_20100416124250                          Apr 16 2010 12:50:06
########## SnapCreator Snapshot (Primary) List for ntap1:myproddb_logs #########
### Snapshot Name ###                          ### Snapshot Timestamp ###
MYPRODDB-daily_20100329222639                          Mar 29 2010 22:34:03
MYPRODDB-daily_20100329222939                          Mar 29 2010 22:37:03
MYPRODDB-daily_20100415115801                          Apr 15 2010 12:05:17
MYPRODDB-daily_20100415161507                          Apr 15 2010 16:22:22
MYPRODDB-daily_20100416143317                          Apr 16 2010 09:40:34
MYPRODDB-daily_20100416143429                          Apr 16 2010 09:41:45
MYPRODDB-daily_20100416143532                          Apr 16 2010 09:42:48
MYPRODDB-daily_20100416102025                          Apr 16 2010 10:27:43
MYPRODDB-daily_20100416124250                          Apr 16 2010 12:50:06
```

10.3 APPENDIX C

```
# ./snapcreator --config myproddb --action restore --policy daily

### You have chosen to do a snap restore on one or more volumes for the Config:
myproddb Policy: daily ###
Are you sure you want to continue (y|n)? y
01. Storage Admin View
02. Database Admin View
Select a View (enter a number or "q" to quit): 02
### Snapshot Menu for netap1 ###
01. MYPRODDB-daily_20100416143503 (Apr 16 2010 14:42:20)
02. MYPRODDB-daily_20100416125302 (Apr 16 2010 13:00:18)
```

03. MYPRODDB-daily_20100416125011 (Apr 16 2010 12:57:28)

04. MYPRODDB-daily_20100416124955 (Apr 16 2010 12:57:12)

05. MYPRODDB-daily_20100416124950 (Apr 16 2010 12:57:06)

06. MYPRODDB-daily_20100416124944 (Apr 16 2010 12:57:01)

07. MYPRODDB-daily_20100416124939 (Apr 16 2010 12:56:55)

Select a snapshot for restore (enter a number, "n" for next filer, or "q" to quit): 01

WARN: You have selected to do a volume restore, All data in netap1:myproddb_data will be reverted to snapshot MYPRODDB-daily_20100416143503

WARN: All data in netap1:myproddb_data from Apr 16 2010 14:42:20 to Apr 21 2010 11:41:46 will be lost!!!

Are you sure you want to continue with the restore (y|n)?y

Snapshot Volume Restore for netap1:myproddb_data using snapshot MYPRODDB-daily_20100416143503 Starting!

INFO: NetApp Snapshot Volume Restore of MYPRODDB-daily_20100416143503 on netap1:myproddb_data Completed Successfully

WARN: You have selected to do a volume restore, All data in netap1:myproddb_logs will be reverted to snapshot MYPRODDB-daily_20100416143503

WARN: All data in netap1:myproddb_logs from Apr 16 2010 14:42:20 to Apr 21 2010 11:41:49 will be lost!!!

Are you sure you want to continue with the restore (y|n)?y

Snapshot Volume Restore for netap1:myproddb_logs using snapshot MYPRODDB-daily_20100416143503 Starting!

INFO: NetApp Snapshot Volume Restore of MYPRODDB-daily_20100416143503 on netap1:myproddb_logs Completed Successfully

Snapshot Menu for netap1

01. MYPRODDB-daily_20100416143503 (Apr 16 2010 14:42:20)

02. MYPRODDB-daily_20100416125302 (Apr 16 2010 13:00:18)

03. MYPRODDB-daily_20100416125011 (Apr 16 2010 12:57:28)

04. MYPRODDB-daily_20100416124955 (Apr 16 2010 12:57:12)

05. MYPRODDB-daily_20100416124950 (Apr 16 2010 12:57:06)

06. MYPRODDB-daily_20100416124944 (Apr 16 2010 12:57:01)

07. MYPRODDB-daily_20100416124939 (Apr 16 2010 12:56:55)

Select a snapshot for restore (enter a number, "n" for next filer, or "q" to quit): q

10.4 APPENDIX D

Script to perform the backup for a DB2 database using NetApp Snapshot.

```
#!/bin/ksh
```

```
db2 connect to myproddb user db2inst1 using db2inst1
db2 set write suspend for database
rsh dbsrv1 snap create dbdata dbdata_snp01
rsh dbsrv1 snap create dblogs dblogs_snp01
db2 set write resume for database
```

Script to perform the recovery of a DB2 database using NetApp snapshot.

```
db2 force applications all
db2stop
snap restore -s dbdata_snap01 dbdata
```

NetApp provides no representations or warranties regarding the accuracy, reliability, or serviceability of any information or recommendations provided in this publication, or with respect to any results that may be obtained by the use of the information or observance of any recommendations provided herein. The information in this document is distributed AS IS, and the use of this information or the implementation of any recommendations or techniques herein is a customer's responsibility and depends on the customer's ability to evaluate and integrate them into the customer's operational environment. This document and the information contained herein may be used solely in connection with the NetApp products discussed in this document.

www.netapp.com