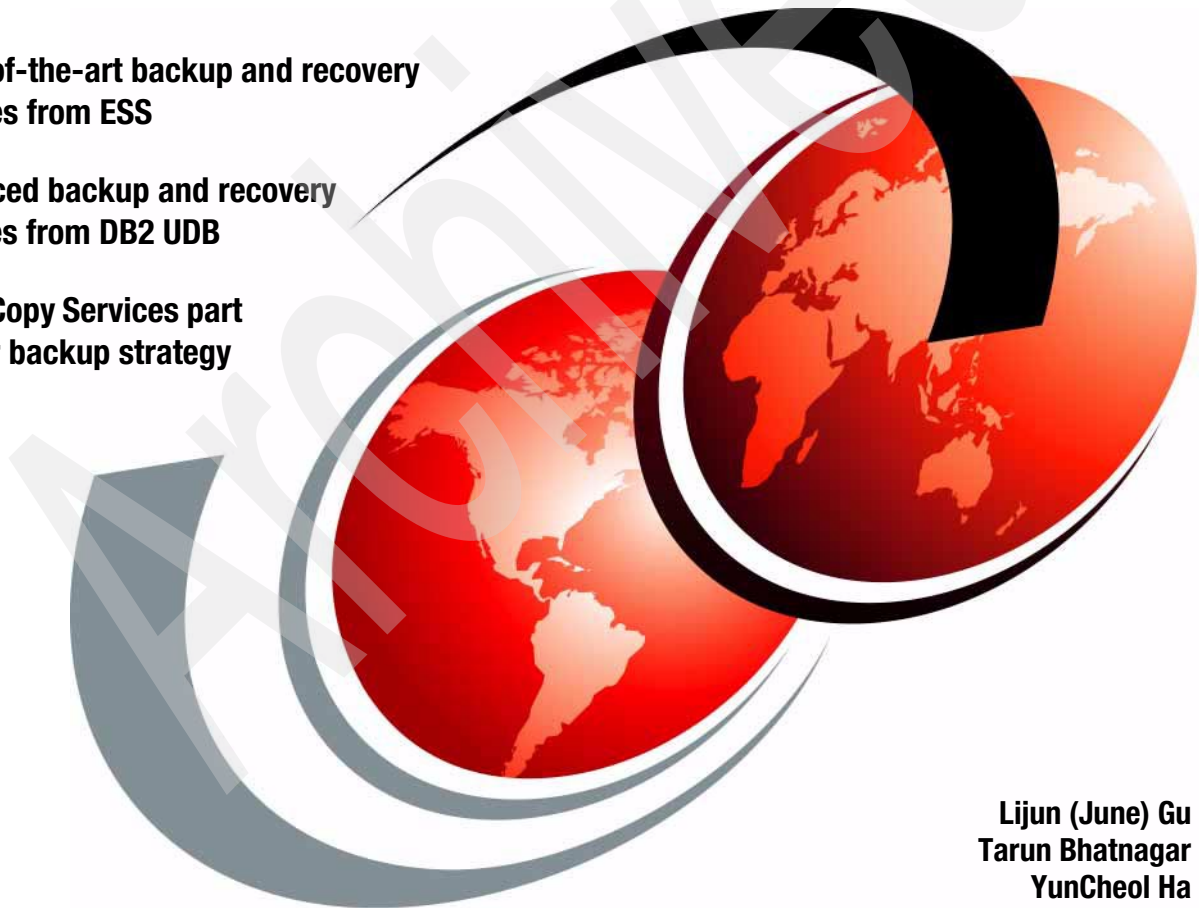


# DB2 UDB Backup and Recovery with ESS Copy Services

State-of-the-art backup and recovery  
services from ESS

Advanced backup and recovery  
features from DB2 UDB

Make Copy Services part  
of your backup strategy



Lijun (June) Gu  
Tarun Bhatnagar  
YunCheol Ha





International Technical Support Organization

**DB2 UDB Backup and Recovery  
with ESS Copy Services**

August 2002

Archived

**Take Note!** Before using this information and the product it supports, be sure to read the general information in “Notices” on page ix.

### **First Edition (August 2002)**

This edition applies to DB2 UDB Version 7.2 and the ESS F model with Copy Services.

Comments may be addressed to:  
IBM Corporation, International Technical Support Organization  
Dept. QXXE Building 80-E2  
650 Harry Road  
San Jose, California 95120-6099

When you send information to IBM, you grant IBM a non-exclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

© Copyright International Business Machines Corporation 2002. All rights reserved.

Note to U.S Government Users – Documentation related to restricted rights – Use, duplication or disclosure is subject to restrictions set forth in GSA ADP Schedule Contract with IBM Corp.

# Contents

<b>Figures</b> .....	vii
<b>Notices</b> .....	ix
Trademarks .....	x
<b>Preface</b> .....	xi
The team that wrote this redbook .....	xi
Notice .....	xiii
Comments welcome .....	xiii
<b>Chapter 1. Enterprise Storage Server</b> .....	1
1.1 ESS concepts .....	2
1.2 Introduction to DB2 UDB .....	13
1.2.1 DB2 Universal Database packaging .....	13
1.2.2 The universal database .....	14
1.2.3 DB2 UDB optimizer .....	16
1.2.4 DB2 UDB utilities .....	17
<b>Chapter 2. DB2 UDB, NAS and SAN terminology and concepts</b> .....	21
2.1 DB2 terminology and concepts .....	22
2.1.1 Instances .....	22
2.1.2 Databases .....	22
2.1.3 Buffer pools .....	23
2.1.4 Page cleaners .....	23
2.1.5 Table spaces .....	24
2.1.6 Containers .....	25
2.1.7 Page size .....	26
2.1.8 Extent size .....	27
2.1.9 Prefetch size .....	27
2.1.10 Tables, indexes, and long data .....	28
2.1.11 DB2 UDB and parallelism .....	29
2.1.12 Registry and environment variables .....	30
2.1.13 A word about DB2EMPFA .....	32
2.2 File protocols .....	32
2.2.1 Network file system protocols .....	33
2.2.2 File I/O .....	34
2.2.3 Local Area Networks (LAN) .....	35
2.3 Storage Area Network terminology and concepts .....	36
2.3.1 SAN storage .....	36

2.3.2 SAN fabric . . . . .	36
2.3.3 SAN applications . . . . .	36
<b>Chapter 3. DB2 UDB backup and recovery . . . . .</b>	<b>39</b>
3.1 Backup and recovery . . . . .	40
3.1.1 Logging . . . . .	40
3.1.2 Backup feature . . . . .	41
3.1.3 Recovery feature . . . . .	47
3.1.4 Incremental backup and recovery . . . . .	54
3.2 Split mirror support . . . . .	59
3.2.1 High availability requirements . . . . .	59
3.2.2 Suspending database . . . . .	59
3.2.3 Initializing database . . . . .	61
<b>Chapter 4. Planning for ESS Copy Services . . . . .</b>	<b>67</b>
4.1 FlashCopy . . . . .	68
4.1.1 Overview . . . . .	68
4.1.2 Planning for FlashCopy on ESS . . . . .	70
4.1.3 Operational considerations . . . . .	72
4.1.4 Performance considerations . . . . .	73
4.2 Peer-to-Peer Remote Copy (PPRC) . . . . .	75
4.2.1 Terminology . . . . .	75
4.2.2 Overview . . . . .	75
4.2.3 Planning for PPRC on an ESS . . . . .	78
4.2.4 Configuring ESS for PPRC . . . . .	82
4.2.5 Performance considerations . . . . .	86
4.3 ESS Copy Services Web interface . . . . .	88
<b>Chapter 5. Scenarios . . . . .</b>	<b>89</b>
5.1 Test setup and configuration . . . . .	90
5.1.1 DB2 UDB and storage configuration . . . . .	90
5.1.2 Network topology . . . . .	92
5.1.3 Snapshot infrastructure . . . . .	93
5.2 Snapshot backup . . . . .	94
5.2.1 FlashCopy in local site . . . . .	95
5.2.2 PPRC with FlashCopy in remote site . . . . .	99
5.2.3 PPRC with FlashCopy in local and remote site . . . . .	103
5.3 Using snapshot . . . . .	105
5.3.1 Making a clone database . . . . .	106
5.3.2 Making standby database . . . . .	112
5.3.3 Making a mirror database . . . . .	114
<b>Related publications . . . . .</b>	<b>117</b>
IBM Redbooks . . . . .	117

Other resources .....	117
Referenced Web sites .....	118
How to get IBM Redbooks .....	119
IBM Redbooks collections .....	119
<b>Index</b> .....	<b>121</b>

Archived





# Figures

1-1	Enterprise Storage System components . . . . .	6
1-2	Schematic of layout of disk drives in ESS and expansion frame . . . . .	7
1-3	A logical disk within a RAID5 array . . . . .	9
2-1	Relationship between buffer pools, table spaces, and instances . . . . .	26
2-2	Table space containers and extents . . . . .	27
2-3	NAS devices use file I/O . . . . .	35
3-1	Recover to the last backup time. . . . .	49
3-2	Recover database to the point of failure . . . . .	50
3-3	Recover tablespace to the point of failure . . . . .	52
3-4	Incremental backup type . . . . .	54
3-5	Combined incremental recovery . . . . .	56
4-1	FlashCopy overview. . . . .	69
4-2	PPRC write cycle . . . . .	76
4-3	PPRC volume states . . . . .	78
4-4	Ethernet and ESCON connection between Copy Services . . . . .	79
4-5	ESS logical paths. . . . .	83
4-6	PPRC with FlashCopy . . . . .	84
5-1	DB2 UDB and ESS mapping . . . . .	91
5-2	SAN physical topology. . . . .	92
5-3	Snapshot infrastructure . . . . .	93
5-4	FlashCopy in local site . . . . .	96
5-5	PPRC with FlashCopy in remote site. . . . .	100
5-6	PPRC with FlashCopy in local and remote site . . . . .	103
5-7	Making a clone database. . . . .	107
5-8	Version recovery . . . . .	109
5-9	Reverse PPRC recovery . . . . .	111
5-10	Hot standby database . . . . .	113
5-11	Mirror database . . . . .	115



# Notices

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

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

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

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

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

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

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

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

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


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

## **COPYRIGHT LICENSE:**

This information contains sample application programs in source language, which illustrates programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. You may copy, modify, and distribute these sample programs in any form without payment to IBM for the purposes of developing, using, marketing, or distributing application programs conforming to IBM's application programming interfaces.

# Trademarks

The following terms are trademarks of the International Business Machines Corporation in the United States, other countries, or both:

AIX®	FlashCopy®	S/390®
AIX 5L™	IBM®	Seascape®
DB2®	IMS™	Sequent®
DB2 Connect™	Informix®	SP™
DB2 Extenders™	OS/2®	StorWatch™
DB2 Universal Database™	OS/390®	System/390®
Enterprise Storage Server™	OS/400®	Tivoli®
ESCON®	Perform™	TotalStorage™
Everyplace™	Redbooks(logo)™ 	Wave®
FICON™	RS/6000®	

The following terms are trademarks of International Business Machines Corporation and Lotus Development Corporation in the United States, other countries, or both:

Working Together®

The following terms are trademarks of other companies:

ActionMedia, LANDesk, MMX, Pentium and ProShare are trademarks of Intel Corporation in the United States, other countries, or both.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

C-bus is a trademark of Corollary, Inc. in the United States, other countries, or both.

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

SET, SET Secure Electronic Transaction, and the SET Logo are trademarks owned by SET Secure Electronic Transaction LLC.

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

# Preface

This IBM Redbook describes how customers can use the IBM TotalStorage Enterprise Storage Server's Advanced Copy Services to perform a snapshot backup and recovery in an IBM DB2 UDB Windows environment. Of these Copy Services, FlashCopy, is capable of performing backups of multi-terabyte databases in minutes. It can also help improve uptime by providing the ability to recover in minutes. Another Copy Service, Peer-to-Peer Remote Copy (PPRC), provides a synchronous remote mirror to protect against disasters. When used in combination, these two Copy Services have the ability to provide multiple instant copies of the database and still protect against disaster.

IBM DB2 UDB is a universal database supporting a wide variety of platforms and applications. With every new release, DB2 UDB continues to grow in the enterprise-class server segment. When a database is used for mission-critical applications, it requires an enterprise-class storage subsystem. The Enterprise Storage Server (ESS) can provide the reliability, scalability, and features required for mission-critical applications. This redbook also provides general information about ESS and DB2 UDB.

## The team that wrote this redbook

This redbook was produced by a team of specialists from around the world working at the International Technical Support Organization, San Jose Center.

**Lijun (June) Gu** is a Project Leader at the International Technical Support Organization (ITSO), San Jose Center, California, where she conducts projects on all aspects of DB2 Universal Database (DB2 UDB) and DB2 OS/390. She is an IBM-Certified Solution Expert of DB2 UDB Database Administrator and an IBM-Certified Specialist DB2 UDB User. She has extensive experience in DB2 UDB and ADSM administration as well as database design and modeling. She holds three master's degrees: MS in Computer Science, MS in Analytical Chemistry, and MS in Soil Science.

**Tarun Bhatnagar** is an IT Architect in the Enterprise Application Development Division of IBM Global Services in New York. He has 13 years of experience in business application design, development, and implementation. He has worked at IBM for five years. His areas of expertise include Web and client server application architecture and design, development, and deployment, database design, application integration and testing. He holds a master's degree in

Computer Science and Applications from Birla Institute of Technology. He has worked on some very large customer engagements and written extensively on DB2 Extenders and Web services. He is also a Cisco Certified Network Associate.

**YunCheol Ha** is an IT Architect of Integrated Technical Services at IBM Korea. He has experience in DB2 Universal Database product installation, database administration, performance tuning, client/server connectivity, database logical/physical design, application support, health check and Query Patroller implementation on both Enterprise Edition (EE) and Enterprise-Extended Edition (EEE) systems. His previous experience includes working as a DB2 and ORACLE database administrator. His expertise is in data warehouses, data marts and operational environments.

Special thanks to the following people for their contributions in providing equipment, and information to be incorporated within these pages:

David Kim, Paul K. Lee  
IBM San Jose

Rakesh Goenka, Enzo Cialini, Dale M. McInnis  
IBM DB2 Toronto Lab

Barry Mellish  
International Technical Support Organization, San Jose Center

Martin Mezger  
IBM Germany

Ravi K. Tadigadapa  
IBM Pune, India

Thanks to the following people for their contributions to this project:

Corinne Baragoin, Will Carney, Mary Comianos, Emma Jacobs, Yvonne Lyon, Deanna Polm, Journal Saniei, Patrick Vabre, Bart Steegmans, Osamu Takagiwa, Ueli Wahli, Paolo Bruni  
International Technical Support Organization, San Jose Center

Monika Shintre  
Fujitsu Consulting

## Notice

This book is intended for Systems Administrators, Storage Specialists, Database Administrators and Database Specialists to use IBM ESS Copy Services for DB2 UDB backup and recovery. We assume a base level of understanding of one of the areas addressed: ESS and/or DB2, and intend that this book will help you to understand your area of expertise in a wider context. The information in this publication has not been subject to formal verification and is provided AS IS. The information in this publication is not intended as the specification of any programming interfaces that are provided by DB2 Universal Database (UDB) or the Extended Edition (EE). See the PUBLICATIONS section of the IBM Programming Announcement for DB2 Universal Database Extended Edition for more information about what publications are considered to be product documentation.

## Comments welcome

Your comments are important to us!

We want our Redbooks to be as helpful as possible. Send us your comments about this or other Redbooks in one of the following ways:

- ▶ Use the online **Contact us** review redbook form found at:  
[ibm.com/redbooks](http://ibm.com/redbooks)
- ▶ Send your comments in an Internet note to:  
[redbook@us.ibm.com](mailto:redbook@us.ibm.com)
- ▶ Mail your comments to the address on page ii.





# Enterprise Storage Server

This chapter gives an overview of Enterprise Storage Server. It also contains an overview of interfaces available for managing the ESS environment. Specifically, the command line interface and the Web based interface are covered. The intent here is to provide a sound conceptual understanding of these services, rather than provide every little detail.

In this chapter the following topics are covered:

- ▶ ESS overview
  - Clusters
  - RAID controllers, device adapters
  - Arrays and ranks
  - Logical disks, LUNs
  - Host adapters
- ▶ DB2 overview
  - Product packaging
  - Universal database
  - Optimizer
  - Utilities

## 1.1 ESS concepts

As businesses become more and more dependent on information technology to conduct their operations and stay competitive, the availability of their processing facilities becomes crucial. Today, most businesses require a high level of availability, which extends to continuous availability, 24 hours a day and seven days a week (24x7) operation. Therefore, the ability to provide continuous availability for the major applications is more often than not a necessity for business survival.

These important demands are fulfilled with the Enterprise Storage Server and its Advanced Copy Services. The ESS Copy Services provides replication of mission critical data; point-in-time with FlashCopy, dynamic synchronous mirroring to a remote with Peer-to-Peer Remote Copy (PPRC) and asynchronous copying to a remote site with Extended Remote Copying (XRC). This last function is only available in a System/390 environment.

### The Seascape architecture

Seascape is a blueprint for comprehensive storage solutions optimized for a connected world. The Seascape architecture integrates leading technologies from IBM, including disk storage, tape storage, optical storage, powerful processors, and rich software function, to provide highly reliable, scalable, versatile, application-based storage solutions that span the range of servers from PCs to supercomputers.

At its heart, Seascape architecture uses an open, industry-standard storage server that can scale up exponentially in both power and performance. Since the storage servers can integrate snap-in building blocks and software upgrades, you can quickly deploy new or improved applications and rapidly accommodate new data and media types. In this way, Seascape storage servers become an essential element for storing, manipulating, and sharing data across the network.

### What RAID is

RAID (redundant array of independent disks) is a way of storing the same data in different places on multiple hard disks. By placing data on multiple disks, I/O operations can overlap in a balanced way, improving performance. Since multiple disks increases the mean time between failure (MTBF), storing data redundantly also increases fault-tolerance.

A RAID appears to the operating system to be a single logical hard disk. RAID employs the technique of *striping*, which involves partitioning each drive's storage space into units ranging from a sector (512 bytes) up to several megabytes. The stripes of all the disks are interleaved and addressed in order.

In a single-user system where large records are stored, the stripes are typically set up to be small (perhaps 512 bytes) so that a single record spans all disks and can be accessed quickly by reading all disks at the same time.

In a multi-user system, better performance requires establishing a stripe wide enough to hold the typical or maximum size record. This allows overlapped disk I/O across drives.

## **Types of RAID**

There are at least nine types of RAID plus a non-redundant array (RAID-0). Here, we will give an overview of RAID-0 through RAID-5.

### ***RAID-0***

This technique has striping but no redundancy of data. It offers the best performance but no fault-tolerance. RAID 0 is a performance oriented striped data mapping technique. Uniformly sized blocks of storage are assigned in regular sequence to all of an array's disks. RAID 0 provides high I/O performance at low inherent cost. (No additional disks are required). The reliability of RAID 0, however is less than that of its member disks due to its lack of redundancy. Despite the name, RAID Level 0 is not actually RAID, unless it is combined with other technologies to provide data and functional redundancy, regeneration and rebuilding.

### ***RAID-1***

This type is also known as disk mirroring and consists of at least two drives that duplicate the storage of data. There is no striping. Read performance is improved since either disk can be read at the same time. Write performance is the same as for single disk storage. RAID-1 provides the best performance and the best fault-tolerance in a multi-user system. RAID-1, also called mirroring, is popular because of its simplicity and high level of reliability and availability. Mirrored arrays consist of two or more disks. Each disk in a mirrored array holds an identical image of user data. A RAID-1 array may use parallel access for high transfer rate when reading. More commonly, RAID-1 array members operate independently and improve performance for read-intensive applications, but at relatively high inherent cost.

### ***RAID-2***

This type uses striping across disks with some disks storing error checking and correcting (ECC) information. It has not been widely deployed in industry largely because it requires special disk features. Since disk production volumes determine cost, it is more economical to use standard disks for RAID systems.

### ***RAID-3***

This type uses striping and dedicates one drive to storing parity information. The ECC information is used to detect errors. Data recovery is accomplished by calculating the exclusive OR (XOR) of the information recorded on the other drives. Since an I/O operation addresses all drives at the same time, RAID-3 cannot overlap I/O. For this reason, RAID-3 is best for single-user systems with long record applications. RAID-3 adds redundant information in the form of parity to a parallel access striped array, permitting regeneration and rebuilding in the event of a disk failure. One stripe of parity protects corresponding strip's of data on the remaining disks. RAID-3 provides for high transfer rate and high availability, at an inherently lower cost than mirroring. Its transaction performance is poor, however, because all RAID-3 array member disks operate in lockstep

### ***RAID-4***

This type uses large stripes, which means you can read records from any single drive. This allows you to take advantage of overlapped I/O for read operations. Since all write operations have to update the parity drive, no I/O overlapping is possible. RAID-4 offers no advantage over RAID-5. Like RAID Level 3, RAID Level 4 uses parity concentrated on a single disk to protect data. Unlike RAID Level 3, however, a RAID Level 4 array's member disks are independently accessible. Its performance is therefore more suited to transaction I/O than large file transfers. RAID Level 4 is seldom implemented without accompanying technology, such as write-back cache, because the dedicated parity disk represents an inherent write bottleneck.

### ***RAID-5***

This type includes a rotating parity array, thus addressing the write limitation in RAID-4. Thus, all read and write operations can be overlapped. RAID-5 stores parity information but not redundant data (but parity information can be used to reconstruct data). RAID-5 requires at least three and usually five disks for the array. It's best for multi-user systems in which performance is not critical or which do few write operations. By distributing parity across some or all of an array's member disks, RAID Level 5 reduces (but does not eliminate) the write bottleneck inherent in RAID-4. As with RAID-4, the result is asymmetrical performance, with reads substantially outperforming writes. To reduce or eliminate this intrinsic asymmetry, RAID-5 is often augmented with techniques such as caching and parallel multiprocessors.

### **ESS overview**

The first of IBM's Enterprise Storage Systems (ESS) were introduced in 1999 to meet the need for high performance, scalable, flexible, and available storage systems with advanced management capabilities. The most recent product in IBM's Seascape architecture family, the ESS, sometimes referred to as Shark, is

a SAN-ready disk storage system providing universal access across all major server platforms. It employs advanced hardware and software technologies to deliver breakthrough levels of performance and maximize data sharing across the enterprise.

The IBM Enterprise Storage System can be configured in a variety of ways to provide scalability in capacity and performance. It employs integrated data caching (both volatile and non-volatile), hardware-level RAID5 support, and redundant systems at all levels.

It provides easily configurable shared or secure access to user-variable quantities of storage via SCSI, Fibre Channel, ESCON or FICON I/O interfaces. A single ESS unit can house up to 11 TB of usable protected storage. Up to 32 GB of cache can be installed in 8 GB increments.

ESS RAID5 storage is configurable by the customer via a convenient Web interface. It can be subdivided into logical disks which are then automatically configured to emulate disk device types that are compatible with the attached host computer systems. Because RAID5 storage is always striped in Enterprise Storage Servers, the I/O benefits of multiple active spindles and parallelism are immediately available; and RAID5 means all data is protected against device failure.

Multiple data paths, both internally and externally, can be specified to each logical disk created on an Enterprise Storage System. This multiplicity both enhances availability and increases I/O bandwidth. It also means that multiple host systems can be attached to either the same (shared) device, or to nonshared devices. Figure 1-1 gives an overview of ESS components.

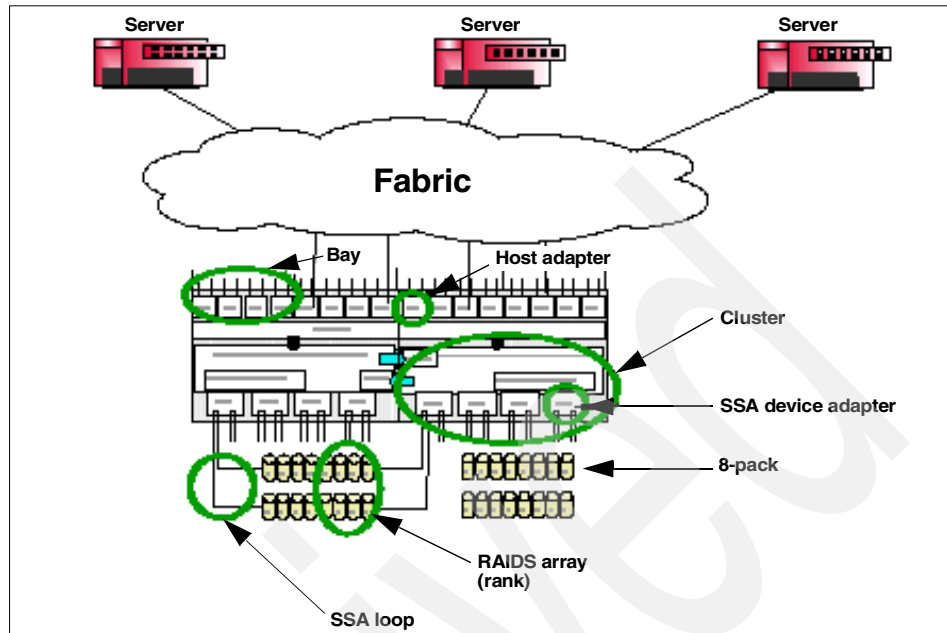


Figure 1-1 Enterprise Storage System components

## Clusters

In the larger computer world, clusters are groupings of stand-alone systems (normally referred to as nodes), which have been networked together to provide high-availability or high performance configurations. When talking about enterprise storage systems, a cluster is a collection of processors, cache, nonvolatile RAM (also known as nonvolatile storage, or NVS), and SSA disk adapters. Each ESS contains two such clusters. Each cluster normally controls one-half of the storage within the ESS, but can assume control of the other half if its partner should fail. Each cluster owns separate cache, disk adapters, and arrays, so under normal conditions does not affect performance of the other cluster. These clusters run separate operating kernels, and communicate via an internal network connection. You can communicate with each cluster via the Web interface tool, ESS Specialist.

## SSA disks, disk adapters, and RAID controllers

The ESS can contain up to 384 disks in a variety of capacities. Currently there are 9, 18 and 36 GB disk drives available. Although the same disk technology and capacities are available for SCSI, FC, FC-AL, ESCON and FICON attachment, ESS internally uses the Serial Storage Architecture (SSA) protocol

for performing disk I/O operations. The SSA disk adapters (DA) provide almost all of the RAID functions within an ESS, such as parity calculations, disk rebuild and sparing, and so on. This provides performance benefits by off loading function from the central processors and cache.

The ESS hardware is configured in a base frame and expansion rack. The base frame of the ESS can hold up to 128 disk drives which are installed in up to sixteen 8-packs. The base unit is divided into two cages, each cage holding up to eight 8-packs. The expansion frame can hold up to 32 8-packs, these are housed in 4 cages. Thus the expansion unit can hold up to a total of 256 disk drives.

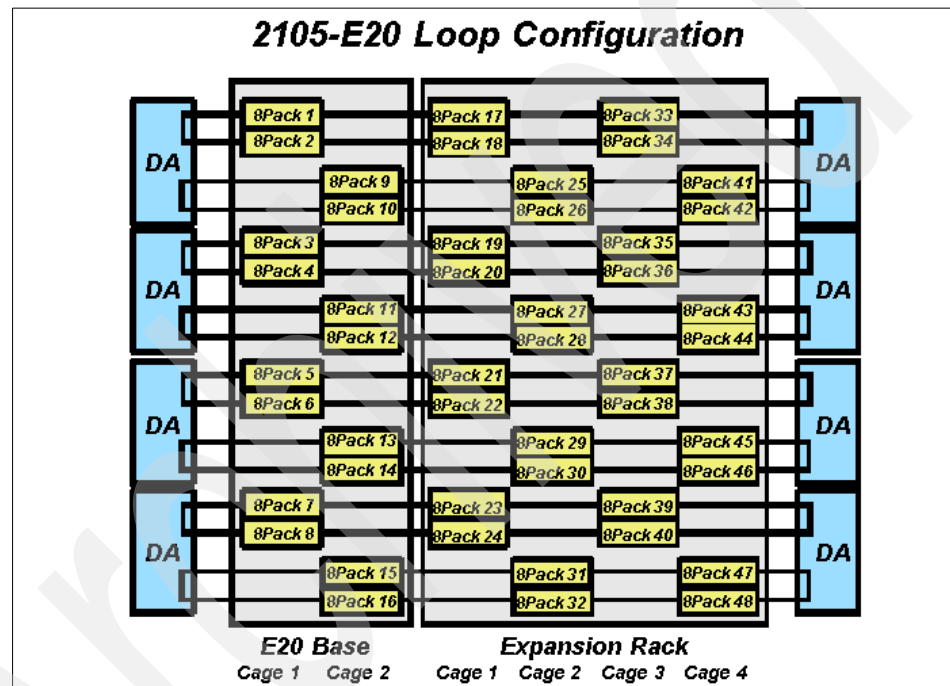


Figure 1-2 Schematic of layout of disk drives in ESS and expansion frame

## SSA loops

SSA uses a non-arbitrated loop communication protocol, in which each disk adapter (DA) can access all devices on the loop. Spatial reuse allows data transfers from all of the devices to be multiplexed on the loop, allowing bandwidth to increase as disk adapters and disks are added to the loops. It also makes it possible to “twin-tail” devices, which means that two adapters can be connected to the same device. Enterprise storage systems use twin-tailing to provide high availability. If a cluster or device adapter should fail, the other cluster can access the storage from the twin adapter.

## 8-packs

An 8-pack is a physical collection of eight disk drives, and these disks are added to enterprise storage systems in multiples of 8-packs.

## Arrays or ranks

An ESS array (also called a rank) is a collection of disks, with data striped across the disks for performance, and parity information stored to protect user data from disk failures. ESS configures its arrays using state-of-the-art RAID5 technology.

ESS provides RAID-5 arrays in sizes of either seven or eight disks. In addition, hot spare disks are reserved in case of disk failure. A RAID5 array is an array configured for high availability of data while minimizing disk usage. Data and parity information is striped across the disks in the array in such a way that if one disk were to fail completely, the data on that disk could be reconstructed on the spare disk using the data and parity information from the other disks of the array. This is commonly called a n-plus-parity configuration, where n is either six or seven disks. To avoid “hot spots,” parity information is striped across all disks, not localized on a single dedicated disk.

Two spare disks are provided within each SSA loop. Single-frame ESS configurations (which have 128 or fewer disks) contain 6+p arrays, with the remaining disks used as spares. When the disk system capacity is increased, 7+p arrays are added.

There is some performance degradation during the period of time after a disk fails and before the data is reconstructed on the spare disk, which is highly workload dependent. Striping data across multiple arrays can help minimize this impact.

**Note:** Do not confuse 8-packs (the physical packaging) with arrays. Arrays typically span two or more 8-packs.

## Logical disks, LUNs and volumes

The storage administrator uses the ESS Specialist software to allocate storage as ESS logical disks. These logical disks appear to the operating system as physical disks, but in fact are logical storage allocated across the multiple physical disks in an ESS RAID array and do not span ESS RAID arrays. Logical disks are always created by “striping” across the array disks. This means that a section of disk space, called a “strip”, is claimed first from one disk, then the next, then the next, in a round-robin fashion until a logical disk of the required size has been built. ESS strips for fixed block devices (non S/390) are 32 kilobytes in size, so when data is read from or written to a logical disk, each 32 kilobytes transferred travels to or from a different disk.



Logical disk, LUNs and Volumes are all terms used to describe the resources outlined in Figure 2-3. This figure illustrates the construction of a logical disk by striping across an ESS RAID-5 array.

**Important:** ESS logical disks are presented to the host operating system as if they were physical disk drives, such as hdisks. If there are multiple paths they are seen as vpaths. Thus, for example, an ESS logical disk appears as an AIX physical disk or volume. The terms used for the grouping of the ESS disk resources are very similar to terms used at the operating system level. When discussing resources at this level, it is important to keep in mind who's point of view you are referencing. Figure 1-3 shows a logical disk.

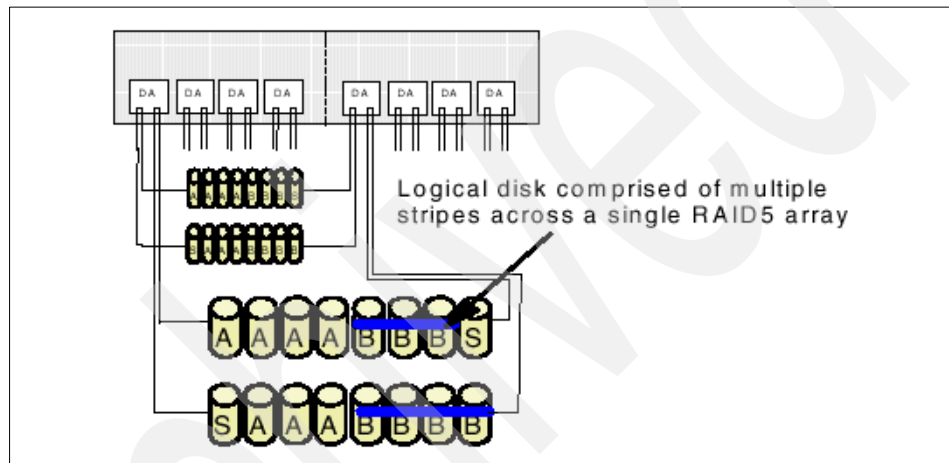


Figure 1-3 A logical disk within a RAID5 array

Logical disks are used to improve both manageability and performance. It is easier to manage a logical disk dedicated to a single function than to manage several smaller pieces of disks. Logical disks can also provide improved performance, especially in a striped RAID configuration. ESS supports many concurrent host accesses to a single logical disk, using all of the disks in an array simultaneously. Fibre channel and SCSI protocol supports this through command-tag-queuing, and OS/390 supports this through Parallel Access Volumes. It should be noted that the ESS does not provide any record or file locking if multiple hosts access the same logical disk. This is normally controlled by the application which will require a distributed lock manager or similar mechanism.

### ***Terminology considerations***

ESS logical disks are sometimes called volumes (which can sometimes cause confusion with equivalent software terms, such as AIX or Veritas logical volumes). They are also sometimes erroneously called LUNs. LUN (logical unit number) was borrowed from SCSI terminology because an ESS logical disk will have a SCSI LUN address when Fibre channel/SCSI technology is used to connect the ESS to its host computer. This is the correct usage of the term LUN.

**Note:** Throughout this redbook, we will always use the term “logical disk” to identify the logical disk created as a series of stripes across a RAID5 array and presented to the host operating system as if it were a physical device.

### **Host adapters**

Host adapters are used to connect the ESS data paths to host computers. They may be SCSI, UltraSCSI, Fibre Channel, ESCON or FICON adapters. Each has its own maximum bandwidth, which should be considered when calculating throughput to your logical disks. For open systems Fibre Channel gives the best performance.

The host adapters reside in service bays, four adapters per bay. Servicing an adapter requires swapping out an entire bay. Therefore, high-availability multipathing configurations should span at least two bays. For optimal performance, if there are two host adapters, bays 1 and 3, or bays 2 and 4 should be used.

### **Host attachment, LUN-masking, and multipathing**

After a storage administrator creates logical disks, they can then use the ESS Specialist to specify which attached hosts and paths can access the logical disk through a technique called LUN-masking. For SCSI and Fibre-attached hosts, logical disks can be assigned as accessible through specific ESS host-adapter ports. For Fibre-Channel attached hosts, the administrator can also authorize host attachment according to the world-wide-name in the adapter in the host.

The storage administrator can make a logical disk available on more than one path, which might be connected to a single, or to multiple host systems. If the logical disk is available on more than one path in a single host system, multipathing software is used to control usage of those paths. The IBM Subsystem Device Driver (SDD) is one software component used to manage these multiple paths. SDD balances I/O traffic across those paths and diverts traffic to good paths in case of a path failure. Note that the ESS does not provide file or record locking. If multiple hosts are to access the same data it is the responsibility of the accessing application to provide locking.

In the case of AIX, each path to a logical disk appears to the operating system and system administrator as an hdisk. SDD recognizes that these separate hdisks represent a single logical disk, and constructs a logical entity (called a vpath) to represent that logical disk to the operating system. The AIX administrator then refers to the vpath when constructing AIX physical volume groups.

When an ESS is attached through a Fibre Channel switch, if switch zoning is not used, a logical disk might appear as an hdisk for every possible path combination. In other words, if a host system is attached through four ports on a switch and connected to four ports on an ESS, there are 16 possible path combinations, and hence 16 hdisks comprising the vpath.

In general a system with a smaller number of hdisks is easier to manage than one with a large number. Storage administrators may want to consider use of switch zoning, LUN-masking, and selection of larger logical disks to minimize the number of hdisks visible to each host system

## Cache management

Each cluster manages cache storage (called cluster cache) and non-volatile storage (NVS.) The total cache ranges in size from 8GB to 32GB. Whenever an application reads from or writes to an ESS, a copy of the data is stored in the cluster cache. Subsequent read operations can then be satisfied from the cluster cache until the data “ages” from the cache. In addition, modified data is later destaged from the cluster cache to the RAID arrays.

Cache management strategies — reads ESS is designed to manage its cache to fit the application's data reference patterns:

- ▶ When an application references data very randomly, the ESS is likely to choose a record-caching strategy, meaning that only the requested physical record is brought into cache. In the case of DB2, this physical record usually corresponds to a DB2 page.
- ▶ In some cases, an application might frequently access a “nearby” physical record, in which case ESS adopts a “track-caching” strategy, meaning that ESS stages additional data into cache, corresponding to an ESS logical track. (An ESS logical track is the same as the physical strip size, roughly 32KB for SCSI and Fibre-Channel attached hosts.)
- ▶ If an application accesses data sequentially, as would likely be the case for database sequential scans, database physical backups, database load operations, and database logging, then ESS recognizes this and adopts a sequential strategy. The sequential strategy incorporates the following actions:

- For sequential reads, ESS may prestage data into the ESS cache. This prestage operates asynchronously from host data requests, with the intent to have data available in the cache in time for the next request from the application. ESS usually prestages several RAID stripes, allowing multiple disks to work in parallel.
- ESS manages the cache storage to achieve a balanced efficiency between sequential and random access requests, preventing just a few sequential applications from dominating the entire cache storage. Sequential accesses tend to age more quickly from the ESS cache than random access requests.
- For sequential writes, ESS combines its cache and RAID management to make optimum use of the disk resources. It essentially converts RAID-5 sequential writes into RAID-3 operations.

### **Non-volatile storage**

The NVS provides long-term protection for modified data in case of a cluster failure. The NVS for each cluster is physically maintained within the other cluster, so that the surviving cluster contains all of the cache-modified data in case of a cluster failure. When an application writes a record, ESS stores two copies of the record in cache (one in cluster cache and the other in NVS), and immediately indicates that the write is complete when both copies are stored safely. The record is doubly-protected at this point. Later, when the data is destaged from cache to disk (and therefore doubly-protected through RAID), the storage in NVS is available for other applications.

### **Managing sequential writes**

Database logging can be a critical element for many workloads. The ESS design (hardware RAID, striping, and nonvolatile memory) work together for highly efficient logging operations. This also applies to any sequential writes, such as database loads. Here is how it all works.

- ▶ Database logging usually consists of sequences of synchronous sequential writes. The patterns may not be purely sequential, meaning that they may periodically rewrite certain physical records, but they do have a generally sequential trend. All log writes will be of size 4K or multiples of 4K. The size of the actual writes depends on the work load and machine hardware. Log archiving functions (copying an active log to an archived space) also tend to consist of simple sequential read and write sequences.
- ▶ When the DBMS writes a log record ESS stores two copies of the log record in cache (one in cluster cache and the other in NVS), and immediately indicates that the write is complete when both copies are stored safely. The log-record is doubly-protected at this point.

- ▶ After several log records are written, ESS determines that this is a sequential write pattern. When data is moved asynchronously to disk, the cache manager sends one or more full-stripes of data to the SSA disk adapter.
- ▶ The ESS cache manager and disk adapter write data to each of the disks in parallel while at the same time internally calculating the parity and sending it to disk. This technique eliminates the traditional RAID-5 write penalty and mimics the operation of a RAID-3 writes.
- ▶ This technique can be more efficient than simple mirroring, RAID-1, or RAID 1+0. For any of those techniques, every megabyte of log data written requires 2 megabytes to be transferred to disk. For ESS, a megabyte of log data written requires approximately 1.16 (16% extra for parity in a 6+P array) megabytes to be transferred to disk. This means fewer disks, less disk or memory bus bandwidth consumed.

## 1.2 Introduction to DB2 UDB

DB2 Universal Database Version 7 (DB2 UDB) is IBM's object-relational database for UNIX, Linux, OS/2, and Windows operating environments. IBM offers DB2 Universal Database packages that provide easy installation, integrated functionality, a rich bundle of development tools, full Web enablement, OLAP capabilities, and flexibility to scale and change platforms.

### 1.2.1 DB2 Universal Database packaging

DB2 UDB (V7.2) for UNIX, Windows, and OS/2 environments is available in the following package options:

- ▶ DB2 Universal Database Personal Developer's Edition (PDE) provides all the tools for one software developer to develop desktop business tools and applications for DB2 Universal Database Personal Edition.
- ▶ DB2 Universal Database Personal Edition (PE) provides a single-user object-relational database management system for your PC-based desktop that is ideal for mobile applications or the power-user.
- ▶ DB2 Universal Developer's Edition (UDE) provides all the tools required for one software developer to develop client/server applications to run on DB2 Universal Database on any supported platform. Database servers and gateways can be set up for development purposes only.
- ▶ DB2 Universal Database Workgroup Edition (WE) a multi-user object-relational database for applications and data shared in a workgroup or department setting on PC-based LANs. Ideal for small businesses or departments.

- ▶ DB2 Universal Database Enterprise Edition (EE) a multi-user object-relational database for complex configurations and large database needs for Intel to UNIX platforms and from uniprocessors to the largest SMP's. Ideal for midsize to large businesses and departments particularly where Internet and/or enterprise connectivity is important.
- ▶ DB2 Universal Database Enterprise-Extended Edition (EEE) provides a high performance mechanism to support large databases and offer greater scalability in Massively Parallel Processors (MPP's) or clustered servers. Ideal for applications requiring parallel processing, particularly data warehousing and data mining All versions of the DB2 UDB work with the ESS. This redbook focuses on the EE and EEE packages.

## 1.2.2 The universal database

DB2 UDB is truly the universal database supporting a wide variety of platforms and applications.

### Universal access

DB2 UDB provides universal access to all forms of electronic data. This includes traditional relational data as well as structured and unstructured binary information, documents and text in many languages, graphics, images, multimedia (audio and video), information specific to operations, such as engineering drawings, maps, insurance claim forms, numerical control streams, or any type of electronic information.

Access to a wide variety of data sources can be accomplished with the use of DB2 UDB and its complimentary products: Relational Connect, DB2 Connect, Data Joiner, and Classic Connect. Sources that can be accessed include: DB2 UDB for OS/390, DB2 UDB for OS/400, IMS, Oracle, MS/SQL Server, Sybase, NCR Teradata and IBM Informix databases.

### Universal application

DB2 UDB supports a wide variety of application types. It is configurable to perform well for both on-line transaction processing (OLTP) as well as decision support systems (DSS). It can also be used as the underlying database for on-line analytical processing (OLAP). DB2 UDB is also accessible from and/or integrated into a wide variety of application development environments. In addition to being able to call SQL functions from within standard programming languages, such as C/C++, COBOL, and Visual Basic, DB2 UDB fully supports Java technology and is accessible from Java applets, servlets and applications. It also participates in Microsoft's OLE DB as both a provider and a consumer.

## **Universal extensibility**

The object-relational capabilities of DB2 UDB provide for its universal Extensibility. Through user defined functions (UDFs) and user defined types (UDTs) the base data types can be extended to include data types specific to your business. The framework of UDFs and UDTs provide the basis for DB2 UDB's extensions in support of image, audio, video, text search, XML among others.

## **Universal scalability**

DB2 UDB scales from pervasive / handheld devices with the DB2 Everyplace Edition, all the way to massively parallel processing environments (MPPs) with DB2 UDB Enterprise - Extended Edition (EEE). The various editions of DB2 UDB outlined above will run on the Palmtop, Laptop, Distributed Servers, Central Servers as well as clustered servers.

The superior scalability of DB2 UDB is made possible through a combination of features that are built into the base product. These include intra-partition parallelism as well as inter-partition parallelism for queries. The database engine also takes advantage of I/O parallelism whenever possible, and features have been built-in to DB2 to recognize the unique characteristics of Disk I/O subsystems that use RAID-5, such as IBM's Enterprise Storage Server.

## **Universal reliability**

DB2 UDB runs reliably across multiple hardware and operating systems. Major reliability features include transactional integrity through database locking as well as built-in Backup and Recovery capabilities. The scheduling of backups can be automated and with V7.2 incremental backups as well as full on-line or off-line backups can be taken. Backups can also be tailored to a single Tablespace. Recovery of a DB2 database can be done to a specific point in time, and can be filtered to only restore those database objects which are required to get you up and running.

## **Universal management**

The DB2 Control Center includes a common integrated tool set to manage local and remote databases across all software and hardware client platforms from a single terminal.

Components of the Control Center include:

- ▶ The Command Center provides a GUI window for inputting database or operating system commands, while allowing for storage, retrieval and browsing of previous commands.
- ▶ The Script Center allows for the creation, modification and execution of database or operating system scripts.

- ▶ The Journal Facility for managing jobs, recovery, alerts and messages.
- ▶ The Visual Explain facility provides a graphical means to display optimization-associated cost information and visual drill down of a query access plan.
- ▶ The Event Analyzer is a flexible tool for summary and historical performance analysis.
- ▶ The Performance Monitor supports online monitoring of buffer pools, sorts, locks, I/O and CPU activity.
- ▶ Smart guides guide database administrators through tasks, such as backup/recovery, performance configuration and object definition.
- ▶ The Alert Center displays objects which are in an exception status.
- ▶ The Index creation wizard helps database administrators build the best possible indexes for a given query workload.

All of the information presented in the GUI control center can also be accessed via a command line interface.

### 1.2.3 DB2 UDB optimizer

The query optimizer is the key component in the performance of any Enterprise Database Server. IBM has made more than a 25-year investment in enhancements to DB2's cost-based, rule-driven optimizer with the goal of keeping it the best in the industry. A major component of this effort is the implementation of the Starburst extensible optimizer in DB2 UDB. This allows IBM to add new intelligence to the optimizer without having to modify the entire query-compilation process.

As DB2 UDB is enhanced with new features and functionality, the optimizer performance improves. An example is extending the optimizer to understand OLAP SQL extensions and multiple levels of parallel query processing. The latter is particularly important in clusters where each node is an SMP. IBM has also built into the optimizer knowledge about how to work with underlying disk subsystems, such as the arrays of disks inside the ESS.

The optimizer takes advantage of its knowledge of the characteristics of the hardware environment; CPU speed, I/O speed, network bandwidth (for federated queries), bufferpool allocations. In addition, it knows the characteristics of the data itself: size of the table, partitioning scheme, uniqueness, existence of indexes, existence of automatic summary tables, when choosing an optimal execution strategy for a given SQL statement. This ability becomes more important as the size of the database increases.



In addition, the optimizer can also perform query re-write automatically. This enables DB2 UDB to execute better performing queries, while keeping the results set the same. This capability is especially important for environments where the SQL is being generated by a tool — true for many decision support applications.

## 1.2.4 DB2 UDB utilities

DB2 UDB utilities are designed to support both very large databases and also small OLTP databases as well. They incorporate a highly scalable software architecture which allows them to exploit a wide variety of hardware platforms. This book does not attempt to give a complete list of the DB2 utilities, we merely outline here those utilities which have the most interaction with the ESS. For more information on DB2 UDB utilities, please refer to *DB2 UDB Administration Guide: Performance V7*, SC09-2945.

### Backup and recovery utilities

DB2 UDB provides a granular and parallel backup and restore utility. Some of the options available to backup include:

- ▶ Online or offline.
- ▶ Entire database, single tablespace or multiple tablespaces.
- ▶ For incremental backups, delta or cumulative backups can be taken.
- ▶ For partitioned databases, all partitions or a subset of the data partitions can be chosen.

The degree of parallelism achieved during the backup and restore process is determined by the number of backup devices available. This parallel capability results in a linear reduction in backup time.

DB2 UDB interfaces with storage management products, such as the IBM Tivoli Storage manager (TSM) product, which manages backup jobs and log archives from multiple servers. The TSM interface keeps track of backups and their associated logs. These products (integrated through the use of DB2 user exits) allow for the automated archival and retrieval of backups and log files to and from off-line storage.

### Data movement utilities

There are some specialized utilities designed to help you move data into and within the database.

### ***Load utility***

DB2 UDB provides the ability to load large volumes of data into the database. The load utility bypasses the transaction logging mechanism, thus decreasing the elapsed time required to load data. The Load Utility utilizes multiple CPU engines in an SMP environment. It can execute in parallel by tablespace. The DB2 UDB Load Utility supports the creation of indexes, backups and catalog statistics gathering (RUNSTATS) during the load execution. It also supports the ability to load data from remote systems.

### ***Autoloader***

In a partitioned environment, the autoloader utility is used to perform the necessary steps to load data from single or multiple flat files into a partitioned table. It splits data according to partition keys, it pipes the data to the appropriate partition, and concurrently loads data into each partition of the table. The Autoloader supports SMP parallelism within a partition as well as cross-partitions.

### ***Data redistribution***

For the DB2 UDB EEE partitioned database, the redistribute utility is used to move data between partitions when a new logical data partition is added or removed. It can also be used to redistribute data if data distribution across the existing data partitions is not uniform.

### ***Table reorganization***

The reorganization utility (REORG) re-clusters the rows of a table on disk in order according to a specified index. Clustering of each individual table provides enhanced performance. REORG can employ I/O parallelism when data is spread across multiple containers, and also executes in parallel for partitioned databases.

### ***Export***

The Export Utility allows for the extraction of data from the database in preparation for the movement of data from one database system to another, or to move from relational format to other formats for use in other applications, such as spreadsheet programs, among others.

### ***Import***

The Import Utility inserts data from an input file into a table or view.

### ***DB2MOVE***

The db2move utility allows data to be moved from one DB2 UDB database to another. These databases can reside on different servers, and can even be on different platforms.

### ***DB2LOOK***

The db2look utility can be used to obtain a point-in-time view of the characteristics of an existing database. It can extract the statistics as well as the required DDL to re-create the database on another server.

Archived



## **DB2 UDB, NAS and SAN terminology and concepts**

When establishing a DB2 UDB server environment that takes advantage of a network attached storage or Storage Area Network, it is helpful to have an understanding of some of the terminology and concepts that may be encountered in product documentation. This chapter is designed to introduce some of the DB2 UDB, NAS, and SAN terminology and concepts that is used in this book.

## 2.1 DB2 terminology and concepts

In this section we describe some of the terminology and concepts that are related to DB2 Universal Database.

### 2.1.1 Instances

DB2 Universal Database sees the world as a hierarchy of several different types of objects. Workstations on which any edition of DB2 Universal Database has been installed are known as system objects and they occupy the highest level of this hierarchy. Systems objects can represent systems that are accessible to other DB2 clients or servers within a network, or they can represent stand-alone systems that neither have access to nor can be accessed from other DB2 clients or servers.

When any edition of DB2 Universal Database is installed on a particular workstation (or system), program files for the DB2 Database Manager are physically copied to a specific location on that workstation and one instance of the DB2 Database Manager is created and assigned to the system as part of the installation process. (Instances comprise the next level in the object hierarchy.) If needed, additional instances of the DB2 Database Manager can be created for a particular system; multiple instances can be used to separate the development environment from the production environment, tune the DB2 Database Manager for a particular environment, and protect sensitive information from a unauthorized access.

Each time a new instance is created, it references the DB2 Database Manager program files that were stored on that workstation during the installation process; thus, each instance behaves like a separate installation of DB2 Universal Database, even though all instances within a particular system share the same binary code. Although all instances share the same physical code, each can be run concurrently with the others and each has its own environment, which can be modified by altering the contents of its configuration file.

### 2.1.2 Databases

In its simplest form, a DB2 Universal Database database is a set of related database objects. In fact, when you create a DB2 UDB database, you are establishing an administrative relational database entity that provides an underlying structure for an eventual collection of database objects (such as tables, views, indexes, and so on). This underlying structure consists of a set of

system catalog tables (along with a set of corresponding views), a set of table spaces in which both the system catalog tables and the eventual collection of database objects will reside, and a set of files that will be used to handle database recovery and other bookkeeping details.

DB2 UDB allows multiple databases to be defined within a single database instance. Each database has its own configuration file, which allows characteristics of the database, such as memory usage and logging, to be fine tuned for optimum performance.

### 2.1.3 Buffer pools

A buffer pool is an area of main memory that has been allocated to the DB2 Database Manager for the purpose of caching table and index data pages as they are read from disk or modified. Using a set of heuristic algorithms, the DB2 Database Manager prefetches pages of data that it thinks a user is about to need into one or more buffer pools and it moves pages of data that it thinks are no longer needed back to disk. This approach improves overall system performance because data can be accessed much faster from memory than from disk. (The fewer times the DB2 Database Manager needs to perform disk I/O, the better the performance.)

Each time a new database is created, one buffer pool, named IBMDEFAULTBP, is also created as part of the database initialization process. On UNIX platforms, this buffer pool consists of 1,000 4K (kilobyte) pages of memory; on all other platforms, this buffer pool consists of 250 4K pages of memory.

**Attention:** The defaults of buffer pools must be always carefully reviewed and changes are needed in most cases.

### 2.1.4 Page cleaners

To prevent a buffer pool from becoming full, page cleaner agents are used to write modified pages to disk at a predetermined interval (by default, when the buffer pool is 60 percent full) to guarantee the availability of buffer pool pages for future read operations. For example, if you have updated a large amount of data in a table, many data pages in the buffer pool may be updated but not written into disk storage (these pages are known as “dirty pages”). Since prefetchers cannot place fetched data pages onto the dirty pages in the buffer pool, these dirty pages must be flushed to disk storage so that prefetchers can store needed data pages in the buffer pool.

## 2.1.5 Table spaces

When setting up a new database, one of the first tasks that must be performed is to map the logical database design to physical storage on a system. This is where table spaces come into play. Table spaces are used to control where data in a database is physically stored on a system and to provide a layer of indirection between the database and the container objects in which the actual data resides.

When a database is first created, the following three table spaces are also created and associated with the default buffer pool IBMDEFAULTBP as part of the database initialization process:

- ▶ A catalog table space named SYSCATSPACE, which is used to store the system catalog tables and views associated with the database.
- ▶ A user table space named USERSPACE1, which is used to store all user-defined objects (such as tables, indexes, and so on) along with user data.
- ▶ A temporary table space named TEMPSPACE1, which is used to store temporary tables that might be created in order to resolve a query.

Additional table spaces can be created as needed.

All table spaces are classified according to how their storage space is managed: A table space can be either a system managed space (SMS) or a database managed space (DMS). With SMS table spaces, the operating system's file manager is responsible for allocating and managing the storage space used by the table space. SMS table spaces typically consist of several individual files (representing data objects such as tables and indexes) that are stored in the file system. With database managed space (DMS) table spaces, the table space creator (and in some cases, the DB2 Database Manager) is responsible for allocating the storage space used and the DB2 Database Manager is responsible for managing it. Essentially, a DMS table space is an implementation of a special-purpose file system that has been designed specifically to meet the needs of the DB2 Database Manager.

Regardless of how they are managed, three types of table spaces can exist: regular, temporary, and long. Tables that contain user data can reside in regular DMS table spaces. (Indexes can also be stored in regular DMS table spaces.) Tables that contain long field data or large object (LOB) data, such as multimedia objects, can reside in long DMS table spaces. Temporary table spaces are classified as either system or user; system temporary table spaces are used to



store internal temporary data that is required during SQL operations such as sorting, reorganizing tables, index creation, and table joins. User temporary table spaces are used to store declared global temporary tables that, in turn, are used to store application specific temporary data.

## 2.1.6 Containers

Every table space is made up of at least one container, which is essentially an allocation of physical storage that the DB2 Database Manager is given unique access to. Containers essentially provide a way of defining what location on a specific storage device will be made available for storing database objects.

Containers may be assigned from file systems by specifying a directory; such containers are identified as PATH containers. Containers may also reference files which reside within a directory. These types of containers are identified as FILE containers and, when used, a specific file size must be specified. Finally, containers may also reference raw devices. Such containers are identified as DEVICE containers, and the device specified must already exist on the system before a DEVICE container can be used. A single table space can span many containers, but each container can only belong to one table space.

Figure 2-1 illustrates the relationship between buffer pools, table spaces, and containers.

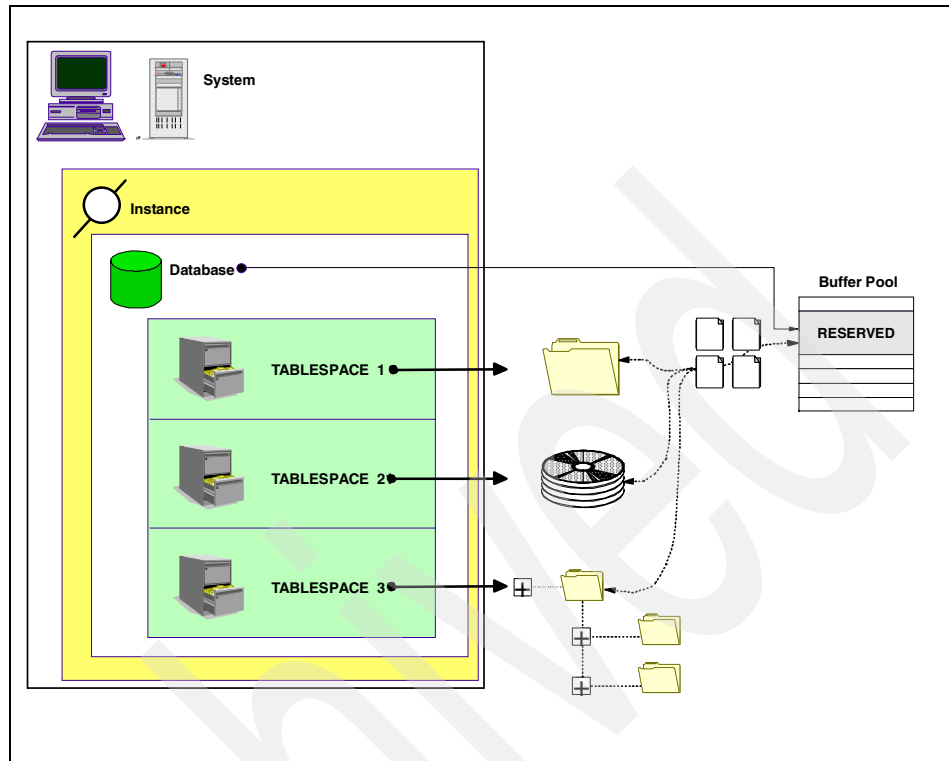


Figure 2-1 Relationship between buffer pools, table spaces, and instances

## Characteristics that affect table space performance

In addition to how a table space is managed, three other table space characteristics must be taken into consideration in order to design a database for optimum performance. These characteristics are

- ▶ Page size
- ▶ Extent size
- ▶ Prefetch size

### 2.1.7 Page size

With DB2 UDB, data is transferred between table space containers and buffer pools in discrete blocks that are called *pages*. (The memory reserved to buffer a page transfer is called an I/O buffer.) The actual page size used by a particular table space is determined by the page size of the buffer pool the table space is associated with. Four different page sizes are available: 4K, 8K, 16K, and 32K. By default, all table spaces that are created as part of the database creation process are assigned a 4K page size.

## 2.1.8 Extent size

An *extent* is a unit of space within a container that makes up a table space. When a table space spans multiple containers, data associated with that table space is stored on all of its respective containers in a round-robin fashion; the extent size of a table space represents the number of pages of table data that are to be written to one container before moving to the next container in the list. This helps balance data across all containers that belong to a given table space (assuming all extent sizes specified are equal). Figure 2-2 illustrates how extents are used to balance data across multiple containers.

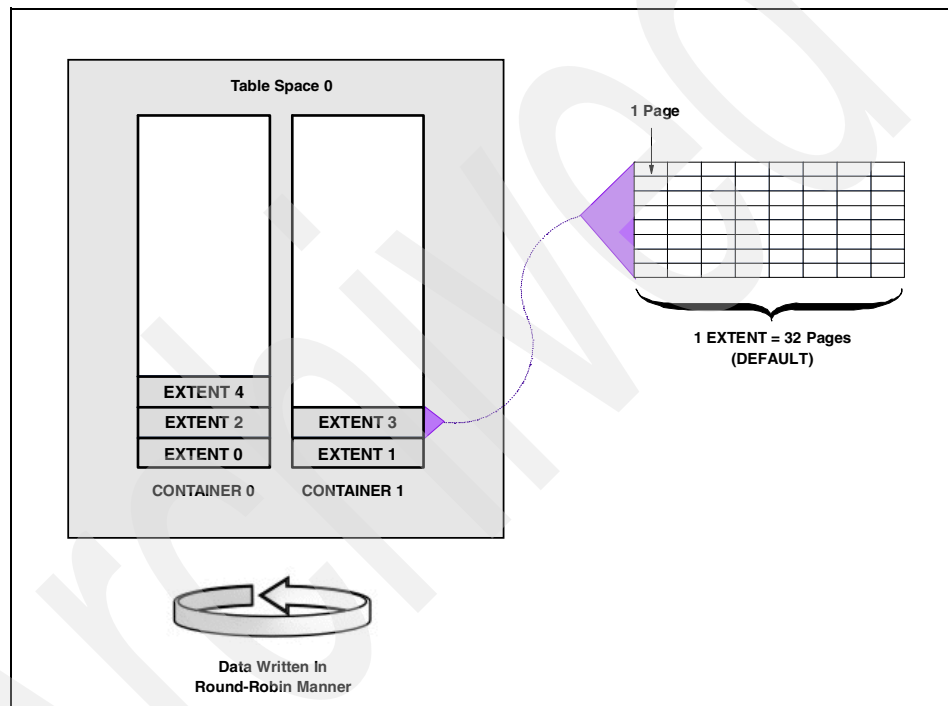


Figure 2-2 Table space containers and extents

## 2.1.9 Prefetch size

Prefetching is a technique that the DB2 Database Manager uses to fetch pages of data that it thinks a user is about to need into one or more buffer pools before requests for the data are actually made. Thus, the prefetch size of a table space identifies the number of pages of table data that are to be read in advance of the

pages currently being referenced by a query, in anticipation that they will be needed to resolve the query. The overall objective of sequential prefetching is to reduce query response time. This can be achieved if page prefetching can occur asynchronously to query execution.

Sequential prefetches read consecutive pages into the buffer pool before they are needed. List prefetches however, are more complex — in this case the DB2 optimizer attempts to optimize the retrieval of randomly located data.

The amount of data being prefetched determines the amount of parallel I/O activity. Ordinarily the database administrator should define a prefetch value large enough to allow parallel use of all of the available containers, and therefore all of the physical devices used.

## 2.1.10 Tables, indexes, and long data

If you look closely at how most data is stored in a database, you will find that it is stored as three separate objects: as a data object, which is where regular user data is stored; as an index object, which is where indexes defined on the table are stored; and as a long field object, which is where long field data is stored if the table contains one or more long data columns. Each of these three objects is stored separately and each can be stored in its own table space provided DMS table spaces are used.

### Tables

Tables are uniquely identified units of storage that are maintained within a table space. Each table is a logical structure that is used to present data as a collection of unordered rows with a fixed number of columns. Every column contains a set of values of the same data type (or one of its subtypes) and the definition of the columns in a table make up the table structure (the rows contain the actual table data). The storage representation of a row is called a *record*, and the storage representation of a column is called a *field*. Each intersection of a row and column in a database table contains a specific data item called a *value*. Data in a table is typically logically related and additional relationships, known as referential constraints, can be defined between two or more tables.

### Indexes

An index is an object that contains an ordered set of pointers that refer to a key in a base table. When indexes are used, the DB2 Database Manager can access data directly and more efficiently because each index provides a direct path to the data through pointers that have been ordered based on the values of the columns that the index is associated with. When an index is created, the DB2 Database Manager uses a balanced binary tree (a hierarchical data structure in

which each element may have at most one predecessor but may have many successors) to order the values of the key columns in the base table that the index refers to.) More than one index may be defined for a given table, and they provide a way to assist in the clustering of data.

## Long data

All data is classified, to some extent, according to its type (for example, some data might be numerical, whereas other data might be textual). Because a table is comprised of one or more columns that are designed to hold data values, each column must be assigned a specific data type. This data type determines the internal representation that will be used to store the data, what the ranges of the data's values are, and what set of operators and functions can be used to manipulate that data once it has been stored.

**Attention:** This data type is not cached.

DB2 Universal Database supports 19 different built-in data types (along with an infinite number of user-defined data types that are based on the built-in data types). Of these built-in data types, five are designed to store data values that exceed 32,700 bytes in length:

- ▶ Varying-length long character string (LONG VARCHAR)
- ▶ Varying-length double-byte long character string (LONG VARGRAPHIC)
- ▶ Binary large object (BLOB)
- ▶ Character large object (CLOB)
- ▶ Double-byte character large object (DBCLOB)

These objects, although logically referenced as part of the table, may be stored in their own table space when the base table is stored in a DMS table space. This allows for more efficient access of both the long data and the related table data.

### 2.1.11 DB2 UDB and parallelism

DB2 UDB uses *parallelism* to optimize performance when accessing a database. There are different ways in which a task can be performed in parallel. Three factors — the nature of the task, the database configuration, and the hardware environment — determine how DB2 will perform a task in parallel. Using these factors, DB2 UDB can initiate any of the following types of parallelism:

- ▶ I/O parallelism
- ▶ Query parallelism

## I/O Parallelism

Parallel I/O refers to the process of writing to, or reading from, two or more I/O devices simultaneously. The DB2 Database Manager can take advantage of parallel I/O in situations where multiple storage containers exist for a single table space. When used, I/O parallelism can provide significant improvements in data throughput.

## Query Parallelism

Query parallelism controls how database operations are performed. DB2 UDB supports two different types of query parallelism: *inter-query parallelism* and *intra-query parallelism*. Inter-query parallelism refers to the ability of multiple applications to query a database at the same time. Each query executes independently of the others, but all are executed at the same time. When intra-partition parallelism is used, what is usually considered to be a single database operation such as index creation, database loading, or an SQL query is subdivided into multiple parts, many or all of which can be run in parallel within a single database partition.

Intra-query parallelism refers to the simultaneous processing of individual parts of a single query, using either *intra-partition parallelism*, *inter-partition parallelism*, or both. When inter-partition parallelism is used, what is usually considered to be a single database operation is subdivided into multiple parts, many or all of which are run in parallel across multiple partitions of a partitioned database. Inter-partition parallelism only applies to DB2 UDB Enterprise-Extended Edition (EEE).

### 2.1.12 Registry and environment variables

In addition to DB2 Database Manager and database configuration parameters, DB2 UDB utilizes several registry and environment variables to configure the system where DB2 UDB has been installed. Because changes made to registry and environment variables impact the entire system, changes should be carefully considered before they are made. (The system command DB2SET is used to display, set, or remove DB2 registry profile variables.) After setting any registry variable, the DB2 Database Manager must be stopped (db2stop), and then restarted (db2start), in order for the changes to take effect.

Two registry variables that should be set when working with NAS are:

- ▶ DB2\_PARALLEL\_IO
- ▶ DB2\_STRIPED\_CONTAINERS

## DB2\_PARALLEL\_IO

When reading data from, or writing data to table space containers, DB2 may use parallel I/O if the number of containers in the database is greater than 1.

However, there are situations when it would be beneficial to have parallel I/O enabled for single container table spaces. For example, if the container is created on a RAID device that is composed of more than one physical disk, performance may be improved if read and write calls are issued in parallel.

To force DB2 UDB to use parallel I/O for a table space that only has one container, you use the DB2\_PARALLEL\_IO registry variable. This variable can be set to asterisk (\*), meaning every table space is to use parallel I/O, or it can be set to a list of table space IDs that are separated by commas. For example, this command would turn parallel I/O on for all table spaces:

```
db2set DB2_PARALLEL_IO=*
```

Where this command would only turn parallel I/O on for table spaces 1, 2, 4, and 8:

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

The DB2\_PARALLEL\_IO registry variable also affects tablespaces with more than one container defined. If this the registry variable is not set, the I/O parallelism used is equal to the number of containers in the tablespace. However, if this registry variable is set, the I/O parallelism used is equal to the result of (prefetch size / extent size). For example, if a tablespace has two containers and the prefetch size is four times the extent size and if the DB2\_PARALLEL\_IO registry variable is not set, a prefetch request for this table space will be broken into two requests (each request will be for two extents). Provided that the prefetchers are available to do work, two prefetchers can be working on these requests in parallel. In the case where the DB2\_PARALLEL\_IO registry variable is set, a prefetch request for this table space will be broken into four requests (one extent per request) with a possibility of four prefetchers servicing the requests in parallel.

In this example, if each of the two containers had a single disk dedicated to it, setting the DB2\_PARALLEL\_IO registry variable might result in contention on those disks since 2 prefetchers would be accessing each of the two disks at once. However, if each of the two containers was striped across multiple disks, setting the DB2\_PARALLEL\_IO registry variable would potentially allow access to four different disks at the same time.

## DB2\_STRIPED\_CONTAINERS

When creating a DMS table space, a one-page tag is stored at the beginning of each container used for identification purposes. The remaining pages are available for storage by DB2 and are grouped into extent-size blocks of data.

When using RAID devices for table space containers, it is suggested that the table space be created with an extent size that is equal to, or a multiple of, the RAID stripe size. However, because of this one-page container tag, the extents will not line up with the RAID stripes; this may cause I/O requests to access more physical disks than would be optimal. This can have a significant impact when the RAID devices are not cached, and do not have special sequential detection and prefetch mechanisms.

To eliminate this problem, the DB2\_STRIPED\_CONTAINERS registry variable can be used to tell DB2 UDB to use a full extent to store the identification tag in each container used. If this variable is set to ON, every table space created is to use a full extent for each container's tag. For example, this command would cause identification tags to be stored in one extent, rather than in one page:

```
db2set DB2_STRIPED_CONTAINERS=ON
```

### 2.1.13 A word about DB2EMPFA

As mentioned earlier, because SMS table spaces are managed by the file system, rather than the DB2 Database Manager, a size for each container used does not have to be specified. That's because the file system is responsible for allocation additional storage space as it is needed. By default, SMS table spaces are expanded one single page at a time. However, in certain work loads (for example, when doing a bulk insert) it might be desirable to have storage space allocated in extents rather than pages.

To force DB2 UDB to expand SMS table spaces one extent at a time, rather than one page at a time, you use the DB2EMPFA utility. The db2empfa tool is located in the bin subdirectory of the sqllib directory in which the DB2 UDB product is installed. Running it causes the multipage\_alloc database configuration parameter (which is a read-only configuration parameter) to be set to YES.

## 2.2 File protocols

In this section we describe some of the terminology and concepts that are related to file protocols, NAS and SAN.



## 2.2.1 Network file system protocols

The two most common file level protocols used to share files across networks are Network File System (NFS) for UNIX and Common Internet File System (CIFS) for Windows. Both are network based client/server protocols which enable hosts to share resources across a network using TCP/IP. Users manipulate shared files, directories and devices such as printers, as if they were locally on or attached to the user's own computer. The NAS devices described in the book are pre configured to support both NFS and CIFS.

### Network File System (NFS)

NFS servers make their file systems available to other systems in the network by exporting directories and files over the network. Once exported, an NFS client can then “mount” a remote file system from the exported directory location. NFS controls access by giving client-system level user authorization based on the assumption that a user who is authorized to the system must be trustworthy. Although this type of security is adequate for some environments, it is open to abuse by anyone who can access a UNIX system via the network. (To get around this, NFS can be made secure by using an isolated network in conjunction with VLANs to control what systems have access to across the network.)

For directory and file level security, NFS uses the UNIX concept of file permissions with User (the owner's ID), Group (a set of users sharing a common ID), and Other (meaning all other user IDs). For every NFS request, the IDs are verified against the UNIX file permissions. NFS is a stateless service. Therefore, any failure in the link will be transparent to both client and server. When the session is re-established the two can immediately continue to work together again. NFS handles file locking by providing an advisory lock to subsequent applications to inform them that the file is in use by another application. The ensuing applications can decide if they want to abide by the lock request or not. This has the advantage of allowing any UNIX application to access any file at any time, even if it is in use. The system relies on “good neighbor” responsibility which, though often convenient, clearly is not foolproof. This is avoided by using the optional Network Lock Manager (NLM). It provides file locking support to prevent multiple instances of open files.

### Common Internet File System (CIFS)

Another method used to share resources across a network uses CIFS, which is a protocol based on Microsoft's Server Message Block (SMB) protocol. Using CIFS, servers create file shares which are accessible by authorized clients. Clients subsequently connect to the server's shares to gain access to the resource. Security is controlled at both the user and share level. Client authentication information is sent to the server before the server will grant access. CIFS uses access control lists that are associated with the shares,

directories, and files, and authentication is required for access. A session in CIFS is oriented and stateful. This means that both client and server share a history of what is happening during a session, and they are aware of the activities occurring. If there is a problem, and the session has to be re-initiated, a new authentication process must be completed.

### 2.2.2 File I/O

One of the key differences of a NAS device, compared to direct access storage (DAS) is that all I/O operations use file level I/O protocols. File I/O is a high level type of request that, in essence, specifies only the file to be accessed, but does not directly address the storage device. This is done later by other operating system functions in the remote NAS device. A File I/O request specifies the file and the offset into the file. For instance, the I/O may specify “Go to byte ‘1000’ in the file (as if the file was a set of contiguous bytes), and read the next 256 bytes beginning at that position”.

Unlike Block I/O, there is no awareness of a disk volume or disk sectors in a File I/O request. Inside the NAS device, the operating system keeps track of where files are located on disk. The OS issues a Block I/O request to the disks to fulfill the File I/O read and write requests it receives. Network access methods, NFS and CIFS, can only handle File I/O requests to the remote file system. I/O requests are packaged by the node initiating the I/O request into packets to move across the network. The remote NAS file system converts the request to Block I/O and reads or writes the data to the NAS disk storage. To return data to the requesting client application, the NAS software re-packages the data in TCP/IP protocols to move it back across the network. This is illustrated in Figure 2-2.

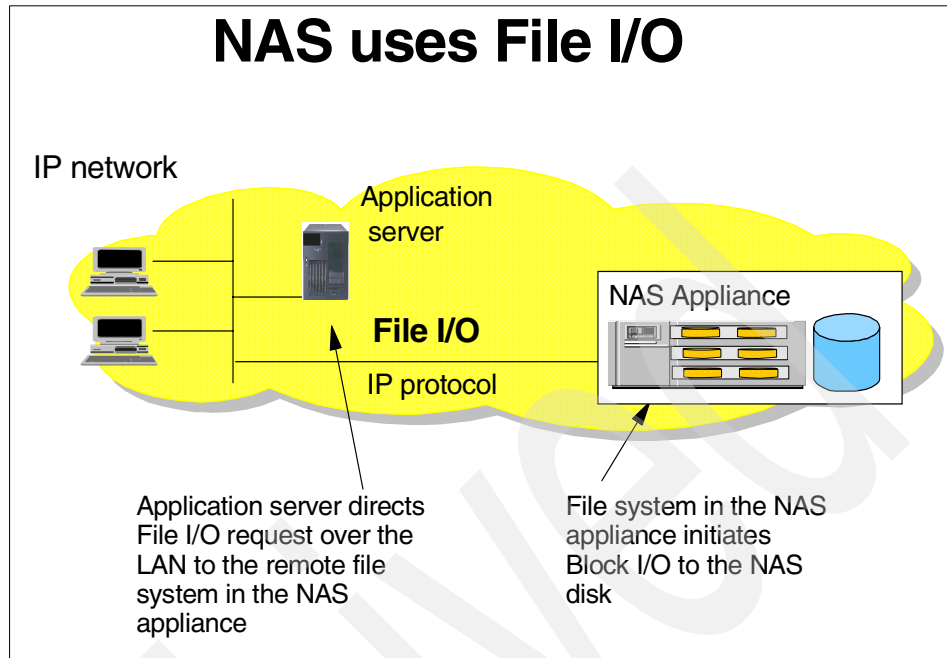


Figure 2-3 NAS devices use file I/O

### 2.2.3 Local Area Networks (LAN)

A Local Area Network (LAN) is simply the connection of two or more computers (nodes) to facilitate data and resource sharing. They proliferated from the mid-1980s to address the problem of “islands of information” which occurred with standalone computers within departments and enterprises.

LANs typically reside in a single or multiple buildings confined to a limited geographic area which is spanned by connecting two or more LANs together to form a Wide Area Network (WAN). The design of LANs are based typically on open systems networking concepts. These concepts are described in the network model of the Open Systems Interconnection (OSI) standards of the International Standards Organization (ISO). LAN types are defined by their topology, which is simply how nodes on the network are physically connected together. A LAN may rely on a single topology throughout the entire network but typically has a combination of topologies connected using additional hardware.

## 2.3 Storage Area Network terminology and concepts

The server infrastructure is the underlying reason for all SAN solutions. This infrastructure includes a mix of server platforms such as Windows NT, UNIX and OS/390. With initiatives such as server consolidation and e-business, the need for SAN will increase.

### 2.3.1 SAN storage

The storage infrastructure is the foundation on which information relies and therefore must support a company's business objectives and business model. In this environment, simply deploying more and faster storage devices is not enough; a new kind of infrastructure is needed, one that provides more enhanced network availability, data accessibility, and system manageability than is provided by today's infrastructure. The SAN meets this challenge. The SAN liberates the storage device, so it is not on a particular server bus, and attaches it directly to the network. In other words, storage is externalized and functionally distributed across the organization. The SAN also enables the centralizing of storage devices and the clustering of servers, which makes for easier and less expensive administration.

### 2.3.2 SAN fabric

The first element that must be considered in any SAN implementation is the connectivity of storage and server components using technologies such as Fibre Channel. SANs, like LANs, interconnect the storage interfaces together into many network configurations and across long distances. Much of the terminology used for SAN has its origin in IP network terminology. The hardware that enables workstations and servers to work with storage devices in a SAN is referred to as a "fabric". The SAN fabric gives any server the ability to connect to any storage device through the use of Fibre Channel switching technology.

### 2.3.3 SAN applications

Storage Area Networks (SANs) enable a number of applications that provide enhanced performance, manageability, and scalability to IT infrastructures. These applications are being driven by parts technology capabilities, and as technology matures over time, we are likely to see more and more applications in the future. A few applications are listed below:

## **Shared repository and data sharing**

SANs enable storage to be externalized from the server and centralized, and in so doing, allow data to be shared among multiple host servers without impacting system performance. The term data sharing describes the access of common data for processing by multiple computer platforms or servers. Data sharing can be between platforms that are similar or different; this is also referred to as homogeneous and heterogeneous data sharing.

## **Data-copy sharing**

Data-copy sharing allows different platforms to access the same data by sending a copy of the data from one platform to the other. There are two approaches to data-copy sharing between platforms: flat file transfer and piping.

## **Data vaulting and data backup**

In most present day scenarios of data vaulting and data backups to near-line or off-line storage, the primary network, LAN or WAN, is the medium used for transferring both the server, file and database server, or end-user client data to the storage media. SANs enable data vaulting and data backup operations on servers to be faster and independent of the primary network, which has led to the delivery of new data movement applications like LAN-less backup and server-free backup.

## **Clustering**

Clustering is usually thought of as a server process providing failover to a redundant server, or as scalable processing using multiple servers in parallel. In a cluster environment, SAN provides the data pipe, allowing storage to be shared.

## **Data protection and disaster Recovery**

The highest level of application availability requires avoiding traditional recovery techniques, such as recovery from tape. Instead, new techniques that duplicate systems and data must be architected so that, upon the event of a failure, another system is ready to go. Techniques to duplicate the data portion include remote copy and warm standby techniques. Data protection in environments with the highest level of availability is best achieved by creating second redundant copies of the data by storage mirroring, remote cluster storage, Remote Copy and Extended Remote Copy (XRC), Concurrent Copy, and other High Availability (HA) data protection solutions. These are then used for disaster recovery situations.

SAN any-to-any connectivity enables these redundant data/storage solutions to be dynamic and not impact the primary network and servers, including serialization and coherency control. Subsystem local copy services, such as SnapShot Copy or Flashcopy, assist in creating copies in high availability environments and for traditional backup and therefore are not directly applicable to disaster recovery or part of SAN.

## DB2 UDB backup and recovery

In this chapter we discuss the general and new features of DB2 UDB backup and recovery. In particular, we explain the DB2 UDB new feature, called suspend I/O write for split mirror image created by ESS Copy Services, and how to use mirror images in DB2 UDB.

The following topics are covered:

- ▶ Backup and recovery
  - Logging
  - Backup feature
  - Recovery feature
  - Incremental backup and recovery
- ▶ Split mirror support
  - High availability requirements
  - Suspending database
  - Initializing database

## 3.1 Backup and recovery

DB2 UDB has its database level backup and recovery utilities, which are accessed by command line interface (CLI) or graphical user interface (GUI), and which record all the information of an executed backup and recovery event on recovery history file.

Only one backup command is used to backup the entire database. This backup file includes database system files, data files, log files, control information, and so on. Database recovery is also performed by one restore command or with the rollforward command. This integrated feature makes it easy for you to manage database backup and recovery.

In this section we explain the database backup and recovery feature, and we also discuss incremental backup, dual logging, and close active log, which are new features in UDB V7.2 PixPak3.

### 3.1.1 Logging

When you make a backup plan, you should consider using the database logging mode. The database records every change of rows in tables, and each database object. These logs or log files are used to recover from applications or system errors.

#### **Circular logging**

By default, the log files are used in a circular fashion. When the last log file is full and the first log has no active transaction, the first log file is reused. With circular logging, you can only do offline database backup. Users cannot use the database while an offline backup is taking place. Tablespace level backup is not supported, either. This backup is only used for the version recovery, that is, recovering a database to the last backup time. Log files are not applied for version recovery.

#### **Archive logging**

The archive mode supports online backup and database recovery using log files called roll forward recovery. The logging mode can be changed from circular to archive by setting *logretain* or *userexit* to on. When roll forward recovery is enabled, log files are kept and not reused, so they can be applied when performing roll forward recovery. You can do online database backup where users can remain connected to the database, while the backup is taking place. You can also do a tablespace backup either offline or online.



There are two type of log files:

- ▶ Active log files which contain current transaction data needed to do rollback during online transaction or to do rollback or commit during crash recovery
- ▶ Archive log files which contain committed data

### Dual logging

DB2 UDB V7.2 PixPak3 adds the dual logging feature to DB2 UDB for UNIX to prevent log failures, which could be caused by administrators, who accidentally delete corrupted active logs or log files at the hardware level. Dual logging can be enabled by setting the DB2 registry *DB2\_NEWLOGPATH2* to *1*. The path is generated by appending a “2” to the current value of the logpath database configuration parameter, after db2stop and db2start. For example, the second log path is */db2/db2\_log2/*, if the logpath is */db2/db2\_log*. Future releases will allow an explicit logpath name.

If an error happens to be on either path, that log path will no longer be used and a message will be written to the db2diag.log. This message appears continuously until the database that tries to access the next log is aware that the errored path is fixed. But you can still use database with the other log path.

The command is

- ▶ **db2set DB2\_NEWLOGPATH=1**

If you don't use this feature, you must mirror the filesystem or raw device where the active logs are located. If active log files are corrupted, you can not restart database.

## 3.1.2 Backup feature

You can use any number of backup features, but you should decide the backup strategy considering how to recover the database. The appropriate backup strategy will protect a company from losing data assets.

All command examples are executed in the Windows 2000 platform, but should be applicable to all environments.

### Offline backup

Offline backup involves shutting the database down before you start the backup and restarting the database after the backup is complete. Offline backups are relatively simple to administer. However, they suffer from the obvious significant disadvantage that neither users nor batch processes can access the database while the backup is taking place. You need to schedule sufficient time to perform the backup to ensure that the periods when the database will be unavailable are acceptable to your users.

Circular logging of a database only can be backed up using an offline database backup. A database and tablespace level backup is allowed in archive logging of a database. This command should be run after all applications connected to the database are disconnected. See Example 3-1.

*Example 3-1 Offline backup in CLI*

---

```
C:\PROGRA~1\SQLLIB\BIN>db2 list applications
```

Auth Id	Application Name	Appl. Handle	Application Id	DB Name	# of Agents
RMRES12	db2bp.exe	14	*LOCAL.DB2.011219181316	SAMPLE	1

---

```
C:\PROGRA~1\SQLLIB\BIN>db2 force application (14)
DB20000I The FORCE APPLICATION command completed successfully.
DB21024I This command is asynchronous and may not be effective immediately.
```

```
C:\PROGRA~1\SQLLIB\BIN>db2 backup database sample to C:\DB2_BACKUP
Backup successful. The timestamp for this backup image is : 20011219102236
```

---

### Online backup

The archive logging mode allows backups to be performed while the database is started and in use. Clearly, if a database is being backed up while users are updating it, it is likely that the data backed up will be inconsistent. The database uses log files during the recovery process to recover the database to a fully consistent state. This approach requires that you retain the recovery log files and indicate to the database when you are about to start the backup and when you have completed the backup. See Example 3-2.

*Example 3-2 Online backup in CLI*

---

```
C:\PROGRA~1\SQLLIB\BIN>db2 list applications
```

Auth Id	Application Name	Appl. Handle	Application Id	DB Name	# of Agents
RMRES12	db2bp.exe	5	*LOCAL.DB2.011219063903	SAMPLE	1

---

```
C:\PROGRA~1\SQLLIB\BIN>db2 backup database sample online to C:\DB2_BACKUP
Backup successful. The timestamp for this backup image is : 20011219100457
```

---

A database level and tablespace level online backup is supported. But you must backup recovery log files to recover the database using an online backup image. Without log files which were active logs during backup, you cannot recover to a database consistent state. You cannot take the log files backup right after the database backup, because these log files are in an active state.

Now a new closing active log file feature is introduced in DB2 UDB V7.2 PixPak3 to backup active log files after an online database backup finishes safely. This command closes and archives active logs and allows users to acquire a complete set of log files up to the point in time when the command is executed. This command can be used only in the archive logging mode. You should execute this command without a database connection. See Example 3-3.

*Example 3-3 Archive active log*

---

```
C:\PROGRA~1\SQLLIB\BIN>db2 archive log for database sample
DB20000I The ARCHIVE LOG command completed successfully.
```

---

After issuing the archive log command, you can backup archive log files and keep this backup with the database backup, which can be used for restoring a database to a point in time when you took the online backup, even though you lose all the archive logs in the production server.

If other applications have transactions in progress, a slight performance degradation will be noticed.

## Database backup

Just one command is needed to take a full database backup. The backup file which is created by the database backup command includes history files, system files, data files, log files, control files, and so on. You do not need to copy all components with each command. You do not need to change the database backup command whenever the database layout is changed. All information is stored in the backup file.

Database level backup can be performed both in circular mode or archive mode. So you can take offline and online backups. After an online database backup, you should take an active log files backup. You can use this database backup file to recover specific tablespaces if this is an online backup file. See Example 3-4.

*Example 3-4 Database level online in CLI*

---

```
C:\PROGRA~1\SQLLIB\BIN>db2 backup database sample online to C:\DB2_BACKUP
Backup successful. The timestamp for this backup image is : 20011219101328
```

```
C:\PROGRA~1\SQLLIB\BIN>db2 archive log for database sample
DB20000I The ARCHIVE LOG command completed successfully.
```

---

The simplest approach to database backup is to perform only full, offline backups at regular intervals. This approach is relatively easy to administer, and recovery is relatively straightforward. However, it may not be practical to take databases offline for the period of time that is necessary to perform full backups at the frequency you need. You may have to adopt a more flexible approach. See Example 3-5.

*Example 3-5 Database offline backup in CLI*

```
C:\PROGRA~1\SQLLIB\BIN>db2 list applications
```

Auth Id	Application Name	Appl. Handle	Application Id	DB Name	# of Agents
RMRES12	db2bp.exe	14	*LOCAL.DB2.011219181316	SAMPLE	1

```
C:\PROGRA~1\SQLLIB\BIN>db2 force application (14)
DB20000I The FORCE APPLICATION command completed successfully.
DB21024I This command is asynchronous and may not be effective immediately.

C:\PROGRA~1\SQLLIB\BIN>db2 backup database sample to C:\DB2_BACKUP
Backup successful. The timestamp for this backup image is : 20011219102236
```

---

For high availability, you should perform an online database backup. If the database is big, you should consider a database backup and tablespace backup combination. You may take a full database backup on the weekend, and you may take a tablespace backup during week. This could be used to release a database backup pending state after a logging mode change.

### Tablespace backup

A tablespace level backup is supported when the database is in archive logging mode. Tablespace backup involves backing up the history file, system tablespace, and user tablespaces. You can take only one tablespace or several tablespaces together in one backup file. If the size of one backup file exceeds the operating system limit, the database will create another backup file, which is sequenced automatically.

For the recovery point of view, you should consider backing up the tablespaces all together, of which tables have referential integrity (RI) or triggers with each other. The related tablespaces at the application level should be backed up together even when they don't have RI and triggers for the data integrity. For example, transactions change more than one table. By keeping related sets of data together, you can recover to a point where all of the data is consistent.

The tablespace backup file also stores tablespace layout information automatically when a tablespace backup is taken. You do not have to change the backup command when you add tablespace containers. But don't forget to backup active log files after backing up tablespaces. See Example 3-6.

*Example 3-6 Tablespace level online backup in CLI*

---

```
C:\PROGRA~1\SQLLIB\BIN>db2 backup database sample tablespace ( data_ts1,  
index_ts1 ) online to C:\DB2_BACKUP  
Backup successful. The timestamp for this backup image is : 20011219141309
```

```
C:\PROGRA~1\SQLLIB\BIN>db2 archive log for database sample  
DB20000I The ARCHIVE LOG command completed successfully.
```

---

This could be used to release the tablespace backup pending state after a point-in-time tablespace recovery or load operation in archive logging mode without the nonrecoverable option.

## **Verification of DB2 backup files**

Backup images are created at the target location that you have the option to specify when you invoke the backup utility. This location can be a directory, a device, a Tivoli Storage Manager (TSM) server or another vendor's server.

On UNIX based systems, file names for backup images created on disk consist of a concatenation of several elements, separated by periods:

*database\_alias.backup\_type.instance\_name.node\_number.catalog\_node\_number.timestamp.sequence\_number*

An example of backup file name in UNIX platform is:

*SAMPLE.0.db2inst1.NODE0000.CATN0000.20011218104751.001*

On other platforms, a four-level subdirectory tree is used:

*database\_alias.database\_type\instance\_name\node\_number\catalog\_node\_number\date\time.sequence\_number*

An example of backup file name in Windows 2000 is:

*SAMPLE.0\DB2\NODE0000\CATN0000\20011219\101328.001*

Database backup could be corrupted even if the backup is completed successfully. These backup files cannot be restored. To prevent this problem at recovery time, you need to verify database backup files. The recovery history file gives you where the database backup is, when the backup is taken and how the backup file is created.

### **Recovery history file**

The DB2 UDB recovery history file (db2rhist.asc) contains logged events from different administrative events like backup, roll forward, load, drop table, reorganize, and so on. This information is automatically updated whenever these events happen.

The success of backups can be verified by listing the backup information using the **list history backup** command. Example 3-7 shows an example.

#### *Example 3-7 Backup history*

```
C:\PROGRA~1\SQLLIB\BIN>db2 list history backup all for sample
```

#### List History File for sample

Number of matching file entries = 11

Op	Obj	Timestamp+Sequence	Type	Dev	Earliest Log	Current Log	Backup ID
----	-----	--------------------	------	-----	--------------	-------------	-----------

B	D	20011219101328001	N	D	S0000010.LOG	S0000011.LOG	
---	---	-------------------	---	---	--------------	--------------	--

Contains 2 tablespace(s):

00001 SYSCATSPACE

00002 USERSPACE1

Comment: DB2 BACKUP SAMPLE ONLINE

Start Time: 20011219101328

End Time: 20011219101348

00009 Location: C:\DB2\_BACKUP\SAMPLE.0\DB2\NODE0000\CATN0000\20011219

Op	Obj	Timestamp+Sequence	Type	Dev	Earliest Log	Current Log	Backup ID
----	-----	--------------------	------	-----	--------------	-------------	-----------

B	D	20011219102236001	F	D	S0000013.LOG	S0000013.LOG	
---	---	-------------------	---	---	--------------	--------------	--

Contains 2 tablespace(s):

00001 SYSCATSPACE

00002 USERSPACE1

Comment: DB2 BACKUP SAMPLE OFFLINE

Start Time: 20011219102236

End Time: 20011219102256

00010 Location: C:\DB2\_BACKUP\SAMPLE.0\DB2\NODE0000\CATN0000\20011219

These lines of the backup history show the following:

- ▶ Operation (Op): B means backup
- ▶ Object (Obj): D means database backup, P means tablespace backup

- Backup type (Type): N means online backup, F means offline backup

The size of db2rhist.asc file is controlled by *rec\_his\_retentin* database configuration parameter or **prune history** command. For more detailed information, please refer to *Data Recovery and High Availability Guide and Reference Data-RCVR-00, DB2 Command Reference, SC09-2951*.

### db2ckbkp

The most important and, of course, the most reliable way is to see whether a backup is recoverable — to actually restore it. At least once, a typical backup should be restored to test if the backup and the restore process itself is valid. There is no way to restore an unsuccessful, incomplete or even missing backup.

The db2ckbkp checks the integrity of the backup image and determines whether or not it can be restored. Some or all parts of the backup can be checked. The backup image must reside physically on the disk. See Example 3-8.

*Example 3-8 db2ckbkp command in CLI*

---

```
C:\PROGRA~1\SQLLIB\BIN>db2ckbkp C:\DB2_BACKUP\SAMPLE.0\DB2\NODE0000\CATN0000\2011219\101328.001
```

```
[1] Buffers processed: ####
```

```
Image Verification Complete - successful.
```

---

## 3.1.3 Recovery feature

A database can become unexpectedly unusable because of user error, software failure, or hardware failure, and different recovery strategies are required to solve the different database failure situations.

In this section, we look at different methods of database recovery. Restoring a database would require us to select an image taken from a backup, using a timestamp to identify an image.

There are two ways of selecting an image:

- The **list history backup** command in CLI
- The database restore panel in Control center

All command examples are executed from a CLI in Windows 2000 platform, but should be applicable to all environments.

Here we show you backup history information and start each recovery from it.

## Crash recovery

When a database crashes while transactions are running because of software failure, hardware failure, or user mistake, the database is in an inconsistent state. The database manager rolls back incomplete transactions, and complete committed transactions that were still in memory when the crash occurred, to move back to the consistent state at the first database connection time. This process is crash recovery.

During crash recovery, the database manager uses an active log file and a log control head file (sqlogctl.lfh). Keeping these two files safe is very important for the database startup whenever a database crashes. When the log control head file is corrupted, there are two ways to recover database. You should use db2dart utility to export data from containers directly or you should contact your IBM service representative.

If a tablespace is corrupted, but necessary for crash recovery, the restart operation succeeds in the archive logging mode. The corrupted tablespaces are in the offline state. You should fix the damaged containers or restore the tablespace with the roll forward operation to use that tablespace again.

If the database is in circular logging mode, the restart operation will fail because a tablespace is damaged and needed for crash recovery. The damaged tablespace list is recorded in the db2diag.log file. You can restart the database again with the following command:

```
db2 restart database database_alias drop pending tablespaces (
tablespace_name )
```

But you should drop this tablespace after the database is restarted. This tablespace cannot be used again without a database restore.

## Version recovery

Databases that are not enabled for archive logging can only do version recovery. They are said to be non-recoverable databases. Version recovery restores a previous version of the database using an image created by a offline backup. No log files are applied, and any transactions committed after the backup are lost. All users must disconnect from the database for version recovery.



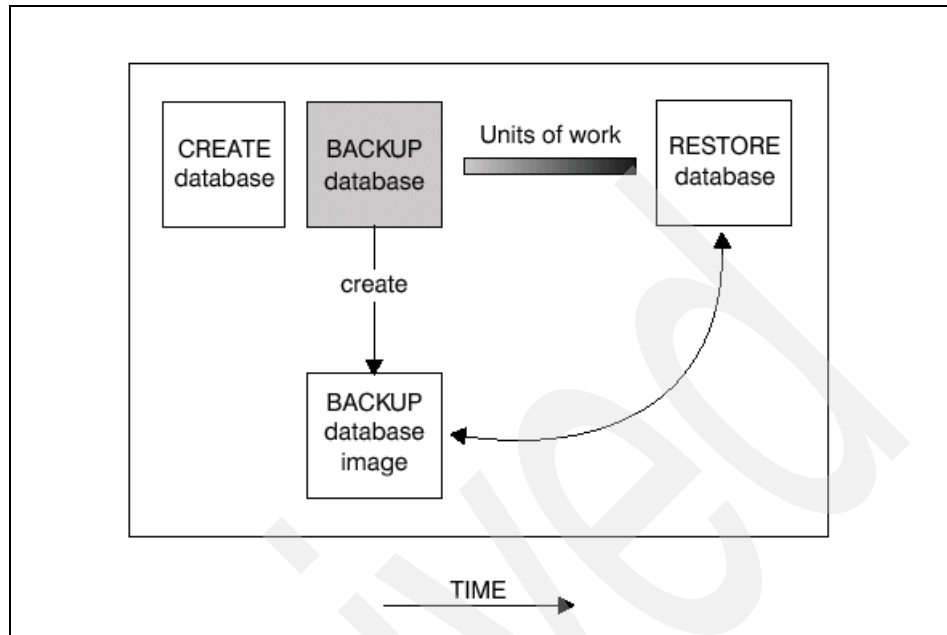


Figure 3-1 Recover to the last backup time

If a database is in circular logging mode, you should consider a backup cycle. More frequent backups will lose less data. Figure 3-1 shows that the units of work already performed after the last backup will be lost.

When a database is corrupted, you can check the latest backup information using the **list history backup** command and restore the database backup using **restore** command. Example 3-9 shows the history backup list and version recovery.

*Example 3-9 Version recovery*

```
C:\PROGRA~1\SQLLIB\BIN>db2 list history backup all for sample
```

List History File for sample

Number of matching file entries = 1

Op	Obj	Timestamp+Sequence	Type	Dev	Earliest Log	Current Log	Backup ID
B	D	20011220233929001	F	D	S0000000.LOG	S0000000.LOG	

Contains 2 tablespace(s):

```
00001 SYSCATSPACE
00002 USERSPACE1
```

```
-----
Comment: DB2 BACKUP SAMPLE OFFLINE
Start Time: 20011220233929
End Time: 20011220233950
-----
```

```
00001 Location: C:\DB2_BACKUP\SAMPLE.0\DB2\NODE0000\CATN0000\20011220
```

```
C:\PROGRA~1\SQLLIB\BIN>db2 restore database sample from C:\DB2_BACKUP taken at
20011220233929
```

```
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.
```

## Database roll forward recovery

Recoverable databases are databases that are enabled to do roll forward recovery. With recoverable databases, you can restore the database and apply the logs to the point of failure. No transactions are lost.

Figure 3-2 shows the units of work performed after a last backup is reapplied; the restored database is using *n* archived log files and 1 active log file. A successful database restore put the database in roll forward pending state when this backup was taken online.

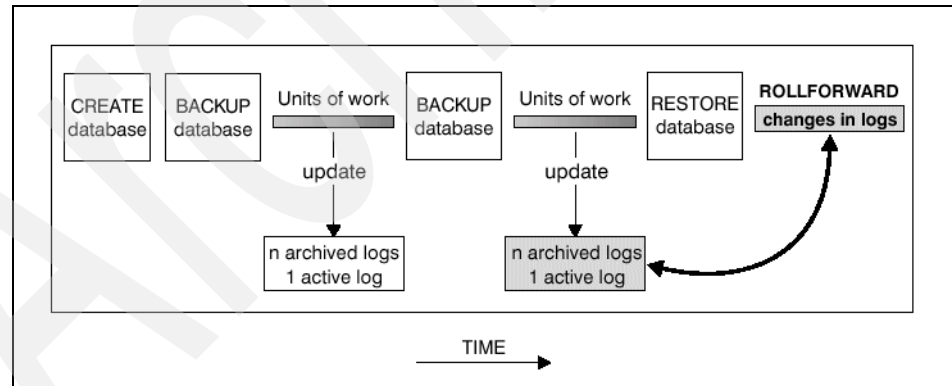


Figure 3-2 Recover database to the point of failure

The database is restored. See Example 3-10.

### Example 3-10 Restore database

```
C:\PROGRA~1\SQLLIB\BIN>db2 restore database sample from C:\DB2_BACKUP taken at
20011219101252
```

```
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 **Rollforward** command is used to release the pending state. See Example 3-11.

*Example 3-11 Roll forward the database to the point of failure*

---

```
C:\PROGRA~1\SQLLIB\BIN>db2 rollforward database sample to end of logs and
complete
```

```
Rollforward Status

Input database alias           = sample
Number of nodes have returned status = 1

Node number                   = 0
Rollforward status            = not pending
Next log file to be read      =
Log files processed            = S0000010.LOG - S0000016.LOG
Last committed transaction     = 2001-12-21-03.55.55.000000
```

```
DB20000I The ROLLFORWARD command completed successfully.
```

---

### Tablespace roll forward recovery

With recoverable databases, you have the option of recovering the whole database or only the tablespaces that need recovery. You can recover tablespaces either offline or online. You can use an image taken from either a previous tablespace backup or a database backup.

Figure 3-3 shows a corrupted tablespace is recovered from a tablespace backup. After a tablespace roll forward recovery, the recovered tablespace is in backup pending state unless the restored tablespaces was rolled forward to the end of logs. The tablespace is usable when you take a tablespace backup.

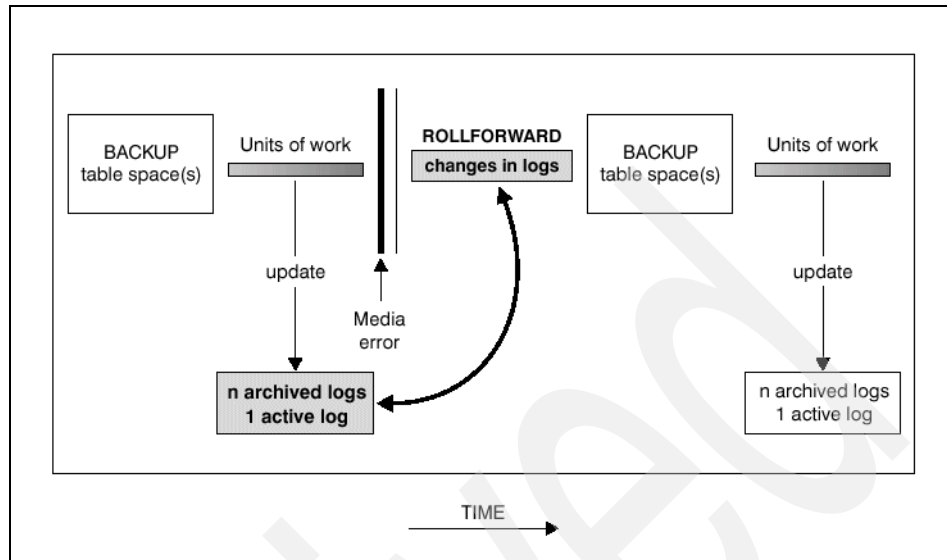


Figure 3-3 Recover tablespace to the point of failure

Example 3-12 show the tablespace recovery to the end of logs.

*Example 3-12 Recover tablespace to end of logs*

```
C:\PROGRA~1\SQLLIB\BIN>db2 restore database sample tablespace ( data_ts1,
index_ts1 ) from C:\DB2_BACKUP taken at 20011220222938
DB20000I The RESTORE DATABASE command completed successfully.
```

```
C:\PROGRA~1\SQLLIB\BIN>db2 rollforward database sample to end of logs and stop
```

#### Rollforward Status

Input database alias	= sample
Number of nodes have returned status	= 1
Node number	= 0
Rollforward status	= not pending
Next log file to be read	= S0000040.LOG
Log files processed	= -
Last committed transaction	= 2001-12-21-06.36.33.000000

```
DB20000I The ROLLFORWARD command completed successfully.
```

The corrupted tablespaces are rolled forward to the point of failure. This put the tablespace backup in a not pending state. See Example 3-13.

### *Example 3-13 Roll forward tablespace point of failure*

---

```
C:\PROGRA~1\SQLLIB\BIN>db2 rollforward database sample to
2001-12-21-06.37.28.000000 and complete tablespace ( data_ts1, index_ts1 )
Rollforward Status
```

```
Input database alias           = sample
Number of nodes have returned status = 1

Node number                    = 0
Rollforward status             = not pending
Next log file to be read      = S0000041.LOG
Log files processed            = -
Last committed transaction    = 2001-12-21-06.36.33.000000
```

```
DB20000I The ROLLFORWARD command completed successfully.
```

---

The tablespace point-in-time recovery puts the tablespace in backup pending state. Since the transaction logs between recovered time and current time become useless related to the tablespace, the tablespace cannot be rolled forward beyond this point if the database should be recovered again. To prevent this tablespace recovery failure, the tablespace is placed in backup pending state. You should take a tablespace backup to use the tablespace. See Example 3-14.

### *Example 3-14 Tablespace backup after recovery*

---

```
C:\PROGRA~1\SQLLIB\BIN>db2 backup database sample tablespace ( data_ts1,
index_ts1 ) to C:\DB2_BACKUP
```

```
Backup successful. The timestamp for this backup image is : 20011220233337
```

---

## **Recovering the history file**

The information of restoring or recovering database is automatically recorded in the history file. This is used for you to recover the database. The history file is always backed up during a database backup or a tablespace backup. You need to recover the history file to know backup, recovery or load information when the history file is corrupted or the entries are accidentally deleted using the **prune history** command.

You only can recover the history file from a database or a tablespace backup file.

### *Example 3-15 History file recovery*

---

```
C:\PROGRA~1\SQLLIB\BIN>db2 restore database sample history file from
C:\DB2_BACKUP taken at 20011220222938
DB20000I The RESTORE DATABASE command completed successfully.
```

---

### 3.1.4 Incremental backup and recovery

Database incremental backup and recovery feature is added in DB2 UDB V7.2. Database and tablespace level incremental backup is supported. This is helpful to recover a database without doing frequent full database backups or long log processing. Especially full database backups of large database, like a data warehouse, is not necessary when a small percentage of data in the data warehouse changes. This will save tape or disk space. Incremental backups are useful when saving space or when saving bandwidth when backing up over the network, but this can take more time to recover a database.

#### Incremental backup

An incremental backup is a backup image that contains only pages that have been updated since the previous backup was taken.

There are two types of incremental backup:

- **Incremental backup:** This is a cumulative backup; all the changes since the last full database backup are backed up.
- **Delta backup:** This is a non cumulative backup; all the changes since the most recent backup which includes full, incremental or delta backup are backed up.

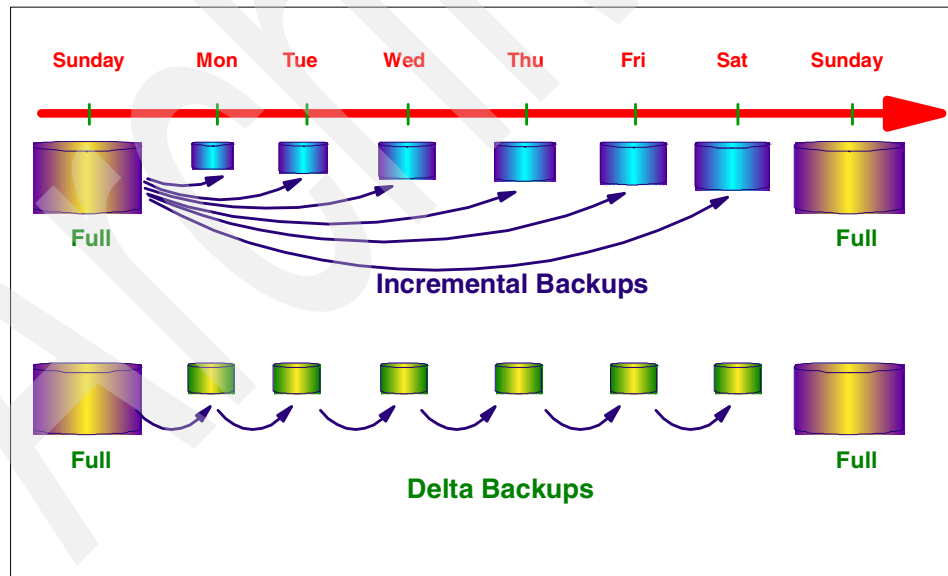


Figure 3-4 Incremental backup type

Figure 3-4 shows two different incremental backups. A combination of two different backups is more flexible in backup planning.

The new database configuration parameter *TRACKMOD* should be set to YES to use the incremental backup feature. See Example 3-16.

---

*Example 3-16 Incremental backup initiating*

---

```
C:\PROGRA~1\SQLLIB\BIN>db2 update db cfg for sample using TRACKMOD YES
DB20000I The UPDATE DATABASE CONFIGURATION command completed successfully.
DB21026I For most configuration parameters, all applications must disconnect
from this database before the changes become effective.
```

---

After changing the database parameter, taking a backup of a database or tablespace is a good point to start an incremental backup. A full database or a tablespace backup, before incremental backups are taken, is necessary to recover a database from the incremental backups. See Example 3-17.

---

*Example 3-17 Full backup for incremental backup consistency*

---

```
C:\PROGRA~1\SQLLIB\BIN>db2 backup database sample to C:\DB2_BACKUP

Backup successful. The timestamp for this backup image is : 20011221011020
```

---

Example 3-18 shows an incremental delta backup and incremental backup. The delta backup is performed with the **backup** command with the *incremental delta* option.

---

*Example 3-18 Incremental delta and incremental backup*

---

```
C:\PROGRA~1\SQLLIB\BIN>db2 backup database sample online incremental delta to
C:\DB2_BACKUP

Backup successful. The timestamp for this backup image is : 20011221154126

C:\PROGRA~1\SQLLIB\BIN>db2 backup database sample online incremental to
C:\DB2_BACKUP

Backup successful. The timestamp for this backup image is : 20011221154315
```

---

## Incremental recovery

Figure 3-5 shows what is needed to recover a database — for the full recovery, full database backup, last incremental backup, all delta backups after last incremental backup, and log files. The restore order is the last delta backup, full database restore, cumulative backup restore, delta backup restore again and then applying logs. Delta backup from Monday through Wednesday are not needed for this recovery. All the changed data is included in the incremental backup on Thursday.

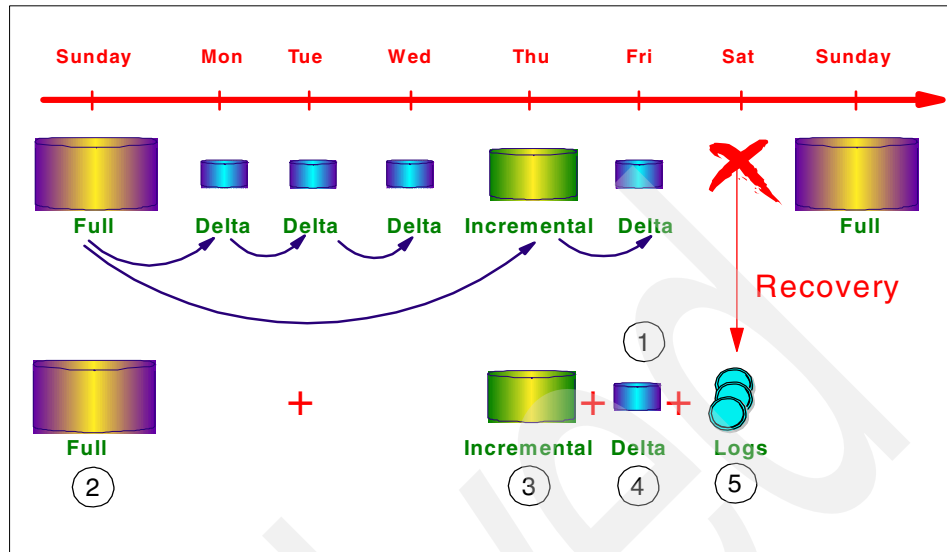


Figure 3-5 Combined incremental recovery

Each backup information is stored in the history file. Example 3-19 shows a backup history with the **list history backup** command. One full database backup, three delta backups, one incremental backup, and one delta backup are recorded in order. One full database backup taken at 20011221153756, one incremental backup taken at 20011221154315, and the last delta backup taken at 20011221154417 are used for the next recovery steps.

**Example 3-19 Incremental backup history**

```
C:\PROGRA~1\SQLLIB\IBM>db2 list history backup all for sample
```

List History File for sample

Number of matching file entries = 10

Op	Obj	Timestamp+Sequence	Type	Dev	Earliest Log	Current Log	Backup ID
B	D	20011221153756001	F	D	S0000001.LOG	S0000001.LOG	
B	D	20011221154126001	E	D	S0000002.LOG	S0000003.LOG	
B	D	20011221154200001	E	D	S0000003.LOG	S0000004.LOG	
B	D	20011221154231001	E	D	S0000004.LOG	S0000005.LOG	
B	D	20011221154315001	O	D	S0000006.LOG	S0000008.LOG	



These incremental backup history lines show the following:

- ▶ Operation (Op): B means backup
- ▶ Object (Obj): D means database backup
- ▶ Backup type (Type): F means offline backup, E means delta backup and O means incremental backup

Example 3-20 shows the manual steps of recovery with incremental backup. The last delta backup should be restored at the first time for the target image. Then the full database backup and the cumulative backup are restored in order. After that, the last delta backup is restored again and logs are applied to the restored database.

*Example 3-20 Manual incremental recovery*

---

```
C:\PROGRA~1\SQLLIB\BIN>db2 restore database sample incremental from
C:\DB2_BACKUP taken at 20011221154417
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.

C:\PROGRA~1\SQLLIB\BIN>db2 restore database sample incremental from
C:\DB2_BACKUP taken at 20011221153756
DB20000I The RESTORE DATABASE command completed successfully.

C:\PROGRA~1\SQLLIB\BIN>db2 restore database sample incremental from
C:\DB2_BACKUP taken at 20011221154315
DB20000I The RESTORE DATABASE command completed successfully.

C:\PROGRA~1\SQLLIB\BIN>db2 restore database sample incremental from
C:\DB2_BACKUP taken at 20011221154417
DB20000I The RESTORE DATABASE command completed successfully.

C:\PROGRA~1\SQLLIB\BIN>db2 rollforward database sample to end of logs and stop
```

Rollforward Status

Input database alias	= sample
Number of nodes have returned status	= 1
Node number	= 0
Rollforward status	= not pending
Next log file to be read	=
Log files processed	= S0000008.LOG - S0000012.LOG
Last committed transaction	= 2001-12-21-23.44.50.000000

DB20000I The ROLLFORWARD command completed successfully.

---

The **restore** command is used with the *incremental* option. The last delta backup is restored two times during the incremental restore operation. During the first restore, only the initial data is read from the images and the complete image is read and processed only during the second restore.

You can also use automatic recovery. The last image that you want to restore should be specified in the automatic recovery command. Each of the steps described above is performed automatically using database history file. So keeping the history file is important for the automatic incremental recovery. If the history file is not able to be used, you should use a manual incremental recovery.

Example 3-21 shows an automatic incremental recovery.

*Example 3-21 Automatic incremental recovery*

---

```
C:\PROGRA~1\SQLLIB\BIN>db2 restore database sample incremental automatic from  
C:\DB2_BACKUP taken at 20011221154417
```

```
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.
```

```
C:\PROGRA~1\SQLLIB\BIN>db2 rollforward database sample to end of logs and stop
```

Rollforward Status

Input database alias	= sample
Number of nodes have returned status	= 1
Node number	= 0
Rollforward status	= not pending
Next log file to be read	=
Log files processed	= S0000008.LOG - S0000012.LOG
Last committed transaction	= 2001-12-21-23.44.50.000000

```
DB20000I The ROLLFORWARD command completed successfully.
```

---

## 3.2 Split mirror support

The DB2 UDB suspend write I/O feature and the `db2inidb` tool were introduced with DB2 V7.1 Fixpak2 to provide faster database backup and recovery even without stopping the database service. In this section we describe why this feature is needed, and how the commands are used to make use of the split mirror features.

### 3.2.1 High availability requirements

Customers cannot afford performing offline or online database backups on a 1 TB live database. A backup always degrades the performance of the production system. Especially if the database is large or in a 24x7 hour environment, it is hard to find a time frame to take a backup that doesn't interfere with the normal operation. To free the production system from the overhead of a backup, a copy or mirror of the database is better for a backup, report or other purposes.

Application Service Providers like SAP require high availability backup, recovery and performance to deliver complex business solutions. Business to business solutions also require continuous service.

New technology in disk drive development has been improved to support high availability. Some intelligent storage servers such as the IBM Enterprise Storage Server (ESS) support the snapshot feature. Snapshot means that identical and independent copies of disk volumes can be established within those storage servers. These copies can normally be established in a very short time. (For example, 5 to 20 seconds depending on the device).

### 3.2.2 Suspending database

Establishing the snapshot doesn't guarantee database integrity unless the database is shut down. The suspended write I/O support is necessary to bring the database into a consistent state to establish the snapshot while the database is running. With the **write suspend** command, the database enters a well defined state where it can recover later using the `db2inidb` tool. A database connection is required for issuing the suspend command. See Example 3-22.

*Example 3-22 Write suspend*

---

```
C:\PROGRA~1\SQLLIB\BIN>db2 connect to sample
```

```
Database Connection Information
```

```
Database server      = DB2/NT 7.2.2
SQL authorization ID = DB2INST1
Local database alias = SAMPLE
```

```
C:\PROGRA~1\SQLLIB\BIN>db2 set write suspend for database
DB20000I The SET WRITE command completed successfully.
```

---

During the suspended period, tablespaces are in a write suspended state. Writing to the database is not allowed, but the database remains online and available for reads. You can see the state with **list tablespaces** command. See Example 3-23.

*Example 3-23 Write suspended tablespaces*

---

```
C:\PROGRA~1\SQLLIB\BIN>db2 list tablespaces
```

```
Tablespaces for Current Database

Tablespace ID          = 3
Name                   = DATA_TS1
Type                   = Database managed space
Contents               = Any data
State                  = 0x10000
Detailed explanation:
    Write Suspended

Tablespace ID          = 4
Name                   = INDEX_TS1
Type                   = Database managed space
Contents               = Any data
State                  = 0x10000
Detailed explanation:
    Write Suspended
```

---

A snapshot occurs when the database write I/O is suspended and the database is in a consistent state. This copy is established in a short time.

Updating or deleting transactions are suspended and will proceed as soon as the database is resumed again for writing, with **write resume** command. Any changes directed toward the data during the **write suspend** are then applied. See Example 3-24.

*Example 3-24 Write resume*

---

```
C:\PROGRA~1\SQLLIB\BIN>db2 set write resume for database
DB20000I The SET WRITE command completed successfully.

C:\PROGRA~1\SQLLIB\BIN>
```

---

This command can be used only for the database in the archive logging mode. But the snapshot feature of storage vendors can still be used for circular logging a database copy with a short shutdown time of the database.

### 3.2.3 Initializing database

The database manager cannot recognize the snapshot image itself. Database initialization is needed to use the snapshot image as a database for different purposes. The db2inidb tool is introduced to bring the snapshot image into a database and a running state. The db2inidb tool can initiate a database crash recovery or put a database in roll forward pending. It is used for the following three cases:

- ▶ **Snapshot:** The database initialized as a snapshot is a clone of the primary database. It is usually initialized on the secondary server, and used for an offline database backup, generating reports, testing or developing a database. This database is called a clone database.
- ▶ **Standby:** The standby database is used for high availability to keep applying archive log files of the primary database at the secondary server or remote site.
- ▶ **Mirror:** When the primary database is corrupted and needs a fast restore, the snapshot is initialized as a mirror, overwriting the corrupted primary database.

The db2inidb utility only needs the snapshot image. The normal database backup cannot be used for initialization. If the snapshot image is taken when the primary database is offline, the copied database can be used without the initializing database.

#### Snapshot

The snapshot image can be attached to a secondary server. The snapshot image should be cataloged as a database at the server and db2inst is started. See Example 3-25.

*Example 3-25 Cataloging database on the secondary server*

---

```
C:\PROGRA~1\SQLLIB\BIN>db2 catalog database sample on C:
DB20000I The CATALOG DATABASE command completed successfully.
DB21056W Directory changes may not be effective until the directory cache is
refreshed.
```

```
C:\PROGRA~1\SQLLIB\BIN>db2start
SQL1063N DB2START processing was successful.
```

```
C:\PROGRA~1\SQLLIB\BIN>db2 list database directory
```

```
System Database Directory
```

Number of entries in the directory = 1

Database 1 entry:

Database alias	= SAMPLE
Database name	= SAMPLE
Database drive	= C:\DB2
Database release level	= 9.00
Comment	=
Directory entry type	= Indirect
Catalog node number	= 0

---

The **db2inidb** command is required to make a clone database. See Example 3-26.

*Example 3-26 Initializing database as snapshot*

---

```
C:\PROGRA~1\SQLLIB\BIN>db2inidb sample as snapshot
```

Operation was successful.

---

This command initiates crash recovery, making the database consistent. A new log chain starts, and the database will not be able to roll forward through any of the logs from the original database. Transactions that were in process at the moment the **write suspend** was issued will be rolled back.

A clone database can be used for taking an offline backup, generating a test database, or generating reports.

## Standby

The snapshot image could be located in a remote site as well as in a local site using a remote copy feature like Peer-to-peer Remote Copy (PPRC). The snapshot image can be attached to a secondary or remote server. After cataloging the snapshot image as a database at the server, the image can be initialized as standby using **db2inidb** command.

This places the database in a roll forward pending state. Crash recovery is not performed, and the database remains inconsistent. See Example 3-27.

*Example 3-27 Initializing database as standby*

---

```
C:\PROGRA~1\SQLLIB\BIN>db2 catalog database sample on C:
DB20000I The CATALOG DATABASE command completed successfully.
DB21056W Directory changes may not be effective until the directory cache is
refreshed.
```

```
C:\PROGRA~1\SQLLIB\BIN>db2start
```

SQL1063N DB2START processing was successful.

C:\PROGRA~1\SQLLIB\BIN>db2inidb sample as standby

Operation was successful.

C:\PROGRA~1\SQLLIB\BIN>db2 connect to sample

SQL1117N A connection to or activation of database "SAMPLE" cannot be made because of ROLL-FORWARD PENDING. SQLSTATE=57019

---

The archive log files of the primary database can be applied to the standby database. These archive log files can be passed to the standby database using a userexit program. If the userexit program is located in the primary database, the *ARCHIVE\_PATH* should be defined as the log directory of the standby database. If the userexit program is located in the standby database, the *RETRIEVE\_PATH* is defined as the archive log directory of the primary database.

The standby database can be rolled forward continuously until roll forward pending state is released using **rollforward complete** command. See Example 3-28

*Example 3-28 Roll forward standby database*

---

C:\PROGRA~1\SQLLIB\BIN>db2 rollforward database sample to  
2001-12-21-23.42.00.000000

Rollforward Status

Input database alias	= sample
Number of nodes have returned status	= 1
Node number	= 0
Rollforward status	= DB working
Next log file to be read	= S0000004.LOG
Log files processed	= S0000001.LOG - S0000002.LOG
Last committed transaction	= 2001-12-21-23.41.49.000000

DB20000I The ROLLFORWARD command completed successfully.

C:\PROGRA~1\SQLLIB\BIN>db2 rollforward database sample to end of logs and  
complete

Rollforward Status

Input database alias	= sample
Number of nodes have returned status	= 1
Node number	= 0

Rollforward status	= not pending
Next log file to be read	=
Log files processed	= S0000001.LOG - S0000012.LOG
Last committed transaction	= 2001-12-21-23.44.50.000000

DB20000I The ROLLFORWARD command completed successfully.

---

Especially when disaster occurs in the data center where a production database exists, a standby database can take over the production database. Archive log files and active log files which were in the primary database and were not transferred to the standby database will be lost.

## Mirror

The snapshot image is used as backup image to restore over the primary database. The snapshot image can be copied back to the primary database volume using an OS copy command like **cp**, **tar**, and so on. Of course, the snapshot feature could be used to copy the snapshot image back to the primary database volume. But the database log files of the primary database should not be copied from a snapshot image. These log files are used for roll forward recovery of a mirror database.

The restored database can be initialized as a mirror using **db2inidb** command. See Example 3-29.

### *Example 3-29 Initializing mirror database*

---

```
C:\PROGRA~1\SQLLIB\BIN>db2 catalog database sample on C:
DB20000I The CATALOG DATABASE command completed successfully.
DB21056W Directory changes may not be effective until the directory cache is
refreshed.
```

```
C:\PROGRA~1\SQLLIB\BIN>db2start
SQL1063N DB2START processing was successful.
```

```
C:\PROGRA~1\SQLLIB\BIN>db2inidb sample as mirror
```

```
Operation was successful.
```

---

The database is placed in roll forward pending state, and the **write suspend** state is turned off. Crash recovery is not performed, and the database remains inconsistent. The log files of the primary database are applied to the mirror database during roll forward recovery and the database is consistent. See Example 3-30.

### *Example 3-30 Roll forward end of logs*

---

```
C:\PROGRA~1\SQLLIB\BIN>db2 rollforward database sample to end of logs and
complete
```



## Rollforward Status

Input database alias	= sample
Number of nodes have returned status	= 1
Node number	= 0
Rollforward status	= not pending
Next log file to be read	=
Log files processed	= S0000000.LOG - S0000000.LOG
Last committed transaction	= 2001-12-15-02.27.32.000000

DB20000I The ROLLFORWARD command completed successfully.

---

The snapshot image should not be initialized on another server to use for the mirror database. An initialized database cannot be used for a mirror database.



## Planning for ESS Copy Services

In this chapter we describe the planning for ESS Copy Services with these topics:

- ▶ Planning and considerations for using FlashCopy
- ▶ Planning and considerations for using Peer-to-peer Remote Copy (PPRC)

## 4.1 FlashCopy

Today, more than ever, organizations require their applications to be available 24 hours per day, seven days per week (24x7). They require high availability, minimal application downtime for maintenance, and the ability to perform data backups with the shortest possible application outage. The prime reason for data backup is to provide protection in case of source data loss due to disaster, hardware failure, software failure, or user errors.

Data copies can also be taken for the purposes of program testing, data mining by database query applications. However, normal copy operations take a long time requiring the prime application to be offline. In addition to the need for 24x7 data processing, it is also necessary to have an instant copy of the data.

FlashCopy allows you to move effectively towards such solutions.

In the following sections, we introduce the new ESS function of FlashCopy.

### 4.1.1 Overview

FlashCopy provides an instant or point-in-time copy of an ESS logical volume. The point-in-time copy functions give you an instantaneous copy, or “view”, of what the original data looked like at a specific point-in-time. This is known as the T0 (time-zero) copy.

When a FlashCopy is invoked, the command returns to the operating system as soon as the FlashCopy pair has been established and the necessary control bitmaps have been created. This process takes only a few seconds to complete. Thereafter, you have access to a T0 copy of the source volume. As soon as the pair has been established, you can read and write to both the source and the target volumes.

The point-in-time copy created by FlashCopy is typically used where you need a copy of production data to be produced with minimal application downtime. It can be used for online backup, testing of new applications, or for creating a database for data-mining purposes. The copy looks exactly like the original source volume and is an instantly available, binary copy. Figure 4-1 illustrates FlashCopy concepts.

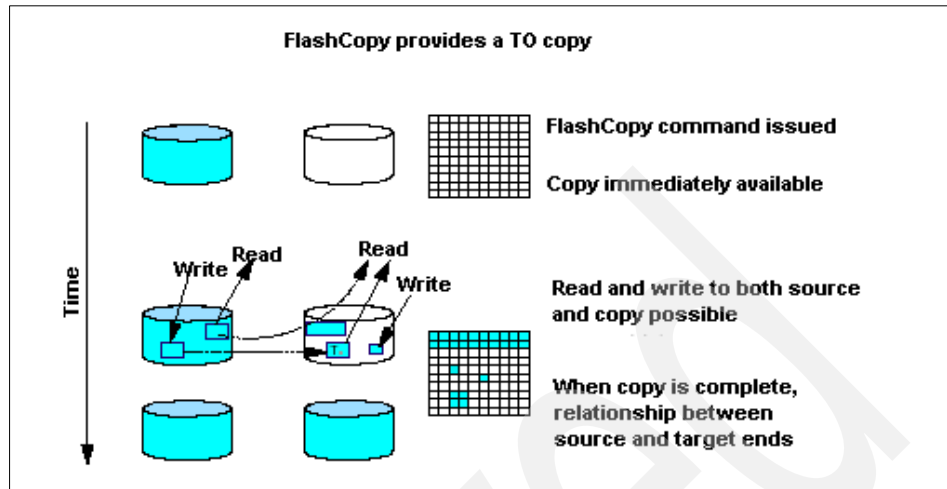


Figure 4-1 FlashCopy overview

**Attention:** FlashCopy is possible only between disk volumes. It also requires a target volume to be defined within the same Logical Subsystem (LSS) as the source volume.

A source volume and the target can be involved in only one FlashCopy relationship at a time. When you set up the copy, a relationship is established between the source and the target volume and a bitmap of the source volume is created.

Once this relationship is established and the bitmap created, the target volume copy can be accessed as though all the data had been physically copied. While a relationship between source and target volumes exists, a background task copies the tracks from the source to the target. The relationship ends when the physical background copy task has completed.

You can suppress the background copy task using the Do not perform background copy (NOCOPY) option. This may be useful if you need the copy only for a short time, such as making a backup to tape. If you start a FlashCopy with the Do not perform background copy option, you must withdraw the pair (a function you can select) to end the relationship between source and target.

At the time when FlashCopy is started, the target volume is basically empty. The background copy task copies data from the source to the target. The FlashCopy bitmap keeps track of which data has been copied from source to target. If an application wants to read some data from the target that has not yet been copied

to the target, the data is read from the source; otherwise, the read is satisfied from the target volume. When the bitmap is updated for a particular piece of data, it signifies that source data has been copied to the target and updated on the source. Further updates to the same area are ignored by FlashCopy. This is the essence of the T0 point-in-time copy mechanism.

Before an application can update a track on the source that has not yet been copied, the track is copied to the target volume. Reads that are subsequently directed to this track on the target volume are now satisfied from the target volume instead of the source volume. After some time, all tracks will have been copied to the target volume, and the FlashCopy relationship will end.

You cannot create a FlashCopy on one type of operating system and make it available to a different operating system. You can make the target available to another host running the same type of operating system.

### 4.1.2 Planning for FlashCopy on ESS

Because FlashCopy invariably will be used on production systems, you should carefully plan the setup of your environment and test it thoroughly. This is an important step to minimize the possibility of error and potential rework.

#### Hardware and software requirements

If you want to use FlashCopy, you need to comply with the following prerequisites:

- ▶ Have a FlashCopy feature purchased and enabled on your Enterprise Storage Server (ESS) by the Customer Engineer (CE). The feature code is dependent on the total disk capacity of your ESS, rather than on the capacity of the volumes that will use FlashCopy.
- ▶ On the server that will have the FlashCopy target volumes attached, you need to have enough SCSI target IDs and/or SCSI or Fibre Channel LUNs available (not occupied by volumes). The ESS can have up to 15 SCSI target IDs each with up to 64 LUNs on one SCSI channel and up to 4095 LUNs on a Fibre Channel port.
- ▶ You need TCP/IP connectivity between the ESS and the host system that will initiate FlashCopy (usually this is the system that will access the FlashCopy target) in order to use Copy Services Command Line Interface (CLI). You can achieve that by connecting the ESS to the company intranet. You have to install the CLI on the host that will be using it.
- ▶ If you have Independent Software Vendor (ISV) software installed that writes directly to disk, you need to contact the ISV regarding their support for ESS Copy Services.

- ▶ Review your volume manager software considerations for FlashCopy:
  - AIX LVM
  - Veritas VxVM
  - HP SAM
  - Sun Solaris DiskSuite
- ▶ The IBM Subsystem Device Driver (which has superseded Data Path Optimizer) fully supports HACMP clusters in both concurrent and non-concurrent access modes. Subsystem Device Driver (SDD) works with FlashCopy volumes.

## Configuration planning

The most important consideration is to have an available volume (LUN) in the logical subsystem (LSS) where the source volume resides. The target LUN has to be of the same size as the source or bigger. The space for target data has to be available even if only the *Do not perform background copy* option will be used.

## Resource planning

When planning your ESS volume layout, it is important to consider the capacity you may need for FlashCopy targets. Bear in mind that the disk space you need is real disk space. You must also consider that a FlashCopy target is restricted to the same LSS as its source volume. So, when you allocate additional storage for FlashCopy targets, consider how much space in each LSS you need to leave unallocated for them.

You cannot initiate a FlashCopy session on a source and target that are already in a FlashCopy session. You need to wait for the FlashCopy task to complete, or until you can withdraw the pair manually. If you have used the *Do not perform background copy* option, you always need to withdraw the pair.

## Data consistency considerations

It is very important to verify that the copy of the data you will be using is fully consistent by using a proper file system check procedure, as provided by your operating system. If you are going to automate your FlashCopy procedures, consider including this check each time when you make the FlashCopy target available to the host. In all cases, before starting the FlashCopy procedure, the target volume must be unmounted; this ensures that there is no data in any system buffers that could be flushed to the target and potentially could corrupt it.

## Test plan and disaster recovery plan

If you plan to use FlashCopy, you need to test your setup. Do not forget that you are dealing with a binary copy of the data which was done out of control of your operating system. Prepare a test plan and, if you are using FlashCopy for backup/restore, prepare a recovery plan.

### 4.1.3 Operational considerations

The following suggestions are intended to help you manage your FlashCopy pairs and, in particular, manage the target you create.

#### Monitoring and managing FlashCopy pairs and volumes

FlashCopy pairs and tasks can be managed by both the ESS Specialist Copy Services Specialist Web panel and the Command Line Interface (CLI) on the host.

The ESS Specialist Web Interface allows you to manage FlashCopy volumes and tasks. You can establish and withdraw a FlashCopy by clicking on the graphical representations of the volumes in the Copy Services Specialist. If you wish to perform a FlashCopy from the CLI, you must create a FlashCopy task within the Specialist and save it. You can either execute your tasks from the Specialist, or you call them with the `rsExecuteTask` command in the CLI.

Using the CLI with predefined tasks enables automation, and it minimizes the danger of a human error when handling physical volumes by their volume numbers or names from the ESS Specialist Copy Services Specialist Web panel.

#### Using a FlashCopy target volume

Remember that if you have established a FlashCopy with the *Do not perform background copy* option, you need to withdraw the FlashCopy pair after you have finished using the FlashCopy target volume. If you choose to perform a full copy, the relationship will be withdrawn automatically when the background copy task ends. The performance issues of using FlashCopy with full copy and Do not perform background copy options are discussed in 4.2.5, “Performance considerations” on page 86.

If you will be using FlashCopy for data backup purposes, change your recovery procedure so that you will be able to recover even when the data has been backed up by a different backup client than the original owner of the LUN, or it has been backed up from a different location in the file system (the target mount point).



Of course, you can perform the FlashCopy from the restored volume to the original LUN using the full copy option that will perform the actual data copy to the target volume.

## Automation

Different operating systems allow different levels of automation. The automation can be done using batch or script files executed before and after the application that uses the FlashCopy target. In the batch file to be run before the application that will use the copy, you should include:

1. Quiescing of an application (switching on the backup mode). A proper quiesce procedure should be provided in your application's documentation.
2. Flushing data to the source volume. This can be accomplished by unmounting the target but sometimes it may be necessary to shutdown the source server.
3. Establishment of FlashCopy pair(s) using the Copy Services Command Line Interface with the `rsExecuteTask` command.
4. Re-mounting the source.

**Note:** Do not attempt to defragment or optimize a FlashCopy source volume while the FlashCopy background copy task is running. This can significantly degrade your performance.

5. Resuming an application (terminating the backup mode) using the procedure described in your application's documentation.
6. Hardware scan for new disks on a target system, in case their definitions are not already present.
7. Verifying the consistency of a FlashCopy target.
8. Mounting the target on a target system.

In the batch file to be run after backup is completed, you should include the following operations:

- Termination of the FlashCopy pair (in case the *Do not perform background copy* option was used) with an `rsExecuteTask` command.
- Unmounting the target from a target system.

### 4.1.4 Performance considerations

This section is intended to give you an idea of the considerations involved when setting up Copy Services of the Enterprise Storage Server (ESS) in order to achieve best performance.

It should assist you to understand the performance impact of ESS Copy Services. As there are many different parameters that influence performance, such as applications, kind of workload and configuration of the Enterprise Storage Server, the information should serve as a guideline when planning ESS Copy Services.

Please keep in mind that the general ESS performance considerations such as the volume placement or amount of storage per host adapter still apply when planning for FlashCopy.

### **Placement of source and target volume**

As we explained earlier in this book, the source and target volume of a FlashCopy pair must be in the same logical subsystem (LSS). In certain configurations of the ESS, a single LSS includes more than one RAID5 Array (rank). In such configurations, we recommend that you use a target and source volume from different ranks for your FlashCopy pair.

Furthermore, we recommend that you use ranks for your source and target volumes that reside on different SSA loops of the Device Adapters (DA) if possible in your ESS configuration. This will distribute the I/O load over more disks if data is copied from the source to the target volume.

If you are making a FlashCopy with the full copy option when all data from the source is physically copied to the target (default) and you do not have to work with the target volume directly after the FlashCopy was issued, we recommend that you wait until the background copy is finished. This will give you better performance for both source and target volume as host I/O requests do not interfere with I/O of the FlashCopy task. The progress of the FlashCopy background copy process could be determined with the Copy Services Web Interface, and the completion of the process with the Command Line Interface.

### **Do not perform background copy option**

Please consider whether you want to select the *Do not perform background copy* option for FlashCopy or not. This option is presented by the Select Copy Options panel of the Task Wizard as *Do not perform background copy if checked*.

If you mainly have read access to the source and the target of your FlashCopy pair, we recommend that you use the *Do not perform background copy* option to minimize I/O traffic to the RAID arrays. Keep in mind that when selecting this option, the relationship between the FlashCopy source and target stays until the pair has to be withdrawn manually.

### **Number of simultaneous FlashCopy pairs**

Also, you need to consider the number of FlashCopy pairs you have active at the same time. The time for a single FlashCopy pair to finish will increase with the amount of FlashCopy pairs you have established, at the same time as data is copied in between all pairs. Try to logically group FlashCopy pairs together and execute the different groups one after the other. When grouping multiple FlashCopy pairs into a single task, keep in mind that all of these pairs will be processed in parallel.

## **4.2 Peer-to-Peer Remote Copy (PPRC)**

PPRC is an established data mirroring technology that has been used for many years in mainframe environments. It is used primarily to protect an organization's data against disk subsystem loss or, in the worst case, complete site failure. In this section, we describe PPRC in detail. We explain how to set it up and give some practical examples of its implementation.

### **4.2.1 Terminology**

In this chapter, we use the following terms:

- ▶ A system where the production applications run is referred to as the primary site or application site.
- ▶ An Enterprise Storage Server where the production data resides, and which is the primary member of a PPRC pair, is referred to as a primary ESS or application ESS.
- ▶ A system where the recovery or test applications run is referred to as a secondary site or recovery site.
- ▶ An Enterprise Storage Server where copies of production data reside, which is used to keep the data current, and which is the secondary member of a PPRC pair, is referred to as a secondary ESS or recovery ESS.

### **4.2.2 Overview**

PPRC is a synchronous protocol that allows real-time mirroring of data from one Logical Unit (LUN) to another LUN in a second ESS. The secondary ESS can be located at another site some distance away. PPRC is application independent. Because the copying function occurs at the disk subsystem level, the application has no knowledge of its existence.

The PPRC protocol guarantees that the secondary copy is up-to-date by ensuring that the primary copy will be written only if the primary system receives acknowledgement that the secondary copy has been written.

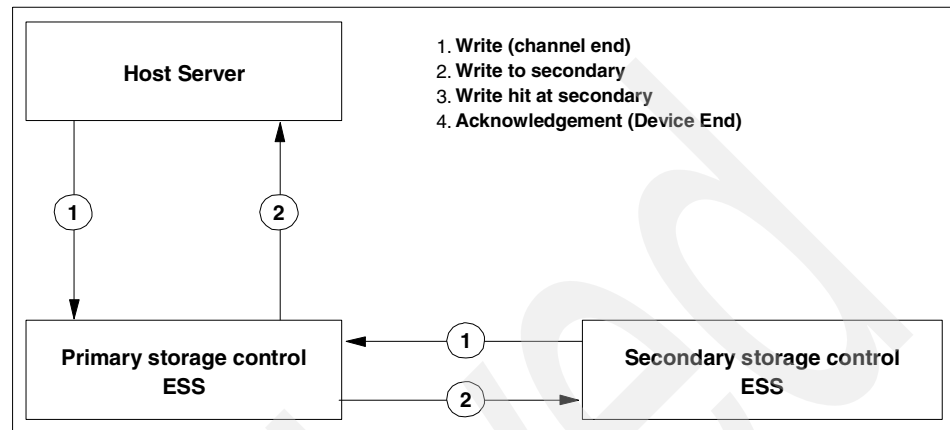


Figure 4-2 PPRC write cycle

1. The host server requests a write I/O to the primary ESS. The write is staged through cache into Non-Volatile Storage (NVS).
2. PPRC dispatches the write over an ESCON channel to the secondary ESS. The write hits the secondary ESS's NVS.
3. The primary then expects acknowledgment of the remote write. If the secondary write fails, the write does not return to the host server and is eventually "aged" from NVS.
4. The write returns to the host server's application.

Once acknowledgement of the write has been received by the primary, both the primary and secondary write I/Os are eligible for destage to disk. Destage from the cache to the disk drives on both the primary and the secondary ESS is performed asynchronously.

If acknowledgement of the remote write is not received within a fixed period of time, the write is considered to have failed, and is rendered ineligible for destage to disk. At this point, the application receives an I/O error, and in due course, the failed write I/O is "aged-out" of each NVS.

### PPRC volume states

Volumes within the Enterprise Storage Server used for PPRC can be found in one of the following states as shown in Figure 4-3.

### ***Simplex***

The Simplex state is the initial state of the volumes before they are used in any PPRC relationship, or after the PPRC relationship has been withdrawn. Both volumes are accessible only when in Simplex state.

### ***Duplex Pending***

Volumes are in Duplex Pending state after the PPRC copy relationship was established, but the source and target volume are still out of sync. In that case, data still needs to be copied from the source to the target volume of a PPRC pair. That may be the case either after the PPRC relationship was just established (or re-established for suspended volumes), or in case the PPRC volume pair re-establishes after a storage subsystem failure. The PPRC secondary volume is not accessible when the pair is in Duplex Pending state.

### ***Duplex***

This is the state of a volume pair that is in sync; that is, both source and target volume containing exactly the same data. Sometimes this state is also referred as the full copy mode. The PPRC secondary volume is not accessible when the pair is in Duplex state.

### ***Suspended***

Volumes are in Suspended state when the source and target storage subsystems cannot communicate any more, and therefore the PPRC pair could not be kept in sync, or when the PPRC pair was suspended manually. During the Suspended state, the primary volume's storage server keeps track of all updates to the source volume for reestablishment of the PPRC pair later on. The PPRC secondary volume is not accessible when the pair is in Suspended state. You can FlashCopy the PPRC secondary when it is in Suspended mode to another volume and use it the way a FlashCopy target can be used. It is necessary to comply with the FlashCopy source consistency requirements.

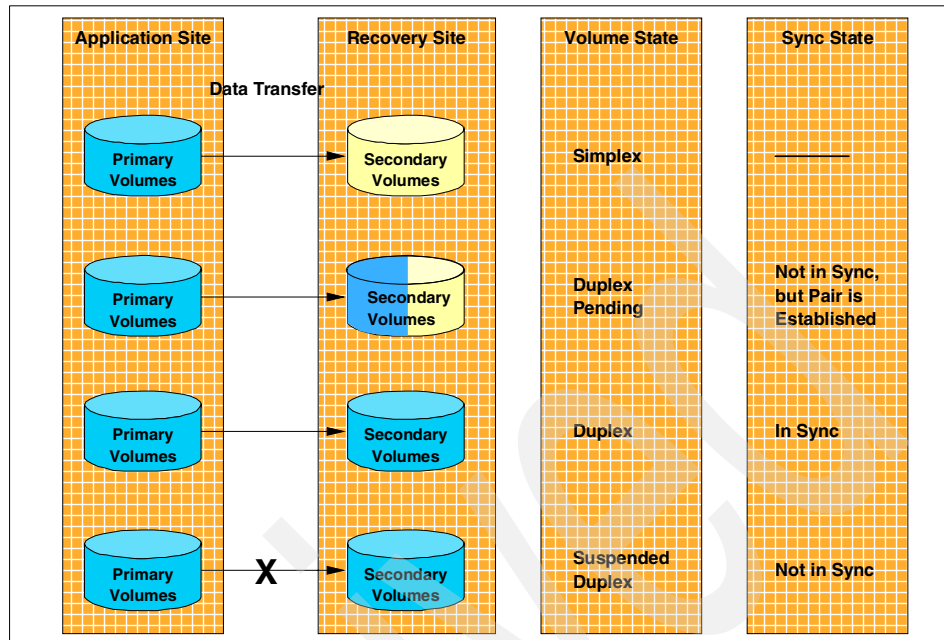


Figure 4-3 PPRC volume states

### 4.2.3 Planning for PPRC on an ESS

The following sections describe the important areas you should consider when planning for PPRC on an Enterprise Storage Server.

#### Hardware and software requirements

PPRC is possible only between Enterprise Storage Servers. Other disk storage units that support PPRC can also communicate only to the same type of unit. You need to have the PPRC feature purchased and PPRC-capable microcode activated on all ESS units that will be used for PPRC.

PPRC operates at a volume level from one LSS to another LSS. That means you need to have the target volumes available on the secondary ESS, and you need to identify the LSSs where the primary and secondary volumes are located.

ESCON connections have to be configured between the units (see 4.2.4, “Configuring ESS for PPRC” on page 82). There can be up to eight ESCON links between the subsystems. A primary ESS can communicate with up to four secondary ESSs, refer to Figure 4-4. A secondary ESS can be connected to any number of ESS primary subsystems.

You will need to purchase ESCON cables and possibly some other equipment, depending on the distance between the primary and the secondary ESS. However, each of the Copy Services Servers can only control two ESSs. Therefore, to have one primary ESS communicate with more than one secondary ESS, this will mean that multiple ESS Copy Services Servers will have to be configured.

The ESS units involved in PPRC must be connected with their standard Ethernet and TCP/IP to the units that are the primary and backup Copy Services servers. All ESS cluster hostnames, including its own, must be added to the cluster hostname list, the `/etc/hosts` file, during installation. This is configured by the IBM CE during the installation of the PPRC feature.

A browser for the ESS Specialist has to be installed on the machines that will be used to control PPRC with the Web interface. See *Implementing the Enterprise Storage Server in Your Environment*, SG24-5420, for the browser recommendations.

If you plan to use CLI, install the Java Developers Kit (JDK) JDK version 1.1.8 on the machines that will run the CLI commands. Check the *Host Attachment Guide* for the latest recommended JDK revision.

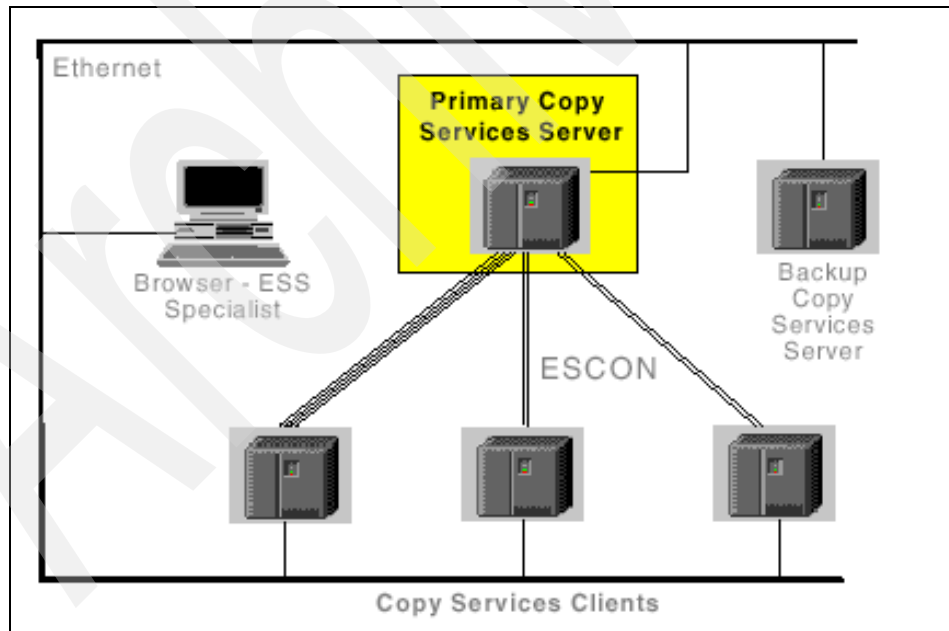


Figure 4-4 Ethernet and ESCON connection between Copy Services

## Configuration planning

An important setting is the CRIT parameter in the ESS Vital Product Data (VPD). This parameter only applies to System 390 volumes. This section is included here for the sake of completeness. CRIT is a parameter that determines the behavior of the PPRC pairs or consistency groups after a failure in communication between the primary and secondary ESS, when all the paths between a pair are lost. This parameter is set using ESS Web CopyServices or through TSO commands.

CRIT (NO) specifies that, following an I/O completion error to the secondary volume, PPRC allows subsequent write requests to the PPRC primary volume. The primary ESS control will perform change recording. The default is NO. CRIT (YES) specifies that if an I/O error to the secondary volume occurs, PPRC either allows or does not allow subsequent writes to the primary, depending on how the storage subsystem is configured. The PPRC pair then remains in a suspended state until you correct the problem and either issue a command to resync the PPRC pair or delete the PPRC pair.

The implementation of CRIT (YES) on the ESS is similar to the implementation on the IBM 3990. There is an option that can be set by the CE in the VPD of the ESS which determines how this CRIT (YES) setting will behave in an error situation:

- ▶ CRIT=YES - Paths (light version)
- ▶ Suspend the pair and do not accept any further writes if the control units can no longer communicate.
- ▶ Suspend the pair and accept further writes if the control units still can communicate with each other. The reason for not being able to copy the data to the remote volume is probably only a device problem on the secondary site and not a disaster. Therefore we continue with write operations to the primary volume. The ESS records which cylinders have changed. After investigation of the problem and after it has been solved, you can resynchronize source and target volume again.
- ▶ Suspend the pair and do not accept any further writes to the primary volume if data cannot be sent to the secondary volume.

## Resource planning

When planning your secondary ESS volume layout for PPRC, optimize your disk capacity. It is important to realize that the capacity needed on the secondary ESS for disaster recovery may not have to be initially as large as the primary ESS. A disaster recovery plan (DRP) requires significant investment financially in technology, people, and process. Every company will be different, but the I/T components of a disaster recovery plan are essentially driven by the applications and data you require for business continuity, should a disaster occur. Some



applications and data will be more critical than others. An organization will typically require its core business systems to be available in a short time, whereas less critical systems quite possibly could be restored over a number of days.

Bearing in mind that the disk space you need for PPRC secondary volumes is real disk space, size your secondary ESS based on your critical business requirements, possibly with some headroom for applications of intermediate importance. Create PPRC pairs for the critical data so that it is copied in real time. Then, if a disaster happens, you will have the core systems available on the secondary copies. After the initial recovery priorities have been handled, you can add more disks ranks for the applications of lower importance and restore them from tape.

### **Data consistency considerations**

In any recovery situation, including disaster recovery scenarios, you may be exposed to so-called lost writes. These are the “in-flight” transactions that have not been committed from memory to the ESS’s NVS. You should expect that uncommitted transactions will be lost. On the other hand, data that was transferred to the ESS and confirmed back as written into the NVS (of the secondary ESS in case of PPRC), will be destaged to disk.

Invariably, a host server will check its file systems after recovering from a crash. At a disaster recovery (DR) site, the host servers may be operational when the primary site fails. So it is important to perform a full file system check on all PPRC secondary volumes before you start using them. Of course, rebooting the servers will achieve the same result. When your database restarts, normal database recovery commences and any partially committed transactions will be rolled back.

### **Test plan and disaster recovery plan**

A DRP is complex — nothing can understate the importance of rehearsals and testing your environment. You only get one shot at getting it right when a real disaster hits.

Carefully set up your PPRC tasks for establishing and terminating pairs. Ensure that they are well tested and documented. Prepare your documentation as if it were intended for someone else; you may not be around when a disaster strikes. Make sure you understand any operating system specific issues related to bringing your PPRC secondaries online. Have them well documented in your recovery operations control book.

## 4.2.4 Configuring ESS for PPRC

In this section, we describe how to set up an ESS in preparation for PPRC.

### ESCON

For PPRC primary to secondary unit (channel to control unit) communication, a maximum of eight ESCON links using modified ESCON protocol can be configured. ESCON channels provide 160 Mbps point-to-point links. While PPRC can be bi-directional, these links are uni-directional. The primary unit ESCON port (the one in channel mode) has to be dedicated for PPRC. The ESCON port on the secondary unit can also be used for S/390 host attachment, provided the ESCON director is used, and the host is connected to it.

ESCON links support distances of up to 2 km with 50 micron multimode optical cables and up to 3 km with 62.5 micron multimode cables. The ESCON channel performance is a function of distance.

By using up to two ESCON Directors, you can extend these distances. You can use Extended Distance Feature (XDF) ports with single mode fiber optic cables, the XDF maximum distance being 20 km. The total maximum distance is 103 km between the two ESSs.

Various channel extenders can also be used to increase the distance between ESS servers. IBM 9729 Optical Wavelength Division Multiplexer (MuxMaster) enables a 50 km distance between MuxMaster units. ESCON is used to attach ESS to it.

There is a new product, IBM 2029 Fiber Saver, also known as Dense Wavelength Division Multiplexer (DWDM), that supports ESCON, FICON, Fibre Channel, and many more protocols, enabling up to a 50 km distance between Fiber Saver units that are ESCON attached to the primary and secondary ESS. You can use it for Fibre Channel, network, and telephone links between the sites as well.

### ESS

PPRC requires logical paths to be established between the primary and the secondary ESS logical control units (or LSS). Each LSS in the primary ESS that will use PPRC requires at least one path to be set to the LSS in the secondary ESS that holds the secondary volumes (Figure 4-5).

Each ESCON link supports 64 logical paths, so even with 16 LSS defined you are able to set up a logical path from each LSS to each LSS with only four ESCON PPRC links. We always recommend that you use all available links for each LSS pair used for PPRC. That gives you maximum protection against ESCON link failure.

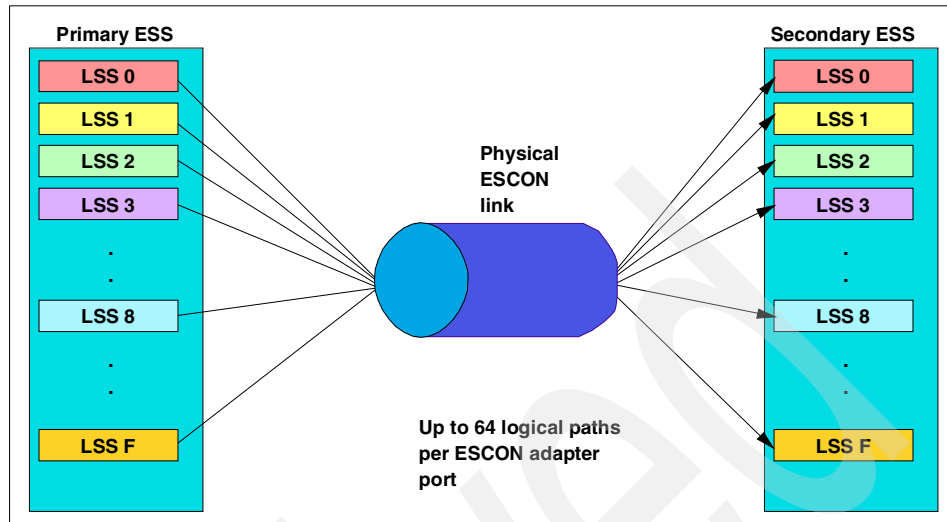


Figure 4-5 ESS logical paths

## Using PPRC with FlashCopy

One good example of combined PPRC with FlashCopy is the snapshot solution for backup and recovery. This solution has been tested by OLTP with ESS, and its description may also help you to design your own snapshot solutions for other applications.

The basic idea of the snapshot is managing a consistent copy of the database in a certain point of time on a secondary site (the t0 copy) while production continues on a primary site on the t2 copy. On the secondary site, a FlashCopy of the consistent t1 copy is created, referred to as the t0 copy, that enables offline backup to tape and application testing (see Figure 4-6).

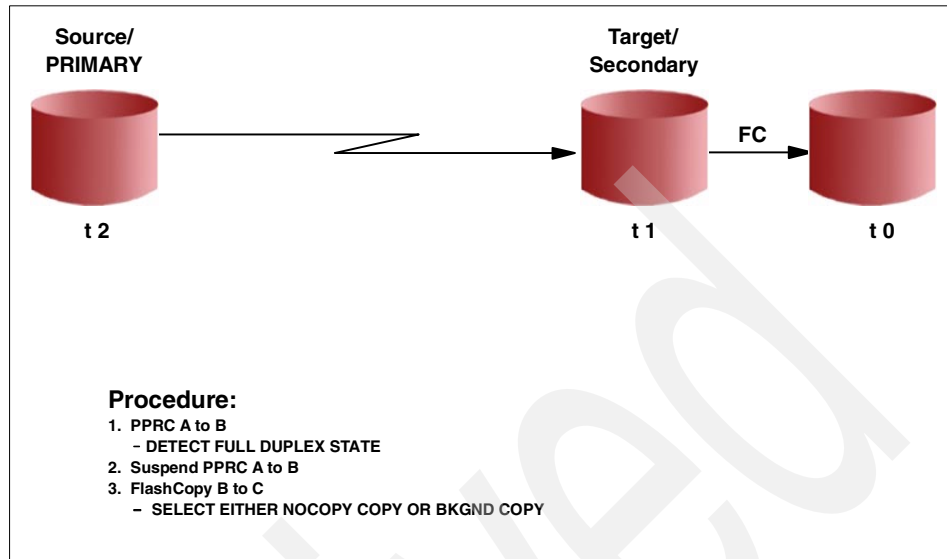


Figure 4-6 PPRC with FlashCopy

The PPRC pairs are normally suspended, and the mirror is split, preventing propagation of logical errors to the t1 secondary copy. At certain checkpoints, the pairs are resumed and resynchronized; this means that the database is copied to secondary. This allows a restart of the application on the last known consistent copy, or a roll forward to a certain point of time using log files. The database active and archive logs are copied constantly; the volumes they are at are not suspended.

In case of an application logical error, which is more frequent than a total site failure, the production resumes on the secondary database, while the primary can be analyzed. In case of a hardware failure, you can switch quickly to the secondary site.

The implementation planning should include identification of logical to physical volume mapping:

1. Preparation of both the primary and secondary site ESSs
2. Setup of PPRC ESCON channels and paths
3. Setup of PPRC tasks
4. Setup of recovery site hosts
5. Testing

Identification of mapping will include creating a list of physical volumes that contain the database and the logs, as well as checking that they are placed on physical volumes properly in such a way that they can be suspended and resumed. For example, the database cannot be on the same physical volume the logs are on.

The preparation of the ESSs means purchasing the PPRC features and installing ESCON adapters, checking that disk space is available on the selected LSS, and installing Java JDK version 1.1.8 on hosts in order to use CLI. You need to have PPRC-dedicated ESCON channels available between the primary and secondary sites.

Tasks have to be set up using the Web interface; and then they are ready to be executed from the CLI. You also need to group the tasks into task groups to be able to carry out the point-in-time critical operations (establish, suspend, resync, FlashCopy) into task groups. The host server on the remote site has to be set up so it can take over in case of a primary site failure. All the PPRC tasks, backup operations, and site takeover have to be tested regularly.

Before you start using snapshot routinely, you have to perform the initialization steps, either with the Web interface or CLI.

1. Establish the paths between primary and secondary ESS.
2. Establish the pairs for all necessary volumes (establish, copy entire volume).
3. Query the pairs if establish is complete.
4. Suspend write I/O to the primary t2 volumes.
5. Withdraw the pairs between primary and secondary ESS (suspend pairs).
6. Resume write I/O to the primary t2 volumes.
7. FlashCopy the t1 volumes on the secondary ESS to t0 copy.
8. Resynchronize the primary t2 and secondary t1 volumes (establish, copy changes only).
9. Dump or backup the t0 copy to tape.

Repeated routine steps will include these:

1. Create safety copy if resynchronization fails (suspend all, recover t1 and FlashCopy t1 to t0).
2. Resynchronize the primary t2 and secondary t1 volumes.
3. Query the status of resynchronize process.
4. Suspend write I/O to the primary t2 volumes.
5. Withdraw the pairs.

6. Resume the write I/O.
7. FlashCopy the t1 volumes to t2.
8. Resynchronize the primary and secondary volumes for BSDS, active and archive logs.
9. Dump or backup the t0 copy to tape.

## 4.2.5 Performance considerations

The following sections are intended to show you the considerations involved when setting up Copy Services of the Enterprise Storage Server (ESS) in order to achieve better performance.

This should help you to understand the performance impact of ESS Copy Services. As there are many different parameters that have an influence on performance, such as applications, type of workload, and configuration of the Enterprise Storage Server, the information should serve as a guideline when planning ESS Copy Services.

Please keep in mind that the general ESS performance considerations, such as volume placement or amount of storage per host adapter, still apply when planning for PPRC.

### Optimized PPRC communication

There were certain modifications made to the ESCON protocol used for PPRC communication of the Enterprise Storage Server; in particular:

- ▶ A larger frame size, which results in less overhead during PPRC communication.
- ▶ Less handshaking between the two communicating ESSs, which makes transfer of data more efficient. The handshake was reduced from 6 down to 3 exchanges.

### Number of ESCON paths between Enterprise Storage Servers

When a host system is sending a write request to a primary volume of a PPRC pair, an I/O complete will be returned to this host once the data is written to the source and target ESS (synchronous copy). This will have some performance impact upon the write I/O operations of the application.

Always make sure that you are using an appropriate number of ESCON paths for PPRC between the source and the target ESS. Increasing the number of the physical ESCON links will increase the maximum overall bandwidth for updating the targets. Using multiple ESCON connections for a PPRC pair (maximum of 8) will improve the response time of an I/O request from the host. Keep in mind that too few paths may result in a bottleneck. A minimum of four paths between the primary and secondary ESS are recommended.

### **Placement of the ESCON adapters used for PPRC**

Distribute the ESCON adapters used for PPRC evenly across the two clusters and the host adapter bays of the ESS. This will distribute the PPRC workload over multiple busses and both clusters.

For example, if there are four ESCON adapters used for PPRC between two Enterprise Storage Servers, place one ESCON adapter in each of the host adapter bays.

### **Grouping of physical and logical paths**

A physical path describes the physical ESCON connection between two Enterprise Storage Servers. A logical path is the connection used for the PPRC copy pair, either between two volumes or two logical subsystems. There could be multiple logical connections established over a single physical connection. This will be most likely the case in a real environment.

Also, consider that multiple logical paths will share the bandwidth of the ESCON path(s) between each other. If there are critical PPRC pairs, we recommend that you separate them on dedicated physical paths so that the I/O traffic of the data copy from the primary to the secondary side will not interfere with I/O traffic of lower critical PPRC pairs.

### **Setup of the secondary ESS**

For disaster recovery reasons, you may be doing PPRC between two or more different Enterprise Storage Servers. Under normal operating conditions, you always have a source (primary side) and a target (secondary side) of a PPRC pair.

One single ESS could have up to four secondary ESSs. However, the number of primary servers of a single secondary server is only limited by the number of available ESCON links. So it may be the case that different primary storage servers are connected to the same secondary ESS. In that case, the I/O traffic of multiple primaries has to be computed by a single secondary ESS.

Furthermore, it may be possible that secondary volumes from different primary storage servers are placed on the same disks within the same Array (rank). In that case, the response time of each primary storage server will increase if other primaries are doing I/O at the same time as all requests are handled simultaneously.

Therefore, when planning your Enterprise Storage Server network, keep in mind how many primary storage servers are connected to the same secondary. Distribute the I/O load evenly across the secondary storage server.

Try to distribute the volumes used for PPRC pairs of multiple primaries across all available RAID arrays within the secondary ESS.

### 4.3 ESS Copy Services Web interface

There are two different methods of using the ESS Copy Services in the Open Systems environment:

- ▶ A Web-based interface
- ▶ A Java-based Command Line Interface (CLI)

For more information on the ESS Copy Services Web interface, refer to *Implementing ESS Copy Services on UNIX and Windows NT/2000*, SG24-5757.



## Scenarios

Database backup and recovery strategies depend on the customer's environments. In this chapter we discuss eleven scenarios which, help customers understand when FlashCopy and PPRC is needed and how to backup and recover a database to continue database services without stopping the server. To understand the scenarios easily, we described the general test environment setup and configuration.

## 5.1 Test setup and configuration

The test setup and configuration involves the database design, storage consideration, network and entire snapshot infrastructure. The database design includes the mapping of the database physical files and storage in a FlashCopy and Peer-to-Peer Remote Copy(PPRC) environment. The network infrastructure shows this test network environment. The snapshot infrastructure explains the server and ESS logical configuration, and also the flows required to understand each backup and recovery scenario.

### 5.1.1 DB2 UDB and storage configuration

The instances are created in each local disk. So this instance is not flashcopied. The setup of the DB2 instance must be the same on the source and on the target machine. Also, the operating system and DB2 version should be the same.

The primary volume (A) and the local FlashCopy volume (B) should be on the same Logical Subsystem (LSS). The PPRC Target volume (C) and the remote FlashCopy volume (D) should be also on the same LSS.

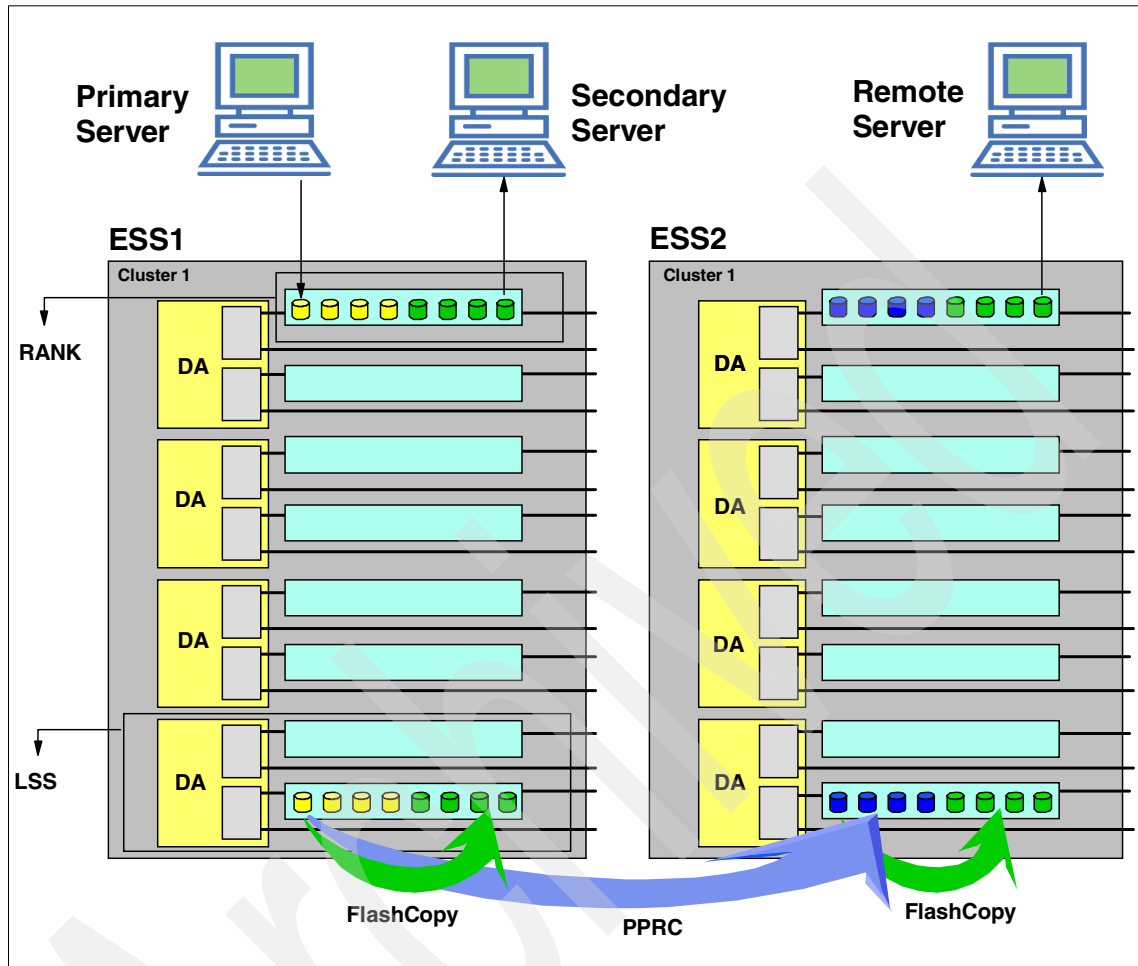


Figure 5-1 DB2 UDB and ESS mapping

In general, it is important to place the transaction logs and the data files on separate physical disks or volumes. Through the ESS is designed to be fully fault-tolerant, this separation provides yet another layer of protection. Should the data files be lost, it would still be possible to recover up to the point of failure using the log files and a backup of the data files. Also, there can be a significant performance benefit in placing these files on separate media. With this in mind, we set up the DB2 UDB database to spread the data and logs across all of the available disks in the ESS for best performance. We also spread the load across

both clusters, all host bus adapters, and all internal logical subsystems (LSSs) available within the ESS. This helps to avoid "hot spots" within the storage subsystem that receive most of the load while other parts of the subsystem are idle.

Furthermore, all of the transaction logs were mirrored on a separate set of disks. The ESS is designed to avoid single points of failure (such as a disk, adapter, CPU, or memory module failure) and most double and triple failures, with its fully redundant, highly available architecture. The possibility exists, however remote, that a combination of hardware failures can occur that would cause data loss — usually during a disaster. Having a mirror of the log is extra insurance against such a scenario. Additionally, in some scenarios, we have a mirror at a remote site to protect against disasters, such as earthquake or fire.

### 5.1.2 Network topology

These servers were connected into a Storage Area Network (SAN). The SAN had the following topology, as shown in Figure 5-2.

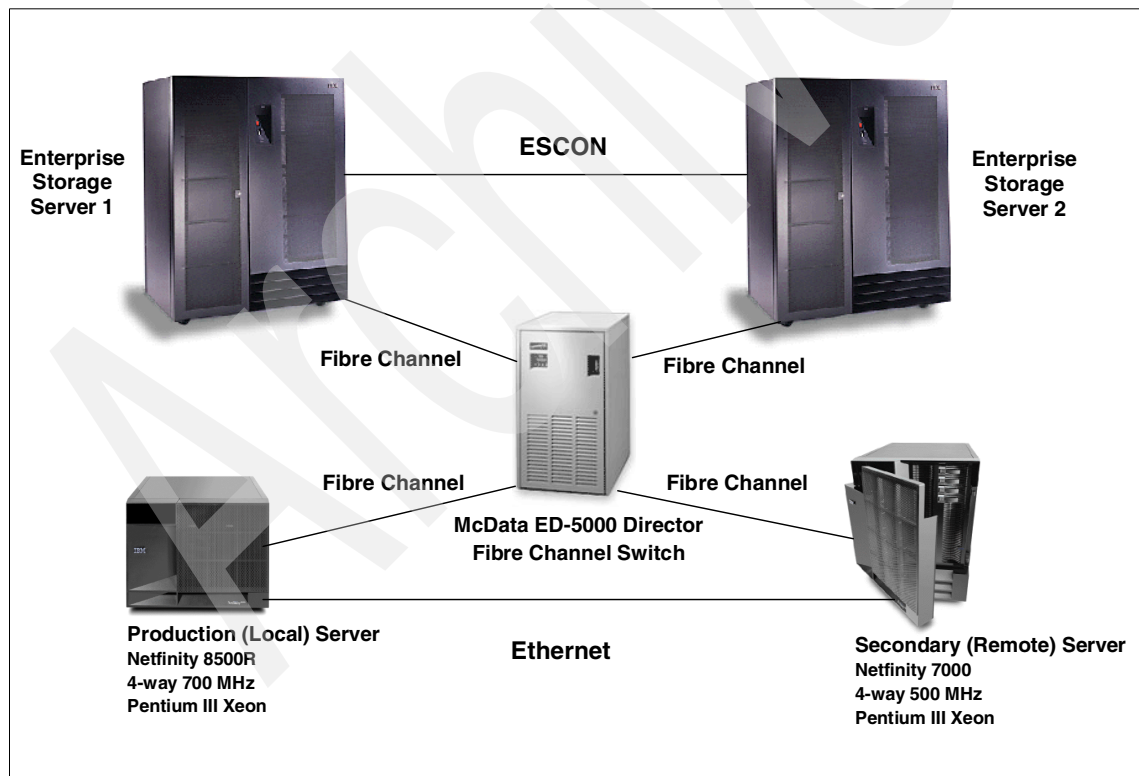


Figure 5-2 SAN physical topology

In this configuration, with multiple Fibre connection to McData switch via multiple ports cards and Subsystem Device Driver (SDD) supports fault tolerance. Even though the switch is an enterprise-class, fully redundant Director, the possibility of disaster and multiple failures still exists. The McData switch is meant to represent a SAN fabric, consisting of several redundant paths and switches from edge to edge.

### 5.1.3 Snapshot infrastructure

To cover DB2 UDB backup and recovery with local FlashCopy and remote PPRC, we devised a three server and two Enterprise Storage Server (ESS) configuration. The primary production server, secondary server and Enterprise Storage Server1 (ESS1) are located at the production site. The remote server and Enterprise Storage Server2 (ESS2) are located at a remote site. Figure 5-3 is a logical view of the environment configuration.

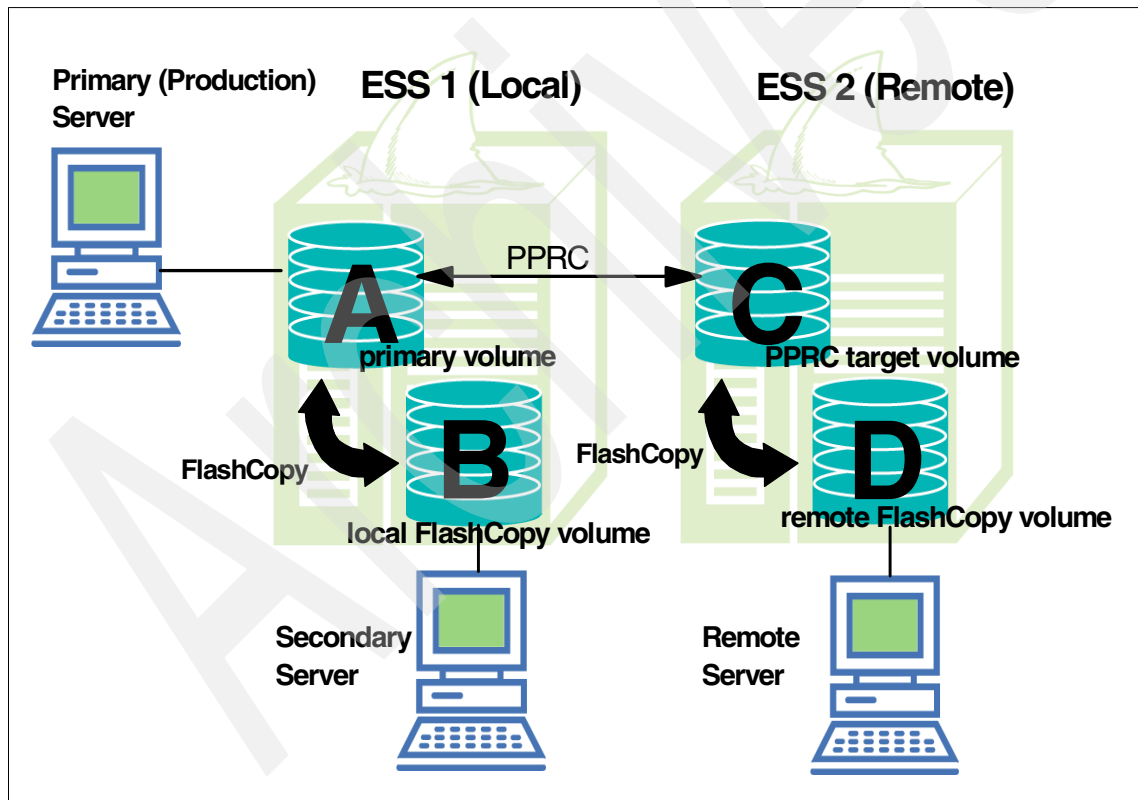


Figure 5-3 Snapshot infrastructure

At the production site, the ESS contains two copies of the production database. The first copy (A) is used for production. The first copy (A) can be flashcopied on to the local volume (B). The flashcopied volume (B) is then sent to tape using the resources of a secondary server (not the production server). After that, the local volume (B) is kept in case a rapid restore is required.

At the remote site, the second ESS keeps a PPRC synchronous volume (C) of the production database. Should the primary server fail, the backup server at the remote site can take over, with typically little or minimal down time. It is also possible to take a FlashCopy of the PPRC synchronous volume (C) to produce a fourth copy (D) that can be used for a hot standby database or additional protection like data mining/business intelligence and application development, and so on.

Ideally, we would recommend having a secondary server at the local site to send the backup to tape, and a remote server at the remote site able to take over in case of disaster.

## 5.2 Snapshot backup

Customers have large databases to accompany their company's growth. They keep their assets in the database system even through the database backup. But the size of database prevents the customers from running the database system. That requires fast database backup without affecting their database and applications. They require the high availability even in a disaster situation.

If you would rather not back up a large database using the DB2 backup utility, you can make copies from a mirrored image by using DB2 UDB write suspend new feature and FlashCopy and PPRC function of ESS Copy Services.

In the lab, we devised five typical scenarios a customer may use to perform a database backup. Using FlashCopy and PPRC, it is possible to make dozens of copies in different configurations.

- ▶ “Scenario 1: Offline FlashCopy” on page 96  
A solution for customers who require fast backup and recovery.
- ▶ “Scenario 2: Online FlashCopy” on page 97  
A solution for customers who require fast backup and recovery, and 24x7 availability.
- ▶ “Scenario 3: Offline PPRC with FlashCopy” on page 100  
A solution for customers who require fast disaster recovery.

- ▶ “Scenario 4: Online PPRC with FlashCopy” on page 101  
A solution for customers who require fast disaster recovery and 24x7 availability.
- ▶ “Scenario 5: Online PPRC with FlashCopy” on page 104  
A solution for customers who require fast backup/recovery, disaster recovery and 24X7 availability.

While these backup scenarios performed as expected under significant load in the lab, we recommend that the backup be performed at times of minimal load (such as during the night). In each case, DB2 UDB had been installed at the secondary server and was standing by. Furthermore, each of our scenarios included error detection mechanisms. We recommend that customers use rigorous error checking in their implementation of these scenarios.

Here are some of the advantages:

- ▶ Eliminates backup operation overhead from the production machine.
- ▶ Represents a fast way to clone systems.
- ▶ Represents a fast implementation of idle standby failover.

There is no initial restore operation, and if a roll forward operation is too slow, or encounters errors, reinstallation is very fast.

Do not start the copied database or applications on the secondary system if the local mirror volume will be flashcopied back to the primary volume for restore. Instead, use an application such as NTBackup or Tivoli Storage Manager to make a flat file-level backup. Restarting the database can modify the logging chain of the database so that it can no longer be used for roll forward recovery. The steps for starting the database on the secondary should only be used for data mining, application development, and so on.

Each of these scenarios is described in further detail in the following sections. Also, using this snapshot image is discussed in 5.3, “Using snapshot” on page 105.

### 5.2.1 FlashCopy in local site

These scenarios require one ESS at the local production site. Through the storage copy function in ESS Copy Services, these scenarios provide very fast backups and restores. The DB2 database backup is performed in the secondary server, therefore this offloads the primary server's backup workload. The flashcopied backup is used for fast production restores, local standby databases, the development environment, data mining, and so on.

For customers who have a large database but a short window of backup time, this is a good solution. But this solution doesn't offer any protection against disasters, such as fire, earthquake, flood, and so on.

This backup can be performed during online or offline time.

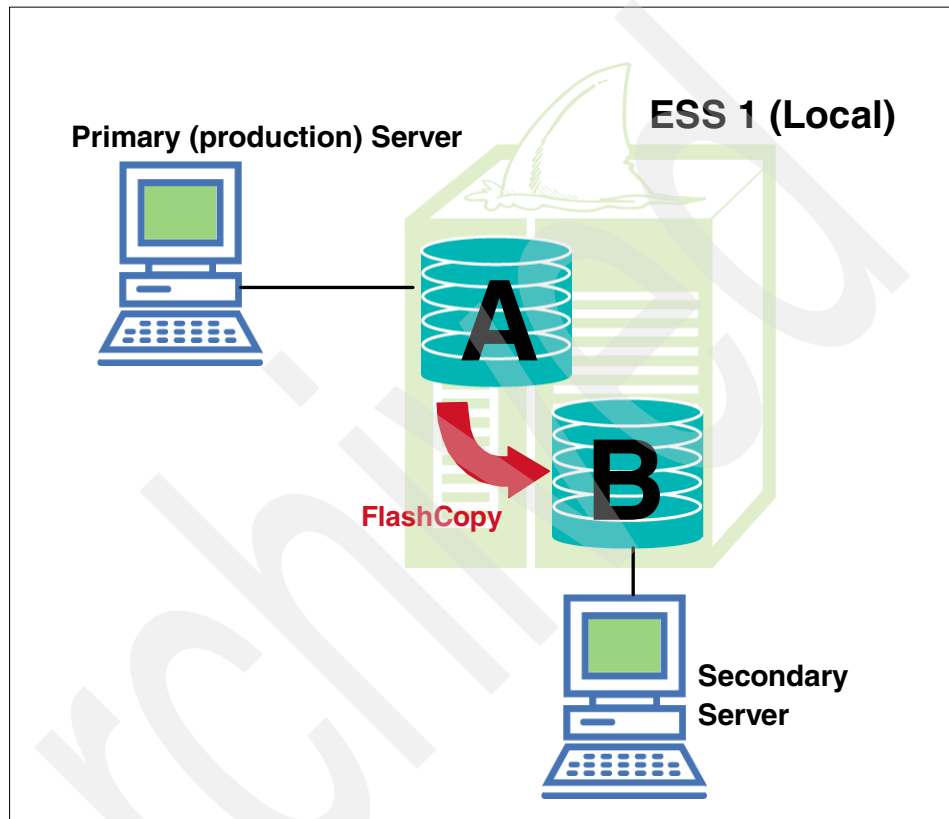


Figure 5-4 FlashCopy in local site

### Scenario 1: Offline FlashCopy

Customers that require a known, consistent state in a certain point of time, and do not require 24x7 availability should use an offline FlashCopy backup. Especially customers that want to have a circular logging database to avoid the backup pending state after massive data loads have to use an offline FlashCopy backup.

This backup offloads the primary database backup workload at the cost of short database downtime. This FlashCopy is also used for logical error recovery and version recovery.



Figure 5-4 shows logically, after all the applications and the primary database shutdown, the primary volume (A) is flashcopied to the local volume (B). If there is a requirement for a complete physical copy of the local volume (B), the local volume (B) is attached to the secondary server and the backup is done at the backup device.

The steps for performing the backup are as follows:

1. Stop all applications connected to the production database:  
`db2 force applications all`
2. Stop the production database:  
`db2stop`
3. Flush file system buffers on the primary server and secondary server.
4. Perform FlashCopy using rsExecuteTask ESS Copy Services CLI command or ESS Specialist with COPY option.
5. Restart the production database:  
`db2start`
6. Restart all applications on the primary server.

Here are optional requirements to take a backup using DB2:

7. Resync the secondary server filesystem to make the server aware of new disks.
8. Catalog the database in volume (B), start the database and take a backup on the secondary server:
  - a. `db2 catalog database database_alias on database_path`
  - b. `db2start`
  - c. `db2 backup database database_alias to /dev/mnt0`

**Note:** You cannot use the suspend I/O command when you have a circular logging database. Instead you can take a FlashCopy after you shutdown the database.

## Scenario 2: Online FlashCopy

Customers that require 24x7 availability can perform an online backup instead of an offline backup. While the backup is not necessarily in a known state, it is still consistent and can be used for recovery.

This backup offloads the primary database backup workload without applications and database shutdown. This FlashCopy is also used for logical error recovery, version recovery or a local hot standby database.

Figure 5-4 shows logically, after the primary database write I/O is suspended, the primary volume (A) is flashcopied to the local volume (B). If there is a requirement for a complete physical copy of the local volume (B), the local volume (B) is attached to the secondary server and the backup is done at the backup device.

**Attention:** DB2 set write suspend for database must be done in pairs using the *same* DB2 connection.

The steps for performing the backup are as follows:

1. Quiesce the production database write I/O:  
**db2 set write suspend for database**
2. Perform FlashCopy using rsExecuteTask ESS Copy Service CLI command or ESS Specialist with COPY option.
3. Resume the production database write I/O:  
**db2 set write resume for database**

Here are optional requirements to take a backup using DB2:

4. Resync the secondary server filesystem to make the server aware of new disks.
5. Catalog the database in the mirror volume (B), initialize the database, and take a backup on the secondary server:
  - a. **db2 catalog database *database\_alias* on *database\_path***
  - a. **db2start**
  - b. **db2inidb *database\_alias* as snapshot**
  - c. **db2 backup database *database\_alias* to /dev/mnt0**

**Note:** You should initialize the copied database to take a backup of the local volume (B) using DB2. You can initialize the local volume (B) as a snapshot, standby, and mirror depending on situation. When you initialize the copied database as a snapshot, you can take online or offline backups at any time.

## 5.2.2 PPRC with FlashCopy in remote site

To add protection from disasters (such as fire, earthquake, and so on), these scenarios require one ESS at the production site and one ESS at the remote site for disaster recovery using Peer-to-Peer Remote Copy (PPRC) function in ESS Copy Services. For the customers who have a large database that requires a business to keep running, even when they happen to face disaster in the local site, this is a good solution.

These scenarios provide fast remote copy for a database and this Peer-to-Peer synchronous mirror image is used for the disaster recovery, for example reverse PPRC recovery, hot standby, and so on. PPRC target volumes can be also used for generating reports, data mining, developing databases in the remote server.

Whenever a PPRC pair is established, data of a PPRC source are copied to a PPRC target volume. As the PPRC relationship between the PPRC source and the PPRC target is kept through suspending and resuming PPRC pairs, the workload and the period of synchronizing the PPRC source with the PPRC target is sharply decreased, especially to the customers that have a circular logging database and establish a PPRC pair frequently.

This backup can be performed during online or offline time.

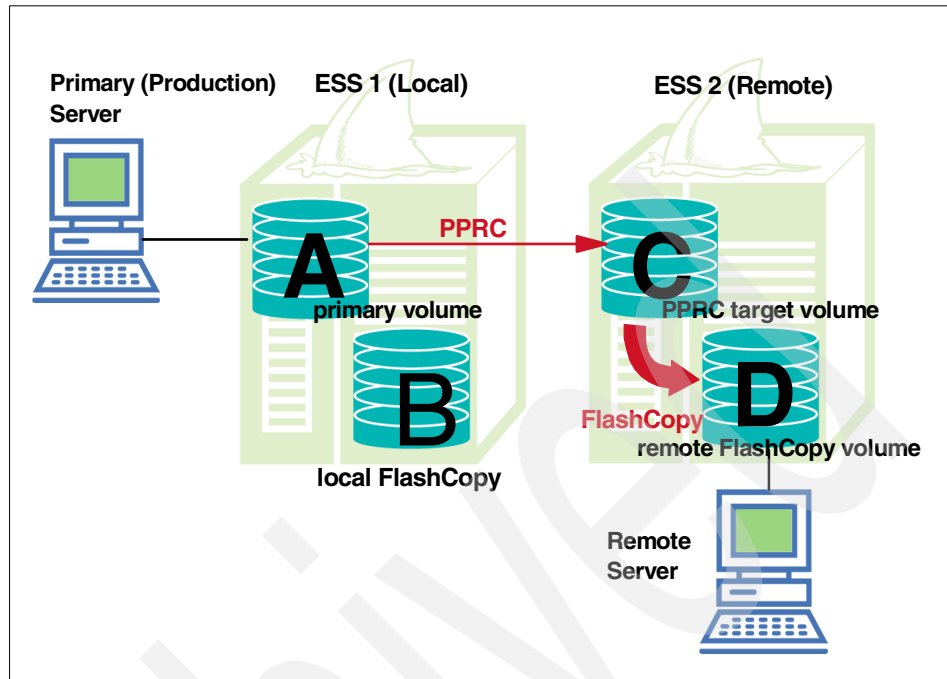


Figure 5-5 PPRC with FlashCopy in remote site

### Scenario 3: Offline PPRC with FlashCopy

This scenario provides disaster recovery protection for customers. Furthermore, the flashcopied remote volume (D) of the PPRC target volume (C) at the remote ESS (ESS 2) can be used to send the backup to tape, eliminating any overhead on the production ESS and production server. This backup requires all the applications and the primary database short downtime for database consistence. This remote volume is used for reverse PPRC recovery, generating report, data mining, developing database in the remote server.

Figure 5-5 shows the PPRC is done from the primary copy(A) to the PPRC target volume (C) during all the applications and the primary database shutdown. The PPRC target volume (C) is flashcopied to the remote volume (D). If there is a requirement for a complete physical copy of the remote volume (D), the volume (D) is attached to the remote server and the backup is done at the backup device.

The following steps are involved in the offline PPRC with FlashCopy. ESS Copy Services CLI command or ESS Specialist Web Interface should be used in the initialization steps:

1. Establish a PPRC pair between the primary volume (A) and the PPRC target volume (C).

2. Wait until the PPRC is completed and two volumes are in sync using `rsQueryComplete` CLI command.
3. Stop all applications connected to the production database.
4. Stop the production database:
  - a. **`db2 force applications all`**
  - b. **`db2stop`**
5. Flush file system buffers on the primary server and the remote server.
6. Perform FlashCopy from the PPRC volume (C) to the remote volume (D) using `rsExecuteTask` CLI command with `COPY` option.
7. Restart the production database:
  - a. **`db2start`**
8. Restart all applications in the production server.
9. Resume the PPRC pair; thus resynchronize all the changed activities in the primary volume (A) during PPRC suspending with the PPRC target volume (C).

Here are optional requirements to take a backup using DB2:

10. Resync the remote Server filesystem to make server aware of new disks.
11. Catalog the database in the remote volume (D), start the database and take a backup on the remote server:
  - a. **`db2 catalog database database_alias on database_path`**
  - b. **`db2start`**
  - c. **`db2 backup database database_alias to /dev/mnt0`**

**Note:** You cannot start up the PPRC target volume (C) as a remote database. You should take a FlashCopy of the PPRC target volume to take a backup in the remote server.

## Scenario 4: Online PPRC with FlashCopy

This scenario provides 24x7 availability and adds disaster recovery protection for customers. Instead of shutting down all applications and UDB, we suspend write I/O on the UDB database using the `set write suspend` and `set write resume` commands introduced in UDB 7.1 (Fixpack 2). The writes are frozen only for the time it takes to invoke FlashCopy on the remote ESS.

For the disaster recovery, the PPRC target volume or standby database can be used. The following step is for a standby database using the new DB2 UDB feature. The PPRC relationship between the primary copy(A) and the PPRC target volume (C) is terminated and the remote volume (D) flashcopied from the PPRC target volume (C) is going to be used for the standby database. When keeping the PPRC relationship, for suspending and resynchronizing the PPRC pair, keep the PPRC target volume for disaster recovery (refer to “Scenario 3: Offline PPRC with FlashCopy” on page 100). Of course this remote volume (D) could be initialized as a snapshot for a clone database. Figure 5-5 on page 100 shows the logical flow.

ESS Copy Services CLI command or ESS Specialist Web Interface should be used in the initialization steps. The steps involved in performing the online PPRC with FlashCopy are as follows:

1. Establish a PPRC pair between the primary volume (A) and the PPRC target volume (C).
2. Wait until the PPRC is completed and the two volumes are in sync using `rsQueryComplete` CLI command.
3. Flush file system buffer, then quiesce the production database write I/O:  
**db2 set write suspend for database**
4. Terminate the PPRC pair.
5. Resume the production database write I/O:  
**db2 set write resume for database**
6. Perform FlashCopy from the PPRC target volume (C) to the remote volume (D) using `rsExecuteTask` CLI command with `COPY` option

Here are optional requirements to take a backup using DB2:

7. Resync the remote server filesystem to make the server aware of new disks.
8. Catalog the database in the remote mirror volume (D), initialize the database, and take a backup on the remote server:
  - a. **db2 catalog database *database\_alias* on *database\_path***
  - b. **db2start**
  - c. **db2inidb *database\_alias* as standby**
  - d. **db2 backup database *database\_alias* [online] to /dev/mnt0**

**Note:** You cannot initialize the PPRC target volume as a remote database. You should take FlashCopy of the PPRC target volume to take a backup in the remote server. You can take an online standby database backup only right after issuing the db2inidb as standby.

### 5.2.3 PPRC with FlashCopy in local and remote site

In this comprehensive solution, which uses the technologies from the four previous scenarios, there are four copies of the database at any time. Copy (A) is the production copy; (C) is its PPRC target volume for disaster recovery. Copy (B) is a local FlashCopy that can be used for rapid restores, while (D) is a copy that can be used to initialize as standby, backup to tape, data mining, and other functions. This scenario would be used by customers requiring 24x7 availability, rapid restores, and disaster recovery capability.

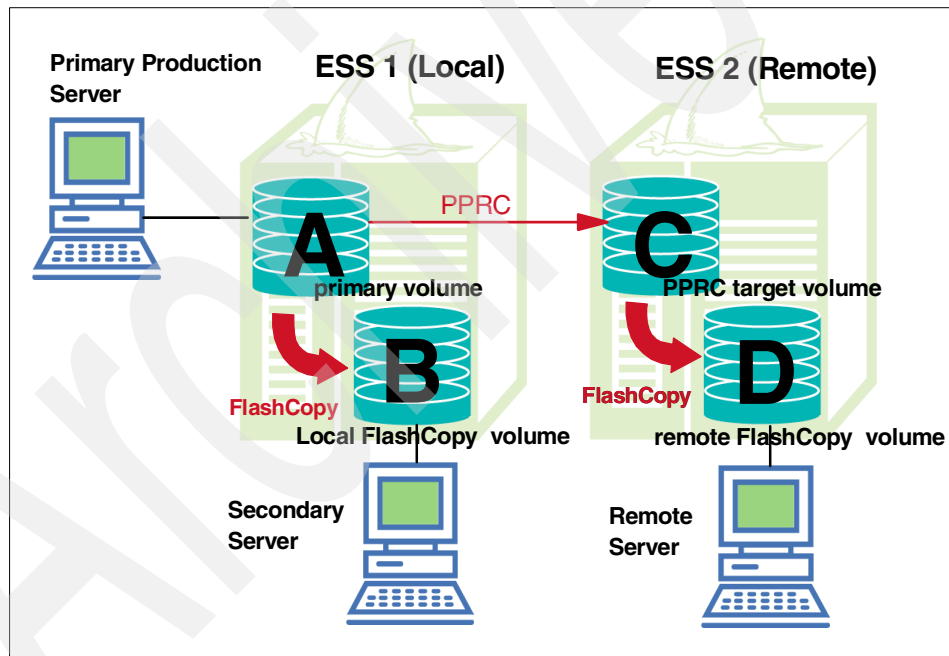


Figure 5-6 PPRC with FlashCopy in local and remote site

## Scenario 5: Online PPRC with FlashCopy

The FlashCopy tasks on ESS1 and ESS2 are usually executed at the same time; these copies can also be individually scheduled at different intervals to allow the customer to recover to different points in time. This allows the maximum level of flexibility for the customer.

Figure 5-6 shows the primary volume (A) is flashcopied to the local mirror volume (B) during the online window, and PPRC is done from the primary volume (A) to the PPRC target volume (C). This PPRC target volume is flashcopied to the remote volume (D). In this scenario, the primary copy (A) and the PPRC target volume (C) are already in a PPRC relationship.

ESS Copy Services CLI command or ESS Specialist Web Interface should be used in each step. The steps involved in performing the online PPRC with FlashCopy are as follows:

1. Quiesce the production database write I/O:  
`db2 set write suspend for database`
2. Perform FlashCopy from the primary volume (A) to the local volume (B) with COPY option.
3. Resume the production write I/O:  
`db2 set write resume for database`
4. Wait until the PPRC between the primary volume (A) and the PPRC mirror volume (C) is completed and the two volumes are in sync.
5. Quiesce the production database write I/O:  
`db2 set write suspend for database`
6. Suspend the PPRC pair between the primary copy(A) and the PPRC synchronous mirror volume (C).
7. Resume the production write I/O:  
`db2 set write resume for database`
8. Perform FlashCopy from the PPRC target volume (C) to the remote volume (D) using rsExecuteTask CLI command with COPY option.
9. Resume and resynchronize all the changed activities in the primary volume (A) during PPRC suspending with the PPRC target volume (C).

Here are optional requirements to take a backup using DB2 in the local or the remote site:

10. Resync the remote server filesystem to make the server aware of new disks.
11. Catalog the database in the local or remote volume (B or D), initialize the database, and take a backup:



- a. **db2 catalog database *database\_alias* on *database\_path***
- b. **db2start**
- c. **db2inidb *database\_alias* as snapshot**
- d. **db2 backup database *database\_alias* to /dev/mnt0**

## 5.3 Using snapshot

The following scenarios illustrate various uses of the flashcopied snapshot mirror images which are copied in “Snapshot backup” on page 94.

The db2inidb tool is necessary in combination with the snapshot feature of ESS Copy Services, to bring the copied database into a running state again after the snapshot was established. The db2inidb tool can perform crash recovery or put a database in a roll forward pending state.

The db2inidb command initializes the snapshot so that it can be used for making a clone database. A read-only clone of the primary database can be used, for example, to create reports, as a standby database and as a backup image. This command can only be issued against the split-off snapshot, and the split-off snapshot must first run db2inidb before it can be used.

These next six scenarios find solutions for several customer requirements:

- ▶ “Scenario 6: Offline FlashCopy Restore” on page 106  
A solution for customers who require a read-only reporting database or user error recovery using *offline* FlashCopy.
- ▶ “Scenario 7: Snapshot initialization” on page 107  
A solution for customers who require read-only reporting database or user error recovery using *online* FlashCopy.
- ▶ “Scenario 8: Version recovery” on page 108  
A solution for customers who require recovery of a corrupted database using offline or online FlashCopy.
- ▶ “Scenario 9: Reverse PPRC recovery” on page 110  
A solution for customers who require recovery of a corrupted database using offline or online FlashCopy taken in a remote site.
- ▶ “Scenario 10: Hot standby database” on page 112  
A solution for customers who require recovery after disaster in the local site using online FlashCopy taken in a remote site.

- “Scenario 11: Mirror database” on page 114

A solution for customers who require recovery of a corrupted database in the local site using online FlashCopy and log files.

A PPRC target image can't be used for DB2 UDB initialization (db2inidb), because DB2 UDB does not know about the PPRC target, but only the flashcopied snapshot.

### 5.3.1 Making a clone database

These scenarios require the secondary server or remote server to attach to the flashcopied snapshot. This snapshot image may be copied with the no copy or copy options.

This clone database is used as an analysis database or development database, and so on. Therefore, this database can offload intensive query/reporting activities without affecting the primary database. This FlashCopy is taken irregularly, whenever needed.

This clone database is also used to take a backup tape or backup database when the primary database has user errors or the database is physically corrupted, but a periodic FlashCopy is needed to keep the recent backup copy.

The FlashCopy snapshot image is attached to the secondary server in a local site or remote site. This snapshot volume flashcopied with the copy option could be flashcopied back to the primary copy and attached to the primary server for version recovery.

#### Scenario 6: Offline FlashCopy Restore

This scenario is based on the FlashCopy taken from “Scenario 1: Offline FlashCopy” on page 96 or “Scenario 3: Offline PPRC with FlashCopy” on page 100. Initializing the flashcopied image is not needed.

This clone database is used for user error recovery. For example, this backup database could be useful for users who accidentally delete or update rows in a table or drop a table or tablespace. A programmer could make a logical error that results in data loss or corruption. Without the primary database restore, these logical errors could be fixed by extracting objects or data from the backup database. This database also is used as an analysis database, such as a reporting database, data mining database, test database, developing database, and so on.

The logical setup for this scenario is shown in Figure 5-7.

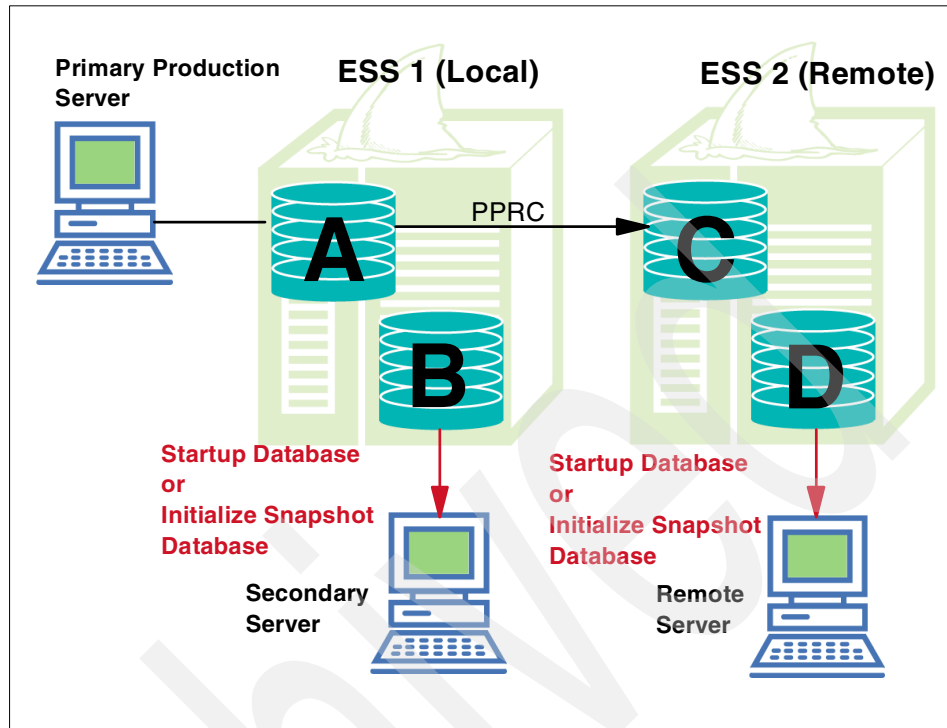


Figure 5-7 Making a clone database

These are the steps for an offline FlashCopy restore:

1. Resync the secondary server filesystem or the remote server filesystem.
2. Catalog the database in the local or the remote mirror volume (B or D), and start the database on the secondary server or remote server:
  - a. `db2 catalog database database_alias on database_path`
  - b. `db2start`
3. Start all applications.

### Scenario 7: Snapshot initialization

This scenario is based on the FlashCopy taken from “Scenario 2: Online FlashCopy” on page 97 or “Scenario 5: Online PPRC with FlashCopy” on page 104. The FlashCopy taken in “Scenario 4: Online PPRC with FlashCopy” on page 101 could be also used for this scenario. Initialization is needed to make the database consistent through crash recovery. A new log chain starts.

The logical setup for this scenario is shown in Figure 5-7 and this database is used such as in “Scenario 6: Offline FlashCopy Restore” on page 106.

These are the steps for the Snapshot initialization:

1. Resync the secondary server filesystem or the remote server filesystem.
2. Catalog the database in the local or the remote volume (B or D), and initialize the database on the secondary server or remote server:
  - a. **db2 catalog database *database\_alias* on *database\_path***
  - b. **db2start**
  - c. **db2inidb *database\_alias* as snapshot**
3. Start all applications.

### **Scenario 8: Version recovery**

Version recovery using FlashCopy is for customers whose production database is physically corrupted or contains logical errors, and they need to restore the primary database to a previous point-in-time from the FlashCopy.

The FlashCopy volume (B) taken in “FlashCopy in local site” on page 95 contains a consistent state in a certain point of time and is used to restore the database. When the primary database crashes or is corrupted, this FlashCopy volume is copied back to the primary volume (A) with data and log files. The copy operation could be done using operating system tools like backup/restore, tar, cp and so on. This scenario uses a reverse flashcopy from the FlashCopy volume (B) to the primary volume (A). If the mirror volume is taken from an online flashcopy, initialization initiates crash recovery.

The local copy (B) is flashcopied back to the primary volume (A). Figure 5-8 shows the logical view of the scenario.

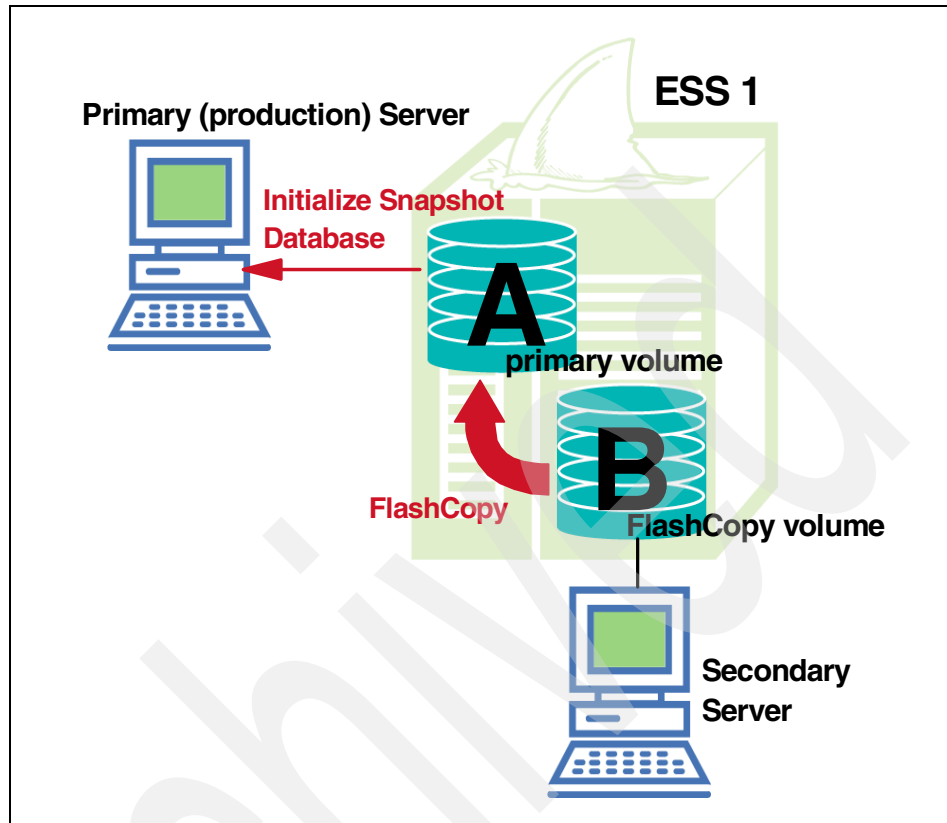


Figure 5-8 Version recovery

The following steps are involved in the version recovery:

1. Stop all applications connected to the production database:  
`db2 force applications`
2. Stop production database:  
`db2stop`
3. Flush file system buffers on the production server and the secondary server.
4. Perform FlashCopy from the local volume (B) back to the primary volume (A) with COPY option.
5. Resync production filesystem.
6. Start production database:  
`db2start`

7. Initialize the snapshot in case of the FlashCopy snapshot (B) taken at online:  
**db2inidb database\_alias as snapshot**
8. Start all applications.

**Tip:** This local volume (B) must be flashcopied with the COPY option to restore the primary database.

### Scenario 9: Reverse PPRC recovery

While the synchronous nature of PPRC protects against disasters, it does not protect against logical errors; the errors are propagated to the PPRC Target on the remote ESS as well. This scenario would protect against logical errors. It would also protect against rolling disasters. The FlashCopy taken at the remote site is a known consistent backup, and can be used to restore the production database faster than possible from tape.

Optionally, if the log files are not damaged and contain no logical errors, the user can use them to roll forward to the point of failure.

In the situation where the production database is corrupted or the production ESS is lost (due to natural disasters, and so on), we can use the backup taken in “PPRC with FlashCopy in remote site” on page 99, or “PPRC with FlashCopy in local and remote site” on page 103 to restore the production database on the ESS1. Figure 5-9 shows the logical view of the scenario.

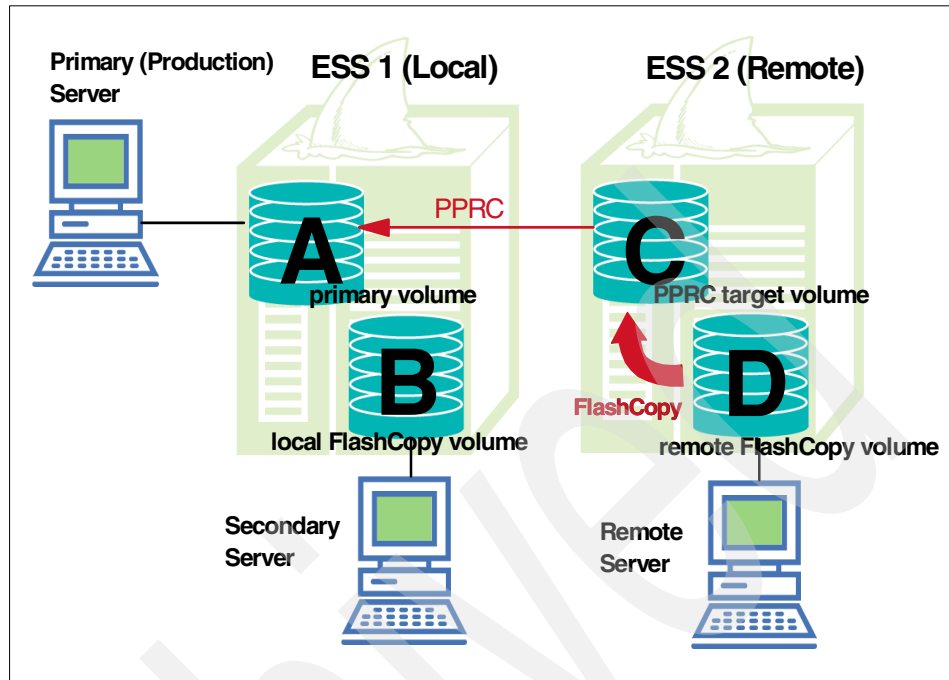


Figure 5-9 Reverse PPRC recovery

1. Terminate the PPRC pair from the primary volume (A) and the PPRC target volume (C).
2. Stop all applications connected to the production database:  
`db2 force applications all`
3. Stop the production database:  
`db2stop`
4. Flush file system buffers on the primary server and remote server.
5. Perform FlashCopy from the mirror volume (D) back to the PPRC target volume (C) at ESS2 with the COPY option.
6. Establish a PPRC pair from the PPRC target volume (C) to the primary volume (A).
7. Wait until the PPRC is completed and the two volumes are in sync using `rsQueryComplete` CLI command.
8. Terminate the PPRC pair.
9. Resync the primary server filesystem to make the server aware of new disks.

10. Start the primary database:

a. **db2start**

b. **db2inidb database\_alias as snapshot**

If volume (D) is online flashcopied.

11. Start all applications on the primary server.

### 5.3.2 Making standby database

An active standby system maintains its own copy of the database by receiving completed logs that are forwarded from the primary system and by applying them (rolling them forward) on its copy of the database. An active standby system can be set up to provide a delay so that it applies logs only after a suitable time has elapsed.

This system is mainly used to take over the production system when the system is not available because of fire, earthquake, flood, and so on. The data in active logs and archive logs that is not moved or copied from the production system to the remote system cannot be applied to the standby database. This data will be lost.

This system can be also used to repair failures within the database itself – for example, if the application program writes invalid data. When such a failure occurs, the standby system can be used to repair the problem and then the repaired copy of the database can be moved back to the primary system.

This scenario is for customers that prepare for natural or artificial disasters and require their businesses to keep going.

The hot standby database can be at the local site or the remote site. But for disaster recovery, we recommend a remote hot standby database.

#### Scenario 10: Hot standby database

The standby database initialized from the volume (D) taken at “Scenario 4: Online PPRC with FlashCopy” on page 101 is located in a remote backup facility. “Scenario 2: Online FlashCopy” on page 97 can be also used for local hot standby database in the secondary server.

After the standby database is initialized successfully, the standby database is in a roll forward pending state, waiting for the archived log files from the production database and sequentially applying the log files, leaving it in a hot standby database. For a userexit program in the standby database or production database, NFS or ftp can be used to get the log files from the production database.



Figure 5-10 logically shows the standby database is initialized on the remote server and it accepts the archive log files from the primary database. When the local site is not available due to various disasters, the remote server takes over the production server to continue the customer's business.

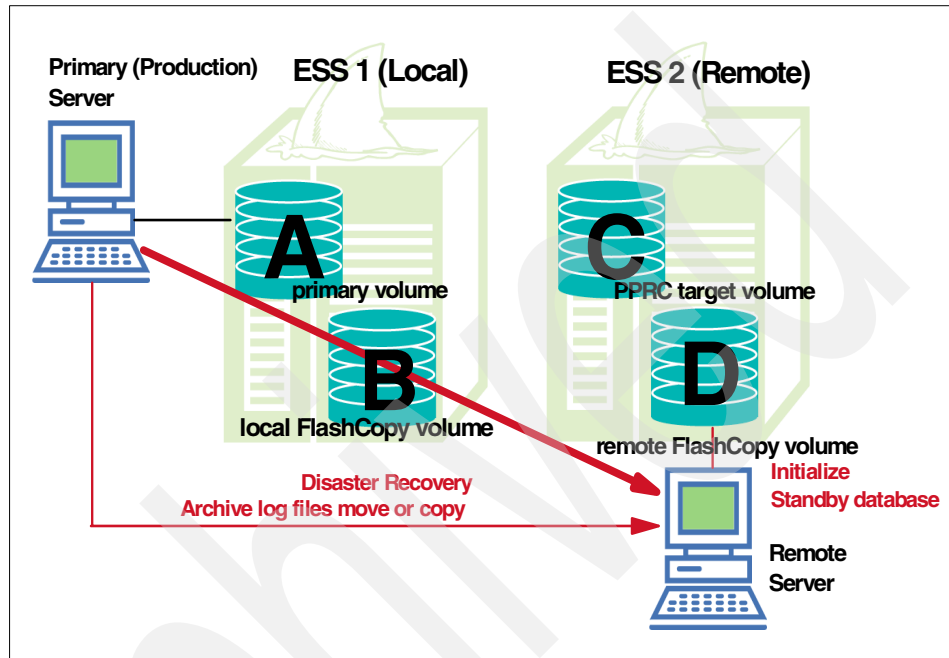


Figure 5-10 Hot standby database

Here are the steps for making a hot standby database:

1. Resync the remote server filesystem to make the server aware of new remote FlashCopy volume (D).
2. Catalog the database in the FlashCopy volume (D) and initialize the database on to the remote server as standby:
  - a. `db2 catalog database database_alias on database_path`
  - b. `db2start`
  - c. `db2inidb database_alias as standby`
3. Copy or move the archive log files from the production database to the standby database.
4. Repeat the roll forward database step whenever new archive log files are available:

`db2 rollforward database database_alias to end of logs`

When the local system crashes, the hot standby database is activated. The hot standby database supports continuous services for users as a production server until the local site recovers from disaster.

The steps for disaster recovery are:

5. Stop the roll forward pending state and activate the hot standby database as new production database:

```
db2 rollforward database database_alias complete
```

6. Start all applications on the remote server.

**Note:** You can restore the corrupted production database using “Scenario 9: Reverse PPRC recovery” on page 110 or the DBMS backup and restore utility after the primary server and primary volume in the local site recover from the disaster.

### 5.3.3 Making a mirror database

While the scenarios in “Making a clone database” on page 106 can be used to make a recovery to the time of the last backup, many customers require a recovery to the moment of failure or any arbitrary point in time. This scenario provides this exact capability, while maintaining extremely fast restores. As long as no transaction logs are damaged, no data is lost.

To recover the primary database as a mirror, don’t initialize the mirror volume on the secondary server.

#### Scenario 11: Mirror database

The mirror database is initialized from the local FlashCopy volume (B) which is the outcome of “Scenario 2: Online FlashCopy” on page 97. When the primary database crashes, this FlashCopy volume B is copied back to the primary volume (A) except for the database log files. The copy operation can be done using operating system tools like backup/restore, tar,cp and so on. This scenario uses reverse flashcopy from the local FlashCopy volume (B) to (A). Initializing the copied database as a mirror places the database in roll forward pending state. The UDB performs roll forward recovery with the old log files as they existed in the production server before the database crash.

Figure 5-11 logically shows the mirror database is flashcopied from local FlashCopy volume (B) back to the primary volume (A) and initialized as a mirror on to the primary server.

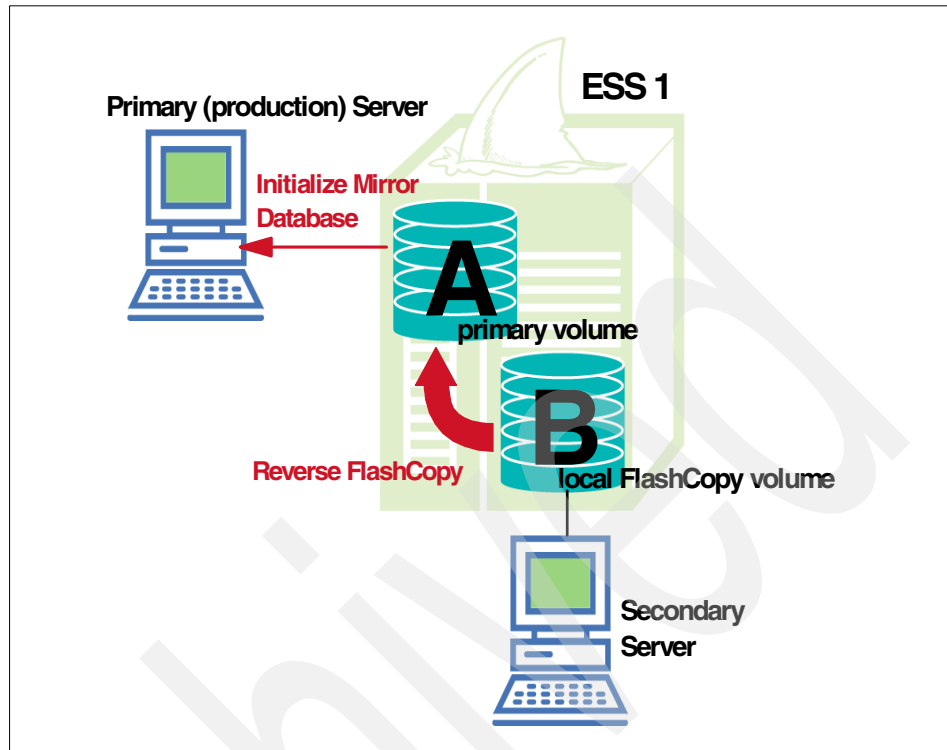


Figure 5-11 Mirror database

Here are the steps for making a mirror database:

1. Flush file system buffers on the primary server.
2. Perform FlashCopy from the local FlashCopy volume (B) back to the primary volume (A) only for data files, not for log files, with the COPY option.
3. Resync the primary server filesystem to make the server aware of changed primary volume (A).
4. Initialize the copied database on to the primary server as a mirror and place the database in roll forward pending state:
  - a. **db2start**
  - b. **db2inidb database\_alias as mirror**
5. Perform a roll forward of the database to end of log or specific point in time:
 

```
db2 rollforward database database_alias to end of logs and stop
```
6. Start all applications on the primary server.

**Note:** The local volume(B) should not be initialized on the secondary server to use this volume as a mirror database on the primary server.

# Related publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this redbook.

## IBM Redbooks

For information on ordering these publications, see “How to get IBM Redbooks” on page 119.

- ▶ *IBM ESS and IBM DB2 UDB Working Together*, SG24-6262
- ▶ *Implementing the Enterprise Storage Server in Your Environment*, SG24-5420
- ▶ *IBM Enterprise Storage Server*, SG24-5465
- ▶ *Implementing ESS Copy Services on UNIX and Windows NT/2000*, SG24-5757
- ▶ *AIX 5L Differences Guide Version 5.1 Edition*, SG24-5765
- ▶ *DB2 UDB V7.1 Performance Tuning Guide*, SG24-6012
- ▶ *IBM StorWatch Enterprise Storage Server Expert Hands-On Usage Guide*, SG24-6102
- ▶ *Implementing Fibre Channel Attachment on the ESS*, SG24-6113
- ▶ *Backing Up DB2 using Tivoli Storage Manager*, SG24-6247

## Other resources

These publications are also relevant as further information sources:

- ▶ *IBM TotalStorage Redbooks Collection*, SK3T-3694-05
- ▶ *IBM Redbooks Data Management Collection*, SK2T- 8038-08
- ▶ *IBM Redbooks RS/6000 Collection*, SK2T-8043-05
- ▶ *DB2 Administration Guide: Performance Version 7*, SC09-2945

## Referenced Web sites

These Web sites are also relevant as further information sources:

### **ESS**

- ▶ ESS Introduction and Planning Guide; also ESS User's Guide  
<http://www.storage.ibm.com/hardsoft/products/ess/refinfo.htm>
- ▶ IBM Subsystem Device Driver/Data Path Optimizer on an ESS, sddpo-v120.doc  
<http://SSDDOM01.storage.ibm.com/techsup/swtechsup.nsf/support/sddl link>
- ▶ ESS Performance White Paper; also Split Mirror Backup / Recovery with IBM's Enterprise Storage Server (ESS) Paper  
<http://www.storage.ibm.com/hardsoft/products/ess/whitepaper.htm>
- ▶ DB2 UDB EEE scalability on RS/6000 S80 and Enterprise Storage Server  
<http://www-4.ibm.com/software/data/bi/teraplex/index.htm>
- ▶ IBM Enterprise Storage Server  
<http://www.ibm.com/storage/ess>
- ▶ IBM List of Supported Servers  
<http://www.storage.ibm.com/hardsoft/products/ess/supserver.htm>
- ▶ IBM Storage Technical Support, including SDD and StorWatch  
<http://www.storage.ibm.com/hardsoft/disk/techsup.htm>
- ▶ IBM ESS Software Technical Support  
<http://ssddom02.storage.ibm.com/techsup/webnav.nsf/support/sdd>

### **DB2**

- ▶ IBM DB2 UDB home page  
<http://www-4.ibm.com/software/data/db2/>
- ▶ IBM manuals for Data Management products  
<http://www-4.ibm.com/software/data/db2/library/>
- ▶ IBM DB2 online support  
<http://www-4.ibm.com/cgi-bin/db2www/data/db2/udb/winos2unix/support/index.d2w/report>
- ▶ Database and Data Management White Papers  
<http://www-4.ibm.com/software/data/pubs/papers/index.html#udbaix>

## **AIX**

- ▶ IBM AIX home page  
<http://www-1.ibm.com/servers/aix/>
- ▶ AIX Documentation Library. Use the search function to search for AIX commands such as: iostat, vmstat, filemon, smit.  
[http://www.rs6000.ibm.com/cgi-bin/ds\\_form](http://www.rs6000.ibm.com/cgi-bin/ds_form)

## **How to get IBM Redbooks**

You can order hardcopy Redbooks, as well as view, download, or search for Redbooks at the following Web site:

[ibm.com/redbooks](http://ibm.com/redbooks)

You can also download additional materials (code samples or diskette/CD-ROM images) from that site.

## **IBM Redbooks collections**

Redbooks are also available on CD-ROMs. Click the CD-ROMs button on the Redbooks Web site for information about all the CD-ROMs offered, as well as updates and formats.





# Index

## A

- an instantaneous copy 68
- application ESS 75
- application site 75
- ARCHIVE\_PATH 63
- automation 73

## B

- binary large object 29
- BLOB 29
- block I/O 34
- buffer pools 23

## C

- CE 70
- character large object 29
- CIFS 33
- CLI 70, 72
- CLOB 29
- clustering 37
- coherency control 38
- column 28
- Command Line Interface 72
- Common Internet File System 33
- Concurrent Copy 37
- configuration planning 71, 80
- consistent backup 110
- containers 25
- crash recovery 48
- CRIT (NO) 80
- CRIT (YES) 80
- Customer Engineer 70

## D

- DAS 34
- data backup 37
- data consistency 81
  - considerations 71
- data pipe 37
- data protection 37
- data sharing 37
- data vaulting 37

- database managed space 24
- database roll forward recovery 50
- databases 22
- data-copy sharing 37
- DB2 UDB and storage configuration 90
- DB2\_PARALLEL\_IO 30–31
- DB2\_STRIPED\_CONTAINERS 30–31
- db2ckbcp 47
- db2dart utility 48
- DB2EMPFA 32
- DB2SET 30
- db2start 30
- db2stop 30
- DBCLOB 29
- Dense Wavelength Division Multiplexer 82
- DEVICE 25
- direct access storage 34
- disaster recovery 37
- disaster recovery plan 80–81
- DMS 24
- do not perform background copy 69
- double-byte character large object 29
- DRP 80–81
- DWDM 82

## E

- environment variables 30
- ESCON 82
- ESS 82
- ESS Specialist Copy Services Specialist 72
- Extended Distance Feature 82
- Extended Remote Copy 37
- extent size 27

## F

- Fibre Channel 36
- Fibre Channel LUNs 70
- FICON 82
- field 28
- FILE 25
- file I/O 34
- FlashCopy 38, 68
- flat file transfer 37

full copy mode 77

## G

group 33

## H

HA 37

HACMP 71

hardware and software requirements 70

high availability 37

## I

I/O parallelism 29–30

IBM 2029 Fiber Saver 82

IBM 9729 Optical Wavelength Division Multiplexer 82

incremental backup 54

incremental delta 55

incremental delta option 55

incremental recovery 55

Independent Software Vendor 70

indexes 28

initializing database 61

instances 22

International Standards Organization 35

inter-query parallelism 30

intra-query parallelism 30

ISO 35

ISV 70

## L

LAN 35

LAN-less backup 37

list history backup 47

Local Area Networks 35

local FlashCopy volume 90

logical control units 82

logical subsystem 69, 71

long data 29

LONG VARCHAR 29

LONG VARGRAPHIC 29

lost writes 81

LSS 69, 71, 82

## M

Microsoft's Server Message Block 33

mirror 61

mount 33

MuxMaster 82

MuxMaster units 82

## N

NAS 33

Network File System 33

protocols 33

Network Lock Manage 33

NFS 33

NOCOPY 69

## O

Open Systems Interconnection 35

operational considerations 72

optimized PPRC communication 86

OSI 35

## P

page cleaners 23

page size 26

parallelism 29

PATH 25

Peer-to-Peer Remote Copy 75

performance considerations 73, 86

pipng 37

planning for ESS Copy Services 67

planning for FlashCopy on ESS 70

point-in-time copy 68

PPRC 75

PPRC target volume 90

PPRC volume states 76

prefetch size 27

primary ESS 75

primary site 75

primary volume 90

## Q

query parallelism 29–30

## R

record 28

recovery ESS 75

recovery feature 47

recovery site 75

Redbooks Web site 119

Contact us xiii

- referential constraints 28
- registry 30
- remote cluster storage 37
- remote copy 37
- remote FlashCopy volume 90
- resource planning 71, 80
- restore 49
- RETRIEVE\_PATH 63
- row 28
- rsExecuteTask command 72
- rsQueryComplete CLI command 101

## S

- SAN applications 36
- SAN fabric 36
- SAN storage 36
- SCSI target IDs 70
- secondary ESS 75
- secondary site 75
- serialization 38
- server-free backup 37
- shared repository 37
- SMB 33
- SMS 24
- Snapshot 61
- SnapShot Copy 38
- Snapshot infrastructure 93
- standby 61
- Storage Area Network 36
- storage mirroring 37
- Subsystem Device Driver 71, 93
- subsystem local copy services 38
- system managed space 24

## T

- T0 68
- T0 (time-zero) copy 68
- T0 copy 68
- table spaces 24
- tables 28
- test plan 81
- test plan and disaster recovery plan 72
- test setup and configuration 90
- TRACKMOD 55

## U

- user 33

- using a FlashCopy target volume 72
- using PPRC with FlashCopy 83

## V

- value 28
- varying-length double-byte long character string 29
- varying-length long character string 29
- version recovery 48

## W

- WAN 35
- warm standby techniques 37
- Wide Area Network 35

## X

- XDF 82
- XRC 37



## DB2 UDB Backup and Recovery with ESS Copy Services

(0.2" spine)  
0.17" <-> 0.473"  
90 <-> 249 pages







**Redbooks**

# DB2 UDB Backup and Recovery

## with ESS Copy Services

**State-of-the-art  
backup and recovery  
services from ESS**

**Advanced backup  
and recovery  
features from DB2  
UDB**

**Make Copy Services  
part of your backup  
strategy**

This IBM Redbook describes how customers can use the IBM TotalStorage Enterprise Storage Server's Advanced Copy Services to perform a snapshot backup and recovery in an IBM DB2 UDB Windows environment. Of these Copy Services, FlashCopy, is capable of performing backups of multi-terabyte databases in minutes. It can also help improve uptime by providing the ability to recover in minutes. Another Copy Service, Peer-to-Peer Remote Copy (PPRC), provides a synchronous remote mirror to protect against disasters. When used in combination, these two Copy Services have the ability to provide multiple instant copies of the database and still protect against disaster.

IBM DB2 UDB is a universal database supporting a wide variety of platforms and applications. With every new release, DB2 UDB continues to grow in the enterprise-class server segment. When a database is used for mission-critical applications, it requires an enterprise-class storage subsystem. The Enterprise Storage Server (ESS) can provide the reliability, scalability, and features required for mission-critical applications. This redbook also provides general information about ESS and DB2 UDB.

**INTERNATIONAL  
TECHNICAL  
SUPPORT  
ORGANIZATION**

**BUILDING TECHNICAL  
INFORMATION BASED ON  
PRACTICAL EXPERIENCE**

IBM Redbooks are developed by the IBM International Technical Support Organization. Experts from IBM, Customers and Partners from around the world create timely technical information based on realistic scenarios. Specific recommendations are provided to help you implement IT solutions more effectively in your environment.

**For more information:**  
[ibm.com/redbooks](http://ibm.com/redbooks)