

Continuous availability databases with IBM DB2 Version 9.5 and EMC SRDF

July 2, 2009

Paul Pendle
Roger Sanders
EMC Corporation

Aruna De Silva
Aslam Nomani
Enzo Cialini
IBM Canada Ltd.
IBM Toronto Lab

Disclaimers and trademarks

Copyright © 2009 EMC Corporation and IBM Corporation.

Printed 2009

The authors believe the information in this publication to be accurate as of its publication date. The information is subject to change without notice.

THE INFORMATION IN THIS PUBLICATION IS PROVIDED "AS IS." - NEITHER EMC CORPORATION NOR IBM CORPORATION MAKE ANY REPRESENTATIONS OR WARRANTIES OF ANY KIND WITH RESPECT TO THE INFORMATION IN THIS PUBLICATION, AND BOTH SPECIFICALLY DISCLAIM IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

The furnishing of this document does not imply giving license to any IBM or EMC patents.

References in this document to IBM products, Programs, or Services do not imply that IBM intends to make these available in all countries in which IBM operates.

Use, copying, and distribution of any EMC or IBM software described in this publication requires an applicable software license.

Trademark Information

EMC, EMC2, and Symmetrix are registered trademarks and TimeFinder, SRDF, and where information lives are trademarks of EMC Corporation.

IBM, AIX, DB2, DB2 Universal Database, and System p, are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both.

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

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

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

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

Table of contents

Disclaimers and trademarks	ii
Abstract	1
Introduction - The technical solution and basic concepts	2
<i>Disaster recovery and continuous availability</i>	3
<i>EMC Symmetrix Remote Data Facility (SRDF) overview</i>	3
<i>Fundamentals of SRDF control operations</i>	4
<i>A note on bandwidth requirements for implementing SRDF</i>	5
<i>Benefits of using DB2 9.5 with SRDF as a DR solution</i>	5
<i>DB2 for LUW command description</i>	6
<i>Command conventions used in this document</i>	6
System configuration	7
<i>Hardware and volume configuration</i>	7
<i>Scenario configuration setup tasks</i>	9
<i>Configuring the volumes on the host systems</i>	10
<i>IBM DB2 9.5 storage configuration</i>	11
<i>Finishing setup on the secondary system</i>	12
<i>Reestablish the SRDF connections for standby capability</i>	15
Usage scenarios	18
<i>Scenario 1 □ Planned (orderly) failover to secondary system</i>	18
<i>Scenario 2 □ Planned failback to primary system</i>	19
<i>Scenario 3 □ Disaster failover to secondary system</i>	20
<i>Scenario 4 □ Disaster failback to primary system</i>	21
Conclusion	23
Appendix A □ List of reference for more information	24
Appendix B □ EMC Symmetrix configuration and management	25
Appendix C □ AIX and Linux file systems command reference	28
<i>Configuring the volumes on AIX systems</i>	28
<i>Configuring the volumes on Linux systems</i>	30

Abstract

Today's highly competitive business marketplace leaves little room for error in terms of availability, continuous operations, or recovery in the event of an unplanned outage. In today's connected world, an event that makes business data unavailable, even for relatively short periods of time, has the potential to be of major impact.

Hence, today's mission-critical database systems must operate 24x7 with the highest degree of availability possible. As databases increase in size, and ad hoc queries place more demands on the continued availability of the system, the time and hardware resources required to back up and recover databases grows substantially while the maintenance windows are either drastically reduced or disappear completely.

EMC[®] Symmetrix[®] Remote Data Facility (SRDF[®]) software, suspend I/O features available in operating systems, and IBM[®] DB2[®] Version 9.5 for Linux[®], UNIX[®], and Windows[®] (DB2 9.5) software combine to provide highly available advanced database technical architectures to meet these demands. In this paper, we demonstrate the use of these features to create a coherent copy of a continuously available DB2 9.5 database to meet the increasing availability requirements demanded in today's business environment.

Introduction - The technical solution and basic concepts

This paper describes a typical Disaster Recovery (DR) configuration using IBM DB2 Enterprise Server Edition Version 9.5 for Linux, UNIX, and Windows (DB2 9.5) with EMC SRDF. The setup shown here is appropriate for maintaining a disaster recovery site at a different location from the primary site, and will work especially well when the two sites are reasonably close (< 200 km apart) – within a metropolitan area, for example.

This configuration is primarily designed to recover from disasters that occur outside of the database application, such as a power failure, communication blackout, earthquake, or operating system crash. They can also be used for planned shutdowns, such as for system or hardware changes to the primary system.

The following figure illustrates our system configuration.

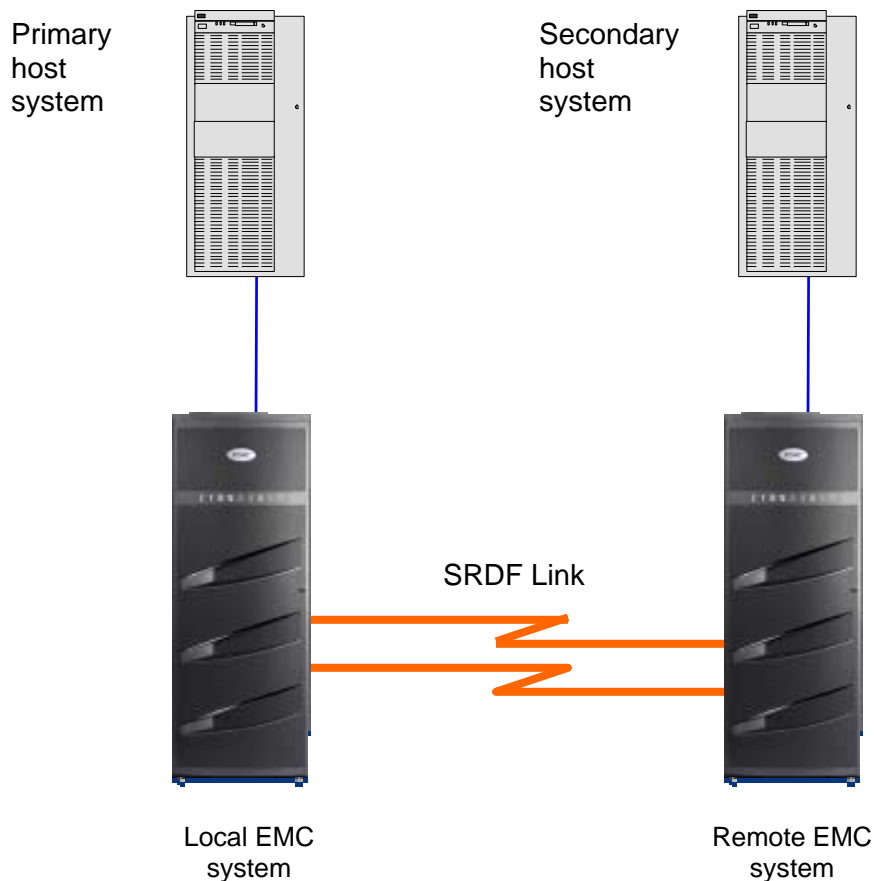


Figure 1. IBM DB2 9.5 and EMC SRDF system configuration

Disaster recovery and continuous availability

Disaster recovery is the ability to recover a data center at a different site, on different hardware, if a disaster destroys the primary site or renders it inoperable. A non-disaster problem, such as a corruption of a key client database, is not a disaster in this sense of the term, unless processing must be resumed at a different location and on different hardware.

Continuous availability is the term that is used to describe systems that run and are continuously available to customers. For this to occur:

- During peak operating periods, transactions must be processed efficiently without sacrificing performance even during a loss of availability.
- Systems must be able to recover quickly when hardware or software failures occur or when disaster strikes.
- Software that powers the enterprise databases must be continuously running and available for transaction processing.

In this paper, we will examine the use of IBM DB2 9.5 and EMC SRDF technologies to provide a DR solution.

EMC Symmetrix Remote Data Facility (SRDF) overview

The EMC SRDF is a *business continuity solution* that provides a host-independent, mirrored data storage technique for duplicating production site data to one or more physically separated remote sites (using target Symmetrix systems). In short, SRDF is a configuration of multiple Symmetrix units, set up to maintain multiple copies of logical volume data in physically separated locations.

SRDF replicates production or primary (source) site data to a secondary (target) site transparently to users, applications, databases, and host processors. The local SRDF device, known as the source (R1) device, is configured in a partner relationship with a remote target (R2) device, forming an SRDF pair. While the R2 device is mirrored with the R1 device, the R2 device is write-disabled to the remote host. After the R2 device becomes synchronized with its R1 device, the R2 device can be split from the R1 device at any time, making the R2 device fully accessible again to its host. After the split, the target (R2) device contains valid data that is a point-in-time copy of R1 data and is available for performing business continuity tasks.

SRDF requires configuration of specific source Symmetrix volumes (R1) to be mirrored to target Symmetrix volumes (R2). If the primary site is no longer able to continue processing when SRDF is operating in synchronous mode, data at the secondary site is current up to the last committed transaction. When primary systems are down, SRDF enables fast failover to the secondary copy of the data so that critical information becomes available in minutes. Business operations and related applications can resume full functionality with minimal interruption.

Figure 2 illustrates a basic SRDF configuration. As shown in the figure, connectivity between the two Symmetrix systems is provided using ESCON, Fibre Channel, or Gigabit Ethernet.

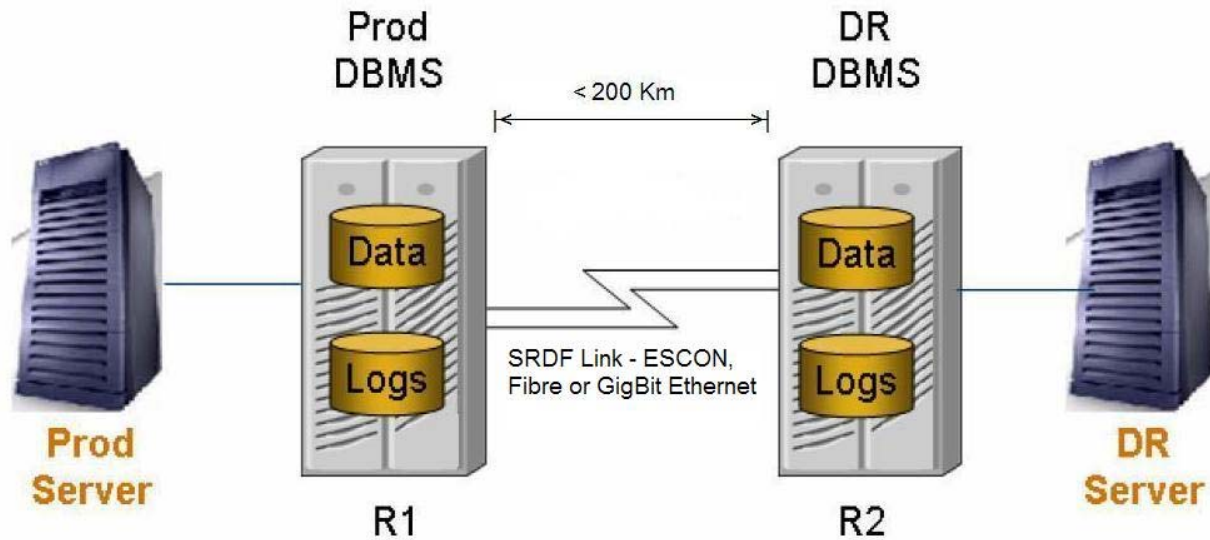


Figure 2. A typical DR configuration using EMC SRDF technology

Fundamentals of SRDF control operations

The EMC Solutions Enabler SYMCLI SRDF commands perform the following basic control operations on SRDF devices:

- **Establish** synchronizes an SRDF pair by initiating a data copy from the source (R1) side to the target (R2) side. This operation can be full or incremental. Changes on the R2 volumes are discarded by this process.
- **Restore** resynchronizes a data copy from the target (R2) side to the source (R1) side. This operation can be full or incremental. Changes on the R1 volumes are discarded by this process.
- **Split** stops mirroring for the SRDF pair(s) in a device group and write-enables the R2 devices.
- **Swap** exchanges the source (R1) and target (R2) designations on the source and target volumes.
- **Failover** switches data processing from the source (R1) side to the target (R2) side. The source side volumes (R1), if still available, are write-disabled.
- **Failback** switches data processing from the target (R2) side to the source (R1) side. The target side volumes (R2), if still available, are write-disabled.

Each of these commands performs multiple operations on the SRDF pairings and as such is called a composite command.

A note on bandwidth requirements for implementing SRDF

In planning for an SRDF configuration, you must compute your bandwidth requirements in advance. You should measure separately the amount of disk I/O required for your data storage and for log storage. You are particularly concerned about the amount of change activity because all modifications to the database must be replicated. Unless you are not concerned about per-transaction response time, be sure to determine the peak I/O requirements rather than just the total or average requirements. Also, be sure to allow for future growth.

Next you must determine the cost of the bandwidth alternatives. If the two sites are in the same building or on the same campus, the cost differential between a lower capacity connection and a higher capacity one is probably insignificant. Using local direct fiber connections will generally mean that you have more than enough bandwidth to meet your needs for full data replication.

If the sites are far apart, you must deal with a communication company to get the bandwidth, and you must use additional equipment to connect them through the communication lines. Often, you will want to use far less bandwidth than the capacity of a fiber cable, so there can be severe limitations on the data that can be sent. Incorrect provisioning of bandwidth for DR solutions can adversely affect production performance and can invalidate the overall solution.

Benefits of using DB2 9.5 with SRDF as a DR solution

DB2 9.5 using SRDF for mirror copy offers the following features and benefits:

- High data availability
- High performance
- Flexible configurations
- Host and application software transparency
- Automatic recovery from a component or link failure
- Significantly reduced recovery time after a disaster
- Increased integrity of recovery procedures
- Reduced disaster recovery complexity, planning, testing, etc.
- Support Business Continuance across and between multiple databases on multiple servers and Symmetrix systems.

DB2 for LUW command description

Suspend I/O command

The suspend command (`SET WRITE SUSPEND FOR DATABASE`) suspends all write operations to a DB2 database (i.e., to table spaces and log files). Read operations are not suspended and are thus allowed to continue. Additionally, applications can continue to process insert, update, and delete operations utilizing the DB2 buffer pools. A database connection is required for issuing the suspend command. It is recommended that you maintain the current session for executing the subsequent resume command.

Resume I/O command

The resume command (`SET WRITE RESUME FOR DATABASE`) resumes all write operations to the suspended DB2 database. A database connection is required for issuing the resume command.

Command conventions used in this document

- Commands that must be run by root will be shown with a shell prompt of `root>` .
- Commands that must be run by the database user (the instance owner or database administrator) will be shown with a shell prompt of `dba>`.
- Commands **in bold face** are shown as the actual commands issued on one (or both) of the systems.

System configuration

This paper illustrates a typical hardware and volume configuration required to set up a DR solution using disk replication with EMC SRDF technology and DB2 9.5.

In this setup, the data and logs are always kept fully synchronized during normal operation. After a disaster, the primary system is stopped if necessary, the data and log volumes are split, and then the secondary system takes over operation. The SRDF links are switched so that the primary volumes are kept synchronized with the updates that are applied by the secondary system. (If the primary Symmetrix volume cannot communicate, such as when the Symmetrix frame has failed or the communication medium is down, then the volumes will have to be resynchronized once the problem is fixed.)

After the disaster problem has been fixed (and the volumes are fully synchronized if synchronization was interrupted by the disaster) an orderly shutdown of the secondary system allows the volumes to be switched back and the primary system to resume its normal role.

NOTE:

For the sake of clarity, we have intentionally excluded OS (that is, IBM AIX®) commands to manage disk volumes from the main discussion in this paper. However, under "[Appendix C- AIX and Linux file system management commands](#)", we have included some examples of command syntax as supplementary material.

Hardware and volume configuration

Our physical configuration consisted of:

- Two EMC Symmetrix Disk Array Systems, one configured as the local SRDF controller and the other as the remote.
- Two IBM System p® servers (host systems) connected via two fiber HBAs (to provide path redundancy) to each Symmetrix system. One system was used as the primary database system and the other as the secondary.
- The two host systems running AIX 5.3-05-03, IBM DB2 Enterprise Server Edition Version 9.5 for Linux, UNIX, and Windows, and EMC Solutions Enabler SYMCLI Base Components v6.3.2.
- The connection between the two Symmetrix systems was made with two fiber links.
- The database application in both hosts made direct use of 10 Symmetrix hyper volumes.
 - The data was kept on eight volumes in a JFS2 journaled file system.
 - Logs were kept in a separate JFS2 journaled file system established on the other two volumes.

The following figure illustrates our system configuration.

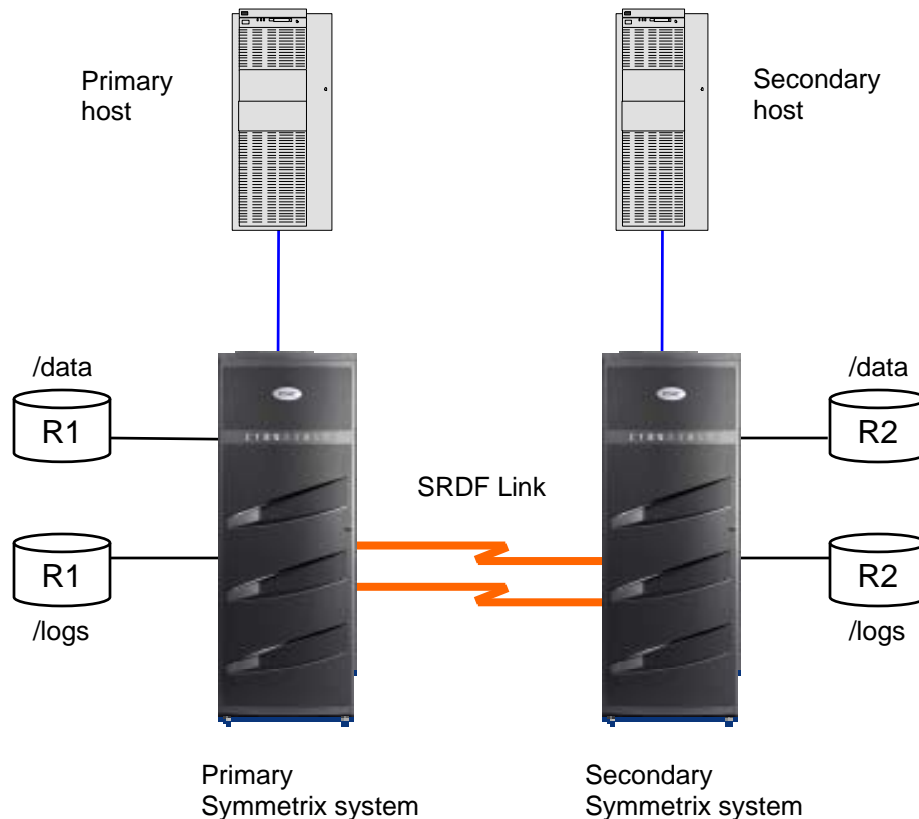


Figure 3. DB2 9.5 and EMC SRDF full data disk replication configuration

When the local (R1) system was operating normally, the data and log volumes were always kept in the synchronized (established) state with the remote (R2) system. When a disaster was simulated (in the local R1 system), both data and log volumes were split and the application was started on the R2 (remote) system. For recovery, the R1/R2 systems were logically *swapped* so that the R1 became the R2 (and vice versa). Thus, the volumes on the primary system (now the R2) were kept synchronized with the changes applied by the backup (now the R1) system; and they were ready to be switched back once the primary site was returned to normal operations.

The following table shows the Symmetrix device group name, AIX volume group name, AIX logical volume name, file system mount points, the Symmetrix device numbers used on each system, SRDF device type, and the AIX hdisk names assigned to the Symmetrix hyper volumes on both systems.

Symmetrix Group	Volume Group	Logical Volume	Mount Point	Symmetrix device ID	device type	hdisk
data	dvg	d1v	/data	0001:0008	R1(R2)	8:15
logs	lvlg	llv	/logs	0009:000A	R1(R2)	16:17
gatekeeper				0241:0244		N/A
VCMDB				0000		7

NOTES:

- It is not a requirement to have the Symmetrix device numbers or the AIX hdisk names the same on both systems; it happened that our demo systems had the same configuration of other disks on the system and the same choices made for the Symmetrix configuration.
- Note that the device type **R1(R2)** is normally an **R1** device on the primary system and an **R2** device on the secondary, but during disaster recovery stages, their personalities are reversed.

Scenario configuration setup tasks

Complete the steps described in “[Appendix B - EMC Symmetrix configuration and management](#)” before moving on to the scenario-specific configuration described below.

Set up Symmetrix consistency group for SRDF operations

This section describes how to define the members of a consistency group containing pairs of SRDF volumes. The volumes might have been associated when the Symmetrix systems were configured by your EMC customer engineer (CE). Alternatively, the relationships can be established using the Solutions Enabler command language. The volumes in each pair must be the same size, and all of the pairs in the group must consist of an R1 on one Symmetrix unit, each associated with an R2 on the other. Use the same procedure on both systems so that both will use the same group names, by simply exchanging the direction.

General procedure

Values required to run the procedure:

<i>GROUPNAME</i>	Name for the Symmetrix Consistency Group consisting of R1 Symmetrix volumes to be mirrored with R2 volumes.
<i>GROUPTYPE</i>	The type of the SYMCLI consistency group. This will be either rdf1 on the R1 machine, or rdf2 on the R2 machine.
<i>SSTART</i>	First volume (hex Symmetrix device number) on the local Symmetrix system. The volume numbers on the remote Symmetrix will have been defined when your Symmetrix was configured.
<i>SEND</i>	Last volume (hex Symmetrix device number) on local Symmetrix.

Create the Symmetrix consistency group for the volumes:

```
root> symcgm create GROUPNAME -type GROUPTYPE
```

Add a range of local devices to the group. The types of devices to be added must match the group type. That is to say, add R1 devices to an RDF1 group type and R2 devices to an RDF2 group type.

```
root> symcgm -cg GROUPNAME -RANGE SSTART:SEND addall dev
```

Continuous availability databases with IBM DB2 9.5 and EMC SRDF

Repeat this step if you want to combine nonconsecutive blocks of volumes into a single group.

Performed on the primary system:

```
root> symcg create db2cg -type rdf1
root> symcg -cg db2cg -RANGE 1:A addall dev
```

Performed on the secondary system:

```
root> symcg create db2cg -type rdf2
root> symcg -cg db2cg -RANGE 1:A addall dev
```

You must place Symmetrix devices into a single consistency group if you plan to perform operations that need to create a point-in-time copy of your data.

Establish the SRDF mirrors

To simplify configuration on the secondary system, the rest of the LVM and database setup steps on the primary system were done with SRDF volumes established. This makes the setup of the secondary system much simpler – as we will see later in the section [Import the disk configuration on the secondary system](#).

General procedure

Value required to run the procedure:

GROUPNAME	Name for the Symmetrix Consistency Group consisting of SRDF volumes to be mirrored.
------------------	---

Initial “establish” of a group of SRDF device pairs:

```
root> symrdf -cg GROUPNAME -noprompt -full establish
```

(Optional) Wait until the process completes.

```
root> symrdf -cg GROUPNAME -i 30 verify -synchronized
```

Performed on the primary system:

```
root> symrdf -cg db2cg -noprompt -full establish
```

There is no need to wait at this time for the “establish” process to complete – it can continue in the background while we proceed with the subsequent setup tasks.

Configuring the volumes on the host systems

For our setup, we configured a volume group with a file system to contain the data, and, on each system, another volume group with a file system to contain the logs. These were created on the primary system, and later imported onto the secondary system from the R2 copy.

Create the physical volume groups, logical volumes and the file systems, as shown in the table below.

Symmetrix Group	Volume Group	Logical Volume	Mount Point	hdisk
Data	dvg	dlv	/data	8:15
Logs	lvg	llv	/logs	16:17

Refer to the examples of AIX and Linux Operating Systems command syntax included under [“Appendix C- AIX and Linux file system command reference”](#) for more information about specific commands required to carry out these tasks.

Make file systems and devices available to DB2 instance

To permit the file systems and devices to be used by the database instance, set up the permissions properly. We show how to provide access to the mount point of the file system, but this is not a requirement. You can create subdirectories from the mount point and have the database created within these subdirectories. However, any other files or directories placed under the mount point will be copied to the SRDF volumes along with the database.

Performed on the primary system:

```
root> chown dba /data /logs
```

IBM DB2 9.5 storage configuration

IBM DB2 9.5 needs storage for creating database configuration and control information. It also requires storage for table spaces, which will in turn hold tables and other database objects. DB2 9.5 provides three types of storage for databases: SMS (*System Managed Storage*), DMS (*Database Managed Storage*) and *Automatic Storage*. For more information on DB2 Database for Linux, UNIX, and Windows storage, refer to the appropriate version of the IBM DB2 Information Center or product manuals. Our configuration demonstrates the use of automatic storage, which is the default type for DB2 9 and later versions.

We created the DB2 Database with Automatic Storage in the /data file system (type JFS2 on AIX) using default containers for the Catalog, User, and Temporary table spaces (by executing the create db command).

General procedure

Values required to run the procedure:

DBNAME	Name of the database to be created.
PATH	Location of the database.

Start the DB2 instance (if it is not already running):

```
dba> db2start
```

Create the database on the file system:

```
dba> db2 create db DBNAME on PATH
```

Performed on the primary system:

```
dba> db2start
```

```
dba> db2 create db testdb on /data
```

Set the location of logs

The default location for the DB2 log files is in the `SQLLOGDIR` directory, which is a relative path under the database path used when creating the database. You would omit setting the location of logs if you were not using a separate Symmetrix device group for the logs and the log path was contained under the database path.

General procedure

Values required to run the procedure:

<i>DBNAME</i>	Name of the database.
<i>LPATH</i>	Location for the logs.

Modify the default location of the DB2 log files:

```
dba> db2 update db cfg for DBNAME using NEWLOGPATH LPATH
```

Performed on the primary system:

```
dba> db2 update db cfg for testdb using NEWLOGPATH /logs
```

Since `NEWLOGPATH` is not an online configurable database parameter, we must deactivate/activate the “testdb” database to enable this change.

```
dba> db2 deactivate db testdb; db2 activate db testdb
```

Finishing setup on the secondary system

The configuration of devices, file systems, and the database must now be made known to the OS on the secondary system. This is largely a matter of telling that system to use the configuration already copied onto the R2 volumes. The R2 volumes are made available to the secondary system by splitting them. We do not have to stop the database on the primary system to do this. We simply use the **db2 suspended I/O** combined with I/O suspend features of the operating systems (such as AIX) to ensure that we have a consistent split file system copy.

Split the R2 volumes from the R1 volumes

General procedure

Value required to run the procedure:

GROUPNAME	Name of a composite group to split
------------------	------------------------------------

Make sure that the **establish** operation started earlier has completed successfully:

```
root> symrdf -cg GROUPNAME -synchronized verify
```

Suspend writes to the database and also to the file systems used by the database (i.e. /data and /logs):

General procedure

Value required to run the procedure:

DBNAME	Name of the database for which I/O writes will be suspended
FSNAME	Name of the file system which I/O will be suspended.
TIMEOUT	Time period (in seconds) that AIX will suspend the I/O on file system FSNAME. Acceptable values for timeout: <ul style="list-style-type: none">• 0/off - the file system will be thawed, and modifications can proceed.• A positive number - the file system is frozen for a maximum of timeout seconds specified by this number.

```
dba> db2 connect to DBNAME
dba> db2 set write suspend for database
root> chfs -a freeze=TIMEOUT FSNAME
```

(Repeat additional freezes for each file system used by the database – i.e., Log path, etc.)

The act of freezing a file system produces a nearly consistent on-disk image of the file system, and writes all dirty file system metadata and user data to the disk. In its frozen state, the file system is *read-only*, and anything that attempts to modify the file system or its contents must wait for the freeze to end.

Split the SRDF volume groups:

```
root> symrdf -cg GROUPNAME -noprompt split
```

Performed on the primary system:

```
root> symrdf -cg db2cg verify -synchronized
dba> db2 connect to testdb
dba> db2 set write suspend for database
root> chfs -a freeze=3600 /logs
root> chfs -a freeze=3600 /data
root> symrdf -cg db2cg enable -noprompt
root> symrdf -cg db2cg split -noprompt
```

```
root> chfs -a freeze=no /data
root> chfs -a freeze=no /logs

dba> db2 set write resume for database
```

Import the disk configuration on the secondary system

The volume manager setup, file system setup, and database setup have already been done on the R1 volumes on the primary system, and mirrored to the R2 volumes; the secondary system merely has to learn about them. Using the AIX volume manager, this step is simple.

General procedure

Values required to run the procedure:

<i>VGNAME</i>	Name of the volume group being imported.
<i>DISK</i>	Device name of the first disk in the group.

Import volume group definition from the physical volumes:

```
root> importvg -yVGNAME DISK
```

The contents of the file systems will have the permissions already set from the mirroring of STD devices on the primary system.

Performed on the secondary system:

```
root> importvg -y dvg hdisk8
root> importvg -y lvg hdisk16
```

Mount the file systems available in the above two volume groups

```
root> mount /data
root> mount /logs
```

NOTE:

- AIX `importvg` command will automatically bring the newly imported volume group online. Use `lspv` to verify whether the volume group is *active* (online). If the VG was not automatically activated (for some reason) you can manually bring it online by issuing `varyonvg` with VG name as input.
- Verify that the logical volumes and the file systems are available after the import of the volume group definition by issuing `lsvg -l` with VG name as input. This command should display the logical volumes and their respective mount points – e.g., `/dev/d1v` on mount point `/data`.
- In case the mount command(s) fail, run the file system check (`fsck`) utility and re-try mounting – e.g., issue `fsck -y /data` to run file system check on the `/data` file system.

Catalog the database on the secondary system

Prerequisites:

- Identical copy of IBM DB2 9.5 Enterprise Server must be installed with same fix-pack levels applied.
- The DB2 instance under which the SRDF R2 Copy of the Database will be cataloged must be identical (i.e., same db2instance owner ID, with same permissions, etc.)

The database is already present on the imported disk volumes, but the IBM DB2 9.5 instance on the secondary system does not know about the imported volumes yet. If your database application requires any auxiliary data or files that are stored outside of the database (such as control scripts), they should be set up at this time if this has not been done yet.

General procedure

Values required to run the procedure:

<i>DBNAME</i>	Name of the database.
<i>PATH</i>	Location of the database.

Start the DB2 instance (if it is not already running):

```
db2> db2start
```

Make the database known to the DB2 instance:

```
db2> db2 catalog database DBNAME on PATH
```

Install database control scripts, auxiliary files, and so on, as required to use the database properly on the secondary system.

Performed on the secondary system:

```
dba> db2 catalog database testdb on /data
```

Reestablish the SRDF connections for standby capability

Now that the database configuration has been imported into the secondary system, it is ready to be used for its purpose: standing by in case of a disaster on the primary system. Hence, for a successful disaster recovery, the secondary system must have fully synchronized copies of the data and logs from the primary system. In this scenario, that requirement is met by keeping all of the volumes synchronized and not mounted on the secondary system during normal processing. (If the secondary system left the volumes mounted, it would have an inconsistent picture of the contents, which could cause the system to crash when the volumes became available again with their contents changed with new data from the primary system.)

Unmounting the database volumes from the secondary system

Before we can reestablish the volumes, we have to prepare the secondary system for the volumes to be unavailable for the duration of normal processing. They will not be available until a disaster or a planned failover occurs, and that might be a long time in the future, in which case

the volumes could have a significant number of changes made without the knowledge of the secondary system.

General procedure

Value required to run the procedure:

FILESYS	Name of a file system to unmount.
----------------	-----------------------------------

Find any running DB2 applications

```
dba> db2 list applications
```

Stop them. This might require application-specific termination procedures.
To stop the database if all applications have already been stopped:

```
dba> db2stop
```

To stop the database, forcibly terminating all connected applications (use this only if no application-specific termination procedure is required):

```
dba> db2stop force
```

Unmount the file systems:

```
root> umount FILESYS
```

Repeat this step for each file system.

Performed on the secondary system:

```
dba> db2stop force
```

```
root> umount /data
```

```
root> umount /logs
```

Reestablish SRDF connections

General procedure

Value required to run the procedure:

GROUPNAME	Name of a composite group that will be established.
------------------	---

Establish each group of devices:

```
root> symrdf -cg GROUPNAME -noprompt establish
```

Performed on the secondary system:

This could be done on the primary instead; we do it on the secondary because the previous steps had to be done there.

```
root> symrdf -cg db2cg -noprompt establish
```

Usage scenarios

The following scenarios illustrate various uses of the documented functionality utilizing the secondary system as a hot standby system (that is, as a system available to take over) to provide continuous availability for both operational requirements (such as maintenance) and in the case of a disaster.

Scenario 1 ▪ Planned (orderly) failover to secondary system

Sometimes you will have adequate warning and reason that the primary system needs to be made unavailable – for a maintenance operation such as a fix pack upgrade on the primary (production) system, for example. With such warning, you can provide an orderly failover.

Unmount the database volumes from the primary system

Performed on the primary system:

You might precede this with a procedure that blocks new transactions from starting for a while.

```
dba> db2stop force
root> umount /data
root> umount /logs
```

Make the database volumes available to the secondary system

The database volumes are no longer being changed by the primary system, and all indications of their being busy (database state, file system journal) have been stopped. So, the secondary system can simply mount the volumes and start the database. The SRDF commands **failover** and **swap** are used to make the volumes unavailable to the primary host, and to exchange the R1/R2 roles of the two hosts. After this, the secondary system acts as the R1 and the primary system acts as the R2 in all commands.

Performed on the secondary system:

```
root> symrdf -cg db2cg -noprompt failover
root> symrdf -cg db2cg -noprompt swap

root> mount /data
root> mount /logs

dba> db2start
```

Now, you must start your application database and ensure that processes using it will be directed to the secondary system instead of to the primary. How you do that depends on your circumstances; perhaps by using the DB2 Automatic Client Reroute feature.

Preparing to fail back

It is important to keep the volumes on the primary Symmetrix unit up to date, if possible, or get them back up to date as soon as possible. Depending on the reason for the planned failover, the primary Symmetrix system might be operational throughout the failover; or it might be unavailable for a while but become available later. As soon as it is available, its volumes should be brought up to date with the ongoing activity on the secondary system.

Performed on either system:

```
root> symrdf -cg db2cg -noprompt establish
```

Because of the **swap**, this **establish** command causes data to be copied from the secondary system (which is now the R1) to the primary (which is now the R2).

Scenario 2 ▪ Planned failback to primary system

After the primary system is ready to resume running the application (which requires that the SRDF volumes are synchronized between the two systems), the procedure to fail back is essentially the same as the planned failover procedure – that is, you just exchange which system runs the commands. (Even though the original secondary system is currently running the application, we still refer to it as the secondary system.) The sole difference is that the **failover** command is replaced by the **failback** command, to indicate that the original R2 system is currently acting as the R1.

Performed on the secondary system:

You might precede this with a procedure that blocks new transactions from starting for a while.

Verify that R1/R2 SRDF pairs are synchronized:

```
root> symrdf -cg db2cg -i 10 verify
```

```
dba> db2stop force
```

```
root> umount /data
```

```
root> umount /logs
```

Performed on the primary system:

```
root> symrdf -cg db2cg -noprompt failback
```

```
root> symrdf -cg db2cg -noprompt swap
```

Performed on the primary system:

```
root> mount /data
```

```
root> mount /logs
```

```
dba> db2start
```

Preparing to resume standby capability

We now ensure that the volumes on the secondary Symmetrix unit will be kept up to date so that it can resume the role of standby in case of a disaster on the primary system.

Performed on either system:

```
root> symrdf -cg db2cg -noprompt establish
```

Because of the second **swap**, these **establish** commands cause data to be mirrored again from the primary system (which is again the R1) to the secondary (which is again the R2).

Scenario 3 ▪ Disaster failover to secondary system

Sometimes you will have no warning before the primary system becomes unavailable. This can happen because of a physical disaster (such as an earthquake or a power failure) or a system failure (such as an OS crash or the loss of network connections between the primary system and the database clients). When there is no warning, you can still fail over to use the secondary system to run the application, but you have lost any opportunity for an orderly shutdown of the primary system. This means that there might be more visible effects such as aborted transactions.

One special case should be mentioned here. If the primary system and the SRDF lines are still operational, then the [planned failover procedure](#) above should be used. This might happen if the primary system lost its network connection and was unable to communicate with its clients but still had an SRDF connection to the secondary Symmetrix system. That would be a disaster to the application, but would still permit an unplanned opportunity to use the planned shutdown procedure.

Make the database volumes available to the secondary system

The database volumes are no longer being changed by the primary, but indications of their being busy (database state, file system journal) might still be present. So, when the secondary system mounts the file systems, there might be a slight delay as the journal is replayed. More significantly, the database will indicate that it is active and so database restart (that is, crash recovery) must be performed before it can be used. The SRDF commands **failover** and **swap** are used to make the volumes unavailable to the primary host, and to exchange the R1/R2 roles of the two hosts. After this, the secondary system acts as the R1 and the primary system acts as the R2 in subsequent commands.

Performed on the secondary system:

Verify that R1/R2 SRDF pairs are synchronized:

```
root> symrdf -cg db2cg -i 10 verify

root> symrdf -cg db2cg -noprompt failover
root> symrdf -cg db2cg -noprompt swap

root> mount /data
root> mount /logs

dba> db2start
dba> db2 restart db testdb
```

Preparing to fail back

It is important to keep the volumes on the primary Symmetrix system up to date, if possible, or to get them back up to date as soon as possible. Depending on the nature of the disaster, the primary Symmetrix system might be operational throughout the failover; or it might be unavailable for a while but become available later. As soon as it is available, its volumes should be brought up to date with the ongoing activity on the secondary system.

Performed on either system:

```
root> symrdf -cg db2cg -noprompt establish
```

Because of the **swap**, this **establish** command causes data to be copied from the secondary system (which is now the R1) to the primary (which is now the R2).

Scenario 4 ▪ Disaster failback to primary system

After the primary system is ready to resume running the application, the normal procedure to fail back is the same process described in Scenario 2. However, if a disaster were to occur at the secondary site after the primary system was ready to take over but before it actually had taken over, a reversal of the disaster takeover procedure could be used.

This is essentially the same as the disaster failover procedure – you just exchange which system runs the commands. (Even if the original secondary system is currently running the application, we still refer to it as the secondary system.) The sole difference is that the **failover** command is replaced by the **failback** command, to indicate that the original R2 system is currently acting as the R1.

Performed on the secondary system (if still operating):

```
dba> db2stop force
```

```
root> umount /data
```

```
root> umount /logs
```

Performed on the primary system:

```
root> symrdf -cg db2cg -noprompt failback
```

```
root> symrdf -cg db2cg -noprompt swap
```

```
root> mount /data
```

```
root> mount /logs
```

```
dba> db2start
```

Now, you must start your application database and ensure that processes using it will be again directed to the primary system instead of the secondary. How you do that depends on your circumstances; perhaps by using the DB2 Automatic Client Reroute feature.

Preparing to resume standby capability

It is important to keep the volumes on the secondary Symmetrix system up to date, if possible, or to get them back up to date as soon as possible. Depending on the nature of the disaster, the secondary Symmetrix system might be operational and able to communicate with the primary Symmetrix system throughout the disaster; or it might be unavailable for a while but become available later. As soon as it is available, its volumes should be brought up to date with the ongoing activity on the primary system. Then it can resume its role of standing by in case of a disaster on the primary system.

Performed on either system:

```
root> symrdf -cg db2cg -noprompt establish
```

Because of the second **swap**, this **establish** command causes data to be copied again from the primary system (which is now the R1) to the secondary (which is now the R2).

Conclusion

Today's highly competitive marketplace leaves little room for error in terms of availability, continuous operations, or recovery in the event of an unplanned outage. Hence, today's mission-critical database systems must operate 24x7 with the highest degree of availability possible.

In this paper, we've addressed those needs by showing how to configure a highly available advanced database technical solution, demonstrating the use of IBM DB2 9.5 for Linux, UNIX, and Windows, EMC Symmetrix Remote Data Facility, and OS suspend I/O features.

Appendix A □ List of reference for more information

1. **Creating No Dataloss Standby Databases with IBM® DB2 Universal Database™ V7.2, EMC TimeFinder™, and EMC SRDF™** by John Macdonald (EMC Corporation) and Enzo Cialini (IBM Canada Ltd./IBM Toronto Software Lab), Aug. 17, 2001
 - This is the original research the present paper is based upon.

2. **EMC Solutions Guide: DB2 for LUW on EMC Symmetrix Storage Systems**, by Paul Pendle, EMC Corporation, 2007.
 - If you want to learn how to use, configure, and manage EMC Symmetrix Storage Systems as the storage solution for an IBM DB2 Database for Linux, UNIX, and Windows, this is the best reference.

3. EMC product/technical documentation:
 - **Symmetrix CLI Quick Reference Guide**
 - **Symmetrix Array Controls CLI v6.3**
 - Many of these documents can be found on the *EMC Powerlink site* at:
<http://Powerlink.EMC.com>

4. **The Data Recovery and High Availability Guide and Reference (IBM DB2 Version 9 for Linux, UNIX, and Windows)**, 2006.
 - This is a good reference to learn all the strategies for setup, configuring, and monitoring of high-availability DB2 database solutions.

5. **IBM DB2 9 and DB2 9.5 for Linux, UNIX, and Windows Information Centers on the Web**
 - <http://publib.boulder.ibm.com/infocenter/db2luw/v9/index.jsp>
 - The place for all the latest information about IBM DB2 9 for this platform.
 - <http://publib.boulder.ibm.com/infocenter/db2luw/v9r5/index.jsp>
 - The place for all the latest information about IBM DB2 9.5 for this platform.

Appendix B □ EMC Symmetrix configuration and management

Planning and configuration are important steps in setting up the storage for a large database implementation on an EMC Symmetrix using SRDF. The EMC customer engineer (CE) must first physically connect the Fibre Channel host bus adapters (HBAs) of the host systems to the Symmetrix system and configure the Symmetrix masking to enable the host HBAs to communicate with the storage fabric. Before you can use Symmetrix volumes, the physical drives in the Symmetrix unit must be logically subdivided into hyper volumes, which appear to your system as physical disks. Then the Symmetrix hyper volumes are configured as R1 or R2 (or other) Symmetrix device types. Once this is complete, the devices must be made known to your operating system (OS). This step is OS-specific; in our case the OS is AIX 5.3.5. The same procedures could be done on other operating systems, such as Linux, HP/UX, or Solaris, but the commands to manage the devices would need to be changed into the appropriate OS-specific equivalents.

SYMCLI commands must be installed and available to the root user doing the initial device configuration and subsequent operational steps. Finally, the SYMCLI database and the VCMDDB must be initialized to include the new devices so that later SYMCLI commands can operate efficiently on those devices.

AIX and Symmetrix volume discovery

Added to .profile configuration of “root” on both systems:

The path to the SYMCLI software was added to the root’s default profile by appending the following command to the PATH statement:

```
export PATH=$PATH:/usr/symcli/bin:/usr/lpp/Symmetrix/bin
```

```
Linux: export PATH=$PATH:/usr/symcli/bin
```

On Both Symmetrix systems:

The EMC CE will ensure that for each host HBA connected Dir/Port and Link is “*Active/Active*” on the Symmetrix Service Processor Menu.

The following process is typically required to make the Symmetrix volumes visible to the host system. Each step is illustrated with an example command and the output result (if applicable).

Performed on both host systems:

1. Initialize communication with the Symmetrix system, and discover device paths and other device information:

symcfg:

Discovers or displays Symmetrix configuration information by rebuilding the set of devices known to the local host. Refreshes or removes Symmetrix information from the host’s Symmetrix database file.

```
root> symcfg discover
```

2. Ensure that the *Symmetrix Volume Configuration Management Database (VCMDB)* device is visible (see example below):

sympd:

Displays information about one or all Symmetrix devices that are visible to the host.

```
root> sympd list -vcm
```

Symmetrix ID: 000187990183							
Device Name	Directors			Device			
Physical	Sym	SA	:P DA	:IT Config	Attribute	Sts	Cap (MB)
/dev/rhdisk7	0000	15C:0	16B:C8	2-Way Mir	N/Grp'd	VCM WD	45

The VCMDB stores access configurations that are used to grant host access to logical devices in a Symmetrix storage array and resides on a special system resource logical device, referred to as the VCMDB device, on each Symmetrix storage array.

3. Initialize Symmetrix front-end masking database information:

symmaskdb:

Allows the administrator to back up, restore, initialize, and show the contents of the device masking VCMDB. Also provides limited conversion and attribute options.

The "symmaskdb" requires the Symmetrix ID (SID), hence run "symcfg" first to get the SID:

```
root> symcfg list
```

S Y M M E T R I X						
SymmID	Attachment	Model	Mcode Version	Cache Size (MB)	Num Phys Devices	Num Symm Devices
000187990183	Local	DMX800	5671	32768	1	707

Only the last three digits of the SID (which is unique) are required:

```
root> symmaskdb -sid 183 init -file /tmp/emc.mask
```

NOTE: If the Symmetrix device masking database has not been initialized before, then the "init" command requires a filename to store the backup information.

4. List the available Symmetrix devices so that you can choose which Symmetrix volumes are available to be presented as LUNs to this host:

symdev:

Displays information for the selected, or all, Symmetrix devices.

```
root> symdev list
```

5. Configure Symmetrix device masking:

symmask:

Allows the administrator to set up and modify Symmetrix device masking functionality.

```
root> symmask list hba
```

Identifier	Type	Adapter	Physical Device Path	Dir:P
10000000c9231556	Fibre	10-70	/dev/rhdisk7	01C:0

The identifier in this case is the World Wide Name, or WWN of the FC-HBA, which is a 64-bit address used to uniquely identify each element in a Fibre Channel network.

Now map (add) the chosen Symmetrix devices discovered in **Step 4** to the above HBA using the Director/FA Port to which the host is connected by the Fibre Channel link(s):

```
root> symmask add dev 1:A -wwn 10000000C9231556 -dir 1C -p 0
root> symmask refresh
```

The SYMCLI device masking provides the ability to assign and mask access privileges of host bus adapters (HBAs) to Symmetrix directors and devices by associating one or more devices with an HBA-to-FA connection (known as a masked channel) that you define in the Symmetrix-based device masking database, VCMDB.

6. On the host (running AIX) detect the new Symmetrix devices presented in Step 5:

```
root> /usr/lpp/emc/Symmetrix/bin/emc_cfgmgr
```

Linux: Initiating a SCSI bus scan on Linux is tricky and depends on the kernel version and other distribution specifics. A common technique used to re-scan the SCSI bus and present the new LUNs to the Linux host is to initialize the `/proc` pseudo file system by executing the following command:

```
echo "scsi scan-new-devices" > /proc/scsi/scsi
```

7. Run SYMCLI discovery to refresh the host's Symmetrix database file with new masking information for the devices discovered in Steps 5-6:

```
root> symcfg discover
```

8. List the available devices so that you can determine how AIX device names are mapped to the Symmetrix volumes available to this host:

```
root> sympd list
```

Symmetrix ID: 000187990183

Device Name	Directors			Device				
Physical	Sym	SA	:P DA	:IT	Config	Attribute	Sts	Cap (MB)
/dev/rhdisk7	0000	01C:0	16B:C8		2-Way Mir	N/Grp'd	VCM WD	45
/dev/rhdisk8	0001	01C:0	02B:CE		2-Way Mir	N/Grp'd	RW	4315
/dev/rhdisk9	0002	01C:0	01A:CE		2-Way Mir	N/Grp'd	RW	4315
/dev/rhdisk10	0003	01C:0	16A:C1		2-Way Mir	N/Grp'd	RW	4315
/dev/rhdisk11	0004	01C:0	15B:C1		2-Way Mir	N/Grp'd	RW	4315
/dev/rhdisk12	0005	01C:0	16B:C0		2-Way Mir	N/Grp'd	RW	4315
/dev/rhdisk13	0006	01C:0	15A:C0		2-Way Mir	N/Grp'd	RW	4315
/dev/rhdisk14	0007	01C:0	16A:C3		2-Way Mir	N/Grp'd	RW	4315
/dev/rhdisk15	0008	01C:0	15B:C3		2-Way Mir	N/Grp'd	RW	4315
/dev/rhdisk16	0009	01C:0	16B:C2		2-Way Mir	N/Grp'd	RW	4315
/dev/rhdisk17	000A	01C:0	15A:C2		2-Way Mir	N/Grp'd	RW	4315

Appendix C □ AIX and Linux file systems command reference

In this Appendix we describe useful file systems-related commands for both AIX and Linux platforms. Other UNIX variants might require different commands to perform similar operations.

Configuring the volumes on AIX systems

The AIX operating system provides a Logical Volume Manager (LVM) to manage large groups of devices as volume groups. Volume groups are further subdivided into logical volumes to be used for file systems.

Configure AIX device variables

When a number of commands must be given to a large list of physical devices (hdisk), it is typically more efficient to define a shell variable to hold that list of physical device names.

Example: Assigning a list of physical devices (hdisk8 – hdisk19) to a shell variable

```
root> datadk='hdisk8 hdisk9 hdisk10 hdisk11 hdisk12 hdisk13 hdisk14'  
root> datadk="$datadk hdisk15 hdisk16 hdisk17 hdisk18 hdisk19"
```

Create volume groups

General procedure

Values required to run the procedure:

<i>VGNAME</i>	Name of the volume group.
<i>PPSIZE</i>	Size of partitions to be used.
<i>DKLIST</i>	List of physical volumes to be included in the group.

The partition size you choose depends on the size of the physical volume (hyper volumes in our setup with EMC Disks). The partition is the unit of allocation for space on the volume, so a large size will waste more space if a request has to be rounded up to a multiple of the partition size.

```
root> mkvg -f -y VGNAME -s PPSIZE DKLIST -B
```

Example: Creating a physical volume group with PPSIZE 64 MB with a pre-defined list of physical devices

```
root> mkvg -f -y dvg -s 64 $datadk -B
```

NOTE: You can also use `smitty vg` to interactively perform the above task.

Create logical volumes to contain JFS2 journaled file systems

General Procedure

Values required to run procedure:

LVNAME	Name for the logical volume.
MAXPP	Maximum number of partitions to be used.
VGNAME	Name of the volume group to contain the logical volume.
DKLIST	List of physical volumes to be included in the logical volume.

Specify the number of partitions to be available to the logical volume (use `lsvg` to find the total number of free PPs in the PV). We set the space initially to 1, so that when a journaled file system is built the logical volume to be used for the journal will be placed in the middle of the available space. This is not especially critical for volumes that are on an external storage like a Symmetrix system, but it does no harm and can have some beneficial effect.

```
root> mklv -y LVNAME -t jfs2 -ac -ex -x MAXPP VGNAME 1 DKLIST
```

Example: Create a logical volume in a physical volume group with a specified list of physical devices

```
root> lsvg dvg (if using the total #of free PPS, MAXPP = TOTAL_PPS -1,
at least one PP needs to reserved to the journal/log device)
```

```
root> mklv -y dlv -t jfs2 -ac -ex -x 250 dvg 1 $datadk
(Use the value from above command to calculate the MAXPP)
```

NOTE: You can also use `smitty lv` to interactively perform the above task.

Create JFS2 journaled file systems

General procedure

Values required to run the procedure:

LVNAME	Name of the logical volume.
MNTPT	Mount point where the file system will be used.
AUTO	Specifies whether the file system is to be mounted on reboot, either yes or no ; usually yes for the primary system and no for the secondary system.
BLKSIZE	Size for file system data/inode blocks. (Values 512, 1024, 2048, and 4096. The default value is 4096.)
FSSIZE	Final size for the file system (in 512 byte blocks, MB or in GB).

Now we create a JFS2 file system on an existing logical volume (created in the previous step). Each file system needs a mount point where it will appear in the computer's directory structure.

```
root> crfs -v jfs2 -d LVNAME -m MNTPT -A AUTO -p rw -t no -a gblksize=ISIZE
root> chfs -a size=FSSIZE MNTPT
root> mount MNTPT
```

Example: Creating a JFS2 file system (/data) of size ~16 GB, with default block size and auto-mount.

```
root> crfs -v jfs2 -d lv -m /data -A yes -p rw -t no -a gblksize=4096
root> chfs -a size=16G /data
root> mount /data
```

NOTE: You can also use `smitty jfs2` to interactively perform the above task.

Configuring the volumes on Linux systems

Similar to AIX, Linux operating system kernel 2.4 and later versions provide a Logical Volume Manager (LVM2) to manage large groups of devices as volume groups. Volume groups are further subdivided into logical volumes to be used for file systems.

Configure Linux SCSI disks for LVM

In Linux all the discovered SCSI disks including the EMC hyper volumes are defined as physical volumes named like `/dev/sd[a-z]`. After deciding on which physical volumes to add to an LVM, using `fdisk` each of these volumes needs to be properly initialized, so they can be recognized by the LVM system .

Example: Configuring a physical volume /dev/sde to be LVM compatible

```
root> sympd list -> Identify the disks to be added on the host system
root> fdisk /dev/sde
  n      -> Add Primary Partition 1 (using the full-disk capacity)
  t      -> change the Type of the new partition to "8e" (i.e. Linux LVM)
  p      -> Verify the Partition table info
  w      -> Save Partition table to disk
```

Create volume groups

We must create physical volumes from the LVM-compatible physical partitions (created in the previous step) before we can add them to the volume group. Next the `/etc/lvmtab` and `/etc/lvmtab.d` files have to be initialized for the LVM to discover the new physical volumes we just defined.

General procedure

Values required to run the procedure:

<i>VGNAME</i>	Name for the volume group.
<i>ESIZE</i>	Size of an extent in an LVM.
<i>DKLIST</i>	List of physical volumes to be included in the group.

The extent size you choose depends on the size of the physical volume (hyper volumes in our setup with EMC Disks). The **size of an extent defaults to 4 MB**, which is fine for most uses, including a setup similar to our system configuration. The extent affects the minimum size of changes that can be made to a logical volume in the volume group, and the maximum size of logical and physical volumes in the volume group. A logical volume can contain at most 65534 extents, so the default extent size (4 MB) limits the volume to about 256 GB.

```
root> pvcreate DKLIST
root> pvscan          -> Initialize LVM to scan and identify new PVs
root> vgscan
root> vgcreate [-sESIZE] VGNAME DKLIST
root> vgdisplay      -> Display the new VG information
```

Example: Creating a physical volume group with default extent size

```
root> pvcreate /dev/sde1 /dev/sdf1 /dev/sdg1
root> pvscan
root> vgscan
root> vgcreate emcvg /dev/sde1 /dev/sdf1 /dev/sdg1
```

Create logical volumes to contain Linux ext3 file systems

General procedure

Values required to run the procedure:

LVNAME	Name for the logical volume.
LVSIZE	The size of the logical volume to be created. Can be specified either in number of extents (-l flag) or in KB, MB, GB, TB (-L flag)
VGNAME	Name of the volume group to contain the logical volume.

Specify the number of partitions to be available to the logical volume.

```
root> lvcreate -l LVSIZE -n LVNAME VGNAME    -> LVZISE in number of extents
root> lvcreate -L LVSIZE -n LVNAME VGNAME    -> LVSIZE in KB,MB,GB or TB
root> lvdisplay          -> Display the new LV information
```

Example: Create a logical volume in a physical volume group

```
root> lvcreate -L 12G -n data1v emcvg
```

Create Linux ext3 file systems (with journaling)

We create a Linux ext3 file system on an existing logical volume (like the one created in the previous step). And we mount this file system and update the `/etc/fstab` to auto-mount at host system reboot.

General procedure

Values required to run the procedure:

LVNAME	Name of the logical volume.
VGNAME	Name of the volume group to contain the logical volume.
MNTPT	Mount point where the file system will be used.

```
root> mke2fs -j /dev/VGNAME/LVNAME
root> mkdir MNTPT          -> Create the new MNTPT (if not existing)
root> mount /dev/VGNAME/LVNAME MNTPT
root> vi /etc/fstab        -> Add an entry for the new MNTPT
/dev/VGNAME/LVNAME      MNTPT      ext3      defaults      1 2
```

Example: Creating an ext3 file system of size ~12 GB, and configure for auto-mount

```
root> mke2fs -j /dev/emcvg/datalv
root> mkdir /data
root> mount /dev/emcvg/datalv /data
root> vi /etc/fstab        -> Add an entry for the new mount-point /data
/dev/emcvg/datalv      /data      ext3      defaults      1 2
```