

---

# Managing SAP with DB2 on OS/390 Using EMC Symmetrix Storage Systems

Date 2/18/2002

---

Copyright © 2001, 2002 EMC Corporation. All rights reserved.

EMC believes the information in this publication is accurate as of its publication date. The information is subject to change without notice.

THE INFORMATION IN THIS PUBLICATION IS PROVIDED “AS IS.” EMC CORPORATION MAKES NO REPRESENTATIONS OR WARRANTIES OF ANY KIND WITH RESPECT TO THE INFORMATION IN THIS PUBLICATION, AND SPECIFICALLY DISCLAIMS IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

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

EMC, EMC<sup>2</sup>, and Symmetrix are registered trademarks and SRDF, TimeFinder, and where information lives are trademarks of EMC Corporation.

SAP is a trademark of SAP AG in Germany and in several other countries all over the world.

All other brand names are trademarks or registered trademarks of their respective owners.

Part Number H509

Printed 2/18/2002

## Table of Contents

<b>Preface .....</b>	<b>5</b>
<b>1 SAP Environments on OS/390 .....</b>	<b>6</b>
1.1.1 DB2 Use in an SAP Environment .....	6
1.1.2 Storage Management .....	7
1.1.3 Cloning SAP Environments.....	7
1.1.4 Backup and Recovery.....	7
1.1.5 Disaster Recovery Versus Disaster Restart.....	7
<b>2 EMC Products to Support SAP Environments .....</b>	<b>8</b>
2.1 EMC Symmetrix.....	8
2.2 EMC TimeFinder .....	9
2.3 EMC Symmetrix Remote Data Facility.....	10
2.4 EMC Consistency Groups (ConGroups).....	11
<b>3 Cloning SAP Systems .....</b>	<b>13</b>
3.1 Cloning the SAP Environment Within the Same LPAR .....	13
3.1.1 Define Target DB2 Subsystem.....	14
3.1.2 Define ICF User Catalogs .....	14
3.1.3 Create DB2 Procedures .....	14
3.1.4 Create Target DSNZPARM.....	15
3.1.5 Cloning the Datasets.....	15
3.1.6 Renaming Cloned Datasets.....	15
3.1.7 Updating the BSDS.....	17
3.1.8 Starting the Target DB2 Instance.....	18
3.1.9 Changing the HLQ of User-Defined Indexes on the DB2 Catalog.....	18
3.1.10 Changing HLQ of Work File Database (DSNDB07).....	19
3.1.11 Altering User Objects to Use the New HLQ .....	20
3.1.12 Performance Recommendations for the Rename Process.....	21
3.2 Cloning SAP to Another LPAR .....	22
3.2.1 Define Target DB2 Subsystem.....	22
3.2.2 Create DB2 Procedures .....	22
3.2.3 Create Target DSNZPARM.....	22
3.2.4 Cloning the Datasets.....	22
3.2.5 Associating Source Data to the Target Environment.....	23
3.2.6 Starting the Target DB2 Instance.....	23
3.2.7 Work Database Considerations for Data Sharing .....	24
3.3 Recoverability of the Cloned Target SAP and DB2 Instance .....	25
<b>4 Backing Up SAP Environments .....</b>	<b>26</b>
4.1 Creating a Split Mirror Backup Environment .....	26
4.1.1 Split Mirror Backup Preparation .....	27
4.1.2 Creating a Split Mirror Backup Using DB2 Suspend and Resume.....	28
4.1.3 Creating a Split Mirror Backup Using Consistent Split.....	29
4.1.4 Creating a Split Mirror Backup Using ConGroups.....	30
4.1.5 Split Mirror Backup Method Comparisons .....	31
4.2 Creating Multiple Split Mirrors .....	31
4.3 Backing Up Split Mirror Environments .....	32
4.3.1 Backup Using DFDSS .....	32
4.3.2 Backup Using FDR.....	33
4.4 Keeping Track of Dataset Placement on Backup Volumes .....	33

---

4.5	DB2 Image Copy Considerations.....	34
4.5.1	Considerations for 32 KB Pages.....	34
4.5.2	SYSLGRNX and SYSCOPY Recovery Considerations .....	34
4.6	Operations With LOG NO Considerations.....	34
<b>5</b>	<b>SAP Recovery Procedures.....</b>	<b>35</b>
5.1	TimeFinder Restore and Parallel Recovery .....	36
5.2	Restoring SAP Using DFDSS.....	36
5.2.1	Full DFDSS Database Restore .....	36
5.2.2	Logical Dataset Restore Using DFDSS .....	36
5.3	Restoring SAP Using FDR.....	37
5.3.1	Full FDR Database Restore.....	37
5.3.2	Logical Dataset Restore Using FDR.....	37
5.4	Optimizing SAP Recovery.....	38
	<b>Appendix A.....</b>	<b>40</b>
	<b>Appendix B.....</b>	<b>43</b>
	<b>References .....</b>	<b>44</b>

## Preface

This document describes how the EMC Symmetrix<sup>®</sup> system manages SAP environments using DB2<sup>®</sup> UDB on OS/390. The intended audience is SAP<sup>™</sup> Administrators, DB2 Systems Administrators, DB2 Database Administrators and storage management personnel responsible for managing SAP systems on OS/390. The information in this document is based on DB2 UDB Version 6.1 and SAP Version 4.6B on OS/390.

This document provides a general description of SAP environments using DB2 UDB on OS/390. It also provides a general description of EMC products and utilities that can be used for SAP and DB2 UDB administration on OS/390. EMC Symmetrix products and utilities can be used to clone SAP environments and to enhance database and storage management backup and recovery procedures. Using EMC products and utilities to manage SAP environments can reduce database and storage management administration, reduce CPU resource consumption, and reduce the time required to clone or recover SAP systems.

This document is divided into the following five sections. Section topics range from general SAP and EMC product descriptions in sections 1 and 2 to detailed discussions of implementation procedures in sections 3, 4, and 5. Sections 3, 4, and 5 include detailed examples and sample Job Control Language (JCL) used to implement the examples.

Section 1, *SAP Environments on OS/390*, provides a high-level overview of the SAP application implemented using DB2 on OS/390.

Section 2, *EMC Products to Support SAP Environments*, describes EMC products used to support the management of SAP environments.

Section 3, *Cloning SAP Systems*, describes procedures to clone SAP systems within and across Logical Partitions (LPARs).

Section 4, *Backing Up SAP Environments*, describes how to back up SAP environments.

Section 5, *SAP Recovery Procedures*, describes how to recover SAP environments.

The appendix provides sample code, which supplements some of the procedures.

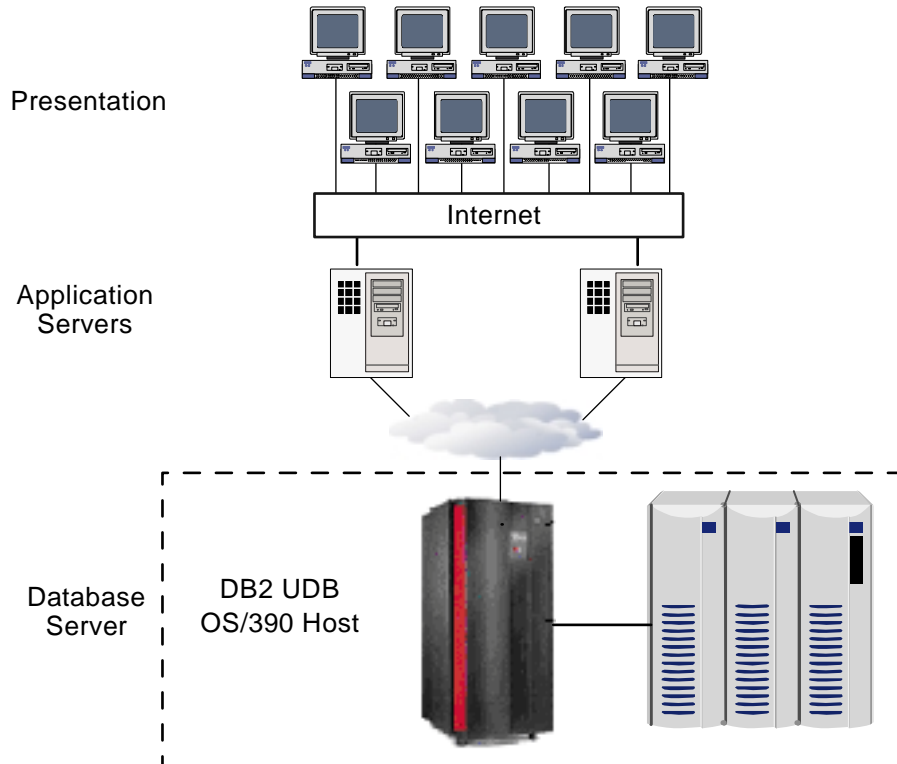
The references section lists documents that contain more information on these topics.

Examples provided in this document cover methods for performing various DB2 functions using a Symmetrix system with EMC software. These examples were developed for laboratory testing and may need tailoring to suit other operational environments. Any procedures outlined in this document should be thoroughly tested prior to production implementation.

# 1 SAP Environments on OS/390

The System 390 processing environment provides a highly scalable and highly available processing platform. Parallel SYSPLEX implementations provide high scalability and availability using DB2 data sharing capabilities. The combination of Parallel SYSPLEX and DB2 data sharing functions make it a preferred choice for providing database server functions for large SAP implementations.

SAP uses a multitier implementation approach, where client processes interact with application servers. Application servers in turn, interact with the database server. All data is stored on the database server. Figure 1 shows an SAP-processing environment using DB2 on OS/390 as the database server.



**Figure 1. High-Level SAP Environment**

Characteristics of an SAP environment that set it apart from a typical DB2 application are:

- **Large number of DB2 objects** — Recent SAP versions can have over 7,000 tablespaces and 19,000 indexes. When combined with the DB2 catalog objects, the total number of objects to be managed within the DB2 instance can be over 40,000.
- **Referential integrity is maintained within the SAP application** — To perform a point in time recovery, the entire DB2 instance must be recovered to a point of consistency. There is no logical division of data that is normally present in most DB2 applications.
- **Dynamic object drop and creation complicate recovery** — The number of objects present in an SAP DB2 instance at any one given time can vary. Standard DB2 recovery procedures are difficult to use in an SAP environment.

## 1.1.1 DB2 Use in an SAP Environment

SAP on OS/390 typically uses DB2 as the database server. DB2 can manage large amounts of data, delivering high performance and high-levels of availability, integrity and security. These features of DB2 provide an environment that can handle SAP's complex application design and the large number of database objects. It is recommended that SAP run on a dedicated DB2 instance. The dedicated DB2

instance is a fundamental requirement to providing storage processor support for managing large SAP environments using DB2 on OS/390.

### 1.1.2 Storage Management

Standard DB2 backup, recovery, and cloning methods are difficult to manage and time-consuming in an SAP environment. This is due to the large number of DB2 objects that SAP introduces into the DB2 instance, which need to be backed up and recovered together. EMC provides alternative solutions to traditional DB2 utilities for cloning SAP systems, providing backup and recovery and providing disaster recovery or business continuance.

### 1.1.3 Cloning SAP Environments

EMC technology enables creation of an instant point-in-time copy of an SAP system. The cloned copy is an identical environment to its source, and can be used for other processing purposes such as backup, recovery, off-line reporting, and testing.

### 1.1.4 Backup and Recovery

Backup and recovery operations using DB2 utilities require intervention by experienced personnel and can be labor-intensive in a large DB2 instance. Recovery of SAP systems is especially complex because the entire SAP system is basically a large referential set. All data in the set needs to be recovered together and to the same recovery point. Dynamic manipulation of objects and application-maintained referential integrity further complicates recovery efforts. Traditional DB2 recovery techniques require multiple passes of the data, which can greatly impact recovery times. Such techniques are generally unworkable in an SAP environment due to the time required to recover all objects. EMC hardware and software can be used to make the process faster and more effective.

### 1.1.5 Disaster Recovery Versus Disaster Restart

It is possible to create a backup environment that can be restarted in the event of a disaster. Restarting this environment instead of recovering it significantly reduces recovery time. The time required to do a traditional DB2 recovery is significantly greater than using EMC technology to restart a DB2 instance and the SAP application at the recovery site.

## 2 EMC Products to Support SAP Environments

EMC provides many hardware and software products that support SAP environments. These products include:

- **EMC Symmetrix** — Symmetrix is a high-performance, integrated cached disk array. It is a fully redundant, high availability storage processor providing non-disruptive component replacements and code upgrades. Symmetrix features high levels of performance, data integrity, reliability, and availability.
- **EMC TimeFinder™** — TimeFinder is a business continuance solution that allows customers to use special devices, called business continuance volumes (BCVs), to create a mirror image of Symmetrix standard devices. These special mirrored devices can be attached to the same or different hosts or LPARs when they are not associated with their standard devices. The standard Symmetrix devices remain online for regular I/O operation while the mirrors are created.
- **EMC Symmetrix Remote Data Facility (SRDF™)** — SRDF is a business continuance solution that allows specific source Symmetrix volume(s) to be mirrored to remote target Symmetrix volume(s).
- **EMC ConGroups** — ConGroups maintain integrity within a database that is spread across multiple Symmetrix units in an SRDF configuration.

### 2.1 EMC Symmetrix

EMC Symmetrix, with its business continuance features and advanced software capability, has led the way in providing real solutions to the difficult problem of protecting mission-critical data. Symmetrix business continuance features improve a wide range of processes, from scheduled backup to disaster recovery while providing high performance access to data.

Symmetrix uses RAID 1 mirroring technology to provide high levels of data availability. In a RAID 1 protection scheme identical data is stored on a redundant mirror. Should either mirror fail, the other mirror provides continuous availability transparent to the host application.

In mirrored configurations, Symmetrix can choose which mirror should service any read operation. A special optimization algorithm has been developed to take full advantage of this. The algorithm is called Dynamic Mirror Service Policy (DMSP). The DMSP algorithm optimizes performance in two ways:

- DMSP balances the load among the physical drives, so that even when the workload is skewed, the loads on the physical drives are as balanced as possible. For example, when a LUN is very busy, the DMSP will use both mirrors to service the read operations of this LUN.
- DMSP minimizes the seek operations on the disks. Whenever the load balancing allows it, each disk reads from one half of its platters: either from the outermost cylinders, or from the innermost cylinders. Because of this, the seek distances are significantly shortened. Seek minimization is a critical optimization because, as disk capacity increased over the years, the data transfer time was improved much more significantly than seek time.

The Symmetrix features a “cache all” architecture. All data flows through cache and all writes are “fast writes.” There is no concept of a “write miss” as in other control unit architectures. The cache management algorithms manage the cache effectively, through techniques such as sequential prefetch and multiple LRU structures, so as to ensuring very high read hit ratios.

In a database environment, the best analogy to the Symmetrix cache is the database bufferpools. The Symmetrix cache behaves like an extension to the database bufferpools, increasing performance for both read and write functions. All data is cached to improve performance. Database systems from all vendors benefit from performance improvements provided by the Symmetrix.

## 2.2 EMC TimeFinder

TimeFinder is a business continuance solution that uses special volumes called Business Continuance Volumes or BCV devices. BCVs are Symmetrix devices that are specially configured in a Symmetrix system to be used as dynamic mirrors. Each BCV device has its own host address and is configured as a stand-alone device.

Once BCVs are associated (established) with standard devices using the `ESTABLISH` command, data is automatically synchronized from the standard device to its associated BCV.

While established, the BCVs are off line to the host. The standard devices remain attached to the host(s) and are on line for I/O operation. BCVs contain a mirrored copy of the standard device that can be used for backup, restore, decision support, and application testing when disassociated, or `SPLIT`, from the standard device.

A business continuance sequence involves establishing the BCV device as an additional mirror of a standard Symmetrix device. Once the BCV is established as a mirror of the standard device, it is not accessible through its original device address. The BCV device may later be `SPLIT` from the standard Symmetrix device with which it was previously paired. The BCV device now has valid data identical to that on the standard device at the time of the `SPLIT`. The BCV can be made available for backup or other processes through its original device address. Once host processing on the BCV device is complete, the BCV may again be re-associated (`RE-ESTABLISHED`) to the same device or to a different device.

Four basic commands are used to manage Business Continuance processes. They are:

- **ESTABLISH** — Associates a BCV with a standard volume and synchronizes the BCV with the standard volume. During this process, the BCV disappears from the host channel and joins the standard volume as an additional (typically the third) mirror. The BCV must be off line to OS/390 to execute an `ESTABLISH`.
- **SPLIT** — Disassociates a BCV from its standard device and makes it available on a host channel as another logical volume. During the period when a device is `SPLIT` the Symmetrix keeps track of changes that are made to both the standard device and the BCV. The BCV can be made available to the host when it is `SPLIT`. The changed track information is used when a `RE-ESTABLISH` or a `RESTORE` operation is requested. The TimeFinder Utility (TFU) allows a relabel of the volumes and rename of the datasets that are contained on the BCVs.
- **RE-ESTABLISH** — Re-associates and resynchronizes the original standard volume to the BCV by copying only the changed tracks from the standard to the BCV rather than the entire volume. When the BCV is re-established, tracks that have changed on the standard device will be written to the BCV. If changes were made to the BCV prior to the `RE-ESTABLISH`, those tracks will also be overlaid. The BCV must be off line to OS/390 to perform the `RE-ESTABLISH`.
- **RESTORE** — Restores from the BCV the tracks that changed on the standard devices since the `SPLIT`. If changes have been made to the BCV, they will also be propagated back to the standard device with the `RESTORE`. A `RESTORE` operation is potentially destructive to data on the standard device. Use caution when restoring data from a BCV.

Refer to Figure 2 for an overview of the above commands.

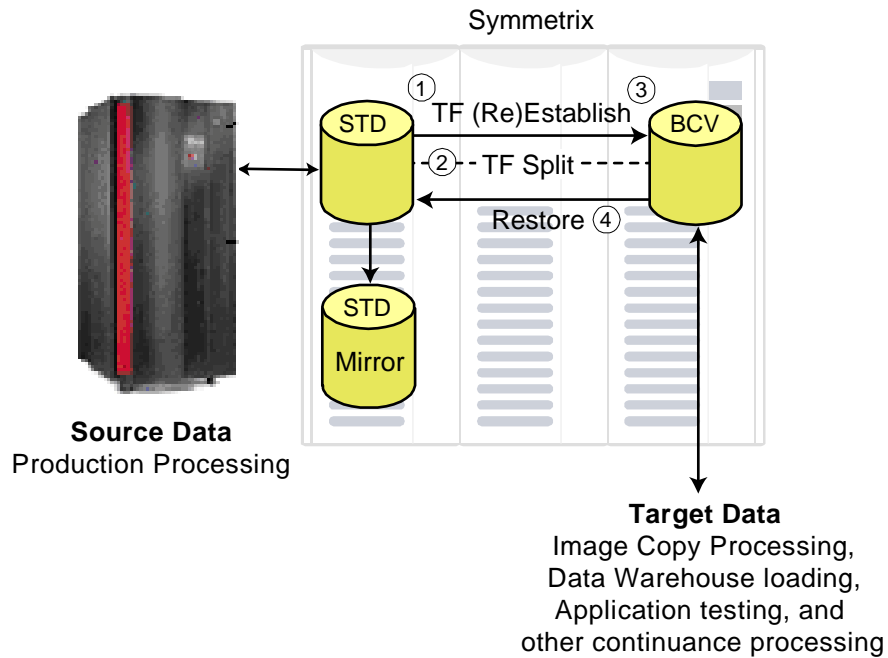


Figure 2. EMC Symmetrix Configured With BCVs and Mirrored Standard Volumes

### 2.3 EMC Symmetrix Remote Data Facility

The Symmetrix Remote Data Facility (SRDF) is a business continuance solution that allows specific source Symmetrix volumes (R1) to be mirrored to remote target Symmetrix volumes (R2). Data mirroring can use Synchronous and Adaptive Copy modes of operations in campus or extended distance. Figure 3 shows an example of an SRDF configuration with a remote host.

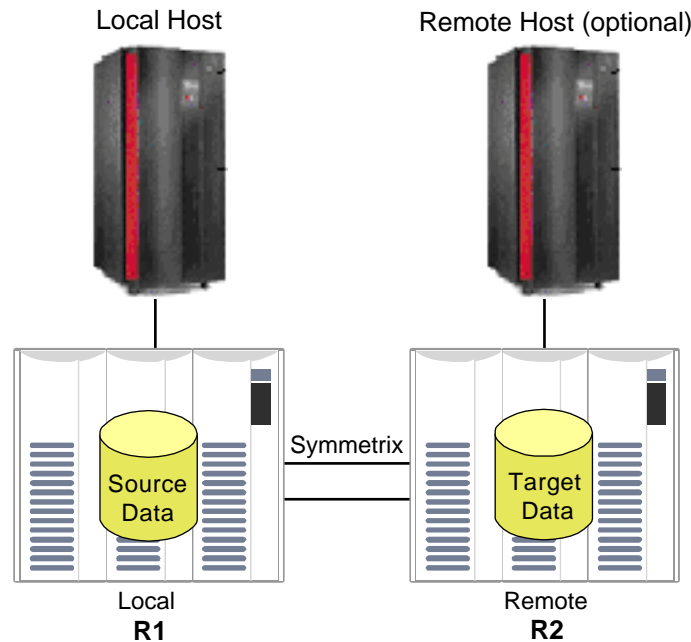


Figure 3. Symmetrix Remote Data Facility Configured With Remote Host

## 2.4 EMC Consistency Groups (ConGroups)

An EMC ConGroup maintains consistency within a database spread across multiple Symmetrix units in an SRDF configuration, by monitoring data propagation from the source (R1) devices in a ConGroup to their corresponding target (R2) devices. ConGroup provides data integrity protection during rolling disasters.

Most applications, and in particular database management systems (DBMS), have dependent I/O logic imbedded in them to ensure data integrity in the event a failure occurs in the host processor, software, or storage subsystem. A dependent write is a write that will not be issued by the application unless some prior I/O has completed.

In a remote disk copy environment, data consistency cannot be ensured if one of these dependent I/Os was remotely mirrored, but its predecessor was not remotely mirrored. This could occur, for example, in a rolling disaster where there is a communication loss that affects only a subset of the disk controllers that are performing the remote copy function.

The ConGroup prevents the predecessor and dependent I/O inconsistency from occurring by intercepting any I/O to a volume that cannot communicate to its remote mirror. It then suspends the remote mirroring for all volumes defined to the ConGroup before completing the intercepted I/O and returning control to the application. In this way, the ConGroup prevents dependent I/O from being issued by the application, thus ensuring the integrity and consistency of the data at the remote site. I/O to the local ConGroup devices can still occur when the SRDF R2 devices are suspended. While these updates are not immediately sent to the remote side, through usual SRDF operations, they are propagated once the affected links are again operational and the ConGroup is resumed.

The following steps refer to Figure 4 on page 12 and describe a series of events that define a ConGroup, execute the ConGroup during a rolling disaster, and recover from the rolling disaster. Circled numbers in Figure 4 map to the steps listed below.

1. A ConGroup is defined to the Consistency Group Task (CGT) using its configuration file interface. Volumes R1(A), R1(C) and R1(E) are defined to a named ConGroup. The ConGroup is then enabled.
2. Normal production processing proceeds with all source Symmetrix data mirrored using SRDF to their remote partners.
3. The last SRDF link in an RA group breaks. The break could indicate the beginning of a rolling disaster. The first write I/O to discover the break is “unit checked” and CGT immediately but temporarily freezes all I/O to all source devices defined in the ConGroup containing the unit-checked disk. The unit-checked I/O does not complete to the application, preventing dependent I/Os from being issued. Freezing I/O in this manner ensures data integrity preservation at the recovery site. Applications writing to non-ConGroup defined devices will not cause a ConGroup invocation and thus will not suspend SRDF relationships.
4. While I/O is temporarily halted to the devices in the ConGroup, CGT issues commands through the source Symmetrix to suspend SRDF relationships for all disk devices defined in the ConGroup. The SRDF suspension requires two I/Os per source Symmetrix to suspend all necessary (source-to-target) mirroring relationships. The length of time I/O is suspended is a function of the number of Symmetrix contained in the ConGroup.
5. Once SRDF relationships have been suspended and data can no longer be propagated for the devices specified in the ConGroup, then I/O is allowed to flow again to the source devices. The time required to perform ConGroup processing is a few milliseconds, and may show up as an elongated I/O for in-flight transactions executing during the ConGroup invocation.
6. Data on remote Symmetrix units S4 and S5 volumes R2(A), R2(C), and R2(E) represent the data on the source Symmetrix units at the instant of the link failure in step 3. From an application and database perspective, the state of the remote data is as if a “system crash” had happened on the production

environment using only the source Symmetrix. ConGroups maintain data integrity on the remote R2 devices such that databases and applications can be restarted using normal restart processing.

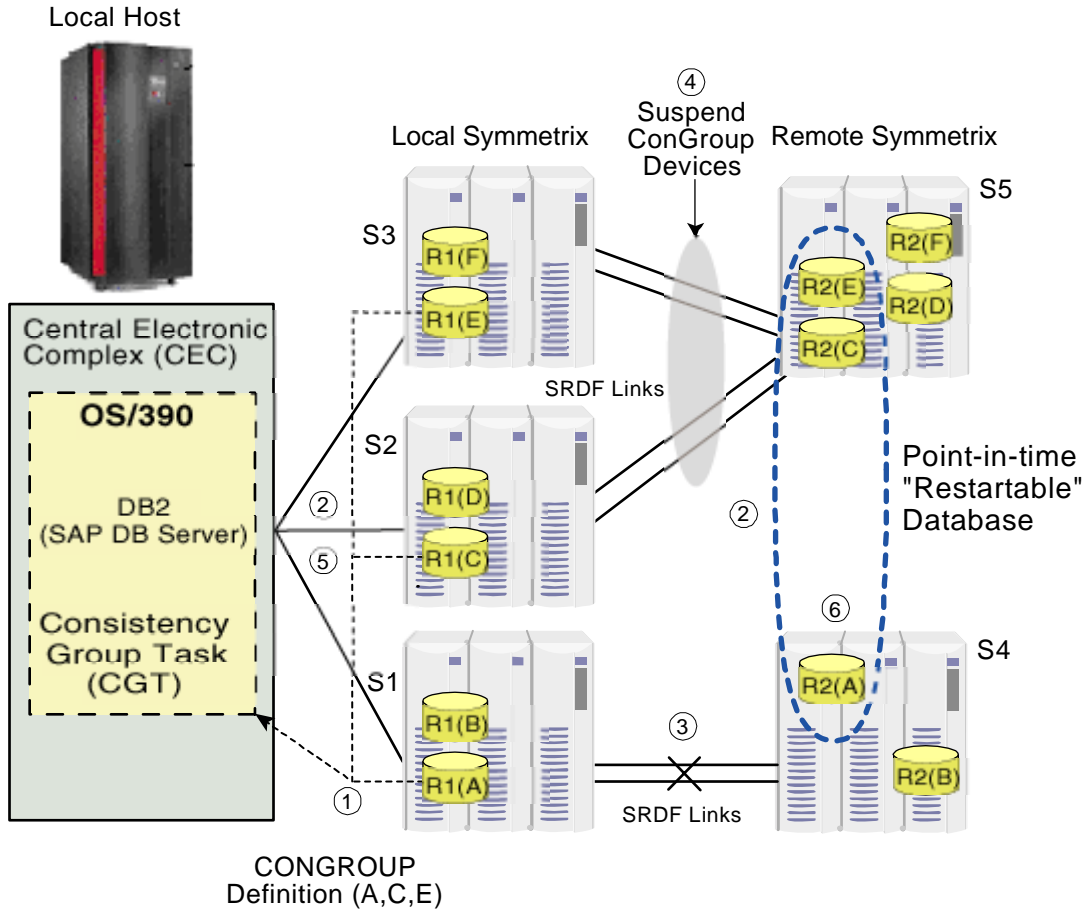


Figure 4. ConGroup for OS/390

### 3 Cloning SAP Systems

This section describes the procedure for cloning an SAP DB2 instance using TimeFinder. If the source DB2 is a member of a data sharing group, this procedure will create a non-data sharing target DB2. All data is moved in the Symmetrix, and metadata manipulation on the target DB2 instance is required. Metadata includes the DB2 high level qualifiers (HLQs), STOGROUP definitions, and BSDS changes. Figure 5 is a simple example of a cloned DB2 instance.

Examples provided within this document use the following nomenclature:

DB2SAP1 = Source DB2 high-level qualifier

DB2SAP2 = Target DB2 high-level qualifier

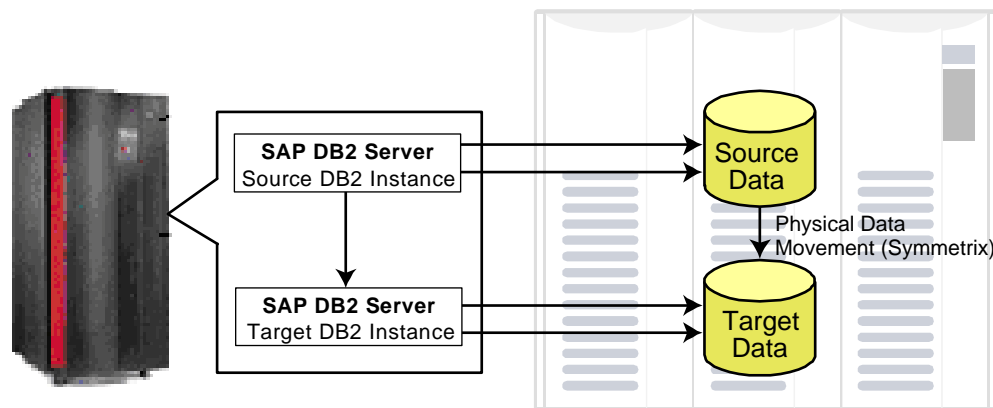
SAP1 = Source DB2 instance

SAP2 = Target DB2 instance

SAPS\*\* = Source volumes

SAPT\*\* = Target volumes

Source and Target must be different DB2 systems



**Figure 5. SAP and DB2 UDB Data Cloning Overview**

The DB2 instance cloning described in this paper assume that if the source DB2 image uses data sharing, then the target DB2 instance is a non-data sharing DB2 instance. This is true regardless of whether the cloning is done within the same LPAR or to a different LPAR.

Creating a cloned DB2 instance that uses data sharing within the same LPAR, or to a different LPAR within the same Sysplex, would require changing the data sharing group name as part of the cloning process. Such a change is not possible, because the data sharing group name cannot be changed once it is set. As a result, all cloned DB2 instances must be non-data sharing.

It is possible to create a cloned DB2 with data sharing enabled if the target DB2 exists in a different LPAR outside of the Sysplex containing the source DB2. The procedures for creating such a clone are not addressed in this paper.

#### 3.1 Cloning the SAP Environment Within the Same LPAR

OS/390 must be prepared to support the target DB2 instance. The following steps are required prior to cloning the SAP environment, and need to be done only once:

1. Define target DB2 subsystem
2. Define ICF user catalogs

3. Create DB2 procedures
4. Create target DSNZPARM

The following sections describe the steps that are performed each time the cloned environment requires a refresh from the production system:

1. Cloning the datasets
2. Renaming cloned datasets
3. Updating the BSDS
4. Starting the target DB2 instance
5. Changing the HLQ of user-defined indexes on the DB2 catalog
6. Changing HLQ of work file DATABASE (DSNDB07)
7. Altering user objects to use the new HLQ

### 3.1.1 Define Target DB2 Subsystem

Define the target DB2 subsystem and its corresponding IRLM subsystem to OS/390:

```
SUBSYS SUBNAME(SAP2) INITRTN(DSN3INI) INITPARM('DSN3EP,*SAP2,S')
SUBSYS SUBNAME(IRS2)
```

### 3.1.2 Define ICF User Catalogs

Define ICF user catalogs for the target DB2 and define aliases for the new system and user datasets:

```
//DSNTCAT EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=*
//SYSIN DD *
DEFINE UCAT -
  (NAME (DB2SAP21 . USER . CATALOG) -
  CYLINDERS ( 20 5) -
  VOL (SAPT00) -
  ICFCATALOG ) -
DATA -
  (CYLINDERS ( 20 5) ) -
INDEX -
  (CYLINDERS ( 20 5) )

DEFINE ALIAS -
  NAME (DB2SAP2) -
  RELATE (DB2SAP21 . USER . CATALOG ) )
```

### 3.1.3 Create DB2 Procedures

Create a target set of DB2 and IRLM procedures in the procedures library. The target xxxxMSTR procedure must be updated to use the new high-level qualifier for the target BSDS. The xxxxMSTR must also be changed to start using the DSNZPARM that is created in the next section, Create Target DSNZPARM.

### 3.1.4 Create Target DSNZPARM

Create a new DSNZPARM member for the target DB2 by applying the following changes to a copy of the source DB2 DSNZPARM:

CATALOG=Target DB2 catalog high-level qualifier  
 IRLMPRC=Target IRLM procedure name  
 IRLMSID=Target IRLM subsystem name  
 ARCPFX1=Target archive high-level qualifier  
 ARCPFX2=Target archive high-level qualifier  
 STORPROC=Target stored-procedure procedure name

If the source DB2 is a member of a data sharing group, change DSHARE to **NO** in the DSN6GRP macro in the target DSNZPARM.

Link edit the new DSNZPARM to the DB2 exit library and give it a different name from the source system DSNZPARM.

### 3.1.5 Cloning the Datasets

Perform a TimeFinder ESTABLISH or RE-ESTABLISH to associate the BCVs with the standard devices. When the standard and BCV devices are synchronized, shut the source DB2 down to achieve a system-wide point of consistency. Then, SPLIT the BCVs to disassociate them from the standard devices. TimeFinder Utility conditions the DB2 volumes by relabeling the volumes and renaming the system datasets to reflect the VCAT name of the target DB2. The detailed steps are as follows:

1. Establish the volumes that contain all the source DB2 datasets: DB2 catalog and directory, user tablespaces, active logs, archive logs, BSDS, ICF catalogs, DBRMLIB.DATA, and RUNLIB.LOAD. The ESTABLISH can be done while processing on the source DB2 continues and is transparent to applications running there.
2. Bring down the source DB2 instance to stop all work and flush the DB2 buffers. In a data sharing environment, all members of the group must be brought down. The SPLIT must not proceed until the last data sharing member has been stopped.
3. SPLIT the BCV volumes using the SRDF Host Component or the EMCTF program.
4. Bring up the source DB2 instance and resume normal processing on it.

The cloned DB2 datasets now exist on the off-line BCV volumes; but the datasets are not cataloged. To use the cloned datasets in the same OS/390 image, the datasets must be renamed and recataloged in the target ICF user catalog created in Define ICF User Catalogs.

### 3.1.6 Renaming Cloned Datasets

The TimeFinder Utility (TFU) completes the dataset cloning process by relabeling the volumes and renaming the datasets.

1. Relabel all BCV volumes to their new VOLSER using TFU.
2. TFU is used to rename DB2 datasets to use the new target high-level qualifiers for the target DB2 instance.

A sample TFU job to perform the relabel and rename is provided below:

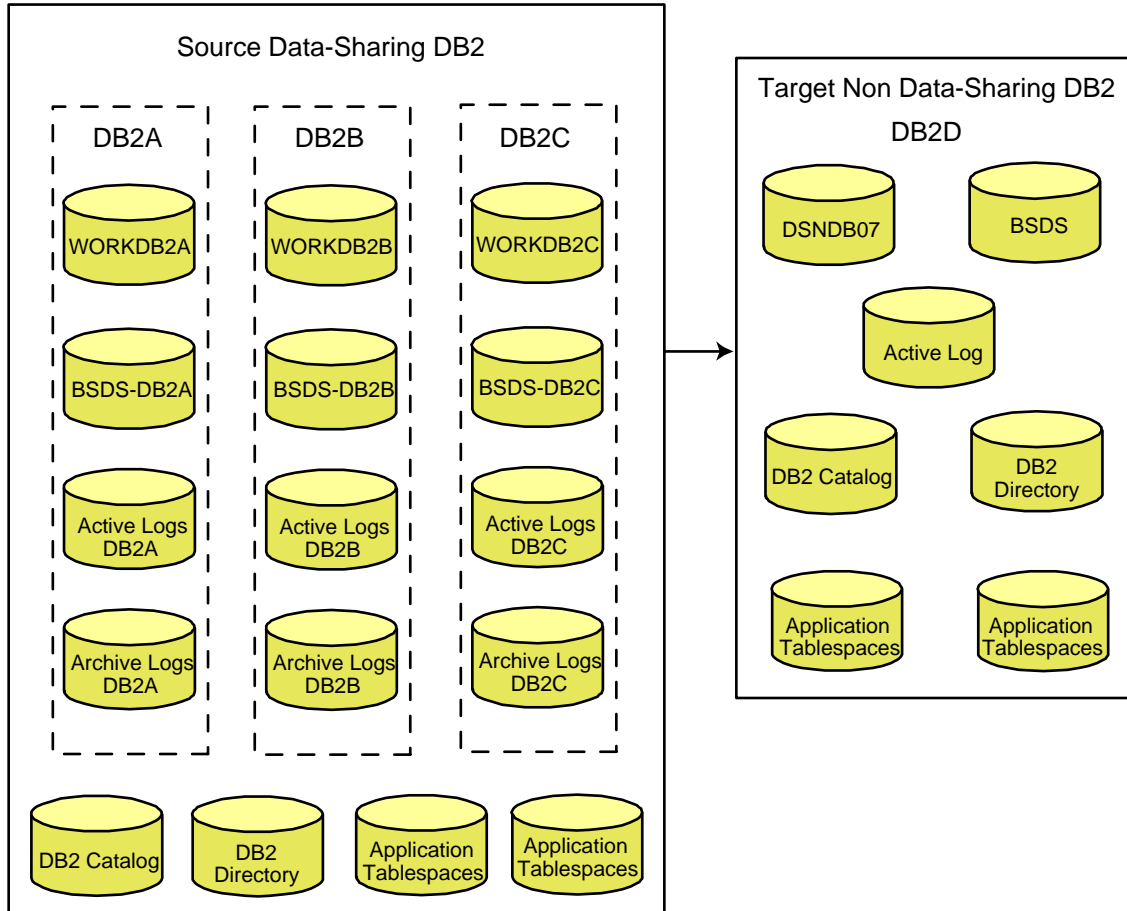
```
//EMCTFU      EXEC PGM=EMCTFU
```

```

//SYSOUT DD SYSOUT=*
//TFINPUT DD *
RELABEL CUU=4120,OLD-VOLSER=SAPS00,NEW-VOLSER=SAPT00
RELABEL CUU=4121,OLD-VOLSER=SAPS01,NEW-VOLSER=SAPT01
*
PROCESS CUU=4120,VOLSER=SAPT00,BOTH
PROCESS CUU=4121,VOLSER=SAPT01,BOTH
*
CATALOG DB2SAP21.USER.CATALOG,DEFAULT
SOURCECATALOG DEFAULT=NO,DIRECT=YES
RENAME DB2SAP1.DSNDBC.DSNDB01.*,
       DB2SAP2.DSNDBC.DSNDB01.*,
       CATALOG=DB2SAP21.USER.CATALOG
RENAME DB2SAP1.DSNDBC.DSNDB06.*,
       DB2SAP2.DSNDBC.DSNDB06.*,
       CATALOG=DB2SAP21.USER.CATALOG
RENAME DB2SAP1.DSNDBC.DSNDB07.*,
       DB2SAP2.DSNDBC.DSNDB07.*,
       CATALOG=DB2SAP21.USER.CATALOG
RENAME DB2SAP1.DSNDBD.DSNDB01.*,
       DB2SAP2.DSNDBD.DSNDB01.*,
       CATALOG=DB2SAP21.USER.CATALOG
RENAME DB2SAP1.DSNDBD.DSNDB06.*,
       DB2SAP2.DSNDBD.DSNDB06.*,
       CATALOG=DB2SAP21.USER.CATALOG
RENAME DB2SAP1.DSNDBD.DSNDB07.*,
       DB2SAP2.DSNDBD.DSNDB07.*,
       CATALOG=DB2SAP21.USER.CATALOG
RENAME DB2SAP1.LOG*,
       DB2SAP2.LOG*,
       CATALOG=DB2SAP21.USER.CATALOG
RENAME DB2SAP1.BSDS*,
       DB2SAP2.BSDS*,
       CATALOG=DB2SAP21.USER.CATALOG
RENAME DB2SAP1.DBRMLIB.DATA,
       DB2SAP2.DBRMLIB.DATA,
       CATALOG=DB2SAP21.USER.CATALOG
RENAME DB2SAP1.RUNLIB.LOAD,
       DB2SAP2.RUNLIB.LOAD,
       CATALOG=DB2SAP21.USER.CATALOG
RENAME hlqsource1.DSNDBC.application data.*,
       hlqtarget1.DSNDBC.application data.*,
       CATALOG=DB2SAP21.USER.CATALOG
RENAME hlqsource1.DSNDBD.application data.*,
       hlqtarget1.DSNDBD.application data.*,
       CATALOG=DB2SAP21.USER.CATALOG
RENAME hlqsource2.DSNDBC.application data.*,
       hlqtarget2.DSNDBC.application data.*,
       CATALOG=DB2SAP21.USER.CATALOG
RENAME hlqsource2.DSNDBD.application data.*,
       hlqtarget2.DSNDBD.application data.*,
       CATALOG=DB2SAP21.USER.CATALOG

```

If the source DB2 instance is a member of a data sharing group then the “work file database” datasets that were cloned from a data sharing source should be renamed to DSNDB07 to support the non-data sharing target DB2. Figure 6 shows the configurations for the source and target DB2 instances where the source DB2 is a member in a three-way, data sharing environment. Note that the target DB2 instance has a different subsystem ID because it is created within the same LPAR as the source DB2 instance.



**Figure 6. DB2 Cloning From a Three-Way Data Sharing Environment to a Non-Data Sharing Environment on the Same LPAR**

### 3.1.7 Updating the BSDS

The next step in the cloning process is to update the target BSDS to reflect the new high-level qualifier for the system datasets. This is done using the DB2 Change Log Inventory utility (DSNJU003). In the example below, the high-level qualifier for the target DB2 instance datasets (DB2SAP2) is specified in the NEWCAT statement. The four target active log dataset definitions are deleted and redefined. The start RBA and end RBA for the target active log datasets must be specified. To find these RBA values, run the DB2 utility DSNJU004 on the target BSDS, and then edit the output. Refer to Appendix A for the sample edit macro code to assist with this process.

```

//DSNTLOG   EXEC  PGM=DSNJU003
//SYSUT1    DD    DISP=OLD,DSN=DB2SAP2.BSDS01
//SYSUT2    DD    DISP=OLD,DSN=DB2SAP2.BSDS02
//SYSPRINT  DD    SYSOUT=*
//SYSIN     DD    *
NEWCAT  VSAMCAT=DB2SAP2
DELETE  DSNAME=DB2SAP1.LOGCOPY1.DS01
DELETE  DSNAME=DB2SAP1.LOGCOPY2.DS01
DELETE  DSNAME=DB2SAP1.LOGCOPY1.DS02
DELETE  DSNAME=DB2SAP1.LOGCOPY2.DS02
DELETE  DSNAME=DB2SAP1.LOGCOPY1.DS03
DELETE  DSNAME=DB2SAP1.LOGCOPY2.DS03
DELETE  DSNAME=DB2SAP1.LOGCOPY1.DS04
DELETE  DSNAME=DB2SAP1.LOGCOPY2.DS04
NEWLOG  DSNAME=DB2SAP2.LOGCOPY1.DS03,COPY1,
        STARTRBA=00211133B000,ENDRBA=00211DDBAFFF
NEWLOG  DSNAME=DB2SAP2.LOGCOPY2.DS03,COPY2,
        STARTRBA=00211133B000,ENDRBA=00211DDBAFFF
NEWLOG  DSNAME=DB2SAP2.LOGCOPY1.DS04,COPY1,
        STARTRBA=00211DDBB000,ENDRBA=00212A83AFFF
NEWLOG  DSNAME=DB2SAP2.LOGCOPY2.DS04,COPY2,
        STARTRBA=00211DDBB000,ENDRBA=00212A83AFFF
NEWLOG  DSNAME=DB2SAP2.LOGCOPY1.DS01,COPY1,
        STARTRBA=00212A83B000,ENDRBA=0021372BAFFF
NEWLOG  DSNAME=DB2SAP2.LOGCOPY2.DS01,COPY2,
        STARTRBA=00212A83B000,ENDRBA=0021372BAFFF
NEWLOG  DSNAME=DB2SAP2.LOGCOPY1.DS02,COPY1,
        STARTRBA=0021372BB000,ENDRBA=002143D3AFFF
NEWLOG  DSNAME=DB2SAP2.LOGCOPY2.DS02,COPY2,
        STARTRBA=0021372BB000,ENDRBA=002143D3AFFF

```

If the source DB2 is a member of a data sharing group, add the DATASHR DISABLE command to the DSNJU003 job to set data sharing mode to “non-data sharing” in the target BSDS.

### 3.1.8 Starting the Target DB2 Instance

The target DB2 instance can now be started. The target DB2 instance will have no knowledge of the user tablespaces, indexes, and datasets. In the target DB2 instance, the DB2 STOGROUPs still point to the source DB2 VCAT. Therefore, even though the underlying datasets have been cloned and renamed, DB2 must be made aware of these new dataset names, by changing the DB2 catalog references. To prevent the newly cloned DB2 instance from allowing access to objects in the original subsystem, start the cloned subsystem with ACCESS(MAINT).

If cloning from a data sharing environment, disable data sharing for the target DB2 instance. In this case, a cold start of the target DB2 is enforced to prevent outstanding units of work from being backed out. However, in this cloning procedure, the cold start is essentially the same as a normal (warm) restart because all members of the data sharing group were brought down prior to the TimeFinder SPLIT. Stopping all source data sharing members ensures there are no outstanding units of work to back out. Simply reply **Y** to the WTOR (Write to Operator), indicating that a conditional restart will be done.

### 3.1.9 Changing the HLQ of User-Defined Indexes on the DB2 Catalog

SAP 4.6B defines two user indexes on the DB2 catalog:

- SAPR3.SYSTBLSP~0 (index on SYSIBM.SYSTABLESPACE)
- SAPR3.SYSTABLE (index on SYSIBM.SYSTABLES)

These indexes must be altered to use the new VCAT specified in *Renaming Cloned Datasets* on page 15. If other user indexes have been defined on the DB2 catalog, the same process must be done for those as well. The following sample Job Control Language (JCL) can be used to perform an alter VCAT of the user indexes to DB2SAP2.

```
//DSNTIRU EXEC PGM=IKJEFT01
//SYSTSPRT DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSTSIN DD *
DSN SYSTEM(SAP2)
  -STOP DATABASE(DSNDB06) SPACENAM(SYSTBLSP)
  -STOP DATABASE(DSNDB06) SPACENAM(SYSTABLE)
  RUN PROGRAM(DSNTIAD) PLAN(DSNTIA61) -
    LIBRARY('DB2SAP2.RUNLIB.LOAD')
  -START DATABASE(DSNDB06) SPACENAM(SYSTBLSP)
  -START DATABASE(DSNDB06) SPACENAM(SYSTABLE)
END
//SYSIN DD *
SET CURRENT SQLID='SAPR3';
ALTER INDEX "SYSTABLE~0" USING VCAT DB2SAP2;
ALTER INDEX "SYSTBLSP~0" USING VCAT DB2SAP2;
```

### 3.1.10 Changing HLQ of Work File Database (DSNDB07)

DSNDB07 is a user-managed database with multiple tablespaces defined within it. At this point in the procedure, the underlying VSAM datasets have been renamed, but the DB2 catalog still points to the source DB2 datasets. Correcting this catalog reference requires dropping the “work file database” and re-creating the database and all its tablespaces with the correct VCAT name. When dropping a database that has user-managed tablespaces, DB2 does not attempt to delete the underlying VSAM datasets. When user-defined tablespaces are re-created, DB2 will not attempt to define the VSAM datasets and it will use the datasets created by the rename process.

The following example changes the DB2 catalog definition for DSNDB07 definitions without deleting the underlying datasets:

```
//STEP1 EXEC PGM=IKJEFT01
//SYSTSPRT DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//SYSTSIN DD *
DSN SYSTEM(SAP2)
  -STOP DATABASE(DSNDB07)
  RUN PROGRAM(DSNTIAD) PLAN(DSNTIA61) PARM('RC0') -
    LIB('DB2SAP2.RUNLIB.LOAD')
END
//SYSIN DD *
DROP DATABASE DSNDB07 ;
COMMIT;
CREATE DATABASE DSNDB07 ;
//*
//STEP2 EXEC PGM=IKJEFT01
//SYSTSPRT DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSTSIN DD *
DSN SYSTEM(SAP2)
  -STOP DATABASE(DSNDB07)
```

```

RUN PROGRAM(DSNTIAD) PLAN(DSNTIA61) -
  LIB('DB2SAP2.RUNLIB.LOAD')
-START DATABASE(DSNDB07)
END
//SYSIN DD *
CREATE TABLESPACE DSN4K01 IN DSNDB07
  BUFFERPOOL BP0
  CLOSE NO
  USING VCAT DB2SAP2;
CREATE TABLESPACE DSN4K02 IN DSNDB07
  BUFFERPOOL BP0
  CLOSE NO
  USING VCAT DB2SAP2;
CREATE TABLESPACE DSN4K03 IN DSNDB07
  BUFFERPOOL BP0
  CLOSE NO
  USING VCAT DB2SAP2;
CREATE TABLESPACE DSN4K04 IN DSNDB07
  BUFFERPOOL BP0
  CLOSE NO
  USING VCAT DB2SAP2;
CREATE TABLESPACE DSN32K01 IN DSNDB07
  BUFFERPOOL BP32K
  CLOSE NO
  USING VCAT DB2SAP2;
CREATE TABLESPACE DSN32K02 IN DSNDB07
  BUFFERPOOL BP32K
  CLOSE NO
  USING VCAT DB2SAP2;

```

If the source DB2 instance is a member of a data sharing group, the original data sharing “work file databases” should be dropped, and DSNDB07 should be created to support the non-data sharing target environment. The remaining “work file databases” that were cloned from the remaining members will not be used for the target DB2 instance and may be dropped, and corresponding datasets may be deleted.

### 3.1.11 Altering User Objects to Use the New HLQ

The final step in the DB2 instance cloning process is to update the target DB2 catalog to reflect the new high-level qualifier of the cloned DB2 datasets:

1. Create new STOGROUPS with the new volumes and the new VCAT name.
2. Stop all target user databases.
3. Execute an ALTER TABLESPACE and ALTER INDEX statement with the USING STOGROUP parameter, specifying the newly created STOGROUP.
4. Start all stopped target user databases.
5. If any existing DDL references the old STOGROUP (source STOGROUP definition), the STOGROUP should be dropped at this point and re-created using the new volume definitions and the new VCAT name. If this is not done, objects created by DDL referencing this old STOGROUP will be created on the source DB2 volumes with the source DB2 VCAT name.

The following sample catalog query will generate the ALTER statements required for step 3 in the previous procedure. Refer to Appendix B for a sample REXX code to add DB2 COMMIT commands to the sequence of the ALTER commands.

```

SELECT 'ALTER TABLESPACE ' CONCAT STRIP(DBNAME) CONCAT '.'
      CONCAT STRIP(TSNAME) CONCAT ' USING STOGROUP NEWSG1;'
FROM SYSIBM.SYSTABLEPART
WHERE PARTITION = 0
AND DBNAME NOT IN ('DSNDB07','DSNDB01','DSNDB06')
UNION
SELECT 'ALTER TABLESPACE ' CONCAT STRIP(DBNAME) CONCAT '.'
      CONCAT STRIP(TSNAME) CONCAT 'PART ' CONCAT DIGITS(PARTITION)
      CONCAT ' USING STOGROUP NEWSG1;'
FROM SYSIBM.SYSTABLEPART
WHERE PARTITION > 0
AND DBNAME NOT IN ('DSNDB07','DSNDB01','DSNDB06')
UNION
SELECT 'ALTER INDEX ' CONCAT STRIP(B.CREATOR)
      CONCAT '.' CONCAT STRIP(A.IXNAME)
      CONCAT ' USING STOGROUP NEWSG1;'
FROM SYSIBM.SYSINDEXPART A,SYSIBM.SYSINDEXES B
WHERE A.IXNAME=B.NAME
AND B.DBNAME NOT IN ('DSNDB07','DSNDB01','DSNDB06')
AND A.PARTITION=0
UNION
SELECT 'ALTER INDEX ' CONCAT STRIP(B.CREATOR)
      CONCAT '.' CONCAT STRIP(A.IXNAME)
      CONCAT ' PART ' CONCAT DIGITS(PARTITION)
      CONCAT ' USING STOGROUP NEWSG1;'
FROM SYSIBM.SYSINDEXPART A,SYSIBM.SYSINDEXES B
WHERE A.IXNAME = B.NAME
AND B.DBNAME NOT IN ('DSNDB07','DSNDB01','DSNDB06')
AND A.PARTITION > 0

```

### 3.1.12 Performance Recommendations for the Rename Process

Rename performance within the OS/390 environment can be optimized by using the following recommendations:

- **TFU Recommendations:**  
Specify empty UCAT so no CLEANUP is required, or delete and redefine the UCAT  
Use SOURCECATALOG DEFAULT = NO  
Use DIRECT = YES (TFU 4.1.0 and higher only)  
Direct EMCTFU output to disk rather than JES output queue
- **GRS Recommendations:**  
Change Ring to Star configuration for SYSPLEX  
RESMIL (default is 10 - reduce to 1)  
ACCELSYS (default is 99 - reduce to 2)  
These settings have been found to improve performance in lab testing.
- **Uniform Distribution of Datasets Across Volumes:**  
If the SAP environment is located on a relatively small number of volumes the renaming process may be longer than if the data is spread across a larger number of volumes. Processing a large number of datasets on a small number of volumes restricts TFU parallelism.
- **The cloning process requires that the database datasets of the target DB2 be renamed via the TFU and that the DB2 catalog entries for these datasets be ALTERED in DB2 to reflect the new names. These two processes can be executed in parallel, provided the target DB2 instance can be started.**

This optimization of the process can be achieved by first renaming the target DB2 system datasets (DB2 catalog, DB2 directory, BSDS, logs), and then restarting the target system with ACCESS(MAINT). The TFU job to rename the remaining datasets and the ALTER job to update the DB2 catalog entries for these datasets can then be submitted in parallel, speeding up the overall cloning process.

## 3.2 Cloning SAP to Another LPAR

Use the following procedure to clone SAP to an LPAR that does not share ICF catalogs with the source LPAR. If the target DB2 instance will reside on an LPAR that shares the same ICF catalogs as the source DB2 then use the process described in *Cloning the SAP Environment Within the Same LPAR* on page 13.

The procedures in this section assume that the target OS/390 system will have the same I/O subsystem definitions as the source OS/390 system. The target OS/390 system must be prepared to support a new DB2 instance. The following steps are required prior to cloning the SAP environment, and they need to be done only once:

1. Define target DB2 subsystem
2. Create DB2 procedures
3. Create target DSNZPARM

The following sections describe the steps that are performed each time the cloned environment requires a refresh from the production system:

1. Cloning the datasets
2. Associating source data to target environment
3. Starting the target DB2 instance

### 3.2.1 Define Target DB2 Subsystem

Define the target DB2 subsystem and its corresponding IRLM subsystem to OS/390. These procedures assume that source data sharing groups will be reduced to a single target, non-data sharing DB2.

```
SUBSYS SUBNAME(SAP1) INITRTN(DSN3INI) INITPARM('DSN3EP,*SAP1,S')
SUBSYS SUBNAME(IRS1)
```

The SDSNEXIT, SDSNLOAD, SDXRRESL and SDSNLINK libraries must be APF-authorized. The hlq.SDSNLINK must be in the linklist.

### 3.2.2 Create DB2 Procedures

Replicate the DB2 and IRLM procedures in the procedures library on the target LPAR.

### 3.2.3 Create Target DSNZPARM

If the source DB2 is a member of a data sharing group, change DSHARE to **NO** in the DSN6GRP macro in the new DSNZPARM. Link edit the new DSNZPARM to the DB2 exit library on the target LPAR.

### 3.2.4 Cloning the Datasets

Perform a TimeFinder ESTABLISH or RE-ESTABLISH to associate the BCVs with the standard devices. When the standard and BCV devices are synchronized, the source DB2 instance is shut down to achieve a system-wide point of consistency. The detailed steps are as follows:

1. Establish the volumes that contain all the source DB2 datasets: DB2 catalog and directory, user tablespaces, active logs, archive logs, BSDS, ICF catalogs, DBRMLIB . DATA, and RUNLIB . LOAD. The ESTABLISH can be done while processing on the source DB2 continues, and is transparent to the application.
2. Bring down the source DB2 to stop all work and flush the buffers. In a data sharing environment, all members of the group must be brought down. The SPLIT must not proceed until the last data sharing member has been stopped.
3. SPLIT the BCV volumes using Host Component or the EMCTF program.
4. Bring up the source DB2 and resume normal processing on it.

The cloned DB2 datasets now exist on the off-line BCV volumes, but dataset are not cataloged.

### 3.2.5 Associating Source Data to the Target Environment

The ICF user catalog must be imported from the source LPAR into the target LPAR and aliases must be created for the high-level qualifier(s). The ICF user catalogs should be cloned on the BCVs so they can be connected to the master catalog. Steps to accomplish these tasks are as follows:

1. Vary the BCVs on line to the target OS/390.
2. Import connect the ICF user catalogs.

```
//IMPORT      EXEC PGM=IDCAMS
//SYSPRINT    DD      SYSOUT=*
//SYSIN       DD      *
IMPORT                                -
  OBJECTS (                            -
    (DB2SAP11.USER.CATALOG             -
      VOLUME(SAPT00)                   -
      DEVICETYPE(3390)))              -
  CONNECT CAT(CATALOG.MVSICFM.VO2890C)
```

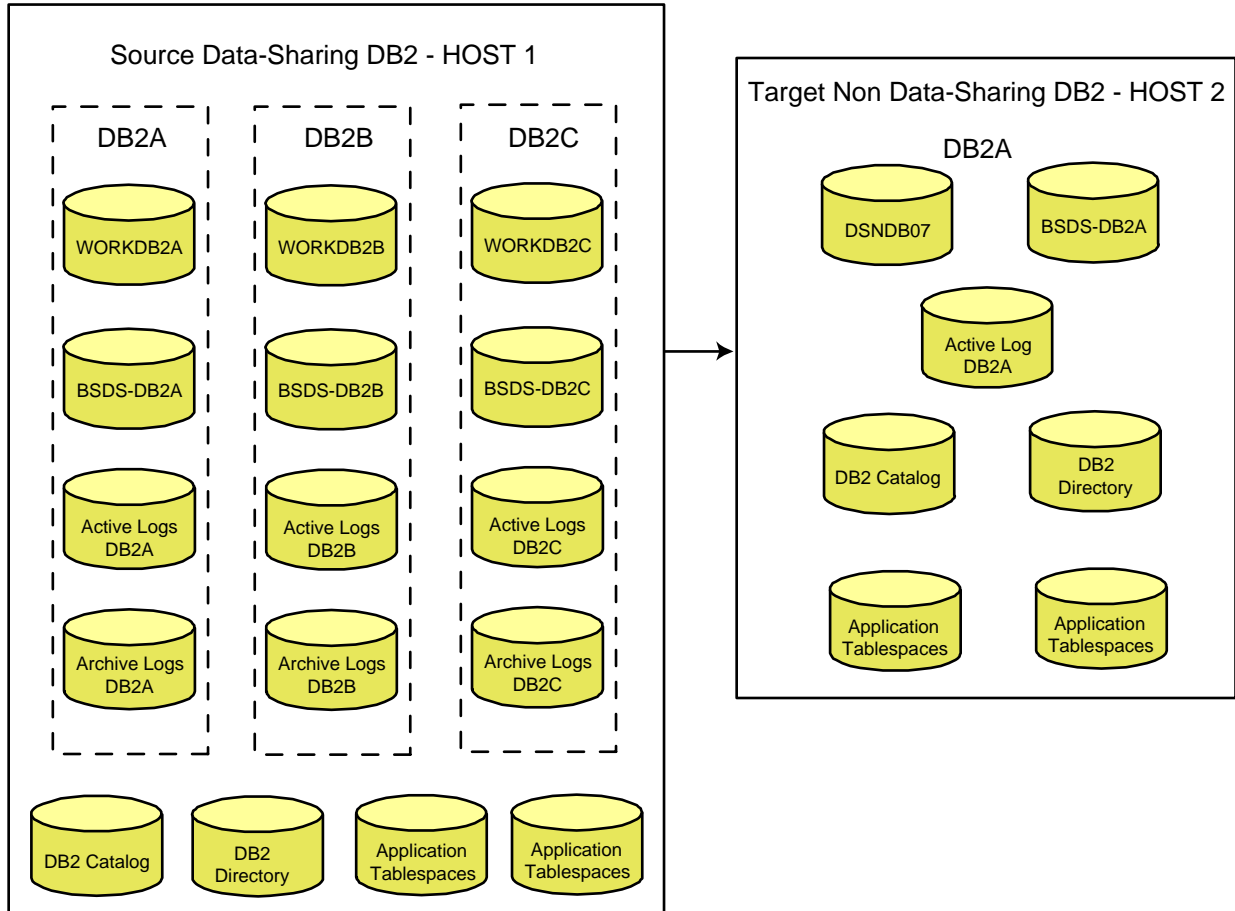
3. Define all aliases.

```
//DSNTCAT     EXEC PGM=IDCAMS
//SYSPRINT    DD      SYSOUT=*

//SYSIN       DD      *
DEFINE ALIAS                                -
  ( NAME(DB2SAP1)                            -
    RELATE(DB2SAP11.USER.CATALOG) )
```

### 3.2.6 Starting the Target DB2 Instance

If the source DB2 is a member of a data sharing group, execute the DATASHR DISABLE command using the DSNJU003 utility to set data sharing mode to non-data sharing. Data sharing must be disabled for the target DB2 instance. Figure 7 shows the configurations for the source and target DB2 instances where the source DB2 is a member of a three-way data sharing environment. In Figure 7, DB2A is a member in the source data sharing group and has been replicated onto a different LPAR but the DB2 subsystem ID remains the same.



**Figure 7. DB2 Cloning From a Three-Way Data Sharing Environment to a Non-Data Sharing Environment on a Different LPAR**

The change from data sharing to non-data sharing for the target DB2 instance enforces a cold start of the target DB2 instance to prevent outstanding units of work from being backed-out. However, in this cloning procedure, the cold start is essentially the same as a normal (warm) restart because bringing down all members of the data sharing group prior to the TimeFinder SPLIT ensures there are no outstanding units of work to back out. Simply reply **Y** to the WTOR, indicating that a conditional restart will be done.

### 3.2.7 Work Database Considerations for Data Sharing

If the source DB2 is a member of a data sharing group, the original “work file databases” datasets of the data sharing members must be renamed to DSNDDB07. This could be done using the IDCAMS ALTER command.

DSNDDB07 is a user-managed database with multiple tablespaces defined within it. At this point in the procedure, the underlying VSAM datasets are renamed but the DB2 catalog still points to the source DB2 datasets. Correcting this catalog reference requires dropping the “work file database,” and re-creating the database and all tablespaces with the correct database name. When dropping a database that has user-managed tablespaces, DB2 does not attempt to delete the underlying VSAM datasets. When the user-defined tablespaces are re-created, DB2 does not attempt to define the VSAM datasets, and uses the datasets created by the rename process.

The following example changes the DB2 catalog definition without deleting the underlying datasets:

```

//STEP1      EXEC PGM=IKJEFT01
//SYSTSPRT   DD    SYSOUT=*
//SYSPRINT   DD    SYSOUT=*
//SYSTSIN    DD    *
  DSN SYSTEM(SAP1)
  -STOP DATABASE(data sharing work database)
  RUN PROGRAM(DSNTIAD) PLAN(DSNTIA61) PARM('RC0') -
    LIB('DB2SAP1.RUNLIB.LOAD')
  END
//SYSIN      DD    *
  DROP DATABASE data sharing work database ;
  COMMIT;
  CREATE DATABASE DSNDB07 ;
//*
//STEP2      EXEC PGM=IKJEFT01,
//SYSTSPRT   DD    SYSOUT=*
//SYSPRINT   DD    SYSOUT=*
//SYSTSIN    DD    *
  DSN SYSTEM(SAP1)
  -STOP DATABASE(DSNDB07)
  RUN PROGRAM(DSNTIAD) PLAN(DSNTIA61) -
    LIB('DB2SAP1.RUNLIB.LOAD')
  -START DATABASE(DSNDB07)
  END
//SYSIN      DD    *
  CREATE TABLESPACE DSN4K01 IN DSNDB07
    BUFFERPOOL BP0
    CLOSE NO
    USING VCAT DB2SAP1;
  CREATE TABLESPACE DSN4K02 IN DSNDB07
    BUFFERPOOL BP0
    CLOSE NO
    USING VCAT DB2SAP1;
  CREATE TABLESPACE DSN4K03 IN DSNDB07
    BUFFERPOOL BP0
    CLOSE NO
    USING VCAT DB2SAP1;
  CREATE TABLESPACE DSN4K04 IN DSNDB07
    BUFFERPOOL BP0
    CLOSE NO
    USING VCAT DB2SAP1;
  CREATE TABLESPACE DSN32K01 IN DSNDB07
    BUFFERPOOL BP32K
    CLOSE NO
    USING VCAT DB2SAP1;
  CREATE TABLESPACE DSN32K02 IN DSNDB07
    BUFFERPOOL BP32K
    CLOSE NO
    USING VCAT DB2SAP1;

```

### **3.3 Recoverability of the Cloned Target SAP and DB2 Instance**

When cloning from a data sharing environment to a non-data sharing environment, archive logs and active logs created prior to the cold start cannot be used for recovery. This is because multiple logs that exist in a data sharing environment cannot be merged into a single log for a non-data sharing environment. When cloning a data sharing DB2 instance, within or across LPARs, the cloned DB2 instance is restartable but not recoverable.

When cloning non-data sharing environments, the cloned DSNZPARM contains past and future archive logs. Cloning archive log datasets within an LPAR requires using a different archive log prefix to prevent duplicate archive log dataset names. Cloning and renaming an archive log requires that the archive log be on DASD. It may be unreasonable to maintain more than a few archive logs on disk. Therefore, it may not be practical to clone a DB2 instance and ensure recoverability within the same LPAR.

## **4 Backing Up SAP Environments**

The size and complexity of information systems has vastly increased in recent years. The requirement for high availability is more important than ever. The 24-by-7 requirement makes it extremely difficult to perform database maintenance operations and to prepare for fast recovery in case of a disaster. Acceptable solutions are ones that don't compromise availability or performance.

To back up application systems as big and complicated as SAP, a Split Mirror Backup solution is recommended. This solution offers a quick way of creating backup environments (as opposed to using DB2 tools) and has no effect on production availability and performance. Split Mirror Backup can provide very fast recovery.

This section describes three ways to create the Split Mirror Backup using EMC TimeFinder. The procedures for recovering from this kind of backup are discussed in SAP Recovery Procedures.

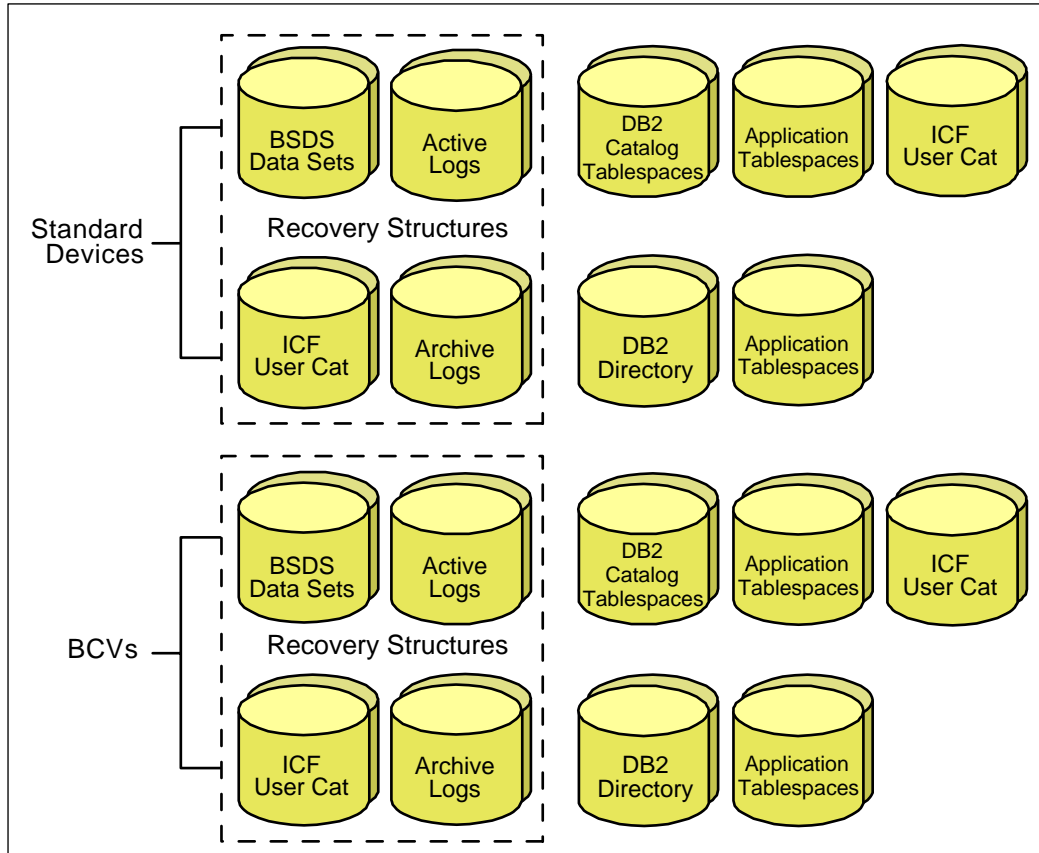
### **4.1 Creating a Split Mirror Backup Environment**

The three ways to create the Split Mirror Backup are based on the same concept of stopping application writes to the production volumes, and then SPLITting the BCVs. This SPLIT process creates a consistent copy of the SAP system on the BCVs. The three ways to create a Split Mirror Backup differ in the mechanisms in which I/Os are stopped, and in the operational procedures.

Creating Split Mirror Backups for the SAP system enables recovery of the entire system to a prior point in time. One reason for performing these backups is to recover from application logical errors that may contaminate the production environment. Other reasons may be due to software, hardware, or physical environment disruption.

A Split Mirror Backup can be used to correct error conditions within the SAP environment in the following ways:

- To restore the SAP environment to the time of the Split Mirror Backup. In this case, all volumes from the Split Mirror Backup are restored and the system is restarted.
- To recover the system to a point in time that is more current than the Split Mirror Backup. This requires only the standard devices containing the affected application datasets be restored from the BCVs. To prepare for this, it is very important to place the DB2 recovery structures (active logs, archive logs, BSDS and their related ICF user catalogs) and SAP application datasets onto separate sets of volumes. Figure 8 shows an example of how this can be done.



**Figure 8. Split Mirror Dataset Placement**

#### 4.1.1 Split Mirror Backup Preparation

Recovery from a Split Mirror Backup is done using the TimeFinder RESTORE, DFDSS, or FDR commands. The DB2 RECOVER utility with the LOGONLY keyword is used to roll forward the split mirror image. In this process, log ranges that are recorded in DSNDB01.SYSLGRNX for the tablespace are applied.

Several DB2 catalog and directory tablespaces require special attention to ensure that they will be recoverable from a Split Mirror Backup. The following tablespaces along with their indexes do not have entries in DSNDB01.SYSLGRNX, even when defined with COPY YES:

DSNDB01.SYSUTILX  
 DSNDB01.DBD01  
 DSNDB01.SYSLGRNX  
 DSNDB06.SYSCOPY  
 DSNDB06.SYSGROUP  
 DSNDB01.SCT02  
 DSNDB01.SPT01

During a RECOVER LOGONLY of these objects, the log range is scanned for updates to be applied, ranging from the HPGRBRBA of the tablespace (kept in the header page of each tablespace) to the end of the log. Theoretically, if a tablespace from this list has not been updated for a long time, then the HPGRBRBA will contain an old RBA value, and corresponding archive logs might have already been deleted. Under these circumstances, recovery might fail for this tablespace.

To prevent the above from happening, a DB2 QUIESCE WRITE(YES) should be done to the seven listed tablespaces prior to creating a Split Mirror Backup. This will update the HPGRBRBA with the current RBA and will ensure recoverability of these seven tablespaces, and shorten the recovery process. To quiesce these tablespaces, use the following sample JCL:

```
//UTIL1      EXEC DSNUPROC,SYSTEM=SAP1,UID='QUIESCE1',UTPROC='',
// LIB='DSN610.PROD.SDSNLOAD'
//*
//DSNUPROC.SYSIN DD *
QUIESCE TABLESPACE DSNDB01.DBD01 WRITE(YES)
QUIESCE TABLESPACE DSNDB01.SYSLGRNX WRITE(YES)
QUIESCE TABLESPACE DSNDB06.SYSGROUP WRITE(YES)
QUIESCE TABLESPACE DSNDB01.SPT01 WRITE(YES)
QUIESCE TABLESPACE DSNDB01.SCT02 WRITE(YES)
//UTIL2      EXEC DSNUPROC,SYSTEM=SAP1,UID='QUIESCE2',UTPROC='',
// LIB='DSN610.PROD.SDSNLOAD'
//*
//DSNUPROC.SYSIN DD *
QUIESCE TABLESPACE DSNDB06.SYSCOPY WRITE(YES)
//UTIL3 EXEC DSNUPROC,SYSTEM=SAP1,UID='QUIESCE3',UTPROC='',
// LIB='DSN610.PROD.SDSNLOAD'
//*
QUIESCE TABLESPACE DSNDB01.SYSUTILX WRITE(YES)
```

#### 4.1.2 Creating a Split Mirror Backup Using DB2 Suspend and Resume

The commands -SET LOG SUSPEND and -SET LOG RESUME were introduced in DB2 v 6.1 (APAR PQ31492). The -SET LOG SUSPEND command forces the log buffers to disk, updates the high-written RBA in the BSDS, and suspends write I/Os to the log. After successful execution of this command, DB2 read operations are possible, but DB2 write operations are suspended. Response time of updating applications is elongated until the -SET LOG RESUME command is issued. Refer to Figure 9, Split Mirror Backup Using DB2 Suspend and Resume, on page 29.

This new functionality can be used to create a Split Mirror Backup of SAP using the following procedure:

1. RE-ESTABLISH (ESTABLISH if the first time) BCV devices to the standard volumes that make up the entire SAP environment (user datasets, DB2 catalog, DB2 directory, active logs, archive logs, BSDS and the related ICF user catalogs), and wait until synchronized.

```
//RUNUTIL    EXEC PGM=EMCTF

//SYSOUT     DD    SYSOUT=*
//SYSIN      DD    *
GLOBAL MAXRC=4, WAIT
RE-ESTABLISH 02, 4120
RE-ESTABLISH 02, 4121
RE-ESTABLISH 02, 4122
RE-ESTABLISH 02, 4123
```

2. Issue the DB2 -SET LOG SUSPEND command and wait for completion.
3. Perform an INSTANT SPLIT of the BCVs to create the Split Mirror Backup. The INSTANT SPLIT can SPLIT a large number of volumes very quickly so the suspend effect is minimal.

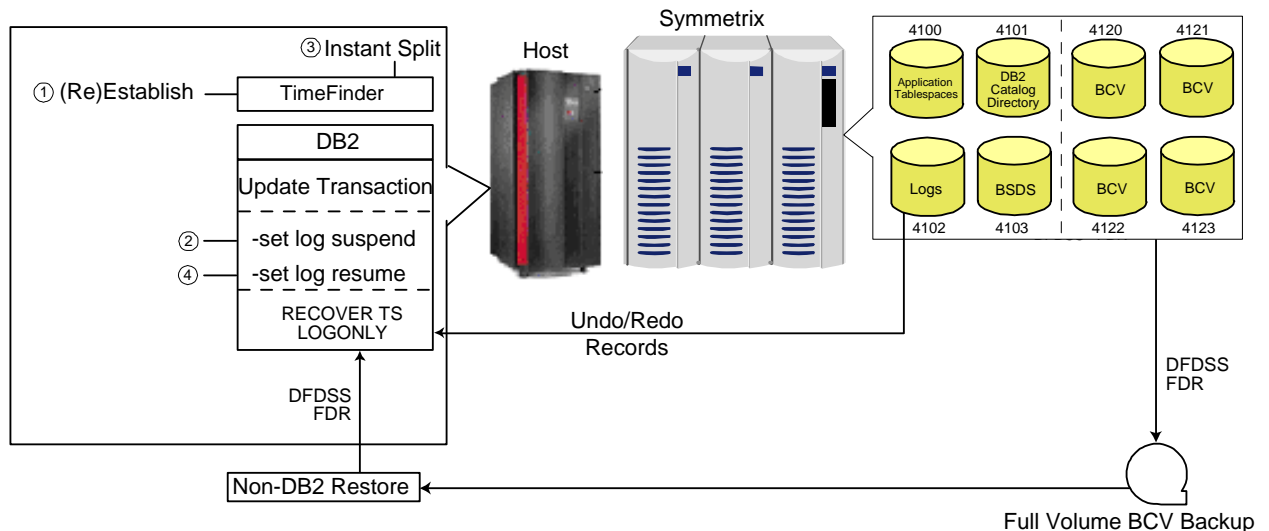
```
//RUNUTIL    EXEC PGM=EMCTF
//SYSUDUMP   DD    SYSOUT=*
//SYSOUT     DD    SYSOUT=*
```

```
//SYSIN      DD      *
GLOBAL      MAXRC=4,NOWAIT
SPLIT      02,4120,INSTANT(Y)
SPLIT      02,4121,INSTANT(Y)
SPLIT      02,4122,INSTANT(Y)
SPLIT      02,4123,INSTANT(Y)
```

In a data sharing environment, the `-SET LOG SUSPEND` command should be issued to all members of the group. The `INSTANT SPLIT` must not be performed until the `-SET LOG SUSPEND` has completed on all members.

This solution has some affect on running transactions and that affect depends on how long it takes to `SUSPEND` and `RESUME` log writes. Write I/O elongation can be expected for updates while the `SUSPEND` is in effect.

4. Issue the DB2 `-SET LOG RESUME` command to return to normal processing.



**Figure 9. Split Mirror Backup Using DB2 Suspend and Resume**

### 4.1.3 Creating a Split Mirror Backup Using Consistent Split

`CONSISTENT SPLIT` is an option of the TimeFinder v4.1 `SPLIT` command and requires microcode 5x66. `CONSISTENT SPLIT` holds all I/Os on the devices that are being `SPLIT`, and performs an `INSTANT SPLIT` to create the Split Mirror Backup environment. From the DB2 perspective, the state of the data on the BCVs is as if the OS/390 system incurred a power outage. The Split Mirror Backup environment is consistent from the dependent write I/O perspective, and when restarted, all running units of work will be resolved. This environment can also be used for recovery. Figure 10 shows how a Split Mirror Backup is created using `CONSISTENT SPLIT`.

To create a Split Mirror Backup environment follow these steps:

1. **RE-ESTABLISH** (**ESTABLISH** if the first time) BCV devices to the standard volumes that make up the entire SAP environment (user datasets, DB2 catalog, DB2 directory, active logs, archive logs, BSDS, and the related ICF catalogs) and wait until synchronized.

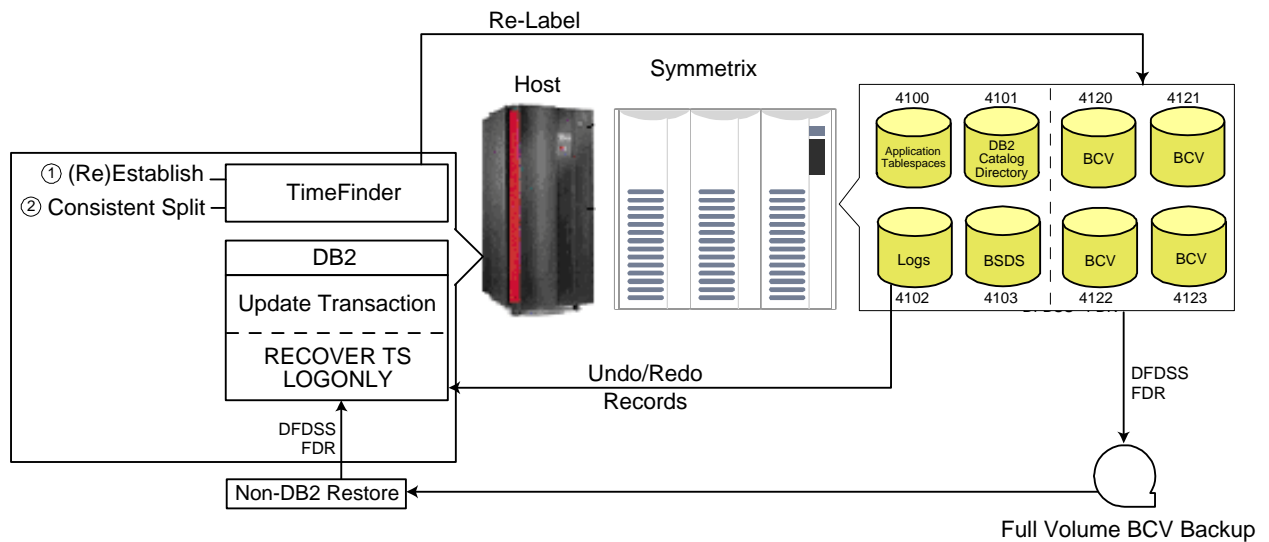
```
//RUNUTIL   EXEC PGM=EMCTF
//SYSUDUMP  DD      SYSOUT=*
//SYSOUT    DD      SYSOUT=*
```

```
//SYSIN DD *
GLOBAL MAXRC=4, WAIT
RE-ESTABLISH 02, 4120
RE-ESTABLISH 02, 4121
RE-ESTABLISH 02, 4122
RE-ESTABLISH 02, 4123
```

2. Perform a CONSISTENT SPLIT across all the devices.

```
//RUNUTIL EXEC PGM=EMCTF

//SYSOUT DD SYSOUT=*
//SYSIN DD *
GLOBAL MAXRC=4, NOWAIT
SPLIT 02, 4120, CONS(LOCAL(BYP))
SPLIT 02, 4121, CONS(LOCAL(BYP))
SPLIT 02, 4122, CONS(LOCAL(BYP))
SPLIT 02, 4123, CONS(LOCAL(BYP))
```



**Figure 10. Split Mirror Backup Using Consistent Split**

#### 4.1.4 Creating a Split Mirror Backup Using ConGroups

In remotely mirrored environments that use SRDF and ConGroups, an explicit ConGroup “trip” can be used to create a consistent image of the SAP environment on the R2 devices. ConGroup can then be immediately resumed using the remote SPLIT option to trigger an INSTANT SPLIT of the R2 BCVs before resuming the ConGroup. Figure 11 shows how a Split Mirror Backup is created using a ConGroup.

To create a Split Mirror Backup environment using this technique, follow these steps:

1. RE-ESTABLISH (ESTABLISH if the first time) R2 BCVs to the R2 devices that make up the entire SAP environment (user datasets, DB2 catalog, DB2 directory, active logs, archive logs, BSDS and the related ICF catalogs) and wait until synchronized.

```
//RUNUTIL EXEC PGM=EMCTF
//SYSUDUMP DD SYSOUT=*
//SYSOUT DD SYSOUT=*
//SYSIN DD *
GLOBAL MAXRC=4, WAIT
```

```
RE-ESTABLISH 02,RMT(4100,00A0)
RE-ESTABLISH 02,RMT(4101,00A1)
RE-ESTABLISH 02,RMT(4101,00A2)
RE-ESTABLISH 02,RMT(4101,00A3)
```

2. Perform an explicit trip of the ConGroup using the following command:

```
F con_group_task,SUSPEND con_group_name
```

Issue the following display command to make sure the ConGroup is suspended:

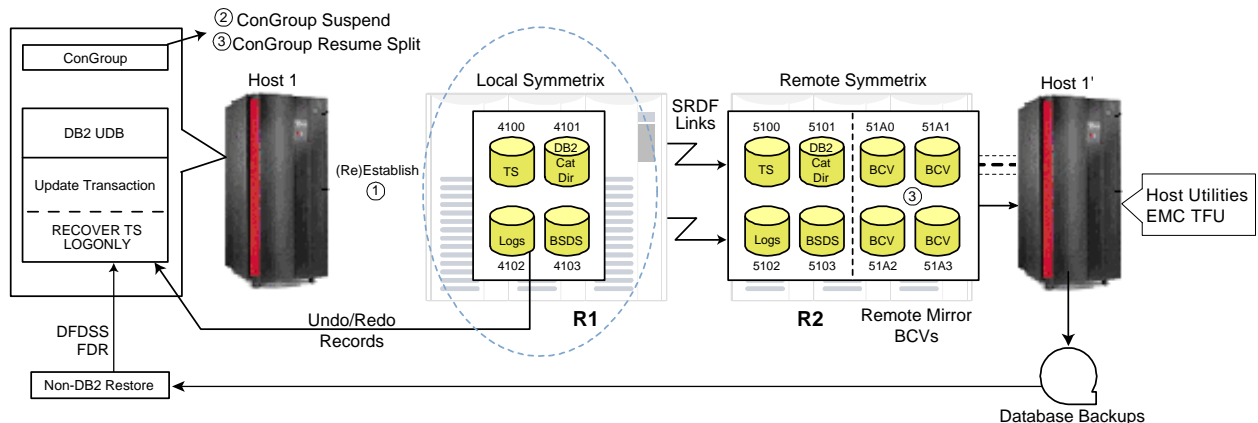
```
F con_group_task,DIS CON con_group_name NOLIST
```

3. Immediately resume the ConGroup with the remote SPLIT option using the following operator command:

```
F con_group_task,RESUME con_group_name,SPLIT
```

The above command will perform an INSTANT SPLIT on all the R2 BCV devices defined to the ConGroup, and will immediately resume the link to continue propagating updates to the remote site. Interpret the results of the following command to make sure the Split Mirror Backup environment is ready:

```
F con_group_task,DIS CON con_group_name NOLIST
```



**Figure 11. Split Mirror Backup Using EMC ConGroup**

#### 4.1.5 Split Mirror Backup Method Comparisons

The DB2 Suspend and Resume solution works in a data sharing environment across OS/390 images and has a performance effect on the live system transactions that are running. The Consistent Split solution has no performance impact and requires TimeFinder release 5.1.0 to support multi-image data sharing environments. The ConGroup solution works in a data sharing environment across OS/390 images and there is no performance impact.

## 4.2 Creating Multiple Split Mirrors

For best results, keep a second set of BCVs to hold a second split-mirror image of the system. There are several reasons for doing this:

- While RE-ESTABLISHing the BCV devices to copy the tracks that have changed while the BCV was SPLIT, the environment is not synchronized or protected. If there is an event that requires recovery, a full volume restore from tape will be required, assuming a tape backup is available.
- Most recoveries will be caused by user or application errors. It may take a while to recognize that the data is corrupted and the logical error might have already been propagated to the Split Mirror Backup environment. A second set of BCVs from a prior point in time may prevent a full volume restore from tape.
- Using multiple sets of BCVs adds the flexibility of creating copies of the system for testing while other copies can serve as a hot standby system that will operate simultaneously.

It is recommended to have at least two sets of BCVs and to toggle between them when creating Split Mirror Backup environments. To eliminate the resynchronization exposure, multiple BCV sets will increase the chances of having a good copy on a BCV to start with in case of a disaster. Restoring the environment from a BCV is significantly faster than restoring from tape.

### 4.3 Backing Up Split Mirror Environments

Once the split-mirror image is created, a backup to tape can be made. The backup must be performed *before* starting the DB2 on the split mirror volumes. Restarting DB2 before making the backup eliminates the option of using this backup as a base for a later point-in-time recovery if required.

A full volume dump to tape of the Split Mirror Backup is recommended. A full volume dump is performed at the volume level. The resulting output tape would normally have a dataset name that reflects the VOLSER that is being backed up. This process requires no renaming of the datasets that are included in the backup.

Alternately, a logical dump may be taken at the dataset level. If a logical dump is taken from the same host, or a different host with shared ICF catalogs, then all datasets must be renamed using the TimeFinder Utility. This rename process is time consuming and recovery from it will take longer than from a full volume dump. If the logical dump is performed from an LPAR that does not share ICF user catalogs with the source LPAR, renaming the dumped datasets is not a requirement.

To prepare for a full volume dump, run the TimeFinder Utility with RELABEL statements to relabel the BCV VOLSERS and vary them on line to the host.

Use the following sample JCL:

```
//EMCTFU      EXEC  PGM=EMCTFU
//SYSOUT      DD    SYSOUT=*
//EMCRENAM    DD    SYSOUT=*
//TFINPUT     DD    *
              RELABEL CUU=4120,OLD-VOLSER=SAPS00,NEW-VOLSER=SAPT00
              RELABEL CUU=4121,OLD-VOLSER=SAPS01,NEW-VOLSER=SAPT01
```

Performing a selective logical dataset recovery from a physical volume backup is discussed in sections Restoring SAP Using DFDSS and Restoring SAP Using FDR.

#### 4.3.1 Backup Using DFDSS

To perform a full volume DFDSS dump, use the following sample JCL:

```
//DUMPA       EXEC  PGM=ADRDSSU
//SYSPRINT    DD    SYSOUT=*
//SYSOUT      DD    SYSOUT=*
//DISK1       DD    UNIT=3390,DISP=SHR,VOL=SER=SAPT00
//TAPE1       DD    DSN=DB2SAP1.FULLBKUP.SAPT00,
//            UNIT=TAPE,DISP=(NEW,KEEP),LABEL=(1,SL),
```

```
//          VOL=SER=TAPE00
//DISK2     DD    UNIT=3390,DISP=SHR,VOL=SER=SAPT01
//TAPE2     DD    DSN=DB2SAP1.FULLBKUP.SAPT01,
//          UNIT=TAPE,DISP=(NEW,KEEP),LABEL=(1,SL),
//          VOL=SER=TAPE01
//SYSIN     DD    *
DUMP FULL INDD(DISK1) OUTDD(TAPE1) -
          OPT(4)
DUMP FULL INDD(DISK2) OUTDD(TAPE2) -
          OPT(4)
```

### 4.3.2 Backup Using FDR

To perform a full FDR volume dump, use the following sample JCL:

```
//DUMP      EXEC PGM=FDR
//SYSPRINT  DD    SYSOUT=*

//DISK1     DD    UNIT=3390,VOL=SER=SAPT00,DISP=OLD
//TAPE1     DD    DSN=DB2SAP1.FULLBKUP.SAPT00,
//          UNIT=TAPE,DISP=(NEW,KEEP),LABEL=(1,SL),
//          VOL=SER=TAPE00
//DISK2     DD    UNIT=3390,VOL=SER=SAPT01,DISP=OLD
//TAPE2     DD    DSN=DB2SAP1.FULLBKUP.SAPT01,
//          UNIT=TAPE,DISP=(NEW,KEEP),LABEL=(1,SL),
//          VOL=SER=TAPE01
//SYSIN     DD    *
DUMP      TYPE=FDR
```

## 4.4 Keeping Track of Dataset Placement on Backup Volumes

The TimeFinder Utility can be run immediately after the Split Mirror Backup is completed to create a recovery report. The rename command to a dummy dataset is used to create the recovery report. The rename will process all volumes in the split mirror, and create a report that contains the mapping of all datasets to their volumes. Information from the recovery report and the tape management system can be used to locate a single dataset for a logical restore even after a backup to tape has been done. This report should be accessible in the event that a recovery is required.

The following sample JCL can be used to produce this report:

```
//EMCTFU    EXEC PGM=EMCTFU
//SYSOUT    DD    SYSOUT=*
//TFINPUT   DD    *
PROCESS CUU=4120,VOLSER=SAPS00,BOTH
PROCESS CUU=4121,VOLSER=SAPS01,BOTH
*
CATALOG DB2SAP21.USER.CATALOG,DEFAULT
SOURCECATALOG DEFAULT=NO,DIRECT=YES
RENAME DUMMY.DATA.SET1.*,
       DUMMY.DATA.SET2.*,
       CATALOG=DB2SAP21.USER.CATALOG
```

A sample of the recovery report is provided below:

```
BAT.MULTIVOL.SAMPLE
VOLUME: BAT241 BAT240
DB2SAP1.BSDS01.DATA
```

```

VOLUME: BAT201
DB2SAP1.BSDS01.INDEX
VOLUME: BAT201
DB2SAP1.BSDS02.DATA
VOLUME: BAT202
DB2SAP1.BSDS02.INDEX
VOLUME: BAT202
DB2SAP11.USER.CATALOG
VOLUME: BAT000
DB2SAP11.USER.CATALOG.CATINDEX
VOLUME: BAT200

```

## 4.5 DB2 Image Copy Considerations

A traditional DB2 image copy may be necessary to guarantee recoverability of some DB2 objects. The following describes the special circumstances that could impact DB2 recovery.

### 4.5.1 Considerations for 32 KB Pages

For tablespaces with 32 KB pages (also 16 KB pages if 3380 devices are used), under certain conditions, the Split Mirror Backup might contain inconsistent pages that cannot be used as a base for a RECOVER LOGONLY recovery. This could happen when a tablespace has non-contiguous extents and the Split Mirror Backup is created at a point where an I/O to the first extent has completed but the I/O to the rest of the page that resides on the second extent, has not yet been issued. The probability this will happen is small. It is possible to reduce the probability even further by forcing a DB2 checkpoint a few minutes prior to creating the split-mirror image. To ensure recoverability of these tablespaces, take occasional traditional DB2 image copies of tablespaces with 32 KB pages.

### 4.5.2 SYSLGRNX and SYSCOPY Recovery Considerations

Performing a recovery using a Split Mirror Backup, requires the archive logs created since DB2 was last shut down normally and prior to the time the Split Mirror Backup was created. DSNDB01.SYSLGRNX entries from that same time are also required. Entries older than that can be deleted from DSNDB01.SYSLGRNX. To delete unneeded entries from DSNDB01.SYSLGRNX use the MODIFY RECOVERY utility. The utility deletes entries from DSNDB01.SYSLGRNX only if it appears that they are not required for recovery, based on information from DSNDB06.SYSCOPY.

When the MODIFY RECOVER utility cleans up recovery information, it deletes outdated image copy entries from DSNDB06.SYSCOPY and the related entries from DSNDB01.SYSLGRNX. If no entries are deleted from DSNDB06.SYSCOPY, no entries will be deleted from DSNDB01.SYSLGRNX. If cleanup of DSNDB01.SYSLGRNX is desired, traditional image copies should be done occasionally to facilitate DSNDB01.SYSLGRNX cleanup.

## 4.6 Operations With LOG NO Considerations

After performing a DB2 operation such as LOAD or REORG TABLESPACE with LOG NO, perform a full DB2 image copy of the tablespace, or take a Split Mirror Backup of the environment. This will ensure recoverability of the affected tablespaces.

The advantage of backing up the environment using a Split Mirror Backup is that it simplifies the recovery process since the same procedures will be used for all objects in the environment.

## 5 SAP Recovery Procedures

When an SAP system recovery is required, a recovery of the entire system to the same point in time is recommended. The most common reason for SAP recovery is a user or application error that corrupted the data. In this case, it is possible to restart the system from a prior Split Mirror Backup. DB2 will roll back or commit all unresolved transactions at restart. Restart is very fast and the DBMS is back on line almost immediately. The amount of data loss depends on how much time passed between the time the split mirror was created and the time the logical corruption occurred.

In most cases it is desirable to recover to the point in time closest to when the database corruption occurred. After deciding to recover to a prior point in time, and determining the RBA of the closest point to the disaster, perform a full DB2 instance recovery. The recovery includes a conditional restart, and uses the Split Mirror Backup as a base applying all logs to it.

Note that if a full instance recovery is to be performed then a Split Mirror Backup, in addition to current recovery structures (BSDS, all archive logs and active logs) is required to complete the recovery process. *Creating a Split-Mirror Backup Environment* on page 26 describes the importance of dataset placement. Before continuing, carefully evaluate the type of restore and select the appropriate method. To be able to roll forward from a Split Mirror Backup, the BSDS, active log, and archive logs must be from the current production DB2 instance. The following procedure describes how to accomplish this:

1. Determine the RBA to which the system should be recovered. To do so, analyze the output from DB2 utilities DSNJU004 and DSN1LOGP.
2. Find a backup that was taken prior to this RBA.
3. If the backup found in step 2 is still on BCVs, run a TimeFinder protected RESTORE to all production volumes except those containing the BSDS and active and archive logs. A protected RESTORE (available in TF v5.0) will preserve the data on the BCV while it is being restored to its standard device. If the backup found in step 2 is no longer on BCVs, perform a full DFDSS or FDR restore from tape to the standard volumes to all production volumes except for those containing active logs, archive logs, and BSDS.

---

For best results, isolate the active logs, archive logs, BSDS and their related ICF user catalogs on a set of separate volumes. Refer to Figure 8 on page 27 for recommended dataset placement.

---

4. Change the DSNZPARM to start the system with DEFER ALL (macro DSN6SPRM).
5. Add a conditional restart record to the BSDS with ENDRBA= (the RBA from step 1).
 

```
//DSNTLOG    EXEC  PGM=DSNJU003
//SYSUT1     DD    DISP=OLD,DSN=DB2SAP1.BSDS01
//SYSUT2     DD    DISP=OLD,DSN=DB2SAP1.BSDS02
//SYSPRINT   DD    SYSOUT=*

//SYSIN      DD    *
CRESTART CREATE, ENDRBA=xxxxxxxxxxxxx, FORWARD=YES, BACKWARD=YES
```
6. Restart DB2 with the new DSNZPARM (conditional restart). DB2 will resolve all units of work that were outstanding at the time of the recovery RBA (ENDRBA). This process involves creating compensation records in the log. When the system is recovered with the LOGONLY option, all log records are applied, including the compensation records.
7. Perform a DB2 catalog and directory RECOVER LOGONLY (in the correct order as documented in the DB2 Administration Guide section *Recovering Catalog and Directory Objects*).

8. Recover the rest of the environment (tablespaces and indexes) with LOGONLY, rebuild indexes if necessary (defined with COPY NO).
9. Change the DSNZPARM to start the system with RESTART ALL the next time it's restarted.
10. RE-ESTABLISH BCVs.
11. Perform a system Split Mirror Backup.

After completing the above steps, the state of the database will be exactly as if DB2 had crashed and been restarted at the time of the specified RBA.

## 5.1 TimeFinder Restore and Parallel Recovery

When a TimeFinder RESTORE is requested, the data is immediately available on the standard volumes even though all of the changed tracks have not yet been written back to the standard devices from the BCVs. If a track that has not been restored is accessed on the standard volume, it will be immediately restored from the BCV. DB2 recovery processes can begin as soon as the TimeFinder restore or the protected RESTORE command has been issued. This parallelism significantly reduces recovery time.

## 5.2 Restoring SAP Using DFDSS

A recovery may be required to a point of consistency that no longer exists on BCVs. In this case, DFDSS may be used to restore the standard volumes from tape to a prior point in time.

### 5.2.1 Full DFDSS Database Restore

To perform a full restore of a volume from tape to a standard volume, use the following JCL:

```
//RESTA      EXEC  PGM=ADRDUSSU
//SYSPRINT   DD    SYSOUT=*
//SYSOUT     DD    SYSOUT=*
//DISK1      DD    UNIT=3390,DISP=SHR,VOL=SER=SAPS00
//TAPE1      DD    DSN=DB2SAP1.FULLBKUP.SAPT00,
//            UNIT=TAPE,DISP=(OLD,KEEP),LABEL=(1,SL),
//            VOL=SER=TAPE00
//DISK2      DD    UNIT=3390,DISP=SHR,VOL=SER=SAPS01
//TAPE2      DD    DSN=DB2SAP1.FULLBKUP.SAPT01,
//            UNIT=TAPE,DISP=(OLD,KEEP),LABEL=(1,SL),
//            VOL=SER=TAPE01
//SYSIN      DD    *
RESTORE INDD(TAPE1) OUTDD(DISK1)
RESTORE INDD(TAPE2) OUTDD(DISK2)
```

If the entire environment is restored (including the ICF user catalogs), no further steps are required. However, if a subset of the SAP environment volumes is restored, then an IDCAMS DEFINE RECATALOG to the cluster page sets on the restored volumes is required to synchronize the ICF user catalogs and the actual volume contents.

### 5.2.2 Logical Dataset Restore Using DFDSS

To perform a logical restore of a single dataset from a full volume dump, follow these steps:

1. Stop the tablespace.

2. Use IDCAMS to delete the VSAM cluster.
3. Perform a dataset restore using the following JCL:

```
//RESTB      EXEC PGM=ADRDSSU
//SYSPRINT  DD   SYSOUT=*
//SYSOUT    DD   SYSOUT=*
//DISK1     DD   UNIT=3390,DISP=SHR,VOL=SER=SAPS00
//TAPE1     DD   DSN=DB2SAP1.FULLBKUP.SAPT00,
//           UNIT=TAPE,DISP=(OLD,KEEP),LABEL=(1,SL),
//           VOL=SER=TAPE00
//SYSIN     DD   *
RESTORE DS(DB2SAP1.DSNDBC.DSNDB01.DBD01.I0001.A001) INDD(TAPE1)
OUTDD(DISK1)
```

4. Perform an IDCAMS DEFINE RECATALOG to the restored cluster dataset.

### 5.3 Restoring SAP Using FDR

A recovery may be required to a point of consistency that no longer exists on BCVs. In this case, FDR may be used to restore the standard volumes from tape to a prior point in time.

#### 5.3.1 Full FDR Database Restore

To perform a full restore of a volume from tape to the standard volume use the following JCL:

```
//RESTA     EXEC PGM=FDR
//SYSPRINT  DD   SYSOUT=*
//DISK1     DD   UNIT=3390,VOL=SER=SAPS00,DISP=OLD
//TAPE1     DD   DSN=DB2SAP1.FULLBKUP.SAPT00,
//           UNIT=TAPE,DISP=(NEW,KEEP),LABEL=(1,SL),
//           VOL=SER=TAPE00
//DISK2     DD   UNIT=3390,VOL=SER=SAPS01,DISP=OLD
//TAPE2     DD   DSN=DB2SAP1.FULLBKUP.SAPT01,
//           UNIT=TAPE,DISP=(NEW,KEEP),LABEL=(1,SL),
//           VOL=SER=TAPE01
//SYSIN     DD   *
RESTORE    TYPE=FDR
```

#### 5.3.2 Logical Dataset Restore Using FDR

To perform a logical restore of a single dataset from a full volume dump, use the following JCL after stopping the tablespace:

```
//RESTB     EXEC PGM=FDRDSF
//SYSPRINT  DD   SYSOUT=*
//TAPE1     DD   DSN=DB2SAP1.FULLBKUP.SAPT00,
//           UNIT=TAPE,DISP=(NEW,KEEP),LABEL=(1,SL),
//           VOL=SER=TAPE00
//DISK1     DD   UNIT=3390,DISP=SHR,VOL=SER=SAPS00
//SYSIN     DD   *
RESTORE    TYPE=DSF
SELECT    DSN=DB2SAP1.DSNDBC.DSNDB01.DBD01.I0001.A001
```

An IDCAMS DEFINE RECATALOG is *not* required when restoring with FDR.

## 5.4 Optimizing SAP Recovery

When performing a recovery using RECOVER LOGONLY, DB2 reads the HPGRBRBA recovery base RBA from the header page of each page set. Then, in one pass on the log, every update to a page set that is detected is re-applied to the appropriate page set. Although not all the page sets have had updates since the split mirror was created, DB2 must perform a dynamic allocation to *every* page set, read the header page, and determine the starting point for the log apply. If the frequency of creating the split mirror is high, most of the page sets probably had no updates at all.

Using RECOVER LOGONLY on all 40,000 SAP page sets (tablespaces and indexes) may take hours, even if there were few updates only to a few page sets since the Split Mirror Backup environment was created. To make the recovery process more efficient, use the following procedure to limit the set of objects to be recovered to updated page sets only. This can reduce the recovery time from hours to minutes. Recovery time depends on how active the system is and the size of the log range to be processed.

To optimize SAP recovery follow these steps:

1. Determine the RBA to which the system should be recovered. To do so, analyze the output from DB2 utilities DSNJU004 and DSN1LOGP.
2. Find a backup that was taken prior to this RBA.
3. Run a DSNJU004 report and find the starting RBA of the third checkpoint before the time of the split mirror found in step 2.
4. Run a DSN1LOGP with RBASTART=(RBA from step 3), RBAEND=(RBA from step 1), and SUMMARY(ONLY). Use the following sample JCL:

```
//STEP1      EXEC PGM=DSN1LOGP
//SYSSUMRY   DD   SYSOUT=*
//SYSPRINT   DD   SYSOUT=*

//BSDS       DD   DSN=DB2SAP1.BSDS01,DISP=SHR
//SYSIN      DD   *
              RBASTART(RBA_from_step_3) RBAEND(RBA_from_step_1) SUMMARY(ONLY)
```

This report lists the completed and running events in this log range, including modified page sets. These modified page sets are the page sets that need to be recovered.

5. If the backup found in step 2 is still on BCVs, run a TimeFinder protected RESTORE to all volumes *except* volumes containing the BSDS, and active and archive logs. If the backup found in step 2 is no longer on BCVs, perform a full DFDSS or FDR restore to all volumes except those containing active logs, archive logs, and BSDS.

---

For best results, isolate the active logs, archive logs, BSDS, and their related ICF user catalogs on a set of separate volumes.

---

6. Change the DSNZPARM to start the system with DEFER ALL (macro DSN6SPRM).
7. Add a conditional restart record to the BSDS with ENDRBA= (the RBA from step 1).

```
//DSNTLOG    EXEC PGM=DSNJU003
//SYSUT1     DD   DISP=OLD,DSN=DB2SAP1.BSDS01
//SYSUT2     DD   DISP=OLD,DSN=DB2SAP1.BSDS02
//SYSPRINT   DD   SYSOUT=*
//SYSIN      DD   *
              CRESTART CREATE,ENDRBA=xxxxxxxxxxxxxx
```

8. Restart the DB2 with the new DSNZPARM (conditional restart).

9. Perform a DB2 catalog and directory RECOVER LOGONLY (in the correct order as documented in the DB2 Administrators Guide section *Recovering Catalog and Directory Objects*) only to the objects that are in the list created in step 4.
10. Recover the rest of the page sets from the list created in step 4 (tablespaces and indexes) with LOGONLY; rebuild indexes if necessary (defined with COPY NO).
11. Change the DSNZPARM to start the system with RESTART ALL the next time it is restarted.
12. Perform a system Split Mirror Backup.

## Appendix A

This REXX takes the output of DSNJU004, parses it, and creates a DSNJU003 job to update the new BSDS with the new VCAT name and the new log datasets.

```

/*----- REXX -----*/
/*
/* member      : ju004
/*
/* type        : edit macro
/*
/* description : reads the output from DSNJU004 and creates a
/*               DSNJU003 JCL with NEWLOG commands with correct
/*               RBAs + NEWCAT command to change the VCAT name.
/*
/* parameters  : THLQ - target high-level qualifier.
/*
/*-----*/

ADDRESS TSO "NEWSTACK"
ADDRESS ISREDIT
  'MACRO (THLQ) PROCESS'
PARSE UPPER VAR THLQ THLQ

IF THLQ = '' THEN
DO
ZEDSMMSG = 'MISSING PARAMETER'
  ZEDLMSG = 'TARGET HIGH-LEVEL QUALIFIER IS MISSING'
  ADDRESS ISPEXEC "SETMSG MSG(ISRZ001)"
  EXIT 12
END

ADDRESS ISPEXEC "VGET (ZUSER) ASIS"
MSG_STATUS = MSG("OFF")
ADDRESS TSO "FREE F(DSNJU004)"
ADDRESS TSO "DELETE '"ZUSER".TEMP.JU004'"
MSG_STATUS = MSG("ON")

/*-----*/
/* CREATE JOB CARD
/*-----*/
ADDRESS TSO "NEWSTACK"
LINE= '//DSNTLOG EXEC PGM=DSNJU003'
QUEUE LINE
LINE= '//SYSUT1 DD DISP=OLD,DSN='THLQ'.BSDS01'
QUEUE LINE
LINE= '//SYSUT2 DD DISP=OLD,DSN='THLQ'.BSDS02'
QUEUE LINE
LINE= '//SYSPRINT DD SYSOUT=*'
QUEUE LINE
LINE= '//SYSIN DD *'
QUEUE LINE
LINE= ' NEWCAT VSAMCAT='THLQ
QUEUE LINE

```

```

ADDRESS ISREDIT
  'X ALL'
  'BNDS 1 122'
  'FIND LOGCOPY1 ALL'
  '(LOG1DS) = FIND_COUNTS'
  'FIND LOGCOPY2 ALL'
  '(LOG2DS) = FIND_COUNTS'
  'DEL X ALL'
  'CHA P"=" 1 " " ALL'

IF (LOG1DS  $\neq$  LOG2DS) THEN
DO
  SAY 'SEVERE ERROR: LOGCOPY1 DS NOT EQUAL LOGCOPY2 DS'
  ADDRESS TSO "DELSTACK"
  EXIT 12
END

DO I=1 TO LOG1DS BY 1
  IF I<10 THEN
    TEXT = 'LOGCOPY1.DS0' || I
  ELSE
    TEXT = 'LOGCOPY1.DS' || I
  'FIND &TEXT FIRST'
  'ISREDIT (CURROW,CURCOL) = CURSOR'
  'ISREDIT (LOGLINE) = LINE 'CURROW
  PARSE UPPER VAR LOGLINE STARBA ENDRBA DUM1 'DSN=' SHLQ '.LOGCOPY' DUM2
  LINE= ' DELETE DSNAME='SHLQ'.'TEXT
  QUEUE LINE
  LINE= ' NEWLOG DSNAME='THLQ'.'TEXT',COPY1,STARTRBA='STARBA', '
  QUEUE LINE
  LINE= '
  QUEUE LINE
  ENDRBA='ENDRBA
END

IF LOG2DS > 0 THEN
DO I=1 TO LOG2DS BY 1
  IF I<10 THEN
    TEXT = 'LOGCOPY2.DS0' || I
  ELSE
    TEXT = 'LOGCOPY2.DS' || I
  'FIND &TEXT FIRST'
  'ISREDIT (CURROW,CURCOL) = CURSOR'
  'ISREDIT (LOGLINE) = LINE 'CURROW
  PARSE UPPER VAR LOGLINE STARBA ENDRBA DUM1 'DSN=' SHLQ '.LOGCOPY' DUM2
  LINE= ' DELETE DSNAME='SHLQ'.'TEXT
  QUEUE LINE
  LINE= ' NEWLOG DSNAME='THLQ'.'TEXT',COPY2,STARTRBA='STARBA', '
  QUEUE LINE
  LINE= '
  QUEUE LINE
  ENDRBA='ENDRBA
END

Q = QUEUED()
IF Q > 0 THEN
DO
  ADDRESS TSO
    "ALLOC F(DSNJU004) DA('"ZUSER".TEMP.JU004') UNIT(SYSDA)

```

```
                TRACKS SPACE(1 5) RECFM(F B) LRECL(80)
                DSORG(PS) CATALOG"
ADDRESS TSO
  "EXECIO "Q" DISKW DSNJU004 (FINIS)"
ADDRESS ISPEXEC
  "EDIT DATASET(' "ZUSER".TEMP.JU004')"
ADDRESS TSO
  "FREE  F(DSNJU004)"
END
ADDRESS TSO "DELSTACK"
EXIT 0
```

## Appendix B

The following REXX could be used to add commits to a long ALTER job created by the SQL in the cloning section. If commits are not added, the job might end abnormally.

```

/*----- REXX -----*/
/*
/* member      : commit
/*
/* type        : edit macro
/*
/* description : repeatedly inserts lines from stem var text into
/*               the member with <space> spacing.
/*
/*-----*/
address isredit
  'macro (space)'
  '(lastline) = linenum .zlast'

if (space = '') then space = 10
i = 0

text.1 = 'COMMIT;'
text.2 = 'SET CURRENT SQLID=''SAPR3'';'

n = 1
do while (text.n ^= 'TEXT.'n)
  n = n + 1
end
text.0 = n-1

do while (i < lastline)
  j = text.0
  do while (j >0)
    text = text.j
    address isredit 'line_after 'i' = (text)'
    j = j - 1
  end
  i = i + space + text.0
  lastline = lastline + text.0
end

exit

```

## References

The information in this document was compiled from the following sources:

- *DB2 FOR OS/390 Administration Guide: System, Version 6*, Second Edition, SC26-9003-01, International Business Machines Corporation, 2000.
- *DB2 FOR OS/390 Utility Guide and Reference Version 6*, Second Edition, SC26-9015-01, International Business Machines Corporation, 2000.
- *DB2 FOR OS/390 Installation Guide Version 6*, Second Edition, GC26-9008-01, International Business Machines Corporation, 2000.
- *DB2 FOR OS/390 SQL Reference Version 6*, Second Edition, SC26-9014-01, International Business Machines Corporation, 2000.
- *R/3 Homogeneous System Copy, UNIX, Windows NT, OS/390, OS/400*, Release 4.6B, SAP.
- *EMC TimeFinder Product Set, Version 4.1, Revision B*, Part Number 300-999-154.
- *EMC ConGroup for MVS, Version 3.3.1, Revision C*, Part Number 300-999-031.