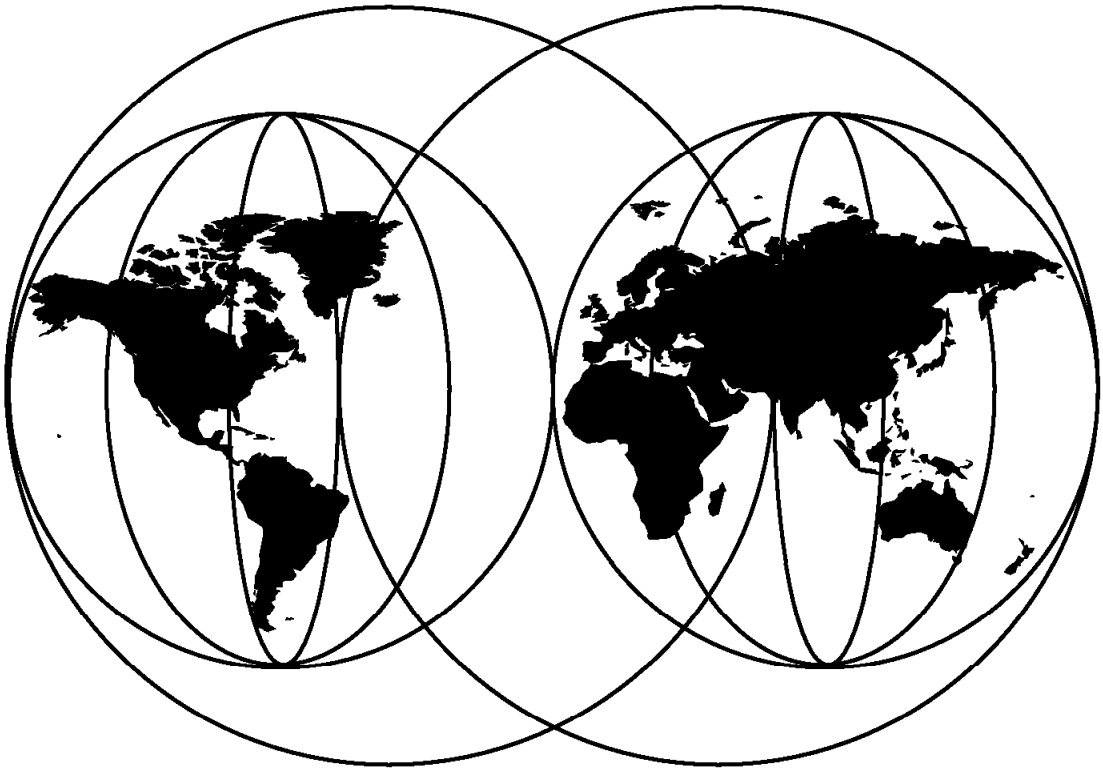
Using RVA and SnapShot for Business Intelligence Applications with OS/390 and DB2

Kathryn Arrell, Paolo Bruni, Robert Catterall,
Mike Downie, Alison Pate, Lee Siegmund



International Technical Support Organization

http://www.redbooks.ibm.com

This book was printed at 240 dpi (dots per inch). The final production redbook with the RED cover will be printed at 1200 dpi and will provide superior graphics resolution. Please see "How to Get ITSO Redbooks" at the back of this book for ordering instructions.

SG24-5333-00

International Technical Support Organization

SG24-5333-00

Using RVA and SnapShot for Business Intelligence Applications with OS/390 and DB2

August 1998

Take Note!

Before using this information and the product it supports, be sure to read the general information in Appendix A, “Special Notices” on page 47.

First Edition (August 1998)

This edition applies to DB2 V5 and SnapShot V1 R2 for use with the OS/390 V2.5.

Comments may be addressed to:
IBM Corporation, International Technical Support Organization
Dept. HYJ Mail Station P099
522 South Road
Poughkeepsie, New York 12601-5400

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.

Contents

Figures

Preface

This redbook is designed to answer questions about how you can use the RAMAC Virtual Array (RVA) and SnapShot in a Business Intelligence (BI) environment on OS/390 using DB2. It is written in the form of a question and answer document. It is based on the experiences of the Teraplex Center in Poughkeepsie, the ITSO in San Jose and customer environments. It is not an in-depth analysis of the environment but detailed material is referenced where it is available.

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 Poughkeepsie Center.

Kathryn Arrell is an ERP Specialist at the International Technical Support Organization, Poughkeepsie Center.

Paolo Bruni is a DB2 Specialist at the International Technical Support Organization, San Jose Center.

Robert Catterall is a DB2 Specialist with the Dallas Systems Center, North American Advanced Technical Support.

Mike Downie is a Storage Instructor with IBM Education and Training. He has six years experience with RVA technology and has been using and training customers on the use of SnapShot since its release.

Alison Pate is a Storage Specialist at the International Technical Support Organization, San Jose Center.

Lee Siegmund is a DB2 Specialist with the Dallas Systems Center, North American Advanced Technical Support.

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

Steven Turner
ITSO, Poughkeepsie

Steve Carbaugh
Andrea Harris
Nin Lei
Luanne McDonough
Dino Tonelli
Teraplex Center, Poughkeepsie

Comments Welcome

Your comments are important to us!

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

- Fax the evaluation form found in “ITSO Redbook Evaluation” on page 59 to the fax number shown on the form.
- Use the electronic evaluation form found on the Redbooks Web sites:

  For Internet users `http://www.redbooks.ibm.com/`
  For IBM Intranet users `http://w3.itso.ibm.com/`

- Send us a note at the following address:

  `redbook@us.ibm.com`

Chapter 1. What Is an RAMAC Virtual Array (RVA)?

This section gives the reader who is unfamiliar with the RAMAC Virtual Array (RVA) an overview of the device architecture and its unique SnapShot function.

1.1 Overview of RAMAC Virtual Array

The RVA is a high-performance RAID 6 storage subsystem. RAID 6 protects against data loss even in the unlikely event that two disks are lost in the array. The RVA RAID implementation has automatic hot-sparing: if a disk fails, the data on that disk is immediately recreated on a spare disk in the array, thus restoring RAID protection. The failed disk can be nondisruptively replaced and becomes the new spare.

The RVA RAID implementation is transparent to users and, unlike most RAID 5 or RAID 6 implementations, has no adverse effect on performance thanks to the advanced architecture of the RVA.

Traditional storage subsystems, such as the 3880, 3990, and RAMAC Array DASD family, use the count key data (CKD) architecture. The CKD architecture defines how and where data is stored on the disk device. Any updates to the data are written directly to the same position on the physical disk from which the data was retrieved prior to the update. This is referred to as *update in place*.

IBM′s RVA provides a high-availability, high-performance storage solution, thanks to its revolutionary virtual disk architecture. To the host, the RVA appears as up to four traditional 3990 subsystems, with up to 256 3380 and/or 3390 volumes. These devices do not physically exist in the subsystem and are referred to as *functional devices*. In reality, the subsystem contains RAID-protected arrays of fixed block architecture (FBA) disk devices.

The RVA translates I/O requests from CKD format to FBA format.

1.2 Log Structured File

The RAID-protected FBA disk arrays that make up the RVA′s physical disk space are sequentially filled with data. New and updated data is placed at the end, as it is on a sequential or log file. We call this a *log structured file*.

Updates leave areas of data that are no longer needed in the RVA log file. A microcode recycle process called *freespace collection* ensures that these areas are removed so that there is always enough freespace for writing. (This is similar to a reorg of the data.) You can control the freespace by observing the net capacity load (NCL) of the RVA and using the IBM Extended Facilities Product (IXFP) program. The RVA′s physical disk space typically should not be filled above 75% NCL. Above that level, the freespace collection process runs with higher priority, and performance degradation may result.

The following tables are used to map the tracks of functional devices to the FBA blocks related to those tracks.

Figure 1 on page 2 shows the virtual disk architecture.

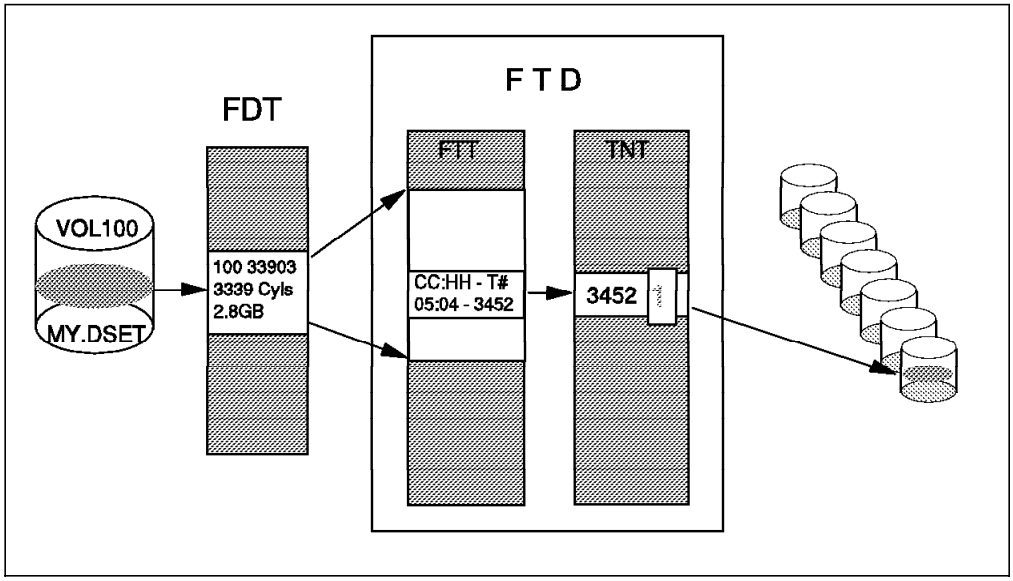


Figure 1. Virtual Disk Operation. In the first FDT box, 100 33903 stands for Volume 100 of a 3390 model 3. In the FTT box, CC is cylinder 5, HH is head 4, T is track 3452.

- Functional device table

  The functional device table (FDT) holds the information about the functional volumes that have been defined to the RVA.

- Functional track directory

  The functional track directory (FTD) is the collective name for two tables that together map each functional track to an area in the RVA's physical storage:

  – Functional track table

    The functional track table (FTT) contains the host-related, that is, the functional device-related track pointers of the FTD.

  – Track number table

    The track number table (TNT) contains the back-end data pointers of the FTD. A reference counter is also part of this table.

Although the FTD consists of two tables, we discuss it as one entity. In fact, each functional track has an entry in the FTD. If a functional track contains data, its associated FTD entry has a pointer to the FBA block where the data starts. The data of a functional track may fit on one or more FBA blocks.

If the data of a functional track is updated, the RVA puts the new data in a new place in the log structured file, and the FTD entry points to the new data location. There is no update in place.

See the RVA redbook *IBM RAMAC Virtual Array*, SG24-4951, for more information.

Chapter 2. What Is the Effect of Data Compression and Compaction?

Because update in place does not occur in the log structured file, data can be compressed. In fact, all data, as soon as it enters an RVA, is compressed (to the extent possible). In addition, the gaps between the records, as they appear on the CKD devices, are removed before the data is written to disk. This is called *compaction*. Experience shows that RVA customers can achieve an overall compression and compaction ratio of 3.6:1.

2.1 Effect of RVA Data Compression

The effect of data compression on the RVA subsystem is that it speeds up the transfer of data to and from the back-end disk arrays to cache and it increases the effectiveness of the cache memory since more data can fit in cache due to its compression. The subsystem Effective Capacity attempts to give users a suggested guideline as to how much traditional non-compressed data will fit in the RVA subsystem based on the current compression algorithm.

Many types of data such as documents, JCL, batch and TSO data compress very well in the RVA. Other applications may sometimes use host compression or internal compression software to achieve the same benefits of compressed data for storage and transmission. This data, when placed on the RVA, will be compressed internally as is all data which enters the RVA subsystem.

However, data which is very random in nature either by content or because of previous compression will not compress as well within the RVA as non-compressed data. Therefore, the user should understand that if the only data that is to be placed on an RVA susbsystem is pre-compressed or highly random numerical data, there may not be as good internal compression as expected and the Effective Capacity of the RVA, which is based on non-compressed data, may be inaccurate. If the compression of the data to be placed on the RVA is known or can be approximated, this number can be used times the physical capacity of the RVA disk arrays to give an approximate effective capacity value for that data.

If the user needs to verify the compression of the data, either because of the unusual type of data, or just because of the large DASD space involved, the San Jose Storage Systems Advanced Technical Support Center (San Jose ATSC) can be contacted, through the IBM marketing support line, at the userid RVATOOL at LSAVMIC1. Support personnel have the skill and the internal tools necessary to verify in advance the effective capacity of the RVA if they have samples of the users′ real application data.

If the DA Capacity of the RVA as shown on the Subsystem Configuration panel or Local Operator Panel of the subsystem is 117.4GB, then the effective capacity of that subsystem is generally thought to be 420GB based on a 3.6 compression.

If the data to be stored on that RVA shows 3.0 compression, then the effective capacity would be 352GB.

If the data to be stored only compressed 2.5 to one, then the effective capacity of the subsystem would only be 293GB.

These numbers would represent the amount of data that should fit in 75% NCL of the RVA subsystem.

Chapter 3. What is the Effect of RVA Compression with DB2 Data?

This chapter describes the results of DASD occupancy obtained experimenting with DB2 data on RVA, and some general recommendations on RVA usage in DB2 environments.

3.1 Effects of Compression

The values of DASD occupancy reported in this section are a subset of the results of measurements that have taken place at the Teraplex Center. The objective of the measurements was to evaluate the impact on performance of DB2 and RVA compression. For a description of the environment and more details on the results of the measurements see the redbook *DB2 for OS/390 and Data Compression*, SG24-5261, which will be available in 4Q98.

The charts in this section give a pictorial representation of DASD savings and relative DASD space occupancy for DB2 objects in the three following cases when compared to the base case of non-DB2 compressed data on a traditional 3390:

- Only DB2 compression
- Only RVA compression
- Both DB2 and RVA compression,

The values reported are based on the information provided by the VSAM catalog and the IXFP Space Reports for datasets with no (or minimal) overallocation.

The tablespace referenced in this section is the same one utilized for the SnapShot implementation in Chapter 7, “How Can I Use SnapShot to Extract Data from DB2?” on page 27.

DASD Allocations of DB2 Objects: The chart in Figure 2 on page 6 summarizes the values relative to DASD occupancy reported in megabytes (MB) for data and indexes; they are the base for the comparative charts reported later.

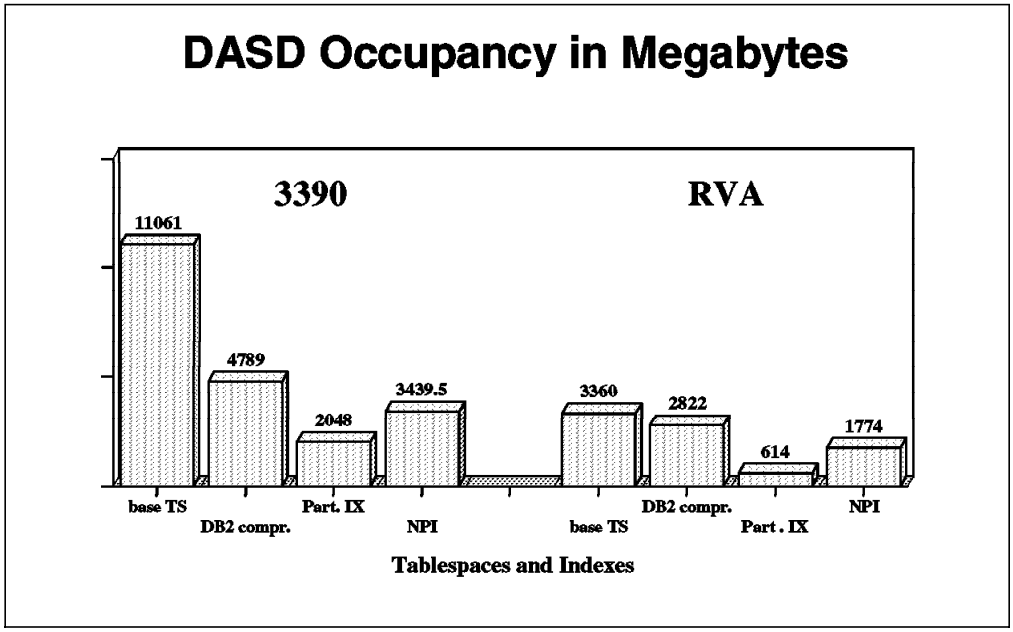


Figure 2. DB2 Objects DASD Occupancy

The set of bars on the left represents the DASD occupancy in megabytes of the table space, base and DB2 compressed, and the two index spaces, relative to partitioning and non-partitioning index, when allocated on a 3390.

The set of bars on the right represents the DASD occupancy in megabytes of the tablespace, base and DB2 compressed, and the two indexspaces, when allocated on an RVA, and automatically RVA compressed.

Tablespace: Figure 3 summarizes the statistics relative to tablespace occupancy showing the relative size in percentage, and provides a quick view of the results of compression.

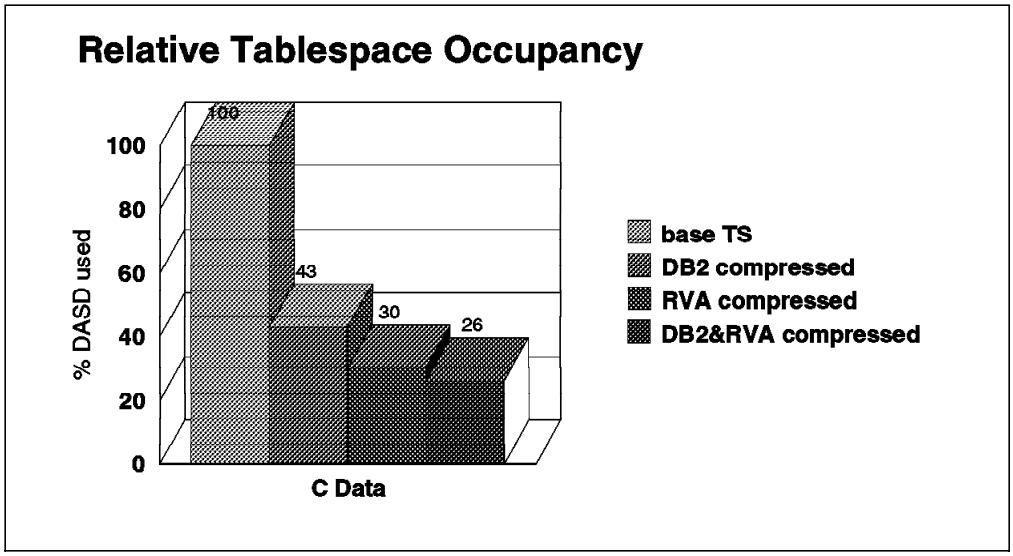


Figure 3. Compressing the Tablespace

The S/390 compression algorithm used by DB2 reduces the occupied space to 43%, with a saving of 57%. RVA's algorithm performs slightly better and reduces the space to 30%, with a saving of 70%. When the DB2 compressed

tablespace is allocated on RVA, the RVA compression introduces a further reduction in space to a compounded 26%, with a total saving of 74%.

This chart suggests that DB2 and RVA compression algorithms give results in the same ballpark, with RVA being able to perform better and add compression to DB2 compressed data.

Indexspaces: Figure 4 summarizes the statistics relative to DASD occupancy for the partitioning and non-partitioning index.

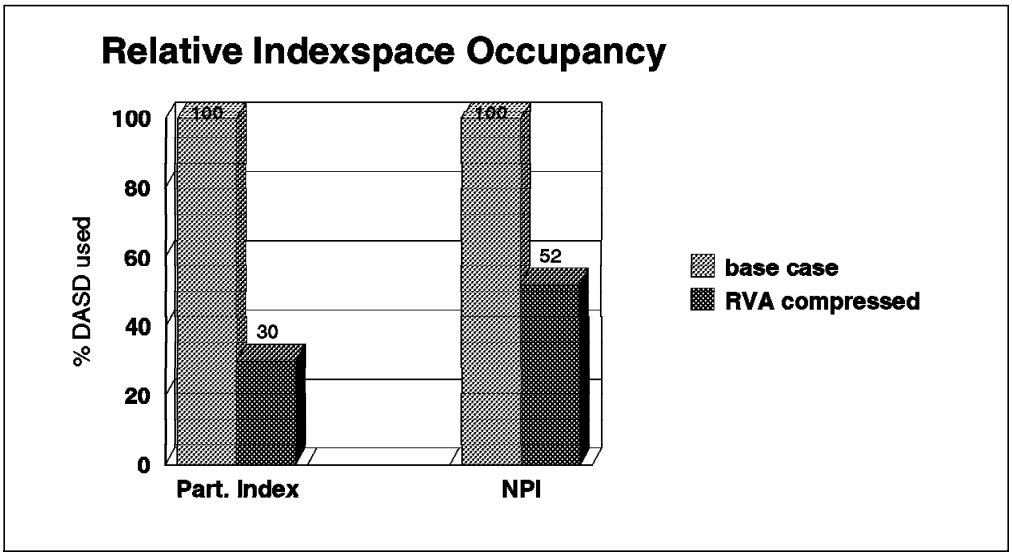


Figure 4. Compressing the Indexes

DB2 does not compress indexes; the savings are provided only by the RVA device and are not as high as one could expect in the case of the non-partitioning index because of the particularly random contents of the chosen key fields. RVA compaction enhances the benefits in DASD savings for the indexes.

This chart points out the compression that can be obtained by RVA on DB2 indexes and shows that for random strings, like the ones used for the NPI, the compression value may be less than anticipated.

Overall DASD Savings: The following charts look at DASD allocations overall, considering data and indexes.

Figure 5 on page 8 summarizes the relative comparisons for tablespace and indexspace savings, and shows also the overall relative percentage.

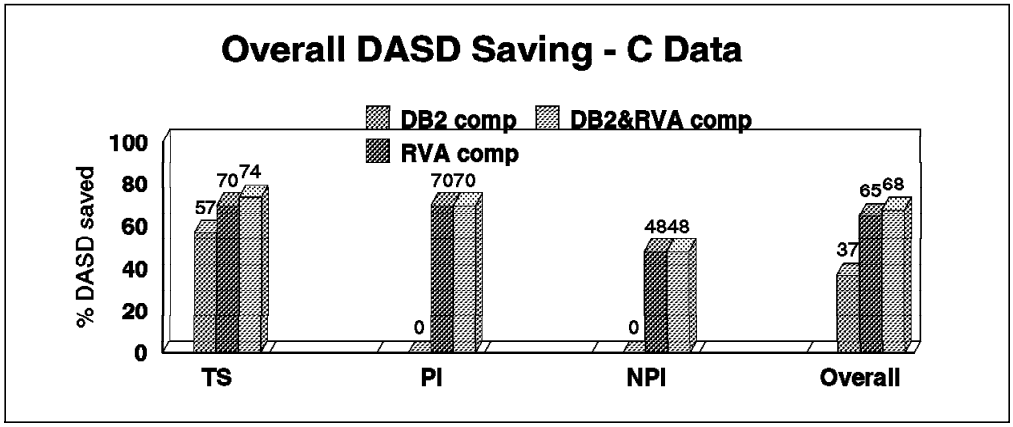


Figure 5. Overall DASD Savings

This chart highlights that, when taking into account the presence of indexes, and considering primarily DASD space allocation, savings provided by the RVA device are more evident. This could be particularly advantageous in environments like data warehousing, where multiple and even larger indexes could be present.

RVA and Archive Logs: The chart in Figure 6 shows compression value and related DASD savings for an archive log allocated on an RVA and produced while inserting and deleting rows from the base tablespace.

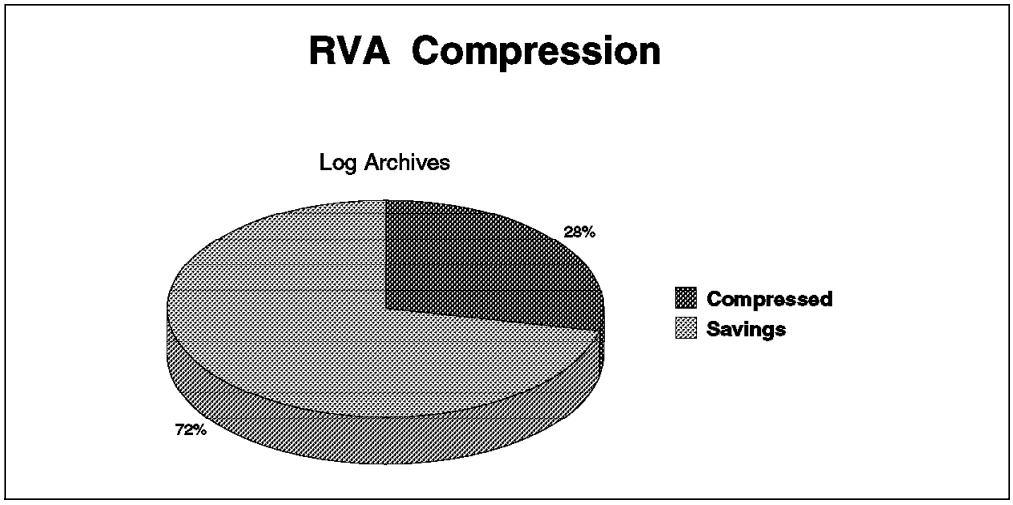


Figure 6. DB2 Archive Log Compression on RVA

This chart reports a compression ratio for the archive log of about 4:1 and highlights the increased possibility of archiving on DASD before destaging on cassettes in order to reduce recovery time for critical applications.

RVA or DB2 Compression?: If we divide the question into three parts, it is expressed by:

- RVA Compression?

  It is really not a question since RVA always compresses, and unless the data is unsuitable (due to previous compression, or random contents, or encryption), the compression ratio is generally over 3:1. The device, besides a compression that is transparent to DB2 and applicable to all DB2 objects,

has characteristics that provide further benefits for DB2, mainly reducing data and storage management operational costs by:

  - Minimizing out of space abends by allowing over allocation at no expense
  - Eliminating the need to reorg to reduce fragmentation
  - Spreading data across more volumes, increasing the available space per volume
  - Reducing manual data set reallocation for tuning
  - Possibly reducing volume queueing
  - Duplicating data instantly
  - Allowing an affordable solution to keeping online archive logs for faster recovery

- DB2 Compression?

  The decision to use DB2 compression is normally based on the necessity of reducing DASD occupancy and I/O for large tablespaces in environments that are non-CPU constrained. The benefits are reduced elapsed times and I/O utilization, a higher hit ratio, a tremendous cascading effect that reduces the amount of DB2 data copied across DASD and cartridges for image copies, logs, archives, remote copies, dumps, and so on. These considerations mostly still apply.

- DB2 and RVA Compression?

  If the environment is not CPU constrained, there are definite benefits in using both types of compression together.

3.2 RVA Performance and DB2

This section reports some general recommendations for improving DB2 performance with RVA. However, not all workloads and environments are the same and these suggestions cannot be effective for all of them.

1. SEQCACHE=SEQ in DSNZPARM

   Generally Business Intelligence applications tend to scan large volumes of data. DB2 uses sequential or dynamic prefetch for this type of processing, and the default option for the cache mode of the DASD controller is set to bypass, because this prevents small cache units from being overutilized by DB2. However, with RVA devices, most users should set the cache mode to sequential, exploiting the capacity of the controller, in order to improve prestaging and to try to achieve a high cache hit ratio.

2. DFSM Dynamic Cache Management Enhanced (DCME)

   If ′May Cache′ of DFSMS DMCE is used, some I/Os can still be affected if the Inhibit Cache Load (ICL) indicator bit is set to on. Even with a large cache available, it may be necessary to force ′Must Cache′ for the DB2 Storage Classes (parameters MSR=5 for random and sequential and BIAS=WRITE).

3. DB2 Vertical Deferred Write Threshold (VDWQT)

   This is the percentage of virtual pool buffers that might be used by updated pages from a single data set. The default of 10% may produce interspersed large bursts of DB2 writes with relatively long periods of inactivity in between; this causes contention on the RVA and it reduces the read hit ratio.

The VDWQT should be reduced to 1% (or even zero) to cause more frequent and more uniformly distributed writes, which will create a higher probability of cache hits.

Chapter 4. What Is SnapShot?

The RVA′s virtual disk architecture enables you to produce almost instantaneous copies of data sets, and volumes. We call this function SnapShot. Snapshot produces copies without data movement. We call making a copy with SnapShot *snapping* the data. The term *snap* also refers to the resulting copy of the data.

Conventional methods of copying data on DASD consist of making a physical copy of the data on either DASD or tape. Host processors, channels, tape, and DASD controllers are involved in these conventional copy processes. Copying may take a lot of time, depending on the availability of system resources.

With SnapShot, copying is achieved with no data movement. Snapping can take just seconds rather than minutes or hours, because data is not moved and the host processor and channels are not involved in data transfer. Furthermore, additional physical disk space is not required to accommodate the snap until either the source or the target is updated. As far as the operating system is concerned, the snap is a real copy of the data; as far as the RVA hardware is concerned, it is a virtual copy of the data.

4.1 How SnapShot Works

A functional device is represented by a certain number of pointers in the FTD. Every used track has a pointer to its back-end data in the FDT.

The RVA′s virtual disk architecture enables you to make a copy of a data set or volume by copying its FTD pointers. As you can imagine, this is a very fast process. No data movement takes place, and no additional back-end physical space is used. Both FTD pointers, the original and the copy, point to the same physical data location. Figure 7 shows how SnapShot works.

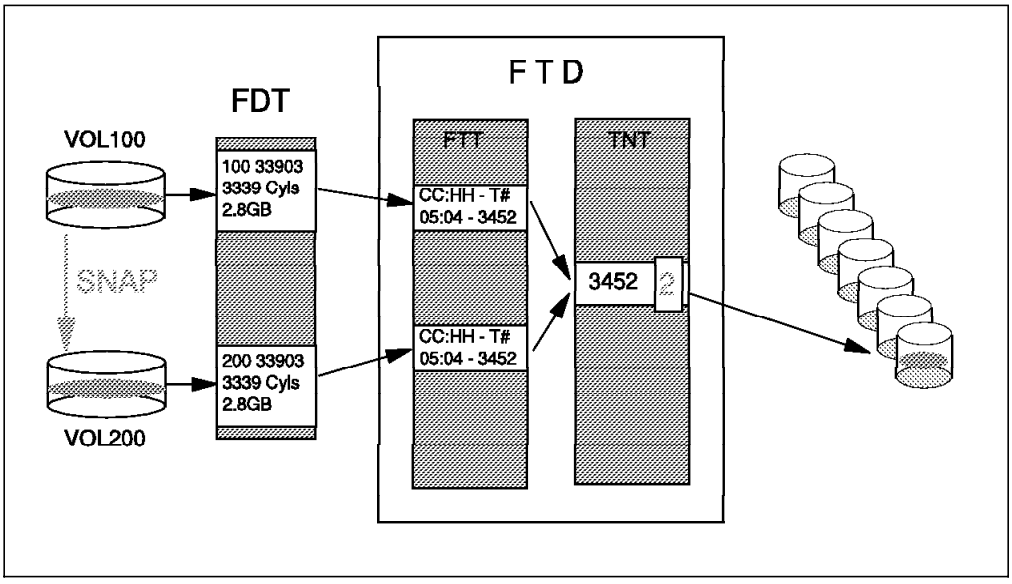


Figure 7. SnapShot Operation

Only when either the original or the copied track is updated is its associated FTD pointer changed to point to the new data location. The other FTD pointer remains unchanged. Additional space is needed in this case. As long as there

is a pointer to a data block in the physical disk storage, the block cannot become free for the freespace collection process. A reference counter in the track number table (TNT) prevents the block from becoming free.

On the RVA side, the SnapShot function is quite simple; on the OS/390 side, some additional operations are necessary. If a data set is snapped, a new data set name is required and given in the snap command. After the RVA performs the snap, OS/390 must update the catalog and VTOC (and the VVDS if the snap is performed on a VSAM data set).

Although the creation of the copy is a logical manipulation of pointers, the data exists on disk, so in this sense it is not a logical copy. As far as the operating system is concerned, the two copies are completely independent of each other.

When you make a copy with SnapShot, both the source and the target must be within the same RVA.

For further information about SnapShot, see *Implementing SnapShot*, SG24-2241.

4.2 What Is Virtual Concurrent Copy?

In this section we describe the functions of virtual concurrent copy and DFSMSdss SnapShot. These functions combine the data duplication function of IBM SnapShot for MVS/ESA with the flexibility and operability of DFSMSdss. Both functions are delivered through a small programming enhancement (SPE) to DFSMSdss.

Integration of SnapShot with DB2 is provided through virtual concurrent copy.

4.2.1 DFSMSdss SnapShot

DFSMSdss SnapShot enables you to take advantage of SnapShot when you are copying data within a single RAMAC Virtual Array (RVA). When you specify DFSMSdss COPY, and both the source and target data sets are in the same RVA, SnapShot is automatically invoked.

There are no changes to JCL, and your copy jobs instantly benefit from SnapShot, both in terms of the speed of the copies and the fact that SnapShot does not use any additional back-end space to make a copy.

4.2.2 Virtual Concurrent Copy

Virtual concurrent copy extends the benefits of SnapShot to DB2 users. DB2 already has integrated support for 3990 concurrent copy through the CONCURRENT keyword in a COPY utility.

When you specify the CONCURRENT keyword on a DB2 image copy, the DFSMS/MVS software can detect whether you have an RVA. If you have an RVA with SnapShot, the virtual concurrent copy function is invoked.

Concurrent copy enables you to take a consistent copy or dump of data, using DFSMSdss, while applications are updating the data.

The logical completion of the point-in-time copy occurs when the source data is snapped into an interim data set called the *working space data set* (WSDS).

Because there is no actual movement of data, snapping can take seconds rather than minutes or hours, and host processor and channels are not involved in the data transfer.

The physical completion occurs when the data is moved by DFSMSdss to the target tape or disk data set. Once the copy is logically complete, the data can be made available for application updates.

Figure 8 shows the virtual concurrent copy operation.

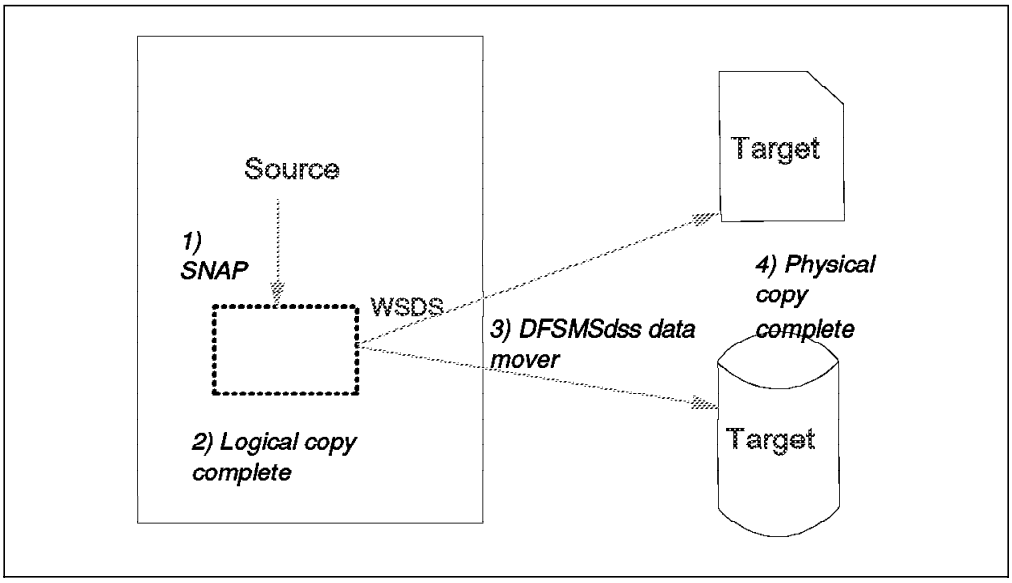


Figure 8. Virtual Concurrent Copy Operation. One target is a copy to disk and the other target is a copy to a tape cartridge.

If you are already using concurrent copy, you do not have to make changes to your JCL to use virtual concurrent copy.

For further information on virtual concurrent copy, see *Implementing DFSMSdss SnapShot and Virtual Concurrent Copy*, SG24-5668.

Chapter 5. How Can I Use SnapShot to Reduce My Backup Window?

With virtual concurrent copy, DB2 users can now exploit SnapShot for DB2 image copies. Using virtual concurrent copy enables you to:

- Make a SHRLEVEL REFERENCE copy of a single table space, within a very short amount of time during which the source object cannot be updated.
- Make a SHRLEVEL REFERENCE copy of a group of table spaces, within a very short amount of time during which the source objects cannot be updated.
- Quickly replicate large amounts of data for application testing purposes.

5.1 Requirements for Using Virtual Concurrent Copy with DB2

The ease with which you can exploit virtual concurrent copy with DB2 depends on the level of DB2 and the version of DFSMSdss you are using.

DB2 Version 3 is the minimum level that enables you to recover, using the DB2 RECOVER utility, from a copy taken outside DB2′s control. DB2 Version 4 is the minimum level that enables you to image copy and recover your DB2 data by utilizing DFSMSdss and SnapShot through DB2 utilities.

To minimize the outage when using virtual concurrent copy to copy a list of DB2 objects with a single DB2 COPY job, ensure that APAR PN92578 (PTF UQ12896 - V4; PTF UQ12897 - V5) is applied to your DB2 subsystems.

APAR PN92578 allows the DB2 COPY utility to improve the availability of the data being copied when the CONCURRENT and FILTERDDN keywords are used. With this APAR on, DFSMSdss first logically copies all data. The logical copies are then available for updating while DFSMSdss completes the physical copies of all of the data.

Without APAR PN92578 applied, each object copied requires its own output data set and the last object copied would not be available for updating until the logical and physical copies of all previous objects in the list are completed.

In addition to applying APAR PN92578 to your DB2 subsystems, you should also apply APAR PQ15353 (PTF numbers were unavailable at the time of writing). APAR PQ15353 improves the performance of the DB2 RECOVER utility when the recover function uses image copies made with the CONCURRENT and FILTERDDN keywords specified.

APAR PQ15353 changes the DB2 RECOVER utility to invoke DFSMSdss only once with a list of all the objects to be restored. Without this APAR applied, the DB2 RECOVER utility invokes DFSMSdss each time for every object being recovered.

5.2 Benefits of Virtual Concurrent Copy with DB2

Using virtual concurrent copy for your DB2 data has many benefits in the areas of DB2 data backup, disaster recovery, test data creation, and database cloning.

5.2.1 DB2 Data Backup

Now that the CONCURRENT keyword functionality invokes virtual concurrent copy, you can use the DB2 COPY utility to backup your DB2 data that resides on an RVA. Thus you can back up single or multiple objects, using DB2 back up and recovery processes, and the only time your data is unavailable for updating is the time required to complete the logical copy of the objects. Because the logical copy is complete once the data is snapped, the object is required to be in a read-only state for just a few seconds (a single data set takes about five seconds to snap). Figure 9 shows how DB2 image copy uses virtual concurrent copy.

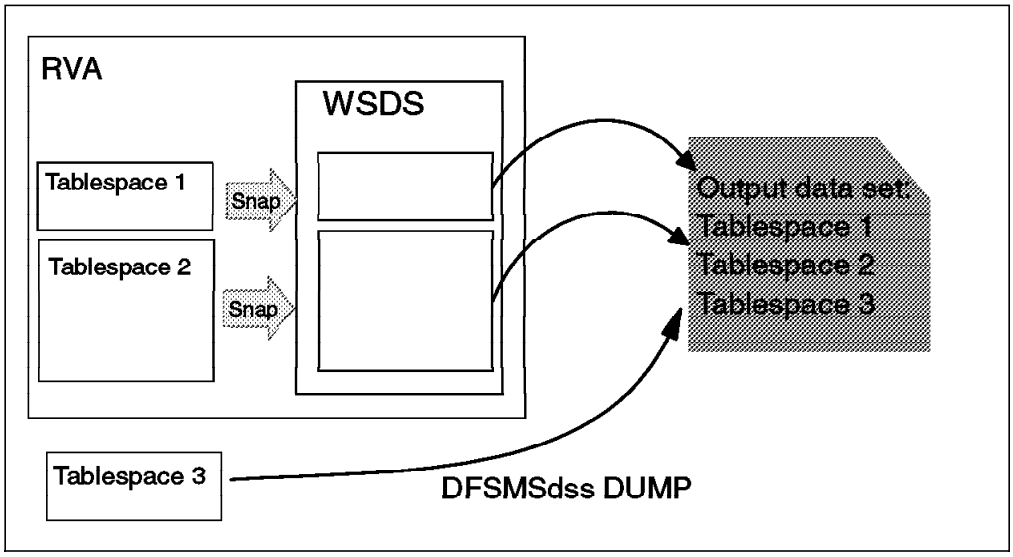


Figure 9. DB2 Image Copy with Virtual Concurrent Copy

A list of objects is specified in the COPY utility control statement. In the example, these are Tablespace 1 and Tablespace 2. These tablespaces are placed into read-only status, and DFSMSdss is invoked to make the copy. DFSMSdss uses virtual concurrent copy and snaps the tablespaces into the working space data set (WSDS). At this point, DB2 can identify that the copy is *logically complete* and places the objects back into read-write status.

DFSMSdss then dumps the copies from the WSDS to an image copy output data set. Once the copy is physically complete, DB2 records the output data set name in SYSCOPY. The output data set is in DFSMSdss DUMP format.

The recover utility does not use SnapShot. Figure 10 on page 17 shows that the DB2 RECOVER command calls DFSMSdss to restore the image copy data set to disk. DB2 can then further process any log records required to complete the recovery operation (if any).

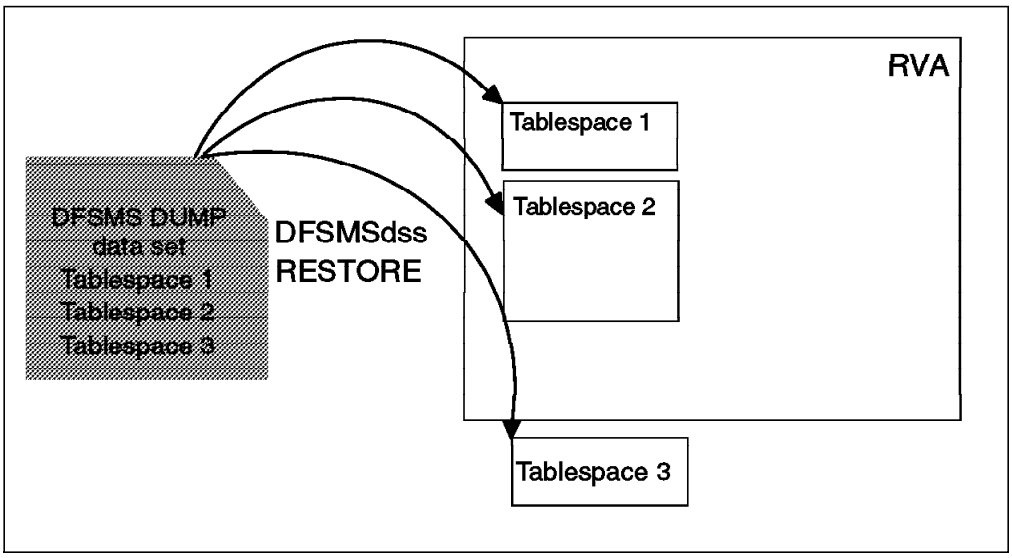


Figure 10. DB2 RECOVER Using CONCURRENT Image Copy Output

SnapShot cannot be used to recover the image copy, so recover will operate at its usual speed. If you have a requirement for a very fast recovery, you should consider using SnapShot to create a point-in-time copy of your DB2 data. See 5.2.2.2, “Using DFSMSdss SnapShot at Volume Level for Disaster Recovery” on page 19, and 5.2.2.3, “Using DFSMSdss SnapShot at the Data Set Level for Disaster Recovery” on page 20. If you are currently making multiple backups of your DB2 data, you can benefit from minimal I/O from virtual concurrent copy while the copies are logically completed.

5.2.1.1 Types of DB2 Image Copies

The DB2 COPY utility enables you to make two levels of copies: SHRLEVEL CHANGE and SHRLEVEL REFERENCE, either of which can be created using CONCURRENT COPY.

SHRLEVEL CHANGE: A SHRLEVEL CHANGE copy is sometimes referred to as a *fuzzy* copy, because some of the data on the copy might be logically inconsistent.

Note:

Never use a SHRLEVEL CHANGE image copy in a recover TOCOPY situation.

When SHRLEVEL CHANGE is used in a recover situation, always recover to a *consistent* point after the copy had been taken. A consistent point to which to recover a DB2 object could be any of the following:

- The point-in-time at which a QUIESCE utility was run against the object. This value can be obtained from SYSIBM.SYSCOPY for records that have an ICTYPE value of *Q* for the object you are going to recover. You need to use the value from the START_RBA column, which is either a relative byte address (RBA) value (in a non-datasharing environment) or a log record sequence number (LRSN) value (in a datasharing environment).
- The point-in-time at which an ARCHIVE LOG MODE(QUIESCE) was performed, but only in a non-data sharing environment. Use the end RBA value of the archive log that was created as a result of the command as the TORBA value in your recover utility job.

- The RBA value when DB2 was stopped, but only in a non-data sharing environment (unless all DB2 members are stopped, at which point you need the highest LRSN value of the member that stopped last). You then use this RBA or LRSN value as either the TORBA or TOLOGPOINT value in your recover utility. You *must* use this RBA or LRSN value as your consistent recovery point.
- The point of currency, that is, all log records associated with the object are applied following the restoration of the image copy.

A recovery to a consistent point in time after the copy was made involves the following processes:

1. The image copy for the object being recovered is restored.
2. All changes to the object that occurred between the time the copy was made and up to the consistent recovery point are applied to the object by using DB2 log data.

SHRLEVEL REFERENCE: A SHRLEVEL REFERENCE copy is sometimes referred to as a *consistent* copy because the data in the copy is logically consistent. You can use a SHRLEVEL REFERENCE copy in a recover TOCOPY situation. You can also use it to recover your data to a consistent point in time after the copy was made by applying log data in the recover process, just as you would when using a SHRLEVEL CHANGE copy.

The output from a virtual concurrent copy is a DFSMSdss DUMP data set. This output cannot be processed directly by the DB2 utilities. If you use your image copy output for processing with another DB2 utility, such as DSN1COPY, make such copies separately from your virtual concurrent copies.

5.2.2 Disaster Recovery

To ensure the consistency of your data for disaster recovery purposes, you can take advantage of virtual concurrent copy to make SHRLEVEL REFERENCE DB2 image copies with minimal outage to your applications. This technique can be used to coordinate your DB2 recovery with another DBMS and to copy all of your data at the same consistent point in time. Also, if your disaster recovery plan involves recovering your DB2 data to the most recent image copy to minimize the amount of DB2 log data to be processed, use virtual concurrent copy to create consistent copies with minimal outage to your applications.

If your DB2 disaster recovery plan uses data set backups or volume dumps to restore your DB2 environment, you can also take advantage of DFSMSdss SnapShot to copy large amounts of data to the same consistent point, with minimal outage to your applications.

5.2.2.1 Using Image Copies for Disaster Recovery

Because you can create up to four output copies of your data with the DB2 COPY utility, you can also use virtual concurrent copy to make the copies and minimize application outages while the copies are being made. The copies can then be sent offsite, and you can use standard DB2 recovery procedures to recover your data.

Note

Even if you use virtual concurrent copy to copy all of your DB2 data, we recommend having all DB2 log data and the most recent copy of your DB2 bootstrap data set (BSDS) available at the disaster recovery site. If any of your DB2 data cannot be recovered with the most recent copy, the previous copy and log data can be used to recover an object to the point in time to which you are recovering the rest of your DB2 subsystem.

5.2.2.2 Using DFSMSdss SnapShot at Volume Level for Disaster Recovery

The process for using SnapShot at the volume level for disaster recovery is only slightly different than the process at the data set level. If you are currently using volume dumps and restores for your disaster recovery planning, you can use virtual concurrent copy to significantly reduce the time required to back up your data, thus minimizing the outage to your applications. You can realize the benefits of virtual concurrent copy without having to change your current disaster recovery processes.

Figure 11 shows the process to follow to use SnapShot for DB2 disaster recovery.

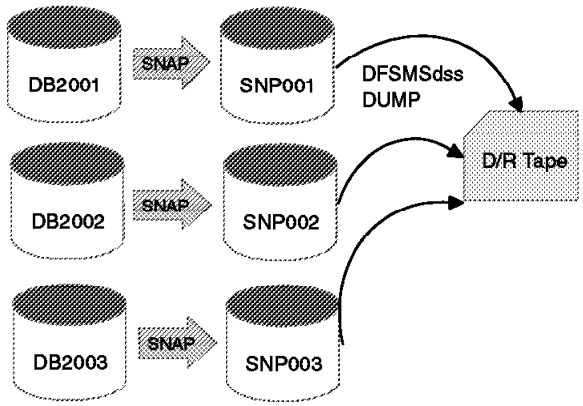


Figure 11. Using SnapShot for DB2 Disaster Recovery

To use volume snaps for disaster recovery, follow this process:

1. Stop your DB2 subsystem. This ensures consistency across both the data and subsystem data sets.
2. Snap all your DB2 data and system data sets, including the catalog and directory, the BSDS, and the system libraries.

   A single volume snap takes about five seconds.

   With volume snap you have the option to snap to a volume with an identical volser and back up the data from another OS/390 server, or you can snap to a different volser. In both cases the data set names are unchanged, but as these are uncataloged you cannot access the data sets.
3. Restart DB2. Applications can update the data.

4. If you need to send copies to a disaster recovery site, back them up to tape. The tapes will need to be restored to disks with the same names (SNP001, SNP002, SNP003).
5. The copy data on the volumes can then be deleted, freeing up any disk space that is used. The amount of disk space depends on the update frequency of your application data.

   If you want to keep a point-in-time copy for use in the event of an operational or procedural failure, you can keep the snapped copies online and snap them back to recover.
6. At your disaster recovery site, restore the backup tapes to volumes that match the volsers that you dumped (in the example in Figure 11 on page 19 these are SNP001, SNP002, and SNP003). To restore the volumes to their correct volsers you can snap back to the original volsers or clip the volser to DB2001, DB2002, and DB2003.
7. If you need to recover at the disaster site, you have a complete, consistent, restartable copy of your production DB2 environment.

The benefit of using volume snaps for disaster recovery is that it provides a consistent point-in-time backup in a very short time. Once you restore the tapes at your recovery site, the recovery process is complete, and no further DB2 recovery actions are required. If you wanted to recover to a higher level of consistency, you could apply log data to your disaster copy; however, this would increase the time for recovery.

5.2.2.3 Using DFSMSdss SnapShot at the Data Set Level for Disaster Recovery

You can now take advantage of wild-carding and filtering within DFSMSdss to back up all DB2 data needed for a disaster recovery, with minimal outage to your applications and using very few DFSMSdss SnapShot control statements.

For a detailed description of using data set snap to create a disaster recovery copy of your data, see *Implementing SnapShot*, SG24-2241.

5.3 SnapShot Positioning

SnapShot provides benefits for customers who require a point-in-time copy of data, where data consistency is of prime importance. Recovering to a consistent point-in-time copy enables you to recover quickly and coordinate recovery of multiple separate applications that may be related. This is especially relevant where your application suite spans both DB2 and non-DB2 applications. If your prime objective is application availability, DB2 provides a comprehensive series of utilities that enable you to run DB2 with a minimum of disruption.

For detailed operational procedures for using virtual concurrent copy with DB2 image copy, see *Implementing DFSMSdss SnapShot and Virtual Concurrent Copy*, SG24-5658.

5.4 Virtual Concurrent Copy Considerations

If you use the DB2 COPY utility with the CONCURRENT keyword the following considerations apply:

- You must supply either a COPYDDN ddname, a RECOVERYDDN ddname, or both.

  Do not specify DASD and tape output data sets in the same COPY utility step.

- If the SYSPRINT DD card points to a data set, you must use a DSSPRINT DD card.
- After a LOAD or REORG LOG(NO), the status of a table space is reset to a copy pending state. With virtual concurrent copy, no write access to the table space is allowed until the copy *physically* completes.
- Check that you have sufficient DASD to support the online virtual concurrent copy images.
- You can specify a list of table spaces to copy.
- You must use the SHRLEVEL REFERENCE option for table spaces with a 32KB page size.
- You *cannot* use a copy made with virtual concurrent copy with the PAGE or ERRORRANGE options. If you specify PAGE or ERRORRANGE, RECOVER bypasses any concurrent copy records when searching the SYSCOPY table for a recoverable point.
- The batch ID that invokes COPY with the CONCURRENT option must provide the necessary authority to invoke the DFSMSdss DUMP command.
- You *cannot* run the following DB2 utilities or stand-alone utilities on copies made by virtual concurrent copy:
  - DSN1COMP
  - DSN1COPY
  - DSN1PRNT
  - MERGECOPY
- When using virtual concurrent copy to make copies of the DB2 catalog and directory for disaster recovery, the following tablespaces *cannot* be included in a list of tablespaces to be copied; each one must be specified as a single object:
  - DSNDB01.SYSLGRNX
  - DSNDB06.SYSCOPY
  - DSNDB01.SYSUTILX
- You *cannot* invoke the CONCURRENT option from the DB2I utilities panel or from the DSNU TSO CLIST.
- The COPY utility will fail if one or more of the table spaces do not reside on a concurrent copy-capable device.
- Incremental image copies are disallowed if the previous copy is a virtual concurrent copy.

Chapter 6. How Can I Use SnapShot to Clone My Database?

SnapShot enables you to copy large amounts of data from one OS/390 system to another with relative ease. In addition, because use of back-end storage is minimal, you may be able to keep multiple versions of test data online to speed regression testing processes.

You can use SnapShot to back up DB2 system volumes or data sets.

You may want to test your DB2 production for Year 2000 readiness, using a test OS/390 image. With its ability to clone data and maintain multiple versions, SnapShot enables you to perform repeatable tests on new applications or program fixes that require regressing data to an earlier point in time, without having to constantly reload the data.

6.1 Making Multiple Copies of Your DB2 Data

If you are currently using the DB2 COPY utility to make more than one backup of your DB2 data, you can use SnapShot to make up to four output copies at a time. Virtual concurrent copy can further help minimize your application outage during the copy process. As you may well be aware when you create multiple backups, the amount of I/O performed during copying increases. Because the I/O for each copy can contend with one another, the COPY utility takes longer to complete.

With virtual concurrent copy, the copy process is logically complete in a very short time and very little I/O is performed during the logical copy phase. Making additional copies has no impact on the time required to complete the logical copy. The I/O for reading the source and writing to multiple targets is performed during the physical copy phase, and during this time applications are allowed to update the source data of the copy operation.

The output from a virtual concurrent copy is a DFSMSdss DUMP data set. This output cannot be processed directly by the DB2 utilities. If you use your image copy output for processing with another DB2 utility, such as DSN1COPY, make such copies separately from your virtual concurrent copies. If your testing process involves restoring your DB2 to a previous point in time with either the DB2 LOAD utility or by restoring a previous copy of your data, DFSMSdss Snapshot or virtual concurrent copy can help reduce the time required to restore your data.

You can use the DB2 COPY utility with virtual concurrent copy to back up your data before beginning your testing, and then use the DB2 RECOVER utility to restore your data. However, a much faster way to perform application testing is to use SnapShot to copy your data before beginning testing.

To use SnapShot to rebuild your test data very quickly, follow this procedure:

1. Choose the source data sets to be backed up. If you will be restoring your data to the point in time at which the copy is made, then also choose which indexes you want to back up. This will enable you to skip the step to RECOVER your indexes, thus saving even more time in the rebuilding of your test data. However, before backing up your data using DFSMSdss for the very first time, you must load the data into the DB2 objects you will be backing up.

2. Establish the consistency point to which you intend to restore your test environment. Ensure that all of your DB2 test data sets, including all indexes, are at this point before you back up your data.
3. Back up the DB2 data sets.
4. Perform your testing.
5. Reset your test environment by “snapping back” the backups taken in step 3.
6. Repeat steps 4 and 5 as many times as necessary to complete your testing.

6.1.1 Application Testing Considerations

When using DFSMSdss to rebuild your DB2 application test data, you must consider the following points:

- If you change the structure of any of your DB2 tables, you must make a new backup of your DB2 data sets. Also, you must perform this backup *after* the structure change has been made and data has been loaded into the DB2 data sets.
- If you decide to change the point in time to which you intend to restore your test DB2 environment to, you must also take new backups of your data. When making these backups, always back up all indexes so that the minimal amount of time is required to reset your test DB2 environment.

6.2 Cloning DB2 Data Using SnapShot

Copying DB2 data from one DB2 subsystem to another can be complex. If you are planning to selectively copy parts of DB2, you must have a deep and detailed understanding of both DB2 and your application. It is recommended that you copy all of the data you are cloning to the same consistent point. (However, in a test environment, achieving complete data integrity may not be important.)

6.2.1 Using SnapShot for Application Test Data

Using SnapShot to extract DB2 test data from one DB2 subsystem for use in another can be a challenging task, though not impossible if you use the proper procedures. Because DB2 tracks every object by the internal identifiers assigned to it within a DB2 subsystem, it is highly likely that you cannot just use DB2 data from one subsystem in another without some additional processes.

The most common method used to access DB2 data created in one subsystem in another is the DB2 utility DSN1COPY. For detailed instructions on how to use DSN1COPY to determine how to translate the internal identifiers from one subsystem to another and then perform the translation, refer to *DB2 for OS/390 Version 5 Utility Guide and Reference*, SC26-8967.

Note

If the copy of the data you plan to copy into another DB2 subsystem was made with the DB2 COPY utility specifying the CONCURRENT keyword, you will have one additional step to perform before you can run the DSN1COPY utility to translate the data.

Because this output is in DFSMSdss DUMP format, you will first need to manually RESTORE the DB2 objects, using DFSMSdss, into VSAM data sets before running the DSN1COPY utility to translate the internal identifiers from one DB2 subsystem to another.

6.2.2 Copying a DB2 Subsystem Using SnapShot

An alternative to translating the internal identifiers is to create a completely cloned DB2 system, which will then require no translation. You can do this by using SnapShot to copy the entire DB2 system including:

- The catalog and directory data sets
- Both boot strap data sets (BSDS)
- All active log data sets
- All application data sets
- Your DB2 load libraries and application load libraries

If you copied your DB2 environment with SnapShot after the original DB2 subsystem had been shut down normally, you do not have to copy any of the DB2 archive log data sets.

DFSMSdss will automatically invoke SnapShot when you perform a DFSMSdss COPY between data sets on the same RVA. Figure 12 shows an example of a DFSMSdss job to copy all data sets with a specified high-level qualifier to another high-level qualifier.

```
//CPYDB2   EXEC PGM=ADRDSSU
//SYSPRINT DD SYSOUT=*
//SYSIN    DD *
 COPY DATASET(INCLUDE(DB2P.**)) -
   RENAMEU(DB2X) CATALOG -
      OUTDYNAM(WORKO1) CONCURRENT
```

Figure 12. COPY All DB2 Data Sets Using DFSMSdss

Using this copy process, you can copy all your DB2 data to a different high-level qualifier. Both copies of the data are on the same OS/390 subsystem on which your DB2 data currently resides. SnapShot is used to copy the data without any data movement, so the copy operation completes very quickly.

Move the new copies of the data to the OS/390 subsystem on which you intend to run the cloned DB2, and run the DFSMSdss COPY step shown in Figure 13 on page 26 step to copy all data sets back to their original high-level qualifier.

```
//CPYDB2   EXEC PGM=ADRDSSU
//SYSPRINT DD SYSOUT=*
//SYSIN    DD *
 COPY DATASET(INCLUDE(DB2X.**)) -
   RENAMEU(DB2P) CATALOG -
      OUTDYNAM(WORK02) CONCURRENT
```

Figure 13. COPY All DB2 Data Sets Using DFSMSdss Back to Orginal HLQ

In the example shown in Figure 13, you may now delete the data sets created with the high-level qualifier of DB2X. You now have an exact duplicate of the original DB2P subsystem that you can start on a different OS/390 subsystem. This process allows you to run two entirely separate DB2 subsystems on two different OS/390 subsystems using the same data. The only increase you will see in NCL usage is from the number of changes that occur on the data.

This cloned DB2 has the same name as the original, so it must run in a different OS/390 environment. You have to ensure that there is no confusion between the data belonging to the original and cloned systems. Maintaining this separation is operationally complex.

Note

If you want to use this process to test your applications on a different OS/390 subsystem, it is not necessary to copy all of your DB2 data. You do not have to copy any of the application data you do not intend to use in your new DB2 subsystem.

If you need to run the cloned DB2 in the same OS/390 environment as the original, you need to *rename* the cloned system. Internally it will still have the same identifiers, but externally the data sets all have different names. This means you also have to rename the VCAT, and make some changes so that all the new data set names are known to DB2.

A procedure to rename a DB2 system is documented in the redbook *DB2 for MVS/ESA Version 4 Data Sharing Implementation*, SG24-4791.

Chapter 7. How Can I Use SnapShot to Extract Data from DB2?

SnapShot is a very fast data copy utility. If you are extracting data from a production operational system, you may be using standard DB2 utilities to move data into a data warehouse environment. SnapShot can minimize the disruption to production transactions when extracting data from DB2.

If you use SnapShot to create a clone of your DB2 subsystem as described in Chapter 6, “How Can I Use SnapShot to Clone My Database?” on page 23, you have two DB2 subsystems that are internally identical although they have different names. This means that the same table, in each DB2, has the same internal identifiers.

Each table in a DB2 subsystem has different identifiers. If you want to copy data from a table and use it in another table in the same DB2, for example as a read-only copy of the database, you have to translate the internal identifiers.

The following process shows the steps to make a copy of a table, and make it available in the same DB2 subsystem. The same process would also work when moving or copying data between DB2 subsystems that are not clones.

This process uses the DB2 utility DSN1COPY to translate the unique internal identifiers that DB2 uses. Each table has a database ID (DBID), a page segment ID (PSID), and an object ID (OBID). Indexes have a DBID and an index ID (ISDBID). The DSN1COPY utility translates these internal identifiers to allow data to be moved between DB2 objects.

These are the steps:

1. Define a new table definition using the DDL definition of the source table as a model. Figure 14 on page 28 shows the JCL used to create the new table definition.

```
//*******************************************************
//DDL      EXEC PGM=IKJEFT01,DYNAMNBR=20
//SYSTSPRT DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSTSIN  DD *
  DSN SYSTEM(DBGT)
  RUN PROGRAM(DSNTEP2) PLAN(DSNTEP51)
  END
//SYSIN    DD *

SET CURRENT SQLID='ALISON';
COMMIT;


  DROP TABLESPACE DB2COMP.ALTSCRY;
  COMMIT;
 CREATE TABLESPACE ALTSCRY  IN DB2COMP
     NUMPARTS 10
     (
     PART   1 USING STOGROUP CPRVA128 PRIQTY 408000 SECQTY 7200 ,
     PART   2 USING STOGROUP CPRVA126 PRIQTY 408000 SECQTY 7200 ,
     PART   3 USING STOGROUP CPRVA125 PRIQTY 408000 SECQTY 7200 ,
     PART   4 USING STOGROUP CPRVA124 PRIQTY 408000 SECQTY 7200 ,
     PART   5 USING STOGROUP CPRVA123 PRIQTY 408000 SECQTY 7200 ,
     PART   6 USING STOGROUP CPRVA122 PRIQTY 408000 SECQTY 7200 ,
     PART   7 USING STOGROUP CPRVA121 PRIQTY 408000 SECQTY 7200 ,
     PART   8 USING STOGROUP CPRVA120 PRIQTY 408000 SECQTY 7200 ,
     PART   9 USING STOGROUP CPRVA119 PRIQTY 408000 SECQTY 7200 ,
     PART  10 USING STOGROUP CPRVA118 PRIQTY 408000 SECQTY 7200
     )

   FREEPAGE 20
   PCTFREE 15
   BUFFERPOOL BP1
   COMPRESS YES
   LOCKSIZE ANY
   CLOSE NO;


     CREATE TABLE LINEITEM (
    L_ORDERKEY CHAR(10) NOT NULL,
    L_PARTKEY INT NOT NULL,
    L_SUPPKEY INT NOT NULL,
    L_QUANTITY REAL NOT NULL,
    L_EXTENDEDPRICE REAL NOT NULL,
    L_DISCOUNT REAL NOT NULL,
    L_TAX REAL NOT NULL,
    L_RETURNFLAG CHAR(1) NOT NULL,
    L_LINESTATUS CHAR(1) NOT NULL,
    L_SHIPDATE DATE NOT NULL,
    L_COMMITDATE DATE NOT NULL,
    L_RECEIPTDATE DATE NOT NULL,
    L_SHIPINSTRUCT CHAR(25) NOT NULL,
    L_SHIPMODE CHAR(10) NOT NULL,
    L_COMMENT VARCHAR(44) NOT NULL)
      IN  DB2COMP.ALTSCRY;
```

Figure 14 (Part 1 of 2). Sample Table Definition

```
 CREATE INDEX PXL@ALI
   ON LINEITEM
 (L_ORDERKEY,L_LINENUMBER)
   FREEPAGE 20
   PCTFREE 15
   BUFFERPOOL BP2
   CLOSE NO
   CLUSTER
  CLUSTER
  (
   PART   1 VALUES('   6000000',9999) USING STOGROUP CPRVA108
              PRIQTY 170500 SECQTY 25575,
   PART   2 VALUES('  12000000',9999) USING STOGROUP CPRVA109
              PRIQTY 170500 SECQTY 25575,
   PART   3 VALUES('  18000000',9999) USING STOGROUP CPRVA110
              PRIQTY 170500 SECQTY 25575,
   PART   4 VALUES('  24080000',9999) USING STOGROUP CPRVA111
              PRIQTY 170500 SECQTY 25575,
   PART   5 VALUES('  30000000',9999) USING STOGROUP CPRVA112
              PRIQTY 170500 SECQTY 25575,
   PART   6 VALUES('  36000000',9999) USING STOGROUP CPRVA113
              PRIQTY 170500 SECQTY 25575,
   PART   7 VALUES('  42000000',9999) USING STOGROUP CPRVA114
              PRIQTY 170500 SECQTY 25575,
   PART   8 VALUES('  48000000',9999) USING STOGROUP CPRVA115
              PRIQTY 170500 SECQTY 25575,
   PART   9 VALUES('  54080000',9999) USING STOGROUP CPRVA116
              PRIQTY 170500 SECQTY 25575,
   PART  10 VALUES('  60000000',9999) USING STOGROUP CPRVA117
              PRIQTY 170500 SECQTY 25575
 );
 COMMIT;
```

Figure 14 (Part 2 of 2). Sample Table Definition

2. To perform a DSN1COPY translation, you need the internal identifiers of the objects that you are copying to and from. Figure 15 and Figure 16 on page 30 show sample queries to extract the internal identifiers from the data and index portions of a table.

```
   SELECT DBID, PSID FROM SYSIBM.SYSTABLESPACE
       WHERE NAME = 'ALTSCRY'
       AND   DBNAME = 'DB2COMP';
   SELECT NAME, OBID  FROM  SYSIBM.SYSTABLES
       WHERE TSNAME='ALTSCRY'
       AND CREATOR='ALISON';
   SELECT DBID, ISOBID  FROM  SYSIBM.SYSINDEXES
    WHERE NAME='PXL@ALI'
    AND CREATOR='ALISON';
```

Figure 15. Sample SQL to Determine DB2 Identifiers of Data

```
   SELECT DBID, PSID FROM SYSIBM.SYSTABLESPACE
       WHERE NAME = 'TSCRY'
       AND   DBNAME = 'DB2COMP';
   SELECT NAME, OBID  FROM  SYSIBM.SYSTABLES
       WHERE TSNAME='TSCRY'
       AND CREATOR='CRY';
   SELECT DBID, ISOBID  FROM  SYSIBM.SYSINDEXES
    WHERE NAME='PXL@CRY'
    AND CREATOR='CRY';
```

Figure 16. Sample SQL Query to Extract Identifiers of Index

Figure 18 on page 31 and Figure 17 show the output from the queries.

```
---------+---------+---------+---------+---------+---------+---------+---
  SELECT DBID, PSID FROM SYSIBM.SYSTABLESPACE
      WHERE NAME = 'ALTSCRY'
      AND   DBNAME = 'DB2COMP';
---------+---------+---------+---------+---------+---------+---------+---
  DBID    PSID
---------+---------+---------+---------+---------+---------+---------+---
   263       2
DSNE610I NUMBER OF ROWS DISPLAYED IS 1
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
---------+---------+---------+---------+---------+---------+---------+---
  SELECT NAME, OBID  FROM  SYSIBM.SYSTABLES
      WHERE TSNAME='ALTSCRY'
      AND CREATOR='ALISON';
---------+---------+---------+---------+---------+---------+---------+---
NAME                 OBID
---------+---------+---------+---------+---------+---------+---------+---
LINEITEM                3
DSNE610I NUMBER OF ROWS DISPLAYED IS 1
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
---------+---------+---------+---------+---------+---------+---------+---
  SELECT DBID, ISOBID  FROM  SYSIBM.SYSINDEXES
   WHERE NAME='PXL@ALI'
   AND CREATOR='ALISON';
---------+---------+---------+---------+---------+---------+---------+---
  DBID  ISOBID
```

Figure 17. Output from Data Query

```
---------+---------+---------+---------+---------+---------+---------+---
  SELECT DBID, PSID FROM SYSIBM.SYSTABLESPACE
      WHERE NAME = 'TSCRY'
      AND   DBNAME = 'DB2COMP';
---------+---------+---------+---------+---------+---------+---------+---
  DBID    PSID
---------+---------+---------+---------+---------+---------+---------+---
   263       8
DSNE610I NUMBER OF ROWS DISPLAYED IS 1
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
---------+---------+---------+---------+---------+---------+---------+---
  SELECT NAME, OBID  FROM  SYSIBM.SYSTABLES
      WHERE TSNAME='TSCRY'
      AND CREATOR='CRY';
---------+---------+---------+---------+---------+---------+---------+---
NAME                 OBID
---------+---------+---------+---------+---------+---------+---------+---
LINEITEM                9
DSNE610I NUMBER OF ROWS DISPLAYED IS 1
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
---------+---------+---------+---------+---------+---------+---------+---
  SELECT DBID, ISOBID  FROM  SYSIBM.SYSINDEXES
   WHERE NAME='PXL@CRY'
   AND CREATOR='CRY';
---------+---------+---------+---------+---------+---------+---------+---
  DBID  ISOBID
```

Figure 18. Output from Index Query

You must perform this query for the source table (the one that you are copying) and the target table (the one that you defined).

3. Use the DB2 STOP DATABASE command to externalize buffers and close the data set. Use the AT COMMIT parameter to ensure that outstanding units of recovery have completed. This tablespace has 10 partitions.

   ```
   STOP DB (database_name) SPACENAM(table-space_name) PART(1:10) AT(COMMIT)
   ```

4. Use SnapShot to copy all the data and index partitions of the table. Figure 20 on page 33 shows a sample of SnapShot JCL to copy our 10-partition table.

   In our example, the DB2 table does not consist of SMS-managed data sets and is not placed on SMS-managed volumes. Therefore, we used the VOLUME parameter to place the data sets on specific volumes we had designated.

   If the files were SMS-managed and were to be placed on a group of volumes in an SMS storage group, then you could either direct the data set allocations via the ACS routines or by specifying a STORCLAS value in the SNAP DATASET command. SMS will attempt to allocate each data set on a volume in the assigned Storage Group. If a volume is selected that is not within the same RVA as the source data set, then SnapShots Volume Preferencing will reject the allocation until a suitable volume is chosen.

   In our example, the DB2 table partitions are the only files on the volumes. If other files were on the volumes and the volumes were enqueued because of these files, the TOLERATEENQFAILURE parameter could have been used to allow Enqueue Failure and continue with the SNAP operation.

We will run a Space Utilization Report to show the Space usage of the RVA prior to the snap of the DB2 tables. The POST-SNAP report in Figure 22 on page 35 shows that there is now Shared space on each volume that was Unique space before the snap, as shown in Figure 19 on page 32.

```
                                                ITSO DB2 COMPRESSION TESTING
                                            DB2 SNAPSHOT TEST - PRE-SNAP SPACE REPORT
XSA/REPORTER                                SPACE UTILIZATION SUMMARY REPORT                22JUN1998  12:56:44
                                                                                            SIBSPUT  V2 R1 L1
SUBSYSTEM 20395                      (NUMBER OF FUNCTIONAL DEVICES: 129)
                                                --FUNCTIONAL CAPACITY (MB)--  % FUNCTIONAL CAPACITY
FDID  DEV   VOLSER    T/P  DEVICE  FUNCT                          NOT                    NOT   --PHYSICAL CAP USED (MB)--   COMP
      ADDR                    TYPE  CAP (MB)    ALLOC    STORED    STORED    ALLOC STORED STORED   SHARED   UNIQUE   TOTAL     RATIO

----  ----  ------    ---  ------  --------  --------  --------  --------   ----- ------ ------  -------- -------- --------   -----

006B  2B6B  RV2B6B     P   33903     2838.0     205.4     204.8    2633.2    7.2    7.2   92.8       0.0     58.5     58.5      3.5
006C  2B6C  RV2B6C     P   33903     2838.0     205.4     204.8    2633.2    7.2    7.2   92.8       0.0     58.6     58.6      3.5
006D  2B6D  RV2B6D     P   33903     2838.0     205.4     204.8    2633.2    7.2    7.2   92.8       0.0     58.6     58.6      3.5
006E  2B6E  RV2B6E     P   33903     2838.0     205.4     204.8    2633.2    7.2    7.2   92.8       0.0     58.5     58.5      3.5
006F  2B6F  RV2B6F     P   33903     2838.0     205.4     202.1    2635.9    7.2    7.1   92.9       0.0     58.8     58.8      3.5
0070  2B70  RV2B70     P   33903     2838.0     205.4     204.8    2633.2    7.2    7.2   92.8       0.0     58.6     58.6      3.5
0071  2B71  RV2B71     P   33903     2838.0     205.4     204.7    2633.3    7.2    7.2   92.8       0.0     58.5     58.5      3.5
0072  2B72  RV2B72     P   33903     2838.0     205.4     204.8    2633.2    7.2    7.2   92.8       0.0     58.6     58.6      3.5
0073  2B73  RV2B73     P   33903     2838.0     205.4     204.7    2633.3    7.2    7.2   92.8       0.0     58.5     58.5      3.5
0074  2B74  RV2B74     P   33903     2838.0     205.4     202.0    2636.1    7.2    7.2   92.8       0.0     58.7     58.7      3.5
0075  2B75  RV2B75     P   33903     2838.0     967.8     468.6    2369.5   34.1   16.5   83.5       0.0    276.3    276.3      1.7
0076  2B76  RV2B76     P   33903     2838.0     967.8     483.5    2354.5   34.1   17.0   83.0       0.0    281.8    281.8      1.7
0077  2B77  RV2B77     P   33903     2838.0     967.8     475.0    2363.0   34.1   16.7   83.3       0.0    280.3    280.3      1.7
0078  2B78  RV2B78     P   33903     2838.0     967.8     482.4    2355.6   34.1   17.0   83.0       0.0    283.5    283.5      1.7
0079  2B79  RV2B79     P   33903     2838.0     967.8     476.0    2362.0   34.1   16.8   83.2       0.0    281.0    281.0      1.7
007A  2B7A  RV2B7A     P   33903     2838.0     967.8     467.9    2370.1   34.1   16.8   83.2       0.0    281.0    281.0      1.7
007B  2B7B  RV2B7B     P   33903     2838.0     967.8     486.6    2351.4   34.1   16.5   83.5       0.0    275.8    275.8      1.7
007C  2B7C  RV2B7C     P   33903     2838.0     967.8     475.6    2362.4   34.1   17.1   82.9       0.0    284.4    284.4      1.7
007D  2B7D  RV2B7D     P   33903     2838.0     967.8     477.8    2360.2   34.1   16.8   83.2       0.0    280.7    280.7      1.7
007F  2B7F  RV2B7F     P   33903     2838.0     966.3     476.0    2362.0   34.1   16.8   83.2       0.4    281.4    281.4      1.7

SELECTED DEVICES SUMMARY                        FUNCTIONAL CAPACITY (MB)  % FUNCT CAPACITY       -------- DISK ARRAY --------
                     SELECTED  TOTAL FUNCTIONAL                  NOT               NOT        -- PHYSICAL CAP USED (MB) --  COMP
                      DEVICES  CAPACITY (MB)          STORED    STORED      STORED STORED       SHARED     UNIQUE    TOTAL   RATIO
                     --------  -------------        --------- ---------    ------ ------       --------  --------  --------  -----
PRODUCTION PARTITION:   129     366104.1             214109.2  151994.9     58.5   41.5          562.8   83018.1   83580.9    2.5
TOTALS:                 129     366104.1             214109.2  151994.9     58.5   41.5          562.8   83018.1   83580.9    2.5


SUBSYSTEM 20395              SPACE UTILIZATION SUMMARY                                      22JUN1998   12:56:44
                                                                                           SIBSPUT  V2 R1 L1
                        NUMBER OF          DISK ARRAY       NET CAPACITY LOAD(%)     COLL FREE SPACE (%)      UNCOLL FREE SPACE(%)
                    FUNCTIONAL DEVICES     CAPACITY (MB)     TEST   PROD  OVERALL     TEST   PROD  OVERALL      TEST   PROD  OVERALL
                    ------------------     -------------    -----  -----  -------     -----  -----  -------     -----  -----  -------
                               129            117880.2         0.0   68.3    68.3       0.0   27.5    27.5        0.0    4.2     4.2
```

Figure 19. Space Useage Report Pre-Snap

```
//SNAPCRY  JOB  ,CLASS=A,MSGLEVEL=(1,1),MSGCLASS=H,NOTIFY=&SYSUID
/*JOBPARM SYSAFF=QP01
//DDSR     EXEC  PGM=SIBBATCH
//SYSPRINT DD SYSOUT=*
//SYSTERM  DD SYSOUT=*
//CTRANS   DD DSN=IXFP.V2R1M0.SACLINK,DISP=SHR
//SYSIN    DD *
 SNAP DATASET( -
   SOURCE('DB2PROOF.DSNDBC.DB2COMP.TSCRY.I0001.A001') -
   TARGET('DB2PROOF.DSNDBC.DB2COMP.TSNAP.I0001.A001') -
   VOLUME(RV2B7F) REPL(YES) TOLENQF(NO))
 SNAP DATASET( -
   SOURCE('DB2PROOF.DSNDBC.DB2COMP.TSCRY.I0001.A002') -
   TARGET('DB2PROOF.DSNDBC.DB2COMP.TSNAP.I0001.A002') -
   VOLUME(RV2B7D) REPL(YES) TOLENQF(NO))
 SNAP DATASET( -
   SOURCE('DB2PROOF.DSNDBC.DB2COMP.TSCRY.I0001.A003') -
   TARGET('DB2PROOF.DSNDBC.DB2COMP.TSNAP.I0001.A003') -
   VOLUME(RV2B7C) REPL(YES) TOLENQF(NO))
 SNAP DATASET( -
   SOURCE('DB2PROOF.DSNDBC.DB2COMP.TSCRY.I0001.A004') -
   TARGET('DB2PROOF.DSNDBC.DB2COMP.TSNAP.I0001.A004') -
   VOLUME(RV2B7B) REPL(YES) TOLENQF(NO))
 SNAP DATASET( -
   SOURCE('DB2PROOF.DSNDBC.DB2COMP.TSCRY.I0001.A005') -
   TARGET('DB2PROOF.DSNDBC.DB2COMP.TSNAP.I0001.A005') -
   VOLUME(RV2B7A) REPL(YES) TOLENQF(NO))
 SNAP DATASET( -
   SOURCE('DB2PROOF.DSNDBC.DB2COMP.TSCRY.I0001.A006') -
   TARGET('DB2PROOF.DSNDBC.DB2COMP.TSNAP.I0001.A006') -
   VOLUME(RV2B79) REPL(YES) TOLENQF(NO))
 SNAP DATASET( -
   SOURCE('DB2PROOF.DSNDBC.DB2COMP.TSCRY.I0001.A007') -
   TARGET('DB2PROOF.DSNDBC.DB2COMP.TSNAP.I0001.A007') -
   VOLUME(RV2B78) REPL(YES) TOLENQF(NO))
 SNAP DATASET( -
   SOURCE('DB2PROOF.DSNDBC.DB2COMP.TSCRY.I0001.A008') -
   TARGET('DB2PROOF.DSNDBC.DB2COMP.TSNAP.I0001.A008') -
   VOLUME(RV2B77) REPL(YES) TOLENQF(NO))
 SNAP DATASET( -
   SOURCE('DB2PROOF.DSNDBC.DB2COMP.TSCRY.I0001.A009') -
   TARGET('DB2PROOF.DSNDBC.DB2COMP.TSNAP.I0001.A009') -
   VOLUME(RV2B76) REPL(YES) TOLENQF(NO))
 SNAP DATASET( -
   SOURCE('DB2PROOF.DSNDBC.DB2COMP.TSCRY.I0001.A010') -
   TARGET('DB2PROOF.DSNDBC.DB2COMP.TSNAP.I0001.A010') -
   VOLUME(RV2B75) REPL(YES) TOLENQF(NO))
//
```

Figure 20. Sample SnapShot JCL for Data

Figure 21 on page 34 shows the sample JCL to copy the ten index portions associated with the table.

```
//SNAPPXL  JOB  ,CLASS=A,MSGLEVEL=(1,1),MSGCLASS=H,NOTIFY=&SY
/*JOBPARM SYSAFF=QP01
//DDSR     EXEC  PGM=SIBBATCH
//SYSPRINT DD SYSOUT=*
//SYSTERM  DD SYSOUT=*
//CTRANS   DD DSN=IXFP.V2R1M0.SACLINK,DISP=SHR
//SYSIN    DD *
 SNAP DATASET( -
   SOURCE('DB2PROOF.DSNDBC.DB2COMP.PXL@CRY.I0001.A001') -
   TARGET('DB2PROOF.DSNDBC.DB2COMP.PXLSNAP.I0001.A001') -
   VOLUME(RV2B6B) REPL(YES) TOLENQF(NO))
 SNAP DATASET( -
   SOURCE('DB2PROOF.DSNDBC.DB2COMP.PXL@CRY.I0001.A002') -
   TARGET('DB2PROOF.DSNDBC.DB2COMP.PXLSNAP.I0001.A002') -
   VOLUME(RV2B6C) REPL(YES) TOLENQF(NO))
 SNAP DATASET( -
   SOURCE('DB2PROOF.DSNDBC.DB2COMP.PXL@CRY.I0001.A003') -
   TARGET('DB2PROOF.DSNDBC.DB2COMP.PXLSNAP.I0001.A003') -
   VOLUME(RV2B6D) REPL(YES) TOLENQF(NO))
 SNAP DATASET( -
   SOURCE('DB2PROOF.DSNDBC.DB2COMP.PXL@CRY.I0001.A004') -
   TARGET('DB2PROOF.DSNDBC.DB2COMP.PXLSNAP.I0001.A004') -
   VOLUME(RV2B6E) REPL(YES) TOLENQF(NO))
 SNAP DATASET( -
   SOURCE('DB2PROOF.DSNDBC.DB2COMP.PXL@CRY.I0001.A005') -
   TARGET('DB2PROOF.DSNDBC.DB2COMP.PXLSNAP.I0001.A005') -
   VOLUME(RV2B6F) REPL(YES) TOLENQF(NO))
 SNAP DATASET( -
   SOURCE('DB2PROOF.DSNDBC.DB2COMP.PXL@CRY.I0001.A006') -
   TARGET('DB2PROOF.DSNDBC.DB2COMP.PXLSNAP.I0001.A006') -
   VOLUME(RV2B70) REPL(YES) TOLENQF(NO))
 SNAP DATASET( -
   SOURCE('DB2PROOF.DSNDBC.DB2COMP.PXL@CRY.I0001.A007') -
   TARGET('DB2PROOF.DSNDBC.DB2COMP.PXLSNAP.I0001.A007') -
   VOLUME(RV2B71) REPL(YES) TOLENQF(NO))
 SNAP DATASET( -
   SOURCE('DB2PROOF.DSNDBC.DB2COMP.PXL@CRY.I0001.A008') -
   TARGET('DB2PROOF.DSNDBC.DB2COMP.PXLSNAP.I0001.A008') -
   VOLUME(RV2B72) REPL(YES) TOLENQF(NO))
 SNAP DATASET( -
   SOURCE('DB2PROOF.DSNDBC.DB2COMP.PXL@CRY.I0001.A009') -
   TARGET('DB2PROOF.DSNDBC.DB2COMP.PXLSNAP.I0001.A009') -
   VOLUME(RV2B73) REPL(YES) TOLENQF(NO))
 SNAP DATASET( -
   SOURCE('DB2PROOF.DSNDBC.DB2COMP.PXL@CRY.I0001.A010') -
   TARGET('DB2PROOF.DSNDBC.DB2COMP.PXLSNAP.I0001.A010') -
   VOLUME(RV2B74) REPL(YES) TOLENQF(NO))
```

Figure 21. Sample JCL to Snap Index

5. Start the database again using the START DATABASE command:

   `START DATABASE (data base name) SPACENAM (tablespace name)`

   The source table is now available for update by your applications.

   SnapShot does not create a physical copy of the data, so we can see from the RVA Space Management Reports in Figure 22 on page 35 that although OS/390 has a data set defined, no additional disk space is used by this copy.

```
                                        ITSO DB2 COMPRESSION TESTING
                                    DB2 SNAPSHOT TEST - POST-SNAP SPACE REPORT
XSA/REPORTER                      SPACE UTILIZATION SUMMARY REPORT                  22JUN1998    13:36:54
                                                                                   SIBSPUT  V2 R1 L1
SUBSYSTEM 20395                (NUMBER OF FUNCTIONAL DEVICES: 129)
                                        --FUNCTIONAL CAPACITY (MB)--  % FUNCTIONAL CAPACITY
FDID  DEV   VOLSER    T/P  DEVICE  FUNCT                      NOT                    NOT   --PHYSICAL CAP USED (MB)--   COMP
      ADDR                 TYPE   CAP (MB)    ALLOC    STORED    STORED    ALLOC STORED STORED   SHARED   UNIQUE   TOTAL     RATIO
----  ----  ------    ---  ------  --------  --------  --------  --------   ----- ------ ------  -------- -------- --------   -----
006B  2B6B  RV2B6B     P   33903     2838.0     608.3     407.4    2430.7   21.4   14.4   85.6     116.5      0.3    116.9     3.5
006C  2B6C  RV2B6C     P   33903     2838.0     608.3     407.4    2430.7   21.4   14.4   85.6     116.5      0.3    116.9     3.5
006D  2B6D  RV2B6D     P   33903     2838.0     608.3     407.4    2430.7   21.4   14.4   85.6     116.5      0.3    116.9     3.5
006E  2B6E  RV2B6E     P   33903     2838.0     608.3     407.4    2430.7   21.4   14.4   85.6     116.5      0.3    116.9     3.5
006F  2B6F  RV2B6F     P   33903     2838.0     608.3     402.0    2436.0   21.4   14.2   85.8     116.5      0.3    116.4     3.5
0070  2B70  RV2B70     P   33903     2838.0     608.3     407.4    2430.7   21.4   14.4   85.6     116.5      0.3    116.9     3.5
0071  2B71  RV2B71     P   33903     2838.0     608.3     407.1    2430.9   21.4   14.3   85.7     116.5      0.3    116.8     3.5
0072  2B72  RV2B72     P   33903     2838.0     608.3     407.4    2430.7   21.4   14.4   85.6     116.5      0.3    116.9     3.5
0073  2B73  RV2B73     P   33903     2838.0     608.3     407.2    2430.8   21.4   14.3   85.7     116.5      0.3    116.8     3.5
0074  2B74  RV2B74     P   33903     2838.0     608.3     401.7    2436.3   21.4   14.2   85.8     114.9      0.3    115.2     3.5
0075  2B75  RV2B75     P   33903     2838.0    1449.7     931.4    1906.6   51.1   32.8   67.2     551.9      0.3    552.2     1.7
0076  2B76  RV2B76     P   33903     2838.0    1449.7     961.3    1876.7   51.1   33.9   66.1     551.9      0.3    563.3     1.7
0077  2B77  RV2B77     P   33903     2838.0    1449.7     944.3    1893.7   51.1   33.3   66.7     551.9      0.3    560.3     1.7
0078  2B78  RV2B78     P   33903     2838.0    1449.7     959.2    1878.9   51.1   33.8   66.2     563.0      0.3    566.6     1.7
0079  2B79  RV2B79     P   33903     2838.0    1449.7     946.3    1891.7   51.1   33.3   66.7     560.0      0.3    561.6     1.7
007A  2B7A  RV2B7A     P   33903     2838.0    1449.7     930.0    1908.0   51.1   32.8   67.2     551.0      0.3    551.4     1.7
007B  2B7B  RV2B7B     P   33903     2838.0    1449.7     967.4    1870.6   51.1   34.1   65.9     568.1      0.3    568.4     1.7
007C  2B7C  RV2B7C     P   33903     2838.0    1449.7     945.6    1892.5   51.1   33.3   66.7     560.7      0.3    561.1     1.7
007D  2B7D  RV2B7D     P   33903     2838.0    1449.7     950.0    1888.0   51.1   33.5   66.5     562.3      0.3    562.6     1.7
007F  2B7F  RV2B7F     P   33903     2838.0    1448.2     947.9    1890.1   51.0   33.4   66.6     562.7      0.3    563.0     1.7

  SELECTED DEVICES SUMMARY                    FUNCTIONAL CAPACITY (MB)  % FUNCT CAPACITY       -------- DISK ARRAY --------
                     SELECTED  TOTAL FUNCTIONAL                  NOT               NOT        -- PHYSICAL CAP USED (MB) --  COMP
                      DEVICES  CAPACITY (MB)          STORED    STORED     STORED STORED       SHARED     UNIQUE    TOTAL   RATIO
                     --------  -------------       --------- ---------    ------ ------       --------  --------  --------  -----
PRODUCTION PARTITION:   129     366104.1            207096.3  159007.8       56.6   43.4       6769.7   76811.2   83580.9    2.5
TOTALS:                 129     366104.1            207096.3  159007.8       56.6   43.4       6769.7   76811.2   83580.9    2.5

SUBSYSTEM 20395             SPACE UTILIZATION SUMMARY                                  22JUN1998    13:36:54
                                                                                       SIBSPUT  V2 R1 L1
                       NUMBER OF          DISK ARRAY       NET CAPACITY LOAD(%)     COLL FREE SPACE (%)      UNCOLL FREE SPACE(%)
                    FUNCTIONAL DEVICES    CAPACITY (MB)    TEST   PROD  OVERALL     TEST   PROD  OVERALL     TEST   PROD  OVERALL
                    ------------------    -------------    -----  -----  -------     -----  -----  -------    -----  -----  -------
                             129             117880.2         0.0   68.3    68.3       0.0   27.5    27.5       0.0    4.2     4.2
```

Figure 22. Space Usage Report Post-Snap

6. Use DSN1COPY to translate the internal identifiers. Figure 23 and Figure 24 on page 36 show samples of the DSN1COPY JCL.

```
          your edit profile using the command RECOVERY ON.
//ALICOP9 JOB MSGLEVEL=(1,1),MSGCLASS=H,REGION=6M,
//      CLASS=A
//JOBLIB   DD DSN=DSN420.SDSNLOAD,DISP=SHR
//         DD DSN=DB2TPLEX.RUNLIB.LOAD,DISP=SHR
//RUNPRNT  EXEC PGM=DSN1COPY
//SYSPRINT DD SYSOUT=*
//SYSXLAT  DD *
263,263
8,2
9,3
/*
//SYSUT1   DD DSN=DB2PROOF.DSNDBC.DB2COMP.TSNAP.I0001.A010,
//    DISP=OLD
//SYSUT2   DD DSN=DB2PROOF.DSNDBC.DB2COMP.ALTSCRY.I0001.A010,
//   DISP=OLD
```

Figure 23. Sample DSN1COPY JCL for Data

```
//ALIXCOPA JOB MSGLEVEL=(1,1),MSGCLASS=H,REGION=6M,
//      CLASS=A
//JOBLIB   DD DSN=DSN420.SDSNLOAD,DISP=SHR
//         DD DSN=DB2TPLEX.RUNLIB.LOAD,DISP=SHR
//RUNPRNT  EXEC PGM=DSN1COPY
//SYSPRINT DD SYSOUT=*
//SYSXLAT  DD *
263,263
11,12
/*
//SYSUT1   DD DSN=DB2PROOF.DSNDBC.DB2COMP.PXLSNAP.I0001.A010,
//    DISP=OLD
//SYSUT2   DD DSN=DB2PROOF.DSNDBC.DB2COMP.PXL@ALI.I0001.A010,
//   DISP=OLD
```

Figure 24. Sample DSN1COPY JCL for Index

Make sure that you stop the target database while you perform the DSN1COPY translation job. Use the STOP DATABASE command.

DSN1COPY rewrites the data to a new data set. Because this is a real data movement, additional disk space is used for the output data set. The space management reports in Figure 25 on page 37 show the disk space used by the new copy of the data.

```
                                        IBM ITSO POKEEPSIE
                                  DB2 SNAPSHOT TEST - POST-DSN1COPY SPACE REPORT
XSA/REPORTER                        SPACE UTILIZATION SUMMARY REPORT                22JUN1998    15:17:21
                                                                                    SIBSPUT  V2 R1 L1
SUBSYSTEM 2O395                (NUMBER OF FUNCTIONAL DEVICES: 129)
                                        --FUNCTIONAL CAPACITY (MB)--  % FUNCTIONAL CAPACITY
FDID  DEV   VOLSER    T/P  DEVICE  FUNCT                        NOT                    NOT   --PHYSICAL CAP USED (MB)--   COMP
      ADDR                 TYPE   CAP (MB)    ALLOC    STORED   STORED    ALLOC STORED STORED   SHARED   UNIQUE   TOTAL     RATIO
----  ----  ------    ---  ------ --------  -------- --------  --------   ----- ------ ------  -------- -------- --------   -----
006B  2B6B  RV2B6B     P   33903    2838.0     608.3     607.0    2231.0  21.4   21.4   78.6     116.5     58.6     175.1     3.5
006C  2B6C  RV2B6C     P   33903    2838.0     608.3     607.0    2231.0  21.4   21.4   78.6     116.6     58.6     175.2     3.5
006D  2B6D  RV2B6D     P   33903    2838.0     608.3     607.0    2231.0  21.4   21.4   78.6     116.6     58.6     175.2     3.5
006E  2B6E  RV2B6E     P   33903    2838.0     608.3     607.0    2231.0  21.4   21.4   78.6     116.5     58.6     175.1     3.5
006F  2B6F  RV2B6F     P   33903    2838.0     608.3     599.2    2238.8  21.4   21.1   78.9     115.0     57.8     172.8     3.5
0070  2B7O  RV2B7O     P   33903    2838.0     608.3     607.0    2231.0  21.4   21.4   78.6     116.6     58.6     175.2     3.5
0071  2B71  RV2B71     P   33903    2838.0     608.3     606.8    2231.2  21.4   21.4   78.6     116.5     58.5     175.0     3.5
0072  2B72  RV2B72     P   33903    2838.0     608.3     607.0    2231.0  21.4   21.4   78.6     116.6     58.6     175.2     3.5
0073  2B73  RV2B73     P   33903    2838.0     608.3     606.9    2231.1  21.4   21.4   78.6     116.5     58.5     175.0     3.5
0074  2B74  RV2B74     P   33903    2838.0     608.3     598.0    2240.0  21.4   21.1   78.9     114.9     57.7     172.6     3.5
0075  2B75  RV2B75     P   33903    2838.0    1449.7    1392.9    1445.1  51.1   49.1   50.9     551.9    276.2     828.2     1.7
0076  2B76  RV2B76     P   33903    2838.0    1449.7    1438.1    1399.9  51.1   50.7   49.3     563.0    281.8     844.7     1.7
0077  2B77  RV2B77     P   33903    2838.0    1449.7    1412.6    1425.4  51.1   49.8   50.2     560.0    280.3     840.3     1.7
0078  2B78  RV2B78     P   33903    2838.0    1449.7    1434.3    1403.7  51.1   50.5   49.5     566.3    283.4     849.7     1.7
0079  2B79  RV2B79     P   33903    2838.0    1449.7    1415.5    1422.5  51.1   49.9   50.1     561.2    280.9     842.2     1.7
007A  2B7A  RV2B7A     P   33903    2838.0    1449.7    1390.7    1447.3  51.1   49.0   51.0     551.0    275.8     826.8     1.7
007B  2B7B  RV2B7B     P   33903    2838.0    1449.7    1446.8    1391.2  51.1   51.0   49.0     568.1    284.3     852.4     1.7
007C  2B7C  RV2B7C     P   33903    2838.0    1449.7    1413.9    1424.1  51.1   49.8   50.2     560.7    280.6     841.4     1.7
007D  2B7D  RV2B7D     P   33903    2838.0    1449.7    1420.8    1417.2  51.1   50.1   49.9     562.3    281.4     843.7     1.7
007F  2B7F  RV2B7F     P   33903    2838.0    1448.2    1418.8    1419.3  51.0   50.0   50.0     562.7    281.6     844.4     1.7


  SELECTED DEVICES SUMMARY                      FUNCTIONAL CAPACITY (MB)  % FUNCT CAPACITY       -------- DISK ARRAY --------
                     SELECTED  TOTAL FUNCTIONAL                  NOT              NOT        -- PHYSICAL CAP USED (MB) --  COMP
                      DEVICES  CAPACITY (MB)          STORED    STORED      STORED STORED       SHARED     UNIQUE    TOTAL   RATIO
                     --------  -------------        --------- ---------    ------ ------       --------  --------  --------  -----
PRODUCTION PARTITION:   129     366104.1             214262.1  151842.1       58.5   41.5       7051.1   80195.2   87246.3    2.5
TOTALS:                 129     366104.1             214262.1  151842.1       58.5   41.5       7051.1   80195.2   87246.3    2.5


                                        IBM ITSO POKEEPSIE
                                  DB2 SNAPSHOT TEST - POST-DSN1COPY SPACE REPORT
XSA/REPORTER                        SPACE UTILIZATION SUMMARY REPORT                22JUN1998    15:17:21
                                                                                    SIBSPUT  V2 R1 L1
NUMBER OF          DISK ARRAY       NET CAPACITY LOAD(%)     COLL FREE SPACE (%)      UNCOLL FREE SPACE(%)
                  FUNCTIONAL DEVICES     CAPACITY (MB)     TEST   PROD  OVERALL     TEST   PROD  OVERALL     TEST   PROD  OVERALL
                  ------------------     -------------    -----  -----  -------    -----  -----  -------    -----  -----  -------
                           129             117880.2         0.0   71.2    71.2       0.0   28.2    28.2       0.0    0.6     0.6
```

Figure 25. Space Usage Report POST DSN1COPY

7. Start the target database. Use the START DATABASE command.
8. You can now issue queries against the new target database, which is a copy of the source.

SnapShot allows you to take an extract from your operational system with a minimum of disruption. If you need a point-in-time copy of the data for time-related analysis, SnapShot is the most effective way to make a copy.

Because the address translation and any further data cleansing and manipulation are performed against a new copy of the data (and not against the production system), there is no adverse impact on performance of the production system.

7.1 Performance Using SnapShot

In our example, the SNAPCRY job and the SNAPPLX job took about 50 seconds to complete. That is not a “typo.” That is 20 volumes of 3390-3 or about 25GB of DASD space copied in just under one minute. Most of the time is taken performing the allocation of the target data sets. This is because we only have to copy the pointers in the FTD (See Figure 1 on page 2). These pointers are

stored in cache memory, and it only takes a few seconds to copy them to another volume's pointers. There is no data movement, there is no channel I/O or cache usage, and there is no increase to the Net Capacity Load of the RVA. The result is a cataloged copy of the DB2 tables and indexes with different data set names. The DB2 internal identifiers, PSID, DBID and OBID are exactly the same as the original DB2 tables. These, of course, must be translated if the table is to be used by the DB2 subsystem for queries or other uses. The jobs shown in Figure 23 on page 35 and Figure 24 on page 36 perform this translation.

Chapter 8. How Should I Manage Data on an RVA?

The RVA architecture is based on a log structured file. A log structured file does not operate on an *update in place* basis. Instead all updated data is written to a new location in the RVA disk array. This has the automatic effect that all data and all I/O activity is evenly spread across all the physical disks in the array. This minimizes any contention on the disks.

The RVA virtual disk architecture has many benefits: it enables SnapShot and Compression, it eliminates the RAID write penalty seen on other RAID subsystems, and it reduces the cost of data storage. However, there are other benefits in the area of storage management that are of particular interest in a database environment. These are:

- Data placement
- Space management

8.1 Data Placement

DB2 administrators have typically wanted to control the placement of data on a storage subsystem for availability and performance reasons.

The high availability architecture of the RVA means that there is no need to separate data for availability reasons. All data is spread on all the disks in the subsystems, and these are protected by RAID 6.

Performance requirements usually include that high activity data sets, tables or indexes should not be placed on the same disk volume. This minimizes queueing on the disk hardware, and on the addressable unit (the volume, or the unit control block (UCB) in OS/390).

Queueing on hardware components is automatically minimized as the RVA spreads data across all the disks in the array. Because updated data is rewritten to a new physical location, this has the effect of a dynamically self-tuning system.

Queueing on the volume can also be minimized with the RVA; the virtual disk architecture allows you to define up to 256 addresses regardless of the capacity that you have installed. This allows you to spread any high-activity data sets (tables, indexes, or logs) across more volumes, or to dedicate a volume to a single high-activity data set, without the waste of disk space that would result with a traditional disk architecture.

The RVA architecture allows you to concentrate on planning only for contention on the volume, and to ignore physical disk contention. This simplifies the storage management effort significantly. Using SMS, you can automate data set allocation and placement. Once the data set has been allocated, it is unlikely that you will have to move it again for performance reasons.

8.2 Space Management

The RVA architecture consumes space in the disk array only when data is written to the volume. When a data set is allocated in a traditional disk subsystem, all the space that is requested is “reserved” so that it is available for the data to expand. This means that a disk volume, although it is 100% allocated to a data set, may actually only have a few tracks of data written. The rest is empty space, but it is empty space that you must pay for.

In an RVA, only the data that you write is actually written to the disk, although OS/390 still “reserves” the space in the VTOC. If more data is written to any volume in the RVA, space is taken out of the “pool” of data that is the net capacity load (NCL) of the RVA. Space on an RVA is managed at a storage subsystem level using the NCL.

Managing the subsystem by monitoring NCL can be done using IXFP functions or commands. First, you could issue the IXFP command DISPLAY NCL periodically. You could capture this data to a file that could be printed. You can also use the IXFP Reporting function and create a Space Utilization Report (See Figure 19 on page 32) to show the space usage on the subsystem and the individual volumes. An extract of the performance data can also be used to create a “comma-delimited” file, which can be downloaded to a PC and used to produce a graph of subsystem space usage using an Excel spreadsheet. Remember that the recommended NCL for an RVA subsystem is 75%. If the subsystem is running around that level, then there should be little need for concern. As the NCL rises above 75%, the subsystem monitors its own “Collected Free Space,” and if that number falls below 10% it begins to collect free space more aggressively. This event is documented to the user by a SIM message, REFCODE=3E41, and should be an indication that you should begin to migrate some less important data off the RVA subsystem. This task may be automatically started if you are using Netview or some similar console monitor.

In a traditional disk subsystem it is important that you do not allocate too much space in a data set if you do not plan to use it. In a database environment you usually need to allocate as much space as you will need in the future to minimize the need to reorganize. This conflict usually results in much time spent on space management and capacity planning.

The RVA simplifies this task. Provided you have enough space in the RVA (the total NCL) for the data that you write, you may allocate much more capacity than you actually have installed. Additional disk capacity can be added nondisruptively to the RVA. This means that since over-allocation does not incur the cost penalty that it does in traditional disk subsystems, there is no need to under-allocate and run the risk of space or fragmentation problems at a later date.

This may be of particular benefit when you have an application that starts small and later grows in size. Over-allocating the space initially enables you to add capacity only when you require it, without having to reload or reorganize your database.

8.3 Systems Managed Storage (SMS)

Managing the RVA in an SMS-managed environment is not any more difficult than traditional DASD. RVA Functional Volumes can be placed in SMS Storage Groups and allocations can be directed to those volumes using standard SMS Storage Class and Management Class ACS routines. The RVA is an RAMAC Array DASD subsystem and SMS will recognize it as a high availability device. It also is a cached control unit that uses DASD Fast Write, so data sets which have a low MSR value as a requirement will be allocated to RVA Functional Volumes that reside within the assigned Storage Groups.

Further enhancements to DFSMS 1.4 now allow the SnapShot capability of the RVA to be a known characteristic and an accessibility attribute that can be used to direct allocations where the SnapShot capability is required for backup or data extraction as we have discussed in this document.

For SnapShot to succeed, it is required that the target data set be allocated in the same physical RVA subsystem as the source data set. To help ensure that this proper allocation occurs, IXFP provides volume preferencing exits that interface with the DADSM allocation process and provide a list of volumes that reside in the same physical RVA subsystem as the source data set, and rejects attempts to select other volumes not in the same physical subsystem.

The RVA supports Extended Format data sets and by using the Data Class ACS routine, an Extended Format data set may be created and stored on the RVA disk array. Data Striping already occurs in the RVA subsystem, but the addition of the Extended Format striped data set allocated on multiple RVA functional volumes allows the host to do parallel I/O to these volumes and increase bandwidth to the data set. In addition, SnapShot supports snapping Extended Format data sets and striped data sets as long as they are non-VSAM.

Chapter 9. How Do I Size an RVA for a Business Intelligence (BI) Environment?

When sizing for the BI environment, one normally would calculate the size of the data to be stored and the number of copies to be made for various information or reporting functions and that would be the end of it. The total amount would be the required DASD space needed. However, when using RVA subsystems, there are some other considerations that may reduce the amount of DASD space required. First, if cloning of DB2 subsystems is to be done and these DB2 subsystem will be used on another host environment, then the DASD space may not have to be totally reproduced and significant space savings can result. Also, once a SnapShot is taken of the production data, that copy can be snapped over and over again, creating as many copies as needed. If the data is primary read-only data, then this will have very little impact on the overall NCL of the subsystem.

Chapter 10. Summary

The IBM RVA DASD subsystem and SnapShot provide a number of benefits for DB2 for OS/390-based business intelligence applications. These include:

- Efficient use of DASD space

  The operational databases and files from which data for data warehouses are extracted can be very large. SnapShot provides the capability to make copies of these data sources with little or no DASD space required for the copies (depending on the frequency with which the source data is updated).

- Minimized backup window

  With SnapShot and the DB2 for OS/390 Concurrent Copy capability, data warehouse databases can be backed up within a small read-only window of time. This capability is especially beneficial when the data warehouse database is required to be in an update-access state virtually around the clock.

- Minimized extract window

  With operational databases and files, near-continuous update availability is often the rule. SnapShot allows point-in-time extracts to be created for subsequent input into a data warehouse, with minimal disruption of source data update processes.

- Optimized data placement

  Business intelligence applications are typically characterized by multiple, concurrently executing scans of very large amounts of data. The spreading of data in a single RVA logical volume across multiple physical disk drives helps to prevent contention that could slow the execution of parallel data scans, thereby enhancing the effectiveness of the DB2 for OS/390 query CPU parallelism capability.

- Fast database cloning

  RVA and SnapShot provide the capability to quickly clone an entire database, which is useful for backup purposes or to move a database to another OS/390 server.

- Reliable access to data

  Increasingly, the availability requirements placed by organizations on their data warehouse databases are the same as those used for operational databases. The RVA DASD subsystem answers this need with the proven reliability of the RAID DASD architecture.

- Non-disruptive servicing

  Hardware service technicians can perform repair and maintenance operations on an RVA DASD subsystem without disrupting data access, further enhancing the availability of critical business intelligence applications.

Appendix A. Special Notices

This publication is intended to help people who plan to use RVA and SnapShot for BI Applications with DB2 on OS/390. The information in this publication is not intended as the specification of any programming interfaces that are provided by SnapShot and DB2. See the PUBLICATIONS section of the IBM Programming Announcement for SnapShot for more information about what publications are considered to be product documentation.

References in this publication to IBM products, programs or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM product, program, or service is not intended to state or imply that only IBM′s product, program, or service may be used. Any functionally equivalent program that does not infringe any of IBM′s intellectual property rights may be used instead of the IBM product, program or service.

Information in this book was developed in conjunction with use of the equipment specified, and is limited in application to those specific hardware and software products and levels.

IBM may have patents or pending patent applications covering subject matter 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 the IBM Director of Licensing, IBM Corporation, 500 Columbus Avenue, Thornwood, NY 10594 USA.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact IBM Corporation, Dept. 600A, Mail Drop 1329, Somers, NY 10589 USA.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The information contained in this document has not been submitted to any formal IBM test and is distributed AS IS. The use of this information or the implementation of any of these techniques is a customer responsibility and depends on the customer′s ability to evaluate and integrate them into the customer′s operational environment. While each item may have been reviewed by IBM for accuracy in a specific situation, there is no guarantee that the same or similar results will be obtained elsewhere. Customers attempting to adapt these techniques to their own environments do so at their own risk.

Any pointers in this publication to external Web sites are provided for convenience only and do not in any manner serve as an endorsement of these Web sites.

Reference to PTF numbers that have not been released through the normal distribution process does not imply general availability. The purpose of including these reference numbers is to alert IBM customers to specific information relative to the implementation of the PTF when it becomes available to each customer according to the normal IBM PTF distribution process.

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

AIX	AIX/6000
AS/400	BookManager
DB2	DFSMS
DFSMS/MVS	DFSMSdss
IBM	MVS/ESA
RAMAC	RISC System/6000
RS/6000	S/390
S/390 Parallel Enterprise Server	System/38
System/390	S/390 Parallel Enterprise Server

The following terms are trademarks of other companies:

C-bus is a trademark of Corollary, Inc.

Java and HotJava are trademarks of Sun Microsystems, Incorporated.

Microsoft, Windows, Windows NT, and the Windows 95 logo are trademarks or registered trademarks of Microsoft Corporation.

PC Direct is a trademark of Ziff Communications Company and is used by IBM Corporation under license.

Pentium, MMX, ProShare, LANDesk, and ActionMedia are trademarks or registered trademarks of Intel Corporation in the U.S. and other countries.

UNIX is a registered trademark in the United States and other countries licensed exclusively through X/Open Company Limited.

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

Appendix B. Related Publications

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

B.1 International Technical Support Organization Publications

For information on ordering these ITSO publications see “How to Get ITSO Redbooks” on page 51.

- *Implementing DFSMSdss SnapShot and Virtual Concurrent Copy*, SG24-5268
- *Implementing Concurrent Copy*, GG24-3990
- *Implementing SnapShot*, SG24-2241
- *RAMAC Virtual Array*, SG24-4951
- *DB2 for OS/390 and Data Compression*, SG24-5261, (available 3Q/98)

B.2 Redbooks on CD-ROMs

Redbooks are also available on CD-ROMs. **Order a subscription** and receive updates 2-4 times a year at significant savings.

CD-ROM Title	Subscription Number	Collection Kit Number
System/390 Redbooks Collection	SBOF-7201	SK2T-2177
Networking and Systems Management Redbooks Collection	SBOF-7370	SK2T-6022
Transaction Processing and Data Management Redbook	SBOF-7240	SK2T-8038
Lotus Redbooks Collection	SBOF-6899	SK2T-8039
Tivoli Redbooks Collection	SBOF-6898	SK2T-8044
AS/400 Redbooks Collection	SBOF-7270	SK2T-2849
RS/6000 Redbooks Collection (HTML, BkMgr)	SBOF-7230	SK2T-8040
RS/6000 Redbooks Collection (PostScript)	SBOF-7205	SK2T-8041
RS/6000 Redbooks Collection (PDF Format)	SBOF-8700	SK2T-8043
Application Development Redbooks Collection	SBOF-7290	SK2T-8037

B.3 Other Publications

These publications are also relevant as further information sources:

- *Using SnapShot Version 1 Release 2*, SC26-7173
- *SnapShot for MVS/ESA Installating and Using Version 1 Release 2*, SC26-7174
- *SnapShot Installation Notes for MVS Version 1 Release 2*, SC26-7175
- *Implementing SnapShot for Version 1 Release 2*, SC26-7176

How to Get ITSO Redbooks

This section explains how both customers and IBM employees can find out about ITSO redbooks, CD-ROMs, workshops, and residencies. A form for ordering books and CD-ROMs is also provided.

This information was current at the time of publication, but is continually subject to change. The latest information may be found at http://www.redbooks.ibm.com/.

How IBM Employees Can Get ITSO Redbooks

Employees may request ITSO deliverables (redbooks, BookManager BOOKs, and CD-ROMs) and information about redbooks, workshops, and residencies in the following ways:

- **Redbooks Web Site on the World Wide Web**

  http://w3.itso.ibm.com/

- **PUBORDER** — to order hardcopies in the United States
- **Tools Disks**

  To get LIST3820s of redbooks, type one of the following commands:

  ```
  TOOLCAT REDPRINT
  TOOLS SENDTO EHONE4 TOOLS2 REDPRINT GET SG24xxxx PACKAGE
  TOOLS SENDTO CANVM2 TOOLS REDPRINT GET SG24xxxx PACKAGE (Canadian users only)
  ```

  To get BookManager BOOKs of redbooks, type the following command:

  ```
  TOOLCAT REDBOOKS
  ```

  To get lists of redbooks, type the following command:

  ```
  TOOLS SENDTO USDIST MKTTOOLS MKTTOOLS GET ITSOCAT TXT
  ```

  To register for information on workshops, residencies, and redbooks, type the following command:

  ```
  TOOLS SENDTO WTSCPOK TOOLS ZDISK GET ITSOREGI 1998
  ```

- **REDBOOKS Category on INEWS**
- **Online** — send orders to: USIB6FPL at IBMMAIL or DKIBMBSH at IBMMAIL

Redpieces

For information so current it is still in the process of being written, look at "Redpieces" on the Redbooks Web Site (http://www.redbooks.ibm.com/redpieces.html). Redpieces are redbooks in progress; not all redbooks become redpieces, and sometimes just a few chapters will be published this way. The intent is to get the information out much quicker than the formal publishing process allows.

How Customers Can Get ITSO Redbooks

Customers may request ITSO deliverables (redbooks, BookManager BOOKs, and CD-ROMs) and information about redbooks, workshops, and residencies in the following ways:

- **Online Orders** — send orders to:

	IBMMAIL	**Internet**
In United States:	usib6fpl at ibmmail	usib6fpl@ibmmail.com
In Canada:	caibmbkz at ibmmail	lmannix@vnet.ibm.com
Outside North America:	dkibmbsh at ibmmail	bookshop@dk.ibm.com

- **Telephone Orders**

United States (toll free)	1-800-879-2755
Canada (toll free)	1-800-IBM-4YOU
Outside North America	(long distance charges apply)
(+45) 4810-1320 - Danish	(+45) 4810-1020 - German
(+45) 4810-1420 - Dutch	(+45) 4810-1620 - Italian
(+45) 4810-1540 - English	(+45) 4810-1270 - Norwegian
(+45) 4810-1670 - Finnish	(+45) 4810-1120 - Spanish
(+45) 4810-1220 - French	(+45) 4810-1170 - Swedish

- **Mail Orders** — send orders to:

IBM Publications Publications Customer Support P.O. Box 29570 Raleigh, NC 27626-0570 USA	IBM Publications 144-4th Avenue, S.W. Calgary, Alberta T2P 3N5 Canada	IBM Direct Services Sortemosevej 21 DK-3450 Allerød Denmark

- **Fax** — send orders to:

United States (toll free)	1-800-445-9269
Canada	1-403-267-4455
Outside North America	(+45) 48 14 2207 (long distance charge)

- **1-800-IBM-4FAX (United States)** or **(+1)001-408-256-5422 (Outside USA)** — ask for:

Index # 4421 Abstracts of new redbooks
Index # 4422 IBM redbooks
Index # 4420 Redbooks for last six months

- **On the World Wide Web**

Redbooks Web Site	http://www.redbooks.ibm.com/
IBM Direct Publications Catalog	http://www.elink.ibmlink.ibm.com/pbl/pbl

Redpieces

For information so current it is still in the process of being written, look at "Redpieces" on the Redbooks Web Site (`http://www.redbooks.ibm.com/redpieces.html`). Redpieces are redbooks in progress; not all redbooks become redpieces, and sometimes just a few chapters will be published this way. The intent is to get the information out much quicker than the formal publishing process allows.

IBM Redbook Order Form

Please send me the following:

Title	Order Number	Quantity

First name Last name

Company

Address

City Postal code Country

Telephone number Telefax number VAT number

- Invoice to customer number
- Credit card number

Credit card expiration date Card issued to Signature

We accept American Express, Diners, Eurocard, Master Card, and Visa. Payment by credit card not available in all countries. Signature mandatory for credit card payment.

List of Abbreviations

APAR	authorized program analysis report
BI	business intelligence
BSDS	boot strap data set
CKD	count key data
CLIST	command list
CPU	central processing unit
DADSM	direct access device space management
DASD	direct access storage device
DB	data base
DBID	data base ID
DBMS	data base management system (System/38)
DD	data definition
DDL	data definition language
DEV	device
DFSMS	data facility storage managed subsystem
DSN	data set name
ERP	Enterprise Resource Planning (new paradigm going one step past MRP and MES)
FBA	fixed block architecture
FDT	functional device table
FTD	functional track directory
FTT	functional track table
GB	gibabyte
I/O	input/output
IBM	International Business Machines Corporation
ISDBID	index ID
ITSO	International Technical Support Oraganization
IXPF	IBM Extended Facilities Program
JCL	job control language (OS/390 and VSE)
KB	kilobyte, 1000 bytes (1024 bytes memory)
LRSN	log record sequence number
MB	megabyte, 1,000,000 bytes (1,048,576 bytes memory)
MVS	multiple virtual storage (IBM System 370 & 390)
NCL	net capacity load
OBID	object ID
PSID	page segment ID
PTF	program temporary fix
RAID	redundant array of independent disks
RAMAC	brand name and trademark of IBM. Note: This is *NOT* ″RAID Architecture With Multilevel Adaptive Cache″
RBA	relative byte address
RVA	RAMAC Virtual Array
SMS	systems managed storage
SPE	small programming enhancement
TNT	track number table
TSO	time sharing option
UCB	IOS unit control block (OS/390 control block)
URL	Universal Resource Locator
VCAT	VSAM catalogue
VCC	virtual concurrent copy
VOLSER	volume serial
VSAM	virtual storage access method (IBM)
VTOC	volume table of contents
VVDS	VSAM volume data set
WSDS	working space data set

Index

ITSO Redbook Evaluation

Using RVA and SnapShot for Business Intelligence Applications with OS/390 and DB2
SG24-5333-00

Your feedback is very important to help us maintain the quality of ITSO redbooks. **Please complete this questionnaire and return it using one of the following methods:**

- Use the online evaluation form found at http://www.redbooks.ibm.com
- Fax this form to: USA International Access Code + 1 914 432 8264
- Send your comments in an Internet note to redbook@us.ibm.com

Which of the following best describes you?
__Customer __Business Partner __Solution Developer __IBM employee
__None of the above

Please rate your overall satisfaction with this book using the scale:
(1 = very good, 2 = good, 3 = average, 4 = poor, 5 = very poor)

Overall Satisfaction _____

Please answer the following questions:

Was this redbook published in time for your needs? Yes____ No____

If no, please explain:

_____

_____

_____

_____

What other redbooks would you like to see published?

_____

_____

_____

Comments/Suggestions: **(THANK YOU FOR YOUR FEEDBACK!)**

_____

_____

_____

_____

_____

SG24-5333-00
Printed in the U.S.A.