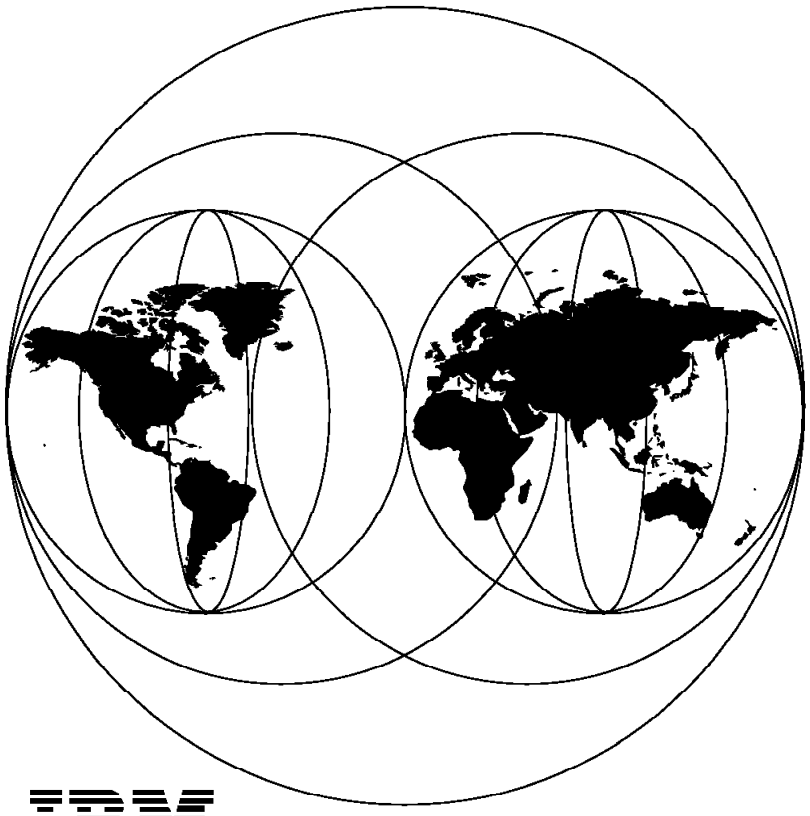
International Technical Support Organization

SG24-4647-00

SQL/DS Version 3 Release 5 Usage Guide

December 1995



IBM

International Technical Support Organization
Boeblingen Center

International Technical Support Organization

SG24-4647-00

SQL/DS Version 3 Release 5 Usage Guide

December 1995

Take Note!

Before using this information and the product it supports, be sure to read the general information under “Special Notices” on page xiii.

First Edition (December 1995)

This edition applies to Version 3 Release 5 Modification 0 of Structured Query Language/Data System for use with VM or VSE.

Order publications through your IBM representative or the IBM branch office serving your locality. Publications are not stocked at the address given below.

An ITSO Technical Bulletin Evaluation Form for reader′s feedback appears facing Chapter 1. If the form has been removed, comments may be addressed to:

IBM Canada Ltd. Laboratory
Information Development
2G/345/1150/TOR
1150 Eglinton Avenue East
North York, Ontario, Canada. M3C 1H7

You can also send your comments by facsimile to (416) 448-6161 addressed to the attention of the RCF coordinator. If you have access to Internet, you can send your comments electronically to **torrcf@vnet.ibm.com;** IBMLINK, to **toribm(torrcf);** IBM/PROFS, to **torolab4(torrcf);** IBMMAIL, to **ibmmail(caibmwt9).**

If you choose to respond through Internet, please include either your entire Internet network address, or a postal address.

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.

Abstract

This document describes the new functions and enhancements of SQL/DS V3R5 in a VM or VSE environment. It also provides information on the administration and usage of SQL/DS V3R5 features and functions.

This document is intended for customers who plan to migrate, install and use SQL/DS V3R5 under both VM and VSE operating systems.

(221 pages)

Contents

Figures

Tables

Special Notices

This publication is intended to provide customers with guidance in the setup, usage and administration of the new features of SQL/DS V3R5.

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.

Any performance data contained in this document was determined in a controlled environment, and therefore, the results that may be obtained in other operating environments may vary significantly. Users of this document should verify the applicable data for their specific environment.

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:

ACF/VTAM
AIX
APPN
C/370
CICS/VSE
DATABASE 2
DataPropagator
DB2
DB2/6000
DFSMS/VM

DRDA
ESA/370
ESCON
ISSC
MVS/ESA
OS/2
PROFS
RS/6000
SQL/DS
SQL/400
Systems Application Architecture

AD/Cycle
AIX/6000
AS/400
CICS
Current
DataHub
DataRefresher
DB2/2
DFSMS
Distributed Relational Database Architecture
ES/9000
ESA/390
IBM
Language Environment
Operating System/400
OS/400
QMF
SAA
Structured Query Language/Data System
System/390
SystemView

VisualAge	VM/ESA
VM/XA	VSE/ESA
VTAM	

The following terms are trademarks of other companies:

DOS	Microsoft Corporation
HP	Hewlett Packard Corporation
Intel	Intel Corporation
Microsoft	Microsoft Corporation
PC Direct	Ziff Communications Company (used by IBM Corporation under license)
Solaris	Sun Microsystems, Inc.
UNIX	X/Open Company Ltd. (registered trademark)
X/Open	X/Open Company Ltd.
Windows	Microsoft Corporation

Other trademarks are trademarks of their respective companies.

Preface

This document is intended to provide technical recommendations, hints and techniques on how to set up and use the new features of SQL/DS V3R5. It contains a complete description of the new SQL/DS V3R5 product functions and enhancements.

This document is the result of one ITSO Boeblingen residency and will benefit both casual and experienced SQL/DS users, SQL/DS application programmers, SQL/DS Database Administrators and all personnel who are directly involved with SQL/DS usage.

The new documentation for SQL/DS V3R5 consists of the following manuals:

1. SQL/DS Installation for IBM VM Systems (GH09-8078)
2. SQL/DS Installation for VSE (GH09-8090)
3. SQL/DS Messages and Codes for IBM VM Systems (SH09-8079)
4. SQL/DS Messages and Codes for VSE (SH09-8091)
5. IBM SQL/DS Data Restore Guide (SC09-2275)
6. All the previously existing SQL/DS V3R4 manuals

   Except for the above mentioned SQL/DS V3R5 manuals, all the other previously existing SQL/DS V3R4 manuals will not be modified.
7. SQL/DS Version 3 Release 5 Usage Guide (SG24-4647)

   This is the only external document in which the new SQL/DS V3R5 functions and enhancements are described.

How This Document is Organized

This document is organized as follows:

- Chapter 1, “Introduction to SQL/DS V3R5”

  This chapter gives a brief overview of SQL/DS, all of the new and changed commands, functions and procedures that are available with SQL/DS V3R5. It also describes the various advantages and gains of migrating to SQL/DS V3R5.
- Chapter 2, “How to Use the New SQL/DS V3R5 Functions”

  This chapter describes how to use the newly available functions.
- Chapter 3, “Performance Benefits of SQL/DS V3R5”

  This chapter discusses performance benefits of SQL/DS V3R5.
- Chapter 4, “Archive and Recovery Strategies”

  This chapter provides several usage scenarios for backup, restore and archive functions using the SQL/DS base product features, the Data Base Services Utility and the new Data Restore feature.
- Chapter 5, “Installation, Migration and Coexistence”

  This chapter describes considerations for installation of SQL/DS V3R5, migration from SQL/DS V2R2 and later, and coexistence with SQL/DS V3R3 and later.

- Chapter 6, “Documentation Changes Introduced with SQL/DS V3R5”

  This chapter describes documentation changes introduced with SQL/DS V3R5.

- Chapter 7, “SQL/DS Client/Server Solutions”

  This chapter describes some ways in which SQL/DS can be used in a Client/Server environment.

- Appendix A, “SQL/DS Version 3 Enhancements Summary”

  This appendix describes all the enhancements in SQL/DS since Version 3 Release 1.

- Appendix B, “SQL/DS Version 3 Features Summary”

  This appendix describes the following SQL/DS features: RXSQL, VMDSS and DataHub.

- Appendix C, “Field Procedures for Cultural Sorts”

  This appendix provides information on using field procedures to perform cultural sorts.

- Appendix D, “Database Directory Expansion Examples”

  This appendix provides examples of the new directory expansion enhancement, and compares it with the function that was available in earlier versions.

Related Publications

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

For the VM environment:

- *SQL/DS Database Administration for IBM VM Systems*, GH09-8083
- *SQL/DS System Administration for IBM VM Systems*, GH09-8084
- *SQL/DS Installation for IBM VM Systems*, GH09-8078
- *SQL/DS Application Programming for IBM VM Systems*, SH09-8086
- *SQL/DS Database Services Utility for IBM VM Systems*, SH09-8088
- *SQL/DS Operation for IBM VM Systems*, SH09-8080
- *SQL/DS Messages and Codes for IBM VM Systems*, SH09-8079
- *SQL/DS Diagnosis Guide and Reference for IBM VM Systems*, LH09-8081
- *SQL/DS SQL Reference for IBM VM Systems and VSE*, SH09-8087
- *SQL/DS Interactive SQL Guide and Reference for IBM VM Systems*, SH09-8085
- *SQL/DS General Information for IBM VM Systems*, GH09-8074
- *Managing the SQL/DS Environment*, SH09-8109
- *SQL/DS Master Index and Glossary for IBM VM Systems*, SH09-8089
- *SQL/DS Quick Reference for IBM VM Systems*, SX09-1140
- *SQL/DS Solutions Directory*, GX09-1218

- *SQL/DS Performance Tuning Handbook for IBM VM Systems and VSE*, SH09-8111
- *VM Data Spaces Support for VM/ESA*, SH09-8107
- *The VM/ESA Systems Handbook*, SB20-0021
- *VM/ESA Performance*, SC24-5642

For the VSE environment:

- *SQL/DS Database Administration for VSE*, GH09-8095
- *SQL/DS System Administration for VSE*, GH09-8096
- *SQL/DS Installation for VSE*, GH09-8090
- *SQL/DS Application Programming for VSE*, SH09-8098
- *SQL/DS Database Services Utility for VSE*, SH09-8100
- *SQL/DS Messages and Codes for VSE*, SH09-8091
- *SQL/DS Operation for VSE*, SH09-8092
- *SQL/DS SQL Reference for IBM VM Systems and VSE*, SH09-8087
- *SQL/DS Interactive SQL Guide and Reference for VSE*, SH09-8097
- *SQL/DS Diagnosis Guide and Reference for VSE*, LH09-8093
- *SQL/DS General Information for VSE*, GH09-8075
- *Managing the SQL/DS Environment*, SH09-8109
- *SQL/DS Master Index and Glossary for VSE*, SH09-8101
- *SQL/DS Quick Reference for VSE*, SX09-1140
- *SQL/DS Performance Tuning Handbook for IBM VM Systems and VSE*, SH09-8111
- *SQL/DS Solutions Directory*, GX09-1218

Publication to be available soon

IBM SQL/DS Data Restore Guide, SC09-2275

International Technical Support Organization Publications

- *Advantages of Migrating to SQL/DS Version 3 Release 4*, GG24-4264-00
- *SQL/DS Version 3 Release 4 Performance Guide*, GG24-4047-01
- *SQL/DS Version 3 Release 4 VMDSS Exploitation*, GG24-4198-00
- *Database Systems Management: IBM SystemView Information Warehouse. DataHub Implementation and Connectivity*, GG24-4031-00
- *Setup and Usage of SQL/DS in a DRDA Environment*, GG24-3733-01
- *SQL/DS Version 3 Release 4 VMDSS Exploitation*, GG24-4198-00
- *DRDA Cross Platform Connectivity and Application*, GG24-4311-00

A complete list of International Technical Support Organization publications, known as redbooks, with a brief description of each, may be found in:

International Technical Support Organization Bibliography of Redbooks, GG24-3070.

To get a catalog of ITSO redbooks, VNET users may type:

```
TOOLS SENDTO WTSCPOK TOOLS REDBOOKS GET REDBOOKS CATALOG
```

A listing of all redbooks, sorted by category, may also be found on MKTTOOLS as ITSOCAT TXT. This package is updated monthly.

How to Order ITSO Redbooks

IBM employees in the USA may order ITSO books and CD-ROMs using PUBORDER. Customers in the USA may order by calling 1-800-879-2755 or by faxing 1-800-284-4721. Visa and MasterCard are accepted. Outside the USA, customers should contact their local IBM office. For guidance on ordering, send a PROFS note to BOOKSHOP at DKIBMVM1 or E-mail to bookshop@dk.ibm.com.

Customers may order hardcopy ITSO books individually or in customized sets, called BOFs, which relate to specific functions of interest. IBM employees and customers may also order ITSO books in online format on CD-ROM collections, which contain redbooks on a variety of products.

ITSO Redbooks on the World Wide Web (WWW)

Internet users may find information about redbooks on the ITSO World Wide Web home page. To access the ITSO Web pages, point your Web browser to the following URL:

```
http://www.redbooks.ibm.com/redbooks
```

IBM employees may access LIST3820s of redbooks as well. The internal redbooks home page may be found at the following URL:

```
http://w3.itsc.pok.ibm.com/redbooks/redbooks.html
```

Acknowledgments

This project was designed and managed by:

Luciano Jose dos Santos
International Technical Support Organization, Boeblingen Center

The authors of this document are:

Christine Jones
ISSC Australia

Marcelo Nicolozzi
IBM Brazil

Mauricio Sordo
ISSC Australia

This publication is the result of a residency conducted at the International Technical Support Organization, Boeblingen Center.

Thanks to the following people for the invaluable advice and guidance provided in the production of this document:

SQL/DS Team
IBM SWS Toronto Laboratory

Chapter 1. Introduction to SQL/DS V3R5

Structured Query Language/Data System (SQL/DS) is the IBM Relational Database Management System for the VM and VSE environments that manages, stores and retrieves data. It consists of a relational database system, a query language and a utility for transferring data and other objects.

SQL/DS supports concurrent access for both interactive and batch applications. It also offers powerful facilities for data integrity, security and recovery.

SQL/DS V3R5 offers your company improved continuous operations and distributed database solutions in the VSE and VM environments. The enhancements in Version 3 Release 5 allow you to take advantage of the strengths of both host and distributed database solutions while leveraging your existing application investments.

SQL/DS is a vital component of IBM′s Information Warehouse strategy and is a key member of the DB2 product family. It represents the strategic relational database product for the VSE and VM operating environments. This new release underscores SQL/DS′s long-standing membership in the DB2 family and IBM′s continuing focus on ensuring compatibility with other IBM solutions.

Highlights of SQL/DS V3R5:

- Provides an enhanced recovery solution
- Improves database availability towards the goal of 24 x 7 operations
- Improves existing archive performance
- Improves compatibility with other IBM DB2 family products
- Improves exploitation of large real storage processors
- Improves support for distributed database applications
- Provides a base for improved decision support

SQL/DS will continue to be the strategic relational database product for the VSE and VM system environments.

Many of the new functions implemented in this release are in response to customer feedback and requests from around the world.

1.1 What is New in SQL/DS V3R5

This topic provides a brief overview of SQL/DS, with all of the new and changed commands, functions and procedures that are available with SQL/DS V3R5. For a detailed explanation of each of these, please refer to Chapter 2, “How to Use the New SQL/DS V3R5 Functions” on page 11.

1.1.1 New Functions and Capabilities

1.1.1.1 CICS Database Switching

This new function enables online CICS transactions to switch among multiple SQL/DS application servers without having to bring down the Online Resource Adapter. Transactions can only switch servers between logical units of work.

This function is applicable in VSE and Guest Sharing.

1.1.1.2 DRDA Limited Accounting String

This function enables the following scenarios:

- Allows SQL/DS (VM) requesters to send up to 16 bytes of installation-dependent data to any SQL/DS server (both VM and VSE) using the DRDA protocol, which are then recorded into USER accounting records.
- Allows SQL/DS (VM) requesters to send accounting information to DRDA servers, from which accounting records may be generated. DB2 for MVS is an example of such a server.
- Allows DRDA requesters to send accounting data to SQL/DS servers (both VM and VSE), from which 16 bytes of user supplied data are recorded into SQL/DS USER accounting records. Examples of such requesters are DB2 for MVS, DDCS for OS/2 (Version 2), DDCS for AIX (Version 2) and SQL/DS.

This function is available for both VM and VSE.

1.1.1.3 Additional CCSID Support

As the number of CCSIDs supported on databases such as DB2/2 and DB2/6000 increases, the demand for support for conversion from these new CCSIDs to compatible host CCSIDs increases as well. This enhancement adds support for Traditional Chinese, Simplified Chinese, Korean, Cyrillic, Windows and Greek code pages.

This enhancement is available for both VM and VSE.

1.1.1.4 Additional National Language Support

This enhancement provides sample field procedures to change the default collating sequence to accommodate the Latin 2 or Cyrillic cultural sort sequence for these regions:

- Slovenia
- Poland
- Romania
- Russia
- Bulgaria
- Serbia
- Montenegro

Without a field procedure, string data is sorted based on the System/390 collating sequence, which causes two major problems. The first is that the System/390 collating sequence is not the appropriate collating sequence for some European alphabets. The second is that SQL/DS cannot collate double

byte vocals correctly; it considers the two component characters individually rather than as a single character.

The technical solution for both problems is to code the field procedures for the affected columns.

1.1.1.5 FORCE without DISABLE

This enhancement changes the SQLCODE returned to applications when the FORCE command is issued without the DISABLE option. SQL/DS will now return an SQLCODE -948 instead of SQLCODE -933. This allows the application program to complete its pending work for example, closing its currently open files, sending messages to the operator console, and releasing previously allocated program resources.

This enhancement is available in VM only.

1.1.2 Performance Enhancements

1.1.2.1 Increased Buffers Maximums

The maximum number of buffers allowed in the database is increased to a new value of 400000, for both directory and data buffers. Increasing the buffer limit allows you to keep more data in storage and exploit all the storage you have available. It may reduce the number of I/Os, thereby improving performance.

This new value is available for both VM and VSE.

1.1.2.2 Archive Performance Enhancements

With this enhancement, SQL/DS database and log archives only archive allocated pages and use increased I/O blocking. This will decrease the time required for archives.

This will be covered in 3.2, “Archive Performance Improvement” on page 88.

1.1.2.3 Utilizing Virtual Disks for VM and VSE

If the operating system is VSE/ESA 1.3 or later or VM/ESA 1.2.1 or later, the internal dbspaces can make use of a virtual disk to improve performance. This virtual disk support allows the use of a data space as a virtual disk. Data that would otherwise reside on DASD can be stored in a virtual disk and since the virtual disk uses main storage instead of DASD, it is much faster than a conventional disk. The virtual disk appears to any program or job as just another disk, only it is faster.

Virtual disk storage, however, is temporary. Anything in a virtual disk is lost whenever the VSE or VM operating system is stopped or restarted. For this reason, **do not** use virtual disks for anything other than internal dbspaces. It does not matter if internal dbspace contents are lost as they are only used as temporary workspace.

While limited to the use of internal dbspaces only, virtual disks will improve the performance of index creation, joins, sorts, and other operations that require temporary workspace.

1.1.2.4 Assembler Host Variables/Even Precision

This enhancement allows the SQL/DS Assembler preprocessor to recognize host variables declared as even precision packed decimal. This supports the use of sargable predicates in Assembler programs, as opposed to residual predicates, when even precision decimal datatype host variables are used. Sargable predicates are much more efficient than residual predicates.

This is available for both VM and VSE.

1.1.3 Usability Enhancements

1.1.3.1 Enhanced Directory Expansion

As the database is used and grows in size, it may run short of either logical or physical space. Currently the directory can be expanded to accommodate more logical pages by expanding the page map table. This enhancement adds support for expanding the directory to increase both the logical and physical space, by expanding the page map table and the allocation bit maps concurrently. This will avoid the need for a database regeneration when running out of directory space. This increases database availability.

This enhancement is available for both VM and VSE.

1.1.3.2 C/370 Decimal Data Type Support

The decimal data type was added to the C/370 compiler in the AD/Cycle C/370 Compiler Version 1 Release 2. This new function allows C/370 applications to use variables declared with the new C/370 decimal data type as SQL/DS host variables. This also allows the SQL/DS DECIMAL data type to be used in C/370 programs, which was previously unsupported. The C/370 fixed decimal type can now be used along with the previously available float or double types as a host language variable for the SQL/DS DECIMAL data type.

This enhancement is available for both VM and VSE.

1.1.3.3 CICS Information in SHOW CONNECT

This new function will help operators identify which agent should be FORCEd by displaying the CICS task number, the CICS terminal ID, and the RMID for all local CICS users as part of the output for the SHOW CONNECT command.

This function is available for VSE and Guest Sharing Users.

1.1.3.4 PACKAGE NAME in SHOW CONNECT

This enhancement allows the package name to be displayed in the SHOW CONNECT command from the operator console. Before this enhancement, only an ID with DBA authority, or the creator of the package, could receive this information.

This enhancement is available for both VM and VSE.

1.1.3.5 ISQL Print Routing

A new option **TOUser** has been added to the SET PRINTRoute command. When the ISQL user specifies SET PRINTRoute TOUser *userid* and subsequently issues the PRINT command (without specifying TOUser) within the same ISQL session, ISQL spools the print output in the same way that it will spool the print output when the user enters PRINT TOUser *userid*.

This function is applicable in VSE only.

1.1.4 Reliability, Availability and Serviceability

1.1.4.1 Default DUMPTYPE = F

This enhancement changes the default DUMPTYPE value from “P” to “F.” This causes SQL/DS V3R5 to take full dumps by default instead of partial dumps. It increases the usefulness of the information that is generated if a problem occurs and the customer will not be required to try to reproduce a problem in order to obtain a full dump for IBM service.

This enhancement is available for both VM and VSE.

1.1.4.2 Support for LE/VSE in Single User Mode

Previously, SQL/DS error condition handling assumed that neither the application nor any associated compiler runtime code could set their own abnormal termination exits through the STXIT VSE macro.

This enhancement provides support for the new LE/VSE TRAP option. When TRAP is active, it needs control over condition handling. Consequently, this support coordinates error condition handling between LE/VSE and SQL/DS and adapts the existing condition handling logic of SQL/DS to the new situation.

This enhancement is available in VSE only.

1.1.5 Standards Support

1.1.5.1 SQLSTATE Changes for SQL92

SQL/DS V3R5 is implementing SQLSTATE changes to support the SQL92 standard and to be consistent with enhancements made by other DB2 family members.

This enhancement is available for both VM and VSE.

1.1.6 New Product Feature

1.1.6.1 SQL/DS Data Restore

This new feature provides customers with many new archive and recovery options designed to help achieve continuous operation. Highlights include:

- Table Level Recovery
- Forward Log Recovery with Point in Time Recovery
- Dual Backup to TAPE, DISK, or both.

This feature is available for both VSE and VM.

1.2 Environments Supported

1.2.1 Products

1.2.1.1 Processors

SQL/DS V3R5 will run on any IBM processor, with the conditional swapping and extended precision floating point features, supported by one of the following operating systems:

- VM/ESA Version 1 Release 2.1 or later (ESA feature only)
- VSE/ESA Version 1 Release 1 or later

  Note: VSE/ESA Version 1 Release 3 or later is required for Virtual Storage Constraint Relief. CICS/VSE Version 2 Release 2 (which comes with VSE/ESA Version 1 Release 3) is required for CICS Reduced Logging and DRDA Application Server Support.

  Note: The VMDSS feature requires an ESA/390 processor.

1.2.1.2 DASD Requirements

SQL/DS V3R5 will support all DASD devices supported by the operating systems through VSAM data management facilities (in VSE) or by VM DASD Block I/O Facilities (in VM). SQL/DS (VM) takes advantage of DFSMS/VM in the VM environment. The VM Data Spaces Support feature will support all current DASD devices supported by the VM/ESA Paging Subsystem.

1.2.1.3 Tape Requirements

SQL/DS will support, through VM or VSE, tape-handling devices supported by those operating systems.

1.2.1.4 Terminal Requirements

Terminal support for preplanned SQL/DS applications is provided though CMS or CICS. SQL/DS does not constrain the types of terminals used for user-written CMS or CICS programs. Any terminals, and any teleprocessing, networking and communications systems supported by those environments, can be used.

ISQL Terminal Support: ISQL will support 3270 compatible terminals.

DBS Utility Terminal Support: The DBS Utility supports a terminal interface through ICCF and CMS.

1.2.1.5 Printer Requirements

In a VM environment, results from ISQL, the interactive facility of SQL/DS, may be printed on any system printer supported by VM, or any remote printer through the RSCS Networking supported by the operating systems.

In a VSE environment, ISQL results may be printed on any system printer using VSE/POWER supported by the VSE/ESA operating systems. Any IBM 3270 Information Display System terminal printer supported by the BMS facility of CICS/VSE can print ISQL results.

1.2.2 Programs

1.2.2.1 VM

VM/ESA: SQL/DS V3R5 (VM) is supported on VM/ESA Version 1 Release 2.1 or later (ESA feature only).

RSCS: ISQL uses Remote Spooling Communications Subsystem (RSCS) Version 3 Release 1.1 or later for terminal printer support.

ACF/VTAM: DRDA flow to non-SQL/DS systems requires ACF/VTAM Version 3 Release 4, or later, in the gateway to external SNA connections. ACF/VTAM Version 3 Release 4 is required for VTAM boundary function and session level security. ACF/VTAM Version 3 Release 4 or later is required for DRDA-RUOW support on VM.

DFSMS/VM: If DFSMS is installed, SQL/DS (VM) uses DFSMS/VM Function Level 221 or later for the copy DBEXTENT function. (DFSMS/VM Function Level 221 is a standard feature shipped with VM/ESA Version 1 Release 2.1, but not a mandatory install feature.)

1.2.2.2 VSE

VSE/ESA: SQL/DS V3R5 (VSE) will run on VSE/ESA Version 1 Release 1 or later operating systems.

SQL/DS V3R5 (VSE) 31-bit support requires VSE/ESA Version 1 Release 3.

The DRDA-RUOW server function in the VSE environment requires:

- VSE/ESA Version 1 Release 3 Modification 1

  If VSE/ESA 1.3.0 is installed, a DRDA-enabling APAR is required.
- ACF/VTAM VSE/ESA Version 3 Release 4

  ACF/VTAM is required for the DRDA Application Server Support.
- CICS/VSE Version 2 Release 2

VSE Components: SQL/DS (VSE) will require the following base components of the operating system package:

- CICS/VSE V2R1 or later is required for SQL/DS (VSE) online support and for ISQL. CICS/VSE V2R2 or later is required for DRDA Application Server Support.
- VSE/VSAM Release 3 is required for database and log storage. Users may wish to use the VSE/VSAM Backup/Restore Feature to perform a User Archive and Restore of the database.
- VSE/POWER is required for SQL/DS (VSE) systems for the ISQL report writer facilities and for the ISQL Remote Power Printer support of SQL/DS (VSE).

  Note: Refer to the appropriate VSE operating system package announcement or planning manuals for more information on the version/release of the base component.

1.2.2.3 SQL/DS Optional Functions

VM DRDA-RUOW Optional Function: The following products are required:

- VM/ESA Version 1 Release 2.1 or later
- ACF/VTAM Version 3 Release 4 or later

VSE DRDA-RUOW Optional Function (Application Server only): The following products are required:

- VSE/ ESA Version 1 Release 3 or later

This option may require more than 16MBs of virtual storage to operate efficiently, which implies that 31-bit addressing in an ESA or VM/ESA supervisor mode may be required.

1.2.2.4 SQL/DS Features

VM RXSQL Version 3 Release 4: The following products are required:

- VM/ESA Version 1 Release 2.1 or later

DataHub Support/VM Version 3 Release 4: The following products are required for DataHub Support/VM:

- VM/ESA Version 1 Release 2.1 or later
- C runtime library

VM Data Spaces Support Version 3 Release 5: The following product is required for the SQL/DS VM Data Spaces Support feature:

- VM/ESA Version 1 Release 2.1 or later

SQL/DS Data Restore Version 3 Release 5: Either of the following products is required for the SQL/DS Data Restore feature:

- VM/ESA Version 1 Release 2.1 or later
- VSE/ESA Version 1 Release 1 or later

1.3 Advantages of Migrating to SQL/DS V3R5

SQL/DS V3R5 offers improved functionality and performance while maintaining compatibility with other DB2 family members.

It is important to remember that by migrating from a previous release, for example from SQL/DS V2R2 to SQL/DS V3R5, all the enhancements of the releases in between are obtained. All these benefits are listed in Appendix A, “SQL/DS Version 3 Enhancements Summary” on page 159.

Throughout this manual, the advantages of migrating to SQL/DS V3R5 are explained in detail. A summary of these benefits is provided here.

1.3.1 Greater Participation in Distributed Database Solutions

- Enable CICS applications to switch between SQL/DS application servers

  This will benefit users who have written online applications with CICS in a VSE environment which have a need to access more than one application server.

- Provide support for additional CCSID conversions

  This addresses the demand for support for conversion from CCSIDs supported on DB2/2 and DB2/6000.

- Enable DRDA Limited Accounting

  This will enable user accounting support similar to that which DB2 for MVS (V2R3 and later) and DDCS (V2R1) have implemented.

1.3.2 Higher Level of Compatibility with Other DB2 Products

- Change SQLSTATE values to match the SQL92 standard

  The SQLSTATEs used in previous releases of SQL/DS comply with the ISO/ANSI SQL2 standard. Changes have been made in SQL/DS V3R5 to support the more recent SQL standard, SQL92.

- Support even precision packed decimal variables in Assembler application programs

  These variables are supported by DB2 for MVS. This function will enable SQL/DS to recognize declared even precision host variables.

1.3.3 Maintain Compatibility with Other Products

- Support packed decimal variables in C/370 application programs

  This allows the previously unsupported SQL/DS DECIMAL data type to be used.

- Extend support for LE/VSE to single user mode applications

  With the introduction of LE/VSE, error condition handling now has to be coordinated between SQL/DS and LE/VSE. This coordination is provided by SQL/DS V3R5.

1.3.4 Performance Improvement

- Improve performance of the database by increasing the number of buffers that may be used

  This allows customers to fully exploit the available processor resources such as the large increase in real storage on System 390 processors, and the ability of the database to use 31-bit addressable virtual storage.

- Improve performance of the database manager archive and the database manager log archive

  This feature will allow customers to improve the performance of the archive process by exploiting multi-block *BLOCKIO and asynchronous I/O in VM, and VSAM controlled buffers in VSE.

1.3.5 Increased Availability

- Provide a more granular recover capability

  This feature provides comprehensive partial database recovery options, ensuring that a minimum number of users is affected for the least possible time, when it is necessary to restore parts of the database.

- Improve the ability to expand the directory

Enhanced Directory Expansion allows users to expand the directory allocation bitmap so that more physical pages may be used without having to regenerate the database.

1.4 Incompatibilities Between Releases

1.4.1 SQL/DS Database Archive Incompatibilities

Archives that were created on prior releases of SQL/DS cannot be restored by the SQL/DS V3R5 database manager. If this is attempted, the database manager will issue message ARI2038E and terminate. See the *SQL/DS Messages and Codes* manual for more details on this message.

1.4.2 SQL/DS VSAM Shareoptions Changes under VSE

In prior releases of SQL/DS (VSE), the VSAM SQL/DS directory, data and log data sets were defined with SHAREOPTIONS(1). In SQL/DS V3R5, these VSAM files **must** now be defined with SHAREOPTIONS(2).

1.4.3 SQLSTATE Values Changes

Many SQLSTATE values have changed in SQL/DS V3R5. The new SQLSTATE values and their former values can be found in the *SQL/DS Messages and Codes* manuals. Changing SQLSTATEs is an incompatible change since many SQLSTATE values that are returned from diagnostic situations will be different from previous releases of SQL/DS. Application programmers should review any programs that use SQLSTATE in the SQLCA each time an SQL statement is executed.

1.4.4 Messages and Codes Changes

Some SQL/DS messages and codes have changed, and some new ones have been added in SQL/DS V3R5. See the *SQL/DS Messages and Codes* manuals for details.

1.4.5 Display CICS Information on SHOW CONNECT

If the package that the connected user is running was created in SQL/DS Version 2 Release 2 or earlier, the CICS information will not be displayed by the SHOW CONNECT command because the RDIIN for V2R2 or earlier does not contain the RDIIN extension area. The package must be reprepped with SQL/DS V3R5 and recompiled to make the CICS information available.

Chapter 2. How to Use the New SQL/DS V3R5 Functions

This chapter describes how to use the functional enhancements and new capabilities available with SQL/DS V3R5. The intent is to provide the reader with a good understanding of these new functions and capabilities.

2.1 New Functions and Capabilities

2.1.1 CICS Database Switching

2.1.1.1 CICS Transactions

CIRB CICS transaction to start the online resource adapter.

CIRT CICS transaction to terminate the online resource adapter.

CIRD CICS transaction to display current CICS transactions in session with an SQL/DS application server.

CIRA New CICS transaction to add connections to another application server.

CIRR New CICS transaction to remove connections from an application server.

CIRC New CICS transaction to change the default application server.

ISQL Interactive SQL facility. In the VSE environment, ISQL is a CICS transaction.

2.1.1.2 Description

Prior to SQL/DS V3R5, users who wrote online applications with CICS/VSE could not use the database switching support that was available to other users. This is because of the way the online resource adapter was implemented. The connections to the SQL/DS application server were established when the online resource adapter was started and were maintained until the online resource adapter was stopped. Until SQL/DS V3R5, the only way to change to a different application server was to stop the currently running online resource adapter and start a new one to establish the connections to the new application server.

Operator intervention was required to stop and start the online resource adapter to switch to a different application server. The application could not just issue a connect to a new application server. All current transactions against that application server had to end, the operator issued CIRT to end the current online resource adapter and then issued CIRB to begin the new online resource adapter to the new application server.

Figure 1 on page 12 and Figure 2 on page 13 illustrate this.

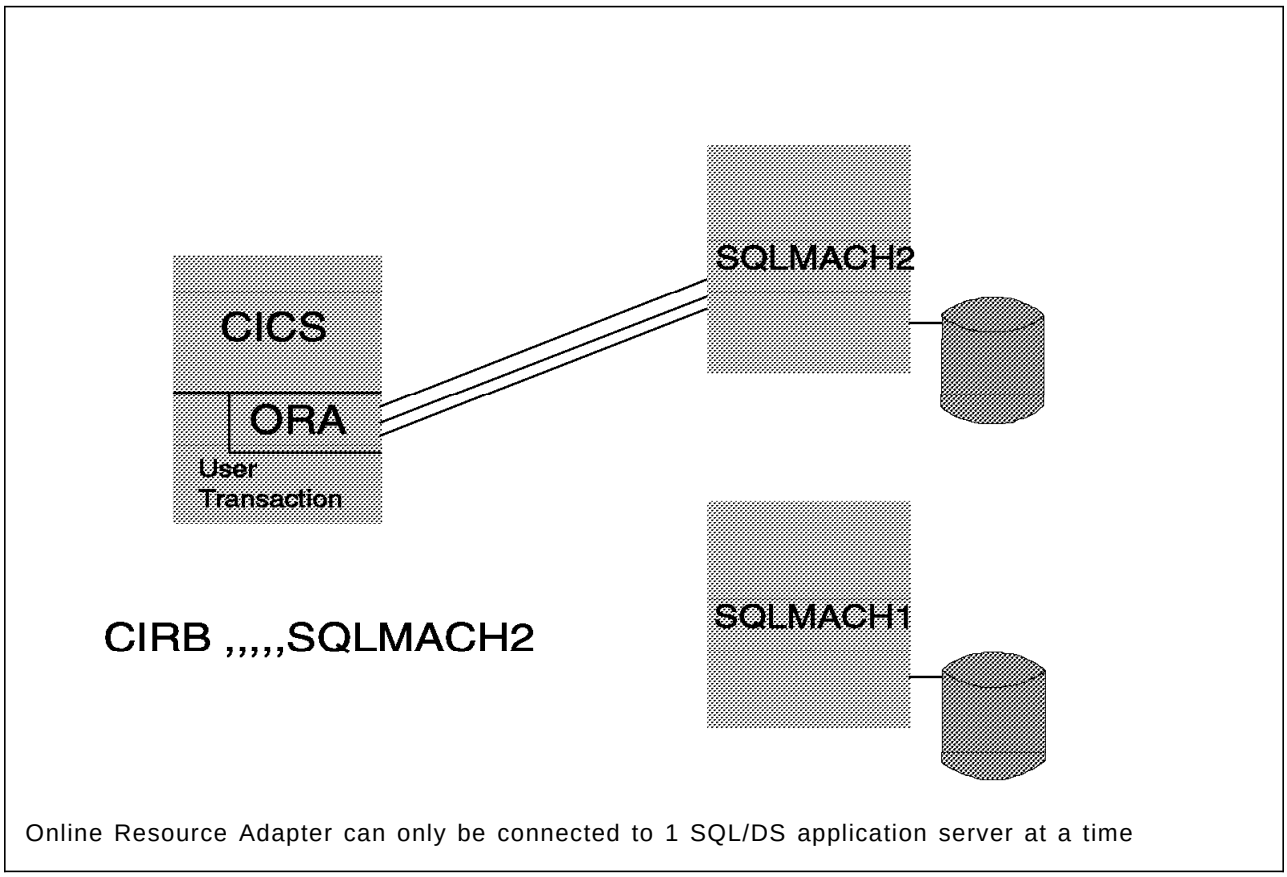


Figure 1. SQL/DS V3R4 Online Resource Adapter - Connection 1

In releases prior to SQL/DS V3R5, where CICS database switching was not available, CICS application developers were limited to the following:

- All transactions in a single CICS region must access the same application server. The only way for CICS online transactions to access multiple SQL/DS application servers was to have multiple CICS regions active, each with an online resource adapter connected to a different server. This tied up system resources in a very inefficient way.

  Since all CICS transactions were forced to access the same server, many installations were forced to have a single large application server instead of several smaller servers. For example, some CICS transactions may need to update inventory data while others update customer data. If these transactions were in the same CICS region, all the data must have been in the same server. With CICS database switching the inventory data and customer data can be split into separate application servers and these same transactions can still access this data. Note that smaller servers may help customers better manage their environment.

- Any single CICS transaction could access only one application server. It can be advantageous to be able to access different servers in different units of work. For example, one may have a CICS transaction that reads data from many separate SQL/DS servers, compiles the information and then updates another SQL/DS application server.

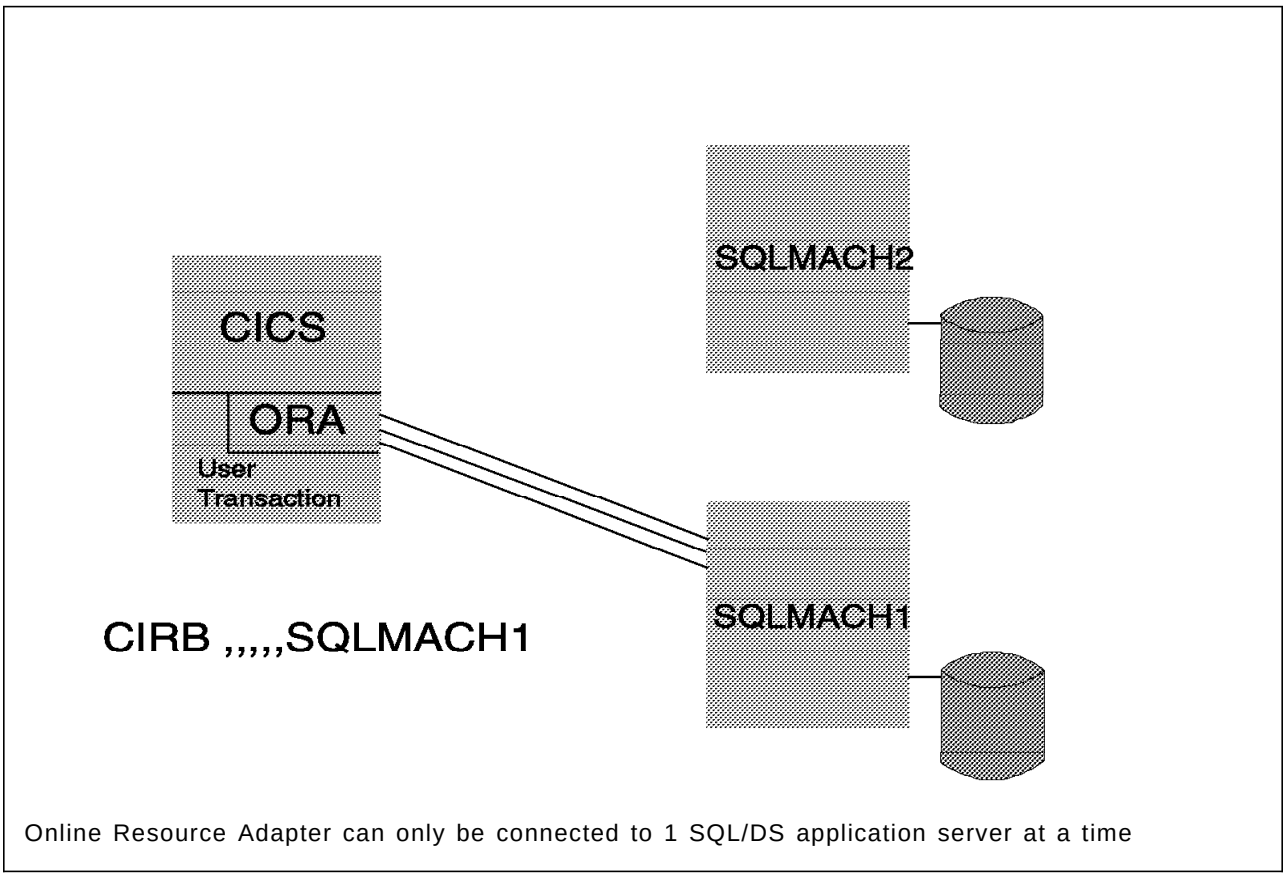


Figure 2. SQL/DS V3R4 Online Resource Adapter - Connection 2

Now, in SQL/DS V3R5, CICS database switching support eliminates these disadvantages. This support allows one resource adapter in one CICS region to connect to multiple application servers. Any CICS transaction in the CICS region can connect to any of the SQL/DS servers to which the online resource adapter has established connections. CICS database switching has the following advantages:

- Different transactions in a CICS region will be able to connect to different SQL/DS application servers.
- Single transactions will be able to connect to different SQL/DS application servers in different units of work.
- Database switching will be very efficient. The current online resource adapter design is very efficient in that connections to an SQL/DS server are established once (through the CIRB transaction) and these connections are reused until the online resource adapter is brought down (through the CIRT transaction).

  Performance of online transactions would be very poor if application server connections needed to be established dynamically every time they were needed. The design of the CICS database switching function takes this into account. In releases prior to SQL/DS V3R5, when an online resource adapter was started, starting another online resource adapter was not allowed. To implement database switching, the current online resource adapter design has been extended to allow connections to multiple application servers instead of impacting performance by establishing dynamic server connections all the time.

- Multiple CICS regions will not be required to connect CICS transactions to multiple SQL/DS application servers. This means that you won't tie up system resources inefficiently.

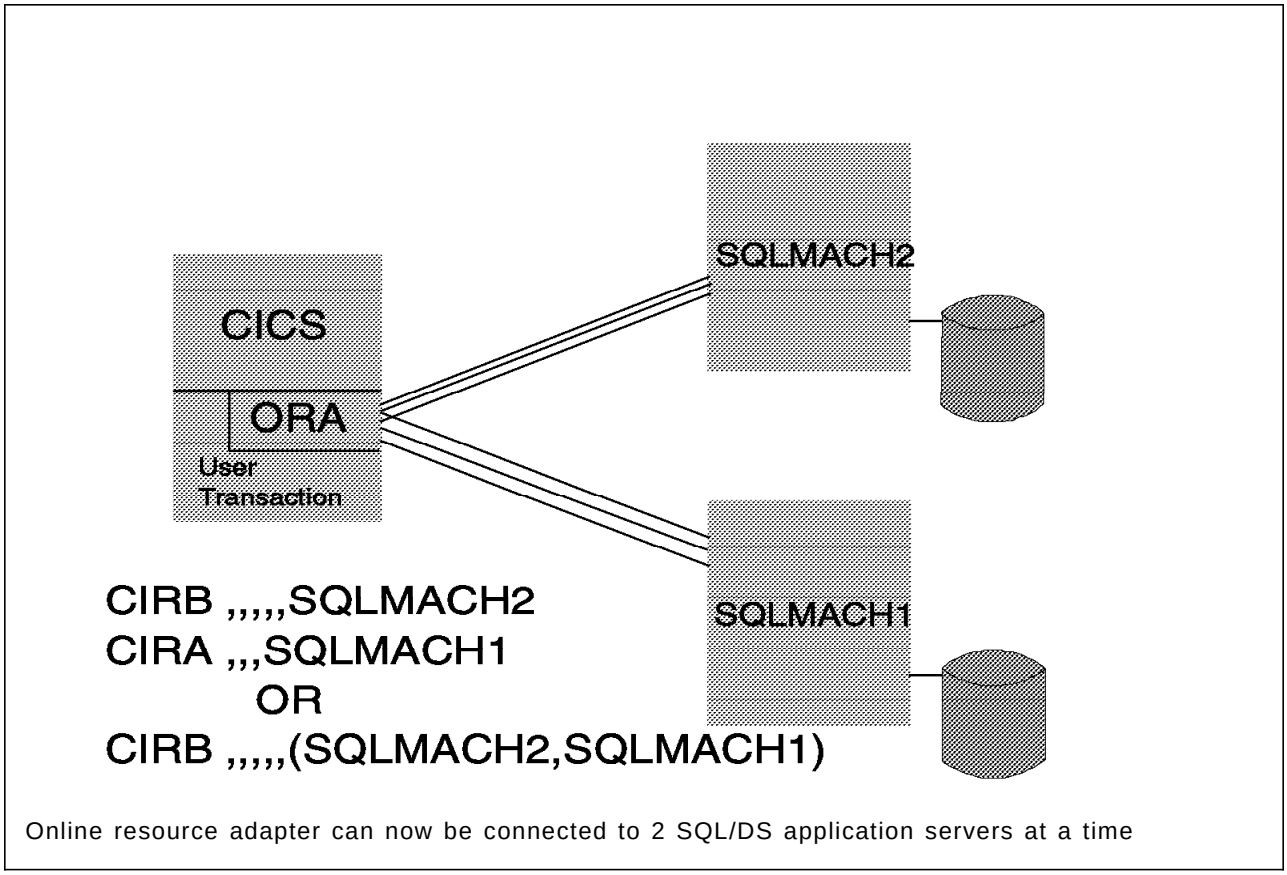


Figure 3. New SQL/DS V3R5 Online Resource Adapter - Two Connections

This is the way the online resource adapter works in SQL/DS V3R5 with CICS database switching. The CIRB transaction still starts the online resource adapter. The CIRB transaction will now accept a list of server-names. This allows multiple connections to be established from one command. After the online resource adapter is started, the new CICS transaction, CIRA, is used to add connections to other SQL/DS application servers. CIRA can be entered multiple times with different **server-names**. This establishes the connections to the specified application server. CIRA also accepts a list of server-names so that multiple connections can be established with one command. Figure 3 and Figure 4 on page 15 illustrate this.

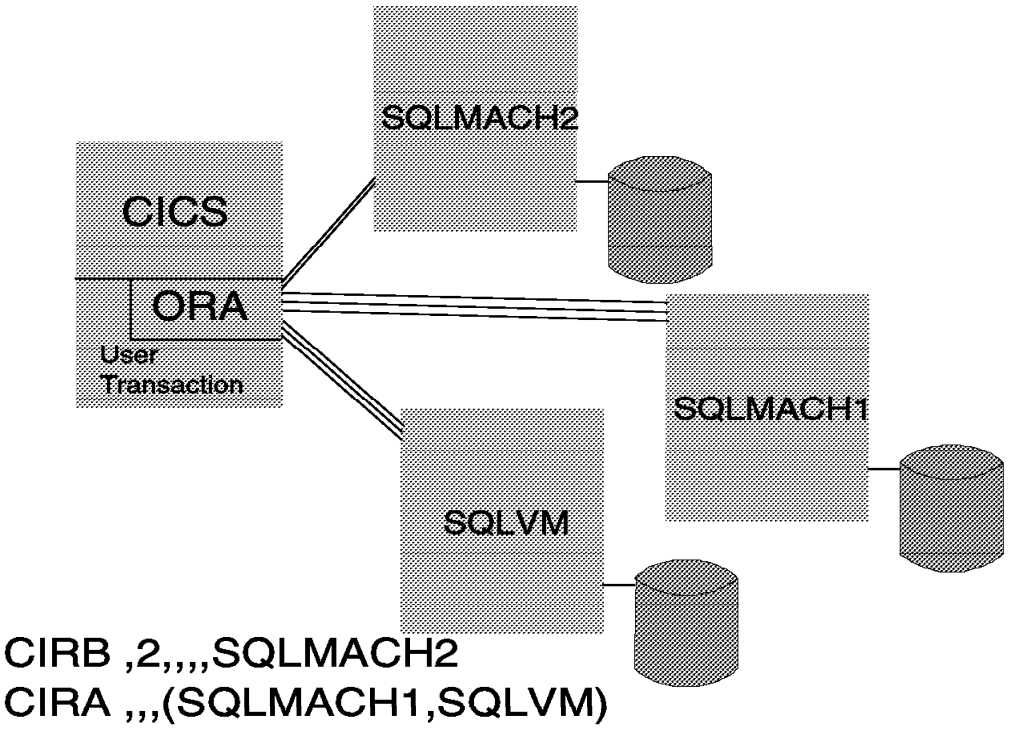


Figure 4. New SQL/DS V3R5 Online Resource Adapter - Three Connections

The new CICS transaction, CIRR, removes the connections to a particular application server or a list of application servers. The online resource adapter is terminated if the CIRR transaction removes the connections to the last application server.

The CIRT transaction can still be used to terminate the online resource adapter. CIRT terminates all connections to all application servers and then terminates the online resource adapter.

The CIRD transaction has been enhanced to accept a server-name parameter to display the transactions accessing a particular application server. The ***** keyword has been added to display all transactions on all of the application servers (that is, CIRD *).

If an application server becomes unavailable for some reason, only the connections to that application server are lost. The online resource adapter remains active and connections to other application servers can still be used. When the application server becomes available again, the CIRA transaction can be used to re-establish connections to it. If there were any in-doubt LUWs associated with this application server, they will be resolved at this time.

If the default application server becomes unavailable, a new default server is not established automatically. Users attempting to connect to the default server will receive a message indicating that the server is not available.

If the default application server must be changed, the new CICS transaction, CIRC, can be used or the online resource adapter can be terminated and restarted with a new default server.

2.1.1.3 Invocation

CICS database switching is invoked by using the SQL/DS CONNECT statement in a CICS transaction. A CICS transaction can connect to an SQL/DS application server, do some work then CONNECT to another application server and do some more work.

The rules for CICS database switching are the same as are allowed by database switching in the VSE batch environment and in the VM environment. The current unit of work must be completed by using the EXEC SQL COMMIT, EXEC SQL ROLLBACK or EXEC CICS SYNCPOINT commands, before the CONNECT statement can be used to switch to a different application server. These rules are described in the CONNECT statement description in the *SQL Reference* manual. Figure 5 below gives an example of how database switching might be coded.

The application programmer must keep in mind that the SQL/DS ROLLBACK and COMMIT commands automatically invoke the CICS SYNCPOINT command. This means that any other resources accessed in the transaction are committed or rolled back at this time also.

```
EXEC SQL CONNECT TO SQLMACH1;
EXEC SQL SELECT NAME INTO :TEMPNAME FROM INVENTORY WHERE
PARTNO=548216;
EXEC SQL ROLLBACK;
EXEC SQL CONNECT TO SQLMACH2;
EXEC SQL UPDATE INVENTORY SET PARTNAME=:TEMPNAME WHERE
PARTNO=548216;
EXEC SQL COMMIT;
```

Figure 5. Example of Database Switching

In releases prior to SQL/DS V3R5, CICS database switching was not possible. A CICS transaction could only access a single application server. An operator was required to stop the online resource adapter and start a new one if another application server needed to be used.

Unless the TO parameter is specified by a CICS/VSE application on a CONNECT statement, the CICS/VSE application will first establish connections to the default server. On subsequent CONNECTs performed by that CICS/VSE application, if the TO parameter is not specified then the connection to the previously connected server will be maintained. For a discussion on how to specify or change the default server, see 2.1.1.4, “CIRB” on page 17, and 2.1.1.8, “CIRC” on page 31.

The rules for establishing the authorization ID have not changed. Authorization IDs are explained in ″Establishing an Implicit Connect for CICS/VSE Transactions″ in the *Application Programming for VSE* manual.

Now, in SQL/DS V3R5, ISQL is also able to take advantage of database switching. ISQL has been enhanced to accept the ′CONNECT ... TO ...′ and the ′CONNECT ... IDENTIFIED BY ... TO ...′ formats of the SQL/DS CONNECT statement. The syntax and use of the CONNECT statement is fully described in the *SQL Reference* manual.

2.1.1.4 CIRB

The syntax for the CIRB transaction has been enhanced to allow a list of server-names to be specified. CIRB has six parameters:

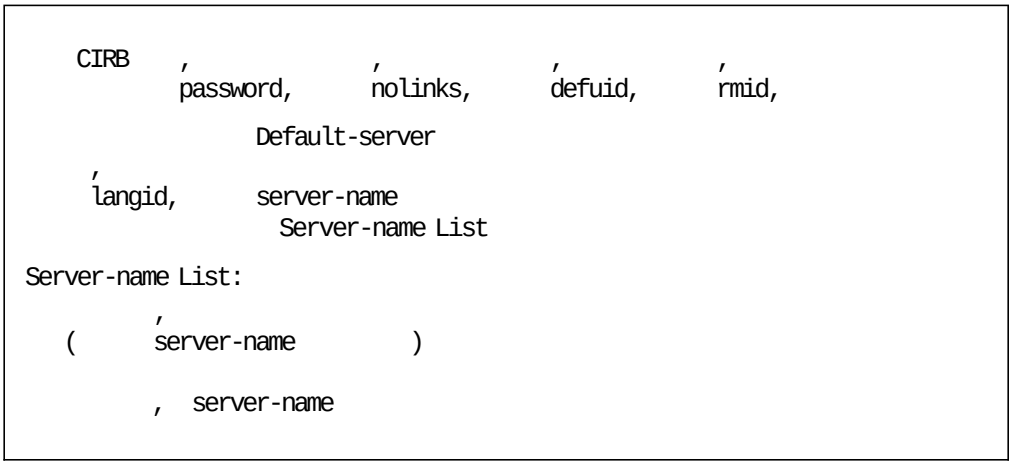


Figure 6. CIRB Transaction Syntax

The parameters are described in the following table:

Table 1 (Page 1 of 2). CIRB Transaction Parameters

Parameter	Default	Description

PASSWORD (positional parameter 1)	SQLDBAPW	This parameter establishes the operator's authority to activate online access to the SQL/DS server. The password identifies the CICS subsystem. The user ID of the subsystem is the CICS APPLID, which defaults to DBDCCICS. The procedure ARIS080D uses the following job control to give the password and user ID to the SQL/DS server:
		`// EXEC ARISQLDS,SIZE=AUTO,PARM=¢SYSMODE=S,`
		`LOGMODE=N,PROGNAME=ARIDBS¢`
		`CONNECT SQLDBA IDENTIFIED BY SQLDBAPW;`
		`GRANT SCHEDULE TO DBDCCICS IDENTIFIED BY CICSPSWD;`
		`COMMIT WORK;`
		The password chosen (CICSPSWD above) must satisfy SQL/DS specifications for a password. This password establishes which password to use when dropping connections through the CIRR or CIRT commands. See 2.1.1.10, "Password Implications on Online Resource Adapter Termination" on page 34 for more details.

NOLINKS (positional parameter 2)	3	This parameter establishes the number of links (paths) that should be initialized to the SQL/DS server. Specify this parameter as a decimal value between 1 and 64. The number must be less than or equal to the value assigned to the NCUSERS initialization parameter of the SQL/DS system. (The NCUSERS default is 5.)
DEFUID (positional parameter 3)	CICSUSER	This parameter identifies the default user ID used for the implicit CONNECT of the SQL/DS online support. This parameter must satisfy SQL/DS specifications for a user ID.

Table 1 (Page 2 of 2). CIRB Transaction Parameters

Parameter	Default	Description
RMID (positional parameter 4)	0	This parameter identifies a unique resource adapter. You must specify it only if your installation has multiple CICS partitions active in the same VSE/ESA system, and if each CICS partition allows online access to the SQL/DS server. For this case, recovery requires that the SQL/DS server know the resource adapter it is servicing. You must specify this parameter as a decimal value between 0 and 63. If the SQL/DS online support detects that this ID is not unique in the system, it issues a message. The CIRB transaction then ends without enabling the resource adapter. There can be only one SQL/DS resource adapter enabled in a single CICS partition. An attempt to enable a second SQL/DS resource adapter causes the SQL/DS online support to issue a message, and the CIRB transaction ends without enabling the second resource adapter. The first one, however, remains in effect.
LANGID (positional parameter 5)	specified at installation	This parameter defines the language the SQL/DS server uses to display error and information messages. The language you specify on this transaction becomes the default language for ISQL. The ISQL welcome logo always appears in the language specified on this transaction. This parameter must take the form of a minimum 1-character, maximum 5-character language ID. You must use one of the language IDs in the LANGID column of the SQLDBA.SYSLANGUAGE table. The language ID must identify a language you have installed on the SQL/DS server. To choose another language, use the SET LANGUAGE command in ISQL. The following IDs can be specified on the CIRB transaction: **AMENG** American English **UCENG** Uppercase English **FRANC** French **GER** German **ESPAN** Spanish **KANJI** Kanji (Japanese) **ITA** Italian **HANZI** Simplified Chinese If this parameter is omitted, the language defaults to the language chosen as the default at installation.
SERVER-NAME (positional parameter 6)	Determined from DBNAME directory or “SQLDS.”	This parameter enables you to specify the application servers that you want to access. If the list format is used to specify multiple servers, the first one in the list becomes the default server. Only the first server-name in the list may be omitted. If this parameter (or the first one in the list) is omitted, the default server is determined from the DBNAME directory. If the DBNAME directory does not specify a default server, then **SQLDS** becomes the default server name.

The behavior of the CIRB transaction was not changed. If you execute the CIRB command and there is already an online resource adapter active, you will receive the message *”ARI0401I SQL/DS Online Resource Adapter is already*

present in this partition." If connections to another application server need to be started, the CIRA command must be used.

The CIRB transaction establishes the default application server. If the server-name parameter is not specified on the CIRB transaction, then the default server is determined from the DBNAME directory. If a single server-name is specified on the CIRB transaction then it becomes the default server. If a server-name list is specified on the CIRB transaction, the first server-name in the list becomes the default server. If the first server-name in the server-name list is blank then the default server is determined in the same way as when the server-name is omitted from the CIRB transaction. For example:

```
CIRB ,,,,,(,SQLMACH2)
```

This starts connections to two servers. The first one is the default server and its name is determined from the DBNAME directory or if it is not specified in the DBNAME directory it defaults to **SQLDS**. The second server is SQLMACH2.

Note that the following examples are not allowed. Only the first server-name in the list can be blank.

```
CIRB ,,,,,(SQLMACH2,)
CIRB ,,,,,(SQLMACH2,,SQLVM)
```

The number of server-names that can be specified on the CIRB command is limited by the size of the input line on the VSE console or a CICS terminal. The VSE console only allows one line of input. A CICS terminal allows much more input. If short server-names are used more can fit on the command. Server-names can be up to 18 characters long. If all of the required server-names cannot fit on the command, the CIRA transaction must be used to establish connections for the remaining server-names.

Figure 7 shows an example of using the server-name list on the CIRB transaction.

```
msg f2
AR 015 1I40I READY
2 cirb ,,,,,(sqlmach1,sqlmach1)
F2-002 ARI0410I Resource Adapter ARI0OLRM is enabled.
F2-002 ARI0450I SQL/DS online support has an
                entry point of 003AA808 RMGL at 00541200.
F2-002 ARI0454I Connections to SQLMACH1 established.
                RMCV at 0055B2E0.
F2-002 ARI0458I The default server is SQLMACH1.
F2-002 ARI0457W Connections to SQLMACH1 already exist.
F2-002 ARI0402E Connections to SQLMACH1 could not be established.
```

Figure 7. Example of CIRB with Duplicate Server Names

The maximum number of application servers to which an online resource adapter can establish connections is only limited by the amount of storage available in the partition where the online resource adapter is running.

Detailed calculations to determine how much storage is required for the online resource adapter and for each connection to an application server can be found in 5.2.1.5, “Online Resource Adapter and Control Transactions” on page 111.

If you try to establish connections to an application server to which connections already exist, the new message *"ARI0457W Connections to <server-name> already exist."* is displayed. No action is taken against that server. If the connections to that server need to be changed they must first be removed using CIRR or CIRT and then re-established using CIRA or CIRB. An example is shown in Figure 8.

```
msg f2
AR 015 1I40I READY
2 cirb ,,,,,(sqlmach1,sqlmach2)
F2-002 ARI0410I Resource Adapter ARI0OLRM is enabled.
F2-002 ARI0450I SQL/DS online support has an
               entry point of 003AA808 RMGL at 00541200.
F2-002 ARI0454I Connections to SQLMACH1 established.
               RMCV at 0055B2E0.
F2-002 ARI0458I The default server is SQLMACH1.
F2-002 ARI0454I Connections to SQLMACH2 established.
               RMCV at 0055C2E0.
2 cirr ,,,sqlmach2
F2-002 ARI0455I Connections to SQLMACH2 are disabled.
2 cira ,5,,sqlmach2
F2-002 ARI0454I Connections to SQLMACH2 established.
               RMCV at 0055A2E0.
```

Figure 8. Example of Changing Connection Settings

Note that each server in the list has its connections established with the same values for password, number of links, RMID, default user ID and language ID that were specified.

If the CIRB parameters for each server are identical, all of the connections can be established with one CIRB transaction, as illustrated in Figure 9.

```
msg f2
AR 015 1I40I READY
2 cirb ,,,,,(sqlmach1,sqlmach2,sqlvm)
F2-002 ARI0410I Resource Adapter ARI0OLRM is enabled.
F2-002 ARI0450I SQL/DS online support has an
               entry point of 003AA808 RMGL at 00541200.
F2-002 ARI0454I Connections to SQLMACH1 established.
               RMCV at 0055A2E0.
F2-002 ARI0458I The default server is SQLMACH1.
F2-002 ARI0454I Connections to SQLMACH2 established.
               RMCV at 0055C2E0.
F2-002 ARI0454I Connections to SQLVM established.
               RMCV at 0055D2E0.
```

Figure 9. Example of CIRB with Server-Name List

All three application servers have the same number of connections, the same default user ID, the same password, the same RMID and the same language ID.

If one or more of the parameters must be different, then all of the connections cannot be established with one CIRB transaction. You will need the CIRA transaction to add additional servers.

2.1.1.5 CIRA

The new CIRA transaction has four parameters:

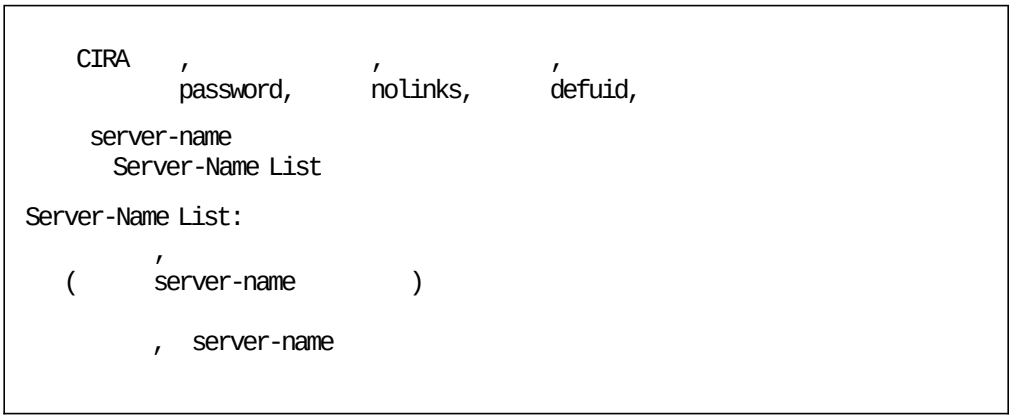


Figure 10. CIRA Transaction Syntax

The parameters are described in the following table:

Table 2. CIRA Transaction Parameters

Parameter	Default	Description

PASSWORD (positional parameter 1)	SQLDBAPW	This parameter establishes the operator's authority to activate online access to the SQL/DS server. The password identifies the CICS subsystem. The user ID of the subsystem is the CICS APPLID, which defaults to DBDCCICS. The procedure ARIS080D uses the following job control to give the password and user ID to the SQL/DS server:
		`// EXEC ARISQLDS,SIZE=AUTO,PARM=¢SYSMODE=S, LOGMODE=N,PROGNAME=ARIDBS¢`
		`CONNECT SQLDBA IDENTIFIED BY SQLDBAPW;`
		`GRANT SCHEDULE TO DBDCCICS IDENTIFIED BY CICSPSWD;`
		`COMMIT WORK;`
		The password chosen (CICSPSWD above) must satisfy SQL/DS specifications for a password. This password establishes which password to use when dropping connections through the CIRR or CIRT commands. See 2.1.1.10, "Password Implications on Online Resource Adapter Termination" on page 34 for more details.

NOLINKS (positional parameter 2)	3	This parameter establishes the number of links (paths) that should be initialized to the SQL/DS server. Specify this parameter as a decimal value between 1 and 64. The number must be less than or equal to the value assigned to the NCUSERS initialization parameter of the SQL/DS system. (The NCUSERS default is 5).
DEFUID (positional parameter 3)	CICSUSER	This parameter identifies the default user ID used for the implicit CONNECT of the SQL/DS online support. This parameter must satisfy SQL/DS specifications for a user ID.
SERVER-NAME (positional parameter 4)	none	This parameter is required and it specifies the additional application servers that you want to access. If this parameter is omitted, the message ARI0400E is issued indicating that an invalid input parameter was entered.

The password, nolinks, defuid and server-name parameters have exactly the same meanings as on the CIRB command. One exception is that the server-name parameter is required on CIRA but is optional on CIRB.

The number of server-names that can be specified on the CIRA command is limited by the size of the input line. As with CIRB, CIRA can be entered on the VSE console or on a CICS terminal. On the VSE console the input is limited to one line. On the CICS terminal it can use the full screen. If short server-names are used more can fit on the command. Server-names can be up to 18 characters long. If all of the required server-names cannot fit on the command, the CIRA transaction must be repeated for the remaining server-names. Figure 11 shows an example using the CIRA transaction with a server-name list.

```
msg f2
AR 015 1I40I READY
2 cirb ,,,,,sqlmach1
F2-002 ARI0410I Resource Adapter ARI0OLRM is enabled.
F2-002 ARI0450I SQL/DS online support has an
               entry point of 003AA808 RMGL at 00541200.
F2-002 ARI0454I Connections to SQLMACH1 established.
               RMCV at 0055A2E0.
F2-002 ARI0458I The default server is SQLMACH1.
2 cira ,,,(sqlmach2,sqlvm)
F2-002 ARI0454I Connections to SQLMACH2 established.
               RMCV at 0055C2E0.
F2-002 ARI0454I Connections to SQLVM established.
               RMCV at 0055D2E0.
```

Figure 11. Example of CIRA with Server-Name List

The maximum number of application servers to which an online resource adapter can establish connections is only limited by the amount of storage available in the partition where the online resource adapter is running. See 5.2.1.5, “Online Resource Adapter and Control Transactions” on page 111 for detailed calculations of the storage requirements.

The CIRA transaction establishes connections to the specified application servers based on the parameters given on the CIRA transaction. If a server-name list is used then connections will be established to each application server in the list using the same set of parameters. For example:

```
CIRA thispw,4,thisid,(sqlmach2,sqlvm)
```

The above command will establish four links to SQLMACH2 with password *“thispw”* and default user ID *“thisid.”* The RMID and the language ID are inherited from the CIRB transaction. If the online resource adapter was started with RMID = 0 and language ID = ameng then any connections started to that same online resource adapter will also have RMID = 0 and language id = ameng. Then CIRA will establish four links to SQLVM with password *“thispw”* and default user ID *“thisid.”* Again the RMID is 0 and the language ID is ameng. If CIRA is entered before CIRB was run, the message *“ARI0411I Resource Adapter is not enabled.”* is displayed.

If one or more of the parameters must be different, then the server-name list format of the CIRA transaction cannot be used. The CIRA transaction would have to be executed separately for each application server that required different parameters. For example, if three links are required to SQLMACH2 and four links are required to SQLVM but the other parameters are the same for both servers, the CIRA transaction must be run for each of them.

```
CIRA thispw,3,thisid,sqlmach2
CIRA thispw,4,thisid,sqlvm
```

If you try to establish connections to an application server that is already connected the new message *"ARI0457W Connections to <server-name> already exist."* is displayed. No action is taken against that server. If the connections to that server need to be changed they must first be removed using CIRR or CIRT and then re-established using CIRA or CIRB.

Consider the following scenario. An online transaction program needs to access three different application servers, SQLMACH2, SQLMACH1 and SQLVM. SQLMACH2 and SQLMACH1 are running in two VSE partitions and SQLVM is running under VM and is accessed via guest sharing. We want SQLMACH1 to be the default server, and we want the default settings for all three servers.

To achieve this we could enter the following sequence of commands. Assume that our CICS region is running in partition 2, SQLMACH2 is running in partition 4 and SQLMACH1 is running in partition 5.

1. Use the CIRB transaction to start the online resource adapter and establish the default application server, SQLMACH1.
2. Use the CIRA transaction to establish connections to SQLMACH2.
3. Use the CIRA transaction again to establish connections to SQLVM.

This is illustrated in Figure 12.

```
F2-002 DFH1500 - DBDCCICS : CONTROL IS BEING GIVEN TO CICS
msg f2
AR 015 1I40I READY
2 cirb ,,,,,sqlmach1
F2-002 ARI0410I Resource Adapter ARI0OLRM is enabled.
F2-002 ARI0450I SQL/DS online support has an
                entry point of 003AA808 RMGL at 00541200.
F2-002 ARI0454I Connections to SQLMACH1 established.
                RMCV at 0055D2E0.
F2-002 ARI0458I The default server is SQLMACH1.
2 cira ,,,sqlmach2
F2-002 ARI0454I Connections to SQLMACH2 established.
                RMCV at 0055C2E0.
2 cira ,,,sqlvm
F2-002 ARI0454I Connections to SQLVM established.
                RMCV at 0055A2E0.
```

Figure 12. Example of CIRB and CIRA

Since the settings for the connections to SQLMACH2 and SQLVM are identical, both connections could be established on the same CIRA command, as illustrated in Figure 11 on page 22.

2.1.1.6 Automatic Restart Resynchronization

If a system or subsystem failure occurs while an online application is trying to commit work and two-phase commit is being used, the unit being committed is called an in-doubt logical unit of work, because the database manager has prepared it for commit or rollback but the system or subsystem failure occurred before the commit completed. In-doubt units of work must be resolved the next time the application server is started.

Note: With CICS/VSE Version 2.2 and later, SQL/DS will use a one-phase commit if at most one external resource has been updated. In this case it is not possible to create an in-doubt unit of work. This means that any CICS transaction

running in CICS/VSE Version 2.2 that updates only SQL/DS resources will not generate in-doubt units of work.

The CICS/VSE restart resynchronization facility, which is started implicitly when you issue CIRB or CIRA, resolves the in-doubt units of work. To enable it, you must update the CICS/VSE tables to include the resynchronization transaction.

CIRB and CIRA assume that restart resynchronization is enabled when they are executed. If, for some reason it has been disabled when CIRB or CIRA is issued, it will display the message *"ARI0466E CICS restart re-synchronization is not available. The <tran> transaction is ended."* and exit. At this point the system programmer should ensure that it has been properly enabled and retry CIRB or CIRA.

For information about the updates, see the *SQL/DS Installation for VSE* manual.

The current implementation of the CICS/VSE restart resynchronization facility allows it to re-synchronize itself with SQL/DS only once. After it has been invoked, CICS discards any information about in-doubt units of work that it did not resolve. This means that there can be scenarios where it is not possible to automatically resolve in-doubt units of work.

When the CIRB or CIRA transaction is started, a connection is made to the READY/RECOVERY agent of the server to get a 'recovery list'. This recovery list provides information on any in-doubt agents that need to be resolved for this server. After this has been done for every server specified in the CIRB or CIRA command, the CICS/VSE restart resynchronization facility is invoked, which will resolve the in-doubt units of work for all of those servers. A subsequent CIRA to connect to another server that also has in-doubt units of work will fail because CICS has discarded the log information. The in-doubt units of work on that server must be resolved manually using the FORCE n COMMIT or FORCE n ROLLBACK commands on the server before the CIRA command will work.

For example, suppose that SQLMACH1 and SQLMACH2 are SQL/DS application servers that run on the same VM system and are accessed via guest sharing. The password used to access SQLMACH1 is ABC and the password used to access SQLMACH2 is DEF. All the other parameters needed by the two databases are the defaults. The connections to SQLMACH1 and SQLMACH2 are established using the following sequence of commands:

```
CIRB abc,,,,,sqlmach1

CIRA def,,,sqlmach2
```

Suppose that CICS transactions accessing these application servers also make updates to the SQL/DS database as well as some other external non-CICS resource, so that CICS will use the two-phase commit process. If a system failure occurs on the VM system while CICS is performing a two-phase commit to both these databases, then both SQLMACH1 and SQLMACH2 will go down. When the system is brought back up and SQLMACH1 and SQLMACH2 are restarted, they will both have in-doubt units of work. If the connections to SQLMACH1 and SQLMACH2 are restarted the same way as before, only the in-doubt units of work on SQLMACH1 will be resolved automatically. The in-doubt units of work on SQLMACH2 will need to be resolved explicitly before the CIRA command for SQLMACH2 will work.

See Figure 13 on page 25 for an example of this.

```
2 cirb abc,,,,,sqlmach1
F2 002 ARI0410I Resource Adapter ARI0OLRM is enabled.
F2 002 ARI0450I SQL/DS online support has an
               entry point of 0039F008 RMGL at 001DF5B4.
F2 002 ARI0454I Connections to SQLMACH1 established.
               RMCV at 0053BF00.
F2-002 ARI0458I The default server is SQLMACH1.
2 cira def,,,sqlmach2
F2-002 ARI0454I Connections to SQLMACH2 established.
               RMCV at 0055A080.

<System Failure occurs>

F2 002 ARI2908I XPCCB, IJBXRUSR = 0483061009000000
F2 002 ARI0406E Error in using system communications facility.
               Request = 15
               Return Code = 4 Reason Code = 7
F2 002 The default server is SQLMACH1.
F2 002 ---------------------------------------------
F2 002 DBDCCICS connected to server SQLMACH1.
F2 002 Status of online SQL/DS applications:
F2 002
F2 002 Transactions holding a link to SQL/DS but not using are:
F2 002
F2 002  TASKNO  TRANID TERMID USER ID   USERDATA TIME SINCE  TOTAL LUW
F2 002                                          LAST ACCESS TIME
F2 002  ______  ______ ______ ________  ________ __________ _________
F2 002  0000041  CISQ          SQLDBA     L080    00:00:06    00:01:34
F2 002
F2 002   TIME= 15:26:15 DATE= 08/14/95
F2 002 ARI0465I Transactions are still active
               for server SQLMACH1.
F2 002 ARI0463I The DISABLE transaction CIRR must delay for a
               30-second interval before attempting the disable.
F2 002 ARI0455I Connections to SQLMACH1 are disabled.
F2 002 ARI0460W Connections to the default server SQLMACH1
               have been disabled.
F2 002 ARI2908I XPCCB, IJBXRUSR = 0483061009000000
F2 002 ARI0406E Error in using system communications facility.
               Request = 15
               Return Code = 4 Reason Code = 7
F2 002 The default server is SQLMACH1.
F2 002 ---------------------------------------------
F2 002 DBDCCICS connected to server SQLMACH2.
F2 002 Status of online SQL/DS applications:
F2 002
F2 002 Transactions holding a link to SQL/DS but not using are:
F2 002
F2 002  TASKNO  TRANID TERMID USER ID   USERDATA TIME SINCE  TOTAL LUW
F2 002                                          LAST ACCESS TIME
F2 002  ______  ______ ______ ________  ________ __________ _________
F2 002  0000141  CISQ          SQLDBA     L083    00:00:06    00:01:34
F2 002
```

Figure 13 (Part 1 of 2). Automatic Restart Resynchronization Failure

```
F2 002  TIME= 15:26:45 DATE= 08/14/95
F2 002 ARI0465I Transactions are still active
                for server SQLMACH2.
F2 002 ARI0463I The DISABLE transaction CIRR must delay for a
                30-second interval before attempting the disable.
F2 002 ARI0455I Connections to SQLMACH2 are disabled.
F2-002 ARI0413I Resource Adapter ARI0OLRM is disabled.

<SQLMACH1 and SQLMACH2 are restarted>

2 cirb abc,,,,,sqlmach1
F2 002 ARI0410I Resource Adapter ARI0OLRM is enabled.
F2 002 ARI0450I SQL/DS online support has an
                entry point of 0039F008 RMGL at 001DF5B4.
F2 002 ARI0454I Connections to SQLMACH1 established.
                RMCV at 0053BF00.
F2-002 ARI0458I The default server is SQLMACH1.
2 cira def,,,sqlmach2
F2 002 ARI0454I Connections to SQLMACH2 established.
                RMCV at 0055A080.
F2-002
F2 002 ARI0438E Automatic restart resynchronization failed.
                A logical unit of work that SQL/DS indicated
                needed to be resolved was not identified by
                the CICS/VSE log as needing resolution.
F2 002 ARI0423A Use the SQL/DS SHOW and FORCE commands to
                COMMIT or ROLLBACK the following units of work:
F2 002 ARI0424I User ID = SQLDBA Agent Identifier = 1
                Server = SQLMACH2
F2 002 The default server is SQLMACH1.
F2 002 ---------------------------------------------
F2 002 DBDCCICS connected to server SQLMACH2.
F2 002 There are no active SQL/DS transactions.
F2 002
F2 002  TIME= 15:33:22 DATE= 08/14/95
F2 002 ARI0455I Connections to SQLMACH2 are disabled.

<From the SQLMACH2 console enter:>
<SHOW ACTIVE>
<FORCE 1 ROLLBACK>

<Now CIRA will work>

2 cira def,,,sqlmach2
F2 002 ARI0454I Connections to SQLMACH2 established.
                RMCV at 0055A080.
```

Figure 13 (Part 2 of 2). Automatic Restart Resynchronization Failure

However if the connections to SQLMACH1 and SQLMACH2 are established with a single CIRB or CIRA command, the in-doubt units of work on **both** servers will be resolved automatically.

See Figure 14 on page 27 for a detailed example of this.

```
2 cirb abc,,,,,(sqlmach1,sqlmach2)
F2 002 ARI0410I Resource Adapter ARI0OLRM is enabled.
F2 002 ARI0450I SQL/DS online support has an
                entry point of 0039F008 RMGL at 001DF5B4.
F2 002 ARI0454I Connections to SQLMACH1 established.
                RMCV at 0053BF00.
F2-002 ARI0458I The default server is SQLMACH1.
F2-002 ARI0454I Connections to SQLMACH2 established.
                RMCV at 0055A080.

<System Failure occurs>

F2 002 ARI2908I XPCCB, IJBXRUSR = 0483061009000000
F2 002 ARI0406E Error in using system communications facility.
                Request = 15
                Return Code = 4 Reason Code = 7
F2 002 The default server is SQLMACH1.
F2 002 ---------------------------------------------
F2 002 DBDCCICS connected to server SQLMACH1.
F2 002 Status of online SQL/DS applications:
F2 002
F2 002 Transactions holding a link to SQL/DS but not using are:
F2 002
F2 002  TASKNO  TRANID TERMID USER ID   USERDATA TIME SINCE  TOTAL LUW
F2 002                                           LAST ACCESS TIME
F2 002  ______  ______ ______ ________  ________ __________ _________
F2 002  0000041  CISQ          SQLDBA     L080    00:00:06    00:01:34
F2 002
F2 002  TIME= 15:26:15 DATE= 08/14/95
F2 002 ARI0465I Transactions are still active
                for server SQLMACH1.
F2 002 ARI0463I The DISABLE transaction CIRR must delay for a
                30-second interval before attempting the disable.
F2 002 ARI0455I Connections to SQLMACH1 are disabled.
F2 002 ARI0460W Connections to the default server SQLMACH1
                have been disabled.
F2 002 ARI2908I XPCCB, IJBXRUSR = 0483061009000000
F2 002 ARI0406E Error in using system communications facility.
                Request = 15
                Return Code = 4 Reason Code = 7
F2 002 The default server is SQLMACH1.
F2 002 ---------------------------------------------
F2 002 DBDCCICS connected to server SQLMACH2.
F2 002 Status of online SQL/DS applications:
F2 002
```

Figure 14 (Part 1 of 2). Successful Automatic Restart Resynchronization

```
F2 002 Transactions holding a link to SQL/DS but not using are:
F2 002
F2 002  TASKNO  TRANID TERMID USER ID   USERDATA TIME SINCE  TOTAL LUW
F2 002                                          LAST ACCESS TIME
F2 002  ______  ______ ______ ________  ________ __________ _________
F2 002  0000141  CISQ          SQLDBA     L083    00:00:06    00:01:34
F2 002
F2 002  TIME= 15:26:45 DATE= 08/14/95
F2 002 ARI0465I Transactions are still active
                for server SQLMACH2.
F2 002 ARI0463I The DISABLE transaction CIRR must delay for a
                30-second interval before attempting the disable.
F2 002 ARI0455I Connections to SQLMACH2 are disabled.
F2-002 ARI0413I Resource Adapter ARI0OLRM is disabled.

<SQLMACH1 and SQLMACH2 are restarted>

2 cirb abc,,,,,(sqlmach1,sqlmach2)
F2 002 ARI0410I Resource Adapter ARI0OLRM is enabled.
F2 002 ARI0450I SQL/DS online support has an
                entry point of 0039F008 RMGL at 001DF5B4.
F2 002 ARI0454I Connections to SQLMACH1 established.
                RMCV at 0053BF00.
F2-002 ARI0458I The default server is SQLMACH1.
F2 002 ARI0454I Connections to SQLMACH2 established.
                RMCV at 0055A080.
```

Figure 14 (Part 2 of 2). Successful Automatic Restart Resynchronization

Assuming CICS restart resynchronization has been properly enabled as described in the *SQL/DS Installation for VSE* manual, the conditions where in-doubt units of work must be resolved explicitly are:

1. CICS log missing. This can be from a CICS log media failure, CICS COLD start which destroys the log contents, or CICS journal is not active so no log data is created.
2. CICS RESYNCH has already been issued. The log data is discarded by CICS after the RESYNCH command has been issued even if it was not used. See Figure 13 on page 25 for an example of this.

To take full advantage of the automatic restart resynchronization the following should be true:

1. All servers with in-doubt units of work must be started on the same CIRB or CIRA transaction. This means they must have the same password, default user ID, language, RMID, and number of links to be started.
2. CICS startup should be START=AUTO which lets CICS determine if the startup will be START=WARM or START=EMER. Any COLD start will erase the log data and automatic restart resynchronization will not be possible.

2.1.1.7 CIRR

To remove connections to an application server, issue the CICS CIRR transaction. The CIRR transaction has four parameters:

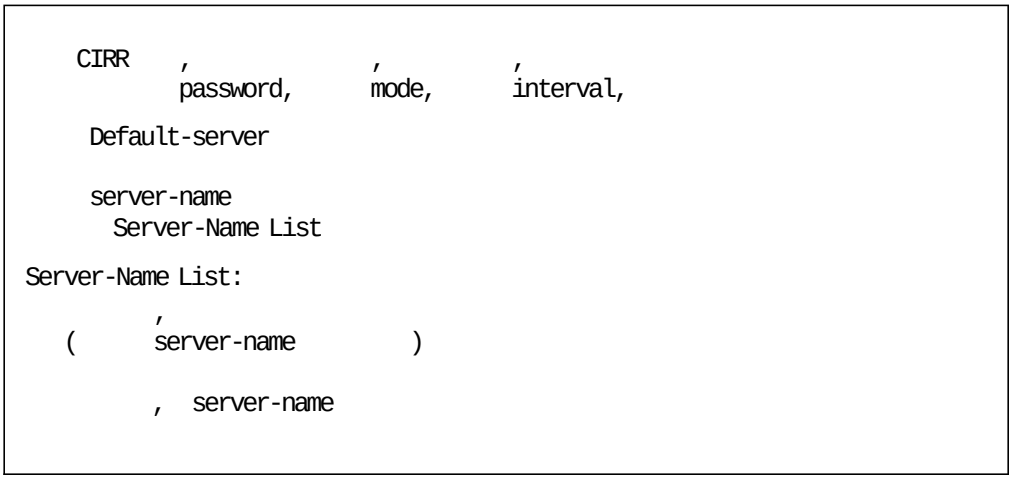


Figure 15. CIRR Transaction Syntax

The password, mode and interval parameters are the same as on the CIRT transaction and are described in the following table:

Table 3 (Page 1 of 2). CIRR Transaction Parameters

Parameter	Default	Description
PASSWORD (positional parameter 1)	SQLDBAPW	This password establishes the operator's authority to terminate the online access to the SQL/DS server. It must be the same password that was supplied for the server by the CIRB or CIRA transaction. Refer to 2.1.1.10, "Password Implications on Online Resource Adapter Termination" on page 34 for more details.
MODE (positional parameter 2)	NORMAL	This parameter establishes the shutdown mode: NORMAL or QUICK. When you specify NORMAL, the CIRR transaction prevents new online users from accessing the specified SQL/DS system. Users who are already doing work, however, can finish. When all SQL/DS users complete their work, no online users can use the specified SQL/DS server. When you specify QUICK, online access is ended immediately. Online SQL/DS users cannot finish their work. Their current logical units of work are rolled back (unless they are already processing a COMMIT WORK). You can change from NORMAL to QUICK. However, once the MODE is QUICK, you cannot change it back to NORMAL.

Table 3 (Page 2 of 2). CIRR Transaction Parameters

Parameter	Default	Description
INTERVAL (positional parameter 3)	30 (seconds)	The number of seconds that the CIRR transaction should delay before freeing the terminal. The value must be an integer value between 0 and 3600. This parameter controls the availability of the CICS terminal (or operator console) once you issue the CIRR transaction. The CICS terminal (or VSE operator console) used to activate the CIRR transaction is unavailable until the transaction ends. This could be a long time if the online application is long-running or if a user left without correctly ending the terminal session. If you issue CIRR PASSWORD,NORMAL,, server-name the terminal is not available until all online SQL/DS users complete their work. The value you specify for interval represents an interval of time measured in seconds. If the CIRR transaction does not finish immediately, it waits the amount of time you specify. When this time ends, the CIRR transaction tries once again to finish processing. If the CIRR transaction does not finish successfully, you receive a message telling you to retry the CIRR transaction later. After issuing the message, the CIRR transaction ends. The shutdown mode is still in effect (the specified SQL/DS server is in the process of shutting down), and the terminal is available for your use.
SERVER-NAME (positional parameter 4)	Determined by CIRB or CIRC transaction.	This parameter enables you to specify the application servers from which you want to remove access. The default server is removed if this parameter is omitted, or if the first parameter in the server-name list is blank. The default server is the one that was established by the CIRB transaction or by the CIRC transaction.

If no server-name is specified the default server-name is used. The default server-name was established by the CIRB or CIRC transaction. The CIRD transaction may be used to display the default server-name in case the user does not know what the default server-name is.

```
msg f2
AR 015 1I40I READY
2 cirr
F2-002 ARI0455I Connections to SQLMACH1 are disabled.
F2-002 ARI0460W Connections to the default server SQLMACH1 have
                    been disabled.
```

Figure 16. Example of CIRR with Defaults

The above example assumes that there are connections to more than one server when the CIRR transaction is entered.

If the password, mode and interval are the same then the server-name list can be used to remove connections from multiple application servers. Since SQLVM was the last active connection, the online resource adapter was terminated.

```
msg f2
AR 015 1I40I READY
2 cirr ,,,(sqlmach2,sqlvm)
F2-002 ARI0455I Connections to SQLMACH2 are disabled.
F2-002 ARI0455I Connections to SQLVM are disabled.
F2-002 ARI0413I Resource Adapter ARI0OLRM is disabled.
```

Figure 17. Example of CIRR with Server-Name List

The CIRR transaction can be used to remove the connections to the application server that were established by the CIRB and CIRA transactions. If CIRR removes the last active connections to the online resource adapter then the online resource adapter is terminated. The CIRB transaction would have to be used to restart it.

The CIRA and CIRR transactions can be entered repeatedly and in any order to add and remove links to application servers as required.

If CIRR is entered to remove connections to a server to which no connections have been established, the message *"ARI0456I Connections to <server-name> do not exist."* is displayed.

If the password given on the CIRR transaction does not match the password that was used to start the connections to the named server, then the connections to that server are not shut down and processing continues with the next server in the list.

2.1.1.8 CIRC

The new transaction, CIRC, can be used to dynamically change the default server. The CIRC transaction has one parameter:

```
CIRC  server_name
```

Figure 18. CIRC Transaction Syntax

The parameter is described in the following table.

Table 4. CIRC Transaction Parameter

Parameter	Default	Description
SERVER-NAME (positional parameter 1)	none	This parameter is required and it specifies the application server that you want to become the default. If this parameter is omitted, the message ARI0400E is issued indicating that an invalid input parameter was entered.

The server-name specified must already have connections established to it, either from the CIRB or CIRA transactions. If connections to the specified server do not exist the message *"ARI0456I Connections to <server-name> do not exist."* is displayed. In this case the CIRA transaction must first be run to establish the connections, then the CIRC transaction is run to make it the default server.

For the following example assume that connections exist to SQLMACH1 and SQLMACH2 and that SQLMACH2 is the current default server.

```
msg f2
AR 015 1I40I READY
2 circ sqlmach1
F2-002 ARI0459I The new default server is SQLMACH1.
               The previous default server was SQLMACH2.
```

Figure 19. Example of CIRC

For this next example assume that connections exist to SQLMACH1 but not to SQLMACH2.

```
msg f2
AR 015 1I40I READY
2 circ sqlmach2
F2-002 ARI0456I Connections to SQLMACH2 do not exist.
2 cira ,,,sqlmach2
F2-002 ARI0454I Connections to SQLMACH2 established.
               RMCV at 0055D2E0.
2 circ sqlmach2
F2-002 ARI0459I The new default server is SQLMACH2. The previous
                    default server was SQLMACH1.
```

Figure 20. Example of CIRC

It is important to note that if the connections to the default server are lost, that server is still identified as the default server. The connections can be lost because the server went down or because the CIRR transaction was used to terminate the connection. Users that are trying to connect to the default server in these cases will receive SQLCODE = -940. If the CIRB or CIRA transaction is used to establish connections to a server that is not ready, the message *"ARI0418A SQL/DS server <server-name> is not ready. Retry the enable transaction <tran> after SQL/DS starts."* is displayed. If there is no active online resource adapter the CIRB transaction must be used. If there is an active online resource adapter the CIRA transaction must be used.

2.1.1.9 CIRT

To end SQL/DS online support, issue the CICS CIRT transaction. The syntax of the CIRT transaction has not changed, and is as follows:

```
CIRT    ,           ,        ,
        password,   mode,    interval
```

Figure 21. CIRT Transaction Syntax

Table 5. CIRT Transaction Parameters

Parameter	Default	Description
PASSWORD (positional parameter 1)	SQLDBAPW	This password establishes the operator's authority to terminate the online access to the SQL/DS system. It must be the same password that was supplied for the CIRA or CIRB transaction. Refer to 2.1.1.10, "Password Implications on Online Resource Adapter Termination" on page 34 for more details.
MODE (positional parameter 2)	NORMAL	This parameter establishes the shutdown mode: NORMAL or QUICK. When you specify NORMAL, the CIRT transaction prevents new online users from accessing the SQL/DS system. Users who are already doing work, however, can finish. When all SQL/DS users complete their work, no online users can use the SQL/DS system. When you specify QUICK, online access is ended immediately. Online SQL/DS users cannot finish their work. Their current logical units of work are rolled back (unless they are already processing a COMMIT WORK). You can change from NORMAL to QUICK. However, once the MODE is QUICK, you cannot change it back to NORMAL.
INTERVAL (positional parameter 3)	30 (seconds)	The number of seconds that the CIRT transaction should delay before freeing the terminal. The value must be an integer value between 0 and 3600. This parameter controls the availability of the CICS terminal (or operator console) once you issue the CIRT transaction. The CICS terminal (or VSE operator console) used to activate the CIRT transaction is unavailable until the transaction ends. This could be a long time if the online application is long-running or if a user left without correctly ending the terminal session. If you issue CIRT PASSWORD,NORMAL the terminal is not available until all online SQL/DS users complete their work. Even with CIRT PASSWORD, QUICK there may be some delay before the CICS terminal allows the CIRT terminal to complete its cleanup process. The value you specify here represents an interval of time measured in seconds. If the CIRT transaction does not finish immediately, it waits the amount of time you specify. When this time ends, the CIRT transaction tries once again to finish processing. If the CIRT transaction does not finish successfully, you receive a message telling you to retry the CIRT transaction later. After issuing the message, the CIRT transaction ends. The shutdown mode is still in effect (the specified SQL/DS system is in the process of shutting down), and the terminal is available for your use.

If links to multiple application servers exist, they will all be removed. Once all of the links have been removed, the online resource adapter is terminated.

```
msg f2
AR 015 1I40I READY
2 cirt
F2-002 ARI0455I Connections to SQLVM are disabled.
F2-002 ARI0455I Connections to SQLMACH2 are disabled.
F2-002 ARI0455I Connections to SQLMACH1 are disabled.
F2-002 ARI0413I Resource Adapter ARI0OLRM is disabled.
```

Figure 22. Example of CIRT with Connections to Three Applications Servers

Note that the message *"ARI0413I Resource Adapter ARI0OLRM is disabled"* is not displayed until the last application server connections have been severed.

When the online resource adapter is not active, the CIRA and CIRR transactions are invalid. The online resource adapter needs to be enabled with the CIRB transaction before the CIRA and CIRR transactions can be used.

```
F2-002 ARI0413I Resource Adapter ARI0OLRM is disabled.
2 cira ,,,sqlmach1
F2-002 ARI0411I Resource Adapter is not enabled.
2 cirr ,,,sqlmach1
F2-002 ARI0411I Resource Adapter is not enabled.
```

Figure 23. Example of CIRA and CIRR after CIRT

2.1.1.10 Password Implications on Online Resource Adapter Termination

The password used on the CIRR and CIRT transactions must be the same one that was used on the CIRA and/or the CIRB transactions. CIRR and CIRT will only shut down the connections to servers where the password matches. If the passwords do not match, that server is not shut down.

Consider the following example:

1. The online resource adapter is started with the command:

   CIRB pw1,5,,,,(SQLMACH1,SQLMACH2)

2. Connections to two new servers are added with the command:

   CIRA ,,,(SQLMACH3,SQLMACH4)

3. Another connection is added to a fifth server with the command:

   CIRA pw2,1,,SQLMACH5

It is not possible to end the online resource adapter with one command in this scenario. The CIRT or CIRR transactions must be run at least three times before the online resource adapter is completely shutdown because three different passwords were used to start it up.

The CIRT transaction issued with no parameters would only shut down the connections to SQLMACH3 and SQLMACH4 because they were the only servers that were started with the default password.

To shut down SQLMACH5, you would have to enter the following command:

CIRT pw2

To bring down the remaining servers and stop the online resource adapter you need to enter:

CIRT pw1 followed by CIRT

The CIRR transaction can also be used, but the server names must be specified. The following shows the CIRR commands that would be equivalent to the CIRT commands in this scenario.

CIRT pw1 is equivalent to CIRR pw1,,,(SQLMACH1,SQLMACH2)

CIRT is equivalent to CIRR ,,,(SQLMACH3,SQLMACH4)

CIRT pw2 is equivalent to CIRR pw2,,,SQLMACH5

If the command:

CIRR ,,,(SQLMACH1,SQLMACH2,SQLMACH3,SQLMACH4,SQLMACH5)

were entered only SQLMACH3 and SQLMACH4 would be disconnected.

Message ARI0464E will be issued for servers SQLMACH1, SQLMACH2 and SQLMACH5 since the passwords don't match.

Similarly, if the command:

CIRR pw1,,,(SQLMACH1,SQLMACH2,SQLMACH3,SQLMACH4,SQLMACH5)

were entered only SQLMACH1 and SQLMACH2 would be disconnected.

Message ARI0464E will be issued for servers SQLMACH3, SQLMACH4 and SQLMACH5 since the passwords don't match.

2.1.1.11 CIRD

To display status information about active CICS transactions that access a SQL/DS server, issue the CICS CIRD transaction. The CIRD transaction has one new parameter:

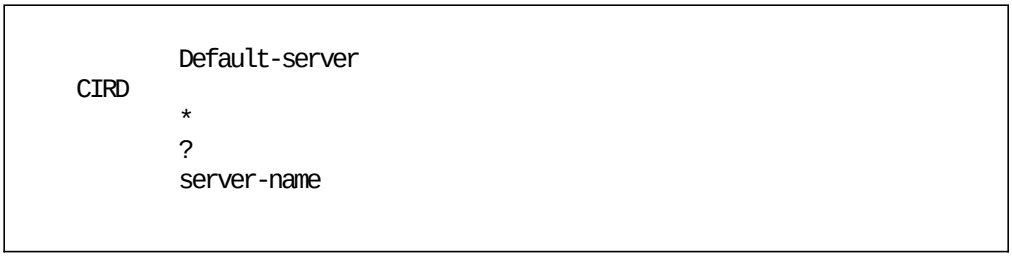


Figure 24. CIRD Transaction Syntax

The parameter is described in the following table:

Table 6. CIRD Transaction Parameters

Parameter	Default	Description
SERVER-NAME (positional parameter 1)	Determined by CIRB or CIRC transaction.	This parameter enables you to specify the application server whose status is to be displayed, or * to display the status of all servers and the details of transactions accessing the servers, or ? to display a list of the connected servers without the transaction details. If this parameter is omitted, the default server-name is the one that was determined by the CIRB or the CIRC transaction.

The new parameter is the server-name. If this parameter is omitted, the default server is the one that was determined by the CIRB or the CIRC transaction. If *

is specified then the normal CIRD display is repeated for each application server. If a specific server-name is used then data is only displayed for that application server. If ? is specified, then a list of connected servers without the transaction details is displayed.

For the following examples, assume that SQLMACH1 is the default server and that connections have been established for SQLMACH1, SQLMACH2 and SQLVM.

Figure 25 shows an example of the information displayed by the CIRD transaction with no parameters.

```
2 cird
F2 002 The default server is SQLMACH1.
F2 002 -------------------------------------------------
F2 002 DBDCCICS connected to server SQLMACH1.
F2 002 Status of online SQL/DS applications:
F2 002
F2 002 Transactions waiting to establish a link to SQL/DS are:
F2 002
F2 002  TASKNO TRANID TERMID USER ID  USERDATA WAIT TIME
F2 002  ------ ------ ------ -------- -------- ---------
F2 002  000033 MKE2                   L222     00:01:32
F2 002  000025 INV    L224   JIM               00:08:32
F2 002
F2 002 Transactions holding a link and now accessing SQL/DS are:
F2 002
F2 002  TASKNO TRANID TERMID USER ID  USERDATA TIME USED    TOTAL LUW
F2 002                                         FOR CURRENT  TIME
F2 002                                         ACCESS
F2 002  ------ ------ ------ -------- -------- ------------ ---------
F2 002  000019 CISQ          DEPT222  L199     00:01:32     00:03:48
F2 002  000037 INV    L209   TERRY             00:00:01     00:00:03
F2 002
F2 002 Transactions holding a link to SQL/DS but not using are:
F2 002
F2 002  TASKNO TRANID TERMID USER ID  USERDATA TIME SINCE   TOTAL LUW
F2 002                                         LAST ACCESS  TIME
F2 002  ------ ------ ------ -------- -------- ------------ ---------
F2 002  000003 CISQ          WILLIAM  L210     00:07:01     00:10:56
F2 002
F2 002 Transactions which previously accessed SQL/DS (not holding link):
F2 002
F2 002  TASKNO TRANID TERMID USER ID  USERDATA TIME SINCE
F2 002                                         LAST ACCESS
F2 002  ------ ------ ------ -------- -------- ------------
F2 002  000003 MKE2          ROBERT   L210     00:20:04
F2 002
F2 002  TIME=14:28:23 DATE=09/01/95
```

Figure 25. Example of CIRD with Defaults

Figure 26 on page 37 shows an example of the information displayed by the CIRD transaction with a server-name specified.

```
2 cird sqlmach2
F2 002 The default server is SQLMACH1.
F2 002 -------------------------------------------------
F2 002 DBDCCICS connected to server SQLMACH2.
F2 002 Status of online SQL/DS applications:
F2 002
F2 002 Transactions waiting to establish a link to SQL/DS are:
F2 002
F2 002  TASKNO TRANID TERMID USER ID  USERDATA WAIT TIME
F2 002  ------ ------ ------ -------- -------- ---------
F2 002  000033 MKE2                   L222     00:01:32
F2 002  000025 INV    L224   JIM              00:08:32
F2 002
F2 002 Transactions holding a link and now accessing SQL/DS are:
F2 002
F2 002  TASKNO TRANID TERMID USER ID  USERDATA TIME USED    TOTAL LUW
F2 002                                         FOR CURRENT  TIME
F2 002                                         ACCESS
F2 002  ------ ------ ------ -------- -------- ------------ ---------
F2 002  000019 CISQ          DEPT222  L199     00:01:32     00:03:48
F2 002  000037 INV    L209   TERRY             00:00:01     00:00:03
F2 002
F2 002 Transactions holding a link to SQL/DS but not using are:
F2 002
F2 002  TASKNO TRANID TERMID USER ID  USERDATA TIME SINCE   TOTAL LUW
F2 002                                         LAST ACCESS  TIME
F2 002  ------ ------ ------ -------- -------- ------------ ---------
F2 002  000003 CISQ          WILLIAM  L210     00:07:01     00:10:56
F2 002
F2 002 Transactions which previously accessed SQL/DS (not holding link):
F2 002
F2 002  TASKNO TRANID TERMID USER ID  USERDATA TIME SINCE
F2 002                                         LAST ACCESS
F2 002  ------ ------ ------ -------- -------- ------------
F2 002  000003 MKE2          ROBERT   L210     00:20:04
F2 002
F2 002  TIME=14:28:23 DATE=09/03/95
```

Figure 26. Example of CIRD with Server-Name

Figure 27 on page 38 shows an example of the information displayed by the CIRD transaction with the * specified.

```
2 cird *
F2 002 The default server is SQLMACH1.
F2 002 There are connections to server SQLMACH1.
F2 002 There are connections to server SQLMACH2.
F2 002 There are connections to server SQLVM.
F2 002 -------------------------------------------------
F2 002 DBDCCICS connected to server SQLMACH1.
F2 002 Status of online SQL/DS applications:
F2 002
F2 002 Transactions waiting to establish a link to SQL/DS are:
F2 002
F2 002  TASKNO TRANID TERMID USER ID  USERDATA WAIT TIME
F2 002  ------ ------ ------ -------- -------- ---------
F2 002  000033 MKE2                   L222     00:01:32
F2 002  000025 INV    L224   JIM               00:08:32
F2 002
F2 002 Transactions holding a link and now accessing SQL/DS are:
F2 002
F2 002  TASKNO TRANID TERMID USER ID  USERDATA TIME USED    TOTAL LUW
F2 002                                         FOR CURRENT  TIME
F2 002                                         ACCESS
F2 002  ------ ------ ------ -------- -------- ------------ ---------
F2 002  000019 CISQ          DEPT222  L199     00:01:32     00:03:48
F2 002  000137 INV    L209   BOB               00:17:34     01:24:03
F2 002
F2 002 Transactions holding a link to SQL/DS but not using are:
F2 002
F2 002  TASKNO TRANID TERMID USER ID  USERDATA TIME SINCE   TOTAL LUW
F2 002                                         LAST ACCESS  TIME
F2 002  ------ ------ ------ -------- -------- ------------ ---------
F2 002  000013 CISQ          LARRY    L210     00:03:01     00:11:36
F2 002
F2 002 Transactions which previously accessed SQL/DS (not holding link):
F2 002
F2 002  TASKNO TRANID TERMID USER ID  USERDATA TIME SINCE
F2 002                                         LAST ACCESS
F2 002  ------ ------ ------ -------- -------- ------------
F2 002  000003 MKE2          LOUISA   L210     01:57:04
F2 002
F2 002  TIME=14:28:23 DATE=09/03/95
F2 002 -------------------------------------------------
F2 002 DBDCCICS connected to server SQLMACH2.
F2 002 There are no active SQL/DS transactions.
F2 002
F2-002  TIME= 14:29:47 DATE= 09/03/95
F2 002 -------------------------------------------------
F2 002 DBDCCICS connected to server SQLVM.
F2 002 There are no active SQL/DS transactions.
F2 002
F2 002  TIME=14:30:23 DATE=09/03/95
```

*Figure 27. Example of CIRD with **

Figure 28 on page 39 shows an example of the information displayed by the CIRD transaction with the ? specified.

```
2 cird ?
F2 002 The default server is SQLMACH1.
F2 002 There are connections to server SQLMACH1.
F2 002 There are connections to server SQLMACH2.
F2 002 There are connections to server SQLVM.
F2 002 ------------------------------------------------
```

Figure 28. Example of CIRD with ?

Some extra information can be derived from the displays. In Figure 28 notice that SQLMACH1 is mentioned as the default server and on the next message that there are connections to SQLMACH1 also. It is possible, with the CIRR transaction, to remove the connections to SQLMACH1. The CIRD command would still show that the default server is SQLMACH1 but the message indicating there are connections to SQLMACH1 would not be displayed. In this scenario, users connecting to the default server would receive SQLCODE = -940 on the CONNECT statement. The CIRA transaction could be used to establish connections to SQLMACH1 again or the CIRC transaction could be used to change the default server to one of the other active servers. Either method allows CONNECT statements to access the default server.

If CIRR or CIRT has been issued to disconnect a server or to shut down the online resource adapter but cannot complete because there are still active transactions against the server, the CIRD transaction will show which transactions and which servers are affected.

Figure 29 on page 40 shows an example of the information displayed by the CIRD transaction with the ? parameter specified. The attempt to remove the connections to SQLMACH2 fails because there are still active transactions. Then the CIRD transaction is used to determine which transactions are still active. The user is found and asked to complete his work. When the CIRR command is retried it completes successfully and the connections to SQLMACH2 are shut down.

```
2 cird ?
F2 002 The default server is SQLMACH1.
F2 002 There are connections to server SQLMACH1.
F2 002 There are connections to server SQLMACH2.
F2 002 There are connections to server SQLVM.
F2 002 -----------------------------------------------
2 cirr ,,1,sqlmach2
F2 002 ARI0463I The DISABLE transaction CIRR must delay for a
               1-second interval before attempting the disable.
F2-002
2 cird ?
F2 002 The default server is SQLMACH1.
F2 002 There are connections to server SQLMACH1.
F2 002 Connections to SQLMACH2 are being disabled.
F2 002 There are connections to server SQLVM.
F2 002 -----------------------------------------------
F2-002
2 cird *
F2 002 The default server is SQLMACH1.
F2 002 There are connections to server SQLMACH1.
F2 002 Connections to SQLMACH2 are being disabled.
F2 002 There are connections to server SQLVM.
F2 002 -----------------------------------------------
F2 002 DBDCCICS connected to server SQLMACH1.
F2 002 There are no active SQL/DS transactions.
F2 002
F2 002  TIME= 19:07:43 DATE= 09/20/95
F2-002
F2 002 -----------------------------------------------
F2 002 DBDCCICS connected to server SQLMACH2.
F2 002 Status of online SQL/DS applications:
F2 002
F2 002 Transactions holding a link to SQL/DS but not using are:
F2 002
F2 002  TASKNO  TRANID TERMID USER ID  USERDATA TIME SINCE  TOTAL LUW
F2 002                                        LAST ACCESS TIME
F2 002  ______  ______ ______ ________ ________ __________ _________
F2 002  0000129 CISQ          CICSUSER L77D     00:00:31    00:00:31
F2 002
F2 002  TIME= 19:07:44 DATE= 09/20/95
F2 002 -----------------------------------------------
F2 002 DBDCCICS connected to server SQLVM.
F2 002 There are no active SQL/DS transactions.
F2 002
F2 002  TIME= 19:07:45 DATE= 09/20/95
F2-002
2 cirr ,,2,sqlmach2
F2-002 ARI0455I Connections to SQLMACH2 are disabled.
```

Figure 29. Example of CIRD in a Disable Scenario

2.1.1.12 ISQL

The ISQL welcome screen has been enhanced to display the default application server established by the CIRB transaction or subsequent CIRC transaction and it allows the target application server to be changed. Figure 30 on page 41 shows an example of the initial ISQL screen.

```
      Welcome to the interactive SQL facility of SQL/Data System


              IIIIIIII  SSSSSSSS  QQQQQQQQQ   LL
                 II     SS       QQ     QQ   LL
                 II     SSSSSSSS QQ     QQ   LL
                 II           SS QQ  QQ QQ   LL
              IIIIIIII  SSSSSSSS  QQQQQQQQQ  LLLLLLLL
                                        QQ

           Default Target Database is SQLDS

       Enter User ID, Password and Target Database, then press Enter

           User ID ==========> ________
           Password ==========> ________
           Target Database ==> ____________________


   To exit now, enter EXIT in user ID field with no password, press Enter.
     To exit later, use the EXIT command. Use the HELP command for help.
```

Figure 30. Initial ISQL Screen

The alternate method for invoking ISQL without having to use the ISQL signon display has also been enhanced to include the server-name. Figure 31 shows the new format.

```
    ISQL  #r-id#    USERID/password
                    USERID/password/server-name


     routine-name
                    (parameter-list)
```

Figure 31. ISQL Command

Where:

r-id Is a 1-4 character CICS transaction identifier or one to four blanks.

Stands for a hexadecimal byte X'FF' positioned immediately before and after r-id to mark the beginning and end of r-id.

USERID/password{/server-name} Is the ISQL signon user ID and password, and optionally the server-name. The user ID and password must be separated by the slash (/). If server-name is specified, you must precede it with the slash (/), and the user ID/password will be used to connect to the specified server. If no server-name is specified, the user ID/password will be used to connect to the default server.

routine-name Is optional. Refer to "Using the ISQL Transaction Identifier" in the *Interactive SQL User's Guide and Reference* manual.

(parameter-list) Is optional. Refer to “Using the ISQL Transaction Identifier” in the *Interactive SQL User's Guide and Reference* manual.

2.1.2 DRDA Limited Accounting String

2.1.2.1 Description

The SQL/DS accounting facility records how resources are consumed on the database manager. Resources are consumed both by individual users, and by processes that cannot be attributed to a single user, such as startup, shutdown, checkpoints, and archives. This information is collected in fixed-length records, 80 bytes long, that describe who or what consumed resources.

The records include up to 16 bytes for installation-dependent data, where one may supply information such as account numbers or project numbers. In releases prior to SQL/DS V3R5, these 16 bytes could only be used when an SQL/DS (VM) application requester used the SQLDS protocol to communicate with an SQL/DS application server (that is, when PROTOCOL=SQLDS or PROTOCOL=AUTO was specified on the SQLINIT EXEC).

There was, however, the opportunity for DRDA requesters to send accounting information to DRDA servers in a general purpose unarchitected DRDA parameter, namely the PRDDTA parameter of the ACCRDB DDM command. DB2 for MVS (Version 2 Release 3 and up) and DDCS (Version 2 Release 1) have implemented this approach for sending accounting data. Now, in SQL/DS V3R5, this function enables similar support for SQL/DS servers and SQL/DS (VM) requesters. Specifically, it enables the following scenarios:

- Allows SQL/DS (VM) requesters to send up to 16 bytes of installation-dependent data to any SQL/DS server (both VM and VSE) using the DRDA protocol, which are then recorded into USER accounting records.
- Allows SQL/DS (VM) requesters to send accounting information to DRDA servers, from which accounting records may be generated. DB2 for MVS is an example of such a server.
- Allows DRDA requesters to send accounting data to SQL/DS servers (both VM and VSE), from which 16 bytes of **user supplied data** are recorded into SQL/DS USER accounting records. Examples of such requesters are DB2 for MVS, DDCS for OS/2 (version 2), DDCS for AIX (version 2) and SQL/DS.

2.1.2.2 Invocation

Setup and use of the SQL/DS accounting exit in the DRDA environment is identical to existing accounting exit support using SQLDS flows. For VM, this is described in the manual *System Administration for IBM VM Systems Version 3 Release 4* in Chapter 11 “Using the Accounting Facility” and Chapter 14 “Creating Installation Exits” (in subsection “Supplying Account Numbers for Users”). For VSE servers, please consult chapter 10 “Using the Accounting Facility” in the manual *System Administration for VSE Version 3 Release 4*.

2.1.2.3 Interactions

Interaction between the user and the SQL/DS accounting exit in the DRDA environment is identical to existing interaction between the user and the SQL/DS accounting exit using SQLDS flows. For VM, this is described in the manual *System Administration for IBM VM Systems Version 3 Release 4* in Chapter 11 “Using the Accounting Facility” and Chapter 14 “Creating Installation Exits” (in subsection “Supplying Account Numbers for Users”). For VSE servers, consult

chapter 10 “Using the Accounting Facility” in the manual *System Administration for VSE Version 3 Release 4*.

Prior to SQL/DS V3R5, accounting data **did** flow from SQL/DS applications to SQL/DS servers when **PROTOCOL=AUTO** was specified on the SQLINIT EXEC. This occurs because, in this case, after handshaking is done the conversation switches from the DRDA to the SQLDS protocol. With the implementation of DRDA Limited Accounting in SQL/DS V3R5, accounting data flows during DRDA handshaking. This means that accounting string data is subject to the following restrictions:

1. The accounting string data is converted to CCSID 500 before being sent to the DRDA server. To ensure that all characters in the string data can be represented in CCSID 500 and in the CCSID used on the server, it is recommended that only the characters **A**-**Z**, **0**-**9** and ′_′ (underscore) be used. If characters other than these recommended ones are used, then those characters may not translate properly when the DRDA server writes out accounting records.
2. The user-specified portion of the accounting string can be at most 16 bytes. This is true for SQL/DS (VM) applications sending accounting data (which is set up in the ARIUXIT user exit) and for non-SQL/DS DRDA requesters sending accounting data to SQL/DS servers.

2.1.2.4 Interfaces

If SQL/DS considers there is accounting data, then SQL/DS will extract 16 bytes from **accounting_data** as described in the following sections and put them into SQL/DS user accounting records.

DDCS to SQL/DS Interface: When applications use DDCS (Version 2) to connect to SQL/DS servers, it is assumed that the PRDDTA parameter of the ACCRDB DDM command has a structure with the format shown in the following table:

Table 7 (Page 1 of 2). DDCS Accounting Layout

Field Name	Length	Description
acct_str_len	1	This one byte length field is the length of a DDCS generated prefix (consisting of fields client_prdid, client_platform, client_appl_name, client_authid and suffix_len) and a user supplied suffix.
client_prdid	8	This is the product ID of the client′s Client Application Enabler (CAE) software. For example, the product ID for CAE/2, CAE/6000 and CAE/DOS is “SQL01011.” This is **not** the product ID for the DDCS gateway itself. The product ID for DDCS can be found in the PRDID parameter of the ACCRDB DDM command.
client_platform	18	The platform the client is on; for example, “AIX,” “OS/2.”
client_appl_name	20	This is the first 20 characters of the user′s application name.
client_authid	8	This is the authid of the user′s application; for example, “PAYROLL.”

Table 7 (Page 2 of 2). DDCS Accounting Layout

Field Name	Length	Description
suffix_len	1	This one byte length field is the length of the user supplied suffix. A value of zero means that no user suffix was supplied.
user_suffix	n	This “n” byte field contains the user suffix. This is the user-supplied accounting data. **Note** that only the first 16 bytes are put into SQL/DS User Accounting records; this field, however, may be greater than 16 bytes in length.
rest_prddta_len	0 or 1	This one byte field contains the length of the **rest** of the product specific data field. This is to accommodate non-accounting product specific data. If there is no such data, then the length of this field is zero. Note that this field is only used by SQL/DS to verify that PRDDTA has accounting data. It is not used for any other purpose.
rest_prddta	0 or n	This field contains the rest of the product specific data and exists to accommodate non-accounting product specific data. If there is no such data, then the length of this field is zero. Otherwise the length of this field is indicated in the field **rest_prddta_len**. Note that this field is ignored by SQL/DS.

SQL/DS V3R5 servers consider that there is accounting data during DRDA handshaking if the PRDDTA parameter was passed on the ACCRDB DDM command and one of the following is true:

- **acct_str_len** plus one equals the length of the PRDDTA parameter as indicated on the ACCRDB DDM command.
- **acct_str_len** plus **rest_prddta_len** plus two equals the length of the PRDDTA parameter as indicated on the ACCRDB DDM command.

If the following conditions are all true:

1. the conditions described in the previous paragraph have been met
2. the product name of the **AR** is “SQL”
3. the product version number of the **AR** is at least “02”
4. **suffix_len** is greater than zero

then, the SQL/DS server inserts into USER records any **user_suffix** data up to 16 characters. If **user_suffix** is greater than 16 characters, then only the first 16 are inserted into the USER record. If **user_suffix** is less than 16 characters then it is padded with blanks. If **suffix_len** is equal to zero, then blanks are inserted into the USER record.

The **user_suffix** is one of the following:

- The value specified by an application with the sqlesact() API
- The value of the DB2ACCOUNT environment variable
- The value of the DFT_ACCOUNT_STR (default accounting string) configuration parameter

- A null string

See the DDCS documentation for more details.

DB2 for MVS to SQL/DS Interface: When DB2 for MVS applications connect to SQL/DS servers, it is assumed that the PRDDTA parameter of the ACCRDB DDM command has a structure with the format shown in the following table:

Table 8 (Page 1 of 2). DB2 for MVS Accounting Layout

Field Name	Length	Description
acct_str_len	1	This one byte length field is the length of the accounting data supplied by DB2 for MVS less 1. This length does not include itself, hence the minus one.
prdid	8	This is the product ID for DB2 for MVS, which includes a three character product name, a two character version, a two character release and a one character modification level. The product ID can be “DSN02030,” “DSN03010” or “DSN04010.”
location_name	16	The DB2 location name for the DB2 for MVS system that created the accounting data.
netid	8	The SNA Netid for the DB2 for MVS system that created the accounting data.
lu_name	8	The SNA LU name for the DB2 for MVS system that created the accounting data.
connection_name	8	The DB2 connection name at the DB2 for MVS system where the application is running.
connection_type	8	The DB2 connection type at the DB2 for MVS system where the SQL application is running. This can be values such as “BATCH,” “DB2CALL,” “REMOTE,” “IMSBMP,” “IMSMPP,” “CICS,” or “DLIBATCH.”
correlation_id	12	The DB2 correlation ID at the DB2 for MVS system where the SQL application is running. For a complete description of the correlation ID field, please consult the description of message DSNV404I in Section 3 of the *DB2 for MVS Messages and Codes (SC26-4892)* manual.
authid	8	The DB2 AUTHID that the SQL application used, prior to name translation and prior to driving the connection exit at the DB2 site where the SQL application is running.
plan	8	The DB2 PLAN that the SQL application used at the DB2 for MVS site running the SQL program.
accounting_data	n	The MVS accounting string associated with the DB2 SQL application’s MVS address space. This string can be up to 142 bytes in length.

Table 8 (Page 2 of 2). DB2 for MVS Accounting Layout		
Field Name	**Length**	**Description**
rest_prddta_len	0 or 1	This one byte field contains the length of the **rest** of the product specific data field. This is to accommodate non-accounting product specific data. If there is no such data, then the length of this field is zero. Note that this field is only used by SQL/DS to verify that PRDDTA has accounting data. It is not used for any other purpose.
rest_prddta	0 or n	This field contains the rest of the product specific data and exists to accommodate non-accounting product specific data. If there is no such data, then the length of this field is zero. Otherwise the length of this field is indicated in the field **rest_prddta_len**. Note that this field is ignored by SQL/DS.

SQL/DS V3R5 servers consider that there is accounting data during DRDA handshaking if the PRDDTA parameter was passed on the ACCRDB DDM command and one of the following is true:

- **acct_str_len** plus one equals the length of the PRDDTA parameter as indicated on the ACCRDB DDM command.
- **acct_str_len** plus **rest_prddta_len** plus two equals the length of the PRDDTA parameter as indicated on the ACCRDB DDM command.

If the following conditions are all true:

1. the conditions described in the previous paragraph have been met
2. the product name of the **AR** is “DSN”
3. the product version number and product release number are at least “02” and “03” respectively (or the product version number of the **AR** is at least “03”)
4. **acct_str_len** indicates that **accounting_data** has data

then, SQL/DS inserts into USER records, any **accounting_data** data up to 16 characters. If there are more than 16 characters, then only the first 16 are inserted into the USER record. If there are less than 16 characters, then blanks are appended onto the end to make up 16 characters in total. If **accounting_data** has zero length, then blanks are inserted into the USER record.

The accounting_data is the MVS accounting string associated with the DB2 for MVS application′s MVS address space.

MVS allows multiple fields to be supplied in the accounting string. In the example below, there are three MVS accounting fields.

```
STEP1 EXEC PGM=X, ACCT=(X,Y,Z)
```

When multiple MVS accounting fields are present, DB2 for MVS uses an ′FF′X value to delimit the fields. In the previous example, DB2 for MVS will produce a five byte accounting string ″XdYdZ″, where the ″d″ character represents the ′FF′X accounting field delimiter.

See the DB2 for MVS documentation for more details.

SQL/DS to DRDA Server Interface: When SQL/DS (VM) applications connect to other DRDA servers (including SQL/DS servers), and there is SQL/DS accounting data to be sent, it is assumed that the PRDDTA parameter of the ACCRDB DDM command has a structure with the format shown in the following table:

Table 9 (Page 1 of 2). SQL/DS Accounting Layout

Field Name	Length	Description
acct_str_len	1	This one byte length field is the length of the accounting data that SQL/DS will supply, less 1. This length does not include itself, hence the minus one.
prdid	8	This is the product ID for SQL/DS, which includes a three character product name, a two character version, a two character release and a one character modification level. The product ID can be “ARI03050”
rdb_id	18	The Relational Database ID to which the SQL/DS application is connecting.
netid	8	The SNA Netid for the SQL/DS system that created the accounting data.
lu_name	8	The SNA LU name for the SQL/DS system that created the accounting data.
VM_node_id	8	The VM node ID for the SQL/DS system where the application is running.
connection_type	8	The DB2 connection type being used by the SQL/DS application. This can be “IMPLICIT,” which means that the application is implicitly connecting to the DRDA server, or “EXPLICIT,” which means that the application has issued an SQL **CONNECT** statement.
extnam	19	The External Name that is sent in the EXTNAM parameter of the EXCSAT DDM Command. This consists of the VM logon ID concatenated with the CMS work unit ID (if applicable).
authid	8	The SQL/DS AUTHID that the SQL application used, prior to name translation (done by VM based on the “:USERID.” tag in the VM Communications Directory being used).
package_name	8	The name of the package that the SQL/DS application is using at the remote server site.
accounting_data	16	The 16 bytes generated by the SQL/DS accounting exit.
rest_prddta_len	0 or 1	This one byte field contains the length of the **rest** of the product specific data field. This is to accommodate non-accounting product specific data. If there is no such data, then the length of this field is zero. Note that this field is only used by SQL/DS to verify that PRDDTA has accounting data. It is not used for any other purpose.

Table 9 (Page 2 of 2). SQL/DS Accounting Layout		
Field Name	**Length**	**Description**
rest_prddta	0 or n	This field contains the rest of the product specific data and exists to accommodate non-accounting product specific data. If there is no such data, then the length of this field is zero. Otherwise the length of this field is indicated in the field **rest_prddta_len**. Note that this field is ignored by SQL/DS.

From an SQL/DS (VM) requester perspective, the above structure represents data that is sent **only** if the accounting exit has been run with a user exit return code of zero. If the accounting exit returns a nonzero return code (which is the case if the SQL/DS supplied exit has not been modified), then this structure is not sent. Instead, the PRDDTA parameter will consist of 16 blanks.

From an SQL/DS server perspective, Version 3 Release 5 servers will consider that there is accounting data during DRDA handshaking if the PRDDTA parameter was passed on the ACCRDB DDM command and one of the following is true:

- **acct_str_len** plus one equals the length of the PRDDTA parameter as indicated on the ACCRDB DDM command.
- **acct_str_len** plus **rest_prddta_len** plus two equals the length of the PRDDTA parameter as indicated on the ACCRDB DDM command.

If the following conditions are all true:

1. the conditions described in the previous paragraph have been met
2. the product name of the **AR** is “ARI”
3. the product version number and product release number is at least “03” and “05” respectively

then, the 16 byte **accounting_data** field will be inserted into USER records.

Note: SQL/DS V3R3 (VM) and SQL/DS V3R4 (VM) applications send 16 blanks in the PRDDTA parameter of the ACCRDB DDM command. Also, SQL/DS V3R3 and SQL/DS V3R4 servers will not recognize the above accounting layout and will continue to put 16 blanks into User Accounting Records.

The following products are required to pass this data:

- SQL/DS V3R5
- DB2/MVS V2R3 and up
- DDCS/2 V2R1
- DDCS/6000 V2R0

Other DRDA Application Requesters: SQL/DS V3R5 servers will consider that there is accounting data during DRDA handshaking if the PRDDTA parameter was passed on the ACCRDB DDM command and one of the following is true:

- **acct_str_len** plus one equals the length of the PRDDTA parameter as indicated on the ACCRDB DDM command.
- **acct_str_len** plus **rest_prddta_len** plus two equals the length of the PRDDTA parameter as indicated on the ACCRDB DDM command.

If the conditions described in the previous paragraph have been met, and the product name, product version number and product release number combinations are not recognized by SQL/DS, then the SQL/DS server will insert into SQL/DS User Records the character string "**pppvvrrm UNKNOWN**," where pppvvrrm is the prdid of the application requester.

2.1.3 Additional CCSID Support

2.1.3.1 Terminology

Code Page

A set of assignments of characters to code points. The code pages have numeric identifiers; for example, 500 is the international code page.

Coded Character Set

A set of characters in which there is a one-to-one relationship between the characters and their coded representations. These character sets also have numeric identifiers. Several code pages can use the same character set. For example, code pages 37 (English), 273 (German), 277 (Danish-Norwegian), and 500 (International) are some of the code pages that use character set 697. However, these code pages do not necessarily use the same subset of the character set.

CCSID Coded Character Set Identifier, a 16-bit number that includes an encoding scheme identifier, character set identifiers, code page identifiers, and other information that uniquely identifies the coded character representation used. For single and double byte CCSIDs, the CCSID is generally the same as the code page. For example, the CCSID for the international code page is 500.

EUC Extended UNIX Code

ROECE Regional Office for East and Central Europe. **Note** The term IBM EE, that is, IBM Eastern Europe, has now replaced ROECE.

2.1.3.2 Description

As the number of CCSIDs supported on databases such as DB2/2 and DB2/6000 increases, the demand for support for conversion from these new CCSIDs to compatible host CCSIDs increases as well. This enhancement addresses this demand, by increasing the support available for Traditional Chinese, Simplified Chinese, Korean, Cyrillic, Greek and Windows code pages. In order to support the requested conversions, it is necessary to add support for two new CCSIDs on the host as well.

Specifically, this enhancement adds support for the following CCSIDS:

- 423: Greek (Coexistence CCSID)
- 1025: Cyrillic Multilingual

and enables support for the following CCSID conversions:

- Traditional Chinese:
  - PC mixed data to host mixed data (950 -> 937), which entails the following conversions:
    - 950 -> 937

        - 1114 -> 937
        - 1114 -> 28709
        - 950 -> 28709
        - 947 -> 835
    - Simplified Chinese:
        - PC mixed data to host mixed data (1381 -> 935), which entails the following conversions:
            - 1381 -> 935
            - 1381 -> 836
            - 1115 -> 935
            - 1115 -> 836
            - 1380 -> 837
        - EUC to host mixed data (1383 -> 935), which entails the following conversions:
            - 1383 -> 935
            - 1383 -> 836
            - 367 -> 935
            - 367 -> 836
            - 1382 -> 837
    - Korean:
        - EUC to host mixed data (970 -> 933), which entails the following conversions:
            - 970 -> 933
            - 970 -> 833
            - 367 -> 933
            - 367 -> 833
            - 971 -> 834
    - Cyrillic:
        - PC ROECE Cyrillic to host Cyrillic Multilingual (855 -> 1025)
        - PC Cyrillic-2 to host Cyrillic Multilingual (866 -> 1025)
        - AIX Cyrillic to host Cyrillic Multilingual (915 -> 1025)
    - Windows code pages:
        - Windows Latin-2 to host Latin-2 (1250 -> 870)
        - Windows Cyrillic to host Cyrillic (1251 -> 1025)
        - Windows Latin-1 to host English (1252 ->37)
        - Windows Latin-1 to host German (1252 ->273)
        - Windows Latin-1 to host Danish (1252 ->277)
        - Windows Latin-1 to host Finnish (1252 ->278)
        - Windows Latin-1 to host Italian (1252 ->280)

- Windows Latin-1 to host Spanish (1252 ->284)
- Windows Latin-1 to host UK English (1252 ->285)
- Windows Latin-1 to host French (1252 ->297)
- Windows Latin-1 to host International (1252 ->500)
- Windows Latin-1 to host Icelandic (1252 ->871)
- Windows Greek to host Greek-423 (1253 -> 423)
- Windows Greek to host International (1253 -> 500)
- Windows Greek to host Greek (1253 -> 875)
- Windows Hebrew to host Hebrew (1255 -> 424)
- Windows Arabic to host Arabic (1256 -> 420)

Invocation: There is no explicit invocation for this function. It is invoked internally when conversion is required.

2.1.4 FORCE Without DISABLE

2.1.4.1 Description

The FORCE operator command of SQL/DS has a DISABLE option. In SQL/DS V2R2 (VM) if the DISABLE option is specified, then:

- The SQL/DS (VM) connection to the application is disabled
- The logical unit of work is rolled back
- The application server makes the VM connection available for another user (after the rollback finishes)
- The application receives an SQLCODE of -933. This is flagged as a SEVERE error to the application. At this point, the application **MUST** issue a CONNECT statement to continue

If the DISABLE option is **not** specified, then:

- The SQL/DS (VM) connection to the application is disabled
- Logical unit of work is rolled back
- The application server makes the VM connection available for another user (after the rollback finishes)
- The application receives an SQLCODE of -948 (or -916 in VM/XA). This is not flagged as a severe error to the application. At this point, the application can issue any SQL statement and an implicit CONNECT will be issued for the application if necessary.

FORCE without DISABLE would not always work when a conversation required VTAM. This function was disabled in SQL/DS Version 3 and made to behave as **FORCE with DISABLE**, to operate in a consistent manner.

The reason why it doesn′t work consistently over VTAM has to do with how VTAM handles SEVER commands. When the operator issues a **FORCE without DISABLE** command, SQL/DS severs the conversation with an IPCODE=x′210′. The SQL/DS requester ought to receive an indication of the SEVER along with the IPCODE of x′210′ and then process it accordingly. However in certain circumstances, VTAM **changes** the IPCODE from x′210′ to x′610′. This can occur

when VTAM needs to send the SEVER **against** the flow of communications (for instance, when the application server is in **RECEIVE** state and the server **sends** the SEVER). As a result the requester will not be able to detect if the DISABLE operand was used.

The purpose of this functional enhancement is to enable support for SQLCODE -948. This new SQL/DS V3R5 function works if VTAM is not involved in communications and sometimes works if VTAM **is** involved in communications. In short, the function does not work in every case, but at least it can work in most of the cases.

This functional enhancement also fixes a problem found in SQL/DS V3R4 when the FORCE command with the DISABLE option is used on a VM server in a DRDA environment. In SQL/DS V3R4, if such a FORCE command is issued, the server can incorrectly issue SQLCODE -916 and **not** sever the connection. With this functional enhancement included in SQL/DS V3R5, the server will instead sever the connection and the requester will respond by returning SQLCODE -30080 to the application.

2.1.4.2 Invocation

The invocation of the FORCE command is not changed. DISABLE **must** now be specified if this is the intent.

2.1.4.3 Interactions

An example scenario is as follows:

- The user accesses the SQL/DS server through ISQL and issues a query
- The operator performs the following:

```
show connect
Status of Connected SQL/DS Users                1994-09-19  10:04:41
   Checkpoint agent is not active.
   User Agent:   1   User-ID: SQLUSRJR SQL-ID: SQLUSRJR
      is R/O  APPL  1CE0
      Agent is processing and is in communication  wait.
                     State started: 1994-09-19  10:04:21
                     Conversation started: 1994-09-19  10:04:13
                     LUWID: SNANETID.*IDENT.A9C04B855CB4.0001
                     EXTNAM: SQLUSRJR.1
                     Requester: SQLDS/VM  V3.5.0   at TOROLAB2
  1  SQL/DS users are active.
  0  SQL/DS users are waiting.
  0  SQL/DS users are inactive.
  0  SQL/DS agents are available.
  4  SQL/DS user connections are available.
ARI0065I SQL/DS operator command processing is complete.
force 1
ARI0230I FORCE ROLLBACK without disable scheduled
         for agent SQLUSRJR because of operator request.
ARI0065I SQL/DS operator command processing is complete.
```

Figure 32. Operator Interactions for FORCE Without Disable

- The application would then receive the following:

```
ARI0503E An SQL error has occurred.
         The current logical unit of work was rolled back,
         and the connection to SQL/DS was severed
         because of operator action.
ARI0505I SQLCODE = -948 SQLSTATE = 57027 ROWCOUNT = 0
ARI0504I SQLERRP: ARIRADPC SQLERRD1: 0 SQLERRD2: 0
ARI0502I Following SQL warning conditions encountered:
         implicit rollback

ARI7021E SQL/DS has issued a ROLLBACK statement.
         All work entered for SQL/DS processing since
         the last COMMIT statement was rolled back.
         You may have to reenter some statements.
```

Figure 33. Message on Requester from Force Without Disable

2.2 Performance Enhancements

2.2.1 Increased Buffer Maximums

2.2.1.1 Description

This enhancement substantially increases the number of page and directory buffers that can be used by the database manager. Prior to SQL/DS V3R5, the maximums were 3500 page buffers and 28000 directory buffers. Now, in SQL/DS V3R5, the maximums have been increased to 400000 page buffers and 400000 directory buffers. A user who allocates the maximum page and directory buffers will be allocating 1.8 gigabytes of virtual storage for these buffers, leaving approximately 200 megabytes for everything else (assuming a 2 Gigabyte virtual machine or partition). The new maximums are much larger than most installations will be able to use.

With the large increase in real storage available on modern System 390 processors and the ability of the database to use 31-bit addressable virtual storage, the number of buffers used before SQL/DS V3R5 was preventing some installations from fully exploiting the available processor resources. Allowing users to increase the number of buffers in SQL/DS V3R5 helps them exploit these resources and improve their database performance.

Note: The performance improvement will depend on the availability of sufficient real storage to back up the virtual storage buffers.

In addition, a variety of small performance improvements has been made to the code for initializing, searching and manipulating the buffers.

2.2.1.2 Invocation

The use of the increased buffer maximums is invoked using the available database manager start up parameters NPAGBUF and NDIRBUF. Users are now able to specify either or both of these parameters to a maximum of 400000.

The minimums and the calculated defaults for NPAGBUF and NDIRBUF are unchanged by this support.

The new maximums can be specified in both Single and Multiple User Mode, under both VM/ESA and VSE/ESA, and regardless of whether the VMDSS feature or DRDA support is being used.

Confirmation of the values used by the database manager is reported on the startup console log via message ARI0016I.

Performance implications are discussed in 3.1, “Increased Buffer Maximums” on page 87.

2.2.2 Archive Performance Enhancements

2.2.2.1 Description

In releases prior to SQL/DS V3R5, the database manager archive process examined each 512 page cell to see if any pages within the cell were allocated. If a cell contained any allocated pages, then all pages within the cell were archived, regardless of whether they were allocated or not. The I/O operations were performed synchronously, with only one I/O operation occurring at any moment. There was no overlap between reading from DASD and writing to tape.

In SQL/DS V3R5, enhancements were made to improve the performance of the database manager and log archiving processes by introducing the following changes:

- Archive only allocated pages. The database manager archive process will continue to examine each cell, however it will only archive pages that have been allocated. Pages that have not been allocated will not be archived.
- Use Multiple-Block *BLOCKIO requests in a single IUCV SEND request. Using Multiple-Block *BLOCKIO, seven pages are chained together to be read with a single IUCV request. The data is stored in a user buffer of 28K. This implementation is only possible for VM. Multiple-Block *BLOCKIO is not available for VSE.
- Use asynchronous I/O to reduce the wait time for DASD read by having concurrent disk and tape operations. This involves performing a complete read operation from disk into one buffer, beginning a second read operation from disk into a second buffer, then writing to tape from the first buffer while the second read operation is still proceeding. This will continue until the archive has completed. The asynchronous I/O is only implemented in VM because it is not compatible with VSAM controlled buffers in VSE.
- Use VSAM controlled buffers in VSE. By using VSAM controlled buffers and sequential processing, VSAM is able to read multiple records with a single I/O request. The number of records will depend on the number of buffers available to VSAM, which will be specified when creating the Access Method Control Block (ACB) for the directory, data and log disks. A second set of ACBs is required in order to avoid interfering with normal I/O processing during online archives. Implementing VSAM controlled buffers will also require using SHAREOPTIONS(2) for the directory, data and log disks.

2.2.2.2 Usage Notes

SQL/DS defaults to 40 buffers for the ACBs. Users can change this number by specifying the BUFFND parameter in the JCL DLBL statements for the directory, data and log disks.

2.2.3 Utilizing Virtual Disk for VM and VSE

2.2.3.1 Virtual Disk Support for VM/ESA

If your operating system is VM/ESA 1.2.1 or later, your internal dbspaces can make use of a virtual disk to improve their performance. A virtual disk is defined by VM in virtual storage, backed up by VM Paging DASD, and will normally be much faster than real DASD. The virtual disk appears to any program just as a real disk, only faster.

NOTE

Remember, virtual disk storage is temporary. All data on a virtual disk is lost when it is detached from a user ID or when the user ID logs off. For this reason, you **MUST NOT** use virtual disk for anything other than internal dbspaces. These dbspaces are only used as temporary workspace, so it does not matter if their contents are lost. Consequently, in order to guarantee the above recommendation, the storage pool containing the virtual disk **MUST NOT** be used for any public or private dbspaces in the same pool using virtual disk; otherwise, anyone can acquire these dbspaces by mistake and it can corrupt your catalogs and the result is unpredictable.

While the use of virtual disks is restricted to internal dbspaces, they can be used to improve the performance of index creations, joins, sorts, and other operations that require temporary workspace.

Remember, anytime you increase your use of virtual storage, you can realize significant performance improvements, but only if you have the real storage to support it.

Note: It is recommended that database generation be done with real minidisks only. If you decide to generate a database which uses a virtual disk, take care to ensure that the virtual disk is used in a pool containing only internal dbspaces.

Using Virtual Disks with Internal Dbspaces: To use a virtual disk with internal dbspaces, you must:

1. Take an archive of your database before making any changes to it.
2. Define a virtual disk in the database manager′s VM Directory entry.
3. Run the SQLADBEX EXEC to add the virtual disk as the first dbextent of a **new** storage pool. The virtual disk **must** be the first dbextent in the new pool. If you have dbextents in the old storage pool where your internal dbspaces are currently defined, you should delete some of these dbextents from the old pool and add them to the new pool. Alternatively, you could add one or more real disk dbextents to the new pool.
4. Run the SQLADBSP EXEC to ′move′ your internal dbspaces to the new pool.
5. Modify the ′CP LINK′ command for the virtual disk in the ′dbname SQLFDEF Q′ file.

6. Modify the database manager's PROFILE EXEC or database start up EXEC to CMS FORMAT and RESERVE the virtual disk and make a duplicate LINK to this virtual disk (this is explained further in the following detailed steps).
7. Take an archive of your database after making the above changes, so that you have an archive which reflects these changes.

A detailed example of how to complete these steps appears below. Please read all the steps before executing any of them.

Take an Archive of Your Database: This will be needed if problems arise during the setup for using virtual disks and you need to restore your database to its previous state without virtual disks.

Define a Virtual Disk in the Database Manager's VM Directory Entry: Determine the size of the virtual disk to be added. It should be large enough for most sorts, but must not be so large as to cause excessive VM paging. Define this virtual disk in the VM Directory, similar to:

```
MDISK 0329 FB-512 V-DISK nnnnnnnn M
```

For information about defining a virtual disk in a VM Directory, refer to the *VM/ESA Planning and Administration* manual.

Add Dbextents to a New Storage Pool: Add the virtual disk, and possibly other dbextents, to a new storage pool. For information about using the SQLADBEX EXEC, refer to the *SQL/DS System Administration for IBM VM Systems* manual.

To avoid using too much real storage, it is recommended that you include at least two dbextents in the new pool. The virtual disk **must** be the first dbextent in the new pool. Other dbextents should be real minidisks to accommodate the overflow from the virtual disk. You can use some of the dbextents from the pool that originally contained the internal dbspaces. Make the total size of the dbextents in the new pool large enough to accommodate your current internal dbspaces.

Also, ensure that you add the virtual dbextent before you add any real minidisks. The database manager searches the dbextents for a free page in the order they were added. If you are using the SQL/DS VMDSS feature, modify your ARISPOOL file to specify sequential allocation (″SEQ″) instead of striping (″STR″) for this storage pool.

Warning: Do not accidentally place the virtual disk in an existing storage pool containing permanent dbspaces. You will lose valuable data and a database restore may be required!

When you run the SQLADBEX EXEC, you specify the actions to be taken by answering the prompts. When you see message ARI6145D, reply 1 (yes), to view the ′dbname SQLADBEX A′ file created from the prompts.

Assume you have the following setup and are changing to use a virtual disk for internal dbspaces:

- Your internal dbspaces are currently defined in pool 3, which also contains permanent dbspaces.
- Pool 3 currently has three dbextents, addresses 324, 325 and 326, which correspond to dbextent numbers 4, 5 and 6

- You have a virtual disk defined at address 329 and you want to add it to the new pool number 8, as dbextent number 9.
- You want to ′move′ dbextents 5 and 6 in pool 3 to the new pool 8.
- You do want to take an archive after these changes.

  Note: Running the SQLADBEX EXEC will cause a break in the continuity of your log archives, so a database archive should **always** be taken.

After starting the SQLADBEX EXEC and entering the information at the prompts, reply 1 (yes) to message ARI6145D. This will display the ′dbname SQLADBEX A′ file in XEDIT, allowing you to review and/or modify the contents. This file specifies the sequence of actions that SQLADBEX will perform. Given our assumptions above, the file will appear as follows:

```
ADD 9 8                          <-- add virtual disk to pool 8
DELETE 5 3                       <-- delete dbextent 5 from pool 3
DELETE 6 3                       <-- delete dbextent 6 from pool 3
ADD 5 8                          <-- add dbextent 5 to pool 8
ADD 6 8                          <-- add dbextent 6 to pool 8
ARCHIVE                          <-- an archive will be taken
```

Refer to *SQL/DS System Administration for IBM VM Systems* for more details, cautions and warnings concerning the adding and deleting of dbextents.

The last step of the SQLADBEX EXEC will update the ′dbname SQLFDEF Q′ file, to match the added and/or deleted dbextents. The ′CP LINK′ command for the virtual disk in this file must be updated; this is documented in a following step.

Move the Internal Dbspaces to the New Storage Pool: This step will ′move′ the internal dbspaces from the old pool to the newly added pool which contains the virtual disk, by using the SQLADBSP EXEC. You may also add permanent dbspaces to the database at this time (except into the new pool) or you can simply redefine the pool where internal dbspaces will be placed. Remember, this new pool, with the virtual disk dbextent, can only contain internal dbspaces. For information about using the SQLADBSP EXEC, refer to *SQL/DS System Administration for IBM VM Systems*.

Modify the ′dbname SQLFDEF Q′ File: In this step, you will edit and modify the ′dbname SQLFDEF Q′ file to change the ′CP LINK′ mode for the virtual disk dbextent. This is required to allow the virtual disk to only be formatted and reserved once per IPL CMS of the database manager. If this step is not performed, the virtual disk must be formatted and reserved prior to each startup of the database manager (that is, SQLSTART).

This is done by having the virtual disk linked to the database manager virtual machine twice. When the ′dbname SQLFDEF Q′ file is used to detach the virtual disk, the second link remains attached to the machine and the formatting of the virtual disk is not lost. This second link and the formatting are done from the database manager′s PROFILE EXEC (this is set up in the next step).

Be sure you have the production minidisk (normally ′Q′) accessed R/W. XEDIT the ′dbname SQLFDEF Q′ file. Locate the line containing the ′CP LINK USERID cuu cuu W′ statement for the address (′cuu′) of the virtual disk. Change the CP LINK MODE character from ′W′ to ′M′.

Warning: if you delete the virtual disk extent and add it again (via SQLADBEX EXEC), you **must** again change the LINK MODE character from 'W' to 'M'.

Modify the PROFILE EXEC: In this step you will modify the database manager's PROFILE EXEC so that the virtual disk will be CMS formatted and reserved each time the database manager virtual machine IPLs CMS. In addition, a second link to the virtual disk will be set up (see the previous step).

Warning: If an error occurs such that the virtual disk is not usable, the database cannot be brought up. In this situation, you must correct the error to make the virtual disk usable, or you must replace the virtual disk with a real minidisk at the same address (and at least the same size). The replacement minidisk must be formatted and reserved before the database is brought up.

It is recommended that a separate EXEC be created to perform the LINK, FORMAT and RESERVE commands, and that this EXEC be called from the PROFILE EXEC. You can place the following statements in your PROFILE EXEC to initialize the virtual disk for usage:

```
¢EXEC PREPVDSK¢                 /* Call EXEC to Prepare Virtual Disk */
 If rc    0 Then Do;
    Say "PREPVDSK rc =" rc;
    Exit rc;
 End
```

Note that the 'If' statement above will cause the PROFILE EXEC to end if an error is returned from the PREPVDSK EXEC. This assumes that the PROFILE EXEC will eventually invoke the SQLSTART EXEC after the virtual disk has been initialized. This is done because the database cannot be started if the virtual disk is not properly initialized. You may need to tailor this processing to suit your particular operational environment.

The following is a sample 'PREPVDSK EXEC':

```
/* REXX */ Trace ¢O ¢;   Address ¢COMMAND¢
/*  This EXEC is used to FORMAT and RESERVE a Virtual Disk,          */
/* that is used as the FIRST Dbextent of a Storage Pool containing   */
/* ONLY SQL/DS INTERNAL DBSPACES.                                    */
/*                                                                   */
/* WARNING: This process, to use a Virtual Disk, requires that the   */
/*          CP Link Mode letter be changed from ¢W¢ to ¢M¢ in the    */
/*          SQLFDEF file for the CP LINK command for the DATABASE    */
/*          Address of the Virtual Disk.                             */
/*                                                                   */
/*  This EXEC should be called from the PROFILE EXEC of the Database */
/* Virtual Machine, to prepare the Virtual Disk for use.             */
/*           (once per LOGON/IPL of the Database Machine)            */
/*                                                                   */
/*  The Virtual Disk MDISK is Linked R/O with an unused address, so  */
/* that subsequent detaches by the SQLFDEF file of the               */
/* normal address will NOT lose the FORMAT/RESERVE information.      */
/*                                                                   */
/*  The Virtual Disk MDISK is Linked again, R/W, with its normal     */
/* address, for the FORMAT/RESERVE processing. This address is then  */
/* Detached.  It will be Linked again later by the SQLFDEF file when */
/* the database machine runs the SQLSTART EXEC.                      */
/*                                                                   */
/*  If the R/O Link of the Virtual Disk is detached by mistake,      */
/*  you MUST run this EXEC before running SQLSTART again.            */
/*                                                                   */
/********* UPDATE THE FOLLOWING 5 VARIABLES AS APPROPRIATE:  ********/
dbname = ¢dbname  ¢              /* Database Name                    */
pdisk  = ¢0cuu¢                  /* Virtual Disk PERMANENT Address   */
vdisk  = ¢0cuu¢                  /* Virtual Disk DATABASE  Address   */
vlabel = ¢DDKnn   ¢              /* Virt Disk Label (Dbextent Number)*/
ufm    = ¢Z¢                     /* Unused Filemode Letter           */

z=Diagrc(8?,¢CP DETACH¢ vdisk) /* Be sure DATABASE Addr is NOT Linked */
z=Diagrc(8?,¢CP LINK *¢ vdisk pdisk ¢RR¢)   /* Get PERMANENT R/O Link */
Parse Var z cprc . ; If cprc  = 0 Then Exit rc
z=Diagrc(8?,¢CP LINK *¢ vdisk vdisk ¢M¢)    /* Get DATABASE  R/W Link */
Parse Var z cprc . ; If cprc  = 0 Then Exit rc
¢SET CMSTYPE HT¢; ¢RELEASE¢ ufm; ¢SET CMSTYPE RT¢
Push vlabel
Push ¢1¢
¢FORMAT¢ vdisk ufm ¢(BLKSIZE 4096¢               /* FORMAT the Vdisk */
If rc  = 0 Then Exit rc
Push ¢1¢
¢RESERVE¢ dbname vlabel ufm                     /* RESERVE the Vdisk */
If rc  = 0 Then Exit rc
¢SET CMSTYPE HT¢; ¢RELEASE¢ ufm; ¢SET CMSTYPE RT¢
z=Diagrc(8?,¢CP DETACH¢ vdisk)  /* Detach the DATABASE address again, */
Exit 0        /* ... it will be re-Linked by SQLFDEF during SQLSTART. */
```

Figure 34. PREPVDSK EXEC Sample

Take an Archive of Your Database: Neither the ADD DBEXTENT nor the DELETE DBEXTENT operation is recorded in the log. Because these operations update the directory, but not the database itself, you can encounter a problem if you normally archive the database, and then try to restore that archive, with the add or delete occurring in between the archive and the restore. For more information about this problem, refer to *SQL/DS System Administration for IBM VM Systems*. This is why an archive should be taken both before and after making the changes to use a virtual disk.

2.2.3.2 Virtual Disk Support for VSE/ESA

If your operating system is VSE/ESA 1.3 or later, your internal dbspaces can make use of a virtual disk to improve their performance. Virtual Disk Support allows you to use a data space as a virtual disk. Data that would otherwise reside on DASD can be stored in a virtual disk and since the virtual disk uses main storage instead of DASD, it is much faster than a conventional disk. The virtual disk appears to any program or job as just another disk, only faster.

NOTE

Remember, virtual disk storage is temporary. Anything in a virtual disk is lost whenever the VSE operating system is restarted. For this reason, you **MUST NOT** use a virtual disk for anything other than internal dbspaces. These dbspaces are only used as temporary workspace, so it does not matter if their contents are lost. Consequently, in order to guarantee the above recommendation, the storage pool containing the virtual disk **MUST NOT** be used for any permanent dbspaces. Also, you **MUST NOT** add either public or private dbspaces in the same pool using the virtual disk; otherwise, anyone can acquire these dbspaces by mistake and it can corrupt your catalogs and the result is unpredictable.

While the use of virtual disks is limited to internal dbspaces, they can improve the performance of index creation, joins, sorts, and other operations that require temporary workspace.

Note: It is recommended that database generation be done with real minidisks only. If you decide to generate a database which uses a virtual disk, take care to ensure that the virtual disk is used in a pool containing only internal dbspaces.

Remember, anytime you increase your use of virtual storage, you can realize significant performance improvements, but only if you have the real storage to support it.

Using Virtual Disks with Internal Dbspaces: To use a virtual disk with internal dbspaces, you must:

1. Modify the IPL procedure and the background initialization procedure to create a virtual disk.
2. Define a VSAM user catalog and dbextent on the virtual disk.
3. Add a label for the dbextent in the cataloged procedure and add the dbextent to a **new** storage pool that will contain only internal dbspaces.
4. Move some of the dbextents from the original pool that contained the internal dbspaces into the new pool or define additional physical dbextents to be added to the new pool.
5. Add internal dbspaces to the new storage pool.
6. Back up the VSAM user catalog defined on the virtual disk so that it can be restored whenever the VSE system is restarted.
7. Modify the application server startup job to restore the VSAM user catalog if the VSE system has been restarted since the VSAM user catalog and dbextent were created.

For more information on virtual disks in VSE/ESA, refer to the *VSE/ESA Planning* and the *VSE/ESA Extended Addressability* manuals. A detailed example of how to complete these steps appears below. Please read all the steps before executing any of them.

Modify IPL Procedure: Modify the IPL procedure to do the following:

- Include ADD statements for the virtual disk addresses
- Increase VSIZE and page data set allocation to accommodate the new virtual disk.

For example, consider a system where we are adding a 20MB virtual disk to a VSE system with VSIZE=75MB:

```
...
009,$$A$SUPX,VSIZE=95M,VIO=576K,VPOOL=194K,LOG
...
ADD 900:906,FBAV
...
DPD VOLID=DOSRES,CYL=209,NCYL=100,TYPE=N,DSF=N
DPD VOLID=DOSRES,CYL=398,NCYL=8,TYPE=N,DSF=N
DPD VOLID=DOSRES,CYL=410,NCYL=29,TYPE=N,DSF=N
...
DLA NAME=AREA1,VOLID=DOSRES,CYL=60,NCYL=3,DSF=N
SVA PSIZE=640K,SDL=300,GETVIS=768K
/+
/*
...
```

This example increases the VSIZE of the VSE operating system from 75MB to 95MB. It reserves virtual addresses 900 through 906 for fixed block architecture virtual disks, and it sets aside an additional 29 cylinders of 3390 DASD on the DOSRES volume starting at address 410 for system paging DASD. (108 cylinders were already being used at addresses 209, and 398. Also, there are 180 4KB pages in every 3390 cylinders. So 29 cylinders is equal to 20MB.)

Define and Initialize a Virtual Disk: Create a procedure (to be invoked as part of the background initialization JCL, for example $0JCL) that will do the following:

- Define the data space size using the SYSDEF command.
- Initialize the virtual disks using the // VDISK command.

For example:

```
...
*  DEFINE THE SIZE OF THE DATASPACE
// SYSDEF DSPACE,DSIZE=20M
*  DEFINE AND INITIALIZE EACH VIRTUAL DISK
// VDISK UNIT=900,BLKS=40320,VOLID=QPVDS1,VTOC=008
/*
/+
...
```

This example reserves approximately 20MB of virtual storage for virtual disks. (Remember you do not need 20MB of real storage to support 20MB of virtual storage.) The virtual disk at address 900 uses approximately 20MB of that storage (40320 512-byte blocks, with eight 512-byte blocks of that reserved for the VTOC). (The virtual disk addresses were defined in “Modify IPL Procedure” on page 61.)

Note: Because of the structure of a virtual disk, blocks must be allocated in multiples of 960. So instead of 2048 512-byte blocks for 1MB of storage, you can only allocate 1920 blocks.

Define a Backup File: Define a sequential file on a real disk for use later when backing up the VSAM user catalog (defined on the next step on a virtual disk).

For example:

```
...
*  DEFINE A SEQUENTIAL FILE FOR VDISK UCAT BACKUP
// EXEC IDCAMS,SIZE=AUTO
   DEFINE NONVSAM (NAME(VDISK1.UCAT.BKUP) -
          DEVICETYPES(3390) VOLUMES(SYSWK1))
/*
...
```

This example creates a backup file called *VDISK1.UCAT.BKUP*.

Define a VSAM User Catalog: Define a VSAM user catalog on the virtual disk using the DEDICATE option.

For example:

```
...
*  DEFINE A VSAM USER CATALOG
// EXEC IDCAMS,SIZE=AUTO
   DEFINE USERCATALOG ( -
          NAME (VDS1.USER.CATALOG) -
          DEDICATE -
          VOLUME (QPVDS1))
/*
...
```

This example creates a VSAM user catalog called *VDS1.USER.CATALOG* on volume *QPVDS1* and dedicates the entire volume for this VSAM user catalog. (Volume *QPVDS1* was defined in “Define and Initialize a Virtual Disk” on page 61.)

Define a Virtual Disk Dbextent: Define a dbextent (VSAM cluster) on the virtual disk.

For example:

```
...
*  ADD CLUSTERS FOR SQL/DS DATA BASE
// DLBL VDSUC1,¢VDS1.USER.CATALOG¢,,VSAM
*  DEFINE CLUSTERS
// EXEC IDCAMS,SIZE=AUTO
   DEFINE CLUSTER (NAME(SQL35.DDSK8.VDSK.DB) NONINDEXED REUSE -
            CNVSZ (4096) BLOCKS(39360) VOL(QPVDS1) -
            RECSZ (4089 4089) SHR(2)) CAT(VDS1.USER.CATALOG)
/*
...
```

This example defines a dbextent named *SQL35.DDSK8.VDSK.DB* in the VSAM user catalog *VDS1.USER.CATALOG*. (The VSAM user catalog was defined in "Define a VSAM User Catalog" on page 62.)

Add a Label for the Virtual Disk Dbextent: Update the cataloged procedure to include a DLBL statement for the new dbextent.

For example:

```
...
*  CATALOG SQL/DS DBEXTENT LABELS
// EXEC LIBR,PARM=¢MSHP¢
ACCESS S=IJSYSRS.SYSLIB
CATALOG DTLDVDSK.PROC R=Y
*  ***************** SQL/DS DBEXTENT LABELS ***************
// DLBL TSQLUC,¢TSQL.USER.CATALOG¢,,VSAM
// DLBL VDSUC1,¢VDS1.USER.CATALOG¢,,VSAM
// DLBL BDISK,¢SQL35.BDISK.DTLD.DB¢,,VSAM,CAT=TSQLUC
// DLBL LOGDSK1,¢SQL35.LOGDSK1.DTLD.DB¢,,VSAM,CAT=TSQLUC
// DLBL DDSK1,¢SQL35.DDSK1.DTLD.DB¢,,VSAM,CAT=TSQLUC
...
// DLBL DDSK7,¢SQL35.DDSK7.DTLD.DB¢,,VSAM,CAT=TSQLUC
// DLBL DDSK8,¢SQL35.DDSK8.VDSK.DB¢,,VSAM,CAT=VDSUC1
/+
/*
...
```

This example adds a DLBL statement for *DDSK8* that identifies dbextent *SQL35.DDSK8.VDSK.DB.* (The dbextent was created in "Define a Virtual Disk Dbextent" on page 62.) *DDSK7* is one of the dbextents that belonged to the original internal dbspace storage pool.

Add Dbextents to a New Storage Pool: Add dbextents to the new storage pool by following instructions included in the *System Administration* manual. (Refer to *"Adding and Deleting Dbextents."*)

To avoid using too much real storage, it is recommended that you include at least two dbextents in the new pool. The first must be the virtual disk. The second should be a physical dbextent that can accommodate the overflow from the virtual disk. You can use some of the dbextents from the original pool that contained the internal dbspaces (in this example DDSK7). Make the total size of both dbextents large enough to accommodate your current internal dbspaces

and make the virtual disk as large as possible without over-committing real storage.

Also, ensure that you add the virtual dbextent before you add any physical dbextents. The database manager searches the dbextents for a free page in the order that they were added.

Warning: Do not accidentally place the virtual disk in an existing storage pool that contains anything other than internal dbspaces. You will lose valuable data.

The following is an example of the ARISADD member, which specifies how procedure ARIS250D will add and delete dbextents to and from pools:

```
POOL 9
ADD 8 9
DELETE 7
ADD 7 9
ARCHIVE
```

This example adds dbextent 8 (DDSK8) to storage pool 9. (DDSK8 was identified in “Add a Label for the Virtual Disk Dbextent” on page 63.) Ensure that you add the virtual disk dbextent to a **new** storage pool (reserved only for internal dbspaces). It also removes dbextent 7 (DDSK7) from the original pool that contained the internal dbspaces. It then adds it to the new pool (pool 9) that contains the virtual disk dbextent.

Back Up the Virtual Disk VSAM User Catalog: Back up the VSAM user catalog defined on the virtual disk into the sequential file on a real disk.

For example:

```
...
// LIBDEF PHASE,SEARCH=IJSYSRS.SYSLIB
*  THIS JOB UNLOADS A VSE/VSAM CATALOG USING THE REPRO COMMAND
// DLBL IJSYSCT,¢VSAM.MASTER.CATALOG¢,,VSAM
// ASSGN  SYS001,DISK,VOL=SYSWK1,SHR
// DLBL   CATOUT,¢VDISK1.UCAT.BKUP¢,999
// EXTENT SYS001,SYSWK1,1,0,33315,75
// DLBL   IJSYSUC,¢VDS1.USER.CATALOG¢,                              X
             ,VSAM
// EXEC IDCAMS,SIZE=AUTO
       REPRO INFILE (IJSYSUC) -
             OUTFILE (CATOUT -
               ENVIRONMENT ( -
                 BLOCKSIZE (2068) -
                 RECORDFORMAT (VARBLK) -
                 RECORDSIZE (516) -
               ) -
             )
/*
...
```

This example unloads the VSAM user catalog *VDS1.USER.CATALOG* to the backup file *VDISK1.UCAT.BKUP.* (The backup file was created in “Define a Backup File” on page 62.) The backup file will be used to restore the VSAM

user catalog on the virtual disk whenever the VSE system is restarted. Restoring the VSAM user catalog redefines the VSAM space and cluster previously defined on the virtual disk and sets the high used RBA to what is was before the system restart, thus allowing the database manager to successfully use it.

Add Internal Dbspaces to the New Pool: Invoke the application server to add only internal dbspaces into this new pool.

For example:

```
...
*  ADD INTERNAL DBSPACES TO THE SQL/DS DATA BASE
// LIBDEF *,SEARCH=PRD2.SQL350
// EXEC PROC=DTLDVDSK
// EXEC ARISQLDS,SIZE=AUTO,PARM=¢SYSMODE=S,STARTUP=S¢
   INTERNAL 50 1024 9
/*
...
```

This example JCL creates 50 internal dbspaces, each of 1024 4KB pages, in storage pool 9. (Storage pool 9 was created in “Add Dbextents to a New Storage Pool” on page 63.)

Add Conditional JCL to Application Server Startup: Create conditional JCL that does the following:

- Check that the dbextent defined earlier on the virtual disk still exists (1 and 2). If it does not, do the following:
  - Disconnect the VSAM user catalog from the master catalog (3)
  - Redefine the VSAM user catalog on the virtual disk (4)
  - Restore the VSAM user catalog from the sequential file (5)
  - If the restore fails for any reason cancel the job (6).
- Start the application server.

For example:

```
...
// DLBL IJSYSCT,¢VSAM.MASTER.CATALOG¢,,VSAM
// ASSGN SYS001,DISK,VOL=SYSWK1,SHR
// DLBL VDSBKUP,¢VDISK1.UCAT.BKUP¢
// EXTENT SYS001,SYSWK1,1,0
// DLBL IJSYSUC,¢VDS1.USER.CATALOG¢,,VSAM
// EXEC IDCAMS,SIZE=AUTO
   LISTCAT CAT(VDS1.USER.CATALOG) ENT(SQL35.DDSK8.VDSK.DB) ALL     (1)
   IF LASTCC NE 0 THEN DO                                          (2)
      EXPORT VDS1.USER.CATALOG DISCONNECT                          (3)
      DEFINE USERCATALOG ( NAME(VDS1.USER.CATALOG) -               (4)
                  DEDICATE VOLUME (QPVDS1))
      IF LASTCC NE 0 THEN CANCEL JOB
      REPRO INFILE (VDSBKUP ENVIRONMENT -                          (5)
                (BLOCKSIZE (2068) -
                RECORDFORMAT (VARBLK) -
                RECORDSIZE (516))) -
             OUTFILE (IJSYSUC)
      IF LASTCC GT 4 THEN CANCEL JOB                               (6)
   END
/*
...
```

Place this sample JCL in front of your current application server startup job. It checks for the existence of dbextent *SQL35.DDSK8.VDSK.DB* in the VSAM user catalog *VDS1.USER.CATALOG*. If it no longer exists, the JCL restores the VSAM user catalog from the backup created in “Define a Backup File” on page 62. The server will then start normally and will use the virtual disk for internal dbspaces.

2.2.4 Assembler Host Variables/Even Precision

2.2.4.1 Description

Prior to SQL/DS V3R5, even precision packed decimal variables were not supported by the SQL/DS Assembler preprocessor. These variables are supported by DB2 for MVS. This new function implemented in SQL/DS V3R5 allows the SQL/DS Assembler preprocessor to recognize host variables declared as even precision packed decimal.

Performance for SQL statements using even precision packed decimal columns may improve. Previously, if assembler programs contained even precision host variables, the preprocessor would convert these variables to odd precision. If these programs performed SQL against tables with even precision DECIMAL data, any predicates would be residual. With this support, the preprocessor will leave the host variables in even precision. Therefore, when these programs access even precision DECIMAL data, predicates will be sargable instead of residual.

2.2.4.2 Invocation

In previous versions of SQL/DS, an assembler program declared a packed decimal by specifying:

- A label
- The DC or DS opcode
- A parameter of the form PLn[’Decimal Constant’]

The precision was calculated by the formula 2n-1 which guaranteed that the packed decimal always had odd precision. With this support, the parameter for the DC or DS opcode may now be specified as:

```
P¢Decimal Constant¢
```

In this case, the precision is determined from the decimal constant specified between quotes. The precision is equal to the length of the decimal constant, excluding the decimal point and the sign. This new format allows for even precision packed decimal variables. Examples of this new format are given below:

```
VAR1 DS P¢123¢      This is decimal(3,0) and is odd precision
VAR2 DS P¢123.45¢   This is decimal(5,2) and is odd precision
VAR3 DS P¢1234¢     This is decimal(4,0) and is even precision
VAR4 DS P¢123.456¢  This is decimal(6,3) and is even precision
```

2.2.4.3 Usage Example

Consider the following sample DECTABLE with several decimal columns defined. There are both **odd** and **even** precision decimal columns in this table.

```
CREATE   TABLE      SQLDBA.DECTABLE
          (
           CDEC20           DECIMAL(2,0)    NOT NULL,
           CDEC21           DECIMAL(2,1)    NOT NULL,
           CDEC22           DECIMAL(2,2)    NOT NULL,
           CDEC30           DECIMAL(3,0)    NOT NULL,
           CDEC31           DECIMAL(3,1)    NOT NULL,
           CDEC32           DECIMAL(3,2)    NOT NULL,
           CDEC33           DECIMAL(3,3)    NOT NULL,
          )
       IN  PUBLIC.DSDECTABLE;
```

Figure 35. Sample DECTABLE Table

Previously, under SQL/DS Version 3 Release 4, even precision host variables were not supported by the Assembler preprocessor, as illustrated in Figure 36 on page 68.

```
*   EXEC SQL BEGIN DECLARE SECTION
*
PINT1    DC    P¢1¢              DECIMAL HOST (1,0) ¢1¢
*ARI0505I SQLCODE = -314 ROWCOUNT = 0
*
PINT12   DC    P¢12¢             DECIMAL HOST (2,0) ¢12¢
*ARI0505I SQLCODE = -314 ROWCOUNT = 0
*
PINT123  DC    P¢123¢            DECIMAL HOST (3,0) ¢123¢
*ARI0505I SQLCODE = -314 ROWCOUNT = 0
*
PDOT10   DC    P¢1.¢             DECIMAL HOST (1,0) ¢1.¢
*ARI0505I SQLCODE = -314 ROWCOUNT = 0
*
PDOT12   DC    P¢1.2¢            DECIMAL HOST (2,1) ¢1.2¢
*ARI0505I SQLCODE = -314 ROWCOUNT = 0
*
PDOT123  DC    P¢1.23¢           DECIMAL HOST (3,2) ¢1.23¢
*ARI0505I SQLCODE = -314 ROWCOUNT = 0
*
*   EXEC SQL END DECLARE SECTION
```

Figure 36. Not Supported Even Precision in V3R4

Because of this, the application programmers had to define their Assembler host variables by using decimal odd precision, as illustrated in Figure 37.

```
*   EXEC SQL BEGIN DECLARE SECTION
*
LHOS2    DS    PL2               DECIMAL HOST (3,0)
*
LINT1    DC    PL2¢1¢            DECIMAL HOST (3,0) ¢1¢
LINT12   DC    PL2¢12¢           DECIMAL HOST (3,0) ¢12¢
LINT123  DC    PL2¢123¢          DECIMAL HOST (3,0) ¢123¢
*
LDOT10   DC    PL2¢1.¢           DECIMAL HOST (3,0) ¢1.¢
LDOT12   DC    PL2¢1.2¢          DECIMAL HOST (3,1) ¢1.2¢
LDOT123  DC    PL2¢1.23¢         DECIMAL HOST (3,2) ¢1.23¢
*
CNAME    DS    CL30¢ ¢           CHAR(30) NAME
*
*   EXEC SQL END DECLARE SECTION
...
...
...
*          LVL         NAME                 SQL TYPE     LENGTH
*          --- -------------------------  ------------ --------
*ARI0561I    1 LHOS2                      DECIMAL           3,0
*ARI0561I    1 LINT1                      DECIMAL           3,0
*ARI0561I    1 LINT12                     DECIMAL           3,0
*ARI0561I    1 LINT123                    DECIMAL           3,0
*ARI0561I    1 LDOT10                     DECIMAL           3,0
*ARI0561I    1 LDOT12                     DECIMAL           3,1
*ARI0561I    1 LDOT123                    DECIMAL           3,2
```

Figure 37. Supported Odd Precision in V3R4

As a result, the predicates involving decimal even precision columns became residual predicates, as shown in Figure 38 on page 69.

```
** SELECT FROM DECTABLE **

   EXEC SQL   SELECT   CNAME
                INTO  :CNAME
                FROM   DECTABLE
               WHERE   CDEC20 = :LINT12


11/11/95   SELECT * FROM REFERENCE_TABLE   SQLVM340

QUERYNO PKGNAME  REFTYPE TNAME    CNAME    DBSSPRED
------- -------- ------- -------- -------- --------
      1 DECH340  SELECT
      1 DECH340  TABLE   DECTABLE
      1 DECH340  COLUMN  DECTABLE CDEC20   N
      1 DECH340  COLUMN  DECTABLE CNAME    N
```

Figure 38. REFERENCE_TABLE Results for V3R4

The consequence of having residual predicates is the use of inefficient access paths to the tables, as illustrated in Figure 39.

```
** SELECT FROM DECTABLE **

   EXEC SQL   SELECT   CNAME
                INTO  :CNAME
                FROM   DECTABLE
               WHERE   CDEC20 = :LINT12


11/11/95         SELECT * FROM PLAN_TABLE                    SQLVM340

QUERYNO PKGNAME  TNAME    ACCESSTYPE MATCHCOLS ACCESSNAME   INDEXONLY
------- -------- -------- ---------- --------- ------------ ---------
      1 DECH340  DECTABLE W                  0 DECTABLE_IX1 N
```

Figure 39. PLAN_TABLE Results for V3R4

Now, under SQL/DS Version 3 Release 5, both odd and even precision host variables are supported by the Assembler preprocessor, as illustrated in Figure 40 on page 70.

```
*   EXEC SQL BEGIN DECLARE SECTION
*
LHOS2    DS    PL2                DECIMAL HOST (3,0)
*
LINT1    DC    PL2¢1¢             DECIMAL HOST (3,0) ¢1¢
LINT12   DC    PL2¢12¢            DECIMAL HOST (3,0) ¢12¢
LINT123  DC    PL2¢123¢           DECIMAL HOST (3,0) ¢123¢
*
LDOT10   DC    PL2¢1.¢            DECIMAL HOST (3,0) ¢1.¢
LDOT12   DC    PL2¢1.2¢           DECIMAL HOST (3,1) ¢1.2¢
LDOT123  DC    PL2¢1.23¢          DECIMAL HOST (3,2) ¢1.23¢
*
PINT1    DC    P¢1¢               DECIMAL HOST (1,0) ¢1¢
PINT12   DC    P¢12¢              DECIMAL HOST (2,0) ¢12¢
PINT123  DC    P¢123¢             DECIMAL HOST (3,0) ¢123¢
*
PDOT10   DC    P¢1.¢              DECIMAL HOST (1,0) ¢1.¢
PDOT12   DC    P¢1.2¢             DECIMAL HOST (2,1) ¢1.2¢
PDOT123  DC    P¢1.23¢            DECIMAL HOST (3,2) ¢1.23¢
*
*   EXEC SQL END DECLARE SECTION
...
*          LVL         NAME                 SQL TYPE     LENGTH
*          --- -------------------------  ------------ --------
*ARI0561I    1 LHOS2                      DECIMAL           3,0
*ARI0561I    1 LINT1                      DECIMAL           3,0
*ARI0561I    1 LINT12                     DECIMAL           3,0
*ARI0561I    1 LINT123                    DECIMAL           3,0
*ARI0561I    1 LDOT10                     DECIMAL           3,0
*ARI0561I    1 LDOT12                     DECIMAL           3,1
*ARI0561I    1 LDOT123                    DECIMAL           3,2
*ARI0561I    1 PINT1                      DECIMAL           1,0
*ARI0561I    1 PINT12                     DECIMAL           2,0
*ARI0561I    1 PINT123                    DECIMAL           3,0
*ARI0561I    1 PDOT10                     DECIMAL           1,0
*ARI0561I    1 PDOT12                     DECIMAL           2,1
*ARI0561I    1 PDOT123                    DECIMAL           3,2
```

Figure 40. Supported Odd and Even Precision in V3R5

As a result, the predicates involving decimal even precision columns can become sargable predicates, as shown in Figure 41.

```
** SELECT FROM DECTABLE **

   EXEC SQL   SELECT   CNAME
                INTO  :CNAME
                FROM   DECTABLE
               WHERE   CDEC20 = :PINT12


11/11/95   SELECT * FROM REFERENCE_TABLE   SQLVM350

QUERYNO PKGNAME  REFTYPE TNAME    CNAME    DBSSPRED
------- -------- ------- -------- -------- --------
      1 DECH350  SELECT
      1 DECH350  TABLE   DECTABLE
      1 DECH350  COLUMN  DECTABLE CDEC20   Y
      1 DECH350  COLUMN  DECTABLE CNAME    N
```

Figure 41. REFERENCE_TABLE Results for V3R5

The consequence of having sargable predicates is the use of efficient access paths to the tables, as illustrated in Figure 42 on page 71.

```
** SELECT FROM DECTABLE **

   EXEC SQL   SELECT   CNAME
                INTO  :CNAME
                FROM   DECTABLE
               WHERE   CDEC20 = :PINT12


11/11/95         SELECT * FROM PLAN_TABLE                    SQLVM350

QUERYNO PKGNAME  TNAME    ACCESSTYPE MATCHCOLS ACCESSNAME   INDEXONLY
------- -------- -------- ---------- --------- ------------ ---------
      1 DECH350  DECTABLE I1                 1 DECTABLE_IX1 W
```

Figure 42. PLAN_TABLE Results for V3R5

2.3 Usability Enhancements

2.3.1 Enhanced Directory Expansion

2.3.1.1 Description

When a database is initially generated, a calculation is made to determine which portion of the directory will be set aside for the page map table, and which portion will be used for the allocation bitmaps. As the database is used and grows in size, the database may run short on either logical or physical space. In prior releases of SQL/DS, the directory could be expanded to accommodate more logical pages by expanding the page map table. This new SQL/DS V3R5 function adds support to use the additional directory space to increase the size of the allocation bitmaps and the page map table concurrently. If it is necessary to expand the directory to hold more dbextent pages, the only available option would be to choose this new support to expand the directory for both dbspace and dbextent pages.

2.3.1.2 Invocation and Interactions

In VM, to increase the size of the allocation bitmaps and the page map table concurrently, use the SQLCDBEX EXEC as follows:

```
SQLCDBEX DB(dbname)
```

Respond 1(Yes) to the following message:

```
ARI6146D  Are you expanding the SQL/DS directory?
          Enter 0(No), 1(Yes) or 111(Quit).
```

If the old and new directory disks are the same size, SQLCDBEX will simply copy the old directory to the new directory. No expansion will be done. If the new directory disk is larger than the old directory disk, respond 2 to the following message to increase the size of the allocation bitmaps and the page map table concurrently.

```
ARI6200D  You have requested to expand the directory.
          Enter 1 to expand the directory to hold more DBSPACE
          pages. Enter 2 to expand the directory to hold more
          DBSPACE and DBEXTENT pages. Enter 111 to quit.
          Enter 1, 2 or 111(Quit).
```

If both the allocation bitmaps and page map table are being expanded, SQLCDBEX will use the same algorithm as used in database generation to determine what portion of the expanded directory should be used for the allocation bitmaps and what portion should be used for the page map table. If the page map table had been previously expanded, the new directory must be large enough to hold the expanded allocation bitmaps and expanded page map table. SQLCDBEX will also determine if the new directory disk is large enough to hold the expanded allocation bitmaps and page map table. If the new directory disk is not large enough, message ARI6199E is displayed. Message ARI6199E follows:

```
ARI6199E  The new directory disk is not large enough
          to hold the previously expanded page map table
          and the new allocation bitmaps. nnnnnnnnn
          additional directory blocks of blocksize nnnn
          are required.
```

If the new directory disk is large enough, message ARI6198D will be displayed. This message will display the number of additional dbspace pages and dbextent pages that can be added to the newly expanded page map table and allocation bitmaps, respectively. The user has the option to quit the directory expansion at this point. Message ARI6198D follows:

```
ARI6198D  Current maximum DBEXTENT pages : nnnnnnnnn
          New maximum DBEXTENT pages :     nnnnnnnnn
          DBEXTENT pages added :           nnnnnnnnn
          Current maximum DBSPACE pages   :nnnnnnnnn
          New maximum DBSPACE pages       :nnnnnnnnn
          DBSPACE pages added             :nnnnnnnnn
          Current directory size          :nnnnnnnnn
          Current directory block size    :nnnnnnnnn
          New directory size              :nnnnnnnnn
          New directory block size        :nnnnnnnnn
          Do you wish to continue with expanding the
          directory to allow the directory to hold
          additional DBSPACE and DBEXTENT pages ?
          Enter 0(No), 1(Yes) or 111(Quit)
```

In VSE, to increase the size of the allocation bitmaps and page map table concurrently, run the ARIMEXBD program with the EXPAND=ALL parameter as follows:

```
// EXEC ARIMEXBD,SIZE=AUTO,PARM='DBPSWD=password,EXPAND=ALL'
```

The EXPAND parameter is added by this function to allow the user to specify whether the additional directory space should be used to expand the page map table (DBSPACE) or both the page map table and allocation bitmaps (ALL). The syntax for invoking ARIMEXBD is:

```
// EXEC ARIMEXBD,SIZE=AUTO,PARM='DBPSWD=password,EXPAND=DBSPACE|ALL'
```

If the EXPAND keyword is specified incorrectly, for example, EXAPND=ALL, an existing message, ARI0006E, will be displayed, as follows:

```
ARI0006E  JCL - EXAPND is an invalid parameter specification.
```

If the EXPAND keyword is specified correctly, but the parameter is not DBSPACE or ALL, for example, EXPAND=YES, an existing message, ARI0008E, will be displayed, as follows:

```
ARI0008E  JCL - EXPAND=YES is invalid. It must be DBSPACE or ALL.
```

If the EXPAND parameter is not specified, it will default to DBSPACE. Expanding the page map tables will allow additional dbspaces to be added.

If EXPAND=DBSPACE is specified, or if EXPAND defaults to DBSPACE, a new message, ARI0945I will be displayed, as follows:

```
ARI0945I  Expanded space will be used for DBSPACE pages.
```

Expanding the allocation bitmaps and page map table will allow additional dbextents and dbspaces to be added. If EXPAND=ALL is specified, ARIMEXBD will use the same algorithm as used in database generation to determine what portion of the expanded directory should be used for the allocation bitmaps and what portion should be used for the page map table. ARIMEXBD will determine if the new directory disk is large enough to hold expanded allocation bitmap and page map table. If the page map table had been previously expanded, the new directory must be large enough to hold the expanded allocation bitmaps and expanded page map table. If the new directory disk is not large enough, messages ARI0941E and ARI0940I are displayed. The messages ARI0941E and ARI0940I follow:

```
ARI0941E  The new directory disk is not large enough.
ARI0940I  Additional 512-byte directory blocks required: nnnnnnnnn
```

If EXPAND=ALL is specified and the new directory disk is large enough, a series of messages will be displayed. These messages display the number of additional dbspace pages and dbextent pages that can be added to the newly expanded page map table and allocation bitmaps, respectively. The messages are:

```
ARI0946I  Expanded space will be used for DBSPACE and DBEXTENT pages.
ARI0947I  Current maximum DBEXTENT pages : nnnnnnnnn
ARI0947I  New maximum DBEXTENT pages :     nnnnnnnnn
ARI0948I  DBEXTENT pages added :           nnnnnnnnn
ARI0947I  Current maximum DBSPACE pages :  nnnnnnnnn
ARI0947I  New maximum DBSPACE pages :      nnnnnnnnn
ARI0946I  DBSPACE pages added :            nnnnnnnnn
ARI0949I  Current directory size (blocks): nnnnnnnnn
ARI0949I  New directory size (blocks) :    nnnnnnnnn
```

When the directory has been successfully expanded, message ARI6201I in VM, or messages ARI0943I and ARI0924I in VSE, will be displayed. **It is strongly recommended that a database archive be taken immediately following the directory expansion.** Database archives taken before the directory expansion can be restored successfully. However, if a log archive references a section of the

directory that did not exist before the expansion, and therefore does not exist in the database archive that was restored, an attempt to apply this log archive will have unpredictable results, and could cause abnormal termination or corruption. Messages ARI6201I, ARI0943I and ARI0924I follow:

```
ARI6201I  Directory expansion completed successfully.
          It is strongly recommended that a database
          archive be taken immediately.
ARI0943I  Directory expansion completed successfully.
ARI0924I  It is recommended that a database archive be taken.
```

Note: An example of directory expansion can be found in Appendix D, "Database Directory Expansion Examples" on page 199.

2.3.2 C/370 Decimal Data Type Support

2.3.2.1 Description

The decimal data type was added to the C/370 compiler in the AD/Cycle C/370 Compiler Version 1 Release 2. This function allows C/370 applications to use variables declared with the new decimal data type as SQL/DS host variables.

This allows the previously unsupported SQL/DS DECIMAL type to be used. It also allows an alternative to the float or double type for storing decimal data in SQL/DS tables.

As an example, if tables had DECIMAL columns defined and decimal support was not available, then rows could not be inserted using decimal host variables. Literals could be used, as is shown in this example:

```
EXEC SQL CREATE TABLE TEST
(SPFLOAT    REAL ,
 DPFLOAT    DOUBLE PRECISION ,
 INT        INTEGER   NOT NULL,
 SINT       SMALLINT,
 DEC150     DECIMAL(15,0),
 DEC83      DECIMAL(8,3) ,
 CHAR1      CHAR(1),
 CHAR10     CHAR(10),
 VCHAR20    VARCHAR(20) ,
 LVCHAR     VARCHAR(200) ,
 DTE        DATE,
 TME        TIME,
 TSTAMP     TIMESTAMP );

EXEC SQL INSERT INTO TEST VALUES
(-5.40E-79, -7.20E+75, 1, -32768, -32768, -12345.678,
¢A¢, ¢TABLE 1¢, ¢ROW 01¢, NULL,
¢1988-01-01¢, ¢12.01.01¢, ¢1988-01-01-12.01.01¢);
```

With the decimal support, a host variable can be used to access the columns defined with the DECIMAL data type. To take advantage of decimal support, declare a host variable with the data type:

decimal(p,s)

where 'p' and 's' represent the precision and scale respectively. Notice that this declaration corresponds directly with the declaration of a table column with a data type of:

DECIMAL[**(p**[**,s**]**)**] **or DEC**[**(p**[**,s**]**)**]

The following example shows how the previous example has been modified to take advantage of decimal support.

```
EXEC SQL BEGIN DECLARE SECTION;

decimal(8,3) fixdec;

EXEC SQL END DECLARE SECTION;

EXEC SQL CREATE TABLE TEST
(SPFLOAT     REAL ,
 DPFLOAT     DOUBLE PRECISION ,
 INT         INTEGER NOT NULL,
 SINT        SMALLINT,
 DEC150      DECIMAL(15,0),
 DEC83       DECIMAL(8,3) ,
 CHAR1       CHAR(1),
 CHAR10      CHAR(10),
 VCHAR20     VARCHAR(20) ,
 LVCHAR      VARCHAR(200) ,
 DTE         DATE,
 TME         TIME,
 TSTAMP      TIMESTAMP );

fixdec = -12345.678;

EXEC SQL INSERT INTO TEST VALUES
(-5.40E-79, -7.20E+75, 1, -32768, -32768, :fixdec,
¢A¢, ¢TABLE 1¢, ¢ROW 01¢, NULL,
¢1988-01-01¢, ¢12.01.01¢, ¢1988-01-01-12.01.01¢);
```

The decimal data type is supported wherever any other C/370 data type is allowed, including structure declarations, and is subject to the current rules regarding the use of structures as host variables.

2.3.2.2 Invocation

This support is invoked when an SQL/DS host variable is declared with the decimal data type. The SQL/DS C language preprocessor is invoked and identifies the host variable as having the decimal data type.

2.3.2.3 Interactions

The host variables will be identified in the table at the end of the source listing.

2.3.2.4 Usage Notes

The LANGLVL(EXTENDED) compiler option is required to use the new decimal data type.

If the C/370 compiler is used instead of LE/370 then the compiler option TARGET(COMPAT) must be used.

2.3.3 CICS Information in SHOW CONNECT

2.3.3.1 Description

If a CICS user requests that the operator terminate a CICS transaction containing SQL/DS statements, the operator should first force the associated SQL/DS agent by using the FORCE command before terminating the transaction. Prior to SQL/DS V3R5, the operator may not be able to determine which agent to force, since multiple agents may use the same SQL/DS user ID. Furthermore, if the agent is not forced before the transaction is terminated, the resource adapter may encounter an error and shut itself down. This would cause the links to SQL/DS through that particular resource adapter to be lost. In SQL/DS V3R4, only the CICS task number representing the AXE transaction is displayed, and only for remote (DRDA) users.

Now, with SQL/DS V3R5, this functional change will help operators identify which agent should be forced by displaying the CICS task number, the CICS terminal ID, and the RMID for all local CICS users as part of the output for the SHOW CONNECT command. This information will be displayed for both VSE and VM (Guest Sharing) users.

The additional information on the SHOW CONNECT command will enable the operator to identify which agent should be forced. This is accomplished by the following steps:

1. The user tells the operator to cancel the task associated with a particular terminal ID. Alternatively, they may ask that a specific task be terminated.
2. The operator then issues the SHOW CONNECT command to determine which agent is associated with either the task ID or the terminal ID that was specified by the user.
3. The operator can then force the correct agent and then terminate the CICS transaction.

2.3.3.2 Invocation

This support is invoked when the SHOW CONNECT command is issued.

2.3.3.3 Interactions

The CICS task number, CICS terminal id, and RMID will be displayed for all local (VSE or VM Guest Sharing) CICS transactions whenever the agent is in work. The CICS terminal ID may contain a value of ′N/A′ indicating that the terminal ID is not available, such as when a user issues queries through ISQL. The CICS task number, the CICS terminal id, and the RMID will not be displayed for batch users, or for agents whose work status is NIW (not in work). For remote (DRDA) users, only the CICS task number representing the AXE transaction will be displayed; the CICS terminal id and the RMID will not be displayed.

Included below are sample outputs for each of the cases where additional information may be displayed.

SHOW CONNECT for CICS Transaction (Version 3 Release 5)

```
F4 004 User Agent: 1 User-ID: JOAO SQL-ID: JOAO
F4 004 is R/O APPL 12BCF
F4 004 Agent is processing and is in communication wait.
F4 004 State started: 1995-09-02 15:21:22
F4 004 Conversation started: 1995-09-02 15:21:22
F4 004 Task no.: 147 RMID: 32 Term. id: 077D
```

Figure 43. CICS Transaction (Version 3 Release 5 Requester)

SHOW CONNECT for ISQL Query

```
F4 004 User Agent: 1 User-ID: JOAO SQL-ID: JOAO
F4 004 is R/O APPL 12BCF
F4 004 Agent is processing and is in communication wait.
F4 004 State started: 1995-09-02 15:21:22

F4 004 Conversation started: 1995-09-02 15:21:22
F4 004 Task no.: 147 RMID: 32 Term. id: N/A
```

Figure 44. ISQL Query

SHOW CONNECT for Agent Not in Work

```
F4 004 User Agent: 2 User-ID: DBDCCICS SQL-ID: DBDCCICS
F4 004 is NIW SUBS

F4 004 Agent is not processing and is in communication wait.
F4 004 State started: 1995-09-03 15:19:57
F4 004 Conversation started: 1995-09-03 15:19:57
```

Figure 45. Agent Not in Work

SHOW CONNECT for Batch User

```
F4 004 User Agent: 2 User-ID: SQLDBA SQL-ID: SQLDBA
F4 004 is R/W APPL 18D9
F4 004 Agent is not processing and is in communication wait.
F4 004 State started: 1995-09-08 15:34:51
F4 004 Conversation started: 1995-09-08 15:34:41
```

Figure 46. Batch User

SHOW CONNECT for DRDA User

```
F4 004 User Agent: 2 User-ID: EDUARDA SQL-ID: EDUARDA
F4 004 is R/W APPL 12FC4
F4 004 Agent is processing and is in communication wait.
F4 004 State started: 1995-09-02 15:23:17
F4 004 Conversation started: 1995-09-02 15:23:15
F4 004 CPU time: 00:00:01
F4 004 LUWID: CAIBMOML.OECGW001.A6773D6F8611.0001
F4 004 EXTNAM: EDUARDA.1
F4 004 Requester: SQLDS/VM V3.5.0 at TOIVMLB6
F4 004 Rmtuser ID: 2
F4 004 LU name: OMPGW001
F4 004 Task no.: 0000134
```

Figure 47. DRDA User Accessing VSE Database

SHOW CONNECT for Guest Sharing

```
User Agent: 1 User-ID: VSEMCH10 SQL-ID: SQLDBA
is R/O SUBS 1796
Agent is not processing and is in communication wait.
State started: 1995-09-08 10:42:55
Conversation started: 1995-09-08 10:42:43
Task no.: 371 RMID: 12 Term. id: N/A
```

Figure 48. VSE Guest Sharing User to VM Database Using ISQL

SHOW CONNECT for VM User

```
User Agent: 1 User-ID: SQLUSRMR SQL-ID: SQLUSRMR
is R/O APPL 178F
Agent is not processing and is in communication wait.
State started: 1995-09-08 10:41:11
Conversation started: 1995-09-08 10:41:04
```

Figure 49. VM Requester Accessing VM Database

```
F4 004 User Agent: 1 User-ID: JOAO SQL-ID: JOAO
F4 004 is R/O APPL 12BCF
F4 004 Agent is processing and is in communication wait.

F4 004 State started: 1995-09-02 15:21:22
F4 004 Conversation started: 1995-09-02 15:21:22
F4 004 Task no.: 147 RMID: N/A Term. id: N/A
```

Figure 50. CICS Transaction (Version 3 Release 4 Requester)

2.3.4 Package Name in SHOW CONNECT

2.3.4.1 Description

This enhancement displays the package name and section number in response to a SHOW CONNECT request received from the operator console. Previously, this information was only displayed when the SHOW CONNECT command was issued from an application (for example, through ISQL or RXSQL), by the package creator or a requester with DBA authority.

2.3.5 ISQL Print Routing

2.3.5.1 Description

A new option **TOUser** has been added to the SET PRINTRoute command in SQL/DS V3R5. When the ISQL user specifies SET PRINTRoute TOUser xxxx and subsequently issues the PRINT command (without specifying TOUser) within the same ISQL session, ISQL will spool the print output in the same way that it will spool the print output when the user enters PRINT TOUser xxxx.

Previously, for VSE SQL/DS only, the PRINT command had the following four options from which the ISQL user could choose to specify where the ISQL print output is to be sent:

1. TOUser
2. TERMid
3. DESTid
4. SYStem

On the other hand, the SET PRINTRoute command only had the following three options from which the ISQL user could choose to specify where ISQL print output is to be sent:

1. TERMid
2. DESTid
3. SYStem

2.3.5.2 Invocation

The new option **TOUser** is mutually exclusive with the other options of the SET PRINTRoute command, namely: DESTid, TERMid, and SYStem.

The SET PRINTRoute command can be invoked with the TOUser option as follows:

```
SET PRINTRoute TOUser
                        userid
```

userid is the VSE POWER user identifier of the user to whom the output is being spooled. An identifier cannot be longer than eight alphameric characters. If the user ID is any number from 1 to 250, ISQL will spool the print output to the POWER remote workstation whose ID is the number specified.

Entering SET PRINTRoute TOUser with no user ID specified will cause ISQL to spool the print output to the ISQL user ID used at the time of logon.

If the option entered with the SET PRINTRoute TOUser command is not one to eight alphameric characters, ISQL will display this message:

ARI7756E SET command processing stopped. The TOUSER *touser* is not correct. TOUSER must be followed by a user identifier of 1 to 8 alphameric characters. Try again.

If the user ID is not entered and the ISQL user ID at the time of logon is not available, ISQL will display this message:

ARI7725E The SET PRINTROUTE command processing stopped. The command is not complete. You must specify a user identifier.

If the SET PRINTRoute TOUser command syntax is valid, ISQL will display this message:

ARI7727I The old PRINTROUTE value was *old*.
The new PRINTROUTE value is queued for *userid*.

2.3.5.3 LIST SET

If the ISQL user had previously entered a SET PRINTRoute command with the TOUser option, the LIST SET * or the LIST SET PRINTRoute command will display this message:

ARI7752I Your reports will be routed to the user identifier *userid*.

2.3.5.4 PRINT

If the ISQL user had previously entered a SET PRINTRoute command with the TOUser option, the ISQL PRINT command will display this message:

ARI7926I Your report has been queued for *userid*.

2.4 Reliability, Availability and Serviceability

2.4.1 Default DUMPTYPE = F

This enhancement changes the default DUMPTYPE value from “P” to “F.” This causes SQL/DS V3R5 to take full dumps by default instead of partial dumps. It increases the usefulness of the information that is generated if a problem occurs and the customer will not be required to try to reproduce a problem in order to obtain a full dump for IBM service.

This enhancement is available for both VM and VSE.

2.4.2 Support for LE/VSE in Single User Mode

2.4.2.1 Description

The TRAP runtime option is a powerful error condition handling facility of the LE/VSE product. Details can be found in the LE/VSE documentation. To enable its error condition handling LE/VSE establishes STXIT AB and STXIT PC. These exits will replace any STXIT settings previously established by SQL/DS.

Note: Currently SQL/DS only establishes STXIT AB exits. It does not establish STXIT PC exits.

Prior to SQL/DS V3R5, the SQL/DS error condition handling assumed that neither the application nor any associated compiler runtime code could set its own STXITs and thereby overwrite the STXITs established by SQL/DS. This assumption could not be maintained when allowing for the new LE/VSE TRAP option. When TRAP is active it needs control over condition handling. This made it necessary to coordinate condition handling between LE/VSE and SQL/DS and adapt the existing condition handling logic of SQL/DS to the new situation.

The new SQL/DS V3R5 condition handling described below only applies to the SQL/DS Single User Mode (SUM) environment. This support is not required in the following instances:

- When an existing application is executed in a non-LE/VSE environment.
- When an application is executed in an LE/VSE environment, TRAP(ON) is specified and the application is running in SQL/DS Multiple User Mode (MUM).
- When an application is executed in an LE/VSE environment and TRAP(OFF) is specified.

Note: SQL/DS and DL/I are currently the only products with which LE/VSE must coordinate condition handling. This manual only describes how LE/VSE coordinates condition handling with SQL/DS.

2.4.2.2 Invocation

This support is not intended to be invoked directly by users. Rather, the LE/VSE product will invoke this support.

2.4.2.3 Initialization

When a user starts an SQL/DS application in SUM, SQL/DS will first get control and initialize the database. The SQL/DS initialization modules will set STXIT AB as before. When initialization is complete, SQL/DS passes control to the user application program. When the user application program gets control, LE/VSE initialization will take place. During LE/VSE initialization, if the user has specified TRAP(ON), LE/VSE will save the SQL/DS STXIT information and issue its own STXIT AB and STXIT PC.

2.4.2.4 Condition Handling

If a program interrupt or abend occurs, the appropriate LE/VSE exit will receive control. LE/VSE will then determine whether or not the error occurred in DL/I. If LE/VSE determines that DL/I was **not** in control, it will then determine whether or not SQL/DS was in control. For that purpose LE/VSE will CDLOAD and invoke the new SQL/DS routine ARICABC. ARICABC will return whether SQL/DS was in control by checking SQL/DS owned flags and comparing the location of where the abend occurred with the locations of SQL/DS components.

Using the above method LE/VSE is able to determine who was in control at the time of the error. If it finds out that DL/I was in control DL/I abend processing occurs. If it finds out that SQL/DS was in control processing continues as described in 2.4.2.5, “Error in SQL/DS.” If the application was in control processing continues as described in 2.4.2.6, “Error in Application Program.”

2.4.2.5 Error in SQL/DS

If the interrupt condition occurred while in SQL/DS code, LE/VSE retrieves the exit information from the SQL/DS STXIT and transfers control to the SQL/DS abend handler. This allows SQL/DS to process the error condition in the same way as it would have done without LE/VSE interference.

2.4.2.6 Error in Application Program

If the interrupt condition occurred while application code was active LE/VSE passes control to the application program. It may then decide to correct the error and continue processing through to normal termination - including normal SQL/DS termination.

If the error cannot be resolved, then one of the following methods should be used by the user to ensure that the SQL/DS AB exit receives control when LE/VSE terminates the application:

- Specify the run-time option ABTERMENC(ABEND) which will instruct LE/VSE to abnormally terminate when there is a condition that is not handled by the user application program.
- Provide a modified run-time assembler user exit (CEEBXITA) that transforms all abnormal terminations into operating system abends. The assembler user exit should check the return code and reason code or the CEEAUE_ABTERM bit, and request an abend by setting the CEEAUE_ABND flag to ON, if appropriate. See LE/VSE documentation for more details.

If one of the above options is used, LE/VSE will do the following:

- Issue a WTO message with the original cancel and interruption code, if applicable.
- Disable its own STXITs and re-enable those of SQL/DS, using the save area originally set up by SQL/DS.

- Issue an SVC 50, which terminates the task and drives SQL/DS's STXIT AB.

After the SVC 50 instruction has been executed, storage contents, register values and the cancel code will probably not be the same as at the time when the original interruption occurred. Original data is delivered through the cancel code displayed in the WTO message or by re-executing the program with TRAP(OFF).

Note: It is important that either ABTERMENC(ABEND) or CEEBXITA be specified. Should an application error remain unresolved, only these will allow LE/VSE to return control to SQL/DS, allowing SQL/DS to do its necessary cleanup.

2.5 Standards Support

2.5.1 SQLSTATE Changes for SQL92

2.5.1.1 Description

An SQLSTATE is a five character value that is returned to the application from the Database Manager each time an SQL statement is executed, to indicate the outcome of the SQL statement. SQLSTATE values are designed so that application programs can test for specific conditions or classes of conditions. The values are comprised of a two-character class code value, followed by a three-character subclass code value. Class code values represent classes of successful (class code 00 and 01) and unsuccessful (all other class codes) execution conditions. In SQL/DS, SQLSTATE (SQLSTT in FORTRAN and RPG) is a character string variable in the SQLCA.

The SQLCA also contains an SQLCODE which is used in SQL to indicate the success or failure of the statement. However, SQLCODE values (with a very few exceptions) have not been standardized across database products.

The SQLSTATE values have been standardized in SQL92. If the SQLSTATE values of all databases comply with SQL92, customers who are writing applications which are either portable across SQL implementations or have to connect to more than one SQL implementation can thus rely on a consistent SQLSTATE being returned to report the same diagnostic condition.

IBM standardized diagnostic reporting across its various databases by complying with the SQLSTATE mechanism proposed in the ISO/ANSI SQL2 draft a few years ago. Unfortunately, the SQLSTATE values in the final published version of SQL92 deviate in many ways from the scheme that they were planning to follow a few years ago. As a result, SQL/DS V3R3 and SQL/DS V3R4 contain a number of SQLSTATEs that are different from the SQL92 standard. Now, in SQL/DS V3R5, these SQLSTATEs comply with SQL92.

DB2 for MVS (Version 4.1 and later), DB2 for OS/400 (Version 3.6 and later), and DB2 Common Server (Version 2.1 and later) all comply with the new SQL92 SQLSTATE values.

There are also a number of SQL/DS specific SQLSTATEs (not supported on other platforms) which are not addressed by SQL92. These SQLSTATEs were not removed since it would create incompatibilities for no benefit.

2.5.1.2 Invocation and Interaction

In releases prior to SQL/DS V3R5, an SQLSTATE value was returned with its corresponding SQLCODE in the SQLCA each time an SQL statement was executed. If warnings or errors were detected, the code set both the SQLCODE and the SQLSTATE value. In SQL/DS V3R5, many SQLSTATE values returned to applications for warnings and errors will be different from prior releases of SQL/DS.

Refer to *SQL/DS V3R5 Messages and Codes for VM* and *SQL/DS V3R5 Messages and Codes for VSE* manuals for detail on the SQLSTATE changes.

2.6 New Product Feature

2.6.1 SQL/DS Data Restore Version 3 Release 5

The new Data Restore feature provides the facilities to back up and restore a database machine in case of a DASD or system failure. It also provides capabilities to recover single tables in the event of application errors. ′Point in time′, or forward log recovery, is possible with this feature, using the SQL/DS log files.

The Data Restore feature allows an individual responsible for the management and maintenance of SQL/DS databases to prepare for and be able to recover from such problems.

Most of the Data Restore feature functions work on physical pages. For this reason, when the database manager is defined in a VM environment, all functions will be processed in VM, even if the VSE Guest Sharing feature is used; and all functions will be processed in VSE, when the database is defined in VSE. This also allows many of the Data Restore commands to work regardless of whether the database is online or offline.

Data Restore runs on all supported releases of VSE/ESA and VM/ESA, and requires SQL/DS V3R5.

For a complete and detailed explanation of this feature, refer to the *IBM SQL/DS Data Restore Guide*. Refer to Chapter 4, “Archive and Recovery Strategies” on page 95 for a discussion of archive and recovery strategies and usage examples.

Chapter 3. Performance Benefits of SQL/DS V3R5

The main objective of this chapter is to provide a description of the performance enhancements of SQL/DS V3R5 compared to SQL/DS V3R4.

3.1 Increased Buffer Maximums

3.1.1 Real Storage

Sufficient real storage must be available to back up the virtual storage allocated to the buffers, to prevent an increase in the paging of the database manager. It should be noted that a page fault halts execution of ***all*** database activity.

3.1.2 System Checkpoints

Between checkpoints, modified pages in the buffers are not written back to DASD unless the buffer is needed for a different page. The modified buffer is first written out and then a different page is read in. With a very large increase in the number of buffers, a larger number of modified pages can accumulate between checkpoints. In the extreme case, ***no*** modified buffers may have been written to DASD between two checkpoints.

Modified buffers ***have*** to be flushed to DASD during system checkpoint processing. A very large number of buffers could increase the time for a checkpoint to complete, if a large percentage of the buffers contain modified pages.

Increasing the number of buffers will not increase the actual number of pages that must be written to DASD. Indeed, in some cases, it may actually decrease the number of I/Os required. Consider when a specific page is updated twice with a short period of time between the updates and:

- with a small number of buffers, the page may be forced out of the buffer between the two updates, requiring it to be read in for the second update and written out a second time;
- with a large number of buffers, the page may be able to stay in the buffers for both updates, and only be written out once.

However, a larger number of buffers may cause many or most writes to be delayed until the next checkpoint, causing an increase in the elapsed time for the checkpoint, due to the larger number of writes that must occur while all users are waiting. While the users′ work will occur slightly faster (as there is less I/O wait time when fewer buffers are written out between checkpoints), this apparent performance improvement is very unlikely to be noticeable to most users; on the other hand, many users may notice the increase in checkpoint elapsed time.

Naturally, it is hoped that the general performance improvement given by the increased number of buffers will outweigh the change in checkpoint behavior. This problem may be somewhat offset by decreasing the checkpoint interval; however, this increase in the frequency of checkpoints is likely to degrade the overall performance benefit provided by the increased number of buffers. This is because checkpoints have a fixed overhead cost, and more frequent checkpoints will generate more overhead.

Again, it may be that the general performance improvement given by the increased number of buffers will outweigh this increase in checkpoint overhead.

3.1.3 Page versus Directory Buffers

Accessing a data page frequently requires accessing a directory block as well. A large increase in the number of page buffers without some increase in the number of directory buffers may cause a decrease in the directory buffer hit ratio, indicating that the directory buffers have become a bottleneck.

3.1.4 Usage Notes

Customers will need to measure buffer hit ratios to tune the number of both page and directory buffers.

3.2 Archive Performance Improvement

In SQL/DS V3R5, enhancements were made to improve the performance of the database manager and log archiving processes by introducing the following changes:

- Archive only allocated pages. The database manager archive process will continue to examine each cell, however it will only archive pages that have been allocated. Pages that have not been allocated will not be archived.
- Use Multiple-Block *BLOCKIO requests in a single IUCV SEND request. Using Multiple-Block *BLOCKIO, seven pages are chained together to be read with a single IUCV request. The data is stored in the user buffer of 28K. This implementation is only possible for VM. Multiple-Block *BLOCKIO is not available for VSE.
- Use asynchronous I/O to reduce the wait time for DASD read by having concurrent disk and tape operations. This involves performing a complete read operation from disk into one buffer, beginning a second read operation from disk into a second buffer, then writing to tape from the first buffer while the second read operation is still proceeding. This will continue until the archive has completed. The asynchronous I/O is only implemented in VM because it is not compatible with VSAM controlled buffers in VSE.
- Use VSAM controlled buffers in VSE. By using VSAM controlled buffers and sequential processing, VSAM is able to read multiple records with a single I/O request. The number of records will depend on the number of buffers available to VSAM, which will be specified when creating the ACB for the directory, data and log disks. A second set of ACBs is required in order to avoid interfering with normal I/O processing during online archives. Implementing VSAM controlled buffers will also require using SHAREOPTIONS(2) for the directory, data and log disks.

The performance improvement for this support will depend on:

- the amount of data to be archived
- the number of VSAM buffers available (for VSE only)
- for database manager archives, the number of allocated and unallocated pages in the database

3.2.1 Test Environment

Four databases were defined for our tests:

- SQLVM340 (SQL/DS V3R4 under VM)
- SQLVM350 (SQL/DS V3R5 under VM)
- SQLVS340 (SQL/DS V3R4 under VSE)
- SQLVS350 (SQL/DS V3R5 under VSE)

Note: The VM CP IND USER command was used to obtain the following measurements from the VM database and VSE machines. The VSE machine under VM was running only with the SQL/DS database partition active.

The databases were defined with four storage pools each and with the following structure:

3.2.1.1 SQL/DS V3R4 Database under VM/ESA

Table 10. SQL/DS V3R4 Database under VM/ESA

Storage Pool	Number of dbextents	Total Number of Pages	Number of Pages Used	% of Space Used
1	2	14,934	1,795	12%
2	3	22,401	14,511	64%
3	3	22,401	14,511	64%
4	3	22,401	14,511	64%
TOTAL	11	82,137	45,328	55%

Note: For the database defined in Table 10 we were using VM data spaces for all storage pools and striping to balance the data among the dbextents.

3.2.1.2 SQL/DS V3R5 Database under VM/ESA

Table 11. SQL/DS V3R5 Database under VM/ESA

Storage Pool	Number of dbextents	Total Number of Pages	Number of Pages Used	% of Space Used
1	2	14,934	1,884	12%
2	3	22,401	14,513	64%
3	3	22,401	14,515	64%
4	3	22,401	14,511	64%
TOTAL	11	82,137	45,423	55%

Note: For the database defined in Table 11 we were using VM data spaces for all storage pools and striping to balance the data among the dbextents.

3.2.1.3 SQL/DS V3R4 Database under VSE/ESA

Table 12 (Page 1 of 2). SQL/DS V3R4 Database under VSE/ESA

Storage Pool	Number of dbextents	Total Number of Pages	Number of Pages Used	% of Space Used
1	2	14,934	1,514	10%
2	3	22,401	14,511	64%
3	3	22,401	14,511	64%

Table 12 (Page 2 of 2). SQL/DS V3R4 Database under VSE/ESA				
Storage Pool	**Number of dbextents**	**Total Number of Pages**	**Number of Pages Used**	**% of Space Used**
4	3	22,401	14,513	64%
TOTAL	11	82,137	45,049	55%

3.2.1.4 SQL/DS V3R5 Database under VSE/ESA

Table 13. SQL/DS V3R5 Database under VSE/ESA				
Storage Pool	**Number of dbextents**	**Total Number of Pages**	**Number of Pages Used**	**% of Space Used**
1	2	14,934	1,507	10%
2	3	22,401	14,511	64%
3	3	22,401	14,511	64%
4	3	22,401	14,511	64%
TOTAL	11	82,137	45,040	55%

3.2.2 Test Results

The results of the tests were obtained in a controlled environment. They may vary depending on the CPU usage, concurrency and external factors. The results were the following:

3.2.2.1 Results for SQLVM340 Database

```
#CP IND USER SQLVM340
USERID=SQLVM340 MACH=XC  STOR=0024M VIRT=V XSTORE=NONE
IPLSYS=CMS      DEVNUM=00026
PAGES: RES=001809 WS=001809 LOCK=000000 RESVD=000000
NPREF=000000 PREF=000000 READS=000119 WRITES=000002
CPU 00: CTIME=00:32 VTIME=000:01 TTIME=000:02 IO=001147
        RDR=000000 PRT=000121 PCH=000000
```

Figure 51. CP IND USER SQLVM340 Before Archive

```
#CP IND USER SQLVM340
USERID=SQLVM340 MACH=XC  STOR=0024M VIRT=V XSTORE=NONE
IPLSYS=CMS      DEVNUM=00013
PAGES: RES=000222 WS=000187 LOCK=000000 RESVD=006000
NPREF=000000 PREF=000000 READS=000000 WRITES=000007
CPU 00: CTIME=00:55 VTIME=000:38 TTIME=002:05 IO=144700
        RDR=000000 PRT=000157 PCH=000000
```

Figure 52. CP IND USER SQLVM340 After Archive

Table 14 (Page 1 of 2). SQL/DS V3R4 Archive under VM/ESA			
Data from #CP IND USER	**Before Archive**	**After Archive**	**Difference**
CTIME	00:32	00:55	00:23
VTIME	00:01	00:38	00:37
TTIME	00:02	02:05	02:03

Table 14 (Page 2 of 2). SQL/DS V3R4 Archive under VM/ESA			
Data from #CP IND USER	**Before Archive**	**After Archive**	**Difference**
I/O	1147	144,700	143,553
Database	**Archive Start Time**	**Archive Stop Time**	**Archive Total Time**
SQLVM340	11:37:59	12:00:46	22min 47sec

3.2.2.2 Results for SQLVM350 Database

```
#CP IND USER SQLVM350
USERID=SQLVM350 MACH=XC  STOR=0024M VIRT=V XSTORE=NONE
IPLSYS=CMS       DEVNUM=00026
PAGES: RES=001613 WS=001612 LOCK=000000 RESVD=006000
NPREF=000000 PREF=000000 READS=000131 WRITES=000012
CPU 00: CTIME=01:04 VTIME=000:02 TTIME=000:04 IO=002058
        RDR=000000 PRT=000263 PCH=000000
```

Figure 53. CP IND USER SQLVM350 Before Archive

```
#CP IND USER SQLVM350
USERID=SQLVM350 MACH=XC  STOR=0024M VIRT=V XSTORE=NONE
IPLSYS=CMS       DEVNUM=00013
PAGES: RES=000214 WS=000178 LOCK=000000 RESVD=006000
NPREF=000000 PREF=000000 READS=000000 WRITES=000015
CPU 00: CTIME=01:08 VTIME=000:11 TTIME=000:25 IO=017105
        RDR=000000 PRT=000296 PCH=000000
```

Figure 54. CP IND USER SQLVM350 After Archive

Table 15. SQL/DS V3R5 Archive under VM/ESA			
Data from #CP IND USER	**Before Archive**	**After Archive**	**Difference**
CTIME	01:04	01:08	00:04
VTIME	00:02	00:11	00:09
TTIME	00:04	00:25	00:21
I/O	2,058	17,105	15,047
Database	**Archive Start Time**	**Archive Stop Time**	**Archive Total Time**
SQLVM350	12:10:45	12:14:48	4min 03sec
Note: Performance improvement was **5.6 times better**. The performance improvement was measured from the results of 'Archive Total Time' obtained between SQL/DS V3R4 and SQL/DS V3R5			

3.2.2.3 Results for SQLVS340 Database

```
#CP IND USER SQLVS340
USERID=SQLVS340 MACH=ESA STOR=0048M VIRT=V XSTORE=NONE
IPLSYS=DEV 0E00 DEVNUM=00039
PAGES: RES=011588 WS=011567 LOCK=000000 RESVD=012000
NPREF=000000 PREF=000000 READS=000001 WRITES=000015
CPU 00: CTIME=05:53 VTIME=001:20 TTIME=001:42 IO=029797
        RDR=000000 PRT=000276 PCH=000000
```

Figure 55. CP IND USER SQLVS340 Before Archive

```
#CP IND USER SQLVS340
USERID=SQLVS340 MACH=ESA STOR=0048M VIRT=V XSTORE=NONE
IPLSYS=DEV 0E00 DEVNUM=00039
PAGES: RES=006390 WS=006390 LOCK=000000 RESVD=012000
NPREF=000000 PREF=000000 READS=000001 WRITES=000018
CPU 00: CTIME=06:12 VTIME=002:45 TTIME=003:46 IO=120371
        RDR=000000 PRT=000350 PCH=000000
```

Figure 56. CP IND USER SQLVS340 After Archive

Table 16. SQL/DS V3R4 Archive under VSE/ESA

Data from #CP IND USER	Before Archive	After Archive	Difference
CTIME	05:53	06:12	00:19
VTIME	01:20	02:45	01:25
TTIME	01:42	03:46	02:04
I/O	29,797	120,371	90574
Database	**Archive Start Time**	**Archive Stop Time**	**Archive Total Time**
SQLVS340	19:35:30	19:53:57	18min 27sec

3.2.2.4 Results for SQLVS350 Database

```
#CP IND USER SQLVS350
USERID=SQLVS350 MACH=ESA STOR=0048M VIRT=V XSTORE=NONE
IPLSYS=DEV 0E00 DEVNUM=00039
PAGES: RES=007957 WS=007957 LOCK=000000 RESVD=012000
NPREF=000000 PREF=000000 READS=000001 WRITES=000021
CPU 00: CTIME=06:17 VTIME=002:56 TTIME=004:04 IO=130403
        RDR=000000 PRT=000418 PCH=000000
```

Figure 57. CP IND USER SQLVS350 Before Archive

```
#CP IND USER SQLVS350
USERID=SQLVS350 MACH=ESA STOR=0048M VIRT=V XSTORE=NONE
IPLSYS=DEV 0E00 DEVNUM=00039
PAGES: RES=006339 WS=006129 LOCK=000003 RESVD=012000
NPREF=000000 PREF=000000 READS=000001 WRITES=000024
CPU 00: CTIME=06:20 VTIME=003:32 TTIME=004:50 IO=140210
        RDR=000000 PRT=000492 PCH=000000
```

Figure 58. CP IND USER SQLVS350 After Archive

Table 17. SQL/DS V3R5 Archive under VSE/ESA

Data from #CP IND USER	Before Archive	After Archive	Difference
CTIME	06:17	06:20	00:03
VTIME	02:56	03:32	00:36
TTIME	04:04	04:50	00:46
I/O	130,403	140,210	9,807
Database	**Archive Start Time**	**Archive Stop Time**	**Archive Total Time**
SQLVS350	19:59:57	20:03:11	3min 14sec

Note: Performance improvement was **5.7 times better**. The performance improvement was measured from the results of 'Archive Total Time' obtained between SQL/DS V3R4 and SQL/DS V3R5

3.3 Assembler Even Precision Packed Decimal Support

Performance of SQL statements using even precision packed decimal column may improve. Previously if assembler programs contained even precision host variables, the preprocessor would convert these variables to odd precision. If these programs performed SQL against tables with even precision DECIMAL data, any predicates would be residual. With this support the preprocessor leaves the host variables in even precision. Therefore when these programs access even precision DECIMAL data, predicates become sargable instead of RESIDUAL. Refer to the *SQL/DS V3R4 Performance Guide* for a discussion of sargable and residual predicates.

3.4 Virtual Disks for VM and VSE

Virtual disks are implemented in data spaces and require an ESA processor. Performance improvement is gained through the reduction of I/Os to disk, as a virtual disk is in memory. In SQL/DS internal dbspaces can be implemented on virtual disks. This results in improvements in sorts and joins, and index creation. Virtual disk characteristics are:

- Both VSE and VM virtual disk can significantly reduce elapsed times by avoiding physical I/Os.
- Both can be used on ES9000 and ESA/370 processors.
- Any data stored on virtual disk must be considered volatile, and will be lost if there is a system failure or an IPL is performed.
- Virtual disk should be used only for temporary files, such as internal dbspaces of SQL/DS or VSE lock files.

Use a VSE virtual disk if:

- You do not need any support provided by VM for a VSE guest.

Use a VM virtual disk if:

- The files are shared across VSE guests, for example, the VSE lock file
- Any virtual disk must survive VSE IPLs (it will not survive a VM IPL)

- This is the only way to exploit available expanded storage (via VM paging for V=V guests)
- A version/release of VSE/ESA 1.3 and above is used
- The VM/ESA or ESA supervisor mode in VSE/ESA is used.

For more information about this subject, please read the topic 2.2.3, “Utilizing Virtual Disk for VM and VSE” on page 55.

3.4.1 Test Results

Four indexes were created on a 100,000 row table in Single User Mode. Each row of the table is 410 bytes long.

Measurements were performed on a 9121-480 running VSE/ESA 1.3 second level with a 62MB V=R area under VM/ESA 1.2.0 using SQL/DS V3R5. The base measurement was obtained using physical disk for internal dbspaces and a comparable measurement was obtained with internal dbspaces on a 20M virtual disk.

The following non-default startup parameters were used:

```
NPAGBUF=1000, NDIRBUF=1000,CHKINTVL=1000
```

The results presented below represent the average of three runs.

SQL/DS Internal Dbspaces in:	1) Physical Disk	2) Virtual Disk	% Improvement
80 Byte Index			
Elapsed Time (sec)	485	288	40.6%
Total CPU Time (sec)	32.36	38.52	-19.0%
Disk I/O (count)	30939	17871	42.2%
60 Byte Index			
Elapsed Time (sec)	423	274	35.2%
Total CPU Time (sec)	27.20	31.92	-17.4%
Disk I/O (count)	27314	17268	36.8%
40 Byte Index			
Elapsed Time (sec)	363	263	27.5%
Total CPU Time (sec)	24.45	27.83	-13.8%
Disk I/O (count)	23687	16702	29.5%
20 Byte Index			
Elapsed Time (sec)	300	253	15.7%
Total CPU Time (sec)	21.92	23.68	-8.0%
Disk I/O (count)	19801	16135	18.5%

Table 18. SUM Create Index Results for Internal Dbspaces in Physical and Virtual Disk

A 40.6% improvement in elapsed time was observed using virtual disks when creating an 80-byte index. This is due to the reduction of I/Os of about 42.2%. However, a 19% additional CPU overhead was also observed. This CPU overhead is a result of additional code to move data to and from data spaces instead of handing it over to the I/O subsystem.

The percentage improvement varies with the size of the index being created. As the sort size gets bigger, more pages will not fit in the page buffers and have to be moved back and forth from disk. With virtual disks and enough real storage to avoid paging problems, these I/Os are avoided, reducing elapsed time.

Chapter 4. Archive and Recovery Strategies

This chapter briefly covers the concepts of the new Data Restore feature. In addition, SQL/DS archive features and database utility functions are discussed in relation to backup, recovery and archiving strategies.

This chapter concentrates on archive and recovery strategies but note that these recommendations are just guidelines for different situations, and each installation should analyze their own requirements.

4.1 Terminology

It is necessary to differentiate the concepts of archiving and recovery, with the commands available in the new SQL/DS Data Restore Version 3 Release 5 feature. The concepts are defined here:

Archive A copy of the database at a given time. It is used as a backup and to recover the database if a failure occurs.

Recovery It is the process of recovering a database or portion of a database, after a system failure.

Backup The physical copy produced by the archive.

In the new Data Restore feature, restore and backup are commands. To differentiate between the concepts here explained and the commands explained later in this chapter, when we talk about the commands, they will appear in uppercase.

4.2 Strategies

A variety of problems can occur in a relational database management system, leading to inaccuracies or loss of data. A power failure can bring the computer to a halt; the disk used to store information could become damaged; users can make errors such as dropping the wrong table or dbspace. Database recovery refers to the processing needed to correct the data when something goes wrong.

The problems that can occur fall into four categories:

1. Application Failures

   A single application fails to complete successfully.

2. User Logic Errors

   The system or application does the requested function, but the request itself was in error; that is, the user or application program did not specify the correct function.

3. System Failure

   The operating systems or SQL/DS database manager ends abnormally because of error conditions or power failure.

4. DASD Failure

   The system may be unable to read data from or write it to the DASD device on which it is stored because the storage medium is unreadable or damaged. Such an error could occur on any of the database minidisks.

With the SQL/DS Data Restore Version 3 Release 5 feature, now there are many options available with SQL/DS to manage the backup and recovery process for an SQL/DS database. You must select an archiving and recovery strategy for each database that best meets the needs and requirements of your environment.

Before you begin, you will need to consider the following:

- Logmode to be used by each SQL/DS database.
- Database availability - archive to be taken online (database up) or offline (database down).
- Archive medium (tape or disk) to hold log archive and full database archive.
- Number of log archives, when LOGMODE=L, to be taken between a full database archive.
- Acceptable time for recovery for the different data in your database.
- DBA workload to perform tasks associated with archiving which require manual intervention.

4.3 Data Restore Feature

In the feature, the following commands are available:

BACKUP Copies the entire database to tape or DASD, or both.

RESTORE Recovers an entire database.

UNLOAD Backs up selected dbspaces.

RELOAD Loads tables from an UNLOAD or a BACKUP.

DESCRIBE Lists all tables that are available for RELOADing from a BACKUP or UNLOAD.

LISTLOG Lists all transactions against tables from the log.

APPLYLOG Applies all the transactions against tables from the log up to a particular timestamp.

TRANSLATE Converts an archive tape from the format created by the SQL/DS ARCHIVE operator command to the format created by the Data Restore feature BACKUP command.

FORMAT Formats a dbextent before using the RESTORE command (VSE only).

SELECT Produces reports from an SQL/DS database when it is down.

These commands can be used in different situations, to recover or back up single tables or the full database. For a detailed explanation of the SQL/DS Data Restore Version 3 Release 5 feature and how to use the commands, refer to the *IBM SQL/DS Data Restore Guide*.

4.4 Full Database Backup and Restore Strategies

If there is a system or DASD failure on one of your database devices, the database must be recovered. The physical device will need to be replaced, and then the database restored from an archive. There are several different archive options available for full database backup and recovery:

- Data Restore feature BACKUP command

- Data Restore feature RESTORE command
- SQL/DS database archive
- SQL/DS database restore
- SQL/DS log archive
- SQL/DS user archive

4.4.1 Data Restore Feature BACKUP Command

The BACKUP command creates a copy of the entire database. The database must be down for this copy which includes the directory and dbextents. It does not include the log. The BACKUP command is performed as an SQL/DS user archive.

The SQL/DS database must be running under LOGMODE=L or LOGMODE=A for 'point in time' recovery to be performed. Point in time recovery processes the transactions stored in successive logs. Data Restore feature RELOAD and RESTORE commands can utilize this copy.

Advantages of using Data Restore feature BACKUP command:

- BACKUP files can be defined to tape or DASD.
- Dual BACKUPs can be taken in one process.
- It can be used to restore tables individually.

Disadvantages:

- The database must be down during the BACKUP

For example, to take a Dual BACKUP, the following SYSIN should be used.

```
OPTIONS DEVICE2=DASD CONFIRM=NO
CONTROL DBNAME=SQLDBA
BACKUP
```

Figure 59. BACKUP SYSIN File for Dual BACKUP Command

Note the DEVICE2=DASD option will cause the second backup to be taken on DASD. Therefore, the BACKUP EXEC must be updated to include the ARCHIV2 FILEDEF associated with the DASD, as shown in Figure 60.

```
/**/
¢SET EMSG ON¢
¢FILEDEF ARCHIV  TAP1 SL 1              (RECFM VB BLOCK 32760¢
¢FILEDEF ARCHIV2 DISK SQLDBA BACKUP A   (RECFM VB BLOCK 32760¢
¢FILEDEF SYSIN DISK BACKUP   SYSIN  A¢
¢FILEDEF SYSPRINT DISK BACKUP SYSPRINT A¢
¢XTS91001¢
Exit rc
```

Figure 60. BACKUP EXEC for Dual BACKUP Command

4.4.2 Data Restore Feature RESTORE Command

The RESTORE command can be used to recover a database after a system or DASD failure. It requires a backup created by the SQL/DS Data Restore Version 3 Release 5 feature BACKUP command.

An SQL/DS database archive can be TRANSLATEd into the BACKUP format using the Data Restore feature TRANSLATE command. Then RESTORE can be used to recover the directory and dbextents.

It may be necessary to apply log archives after the RESTORE is complete.

4.4.3 SQL/DS Database Archive

An SQL/DS database archive is a copy of the entire database. It is processed using SQL/DS database facilities.

With SQL/DS V3R5, the time required for this process has been reduced significantly. The database can remain up while processing the archive. The copy of the database that is created, can be used for recovery.

This database archive does not include the log. To restore a database from a database archive, the current log and any log archives must also be available.

The main advantages are:

- Database can remain online when the database archive is running
- Time required for the archive has been reduced significantly

The disadvantages are:

- Cannot exploit particular DASD characteristics for improved performance
- If a checkpoint is required during the online archive, it must wait until the online archive has completed. This will require all users to also wait.

Refer to 3.2, “Archive Performance Improvement” on page 88 for a discussion on the performance improvement for this facility in SQL/DS V3R5.

4.4.4 SQL/DS Log Archive

In VM the log can be archived to tape or DASD. In VSE the log can only be archived to tape. Only SQL/DS facilities can be used to archive the log. The database does not need to be shut down to take a log archive, and since the log is much smaller than the database, it takes less time. This can be used instead of the full database archive to improve database availability. One database archive is required to precede each sequence of log archives.

There are two advantages to log archiving:

- It usually takes less time because only the log is being archived, not the directory and dbextents.
- If the last database archive is unreadable or unavailable, you can restore the database from either a back level user archive or a back level SQL/DS database archive, and then, apply the changes that were recorded in all subsequent logs.

There are two disadvantages to log archiving:

- No logical units of work can be active during the checkpoint that immediately precedes the archive itself.
- It takes longer to restore the database. Each log archive needs to be applied in the order it was archived.

Refer to the *SQL/DS System Administration for VM* and the *SQL/DS System Administration for VSE* manuals, for a complete description of SQL/DS archiving.

4.4.5 User Archives

User archives are database archives done with non-SQL/DS facilities, such as the VMBACKUP Management System, the VM DASD Dump Restore service program (DDR), or the VSE/VSAM IDCAMS Backup/Restore feature. The new Data Restore feature BACKUP function runs as a user archive.

User archives include the SQL/DS directory and all dbextents, but not the log.

Because the SQL/DS code for archiving is DASD-independent, it does not take advantage of particular DASD characteristics to improve performance. User archives can implement an efficient method of archiving which may be faster than the SQL/DS archive facilities.

Advantages of using user archives:

- Improve performance in some cases, by taking advantage of DASD characteristics.
- Fast non-SQL/DS facilities can be used for this, such as DDR.

Disadvantage of using user archives:

- Database has to be down to take the user archive.

4.4.6 SQL/DS Database Restore

If a system failure causes the database manager to end abnormally, as long as the current log is available, recovery is automatic. Restart recovery is performed the next time the application server is started.

If an uncorrectable I/O error occurs on any of the devices containing the directory or dbextents, the application server ends abnormally. It will be necessary to restore the database from the most recent archive tapes.

The process is simple, since it requires only an update to the database startup parameters, and the archive tapes. The current log must be available. Because of this, it is advisable to use Dual Logging to avoid problems such as I/O errors on the device where the log is kept.

4.5 Partial Database Backup and Restore Strategies

Sometimes, due to a batch program error or human error, data becomes corrupted or is lost, and it is necessary to recover that data. There are several ways to recover the data and these processes are reviewed in this section.

The UNLOAD, RELOAD, APPLYLOG and TRANSLATE commands in the new Data Restore feature help the Database Administrator to recover this data promptly and with minimum disruption to the other users.

4.5.1 Data Restore Feature UNLOAD Command

This command allows the user to specify particular dbspaces for unloading. This may be used as an application backup, or to protect data that is more sensitive than the rest of the database.

This UNLOAD may then be used by the RELOAD command for recovery of particular tables. UNLOAD can be used whether the database server is online or offline. This could be particularly useful, if there has been a system failure on a disk other than where your data resides.

Your dbspaces could be UNLOADed before the database is recovered, then RELOADed after the database has been restored.

There is no forward log recovery support for UNLOADed data.

4.5.2 Data Restore Feature RELOAD Command

When a table has been corrupted, or dropped accidentally, you can use the RELOAD command to recover that table. The RELOAD can be processed from a backup provided by either a Data Restore feature BACKUP or UNLOAD commands. Note that RELOAD can also be run directly from an SQL/DS archive tape (without using TRANSLATE), although it will be slower.

PURGE Deletes all rows from the specified table and reloads all rows from the BACKUP or RELOAD.

NEW Creates the table and reloads all rows.

ADD Inserts rows into an existing table.

REPLACE Drops original table, recreates the table, reloads all the data and will recreate all indexes, referential integrity, views, grants, comments and labels.

Each of these options can be used in different circumstances depending on the problem. For example, it is possible to recreate a dropped table using the REPLACE option.

4.5.3 Data Restore Feature APPLYLOG Command

This command can be used for point in time recovery after a RELOAD has been run for a table. (RECOVERY=YES must be specified in the RELOAD for this command to run, and the database running under LOGMODE=L or LOGMODE=A.)

Use the LISTLOG command first to see what transactions are listed in the log, then APPLYLOG can be used to apply them up to a particular timestamp. This can be very useful if a lot of transactions were recorded against a particular table before it was accidentally dropped, or corrupted.

Note: The APPLYLOG command is **not** available for UNLOADed data.

4.5.4 Data Restore Feature TRANSLATE Command

This command will format an SQL/DS database archive into a Data Restore feature BACKUP file. This file can be used for the RESTORE and RELOAD commands.

To use this command for restoring, it is necessary to have a large amount of DASD available to hold the files ″translated″ from the database archive.

The TRANSLATE command creates four files on the A-disk of the machine running this command. Two of the files could be very large so special attention should be given to check that space is available before running this process. After successfully running the TRANSLATE, it is advisable to run the DESCRIBE command against the BACKUP DATA file to verify what tables can be reloaded from the database archive.

When running DESCRIBE, RELOAD and RESTORE commands using the TRANSLATEd database archive, the EXECs must be edited to update the FILEDEF for the ARCHIV label in VM. In VSE, a DLBL must be provided.

TRANSLATE could take a while depending on the size of the database archive.

This command is good when used to RELOAD particular tables and there are no User Archives taken using the BACKUP command.

To minimize database down time, an online SQL/DS archive can be created, instead of bringing down the database to create a BACKUP. This SQL/DS archive file can then be translated into a BACKUP file later, to be used if a RESTORE or RELOAD is needed.

4.5.5 Database Services Utility

The DBS Utility is an SQL/DS supplied program that allows loading and reloading data into, or unloading data from, an SQL/DS database. If the amount of data to be processed is large, or if exact sequences of database commands are to be used on a periodic basis, consider using this utility.

Usually, the DBS Utility is used for large-scale processing of SQL/DS databases; input to the utility, as well as its output, is in the form of sequential data sets.

The DBS Utility manipulates objects such as dbspaces, tables, packages. The four primary DBS Utility commands are DATALOAD, DATAUNLOAD, UNLOAD, and RELOAD.

The DBS Utility cannot be used for the purposes of partial recovery.

Refer to the *SQL/DS Database Services Utility for VSE* and the *SQL/DS Database Services Utility for VM* manuals, for a complete description of this application program.

4.6 Recovering a Table with Data Restore Feature

This example shows the steps used to recover a table using the new Data Restore feature.

A table NICOLOZI.TST was created and loaded with data. The database was running with LOGMODE=L, and a database archive was taken after the data was loaded.

Then, more transactions were run against the table. Records were inserted, updated and deleted. Then, by simulating a user error, the table was dropped.

We used the following sequence of SQL/DS Data Restore Version 3 Release 5 commands to restore the table.

1. TRANSLATE

   This process reformatted the SQL/DS database archive to a format that the SQL/DS Data Restore Version 3 Release 5 feature could use with best performance. The following EXEC file was used to generate the backup file.

```
/**/
¢SET EMSG ON¢
¢FILEDEF ARIARCH TAP1 SL 1           (RECFM FB BLOCK 28672 LRECL 4096¢
¢FILEDEF ARCHIV  DISK BACKUP  DATA A (RECFM VB BLOCK 32760¢
¢FILEDEF SYS0001 DISK SYS0001 DATA A (RECFM F  BLOCK 4096¢
¢FILEDEF HEADER  DISK HEADER  DATA A (RECFM F  BLOCK 4096¢
¢FILEDEF DIRWORK DISK DIRWORK DATA A (RECFM F  BLOCK 0512¢
¢FILEDEF SYSIN DISK TRANS   SYSIN  A¢
¢FILEDEF SYSPRINT DISK TRANS   SYSPRINT A¢
¢XEDIT TRANS SYSIN A¢
¢XTS91001¢
Exit rc
```

Figure 61. TRANSLATE EXEC

   After running this EXEC, the results were stored in the TRANS SYSPRINT file. A copy of this file is displayed in Figure 62 on page 103.

```
XTS9-143 TRANSLATE
XTS9-143 /*
XTS9-196 Do you want to continue the TRANSLATE process ?
XTS9-406 Enter 0(CANCEL) or 1(CONTINUE)
XTS9-403 Reply is 1
XTS9-100 SQL/DS Data Restore Version 3.5.0
XTS9-193 Mount first tape of SQL/DS archive
XTS9-406 Enter 0(CANCEL) or 1(CONTINUE)
XTS9-403 Reply is 1
XTS9-013 Table SQLDBA  .ACTIVITY          may be reloaded
XTS9-013 Table DATARFTR.CMD               may be reloaded
XTS9-013 Table SQLDBA  .DEPARTMENT        may be reloaded
XTS9-013 Table SQLDBA  .EMP_ACT           may be reloaded
XTS9-013 Table SQLDBA  .EMPLOYEE          may be reloaded
XTS9-013 Table NICOLOZI.HHHHH             may be reloaded
XTS9-013 Table SQLDBA  .IXLNGVAR          may be reloaded
XTS9-013 Table SQLDBA  .LMBRINS           may be reloaded
XTS9-013 Table CHRIS   .MAU               may be reloaded
XTS9-013 Table SQLDBA  .PROJ_ACT          may be reloaded
XTS9-013 Table SQLDBA  .PROJECT           may be reloaded
XTS9-013 Table EXAMPLE .ROUTINE           may be reloaded
XTS9-013 Table SQLDBA  .ROUTINE           may be reloaded
XTS9-013 Table SQLDBA  .STORED QUERIES    may be reloaded
XTS9-013 Table DATARFTR.SYSCATALOG        may be reloaded
XTS9-013 Table DATARFTR.SYSCOLAUTH        may be reloaded
XTS9-013 Table DATARFTR.SYSCOLUMNS        may be reloaded
XTS9-013 Table DATARFTR.SYSINDEXES        may be reloaded
XTS9-013 Table DATARFTR.SYSKEYCOLS        may be reloaded
XTS9-013 Table DATARFTR.SYSKEYS           may be reloaded
XTS9-013 Table SQLDBA  .SYSLANGUAGE       may be reloaded
XTS9-013 Table DATARFTR.SYSTABAUTH        may be reloaded
XTS9-013 Table SQLDBA  .SYSTEXT1          may be reloaded
XTS9-013 Table SQLDBA  .SYSTEXT2          may be reloaded
XTS9-013 Table DATARFTR.SYSUSAGE          may be reloaded
XTS9-013 Table DATARFTR.SYSVIEWS          may be reloaded
XTS9-013 Table NICOLOZI.TEST              may be reloaded
XTS9-013 Table NICOLOZI.TST               may be reloaded
XTS9-007 Processing successfully completed
```

Figure 62. TRANSLATE SYSPRINT

Four files were generated on disk and one of them is the backup file (BACKUP DATA A file). This is required as input for the next steps.

Before running a RESTORE or RELOAD command, it is possible to run the DESCRIBE command to ensure that the database archive just TRANSLATEd contains the tables we need to recover.

2. DESCRIBE

   This process read the backup file and listed our table as available for reload. The DESCRIBE process is displayed in Figure 63 on page 104:

```
XTS9-143 CONTROL DBNAME=SQL350
XTS9-143 DESCRIBE
XTS9-143 /*
XTS9-196 Do you want to continue the DESCRIBE  process ?
XTS9-406 Enter 0(CANCEL) or 1(CONTINUE)
XTS9-403 Reply is 1
XTS9-100 SQL/DS Data Restore Version 3.5.0
XTS9-192 Processing SQL/VSE  archived by SQL/DS on (12/10/95-13:39:30)
XTS9-013 Table SQLDBA  .ACTIVITY          may be reloaded
XTS9-013 Table DATARFTR.CMD               may be reloaded
XTS9-013 Table SQLDBA  .DEPARTMENT        may be reloaded
XTS9-013 Table SQLDBA  .EMP_ACT           may be reloaded
XTS9-013 Table SQLDBA  .EMPLOYEE          may be reloaded
XTS9-013 Table NICOLOZI.HHHHH             may be reloaded
XTS9-013 Table SQLDBA  .IXLNGVAR          may be reloaded
XTS9-013 Table SQLDBA  .LMBRINS           may be reloaded
XTS9-013 Table CHRIS   .MAU               may be reloaded
XTS9-013 Table SQLDBA  .PROJ_ACT          may be reloaded
XTS9-013 Table SQLDBA  .PROJECT           may be reloaded
XTS9-013 Table EXAMPLE .ROUTINE           may be reloaded
XTS9-013 Table SQLDBA  .ROUTINE           may be reloaded
XTS9-013 Table SQLDBA  .STORED QUERIES    may be reloaded
XTS9-013 Table DATARFTR.SYSCATALOG        may be reloaded
XTS9-013 Table DATARFTR.SYSCOLAUTH        may be reloaded
XTS9-013 Table DATARFTR.SYSCOLUMNS        may be reloaded
XTS9-013 Table DATARFTR.SYSINDEXES        may be reloaded
XTS9-013 Table DATARFTR.SYSKEYCOLS        may be reloaded
XTS9-013 Table DATARFTR.SYSKEYS           may be reloaded
XTS9-013 Table SQLDBA  .SYSLANGUAGE       may be reloaded
XTS9-013 Table DATARFTR.SYSTABAUTH        may be reloaded
XTS9-013 Table SQLDBA  .SYSTEXT1          may be reloaded
XTS9-013 Table SQLDBA  .SYSTEXT2          may be reloaded
XTS9-013 Table DATARFTR.SYSUSAGE          may be reloaded
XTS9-013 Table DATARFTR.SYSVIEWS          may be reloaded
XTS9-013 Table NICOLOZI.TEST              may be reloaded
XTS9-013 Table NICOLOZI.TST               may be reloaded
XTS9-007 Processing successfully completed
```

Figure 63. DESCRIBE SYSPRINT

We can see that the table NICOLOZI.TST exists on our TRANSLATEd file. The next step then is to RELOAD the table.

3. RELOAD

   This process selected our table, NICOLOZI.TST and reloaded it into the database. The SYSIN file is using the following option or control statements.

```
OPTIONS RECOVERY=YES DEVICE=DASD
CONTROL DBNAME=SQL350 VERIFY=NO
RELOAD  CREATOR=NICOLOZI,TNAME=TST,FUNCT=NEW
        DBSPACE=NICOLOZI
```

Figure 64. RELOAD SYSIN

RECOVERY = YES This command will cause the reload process to read the current log and identify any transactions against our table.

DEVICE = DASD This command must be specified to identify that the backup file is not on tape.

VERIFY = NO This command must be specified because the backup was translated from an SQL/DS database archive and not taken by the SQL/DS Data Restore Version 3 Release 5 feature BACKUP command

FUNCT=NEW This command is used when the table does not exist and must be created from the BACKUP file.

The RELOAD SYSPRINT in Figure 65 shows the process of the RELOAD.

```
XTS9-143 OPTIONS RECOVERY=YES DEVICE=DASD
XTS9-143 CONTROL DBNAME=SQL350 VERIFY=NO
XTS9-143 RELOAD  CREATOR=NICOLOZI,TNAME=TST,FUNCT=NEW
XTS9-143         DBSPACE=NICOLOZI
XTS9-143 /*
XTS9-196 Do you want to continue the RELOAD    process ?
XTS9-406 Enter 0(CANCEL) or 1(CONTINUE)
XTS9-403 Reply is 1
XTS9-100 SQL/DS Data Restore Version 3.5.0
XTS9-192 Processing SQL/VSE  archived by SQL/DS on (12/10/95-17:59:50)
XTS9-182 Following files are needed for recovery
XTS9-195 ARCHIVE      currently mounted
XTS9-179 Current log
XTS9-406 Enter 0(CANCEL) or 1(CONTINUE)
XTS9-403 Reply is 1
XTS9-102      2344 rows loaded, procedure completed
XTS9-128      2344 rows loaded into (NICOLOZI.TST)
XTS9-101         1 tables successfully processed
XTS9-314 COMMIT WORK successful for reload of data, creating required objects
CONNECT SQLDBA  ;
COMMIT WORK ;
XTS9-184 Processing current log
XTS9-407 Enter 0(CANCEL),1(CONTINUE) or 111(SKIPFILE)
XTS9-403 Reply is 1
XTS9-007 Processing successfully completed
```

Figure 65. RELOAD SYSPRINT

Up to this point, we recovered the table as it was when the database archive took place. If it is necessary to apply the log changes, then the following steps must be followed.

4. LISTLOG

   This command generated a list of all the transactions made against our table since the last database archive. Each transaction is accompanied by a timestamp.

```
XTS9-143 CONTROL DBAPW=********
XTS9-143 LISTLOG
XTS9-143 /*
XTS9-196 Do you want to continue the LISTLOG   process ?
XTS9-406 Enter 0(CANCEL) or 1(CONTINUE)
XTS9-403 Reply is 1
XTS9-100 SQL/DS Data Restore Version 3.5.0

C 00006636 1995-284-10-30-37-493408
INSERT INTO †NICOLOZI†.†TST† VALUES (¢A         ¢,       123,00 )

C 00006636 1995-284-10-30-42-648704
INSERT INTO †NICOLOZI†.†TST† VALUES (¢B         ¢,       234,00 )

C 00006636 1995-284-10-30-48-248048
INSERT INTO †NICOLOZI†.†TST† VALUES (¢C         ¢,       345,00 )

C 00006637 1995-284-10-31-22-597680
UPDATE †NICOLOZI†.†TST† SET †A† =¢AA        ¢,†B† =       123,00
        WHERE †A† =¢A         ¢ AND †B† =       123,00

C 00006638 1995-284-10-31-36-665632
DELETE FROM †NICOLOZI†.†TST† WHERE †A† =¢C         ¢ AND †B† =       345,00

C 00006645 1995-284-10-32-42-727216
DROP TABLE  †NICOLOZI†.†TST†
XTS9-185 Forward recovery stopped due to DROP TBL
XTS9-186 Timestamp of statement is 1995-284-10-32-42-727216
```

Figure 66. LISTLOG SYSPRINT

5. APPLYLOG

   The DROP TABLE NICOLOZI.TST transaction was identified and the timestamp used in the APPLYLOG SYSIN file. The table was restored to the state it was in before the DROP TABLE transaction.

```
CONTROL DBAPW=SQLDBAPW
APPLYLOG END=1995-284-10-32-42-727216
```

Figure 67. APPLYLOG SYSIN

```
XTS9-143 CONTROL DBAPW=********
XTS9-143 APPLYLOG END=1995-284-10-32-42-727216
XTS9-143 /*
XTS9-196 Do you want to continue the APPLYLOG  process ?
XTS9-406 Enter 0(CANCEL) or 1(CONTINUE)
XTS9-403 Reply is 1
XTS9-100 SQL/DS Data Restore Version 3.5.0
XTS9-189 Timestamp 1995-284-10-32-42-727216 reached in log file
XTS9-007 Processing successfully completed
```

Figure 68. APPLYLOG SYSPRINT

4.7 Making Backups with Data Restore Feature

The new SQL/DS Data Restore Version 3 Release 5 feature can also be used to back up dbspaces while the database is offline. If there has been a DASD or system failure, but the minidisk with your table is still OK, then an UNLOAD and RELOAD can be used to recover the table. In this example, we will take an UNLOAD of a dbspace and RELOAD of a table.

```
CONTROL DBNAME=SQL350,DBAPW=SQLDBAPW
UNLOAD  MODE=OFFLINE,DBSPACE=(NICOLOZI),COND=INCLUDE
```

Figure 69. UNLOAD SYSIN

Note in Figure 69 the option MODE=OFFLINE, this will allow the UNLOAD command to run when the database is down. COND=INCLUDE indicates that we want to unload only the table(s) in the NICOLOZI dbspace. If we had used COND=EXCLUDE, the UNLOAD would have included all the dbspaces except the NICOLOZI dbspace.

Figure 70 shows the SYSPRINT of the UNLOAD.

```
XTS9-143 CONTROL DBNAME=SQL350,DBAPW=********
XTS9-143 UNLOAD  MODE=OFFLINE,DBSPACE=(NICOLOZI),COND=INCLUDE
XTS9-143 /*
XTS9-196 Do you want to continue the UNLOAD    process ?
XTS9-406 Enter 0(CANCEL) or 1(CONTINUE)
XTS9-403 Reply is 1
XTS9-100 SQL/DS Data Restore Version 3.5.0
XTS9-309 Processing SQL/DS version 3
XTS9-160 External labeling of this unload is
XTS9-142 Base SQL350   Date 13/10/95 Time 10:03:22
XTS9-013 Table CHRIS   .MAU                may be reloaded
XTS9-013 Table NICOLOZI.TEST               may be reloaded
XTS9-013 Table NICOLOZI.TST                may be reloaded
XTS9-006 Processing DDSK1
XTS9-005       131 blocks saved
XTS9-007 Processing successfully completed
```

Figure 70. UNLOAD SYSPRINT

The RELOAD SYSPRINT in Figure 71 on page 108 shows the process of the RELOAD. All DATARFTR tables loaded in the process are part of the SQL/DS Data Restore Version 3 Release 5 feature control tables.

```
XTS9-143 OPTIONS RECOVERY=YES
XTS9-143 CONTROL DBNAME=SQL350
XTS9-143 RELOAD  CREATOR=NICOLOZI,TNAME=TST,FUNCT=REPLACE
XTS9-143         DBSPACE=NICOLOZI
XTS9-143 /*
XTS9-196 Do you want to continue the RELOAD    process ?
XTS9-406 Enter 0(CANCEL) or 1(CONTINUE)
XTS9-403 Reply is 1
XTS9-100 SQL/DS Data Restore Version 3.5.0
XTS9-192 Processing SQL/VSE  archived by SQL/DS on (12/10/95-13:39:30)
XTS9-182 Following files are needed for recovery
XTS9-195 ARCHIVE      currently mounted
XTS9-179 Current log
XTS9-406 Enter 0(CANCEL) or 1(CONTINUE)
XTS9-403 Reply is 1
XTS9-102       788 rows loaded, procedure completed
XTS9-128      2344 rows loaded into (NICOLOZI.TST)
XTS9-128       414 rows loaded into (DATARFTR.SYSCOLUMNS)
XTS9-128        53 rows loaded into (DATARFTR.SYSCATALOG)
XTS9-128       145 rows loaded into (DATARFTR.SYSTABAUTH)
XTS9-128        65 rows loaded into (DATARFTR.SYSINDEXES)
XTS9-128         4 rows loaded into (DATARFTR.SYSVIEWS)
XTS9-128        17 rows loaded into (DATARFTR.SYSKEYCOLS)
XTS9-128        13 rows loaded into (DATARFTR.SYSKEYS)
XTS9-128        70 rows loaded into (DATARFTR.SYSUSAGE)
XTS9-128         3 rows loaded into (DATARFTR.SYSCOLAUTH)
XTS9-101        10 tables successfully processed
XTS9-314 COMMIT WORK successful for reload of data, creating required objects
```

Figure 71. RELOAD SYSPRINT

4.8 Recommendations

If your business needs include 24x7 database availability, a possible archiving and recovery strategy can be:

- Take full database and log archives online.

  In SQL/DS V3R5, both archives have been improved and are faster. SQL/Master can be used to automate these tasks.

- If recovering from system or DASD failures, then you can use the SQL/DS restore facilities.

- If recovering from application failures or user logic errors, then you can use the following command sequence:

  1. Data Restore feature TRANSLATE command

     To speed up recovery time the TRANSLATE could be run immediately after the archive rather than waiting until a failure occurs. If the database is small, the TRANSLATE command could be skipped.

  2. Data Restore feature RELOAD command
  3. Data Restore feature LISTLOG command
  4. Data Restore feature APPLYLOG command

If your business needs allow you to interrupt the availability of the database for maintenance purposes, then you can use the improved SQL/DS V3R5 archive facility, or the new Data Restore feature BACKUP command. You can use SQL/Master to automate both cases.

Chapter 5. Installation, Migration and Coexistence

This chapter describes general considerations for installation, migration from SQL/DS V2R2 and later, and coexistence with SQL/DS V3R3 and later.

5.1 Terminology

Installation Set of activities that a customer performs to install SQL/DS V3R5.

Migration Set of activities that a customer performs to move operations from one release of SQL/DS to a later release of SQL/DS.

Coexistence Is using a new release of SQL/DS in the same environment with other releases of SQL/DS.

Compatibility Is the ability to execute programs on a release of SQL/DS that were written to execute on another release of SQL/DS.

5.2 Considerations for Each Command or Function

The migration paths applicable to SQL/DS V3R5 are migration from SQL/DS V2R2, SQL/DS V3R1, SQL/DS V3R2, SQL/DS V3R3 and SQL/DS V3R4. The Version 3 Release 5 Installation documentation will include the migration steps required by the new functions of this release.

Programs written for SQL/DS V2R2, SQL/DS V3R1, SQL/DS V3R2, SQL/DS V3R3 and SQL/DS V3R4 will not have to be explicitly re-preprocessed to operate under Version 3 Release 5. Migration among supported releases of VM/ESA operating systems will not affect SQL/DS (VM) Version 3 Release 5.

Migration among supported releases of VSE/ESA operating systems will not affect SQL/DS (VSE) Version 3 Release 5.

The process of moving a database from VSE/ESA to VM/ESA (and vice versa) is unchanged from previous releases. Since the data stored in an SQL/DS database is system-independent, there is no need to convert the data in a database when it is moved from VSE to VM (or from VM to VSE). Before migrating, ensure that SQL/DS is installed on both systems at the same release level. Move the data by taking an SQL/DS database archive on the VM (or VSE) system and then restore that archive on the other system.

For information regarding migration from any environment to a VM/ESA system with the SQL/DS VM Data Spaces Support feature, refer to *SQL/DS VM Data Spaces Support for VM/ESA V3R3.*

It is possible to move data between SQL/DS and other DB2 Family Databases (such as DB2 for MVS, DB2 for OS/2 or DB2 for AIX) using products such as DataPropagator Relational and DataRefresher. Refer to Chapter 7, “SQL/DS Client/Server Solutions” on page 143.

5.2.1 CICS Database Switching

5.2.1.1 Migration and Installation Considerations

Entries in DFHPCT for TRANSID=CIRA, TRANSID=CIRR and TRANSID=CIRC need to be added. These entries can be found in ARIS35PC, a source member provided by SQL/DS.

```
*
RSQL4    DFHPCT TYPE=ENTRY,                          X
               TRANSID=CIRA,                         X
               PROGRAM=ARIRCONT,                     X
               TWASIZE=0,                            X
               DTB=YES,                              X
               SPURGE=YES,                           X
               TPURGE=YES
*
RSQL5    DFHPCT TYPE=ENTRY,                          X
               TRANSID=CIRR,                         X
               PROGRAM=ARIRCONT,                     X
               TWASIZE=0,                            X
               DTB=YES,                              X
               SPURGE=YES,                           X
               TPURGE=YES
*
RSQL6    DFHPCT TYPE=ENTRY,                          X
               TRANSID=CIRC,                         X
               PROGRAM=ARIRCONT,                     X
               TWASIZE=0,                            X
               DTB=YES,                              X
               SPURGE=YES,                           X
               TPURGE=YES
```

CICS Restart Resynchronization support is shipped with CICS/VSE V2R1, and it is assumed that this support has been enabled. See the *SQL/DS Installation for VSE* manual for more details on installing this support.

5.2.1.2 Coexistence Considerations

The behavior and syntax of the CIRB has not changed. It can continue to be used in the same manner as in previous releases. New parameters to existing CICS transactions CIRB, CIRD and ISQL are added at the end and use defaults if not specified so that existing procedures will continue to work in the same way as they did previously.

This design is compatible with VisualGen. VisualGen documentation states that you cannot connect to different application servers in the same transaction in a VSE online environment because SQL/DS does not support it. This change removes this restriction from SQL/DS. This is also compatible with earlier releases of SQL/DS. It will be possible for an SQL/DS V3R5 online resource adapter to establish connections to earlier supported versions of SQL/DS application servers.

ISQL and CICS/VSE application programs that access SQL/DS data written to take advantage of CICS database switching cannot be run with an earlier version of the online resource adapter. The CONNECT statement that allows database switching is not supported by earlier versions of the online resource adapter. Preprocessing and linkediting would be successful, but the application would fail at run time.

5.2.1.3 Control Blocks

Many online resource adapter control blocks will be expanded to allow connections to multiple application servers.

5.2.1.4 Storage

Storage usage to support the CICS interface to the database is proportional to the total number of active links to all servers.

It is important to remember that **ALL** of the storage for the online resource adapter must come from below the 16 megabyte line. This is because the online resource adapter is a CICS task related user exit and it must be RMODE 24 and CICS always calls it in AMODE 24.

5.2.1.5 Online Resource Adapter and Control Transactions

- Size of the following phases:

Phase Name	Size (bytes)
ARIRCONT	322,992
ARI00LRM	58,024
ARIMS001	196,624
ARICMOD	304
ARICDIRD	13,128
Total	591,072

  Note: The size of the ARICDIRD phase is dependent on the number of entries defined in the DBNAME directory. The size given assumes that the IBM supplied default DBNAME directory is being used.

- Global Control Blocks
  - There is one **RA Anchor Workarea** per online resource adapter. Its size is 248 bytes.
  - There is one **RA Server Workarea** per connected server. The size of any RA Server Workarea is based on the number of links to that server and can be calculated by the following formula:

    $$440 + links \times 33{,}688 + round\ bytes$$

    where:
    - "links" is the number of links, and
    - "round" is a round off allowance, calculated as follows:

      round = 8 x (TRUNC ((links + 3) / 4) - 1)

  - Execution Time Control Blocks:

    There is one **Transaction Workarea** per CICS transaction that issues EXEC SQL statements. Its size is 3568 bytes.

For example, if we issue the following CIRB transaction:

```
CIRB ,5,,,,(SQLMACH1,SQLMACH2)
```

the amount of storage used is calculated as follows:

```
storage = Size of Phases + Global Control Blocks
        = Size of Phases + RA Anchor Workarea + 2(RA Server Workarea)
        = 591,072 + 248 + 2 x (440 + links x 33,688 + round)
        = 591,320 + 2 x (440 + 5 x (33,688) + 8)
        = 591,320 + 2 x (168,888)
        = 929,096 bytes
```

Calculation of “round” is as follows:

```
round = 8 x (TRUNC((links + 3) / 4) - 1)
round = 8 x (TRUNC(( 5 + 3) / 4 ) - 1)
round = 8 x (TRUNC(2) - 1)
round = 8(2 - 1)
round = 8
```

The amount of storage required to execute the CIRA transaction is calculated somewhat differently because the phases and the RA Anchor Workarea have already been loaded into storage by a previous CIRB transaction. The additional storage used as a result of executing CIRA is calculated to be the sum of the sizes for each RA Server Workarea it has set up.

For example, if we issue the following CIRA transaction:

```
CIRA ,2,,SQLMACH3
```

the amount of additional storage used is calculated as follows:

```
storage = Size of RA Server Workarea
        = 440 + links x 33,688 + round
        = 440 + 2 x (33,688) + 0
        = 440 + 67,376
        = 67,816 bytes
```

Calculation of “round” is as follows:

```
round = 8 x (TRUNC((2+3) / 4) - 1)
round = 8 x (TRUNC(1.25) - 1)
round = 8 x (1 - 1)
round = 0
```

5.2.1.6 ISQL

```
ISQL size = 580 bytes + online resource adapter requirements
          +  45 bytes for each ISQL user
          + 180 bytes for each additional message repository.
```

5.2.2 DRDA Limited Accounting

5.2.2.1 Migration

There are no migration considerations for this function.

5.2.2.2 Installation Considerations

There are no installation considerations for this function.

5.2.2.3 Coexistence Considerations

SQL/DS(VM) V3R3 and V3R4 applications will continue to send 16 blanks in the PRDDTA parameter of the ACCRDB DDM command. Also, SQL/DS V3R3 and V3R4 servers will not recognize accounting data in the PRDDTA parameter of the ACCRDB DDM command. They will continue to put 16 blanks into User Accounting Records.

5.2.3 SQLSTATE Changes to Match SQL92

5.2.3.1 Migration

This function will not introduce any changes to migration.

5.2.3.2 Installation Considerations

There are no installation considerations for this function.

5.2.3.3 Coexistence Considerations

Changing SQLSTATEs is an incompatible change since many SQLSTATE values that are returned for diagnostic situations will be different from previous releases of SQL/DS. As DB2 for MVS (Version 4.1 and later), DB2 for OS/400 (Version 3.6 and later) and DB2 common server (Version 2.1 and later) are all switching to the new SQL92 SQLSTATE values, the subsequent uniformity of SQLSTATE processing across the IBM databases makes this change attractive to application programmers.

5.2.4 SQL/DS Data Restore Version 3 Release 5

5.2.4.1 Migration

Not applicable.

5.2.4.2 Installation Considerations

Refer to the *IBM SQL/DS Data Restore Guide* for details.

5.2.4.3 Coexistence Considerations

SQL/DS V3R5 database archive tapes can be used as input to the restore process.

Archives from previous releases of SQL/DS are not supported.

5.2.5 Increased Buffer Maximums

5.2.5.1 Migration

There are no migration considerations for this function.

5.2.5.2 Installation Considerations

There are no installation considerations for this function.

5.2.5.3 Coexistence Considerations

There are no coexistence considerations for this function.

5.2.5.4 Resources

Naturally, specifying a larger number of buffers will increase the amount of virtual storage used by the buffers and their associated BUFFCONT structures.

In addition, the Hash Chain Anchor Tables (HCAT) will be increased in size to improve performance.

5.2.5.5 Control Blocks

The control block will have several fields added, several fields changed and two fields deleted.

Two new dynamically sized HCATs will be created, one for the page buffers and one for the directory buffers.

5.2.5.6 Storage

The YTABLE4 control block will decrease in size by 992 bytes.

The two dynamic HCATs created will have sizes that depend on the NPAGBUF and NDIRBUF startup parameters. The two HCATs are not necessarily the same size; each HCAT's size is determined as follows:

Table 19. Hash Chain Anchor Table Sizes

Number of Buffers range	HCAT Size in bytes
10 to 2048	1K
2049 to 4096	2K
4097 to 8192	4K
8193 to 16384	8K
16385 to 32768	16K
32769 to 65536	32K
65537 to 131072	64K
131073 to 262144	128K
262145 to 400000	256K

Customers who do not change their current number of buffers will require approximately 4K to 8K more virtual storage for the HCATs, which are being increased in size to improve performance.

5.2.6 Additional CCSID Support

5.2.6.1 Migration

For SQL/DS V3R5, rows to enable the new conversions will be added to the table SYSTEM.SYSSTRINGS and to the file ARISSTR MACRO during migration.

5.2.6.2 Installation Considerations

For SQL/DS V3R5, rows to enable the new conversions will be added to the table SYSTEM.SYSSTRINGS and to the file ARISSTR MACRO during installation.

5.2.6.3 Coexistence Considerations

There are no coexistence considerations for this function.

5.2.7 Support for LE/VSE in Single User Mode

5.2.7.1 Migration

There are no migration considerations for this function.

5.2.7.2 Installation Considerations

There are no installation considerations for this function.

5.2.7.3 Coexistence Considerations

When TRAP(OFF) is specified in the LE/VSE environment, usual SQL/DS abend exit processing takes place. The new SQL/DS support for the LE/VSE TRAP option is not required in the following circumstances:

- When an existing application is executed in a non-LE/VSE environment.
- When an application is executed in an LE/VSE environment with TRAP(ON) specified and the application is running in SQL/DS MUM.

For these cases abend handling will be as before.

Information on source and/or object compatibility for programs compiled with an existing DOS/VS COBOL or PL/I compiler, which are now to be executed under LE/VSE, can be obtained from the LE/VSE documentation.

5.2.8 Enhanced Directory Expansion

5.2.8.1 Migration

There are no migration considerations for this function.

5.2.8.2 Installation Considerations

There are no installation considerations for this function.

5.2.8.3 Coexistence Considerations

There are no coexistence considerations for this function.

5.2.8.4 Resources

There are no additional resources required as a result of this function. Of course, to expand the directory, then a larger directory minidisk (in VM) or a larger VSAM extent (in VSE) is required.

5.2.9 Archive Performance Enhancements

5.2.9.1 Migration

For VSE, the SHAREOPTIONS value for the SQL/DS directory, data and log data sets must be set to 2. This allows the directory, data and log files to be opened by any number of requests for input processing even if one request is using it for output processing. But only one request can open the file for output at any one time. Without setting the SHAREOPTIONS value to 2, the database manager and log archives will fail because the second set of ACBs cannot be opened.

The next example shows how to change this value.

VSAM Clusters Without DATA(name) Attribute: Prior to SQL/DS V3R5, the manuals recommended defining VSAM clusters as is shown in Figure 72. Because of this, the DATA names for these clusters were generated by VSAM; therefore, when there was a need to alter a value it was necessary to do a LISTCAT to obtain those names.

```
// JOB SQLVS340 DEFINE SQL/DS DATABASE VSAM CLUSTERS
// DLBL   IJSYSUC,¢SQLPCAT.USER.CATALOG¢,,VSAM
// EXEC   IDCAMS,SIZE=AUTO

   DEFINE CLUSTER /* DEFINE SQL/DS DATABASE DIRECTORY */  -
          ( NAME  (SQLVS340.BDISK.SQLDIR40)               -
            CNVSZ (512)                                   -
            CYL   (40)                                    -
            NONINDEXED                                    -
            VOL   (SYSWK3)                                -
            RECSZ (505 505)                               -
            REUSE                                         -
            SHR   (1)         )                           -
            CAT   (SQLPCAT.USER.CATALOG)

   DEFINE CLUSTER /* DEFINE SQL/DS DATABASE LOG */        -
          ( NAME  (SQLVS340.LOGDSK1.SQLLOG)               -
            CNVSZ (4096)                                  -
            CYL   (50)                                    -
            NONINDEXED                                    -
            VOL   (SYSWK3)                                -
            RECSZ (4089 4089)                             -
            REUSE                                         -
            SHR   (1)         )                           -
            CAT   (SQLPCAT.USER.CATALOG)

   DEFINE CLUSTER /* DEFINE SQL/DS DATABASE DBEXTENT 1 */ -
          ( NAME  (SQLVS340.DDSK1.POOL1)                  -
            CNVSZ (4096)                                  -
            CYL   (50)                                    -
            NONINDEXED                                    -
            VOL   (SYSWK4)                                -
            RECSZ (4089 4089)                             -
            REUSE                                         -
            SHR   (1)         )                           -
            CAT   (SQLPCAT.USER.CATALOG)

   ...

   DEFINE CLUSTER /* DEFINE SQL/DS DATABASE DBEXTENT 11*/ -
          ( NAME  (SQLVS340.DDSK11.POOL4)                 -
            CNVSZ (4096)                                  -
            CYL   (50)                                    -
            NONINDEXED                                    -
            VOL   (SYSWK4)                                -
            RECSZ (4089 4089)                             -
            REUSE                                         -
            SHR   (1)         )                           -
            CAT   (SQLPCAT.USER.CATALOG)

/*
/&
```

Figure 72. VSAM Clusters Without DATA(name) Attribute

DATA(name) Attributes Generated by VSAM: Figure 73 shows the LISTCAT job necessary to list the SQL/DS VSAM catalog contents.

```
// JOB  LISTCAT LIST SQL/DS VSAM CATALOG CONTENTS
// DLBL IJSYSUC,¢SQLPCAT.USER.CATALOG¢,,VSAM
// EXEC IDCAMS,SIZE=AUTO

   LISTCAT ALL -
   CATALOG (SQLPCAT.USER.CATALOG                    )

CLUSTER ------- SQLVS340.BDISK.SQLDIR40
   DATA ------- T34A4F58.VSAMDSET.DFD95273.TABC01C5.T34A4F58

CLUSTER ------- SQLVS340.LOGDSK1.SQLLOG
   DATA ------- T61EC760.VSAMDSET.DFD95273.TABC01C5.T61EC760

CLUSTER ------- SQLVS340.DDSK1.POOL1
   DATA ------- T7706048.VSAMDSET.DFD95273.TABC01C5.T7706048

...

CLUSTER ------- SQLVS340.DDSK11.POOL4
   DATA ------- T6348014.VSAMDSET.DFD95273.TABC01C6.T6348014

EOJ LISTCAT   MAX.RETURN CODE=0000
```

Figure 73. DATA(name) Attributes Generated by VSAM

ALTER SHR(2) With Names Generated by VSAM: Using the names from the previous LISTCAT, it is possible to run the job in Figure 74 to alter the value of SHAREOPTIONS for all the data sets.

```
// JOB SQLVS350  ALTER SQL/DS DATABASE VSAM DEFINITIONS
// DLBL   IJSYSUC,¢SQLPCAT.USER.CATALOG¢,,VSAM
// EXEC   IDCAMS,SIZE=AUTO

          /**** CLUSTER SQLVS340.BDISK.SQLDIR40 *** */
   ALTER  T34A4F58.VSAMDSET.DFD95273.TABC01C5.T34A4F58   -
          SHR   (2)                                      -
          CAT   (SQLPCAT.USER.CATALOG)

          /**** CLUSTER SQLVS340.LOGDSK1.SQLLOG *** */
   ALTER  T61EC760.VSAMDSET.DFD95273.TABC01C5.T61EC760   -
          SHR   (2)                                      -
          CAT   (SQLPCAT.USER.CATALOG)

          /**** CLUSTER SQLVS340.DDSK1.POOL1 *** */
   ALTER  T7706048.VSAMDSET.DFD95273.TABC01C5.T7706048   -
          SHR   (2)                                      -
          CAT   (SQLPCAT.USER.CATALOG)

   ...

          /**** CLUSTER SQLVS340.DDSK11.POOL4 *** */
   ALTER  T6348014.VSAMDSET.DFD95273.TABC01C6.T6348014   -
          SHR   (2)                                      -
          CAT   (SQLPCAT.USER.CATALOG)
/*
/&
```

Figure 74. ALTER SHR(2) With Names Generated by VSAM

VSAM Clusters With DATA(name) Attribute: We recommend from now on to give a name to the DATA data sets when defining the SQL/DS database VSAM clusters, as shown in Figure 75 on page 120.

The only naming convention is to add ″.DATA″ to the cluster name. The results are shown in Figure 76 on page 121 after running a LISTCAT.

```
// JOB SQLVS350 DEFINE SQL/DS DATABASE VSAM CLUSTERS
// DLBL   IJSYSUC,¢SQLPCAT.USER.CATALOG¢,,VSAM
// EXEC   IDCAMS,SIZE=AUTO

   DEFINE CLUSTER /* DEFINE DATABASE DIRECTORY */        -
          ( NAME  (SQLVS350.BDISK.SQLDIR40)             -
            CNVSZ (512)                                 -
            CYL   (40)                                  -
            NONINDEXED                                  -
            VOL   (SYSWK3)                              -
            RECSZ (505 505)                             -
            REUSE                                       -
            SHR   (1)         )                         -
          DATA                                          -
          ( NAME  (SQLVS350.BDISK.SQLDIR40.DATA) )      -
            CAT   (SQLPCAT.USER.CATALOG)

   DEFINE CLUSTER /* DEFINE DATABASE LOG */             -
          ( NAME  (SQLVS350.LOGDSK1.SQLLOG)             -
            CNVSZ (4096)                                -
            CYL   (50)                                  -
            NONINDEXED                                  -
            VOL   (SYSWK3)                              -
            RECSZ (4089 4089)                           -
            REUSE                                       -
            SHR   (1)         )                         -
          DATA                                          -
          ( NAME  (SQLVS350.LOGDSK1.SQLLOG.DATA) )      -
            CAT   (SQLPCAT.USER.CATALOG)

   DEFINE CLUSTER /* DEFINE DATABASE DBEXTENT 1 */      -
          ( NAME  (SQLVS350.DDSK1.POOL1)                -
            CNVSZ (4096)                                -
            CYL   (50)                                  -
            NONINDEXED                                  -
            VOL   (SYSWK4)                              -
            RECSZ (4089 4089)                           -
            REUSE                                       -
            SHR   (1)         )                         -
          DATA                                          -
          ( NAME  (SQLVS350.DDSK1.POOL1.DATA) )         -
            CAT   (SQLPCAT.USER.CATALOG)

   ...

   DEFINE CLUSTER /* DEFINE DATABASE DBEXTENT 11 */     -
          ( NAME  (SQLVS350.DDSK11.POOL4)               -
            CNVSZ (4096)                                -
            CYL   (50)                                  -
            NONINDEXED                                  -
            VOL   (SYSWK4)                              -
            RECSZ (4089 4089)                           -
            REUSE                                       -
            SHR   (1)         )                         -
          DATA                                          -
          ( NAME  (SQLVS350.DDSK11.POOL4.DATA) )        -
            CAT   (SQLPCAT.USER.CATALOG)
/*
/&
```

Figure 75. VSAM Clusters With DATA(name) Attribute

DATA(name) Attributes Defined by User: Figure 76 displays the results from the LISTCAT to list the SQL/DS VSAM catalog contents. Both CLUSTER and DATA names are similar and are related.

```
// JOB  LISTCAT LIST SQL/DS VSAM CATALOG CONTENTS
// DLBL IJSYSUC,¢SQLPCAT.USER.CATALOG¢,,VSAM
// EXEC IDCAMS,SIZE=AUTO

   LISTCAT ALL -
   CATALOG (SQLPCAT.USER.CATALOG                    )

CLUSTER ------- SQLVS350.BDISK.SQLDIR40
   DATA ------- SQLVS350.BDISK.SQLDIR40.DATA

CLUSTER ------- SQLVS350.LOGDSK1.SQLLOG
   DATA ------- SQLVS350.LOGDSK1.SQLLOG.DATA

CLUSTER ------- SQLVS350.DDSK1.POOL1
   DATA ------- SQLVS350.DDSK1.POOL1.DATA

...

CLUSTER ------- SQLVS350.DDSK11.POOL4
   DATA ------- SQLVS350.DDSK11.POOL4.DATA

EOJ LISTCAT   MAX.RETURN CODE=0000
```

Figure 76. DATA(name) Attributes Defined by User

ALTER SHR(2) With Names Defined by User: The previous naming exercise simplifies our task of listing the VSAM DATA files to change their SHAREOPTIONS value to 2 as shown in Figure 77.

```
// JOB SQLVS350 ALTER SQL/DS DATABASE VSAM DEFINITIONS
// DLBL   IJSYSUC,¢SQLPCAT.USER.CATALOG¢,,VSAM
// EXEC   IDCAMS,SIZE=AUTO

          /* ALTER SQL/DS DATABASE DIRECTORY  */
   ALTER  SQLVS350.BDISK.SQLDIR40.DATA           -
          SHR   (2)                              -
          CAT   (SQLPCAT.USER.CATALOG)

          /* ALTER SQL/DS DATABASE LOG        */
   ALTER  SQLVS350.LOGDSK1.SQLLOG.DATA           -
          SHR   (2)                              -
          CAT   (SQLPCAT.USER.CATALOG)

          /* ALTER SQL/DS DATABASE DBEXTENT 1 */
   ALTER  SQLVS350.DDSK1.POOL1.DATA              -
          SHR   (2)                              -
          CAT   (SQLPCAT.USER.CATALOG)

   ...

          /* ALTER SQL/DS DATABASE DBEXTENT 11 */
   ALTER  SQLVS350.DDSK11.POOL4.DATA             -
          SHR   (2)                              -
          CAT   (SQLPCAT.USER.CATALOG)
/*
/&
* $$ EOJ
```

Figure 77. ALTER SHR(2) With Names Defined by User

5.2.9.2 Installation Considerations

For VSE, the SHAREOPTIONS value for the SQL/DS directory, data and log data sets must be set to 2. This allows the directory, data and log files to be opened by any number of requests for input processing even if one request is using it for output. But only one request can open the file for output at any one time. Without setting the SHAREOPTIONS values to 2, the database manager and log archives will fail because the second set of ACBs can not be opened.

Refer to *SQL/DS V3R5 Installation for VSE* for more details on how to do this.

5.2.9.3 Coexistence Considerations

There are no coexistence considerations for this function.

5.2.10 Display CICS Information on SHOW CONNECT

5.2.10.1 Migration

There are no migration considerations for this function.

5.2.10.2 Installation Considerations

There are no installation considerations for this function.

5.2.10.3 Coexistence Considerations

The output for the SHOW CONNECT command will be different depending on the level of the code used by the application requester and the application server. The possible combinations are described below:

1. If both the application requester and application server are running Version 3 Release 5 code, then the CICS task number, the RMID and the CICS terminal number will be displayed.
2. If either the application requester or the application server is running Version 3 Release 4 code without APAR PN62248 applied, or is running earlier releases of SQL/DS, then the CICS task number, the RMID and the CICS terminal ID will not be displayed.
3. If the application server is running Version 3 Release 5 code and the application requester is running Version 3 Release 4 code with APAR PN62248 applied, then the CICS task number will be displayed, and the value ′N/A′ will be displayed for the RMID and the CICS terminal ID.
4. If the application server is running Version 3 Release 4 code with APAR PN62248 applied and the application requester is running Version 3 Release 5 code or Version 3 Release 4 code with APAR PN62248 applied, then only the CICS task number will be displayed.

The following matrix summarizes these cases. In the matrix, ′display′ indicates that the field contains a valid value, ′N/A′ means that the value ′N/A′ will be displayed for that field, and ′not displayed′ indicates that the field will not be displayed for the SHOW CONNECT command.

Note: APAR PN62248 has been superseded by APAR PN77565.

Table 20. SHOW CICS INFO from SHOW CONNECT Coexistence Matrix

		Application Requester Level		
Application Server Level	**Field**	**Version 3 Release 5**	**Version 3 Release 4 With APAR PN62248**	**Version 3 Release 4 Without APAR PN62248 or pre-Version 3 Release 4**
Version 3 Release 5	Task no.	display	display	not displayed
	RMID	display	N/A	not displayed
	Term id	display	N/A	not displayed
Version 3 Release 4 With APAR PN62248	Task no.	display	display	not displayed
	RMID	not displayed	not displayed	not displayed
	Term id	not displayed	not displayed	not displayed
Version 3 Release 4 Without APAR PN62248 or pre-Version 3 Release 4	Task no.	not displayed	not displayed	not displayed
	RMID	not displayed	not displayed	not displayed
	Term id	not displayed	not displayed	not displayed

5.2.11 Assembler Even Precision Packed Decimal Support

5.2.11.1 Migration

There are no migration considerations for this function.

5.2.11.2 Installation Considerations

There are no installation considerations for this function.

5.2.11.3 Coexistence Considerations

There are no coexistence considerations for this function.

5.2.12 C/370 Decimal Data Type Support

5.2.12.1 Migration

There are no migration considerations for this function.

5.2.12.2 Installation Considerations

There are no installation considerations for this function.

5.2.12.3 Coexistence Considerations

Requires AD/Cycle C/370 Compiler Version 1 Release 2.

5.2.13 FORCE without DISABLE Produces -948 Instead of -933

5.2.13.1 Migration

There are no migration considerations for this function.

5.2.13.2 Installation Considerations

There are no installation considerations for this function.

5.2.13.3 Coexistence Considerations

There are no coexistence considerations for this function.

5.2.14 Display Package Name on SHOW CONNECT

5.2.14.1 Migration

There are no migration considerations for this function.

5.2.14.2 Installation Considerations

There are no installation considerations for this function.

5.2.14.3 Coexistence Considerations

There are no coexistence considerations for this function.

5.3 Coexistence with Complementary Products

5.3.1 CIMAPPS

SQL/DS V3R5 will include the SQL/DS V3R4 APAR PN43664 which allows VS COBOL II applications using SQL/DS to use the DYNAM compile option.

5.3.2 Coexistence with SQL/DS V2R2 and Later Releases

It is recommended that all databases accessed by SQL/DS V3R5 user machines be migrated to SQL/DS V3R5. However, if database and user levels are different, certain restrictions apply:

1. Packages that are migrated from a previous release of SQL/DS will continue to work with the existing load module or phase, unless they involve a documented incompatibility. (For a description of incompatibilities, consult the appendix F ″SQL/DS Incompatibilities Between Releases″ in the *SQL/DS System Administration* manual, or see 1.4, “Incompatibilities Between Releases” on page 10.)
2. In a database that has been migrated from a previous release, any package that has not been reprepped in Version 3 Release 3 or later will automatically go through a dynamic reprep when it is executed for the first time in Version 3 Release 5. After migration, it is recommended that the DBSU REBIND PACKAGE command be issued for each package, to force a dynamic reprep and avoid the dynamic reprep that may otherwise occur on first invocation of the package.

3. Preprocessing is not supported between an SQL/DS V2R2 user and an SQL/DS V3R5 database, and vice versa.

4. There are certain restrictions when preprocessing an application between an SQL/DS V3R1 user and an SQL/DS V3R5 database, and vice versa. See the incompatibility ″Validation of Host Variables″ in the appendix ″SQL/DS Incompatibilities Between Releases″ in the *SQL/DS System Administration* manual for more details.

5.4 Storage Concerns

5.4.1 VM Saved Segment Sizes

In SQL/DS V3R5, the sizes of the saved segments have changed. The new segment sizes are as follows:

Table 21. Saved Segments for Base Code Only

Component Name	SYSNAME	Size in Kb	# of SYSHRSG	SYSPGCT
Resource Adapter	SQLRMGR	256	4	64
Message Repository	LANGS001	320	5	80
ISQL	SQLISQL	384	6	96
DBSS	SQLSQLDS	1280	20	320
RDS	SQLXRDS	1792	28	448

Table 22. Saved Segments for Base Code with DRDA

Component Name	SYSNAME	Size in Kb	# of SYSHRSG	SYSPGCT
Resource Adapter	SQLRMGR	1152	18	288
Message Repository	LANGS001	320	5	80
ISQL	SQLISQL	384	6	96
DBSS	SQLSQLDS	1280	20	320
RDS	SQLXRDS	2624	41	656

SYSHRSG Segment Size/64KB

SYSPGCT Segment Size/4KB

5.4.2 Minimum VM Machine Sizes

If only the base code has been installed, 7MB are required for the database machine; if the DRDA code has been installed as well, 8MB are required.

The recommended minimum virtual storage increase for a user virtual machine, if DRDA code is installed, is now 995K. For each message repository not in a saved segment add 200 kilobytes to the user machine storage requirement.

5.4.3 Minimum VSE Partition Sizes

The recommended minimum partition size is 5 megabytes, plus 204 kilobytes times the maximum number of concurrently active users.

Chapter 6. Documentation Changes Introduced with SQL/DS V3R5

This chapter describes the documentation requiring changes for SQL/DS V3R5.

The new documentation for SQL/DS V3R5 consists of the following manuals:

1. SQL/DS Installation for IBM VM Systems (GH09-8078)
2. SQL/DS Installation for VSE (GH09-8090)
3. SQL/DS Messages and Codes for IBM VM Systems (SH09-8079)
4. SQL/DS Messages and Codes for VSE (SH09-8091)
5. IBM SQL/DS Data Restore Guide (SC09-2275)
6. SQL/DS Version 3 Release 5 Usage Guide (SG24-4647)

Except for the above mentioned SQL/DS V3R5 manuals, the other previously existing SQL/DS V3R4 manuals will not be modified.

The changes to these unmodified Version 3 Release 4 manuals are identified in this chapter.

6.1 For SQL/DS under VM/ESA

6.1.1 SQL/DS Database Administration for IBM VM Systems

No changes in this manual for this release.

6.1.2 SQL/DS System Administration for IBM VM Systems

6.1.2.1 Chapter 1.Planning for Installation

In the section *Virtual Storage Requirements*, database and user machine size storage requirements have changed. Refer to 5.4, “Storage Concerns” on page 126 for details.

6.1.2.2 Chapter 5.Operating the Online Support for VSE Guest Sharing

Refer to 2.1.1, “CICS Database Switching” on page 11 for details on starting and stopping the online resource adapter with the new CICS transactions.

6.1.2.3 Chapter 11.Using the Accounting Facility

Refer to 2.1.2, “DRDA Limited Accounting String” on page 42 for details on the new DDCS accounting string layout.

6.1.2.4 Chapter 13.Choosing a National Language and Defining Character Sets

In the section *Specifying an IBM-Supplied Character Set at Run Time*, these characters sets are added to the list of IBM-supplied character sets:

- CYRILLIC
- GREEK-423

In the section *CCSID Conversion* the following entries are added to the table of SBCS CCSIDs.

Table 23. SBCS CCSIDs

CCSID	Character Set	Code Page	CHARNAME	Description
423	218	423	GREEK-423	Greek (Coexistence)
1025	1150	1025	CYRILLIC	Cyrillic Multilingual

6.1.2.5 Appendix A. Virtual and Real Storage Requirements

The virtual storage calculation for database machines, as shown in the figure *Initial Storage Requirements of SQL/DS Database Machines*, is updated to account for the dynamically sized HCATs, as follows:

1. The "BASE" line will be changed from 400480 to 400000, due to the elimination of the current static HCATs in YTABLE4.
2. The two lines under the "SQL/DS Buffers" section will both be changed to:

```
Formulas/Constraints                    SUM        MUM
--------------------                    ---        ---
SQL/DS Buffers
* 4 x Maximum of ( the power of 2
  that is >= (NPAGBUF/8) ) or 256)
  + NPAGBUF x 4140                     59040      125344
* 4 x Maximum of ( the power of 2
  that is >= (NDIRBUF/8) ) or 256)
  + NDIRBUF x 560                       8864       17824
```

6.1.2.6 Appendix C. Maximum Values

The maximum values for NDIRBUF and NPAGBUF are updated from 28000 and 3500 respectively, to 400000 each.

6.1.3 SQL/DS Application Programming for IBM VM Systems

6.1.3.1 Appendix A. Using SQL in Assembler

The section titled *Declaring Host Variables* contains the following list of examples to which even precision decimal declarations should be added.

F

F'5'

H

H'100'

CL255

CL5'ABCDE'

H,CL5

H'5',CL5'ABCDE'

D

D'2.5E10'

PL2

PL5'123.45'

P′123′

P′123.45′

P′1234′

P′123.456′

H,CL32767

The section *Defining SQL/DS Data Types for Assembler Language* contains a table mapping the ″SQL/DS Keyword″ to the ″Equivalent Assembler Declaration″. The description of the equivalent Assembler declaration for the SQL/DS keyword **DECIMAL**[**(p**[**,s**]**)**] **or DEC**[**(p**[**,s**]**)**] must be changed to read as follows:

PLn[′**decimal constant**′]

or

P′**decimal constant**′

For declarations using PL, the precision is 2n-1 (n is the number of bytes). For declarations using P, the length of the decimal constant, excluding the decimal point and sign, is the precision. For declarations using P or PL, the scale is that of the decimal constant. If the constant is not specified (for declarations using PL only), the scale is 0.

In the section *Defining SQL/DS Data Types for Assembler Language*, there is a note stating that assembler does not support even precision packed decimal variables. This note **must** be ignored.

6.1.3.2 Appendix B. Using SQL in C

The section titled *Declaring Host Variables* contains the following list to which **decimal** must be added.

The definition of a host variable is subject to the following rule:

- A data object declared as a scalar variable or structure element may have any one of the following basic C datatypes:

  short Short integer

  long Long integer

  float Floating-point

  double Double-precision floating-point

  decimal **Decimal**

  The keyword **int** is optional in the declaration of ...

The section titled *Defining SQL/DS Data Types for C* contains a table mapping the ″SQL/DS Keyword″ to the ″Equivalent C Declaration″. The description of the equivalent C declaration for the SQL/DS keyword **DECIMAL**[**(p**[**,s**]**)**] **or DEC**[**(p**[**,s**]**)**] must be changed to read as follows:

decimal(p,s)

If your version of the C compiler does not provide support for the decimal datatype, C short, long, float and double host variables are supported for conversion to and from DECIMAL columns.

To preserve decimal places: if p<7 use float; else use double.

6.1.4 SQL/DS Database Services Utility for IBM VM Systems

No changes in this manual for this release.

6.1.5 SQL/DS Operation for IBM VM Systems

6.1.5.1 Chapter 2. SQL/DS Online Support for VSE Guest Sharing

- The following description for **Task No.** must be added after the description of ′CPU time′ for the SHOW CONNECT command:

  This is the CICS task number for VSE CICS users accessing the VM database via guest sharing. The task number can be used to identify which agent should be forced before terminating the CICS transaction. This field is not displayed if there are no CICS tasks associated with the agent (for example, when an agent is not in work (NIW)).

- The following description for **RMID** must be added to the SHOW CONNECT command:

  This is the Resource Manager id specified by the CIRB transaction for VSE guest sharing users. This identifies which resource adapter is used by the agent, since multiple resource adapters may exist. The RMID can be used, along with the CICS task number, to identify which agent should be forced before terminating a CICS transaction.

- The following description for **Term. id** must be added to the SHOW CONNECT command:

  This is the CICS terminal id for VSE guest sharing users. This identifies which CICS terminal is accessed by the CICS user. The terminal id can be used, along with the CICS task number and the RMID, to identify which agent should be forced before terminating a CICS transaction. The value ′N/A′, indicating that the terminal id is not available, is displayed if there is no terminal id associated with the agent.

6.1.5.2 Appendix A. SQL/DS Initialization Parameters

The maximum values for NDIRBUF and NPAGBUF must be updated in the table in this appendix, from 28000 and 3500 respectively, to 400000 each.

6.1.6 SQL/DS Diagnosis Guide and Reference for IBM VM Systems

6.1.6.1 Appendix A. RDIIN

Figure 144 must be changed to the following:

```
Dec(Hex) RDIEXT

 0 (0)     RDIEXTEC - eyecatcher †RDIEXT†

 8 (8)     RDIEXTL - length of RDIEXT     RDIEXTA - pointer declarations
                                            RDIDBNMP (1) - points to the
                                                           database name

16(10)     RDICONSP - points to the       RDIBPOPT - points to the bind
                      consistency token               prep options

24(18)     RDIPIPDP (2) - points to the   RDICICSP - points to the CICS
                          connect indicator           information required
                                                      by the SHOW CONNECT
                                                      command
32(20)
                Reserved
40(28)
                Reserved
48(30)
```

Figure 148 must be changed to the following:

```
 0  record type ¢CA¢

 2  agent number

 6  work status (0-NIW, 1-NEW, 2-R/O, 3-R/W)

 7  subsystem or application (0-APPL, 1-SUBS)

 8  agent processing status (1-not processing SQL operator command
                             2-processing SQL operator command
                             3-not processing and in wait
                             4-processing and in wait
                             5-waiting log archive checkpoint
                             6-processing LPAGEBUF RSCP)

 9  wait status (4-communication, 5-lock, 6-checkpoint, 7-out of page,
                 8-out of block, 9-I/O, A-WITH LPAGE...)

10  filler

12  tranid (fixed)

16  resource consumption (fixed)

20  rollback/commit type (R-rollback, C-commit or blank)

21  rollback/commit status (3-active, 4-scheduled)

22  rollback/commit requester (4-scheduled, 5-user, 6-system, 8-lock limit,
                               9-DBSS limit)

23  filler

24  package creator

32  package name

40  package section number (fixed)

42  CICS task number (valid only if offset 7 is 1-SUBS and
                      offset 6 is not 0-NIW)

50  RMID (valid only when CICS task number is valid)

53  CICS terminal id (valid only when CICS task number is valid)

61  unused
```

6.1.7 SQL/DS VM Data Spaces Support for VM/ESA

No changes in this manual.

6.2 For SQL/DS Under VSE/ESA

6.2.1 SQL/DS Database Administration for VSE

No changes required in this manual.

6.2.2 SQL/DS System Administration for VSE

6.2.2.1 Chapter 1.Planning for Installation

In the section *Virtual Storage Requirements*, database and user machine size storage requirements have changed. Refer to 5.4, “Storage Concerns” on page 126 for details.

6.2.2.2 Chapter 5.Operating the Online Support

Refer to 2.1.1, “CICS Database Switching” on page 11 for details on starting and stopping the online resource adapter with the new CICS transactions.

6.2.2.3 Chapter 10.Using the Accounting Facility

Refer to 2.1.2, “DRDA Limited Accounting String” on page 42 for details on the new DDCS accounting string layout.

6.2.2.4 Chapter 12.Choosing a National Language and Defining Character Sets

In the section *Specifying an IBM-Supplied Character Set at Run Time*, these characters sets are added to the list of IBM-supplied character sets:

- CYRILLIC
- GREEK-423

In the section *CCSID Conversion* the following entries are added to the table of SBCS CCSIDs.

Table 24. SBCS CCSIDs

CCSID	Character Set	Code Page	CHARNAME	Description
423	218	423	GREEK-423	Greek (Coexistence)
1025	1150	1025	CYRILLIC	Cyrillic Multilingual

6.2.2.5 Appendix A. Processor Storage Requirements

The virtual storage calculation for database partitions, as shown in the figure *Initial Storage Requirements of SQL/DS Database Partitions*, is updated to account for the dynamically sized HCATs, as follows:

1. The “BASE” line will be changed from 400480 to 400000, due to the elimination of the current static HCATs in YTABLE4.
2. The two lines under the “SQL/DS Buffers” section will both be changed to:

```
Formulas/Constraints                    SUM        MUM
--------------------                    ---        ---
SQL/DS Buffers
* 4 x Maximum of ( the power of 2
  that is >= (NPAGBUF/8) ) or 256)
  + NPAGBUF x 4140                     59040      125344
* 4 x Maximum of ( the power of 2
  that is >= (NDIRBUF/8) ) or 256)
  + NDIRBUF x 560                       8864       17824
```

6.2.3 SQL/DS Application Programming for VSE

6.2.3.1 Chapter 10. Special Topics

Add these comments to the following subsection

Condition Handling with LE/VSE: The SQL/DS environment is sensitive to errors or conditions. A failing SQL/DS transaction or application can potentially leave an SQL/DS database in an inconsistent state. For this reason, it is essential that SQL/DS knows about the failure of a transaction or application that has been updating a database so that it can perform database rollback.

When a user runs an application with the TRAP(ON) run-time option of LE/VSE and the application is running in Single User Mode, LE/VSE and SQL/DS keep track of calls to and returns from SQL/DS. If a program interrupt or abend occurs when the application is running, the LE/VSE condition manager is informed whether the problem occurred in the application or in SQL/DS. If the program interrupt or abend occurs in SQL/DS, the LE/VSE condition handler passes the condition back to SQL/DS.

If a program interrupt or abend occurs in the application outside SQL/DS, the LE/VSE condition manager will perform its own condition handling actions. If the condition manager gets control then the user **must** do one of the following:

- Resolve the error completely so that the application can continue.
- Make sure that the application terminates abnormally by using the ABTERMENC(ABEND) run-time option of LE/VSE to transform all abnormal terminations into operating system abends in order to cause SQL/DS to do the necessary recovery processing when the SQL/DS server is warm started.

  Note: The following methods are available for specifying any LE/VSE run-time options, including ABTERMENC(ABEND):

  1. As an installation wide default via the CEEDOPT assembler language source file.
  2. In the assembler user exit routine CEEBXITA.
  3. As an application default via the CEEUOPT assembler language source file. CEEUOPT is assembled into an object module which is linked with the application program.
  4. In JCL via the PARM parameter of the JCL EXEC statement.
  5. In PL/I source code via the PLIXOPT string.

  See LE/VSE documentation for more details.

- Provide a modified run-time assembler user exit (CEEBXITA) that transforms all abnormal terminations into operating system abends. The assembler user exit should check the return code and reason code or the CEEAUE_ABTERM bit, and request an abend by setting the CEEAUE_ABND flag to ON, if appropriate. **Note.** The CEEBXITA assembler user exit is

intended for use by the application programmer. It is not intended for SQL/DS use. See LE/VSE documentation for more details.

6.2.3.2 Appendix A. Using SQL in Assembler

The section titled *Declaring Host Variables* contains the following list of examples to which even precision decimal declarations should be added.

F

F′5′

H

H′100′

CL255

CL5′ABCDE′

H,CL5

H′5′,CL5′ABCDE′

D

D′2.5E10′

PL2

PL5′123.45′

P′123′

P′123.45′

P′1234′

P′123.456′

H,CL32767

The section *Defining SQL/DS Data Types for Assembler Language* contains a table mapping the ″SQL/DS Keyword″ to the ″Equivalent Assembler Declaration″. The description of the equivalent Assembler declaration for the SQL/DS keyword **DECIMAL**[**(p**[**,s**]**)**] **or DEC**[**(p**[**,s**]**)**] must be changed to read as follows:

PLn[′**decimal constant**′]

or

P′**decimal constant**′

For declarations using PL, the precision is 2n-1 (n is the number of bytes). For declarations using P, the length of the decimal constant, excluding the decimal point and sign, is the precision. For declarations using P or PL, the scale is that of the decimal constant. If the constant is not specified (for declarations using PL only), the scale is 0.

In the section *Defining SQL/DS Data Types for Assembler Language*, there is a note stating that assembler does not support even precision packed decimal variables. This note **must** be ignored.

6.2.3.3 Appendix B. Using SQL in C

The section titled *Declaring Host Variables* contains the following list to which **decimal** must be added.

The definition of a host variable is subject to the following rule:

- A data object declared as a scalar variable or structure element may have any one of the following basic C datatypes:

  short Short integer

  long Long integer

  float Floating-point

  double Double-precision floating-point

  decimal **Decimal**

  The keyword **int** is optional in the declaration of ...

The section titled *Defining SQL/DS Data Types for C* contains a table mapping the ″SQL/DS Keyword″ to the ″Equivalent C Declaration″. The description of the equivalent C declaration for the SQL/DS keyword **DECIMAL**[**(p**[**,s**]**)**] **or DEC**[**(p**[**,s**]**)**] must be changed to read as follows:

decimal(p,s)

If your version of the C compiler does not provide support for the decimal datatype, C short, long, float and double host variables are supported for conversion to and from DECIMAL columns.

To preserve decimal places: if p<7 use float; else use double.

6.2.4 ISQL Guide and Reference for VSE

6.2.4.1 ISQL Commands

In the ISQL SET PRINTRoute command, another option has been added. You can now specify TOUser to identify the VSE POWER user to whom the output will be spooled. Refer to 2.3.5, “ISQL Print Routing” on page 79 for details of the command syntax.

6.2.5 SQL/DS Database Services Utility for VSE

No changes in this manual.

6.2.6 SQL/DS Operation for VSE

6.2.6.1 Starting and Stopping SQL/DS Online Support

Refer to 2.1.1.1, “CICS Transactions” on page 11 for details of the new CIRA and CIRC transactions, and the enhanced CIRB and CIRD transactions.

6.2.6.2 Operating the Application Server

The **SHOW CONNECT** command has changed to include Term id and RMID. Figure 78 shows the output of the SHOW CONNECT command.

```
MSG F4
AR 015 1I40I  READY
F4 004 ARI0062A SQLDS:
F4 004          Enter an SQL/DS operator command.
4 show connect
F4 004 Status of Connected SQL/DS Users             1994-12-02 15:23
F4 004    Checkpoint agent is not active.
F4 004    User Agent:   1   User-ID: SAMUEL   SQL-ID: SAMUEL
F4 004       is R/O  APPL  12BCF
F4 004       Agent is processing and is in communication  wait.
F4 004            State started: 1994-12-02  15:21:22
F4 004            Conversation started: 1994-12-02  15:21:22
F4 004            Task no.: 147      RMID:  32 Term. id: 077D
F4 004    User Agent:   2   User-ID: MARISSA  SQL-ID: MARISSA
F4 004       is R/W  APPL  12FC4
F4 004       Agent is processing and is in communication  wait.
F4 004            State started: 1994-12-02  15:23:17
F4 004            Conversation started: 1994-12-02  15:23:15
F4 004            CPU time: 00:00:01
F4 004            LUWID: CAIBMOML.OECGW001.A6773D6F8611.0001
F4 004            EXTNAM: MARISSA.1
F4 004            Requester: SQLDS/VM  V3.3.0   at TOIVMLB6
F4 004            Rmtuser ID: 2
F4 004            LU name: OMPGW001
F4 004            Task no.: 0000134
F4 004    User Agent:   3   User-ID: LAURA    SQL-ID: LAURA
F4 004       is R/O  APPL  12FC6
F4 004       Agent is processing and is in I/O  wait.
F4 004            State started: 1994-12-02  15:23:24
F4 004            Conversation started: 1994-12-02  15:23:15
F4 004            CPU time: 00:00:01
F4 004            LUWID: CAIBMOML.OECGW001.A6773D6E52C9.0001
F4 004            EXTNAM: LAURA.1
F4 004            Requester: SQLDS/VM  V3.3.0   at TOIVMLB6
F4 004            Rmtuser ID: 3
F4 004            LU name: OMPGW001
F4 004            Task no.: 0000133
F4 004    User ID: ANDREW    SQL ID: ANDREW
F4 004       User is inactive.
F4 004            State started: 1994-12-02  14:58:18
F4 004            Conversation started: 1994-12-02  14:58:16
F4 004            CPU time: 00:00:00
F4 004            LUWID: CAIBMOML.OECGW001.A67737DC6BEB.0001
F4 004            EXTNAM: ANDREW.1
F4 004            Requester: SQLDS/VM  V3.3.0   at TOIVMLB6
F4 004            Rmtuser ID: 1
F4 004            LU name: OMPGW001
F4 004            Task no.: 0000110
F4 004   3  SQL/DS remote users are active.
F4 004   1  SQL/DS remote users are inactive.
F4 004   1  SQL/DS agents are available.
F4 004   1  SQL/DS remote connections are available.
F4 004 ARI0065I SQL/DS operator command processing is complete.
```

Figure 78. SHOW CONNECT Command

The description for **Task No.** in the SHOW CONNECT command has changed to the following:

This is the CICS task number for CICS users, or the CICS task number of a connected APPC-to-XPCC Exchange Transaction (AXE) for remote (DRDA) users. For CICS users, the task number can be used to identify which agent should be forced before terminating the CICS transaction. For remote users, the task number, together with the LU Name, can be used to terminate a particular remote user-initiated AXE transaction by using the CICS CEMT transaction, thereby forcing a specific remote user to end his work and terminate the conversation. This field is not displayed if there are no CICS tasks associated with the agent (for example, when an agent is not in work (NIW)).

The following description for **RMID** has been added to the SHOW CONNECT command:

This is the Resource Manager id specified by the CIRB transaction. This identifies which resource adapter is used by the agent, since multiple resource adapters may exist. The RMID can be used, along with the task number and the CICS terminal id, to identify which agent should be forced before terminating a CICS transaction.

The following description for **Term. id** has been added to the SHOW CONNECT command:

This is the CICS terminal id, which identifies which CICS terminal is accessed by the user. The CICS terminal id can be used, along with the RMID and the task number, to identify which agent should be forced before terminating a CICS transaction. The value 'N/A', indicating that the terminal id is not available, is displayed if there is no terminal id associated with the agent.

6.2.6.3 Appendix A. SQL/DS Initialization Parameters

The maximum values for NDIRBUF and NPAGBUF must be updated in the table in this appendix, from 28000 and 3500 respectively, to 400000 each.

6.2.7 SQL/DS Diagnosis Guide and Reference for VSE

6.2.7.1 Appendix A. RDIIN

Figure 139 on page 273 will be changed to the following:

```
Dec(Hex) RDIEXT

 0 (0)     RDIEXTEC - eyecatcher †RDIEXT†

 8 (8)     RDIEXTL - length of RDIEXT     RDIEXTA - pointer declarations
                                            RDIDBNMP (1) - points to the
                                                           database name

16(10)     RDICONSP - points to the       RDIBPOPT - points to the bind
                      consistency token               prep options

24(18)     RDIPIPDP (2) - points to the   RDICICSP - points to the CICS
                          connect indicator           information required
                                                      by the SHOW CONNECT
                                                      command
32(20)
                Reserved
40(28)
                Reserved
48(30)
```

6.3 For SQL/DS Under Both VSE/ESA and VM/ESA

6.3.1 SQL/DS Performance Tuning Handbook for IBM VM Systems and VSE

6.3.1.1 Chapter 2. Measuring Performance

The SHOW CONNECT operator command now has more information for VSE users. It also displays:

- The CICS terminal ID
- The Resource Manager ID (RMID)

Refer to 2.3.3, “CICS Information in SHOW CONNECT” on page 76 for details on the enhanced SHOW CONNECT command.

6.3.2 SQL Reference for IBM VM Systems and VSE

There are no changes for this manual.

Chapter 7. SQL/DS Client/Server Solutions

IBM's Distributed Relational Database Architecture (DRDA) provides a standard way to access data and a way to use many client/server tools and applications. DRDA allows relational database products to communicate with each other and allows non-relational database products to access data in relational databases. It does this by providing a standard format for data exchange and the rules governing the protocol for that exchange. DRDA makes the location of the data transparent to the end user. SQL/DS opens the door to quicker development of distributed solutions and easier-to-use applications. Developers and end users can benefit from using popular PC programs and tools while the company's data remains secure on the enterprise server. With SQL/DS V3R3 and above, users can access data wherever it is.

7.1 Distributing Data with SQL/DS

Today, the DB2 family consists of the following base products:

DRDA Servers	Requesters	Client Application Enablers	Communications Protocols
SQL/DS (VSE)	SQL/DS (VM)	DB2 for AIX	LU 6.2
SQL/DS (VM)	DB2 for MVS	DB2 for OS/2	APPC/APPN
DB2 for MVS	DB2 for OS/400	DB2 for Windows	NetBIOS
DB2 for OS/400		DB2 for DOS	TCP/IP
DB2 for AIX		DB2 for HP UX	IPX/SPX
DB2 for OS/2		DB2 for Solaris	
DB2 for Solaris			
DB2 for HP UX			

Figure 79. DB2 Family Base Products

The family also includes complementary products such as IBM's Data Replication solutions or our Query solutions including the new Visualizer product, and the DB2 Family continues to grow. Any requester can be connected to any server with the appropriate communications protocol, and as business needs change, the system implementation can change with little impact to the existing applications. The DB2 Family solution opens the door to "mixing and matching" servers and requesters that meet the business needs. Also DRDA protocols along with products such as DataJoiner mean that non-IBM solutions can be added just as easily.

7.2 How to Make Data Available?

Accessing corporate data quickly and easily with workstation tools is a growing need for today's businesses. Today, SQL/DS DBAs have a better solution for their PC users, with direct access to the SQL/DS database. Using the SQL/DS solution provides support for:

- Many desktop operating systems that are already installed, and the applications running on them

- A wide range of communication protocols implemented by the different departmental LANs
- Different access methods to retrieve the data
- A variety of servers and operating systems used as local servers.

SQL/DS uses IBM's DRDA solution to provide the answers to these needs. The solution consists of DB2 Client (Client Application Enabler), Distributed Database Connection Services (DDCS), SQL/DS, and an SNA LU 6.2 connection between DDCS and SQL/DS. SQL/DS customers can have a DRDA solution by simply adding DDCS and DB2 Client to their enterprise. DDCS gives the workstation users transparent access to SQL/DS. DB2 Client enables them to send data requests to SQL/DS via DDCS.

Figure 79 on page 143 shows which parts of IBM's DRDA solution can be added to an SQL/DS environment. The DB2 workstation servers can be DataJoiner on AIX, or a local DB2 database or a DDCS server on OS/2, AIX, or another UNIX platform. DRDA host servers can be DB2 for MVS/ESA, DB2 for AS/400 or any other IBM DB2 family database that can be a DRDA server just as SQL/DS can in Figure 80.

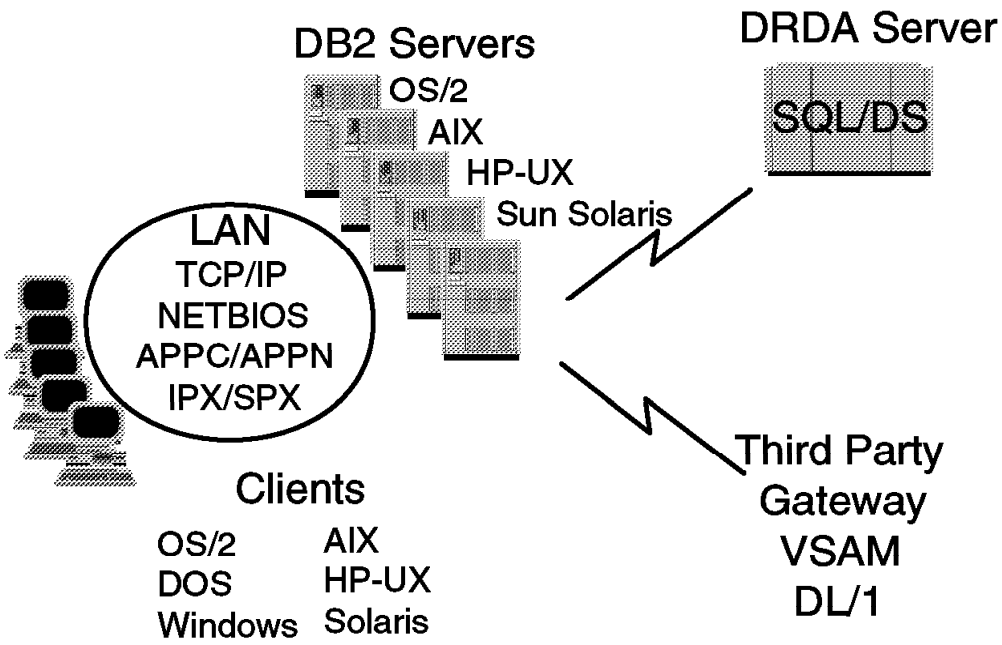


Figure 80. Connecting the Enterprise

SQL/DS can be used as both a DRDA server and DRDA requester.

7.2.1 Who Can Access SQL/DS Data?

Anyone running DB2 Client can access SQL/DS data. In fact, any DRDA application requester can access SQL/DS. DB2 Client provides run-time support to allow applications to access local or remote database servers. For example, PC users running spreadsheet programs can access DB2 family databases when they use DB2 Client. SQL/DS is one of the remote database servers that applications can transparently access when you use DB2 Client. DB2 Client is available for popular platforms such as OS/2, DOS, Windows, AIX, HP-UX, and the Sun Solaris operating environment. DB2 Clients can communicate with DDCS servers through many popular communication protocols. The protocols supported are TCP/IP, NetBIOS, APPC, and IPX/SPX. Table 25 lists the combinations of communication protocols, DB2 Client clients, and DDCS servers that are supported today.

Table 25. Supported Communication Protocols

DB2 Clients	Protocols	DDCS OS/2	DDCS AIX	DDCS HP-UX	DDCS Sun Solaris
DOS V1.2	NetBios IPX/SPX TCP/IP APPC	Yes Yes Yes No	No Yes(1) Yes No	No Yes(2) Yes No	No Yes(2) Yes No
OS/2 V2.1	NetBios IPX/SPX TCP/IP APPC	Yes Yes Yes Yes(3)	No Yes Yes Yes	No No Yes No	No No Yes No
Windows V2.1	NetBios IPX/SPX TCP/IP APPC	Yes Yes Yes No	No Yes Yes No	No Yes Yes No	No Yes Yes No
AIX V2.1	NetBios IPX/SPX TCP/IP APPC	No No Yes No	No No Yes No	No No Yes No	No No Yes No
HP-UX V2.1	NetBios IPX/SPX TCP/IP APPC	No No Yes No	No No Yes No	No No Yes No	No No Yes No
Solaris V2.1	NetBios IPX/SPX TCP/IP APPC	No No Yes No	No No Yes No	No No Yes No	No No Yes No

1. Supported natively by DB2 for AIX, and also provided by FireFox, Inc. NOV*IX for NetWare product.
2. Provided by FireFox, Inc. NOV*IX for NetWare product.
3. If the SNA network supports APPN, DB2 Client for OS/2 can also participate in the APPN network.

7.2.2 How DB2 Clients Can Access SQL/DS

DDCS provides PC and UNIX users with transparent read and update access to enterprise data stored in an SQL/DS server utilizing the DRDA protocol. DDCS supports X/Open's CLI standard, and is available on all of the DB2 Unix and PC server platforms such as OS/2, AIX, and the HP-UX and the Sun Solaris operating environment. As discussed in the preceding section, it also supports many popular communication protocols. This allows you to connect your PC or UNIX client operating systems to a DDCS server. IBM's DB2 CLI (two call-level interface), widens your application choices to include non-Windows solutions on a variety of Intel and UNIX platforms. Figure 81 shows how TCP/IP provides you with the greatest selection of client operating systems for DB2 Client.

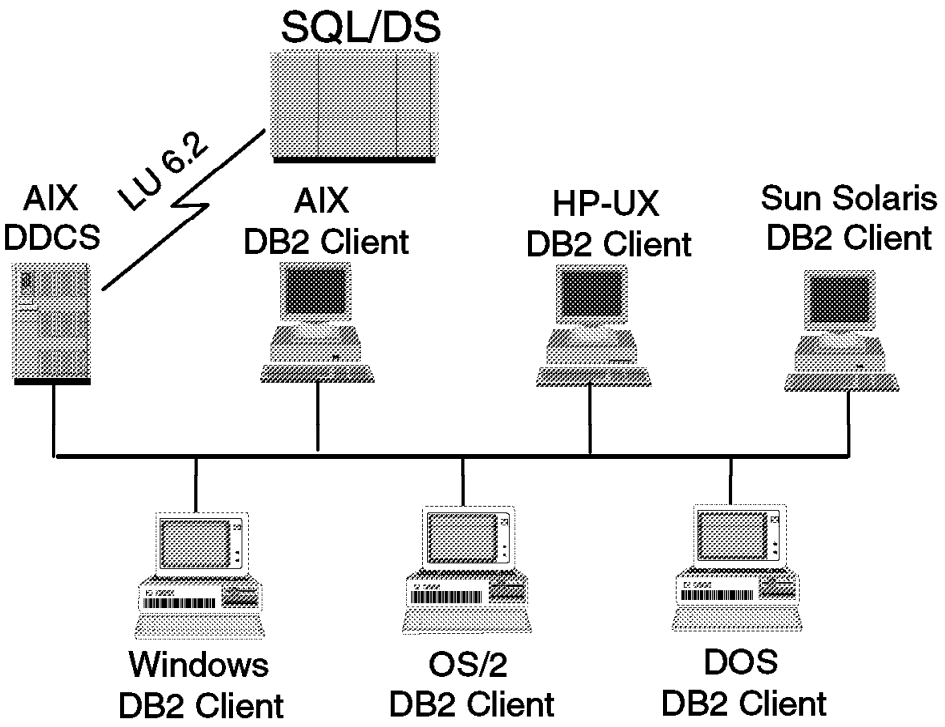


Figure 81. Client Support for SQL/DS

7.2.3 Setting Up a High-Performance Connection

An LU 6.2 connection can give you the fastest access to SQL/DS. The AIX SNA Server/6000 has an SNA Channel Connectivity feature that provides SNA support for the direct attachment of an RS/6000 workstation to an IBM System/390 host. Support is included for the Block Multiplexer and the ESCON channel environments. Connecting DDCS to SQL/DS with these features will give a fast DDCS gateway.

Customers are implementing this type of connection to SQL/DS because performance is critical when their users are accessing data. For example, a customer in the manufacturing sector has switched from a third-party gateway because of dissatisfaction with its access time to SQL/DS. They are now using

an SNA connection from DDCS to SQL/DS to improve their access time to SQL/DS.

7.3 How to Enable Application Solutions?

In the past, application developers on VSE and VM who developed COBOL, FORTRAN, and C applications that accessed SQL/DS had a choice between dynamic or static SQL requests with SQL/DS. Usually, the programmer chose static SQL requests because it provided better application performance. Generating static SQL requests required a precompiler. Prior to DRDA, SQL/DS precompilers for languages such as COBOL, FORTRAN, and C were only available on VSE and VM. Many developers felt that it was not feasible to run a COBOL application on the PC if they could only issue dynamic SQL requests to SQL/DS, because the PC application would not deliver the performance that people are used to.

Today, the DB2 family provides precompilers for PC versions of COBOL, FORTRAN, C, and C++ application development tools. These precompilers support SQL/DS, and are among the many features found in the DB2 Software Developer′s Kit (SDK).

The DB2 SDK is a collection of tools that helps create database applications. It includes:

- Precompilers for C, C++, COBOL, and FORTRAN
- An API for implementing precompiler support for compilers not specifically supported by the DB2 SDK precompiler
- DB2 Client (Client Application Enabler) to provide client function
- Programming libraries, header files, code samples, and a complete set of documentation for developing applications using the DB2 CLI (Call Level Interface) or embedded SQL
- Interactive SQL.

A DB2 SDK is available for each of the DB2 Family PC and workstation clients. Together, DB2 SDK and DB2 Client provide application developers with three methods for accessing DB2 Family data:

1. Embedded SQL
2. DB2 CLI
3. ODBC for Windows applications

With DB2 SDK, you can develop applications that will access data on SQL/DS, a local DB2 database, or both. Applications developed with DB2 SDK can access a local DB2 database and an SQL/DS database in a single LUW (logical unit of work). This can be done when the local DB2 database has DRDA 2 (two-phase commit) support and only one database is being updated within the LUW, that is, multi-read, single-update.

DRDA 2 is known as Distributed Unit of Work. It allows an application to send data manipulation requests to multiple databases within a single LUW. Furthermore, the application can update more than one database within the LUW if each database supports DRDA 2. This is because DRDA 2 supports the two-phase commit process, which ensures all of the updates are committed to each database at the same time to maintain transaction integrity. DRDA 2

support allows a programmer to write code that can update the customer table in the headquarters database and update the orders table in the regional database with a single transaction (also known as an LUW). Setting up a distributed database gives more flexibility when developing applications. VM applications can access DB2 Version 2 databases on OS/2 or UNIX.

7.4 How to Build Distributed Database Applications

The key to build successful distributed data applications is to understand:

- Who needs to access the information

In some cases multiple sites will need to access data. Once it is known who needs access, it is possible to decide where to put the data and the application logic in order to provide users with the shortest, fastest path to the information they need.

- How many people will need access how often

Knowing how many people will access the data and how frequently lets you anticipate the demand on the network from their inquiries, and helps you decide where to put the data for optimal access. If there are many queries on one database from one department, you may want to provide them with a copy of the main database on a local workstation server to reduce the network demands.

7.4.1 Deciding Where the Data Should Reside

Deciding where to locate the data is one of the first decisions you need to make when implementing distributed database solutions. This decision itself can be broken into two parts:

1. Where to locate the source database
2. Whether to use remote access or data replication

To make these decisions you need to understand who uses the data, how often, and for what purpose.

There are many considerations for locating the source database, but the primary factors involve the number of people who will need to access it and how significant the resulting demand will be on your network. Table 26 on page 149 lists some of the considerations that may influence the decision of where to locate the source data:

Table 26. Choosing Between Local and Central Data Servers		
Consideration	**Local Server**	**Central Server**
Number of people accessing data	Few	Many
Physical location	Close together, local	Far apart, global
Logical location	Together	Apart
Need different subsets of the source information	Yes	No
Availability of information even if network or central server unavailable	Yes	No
Database combines input from multiple sources	No	Yes
Local data propagated to central database	Yes	No

Often when you are considering implementing distributed solutions you need to choose between a solution with data replication and one with remote access. Determining the optimum solution for your users depends on a number of factors; however, the key to deciding which one is best is very dependent on frequency of access. Typically, infrequent access and/or a strong data currency requirement for up-to-the-second information means a Remote Access solution. If your major concerns are about off-loading the host to reduce network demands and availability of information regardless of the status of the rest of the system, a data replication solution is probably right. Each application may be different. Table 27 shows the kinds of requirements you need to consider and how they will affect your decision.

Table 27. Choosing Between Data Replication and Remote Access		
Requirement	**Frequent Access**	**Infrequent Access**
Read Only Access	Data Replication	Remote Access
Network Traffic Management	Data Replication	Remote Access
Off-load the host	Data Replication	Data Replication
Up-to-the-second data currency for operational applications	Remote Access	Remote Access
Availability of information even if network or server unavailable	Data Replication	Data Replication
Information for ad hoc decision support	Data Replication	Remote Access
Only need a subset of the data available	Data Replication	Remote Access
Standard set of customized information derived from the data source	Data Replication	Remote Access

7.4.2 Deciding Where the Application Logic Should Reside

You also need to consider where your application logic should execute for each of your solutions. Many of the considerations in this decision are similar to the ones you considered when deciding where the data should reside. Distributing the application logic does not automatically mean that it should move from the central server to a local server. How often the application will execute and its impact on the network play a key role in deciding where to locate the application

logic. Table 28 on page 150 shows some of the areas you will have to consider as you make your decision.

Table 28. Choosing Where to Execute Application Logic

Consideration	Frequent Execution	Infrequent Execution
Network traffic performance	Local	Central
Existing central server application	Local	Central
Security	Central	Central
Data replicated to local server	Local or Central	Central
Data source moved to local server	Local	Local
New application	Local	Local or Central
Widely dispersed users	Central	Central
Application needs to be available independent of the status of the rest of the network	Local	Local

7.4.3 Application Development Environment

Distributed solutions apply to your Application Development Environment as well. You can take advantage of new workstation tools and Rapid Application Development techniques to build your new distributed solutions and to maintain your existing host-based applications.

For example, VSE customers can use the Distributed Workstation Feature with IBM′s VisualGen to develop code with a drag-and-drop, point-and-click interface. Built-in change control procedures allow you to check out source code from the mainframe and work with it on the workstation.

In addition, new solutions such as IBM′s Visualizer let you put query and report development into the hands of the end users. Other products such as IBM′s VisualAge help you build new object-oriented solutions.

7.5 An Example of Implementing Distributed Solutions

For client/server and distributed data implementations to be successful, there must be an understanding of the enterprise needs and how those needs are interconnected, and a cohesive plan that stages the delivery of the required solutions.

Leveraging existing investments lets stage the implementation of distributed solutions so costs and potential disruption can be minimized. Let′s analyze step by step this implementation with the next example.

7.5.1 Retail Customer Environment

Consider how an SQL/DS customer who runs a retail franchise company could benefit from a distributed system. This customer runs an operation that centralizes most functions to improve productivity and ensure consistency throughout the company. Any store looks the same in each city in the world where the company does business. Headquarters determines which products the stores will carry and how they will be marketed, and ensures that this information is available to Customer Service in each store. Responding quickly

to marketplace trends provides the company and stores with a competitive edge. Competition is fierce and the customer wants to identify market trends first, to retain a leadership position.

The key priorities of the business are re-engineering and streamlining the business processes while keeping a close eye on costs. For the Information Services department, this means implementing new solutions quickly while maximizing the use of the existing systems and ensuring flexibility to respond to future, unanticipated business changes.

7.5.2 Current Installation Configuration

Currently the company has an ES/9000 mainframe with VSE/ESA V1R3 installed. SQL/DS V3R1 provides robust database capabilities. I/S provides centralized Data, Systems and Application support. All the corporate data resides on the ES/9000, including databases for Sales and Inventory.

Employees have PCs with personal productivity tools installed, such as spreadsheets and word processors to manipulate data and obtain information. More and more they are finding that they also need to access corporate information so they can make better decisions faster, but they want to use their workstation tools to do so. The lack of easy access to this information is increasing frustration. Consequently, it is decided to start building departmental applications that meet the need to share information with each other, and so that all employees can have the necessary information at their fingertips.

Increasingly, the need to use information is driving the dispersal of corporate information assets throughout the enterprise. Often, data is entered into multiple systems, which potentially introduces errors, creates discrepancies between data sources, and generally reduces productivity.

7.5.3 Current Situation: Inability to Share Data

Groups throughout the enterprise have examined their business processes, and ask I/S to provide some solutions to deliver information to two key business areas: Product Acquisition and Merchandising. Together these areas are the lifeblood of the organization, as their decisions result in a profit or loss situation.

The Product Buyers decide which new products to buy, which to discontinue, and how many of each item to order. They base their decisions on rapidly changing marketplace trends and buying patterns. The faster they can identify emerging patterns, the faster they can respond. To improve their analysis process, they developed a set of workstation applications to analyze the sales data. However, the information is always a week old by the time it is available and re-entered into their applications. They would like to get information faster, and to know that the stores have it available electronically.

The Merchandising Managers develop marketing programs to drive product sales. The Buyers provide them with product information and general buying trends, and the Store Managers provide them with weekly sales information. If they had access to the Store Managers′ knowledge of detailed demographic information and local buying patterns, they could provide tailored programs to the various areas.

After talking with the groups involved, I/S quickly identifies that the key problems to be addressed are:

- Inability to share electronic data between different platforms
- Inability to share electronic data between remote systems
- Islands of data throughout the enterprise
- End-user frustration
- Demand for workstation tools.

They need to solve these problems quickly within their budget constraints and using existing system investments wherever possible.

7.5.4 Solution: Connect the Data

They decide that the best solution is to connect the islands of data so everyone can access the information they need, and to enable the distribution of data to the people who need it. They do this quickly with these steps:

1. Upgrade from SQL/DS V3R1 to SQL/DS V3R5

This enables the enterprise to distribute the data, and is the first half of the bridge to distributed data.

2. Install DDCS and DB2 Client on the workstations

DDCS turns the mainframe into an enterprise server. Here, it connects SQL/DS V3R5 with DB2 for OS/2 Version 2 to provide an architected solution for sharing and distributing data. The Product Buyers can now bring the query results into their workstation applications. The Store Managers can query the corporate inventory database to locate a specific product that a customer wants. The Merchandising Managers and the Product Buyers now have access to the local customer demographics for each store. The Product Buyers alter their applications to look at trends on a more detailed level.

3. Install DataPropagator-Relational on the DB2 for OS/2 workstations at each store

This lets them update the sales information automatically. They choose to update the sales figures each night, which provides the Product Buyers and the Merchandising Managers with accurate information so they can respond quickly to buying trends. Each store configures DataPropagator-Relational to send an update of its sales database to the central SQL/DS database each night. Departmental applications can now query the data on the host, or use DataPropagator to pull a copy of the host database down to a local server.

7.5.5 Current Situation: More Solutions Required

I/S has solved the original problems and users can now share data throughout the enterprise and use the data with their workstation tools in a timely way. The results have reduced end-user frustration and have resulted in faster, more informed decision-making. The results show on the bottom line because the stores have the products they need when they need them. Customer Surveys indicate that the stores nearly always have what they want or can locate it quickly.

The drive to improve business processes continues. Now that more information is available, end users have new ideas about applications they′d like to help them do their jobs better. They were very impressed with how quickly their I/S department was able to meet their original distributed data needs. Now they want the new applications as quickly, which puts pressure on the resource-constrained I/S department to meet all the new requests. Connecting the enterprise and making the data available has also increased network traffic with some impact on overall system performance. Also surfacing are a number

of different departmental databases (and in some cases different database vendors) that are proliferating throughout the enterprise.

The Product Buyers and the Merchandising Managers wanted the sales data consolidated onto their local server in order to optimize their access to it. As the team looked at who else used the Sales information, they found that the Finance Department was running queries against the Sales database and moving the results into a database on an RS/6000 where they used a set of financial applications they had purchased.

The Customer Service department also came to the I/S department with a new application request. Customers frequently called them with questions about how a product worked, or to complain that the store personnel didn′t seem to know much about the products they sold. They needed a way to access information quickly to answer these questions. They also noticed that store personnel would call with questions as well. Most of this information was available in binders, but it wasn′t always up to date and sometimes the updates would get lost. They also mentioned that they had an existing third-party workstation database that provided them with Material Safety Data Sheets (MSDS) that contained information about such matters as safe handling of products such as paint, the hazardous nature of certain chemicals, and safe-disposal recommendations for such products. The new application would need to let them see both Product Information Forms and the Material Safety Data Sheets safety sheets at the same time.

From these discussions, the I/S department determined that it needed to address the following issues:

- The demand for new applications
- Where to store the data
- Better tools so that they can deliver applications faster with their limited resources
- Where it is most appropriate to keep data, in order to optimize data access
- How to connect multiple databases types (including multiple database vendors)
- How to reduce network traffic so that performance levels are restored

7.5.6 Solution: Distribute the Data

They decide that the best approach is to distribute the data for the new applications to minimize any additional network demands, and to use new workstation development tools that help them build applications more rapidly. Focusing on these areas, they can respond quickly to the requirements they are receiving. They also decide to implement two pilot projects to test their approach:

- Customer Service
  The new application will provide a visible improvement to customers and is independent of other business processes.
- Finance
  The new application is independent of other business processes and can provide a significant productivity increase.

Starting with these applications will let I/S develop their skills with distributed solutions and gain familiarity with the new tools that help with Rapid Application Development. They use the following products to address each of these issues:

- DB2 for OS/2 as a local server

I/S builds departmental applications with data for local use without generating additional network demand. Customer Service is the primary user of the Product Information Database, and requires rapid response time to meet Customer Service business process objectives. Implementing a local database provides Customer Service with the rapid response it needs, and allows the Product Buyers to make their occasional queries transparently through the DDCS connectivity. Using a copy of the SQL/DS database on the local server increases the availability of the data while reducing the network traffic. The copy results in faster access for the workstation applications, and improves overall system performance by reducing the enterprise server workload and reducing demands on the network.

- DataPropagator

They use DataPropagator to put an updated copy of the host database onto the local server. They do this each night when network demand is low. To minimize the demands on the network, they move the sales figures each night from the mainframe to the RS/6000 so Finance can continue to use its RS/6000 applications.

- VisualGen

After examining a number of Rapid Application Development tools for the workstation, they decide that VisualGen best meets the needs of their VSE system. Combined with VSE′s Distributed Workstation Feature, VisualGen provides them with a point-and-click Graphical User Interface (GUI) workstation platform as their application development environment for building distributed database applications. They work with all the departments that may be affected by the changes, and ensure they clearly understand who needs what data where. This allows them to determine where to store the data and how to distribute it to ensure optimal access with minimum impact to network traffic and performance.

- DataJoiner

This product allows queries to execute across all the different database types that they now know are installed throughout the enterprise. Customer Service could use the Product Information Database and the Material Safety Data Sheets Database at the same time—one query provides information from both the DB2 for OS/2 server and the third-party server. DataJoiner provides the solution to integrate the vastly different databases that had proliferated throughout the enterprise.

- Visualizer

Visualizer Query provides Customer Service with an easy-to-use GUI tool to quickly query their local Product Information database and the MSDS database in an ad hoc fashion as customer questions or complaints arise. Product Buyers can use the same Visualizer Query tool to do their occasional distributed queries against the host SQL/DS data through DDCS. Visualizer Query provides them with:

- Graphical query
- Statistics (over 50 statistical functions)
- Charting

- Reporting
- A procedural language to generate data in the above formats automatically.

7.5.7 Duplicate Data Sources and Processes

Using new Rapid Application Development techniques, I/S has completed several distributed data projects with great success. The discussion of the new applications and the close examination of the business processes has made it clear that there is a lot of duplicate data throughout the enterprise. Consolidating some existing applications could continue to optimize data access and ensure accurate, timely information.

Now the pilot projects are a proven success and more work is moving from the host to the workstation. VisualGen is key to developing new applications; queries are made from the workstations against the database, and the results are returned for use in the workstation productivity applications. And the new corporate applications are being built in the distributed environment. The mainframe host continues to control the data, systems, and query management for the enterprise, and ensures that all the distributed databases remain synchronized. Application support is provided for both host and workstation; some corporate applications remain on the host and continue to be maintained.

New products, promotions, and price changes are all decided centrally, and details are sent to each store by memo. The Store Managers would like information about the various new products and would also like to have details about promotions and price changes as quickly as possible. Another problem with price promotions is that each of them has to update their individual store systems with the correct prices, and sometimes mistakes are made when retyping the information. They have to update their applications for each changed product, which doesn′t take that long to do but is one more activity that has to be scheduled. They would like to be able to pull this information from the enterprise mainframe application and propagate the changes automatically across their systems.

Product information changes on a regular basis, and they would like that information available to them quickly, without the risk of being outdated. Again, they know this data is available on the enterprise mainframe and think the new systems could make it available to them as well. They know that their own Customer Service areas used to experience problems with rapid accurate responses to customers because the information is not readily available. The Store Managers have heard about the new Corporate Customer Service system and would like access to the same information so they can improve their Customer Service in the same way.

The I/S department now has a new set of challenges to solve:

- Consolidating duplicate data sources throughout the enterprise
- Integrating legacy applications into the newer systems
- Continuing to minimize the demands on the network
- Providing innovative application solutions

7.5.8 Optimize the Data Access

The solution here is to review the existing applications and the data with the various departments and business process experts, and together determine where application logic should reside. They all agree that the data management applications should remain on the enterprise server and that the legacy data should also remain on the mainframe. As they look at each of the areas, they determine that some of the existing applications are used by multiple departments but not throughout the enterprise, and they decide that they should add some enterprise servers to handle the application logic.

The Business Process experts also point out that they could make the applications on these enterprise servers execute some of the business rules, and therefore automate the business process more. For example, when product information and changes come in, they could encode some business rules to notify certain agents to initiate various processes. So, when a product change is sent to a buyer and they update the database, the server could also send a notification to all the Customer Service departments throughout the enterprise that there has been a product change.

They also look at the Sales applications, and determine that the sales data is mission-critical and needs to be available throughout the enterprise. They decide that they will keep the existing applications and data on the mainframe and distribute copies of the data to people who need to use it with local applications. Everyone agrees that a daily update is adequate for their business needs.

They install the following products to help address the new business needs:

- DataPropagator

Automatically distributes copies of the database to the servers that need it

- DB2/x - Enterprise Server

Consolidates common application logic and executes the business rules

- Flowmark

Automates some of the business processes and workflow

7.5.9 Changing Business Needs

Both the development of new applications and much of the maintenance of legacy applications can be done on workstations. The enterprise server now has responsibility for data management and systems management. All other functions are primarily executed in the distributed environment, making everyone more productive by giving them the information they need when and where they need it.

After a thorough review of the company's business processes and of who uses what information, it has been decided which applications will move to a local server and which will remain on the enterprise server. The criteria used for making these decisions included who needed to be able to access the information, the need for the attributes of the mainframe operating system environment, and whether changing these applications would provide more benefit than writing new ones.

Throughout the process, members of the I/S department exploited the skills gained in building mainframe applications and in understanding the linkages

between systems. As they worked with the various applications, they expanded their skills and acquired new tools. As they learned more about distributing data and as they identified the challenges of the complex environment, they brought great value to the discussions about providing the optimal systems for their enterprise. Their efforts showed throughout the organization with:

- Better customer service
- Faster, more informed decision-making
- An integrated solution that maximized efficiency while minimizing costs and leveraging the existing investments.

By adding to and extending its existing systems, this SQL/DS customer has transformed its business and systems into a flexible, scalable solution that can address its business needs for the future.

7.6 SQL/DS and DRDA

Creating systems for the future is easy for customers who have SQL/DS. For some, it means continuing with a strategy of distributing data by extending their current environments; for others, it starts with upgrading existing systems to levels that are enabled for distributed data. Today, SQL/DS V3R3 and above provide the distributed database function for the VSE and VM operating systems and let you take advantage of new operating system level features such as dataspaces; other DB2 family members can also provide distributed database capability across IBM and non-IBM platforms. This scalable DB2 family helps you:

- Provide your users with transparent access to data required to conduct their business
- Manage the data asset wherever it is
- Grow and change your system transparently as your business needs change
- Provide products and solutions across platforms optimizing the resources and capabilities of each platform and tool
- Enable transparent movement of applications and/or data between systems
- Access and work with data using powerful new decision-support tools

These new products allow you to put your applications and data where it makes the most sense for your enterprise. You can change the way your enterprise does business so you are more responsive to your customers (internal and external) and more cost-effective and competitive. Just as your business is changing in this new client/server world, so is IBM′s. Delivering solutions in this new client/server environment is exciting because we can now draw on more family products and are no longer limited to mainframe solutions, so now we can make tools like VisualGen available to our users. VisualGen lets you exploit the strengths of the workstation and rapid application development to build new applications or maintain and modify your mainframe applications. Our solutions mean you move forward with your future strategy while leveraging your current investment and minimizing the disruption in your enterprise.

The following publications are considered particularly suitable for a more detailed discussion of the topics covered in this chapter.

- *Distributed Relational Database Connectivity Guide,* SC26-4783

- *Introduction to Distributed Relational Data,* GG24-3200
- *Setup and Usage of SQL/DS in a DRDA Environment,* GG24-3733

Appendix A. SQL/DS Version 3 Enhancements Summary

This appendix briefly describes the main enhancements introduced in the SQL/DS product since SQL/DS V3R1.

A.1 SQL/DS Version 3 Release 1

A.1.1 SAA Enhancements

The following enhancements are introduced in SQL/DS V3R1 to provide increased compatibility with the IBM System Application Architecture Common Programming Interface (SAA CPI) Database Reference.

A.1.1.1 VARCHAR and VARGRAPHIC Compare

In SQL/DS Version 3, **prior to making the comparison**, between VARCHAR or VARGRAPHIC strings of different lengths, all trailing blanks are removed from both strings. In this example, COL1, COL2 and COL3 will compare equal.

```
COL1 = ¢ABC ¢  <===  VARCHAR, 4 Bytes, 1 Trailing Blank
COL2 = ¢ABC  ¢ <===  VARCHAR, 5 Bytes, 2 Trailing Blanks
COL3 = ¢ABC¢   <===  VARCHAR, 3 Bytes, 0 Trailing Blanks
```

The main advantage of this enhancement is to provide compatibility and portability with the other relational database management products.

A.1.1.2 UNION Compatibility

In SQL/DS Version 3, the SAA rules for compatibility of columns participating in the UNION and UNION ALL operations are followed. It is no longer required that the corresponding columns of the result tables of a UNION and UNION ALL have **identical data types and lengths** - it only needs to be **compatible data types**.

The main advantages of this enhancement are:

- Simplified formulation of certain types of queries involving the UNION operator.
- Improved execution in some cases because intermediate steps can now be avoided to obtain the same results.

A.1.1.3 SET Parameter Marker

This enhancement removes a previous restriction with regard to the use of parameter markers. It is now acceptable to specify a parameter marker (?) within an SQL expression of the SET clause in an UPDATE statement. For example:

```
UPDATE TABLEA SET COLB = ?
UPDATE TABLEA SET COLB = LENGTH (COLC * ? + 1)
```

The benefits of this enhancement are:

- Greater compliance with the SAA CPI definition which results in better portability.
- Increased performance for SQL dynamic statements in that the 'PREP', 'EXEC' sequence need not be repeated for each specific value of the

parameter marker, but may be replaced by a single 'PREP' followed by multiple 'EXEC's.

- Simplified programming due to only requiring one 'PREP' followed by multiple 'EXEC's.

A.1.1.4 Enhancement for COBOL and FORTRAN

The main benefit of this enhancement is to enable COBOL and FORTRAN programs to dynamically describe data attributes for column names in SQL statements and use virtual storage for host variables by using the DESCRIBE statement.

For an example of a VS COBOL II and additional information on the use of SQLDA, refer to the *SQL/DS V3R1 Usage Guide - GG24-3587*.

A.1.2 Functional Enhancements and New Capabilities

A.1.2.1 SQL/DS Procedure Language Interface (RXSQL) Feature

SQL/DS Procedure Language Interface is the product name for RXSQL which is a new optional feature of SQL/DS Version 3. RXSQL allows VM REXX to access SQL/DS in the VM environment.

A.1.2.2 Character Subtype

With this support, columns can now be identified as subtypes of the CHARACTER, VARCHAR or LONG VARCHAR data types. The subtypes specification instructs SQL/DS on how to interpret the data. The following subtypes may be specified:

- SBCS DATA

  Each character is represented by one byte.

- MIXED DATA

  Characters are represented by either single-byte or double-byte characters. Any MIXED data value can contain all single-byte character set (SBCS) characters, all double-byte character set (DBCS) characters, or a combination of both.

- BIT DATA

  The data is represented in binary format and is not translated into characters.

The objective of this feature is to provide a mechanism for flagging bit data, on which programs will not have to perform code point transformations, described in *SQL/DS SQL Reference for IBM VM Systems and VSE*.

A.1.2.3 Field Procedures

A field procedure is a user-written exit routine to transform values in a single short-string column (CHAR, VARCHAR, GRAPHIC or VARGRAPHIC).

A field procedure may be used to:

- Provide an alternate collating sequence

  For example, a telephone directory may require that names such as "McCabe" and "MacCabe" appear next to each other, an effect that the

standard EBCDIC sorting sequence does not provide. May also be used to facilitate National Language Support.

- Data compaction

  This will have the effect of saving physical DASD storage space. However, the savings in DASD space must be weighed up against the processing overhead of the field procedure.

A.1.2.4 SQLSTATE Support

SQLSTATE is a code set by the database manager after an SQL statement is executed to indicate the success or failure of the SQL statement. The value returned in SQLSTATE is common to the SAA-participating database products (DB2/MVS, SQL/400, DB2/2, DB2/6000).

The main purpose of this new set of return codes is to provide application programs with common codes for common error conditions thus enhancing program portability. This enhancement is also required for DRDA support which became available in later releases of SQL/DS (V3R3 for VM and V3R4 for VSE).

A.1.3 Performance Enhancements

A.1.3.1 MIN/MAX in a SELECT

Version 3 of SQL/DS accesses only the first or the last key in the index depending on whether the index is in ascending or descending sequence to satisfy the following queries:

```
SELECT MAX(COL1) FROM TABLE1
SELECT MIN(COL1) FROM TABLE1
```

where 'COL1' is the first column of an index on 'TABLE1' and there are no predicates.

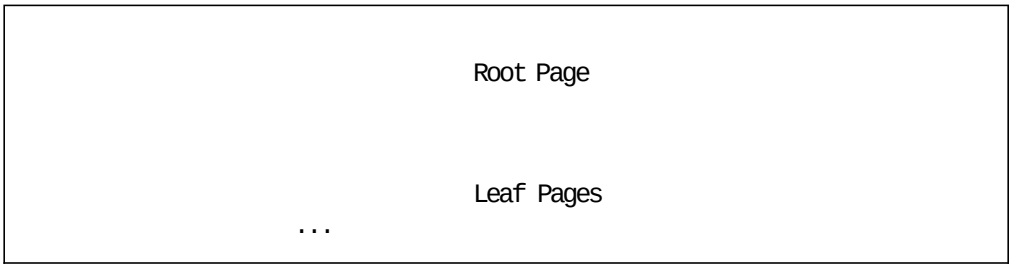


Figure 82. Index Processing under SQL/DS V3R1

A.1.3.2 Better Choice of TID for Insert

This enhancement improves the search algorithm used to find a page within a dbspace where a row will be inserted. The benefit is a reduction in the number of I/Os required to find the insert location and unnecessary directory writes have been eliminated.

This can be particularly effective when inserting into:

1. A table (including DATALOAD) which does not have an index defined.
2. A large table with a key value that is not clustered and has limited free space.

A.1.3.3 Efficient Statistics Collection

Statistics in SQL/DS are data describing the database objects such as dbspaces, tables, indexes and columns of the tables.

The statistics are stored in five different system catalog tables: SYSCATALOG, SYSCOLSTATS, SYSCOLUMNS, SYSDBSPACES and SYSINDEXES.

The enhancement ″Efficient Statistics Collection″:

- changes the way in which the statistics are collected

  From V3R1 onwards, SQL/DS gathers statistics in parallel with the process of populating the objects whenever this is feasible.

- expands the set of statistics collected when an index is created

  No new statistics are added, but SYSCATALOG.ROWCOUNT, SYSCATALOG.NPAGES and SYSDBSPACES.NACTIVE are updated also when an index is created or reorganized.

Statistics are collected as a result of the following commands:

- UPDATE STATISTICS FOR .... (table or DBSPACE)
- UPDATE ALL STATISTICS FOR .... (table or DBSPACE)
- DATALOAD/RELOAD (DBSU commands)
- CREATE/REORGANIZE INDEX

A.1.3.4 Exploitation of Multi-Block *BLOCKIO

SQL/DS has been enhanced to exploit the multi-block *BLOCKIO feature of VM/SP Release 6 and VM/ESA. Write requests to a single dbextent are blocked together so that up to 256 non-contiguous DASD blocks may be written in one IUCV request. This improvement saves physical I/Os when writing page buffers. It does not apply to database directory I/O. It is also beneficial during heavy update workloads, DATALOAD, RELOAD, CREATE INDEX and REORGANIZE INDEX. The availability of multi-block *BLOCKIO is detected automatically during SQL/DS initialization.

A.1.3.5 Allow Synchronous APPC/VM Option

This enhancement provided users using VM/SP Release 6 or VM/ESA with the ability to choose synchronous processing between the user and database machines, by specifying “SQLINIT ... SYNC(Y).” The reduced CPU overhead results in higher throughput and lower response times, especially for workloads that have high communication costs due to numerous DB calls (RDSCALL) per transaction and relatively light processing for each call.

A.1.3.6 Use of TID Instead of Hash for Gatename

In SQL/DS Version 3, key hashing no longer occurs for a unique index involving key level locking. Instead, the row′s identifier (TID) is used as the gatename as there is a one-to-one relationship between the unique index key and the TID. This eliminates the processing overhead of hashing the key into a gatename when a unique index exists and key level locking is required. This also leads to better concurrency as only one key will be locked instead of many keys that could hash to the same gatename.

A.1.4 Usability Enhancements

A.1.4.1 Dbextent Management

There are four parts to this enhancement:

- COPY DBEXTENT

  The COPY DBEXTENT function is used to aid disk migration. By moving the directory, logs, and dbextents to different devices, this function can also be used to control device and channel utilization (improving load balancing).

  Users who are migrating their databases to new DASD devices incompatible with their existing ones may use this function instead of archiving and restoring their databases.

- Directory Expansion

  Once a dbspace is added to the database, the user cannot change its size or remove it. Therefore, there is a tendency to overcommit the size of the dbspaces. The disadvantage to this approach is that the user may run out of page table space in the directory very quickly. These users may use the expand directory function to increase the page table area in the directory to allow them to add more dbspaces. Sometimes the growth of the tables in the dbspaces may exceed expectations and the size of the dbspaces are undercommitted. The user will have to drop the dbspaces, acquire larger ones and reload their data. If the directory is filled with dbspaces that are too small, the user will have to use the expand directory function to increase the page table area for the larger dbspaces. The database must be shut down before invocation of the expand directory function.

- DELETE DBEXTENT

  The DELETE DBEXTENT function allows users to remove physical dbextents from the database. The enhanced ADD DBEXTENT function, combined with the new DELETE function, can be used to re-assign DASD to another storage pool, and split up and merge dbextents.

- SHOW POOL Command

  The SHOW POOL command displays information about the dbextents defined to the database.

  The SHOW POOL command supersedes the SHOW DBEXTENT command. The SHOW DBEXTENT command remains unchanged. The SHOW POOL command displays not only the information about dbextents defined, but also the limit to the total size of the dbextents allowed in the directory. The user may use this command to monitor the usage of dbextents.

A.1.5 RAS Improvements

A.1.5.1 Enhanced Tracing Support

SQL/DS Tracing Support has been enhanced to allow the tracing of:

- The Data System Control (DSC) component in the database machine/partition for VM and VSE.

  To trace the DSC component, the new parameter TRACDSC must be specified in the SQLSTART EXEC for VM, in the JCL invoking the ARISQLDS program in the VSE environment or in the parameter data set for both VM

and VSE. The user can also turn the trace on or off by using the TRACE operator command in multiple user mode.

- The VM Resource Adapter (VRA) on the user machine.

  To start the trace of the VRA, the new parameter TRACERA must be specified in the SQLINIT EXEC on the user machine in order to trace the user application. To stop tracing, execute the SQLINIT EXEC without the TRACERA parameter.

Trace output can be directed to either disk or tape file as specified by the user and the ARIMTRA utility program is used to format and print the trace files.

A.1.5.2 Trace Formatter-RDS Enhancement

Previously, information about Data Manipulation Language (DML) statements used in the diagnosis of problems was provided in hexadecimal format. With this enhancement, the information is provided in a more usable and understandable format.

The ARIMTRA program is invoked directly in the VSE environment and is invoked by the SQLTRFMT EXEC in the VM environment.

A.1.5.3 Optimizer Cost Refinement

The purpose of this enhancement is to refine the cost estimation capabilities of the optimizer. This change will not alter any external function in SQL/DS, but will improve its performance characteristics.

The following changes have been made:

- The sort cost has been refined to calculate the sort width based on all the columns referenced, not just those in the select list.
- The filter factor calculations have been refined to more accurately predict the number of rows selected by a predicate. Furthermore, the default filter factors used have been made more selective and brought in line with other IBM RDBMS products.
- In queries where a start query can be identified and a cartesian join is likely to offer a good path, a cartesian join will be considered. This will permit the correct join order selection for example for queries where a large data table is joined with a small code table.

A.1.5.4 Complex SQL Statements

More complex SQL/DS statements can be processed successfully due to an increase in the size of the control block which holds the SQL statement from 16K bytes to 32K bytes.

As a result of this change, application and system programmers will find that larger, more complex queries which previously resulted in an SQLCODE = -101 (ARIBUTL - SQL statement is too complex. Exceeded SQL/DS internal limitations.) will now run successfully.

A.1.5.5 Archive Enhancements

The following enhancements have been made to improve availability and serviceability:

- SHOW LOGHIST Command

  This operator command has been added to display the contents of the log history area (HSDS). This information will assist operations greatly in situations where recovery is required.

- Save Log History area

  For VM, the log history area will be copied to the SQL/DS database machine′s A-disk immediately after a successful archive (log, SQL/DS or user). This A-disk copy will be used during a subsequent restore, in the event the log file is found to be unusable. **This support is not provided for VSE**.

A.1.5.6 CMS Shared File System (SFS) Support

SQL/DS V3R1 may be installed using the CMS Shared File System (SFS) which is available for VM/SP Release 6 or VM/ESA.

The CMS Shared File System allows users to organize their files into groups known as directories and selectively share them with other users.

With this support it is possible to install SQL/DS service and production files into these shared file directories with default names
VMSYS:SQLMACH.SQL310.SERVICE
and
VMSYS:SQLMACH.SQL310.PRODUCTION
respectively. The work disk (A-disk) for the database machine and the user facility server machine can also be installed in directories.

The database extents, log disks and directory disk **cannot** be part of the CMS shared file system.

A.2 SQL/DS Version 3 Release 2

The main content of SQL/DS V3R2 is directed at compliance with SAA standards and Federal International Processing (FIPS) standards. This compliance affords advantages for user productivity and for protection of the investment in data processing. In addition, new solutions are possible with the enhancements available with COBOL II support in VSE.

Besides standards compliance, performance enhancements have been included in this release, such as the Enhanced Index Only Access.

To achieve the benefits of SAA support for the COBOL language, VSE/ESA users will have COBOL II preprocessor support integrated as part of SQL/DS V3R2. Users can take advantage of COBOL II enhancements, such as added support for processing WHENEVER statements and support for literals of 160 characters.

A.2.1 Expanded SQL/DS Compatibility

A.2.1.1 VM/ESA System Support

SQL/DS Version 3 runs in both Enterprise Systems Architecture (ESA) environments, VM/ESA and VSE/ESA. SQL/DS performance is better in the VM/ESA environment compared with the VM/XA SP Release 2 environment. For example, it exploits the multi-block *BLOCKIO capability of VM/ESA, for improved dataload performance.

A.2.1.2 VSE/ESA System Support

SQL/DS exploits the capabilities of VM/ESA Release 1.1. For example, there are significant performance enhancements for complex queries, through exploitation of large memory and VM/ESA handshaking. SQL/DS Version 3 Release 2 uses VSE/ESA capabilities to provide performance improvements and additional functions.

A.2.2 Standards Compliance

The SQL/DS V3R2 product adheres to the following standards:

- International Organizational for Standardization (ISO) 9075-1989
- ANSI X.3.135-1989
- ANSI X.3.168-1989
- USA Federal Information Processing Standards (FIPS) 127-1
- IBM SAA Standards

The first four documents will be referred to collectively as SQL-89 hereafter.

The main content of SQL/DS V3R2 is directed at compliance with standards. For those customers that require close compliance with the Federal Information Processing Standard (FIPS) 127-1, such as governments or their agencies, SQL/DS V3R2 is of great value.

Advantages for users working with a product compliant with these standards are:

- A user′s education can be applicable for a longer time and for a wider range of products.
- Standards can increase an application′s life expectancy, freeing program developers for work on the application backlog.
- As a Systems Application Architecture (SAA) product, SQL/DS Version 3 Release 2 supports most of the language enhancements.

A.2.3 SQL Language Standards Compliance

SQL/DS Version 3 Release 2 includes modifications to comply with the SQL Language Standards. Following is a list of these modifications.

A.2.3.1 Support for WITH CHECK OPTION Clause

In SQL/DS V3R2, the new WITH CHECK OPTION clause of the CREATE VIEW statement allows users to have all inserts and updates on a view checked against the view definition. If the associated value or values do not fall within the definition established when the view was created, the insert or update will not be allowed.

The WITH CHECK OPTION can be used to allow SQL/DS applications to implement the concept of data domain.

A.2.3.2 Support for COBOL DISPLAY SIGN LEADING SEPARATE Attribute

In this section the term COBOL implies both COBOL and COBOL II.

Host variables in COBOL applications can now be declared with the DISPLAY SIGN LEADING SEPARATE attribute. COBOL variables used in SQL statements must be type-compatible with the columns of the tables with which they are to be used (stored, retrieved, or compared).

An SQL/DS numeric column is compatible with the following:

- COBOL variable of PICTURE S9(4) COMPUTATIONAL
- PICTURE S9(9) COMPUTATIONAL
- PICTURE S9(p)(V9(s))
- COMPUTATIONAL-3
- COMPUTATIONAL-2
- COMPUTATIONAL-1

A.2.3.3 Deferred Unique Integrity Checking

With deferred UNIQUE integrity checking, the intermediate result of the searched update is not checked for uniqueness.

SQL/DS no longer disallows searched updates, involving more than one row, on columns with a primary key or unique constraint.

Deferred unique integrity checking is not available for unique indexes in non-recoverable storage pools.

Consider this example showing the situation prior to this release.

C1	C2
1	A
2	B
3	C
4	D
5	E
6	F

Table 29. Deferred Unique Integrity Checking

```
Before V3R2, the command:  UPDATE T1 SET C1 = C1 + 1

if INDEX(C1) was ASCending, then the command would have FAILED
if INDEX(C1) was DESCending, then the command would have WORKED
```

A.2.3.4 A Single Value in an IN Predicate List

An IN predicate list can now contain a single value. To select a row from a DEPARTMENT table that concerns manager number 10, the user could type either of the following statements to produce the same result:

```
SELECT * FROM DEPARTMENT WHERE MANAGER = ¢UB010¢

SELECT * FROM DEPARTMENT WHERE MANAGER IN (¢UB010¢)
```

Both statements are treated the same because of query transformation. The IN predicate will be converted to the equal predicate (′=′). Please refer to the *SQL/DS V3R4 Performance Guide*.

A.2.3.5 SQL Style Comments

New language independent comments can be added within static SQL statements, SQL statements in DBS Utility, and DBS Utility commands. The comments begin with two consecutive hyphens (--) and end at the end of the line.

These rules apply to the use of SQL comments:

- The two hyphens must be on the same line, not separated by a space.
- Comments can be started whenever a space is valid (except within a delimiter token or before or between ′EXEC′ and ″SQL′).
- Comments are terminated by the end of the line.
- Comments are not allowed within statements that are dynamically prepared (using PREPARE or EXECUTE IMMEDIATE) or prepared using any of the extended dynamic PREPARE statements.
- In COBOL, the hyphens must be preceded by a space.

A.2.3.6 Extended Support for Negative Indicator Variables

When used in predicates, indicator variables give the user the ability to alter the predicate clause of a query without having to re-preprocess the application. This avoids duplicating SQL statements in progress. For example:

```
SELECT * FROM TABLE1
       WHERE COL1 = :COL1:COL1IND
       OR COL2 LIKE :COL2:COL2IND
```

Assigning -1 to COL2IND will effectively remove the second predicate from the WHERE clause. Please refer to the *SQL Reference* manual for references to host variables.

A.2.3.7 Enhancements to Decimal Data Type

The DECIMAL data type now allows **up to 31 decimal digits** of precision and enforces the precision specified. NUMERIC is recognized as a synonym for the enhanced DECIMAL data type.

In this example, the result of multiplying two fields defined as DECIMAL (10,0), will be DECIMAL (20,0) and not FLOAT since the precision of the result of multiplication is min(31,p+p′) and the scale is min(31,s+s′).

The advantages of this enhancements are:

- Better Precision

- Intermediate Results
- Larger Decimal Values

A.2.3.8 TIMESTAMP Arithmetic

The rules for date/time arithmetic have been extended to include subtraction of TIMESTAMPs to obtain TIMESTAMP durations. Several scalar functions have also been extended to accept TIMESTAMP durations as parameters.

The result of subtracting one timestamp (TS2) from another (TS1) is a timestamp duration that specifies the number of years, months, days, hours, minutes, seconds, and microseconds between the two timestamps. The data type of the result is DECIMAL(20,6).

The result of adding a duration to a timestamp, or of subtracting a duration from a timestamp, is itself a timestamp.

For example,

```
COL1 has a value of ¢1994-04-25-12.10.56.123456¢
and
COL2 has a value of ¢1993-03-14-08.23.45.654321¢

then COL1 - COL2 is ¢0001-01-11-03-47-10.469135¢
```

A.2.4 Performance Enhancements

A.2.4.1 Optional CMS Work Unit Support

The new WORKUNIT parameter of the SQLINIT EXEC allows the user to control whether or not CMS work unit support is to be used for an application.

CMS work units are supported in the VM/SP and VM/ESA operating systems and are a central concept to maintaining the integrity of data. A CMS work unit is a group of related operations that can be either committed or rolled back as a unit. When the operations associated with a work unit are committed or rolled back, new operations can also be committed or rolled back.

The advantage of this enhancement is less overhead when CMS work units are not required.

A.2.4.2 No Locking of Non-leaf Pages

Performance improvements are achieved with a reduction in locking. With this enhancement, only the root page and the involved leaf pages of an index are locked when necessary. Locks on non-leaf pages are eliminated.

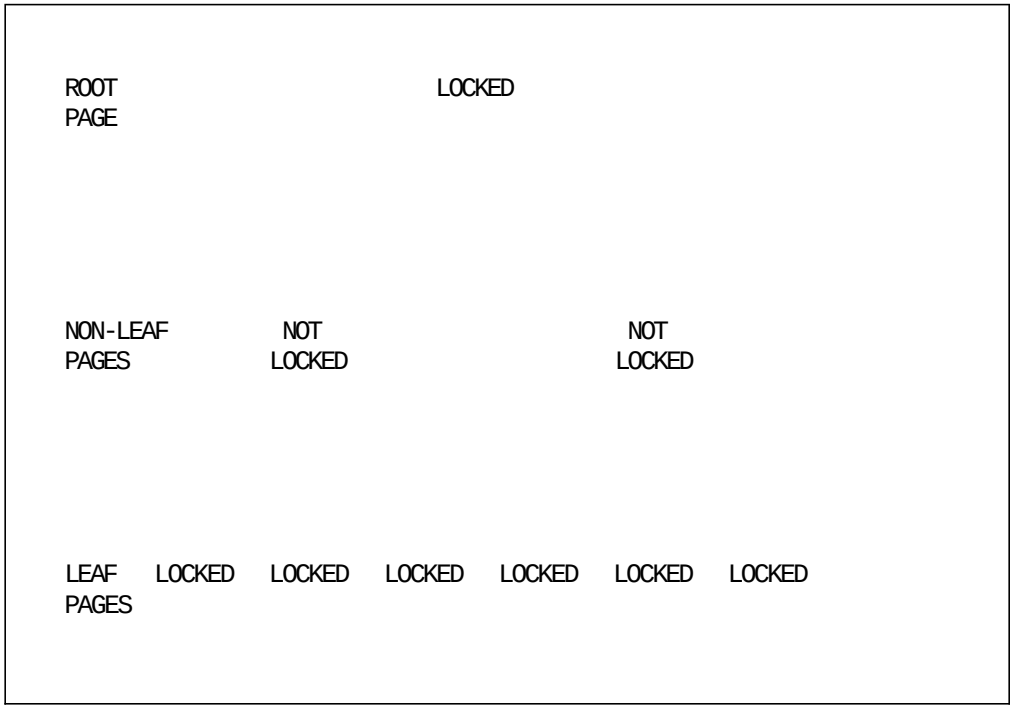


Figure 83. SQL/DS V3R2 No Locking on Non-leaf Pages

Advantages of this are:

- Reduced number of locks
- Reduced index locking conflict

A.2.4.3 Eliminate Lock on Successor Key

This feature is only applicable to non-unique indexes containing duplicates. When deleting a tuple identifier (TID) from the index, the successor key is only locked if the deletion of the TID causes a deletion of the index key. Performance improvements are obtained when the successor key is not locked.

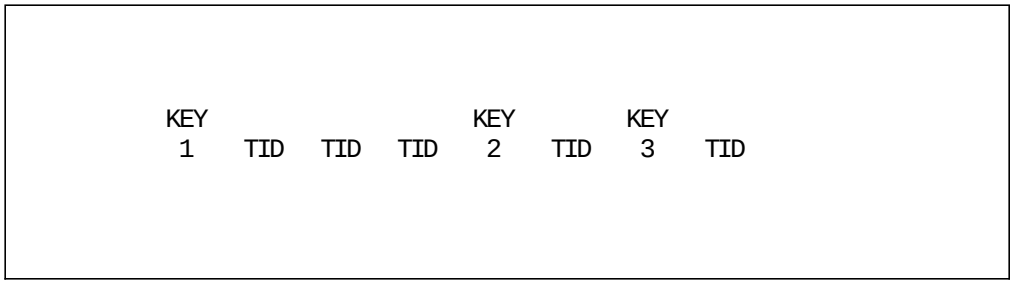


Figure 84. SQL/DS V3R2 Eliminate Lock on Successor Key

From the previous example, deletion of a TID from KEY = 1 does not cause the successor KEY= 2 to be locked.

On the other hand, deletion of the TID from KEY = 2 will cause the successor KEY = 3 to be locked.

A.2.4.4 Enhanced Index Only Access

Previously, index only access was only possible if there were no sargable predicates which were not key matching predicates. Now, index only access is possible if all columns referenced are present in the index. Performance is enhanced because data page access is not required.

Index only access is possible for both selective and non-selective index scans. For index only access, the model query is:

```
SELECT C1, C2 FROM T1 WHERE C1 = 50
```

and we assume that an index exists on columns C1 and C2.

In some cases, it can be reasonable to create an index that includes data just to improve the performance of certain common queries.

Another case where index only access is beneficial is a table with very long rows where the portion of the row retrieved is small compared to the size of the row. If a query needs only three or four relatively short columns of that data, an index on those columns might be worthwhile merely to avoid the cost of scanning all data pages, and extracting the useful data.

A.2.4.5 MIN/MAX in a Subquery

This enhancement will allow the Version 3 Release 1 MIN/MAX enhancement to work in subqueries. The Version 3 Release 1 MIN/MAX enhancement reduces the cost of selecting the minimum and maximum value of the first column of an index by fetching the value from the first or last key instead of scanning the entire index.

Previously, the MIN/MAX enhancement was restricted to queries with no predicates whatsoever. Version 3 Release 2 allows predicates anywhere except in the query block that wishes to use the MIN/MAX fetch. When selecting the maximum value of the first column of an index, the column still must be a non-nullable column in order to take advantage of the enhancement.

A.2.5 Usability Enhancements

SQL/DS V3R2 addresses numerous requests from customers to include Usability Enhancements.

A.2.5.1 Support for RPG Programming Language

The RPG programming language is now supported. Now RPG joins Assembler, C, COBOL, FORTRAN and PL/I as the Programming Languages supported by SQL/DS for VM.

A.2.5.2 New REBIND PACKAGE Command

The new DBS Utility command, REBIND PACKAGE, allows users to easily rebind existing packages. This is very useful after table reorganization. This new REBIND PACKAGE command offers the advantage of not needing intermediate files to recreate the package.

When the SQL/DS application server loads the package, it checks that the package is still valid. A package may not be valid if one of its dependencies has been dropped. For example, some index that the package uses may have been dropped.

A successful rebinding has no negative effect on the program except for a slight delay in processing the first SQL statement. To minimize this delay the DBSU REBIND PACKAGE command can be used to rebind the invalid package without unloading and reloading it, after it has been invalidated, but before it is executed. The REBIND PACKAGE can be used in single or multiple user mode.

```
REBIND PACKAGE (package_name)
```

package_name Identifies the package to rebind.

A.3 SQL/DS Version 3 Release 3

A.3.1 Functional Enhancements and New Capabilities

A.3.1.1 Virtual Storage Constraint Relief under VM

SQL/DS code can run in 31-bit addressing mode on a database machine, so if SQL/DS is installed in a VM system which runs XA or XC mode, it is possible to extend the virtual machine size from 16MB to 2GB. This allows the database manager to operate with more concurrent users, more page buffers, and more directory buffers when running AMODE(31). This can result in a better concurrency usage, greater throughput, reduced database I/O, reduced lock escalations and an overall improved performance.

In 370 mode, only 24-bit addressing is available. It has been kept for compatibility with existing applications.

To provide improved performance, sufficient real storage must be available to prevent unacceptable increases in VM paging.

A.3.1.2 VM Data Spaces Support (VMDSS) Feature

The VMDSS Feature allows the database machine to take advantage of the VM/ESA data space support of the ES/9000 processor. Performance can improve dramatically with sufficient real storage, as SQL/DS cooperates with VM and exploits new functions and interfaces. A more detailed explanation of this Feature is on Appendix B, “SQL/DS Version 3 Features Summary” on page 185.

A.3.1.3 DRDA Support for SQL/DS under VM

With the implementation of the Distributed Relational Database Architecture (DRDA), an application running on an SQL/DS database management system can now access data in a non-SQL/DS database management system that implements the DRDA architecture, such as the DB2 or OS/400 database management system.

Users can preprocess a program against a non-SQL/DS server, reload an SQL/DS portable package on a non-SQL/DS database management system, or bind non-SQL/DS statements to other application servers. Users on non-SQL/DS database management systems such as DB2 and OS/2 can also access data in an SQL/DS application server.

A DRDA environment provides full SQL client-server architecture through the use of application requesters and application servers. A DRDA environment provides

the architecture for Remote Unit of Work (RUOW) access to data that is distributed across different installations: the application requester and the application server do not have to be running with the SQL/DS database manager.

Using Remote Unit of Work to access other SAA systems, a user or application program accesses one database from within a logical unit of work. Any number of tables can be accessed from the same database. Many databases can be accessed from one application program. An application program can, for example, generate a report based on data from several databases.

A.3.1.4 Improved Package Management for VM

Prior to Release 3, only 10 active packages were allowed in a logical unit of work (LUW). This restriction has been lifted with improved package management. Using new startup parameters, the system administrator can now define the maximum number of active packages in an LUW and the percentage of package cache entries that will remain loaded in storage at the end of an LUW.

A.3.1.5 Simplified Chinese for VM

In Version 3 Release 3, Simplified Chinese has again been added to the list of languages supported.

A.3.2 Standards Compliance

A.3.2.1 Multivendor Integration Architecture Conformance for VM

In Version 3 Release 3, the SQL/DS database manager partially conforms to the Multivendor Integration Architecture. This support includes revised JIS SQL (ANSI/ISO SQL2) national character string support. The character N can now be used as a synonym for G to indicate a graphic string constant. It also includes variable declaration support for COBOL 85 DBCS.

Now for COBOL, PICTURE N(n) is a synonym for PICTURE G(n).

A.3.3 Usability Enhancements

A.3.3.1 CONCAT Keyword Support for VM

To allow greater portability of packages among application servers with different system default CCSIDs, it is recommended to use the CONCAT keyword instead of the operator (||). The operator, (||), remains as a synonym for CONCAT. CONCAT is now a reserved word.

For example, the query:

```
SELECT FIRSTNME || ' ' || LASTNAME FROM EMPLOYEE
```

may be written as:

```
SELECT FIRSTNME CONCAT ' ' CONCAT LASTNAME FROM EMPLOYEE
```

A.3.4 RAS Improvements

A.3.4.1 Trace in Memory for VM

Performance of the trace facility has been improved with the trace in memory option. This option allows trace records to be stored in a wrap-around trace buffer in memory. As no I/O is performed while trace is generating records, processing is not affected by I/O delays.

A.3.4.2 DUMPTYPE Parameter for VM

The DUMPTYPE=F startup parameter now issues a CP VMDUMP command instead of a CP DUMP command. Running with DUMPTYPE=F it is possible to get a full virtual machine dump of the database machine on some error conditions and trace points. The dump generated includes the whole virtual machine, as well as any saved segments the virtual machine uses. This type of dump may be sent by tape or over the RSCS network to an IBM representative instead of having to print the dumps and send the hardcopy format.

A.3.4.3 Improved Communications Error Reporting for VM

Now, error information is automatically included in the SQLCA when a severe communication error is encountered at the application requester, making it easier to find the source of severe communication errors. Previously, communication errors received through IUCV and APPC were mapped to a common set of error indicators. Now, error information is automatically included in the SQLCA when a severe communication error is encountered at the application requester. The information reported in the SQLCA includes the communication protocol in effect, the function executed, and all possible error fields.

A.3.4.4 First Failure Data Capture for VM

This function is automatically invoked when an internal problem is detected. This support reduces the need to recreate an error since vital control block information is captured at the point the error is detected. This support is only available when using the DRDA protocol.

A.3.4.5 Collecting Service Information from the SQLCA for VM

The SQLSTATE, SQLERRMC, SQLWARN and SQLERRP fields in the SQLCA provide additional information about the application servers to which a connection is established or warnings of possible error conditions. This is quite useful when diagnosing problems in a distributed environment by providing additional information about the application server or warnings of possible error conditions.

A.4 SQL/DS Version 3 Release 4

A.4.1 Functional Enhancements and New Capabilities

A.4.1.1 DRDA Application Server Support for SQL/DS under VSE/ESA

The VSE SQL/DS database manager can now participate in the Distributed Relational Database Architecture (DRDA) remote unit of work as an application server for an application requester running under the following IBM database managers:

- DATABASE 2 (DB2)
- DATABASE 2 OS/2 (DB2/2)
- DATABASE 2 AIX/6000 (DB2/6000)
- OS/400
- SQL/DS on VM

or any other relational database manager that implements the DRDA remote unit of work feature.

With this facility, you can position your VSE SQL/DS database manager as a data server for intelligent workstations.

An enhanced SHOW SYSTEM operator command provides additional information about the application server. A new SHOW CONNECT operator command shows the status of all connected remote users.

The DRDA remote unit of work facility can be optionally installed. If installed, it may require 31-bit addressing and therefore, VSE/ESA Version 1.3 with ESA or VMESA supervisor mode.

A.4.1.2 CCSID Support for VSE

This support provides automatic Coded Character Set Identifier (CCSID) conversion for data and statements whose source and target have different code pages and/or character sets and/or encoding schemes. CCSID support adheres to the IBM Character Data Representation Architecture (CDRA). This architecture defines services, supporting resources, and conventions to achieve a consistent representation of double-byte and single-byte character set data.

The following changes have been made as a result of CCSID support:

- New preprocessing options have been added
- A CCSID OPTION has been added to the CREATE TABLE and ALTER TABLE statements.
- The DESCRIBE statement has been enhanced to return information in the SQLDA regarding the CCSID of a column
- Additional rows have been added to the

  SYSTEMS.SYSOPTIONS,

  SYSTEMS.SYSCHARSETS,

  SYSTEMS.SYSCCSIDS and

SYSTEMS.SYSSTRINGS catalog tables.

A.4.1.3 Virtual Storage Constraint Relief for VSE

You can use 31-bit addressing to increase the size of addressable memory from 16 megabytes to 2 gigabytes. This makes it possible for your SQL/DS database manager under VSE to have more concurrent users, more page buffers, or more directory buffers. 31-bit addressing has already been available for VM with SQL/DS V3R3.

Because CICS Version 2 Release 2 also supports 31-bit addressing, you can increase the size of a CICS partition from 16 megabytes to 2 gigabytes. The SQL/DS database manager accommodates this increased CICS partition size by accepting requests from a transaction regardless of whether it is running above or below the 16 megabyte line.

You must run VSE/ESA Version 1.3 or later with either ESA or VMESA supervisor mode, to use 31-bit addressing.

To provide improved performance, sufficient real storage must be available to prevent unacceptable increases in paging of the database partition.

A.4.1.4 Enhanced SHOW STORAGE Command

A new SHOW STORAGE operator command is provided to let you determine the system load and avoid problems caused by insufficient storage. This command displays information about system storage currently being used and the maximum total storage usage.

The SHOW STORAGE command can be used together with the RESET HIGHSTOR command. By resetting the HIGHSTOR value, performing a function and then invoking the SHOW STORAGE command, you can determine the maximum storage needed to perform the function.

A.4.1.5 Improved EXPLAIN Capabilities

The EXPLAIN statement is used to analyze Data Manipulation Language (DML) statements: SELECT, UPDATE, INSERT and DELETE. The EXPLAIN command could be called the window to the optimizer. The optimizer analyzes each query and determines the best method for performing the request. The EXPLAIN tool provides the ability to examine the optimizer′s analysis and decisions and to provide information about the structure, execution, and approximate cost of the SQL statement being analyzed.

In V3R4, the EXPLAIN function has been enhanced as follows:

- Several new columns have been added to the existing EXPLAIN tables.
- The EXPLAIN function can now be used as a preprocessing option for all static SQL statements embedded in an application program.
- A DBS utility job file is shipped with the SQL/DS product to generate the new EXPLAIN tables, indexes, and views.

The enhanced EXPLAIN facility is more compatible with the DB2 EXPLAIN facility.

EXPLAIN has been used interactively using a program such as ISQL, DBSU or QMF. EXPLAIN statements can also be issued from within an application program using both static and dynamic SQL.

New columns and changes in the EXPLAIN tables are introduced in SQL/DS V3R4 to gain compatibility with MVS/DB2. In SQL/DS V3R4, the user can use implicit EXPLAIN using the EXPLAIN(YES) preprocessing parameter in preprocessing applications in COBOL, C, PL/I, FORTRAN and RPG. The user can also use the EXPLAIN(YES) option of the CREATE PACKAGE statement in Assembler and RXSQL applications. Note that the actual EXPLAIN processing occurs during the ″PREPARE″ phase of the SQL statement. A DBSU job file to create EXPLAIN tables, indexes and views is shipped with SQL/DS V3R4.

EXPLAIN is a powerful tool for understanding the access path which is selected by the SQL/DS optimizer to access data in order to satisfy an SQL DML request. It also provides additional information about the needs for resources which are required for the applications to run in an efficient way.

A.4.1.6 Reduced CICS Logging for VSE

A logical unit of work that is accessing SQL/DS data may also be accessing data from one or more other sources such as a DL/I database or a VSAM data set. CICS transactions normally use two-phase commit protocols to ensure the integrity of all such recoverable resources.

With two-phase commit protocols, when an application program requests syncpoint processing, CICS polls all resource managers accessed in the logical unit of work. If all resource managers agree to commit, CICS commits all data. If any resource manager does not agree, all data changed in that logical unit of work is backed out.

The two-phase-commit protocol is necessary but expensive in that it forces several CICS log I/Os for each commit. However it need not be used under the following circumstances:

1. When all recoverable resources in an LUW are accessed for read-only.
2. When just one resource, such as SQL/DS data, is being accessed.

   SQL/DS may be the only resource accessed in the LUW, or all other resources in the LUW besides SQL/DS are accessed for read-only.

In Version 3 Release 4, SQL/DS will automatically inform CICS that it can handle the single phase commit.

If CICS then determines that the above circumstances exist, it will use the single phase commit protocol, and will inform SQL/DS as appropriate. This will reduce I/Os to the CICS log and sometimes to the SQL/DS log as well.

This enhancement requires CICS/VSE Version 2.2 or later.

A.4.1.7 Multiple Application Server Support for VSE

In V3R4, up to 36 SQL/DS application servers can be active at the same time in your VSE system.

Previous releases of the SQL/DS database manager permitted an application program to access just one database under VSE. An application program running in a batch partition can now access multiple SQL/DS application servers.

Each application server is identified with a unique dbname. Batch application programs can use a dbname in a CONNECT statement to access an application server; however this facility is still not allowed for online programs. The

CURRENT SERVER special register returns the dbname of the current application server. Each application server must be accessed in a separate logical unit of work.

Access to multiple databases is not available for CICS transactions. However, transactions running in different CICS partitions can access different application servers.

A.4.1.8 Host Structure Support

Several of the host languages that are supported by the SQL/DS database manager let you define variables in structured formats. For example, a COBOL program might specify the following structure for a three line address:

```
04  ADDRESS
    05 LINE-1
    05 LINE-2
    05 LINE-3
```

A programmer can specify ADDRESS to refer to all three lines.

The SQL/DS database manager has been enhanced to provide this kind of support for two levels of structured variables. If you code a host structure in an SQL declaration, you can use the name of that host structure in any SQL statement where you would otherwise code a list of host variables.

This facility is available for SQL code embedded in application programs written in C, COBOL, PL/I, and RPG. If you code a structure with more than two levels, all of the lowest two level substructures can be used as host structures by the SQL/DS database manager.

A.4.1.9 Removal of 512 Host Variable Restriction

The previous maximum number of host variables allowed in a program module was 512. This restriction has been removed. The number of host variables is now restricted only by the size of storage.

A.4.1.10 Cascade Delete Enhancement for Referential Integrity

Referential integrity ensures that references in one table to data in another table are always valid. In previous releases, you could specify cascade delete constraints to ensure that if a value was deleted from a parent table, the corresponding row of a dependent table would also be deleted. However, this support was restricted to a single level cascade delete.

V3R4 expands the capabilities of the cascade delete constraint. When a row is deleted from a dependent table because of a cascade delete, the delete rule that exists between the dependent table and any tables that are its dependants will be processed. For example, suppose a cascade delete constraint exists for Table_A that results in a row being deleted from Table_B. Table_B, in turn, might have a delete rule that then causes a value in Table_C to be set to NULL.

This enhancement can ensure integrity across several tables and reduce the programming effort needed to develop an application.

A.4.1.11 Eliminate Data Locks When Index-Only Access

Now with V3R4 locking of data rows or pages has been eliminated in cases of index-only access, when there is a read-only operation, isolation level is Repeatable Read, and row or page level locking is in effect for the key. An exception to this rule is the case in which a scan is being performed with the intent to update or delete data. In this case locks on data rows or pages will still be required.

Locks on data rows or pages will still be acquired even if index-only access and isolation level is CURSOR STABILITY, because a lock on an index key is only an instant lock if the isolation level is CS, and a medium lock on the data is needed to protect the integrity of the data. This enhancement allows multiple users reading and one user updating to share a row.

In V3R4, SQL/DS will automatically inform CICS that it can handle the single phase commit protocol. If CICS then determines that the above circumstances are true, it will use the single phase commit.

A.4.1.12 Improved Package Management (VSE)

The previous restriction of 10 active packages in a logical unit of work has been removed. Using new startup parameters, the system administrator can now define the maximum number of active packages in a logical unit of work and the percentage of package cache entries that will remain loaded in storage at the end of a logical unit of work.

This enhancement allows the development of more complex structured programs composed of multiple modules.

A.4.1.13 C NUL-Terminating String for VSE

All C input strings with a length greater than one are now handled as varying-length strings and must be terminated with a NUL (X′00′) byte.

The SQL/DS system interprets a character string in C as NUL-terminated if the length of the string is greater than one and less than 32,768 bytes.

The NUL-byte is mandatory when the SQL/DS system receives data from a NUL-terminated stream. An SQLCODE -302 (SQLSTATE ′22510′) is received if the NUL-byte is not found within the defined length of the string. This means that the maximum number of bytes of data that can be stored in a NUL-terminated string is one less than the defined length of the string.

A.4.1.14 Support for the IBM DATABASE 2 AIX/6000 Database Manager for VM

The IBM DATABASE 2 AIX/6000 (DB2/6000) database manager implements the DRDA remote unit of work feature. Your VM SQL/DS database manager can function as an application server for your DB2/6000 database managers.

A.4.1.15 Simplified Chinese for VSE

Simplified Chinese has again been added to the list of languages supported; messages and HELP text can now be loaded and displayed in Simplified Chinese.

A.4.2 Standards Compliance

A.4.2.1 Additional DBCS Support for VSE

In V3R4, the SQL/DS database manager provides additional DBCS support. This support includes revised JIS SQL (ANSI/ISO SQL) national character string support. The character N can now be used as a synonym for G to indicate a graphic string constant. It also includes variable declaration support for COBOL85 DBCS.

A.4.3 Usability Enhancements

A.4.3.1 Dual Logging Enhancement for VM

If you are using dual logs under VM and one is damaged, it can be replaced with a new minidisk. SQL/DS will then copy the good log to the new one without the necessity of doing COLDLOG.

Now with SQL/DS V3R4, if a page is damaged on one log, startup will attempt to fix it using the good log. This was true before for log data pages but has been extended to the log CID and HSDS pages.

Also, if either the primary or secondary log is damaged, the log can be replaced and the next startup will copy the information from the good log to the new one. This is true for VM only because of the nature of data sets in VSE. When a log data set is redefined, it cannot be read until it is first written. The COLDLOG process first writes the log CID and HSDS pages before actually reading any pages. In order to determine that one of the dual logs is damaged and needs to be copied from the other log, it must first be read. So, in VM, now it is possible to copy the good log into the new one without a COLDLOG.

Also, as no COLDLOG is required, the log history continuity is preserved.

The documentation on replacing damaged disks/data sets has been completely re-written in the System Administration manual.

A.4.3.2 Using Alternate Tape Drives when Archiving for VSE

Previously, facilities for database and log archiving were allocated dynamically; that is, the operator would be prompted for a CUU volume to dynamically assign to a tape drive. With V3R4, you have the option of statically assigning multiple tape drives for a database or log archive.

This new option can save you time when an archive operation requires multiple tapes since you need not wait while a tape is being rewound and unloaded.

Customers that rely on SQL/DS backup and restore procedures are required to take database archives occasionally. The backup copies of the database are written to tape. Customers typically have more than one physical tape drive. SQL/DS prompts for the physical address (cuu) of the tape drive to be used to write the tape archive. The operator enters the physical (cuu) address.

Customers with a large database may not be able to write the entire database onto a single tape. When the first tape is filled, the tape drive will rewind and unload the filled tape before the archive process can continue writing the archive to the next tape. The rewind and unload process may result in a delay of 8-10

minutes. This delay may be unacceptable for customers that require two (or three or more) tapes.

A.4.3.3 CONCAT Keyword Support for VSE

To allow greater portability of packages among application servers with different system default CCSIDs, it is recommended to use the CONCAT keyword instead of the operator (||). The operator, (||), remains as a synonym for CONCAT. CONCAT is now a reserved word.

For example, the query:

```
SELECT FIRSTNME || ' ' || LASTNAME FROM EMPLOYEE
```

may be written as:

```
SELECT FIRSTNME CONCAT ' ' CONCAT LASTNAME FROM EMPLOYEE
```

A.4.3.4 Message Text Changes for VSE

Messages sent to the operator console can contain more than one line of text. Previously, each line of text was sent separately. In V3R4, all text from a message is sent as a single output message. There is no change for a human operator. An automated operator, however, will receive just one output message instead of one for each line of a multi-line message.

A.4.3.5 Package Dbspace Full Condition Handling

Previously, when a package was dynamically re-preprocessed and a package dbspace full condition occurred, the re-preprocess would fail with an SQLCODE of -946 and the user would then need to explicitly recreate the package.

If this condition occurs with V3R4, the database manager will automatically search for a non-full package dbspace that can accommodate the re-preprocessed package. Manual intervention will only be required if all package dbspaces are full (if they already contain 255 packages or do not have enough free space to hold the additional package).

A.4.3.6 The Connectable and Unconnectable States for VM

When a severe error occurs, the logical unit of work is rolled back and the communication link is severed. This causes the application to enter a ″connectable and unconnected″ state. Previously, if the next SQL statement was not a CONNECT statement, SQL/DS caused the application to abend.

However, if this condition occurs in V3R4, SQL/DS does not cause the application to abend, but issues a severe error code instead; SQLCODE of -900 and SQLSTATE of 51018.

A.4.4 RAS Improvements

A.4.4.1 Expanded Deadlock Detection Message Text

In earlier releases when a logical unit of work was rolled back because of a deadlock, the message corresponding to SQLCODE = -911 only identified the user who was rolled back but did not identified the other users involved in the deadlock or the resources for which the user was waiting.

The old message was:

SQLCODE = -911
The current logical unit of work was rolled back because of a deadlock.

Now the message is changed to:

SQLCODE = -911
The current logical unit of work was rolled back because of a deadlock. It was waiting for a *lock_type* **lock in DBSPACE =** *dbspace_num* **held by user** *(sqlid or jobname)*

The *jobname* is only available in VSE. It will be provided when a batch job is rolled back because of a deadlock. The *sqlid* will be provided in all other scenarios.

A.4.4.2 Processing a DROP TABLE Statement

When a DROP TABLE statement is executed, the rows of the specified table are not dropped immediately; instead, the table is marked for deletion by adding a row to the SYSDROP system catalog table. After the current logical unit of work (LUW) is committed, a new LUW is started and the table is dropped.

For each page that contains data for the table being dropped, an equal number of shadow pages must be retained until the delete operation is completed, and the LUW can be committed. If the table occupies a large number of pages, it is possible to run out of physical pages, or to encounter a storage pool full condition.

With this enhancement, checkpoints are possible as rows are deleted, freeing shadow pages more quickly. As well, the database manager will no longer abend if a storage pool full condition occurs during a DROP TABLE operation.

A.4.4.3 Enhancement to COLDLOG Processing

Previously, running a COLDLOG operation could destroy log data needed for recovery if any logical units of work were unresolved when the database was last shut down. In V3R4, the operator will get a warning message if this condition exists and will be able to cancel the COLDLOG operation before any data is lost.

If you have already redefined your log data set, however, this enhancement will not be as useful since the log data has already been erased.

A.4.4.4 First Failure Data Capture for VSE

First Failure Data Capture is automatically invoked when an internal problem is detected. This support reduces the need to recreate an error since vital control block information is captured at the point the error is detected.

A.4.4.5 Trace in Memory for VSE

Performance of the trace facility has been improved with the trace in memory option. This option allows you to store trace records in a wrap-around trace buffer in memory. The trace buffer size is defined by specifying the TRACEBUF parameter when the TRACE is started. Trace records stored in the trace buffer may be then written to the output file on a disk or tape when trace is turned off by an operator command or if the SQL/DS database system is shut down normally or abnormally. The output file must then be formatted in the same way as if it was created without the trace in memory option.

A.4.4.6 Improved Storage Trace for VM

The storage trace facilities have been enhanced to provide relevant trace point information about the SQL/DS application server. The new facility provides consolidated trace information for system and working storage.

It can be invoked either during database startup with a new STARTUP command parameter of after the database is started with the TRACE operator command.

A.4.5 Library Enhancements

A.4.5.1 SQL/DS Performance Tuning Handbook (SH09-8111)

This new manual provides information for a more effective tuning of the database.

A.4.5.2 SQL/DS Performance Guide Redbook (GG24-4047)

This manual provides extensive performance information and guidelines for a Database Administrator and also for the Data Analyst and the Application Programmer, in achieving cost-effective data systems in terms of response times, throughput and availability of SQL/DS.

Appendix B. SQL/DS Version 3 Features Summary

This appendix describes the main features introduced in the SQL/DS product since SQL/DS V3R1. It does not describe the new Data Restore feature which is covered in 4.3, “Data Restore Feature” on page 96.

B.1 SQL/DS Version 3 RXSQL Feature

This topic describes the SQL/DS Procedure Language Interface (RXSQL). The following subjects are covered:

- Description
- Advantages and Benefits of using RXSQL
- Environment
- RXSQL supplied EXECs
- Tips and Techniques
- Program Offering compared to Program Product

B.1.1 Description

The SQL/DS Procedures Language Interface is the product name for RXSQL. RXSQL is an optional feature of SQL/DS Version 3.

RXSQL allows REXX to access SQL/DS. SQL, the language used to interface to SQL/DS, can be imbedded in procedural languages such as C, PL/1 or COBOL. RXSQL extends this support to REXX EXECs. Unlike the other language interfaces however, RXSQL statements are not preprocessed.

NOTE

This feature has not changed for SQL/DS V3R5 and the existing version for SQL/DS V3R4 can be used with SQL/DS V3R5.

Compiling REXX EXECs has no effect on the RXSQL operations contained in the EXECs.

The REXX EXECs may contain RXSQL commands. RXSQL performs some initial checking on these commands, transforms them into standard run-time SQL operations, and passes them to SQL/DS. Data retrieved from SQL/DS, and warning or error information from SQL/DS, is returned to the EXEC as REXX variables.

REXX variables are used to do the following:

- Hold SQL statements
- Pass input values to SQL statements being processed by SQL/DS
- Retrieve values fetched from the database
- Receive information about the outcome of the SQL statements after they are processed.

RXSQL allows the use of dynamic, as well as extended dynamic, SQL statements.

Dynamic statements support the preparing and executing of SQL statements within a program while the program is running. These dynamically prepared statements last only for the life of the LUW.

Extended dynamic statements support direct creation and maintenance of packages for SQL/DS, and the subsequent use of SQL statements that have been stored in the packages. These statements provide a function similar to that provided by the SQL/DS preprocessors.

RXSQL started out as an internal tool in 1984. In 1986 it became a Program Offering (PO). With the release of SQL/DS V3R1, it became a Program Product (PP). (A PO is software released by IBM, but the purchaser is not guaranteed any regular service. The purchaser must ask his/her SE to get answers to problems, or get fixes. With a PP, full service is available.)

B.1.2 Advantages and Benefits of using RXSQL

- REXX EXECs can retrieve and update information in SQL/DS databases
- No preprocessing or compiling
- Simple vehicle for application prototyping
- Quickly produce ad hoc reports
- Ad hoc queries can be made without ISQL, QMF or AS
- REXX EXECs for data administration tasks
- Monitor database performance using operator commands
- Access to online SQL/DS HELP

B.1.3 Operating Environment

Required Software:

- VM/ESA Version 1 Release 2.1
- SQL/DS V3R3 or later
- REXX (comes with VM).

CMS work unit support came with VM/SP 6. CMS work unit support allows a virtual machine to have multiple independent paths into one or more SQL/DS databases. It is possible to have multiple active SQL/DS logical work units (LUWs), each accessing the same or separate databases. The application program (REXX EXEC) can switch between work units while CMS, the SQL/DS resource manager (RM), and RXSQL all maintain the necessary information.

Before issuing any RXSQL commands, ensure that the following have been done:

- The RXSQL production and SQL/DS production disks have been accessed.
- The default database has been established:

  ```
  SQLINIT DB(dbname)
  ```

- The RXSQL message repository (new to RXSQL 3.1) has been activated:

  ```
  SET LANG (ADD ELO USER
  ```

Figure 85 on page 187 is a pictorial representation of the RXSQL Interface. The numbers in the text following the figure correspond to the numbers in the figure.

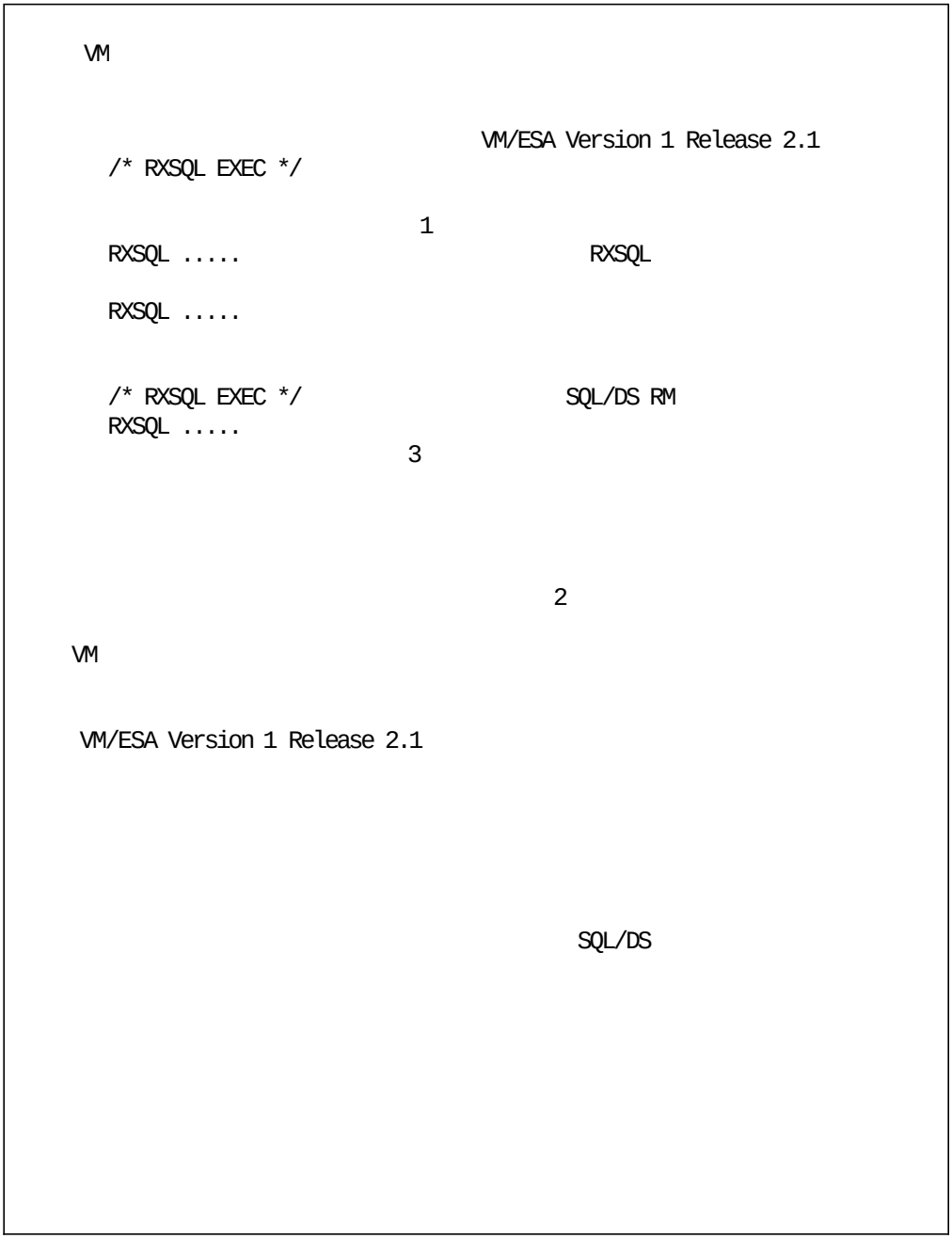


Figure 85. RXSQL Interface

1 The RXSQL code will be loaded as a NUCLEUS extension on the first invocation of RXSQL.

The RXSQL data space (storage for control blocks and so on) is allocated on the first invocation of RXSQL in a REXX program.

2 An RXSQL command is intercepted by RXSQL and transformed into an appropriate SQL/DS command, if necessary. Certain RXSQL commands are meaningful to RXSQL only (SQLISL, STATE) and these are not passed on to SQL/DS. RXSQL passes information to SQL/DS and SQL/DS passes information back to RXSQL. RXSQL makes this information known to the REXX EXEC through REXX variables.

The interface to SQL/DS is through the SQL/DS Resource Adapter, just as any other SQL/DS application program.

3 The RXSQL data space remains until an abend or an end-of-command occurs, or until the nucleus extension is dropped via NUCXDROP RXSQL. It is not recommended that you issue NUCXDROP RXSQL from an RXSQL EXEC, unless you really want to get rid of the current RXSQL environment.

The RXSQL code stays as a nucleus extension until an abend or a NUCXDROP occurs.

B.1.4 RXSQL Supplied EXECs

RXSQL comes with several general purpose EXECs.

- RXSELECT

```
RXSELECT * from system.sysoptions
RXSELECT select * from rxemp where workdept = ¢B09¢
```

RXSELECT will display the output of any valid SQL SELECT statement. RXSELECT uses XEDIT to display the rows it retrieves.

Unless otherwise specified (via the RXCASE command), the SQL statement will be translated to uppercase and passed to SQL/DS.

RXSELECT can be invoked from the CMS command line or from another EXEC. If it is invoked from another EXEC, remember that a COMMIT or a ROLLBACK will be issued from RXSELECT, thus causing the work done in the current logical unit of work to be committed or rolled back.

- RXSQLEX

```
RXSQLEX delete from rxemp where workdept = ¢B09¢
RXSQLEX drop table rxemp
```

RXSQLEX causes an EXECUTE IMMEDIATE to be performed on the statement that has been given to RXSQLEX.

RXSQLEX can be invoked from the CMS command line or from another EXEC. If it is invoked from another EXEC, remember that a COMMIT or a ROLLBACK will be issued from RXSQLEX, thus causing the work done in the current logical unit of work to be committed or rolled back.

- RXCASE

```
RXCASE UPPER
```

RXCASE sets a global variable (in LASTING GLOBALV) that is used by RXSELECT and RXSQLEX to determine whether the SQL statement that it has received should be passed to SQL/DS as entered, or translated to uppercase. The global variable affects only the user who has set it, and it remains in effect until overridden with another invocation of RXCASE.

- RXSQLOP

```
RXSQLOP counter *
RXSQLOP show users
```

Use RXSQLOP to issue operator commands to SQL/DS. RXSQLOP uses XEDIT to display the data retrieved from SQL/DS.

RXSQLOP can be invoked from the CMS command line or from another EXEC. If it is invoked from another EXEC, the current logical unit of work should be committed or rolled back before invoking the operator command.

- RXSQLVL

RXSQLVL will find out what the current level of RXSQL is, and depending on how it has been invoked, either display the result on the screen, or put it on the stack.

- RXSQLHLP

  RXSQLHLP retrieves help information from the RXSQL and SQL/DS help tables.

- RXSQLANG

  RXSQLANG is used when different national languages (for example, American English and French) are installed for RXSQL and SQL/DS, and there is a requirement to override the default languages when displaying help via RXSQLHLP.

B.1.5 Program Offering - Program Product

PO = Program Offering which works with SQL/DS up to and including V2R2

PP = Program Product which is a feature of SQL/DS V3R1

The names of some RXSQL predefined variables have changed. The PO used the following variables:

- SQLSTATE - output from RXSQL STATE operation
- SQLNAMES - output from RXSQL NAMES operation
- SQLSTMT - output from RXSQL STMT operation

The names used in the PP are now (respectively):

- RXSQLSTATE
- RXSQLNAMES
- RXSQLSTMT

- New SQL/DS variable - SQLSTATE

  In SQL/DS V3R1, there was a new variable returned by SQL/DS in the SQLCA called SQLSTATE.

- RXSQL-specific HELP

  In the PO, RXSQLHLP (RXSQL-supplied EXEC to display HELP information) displayed only SQL/DS help information. With the PP, RXSQLHLP will display RXSQL, as well as SQL/DS, help information.

- CMS Message Repository

  The PP uses a CMS message repository.

- Installation and Service EXECs

  New installation EXECs have been provided. As well, since RXSQL will be serviced through the standard IBM channels, corrective and preventive service EXECs are supplied. A portable package is now provided, along with an EXEC to install it in your database(s). This precludes the need for doing an SQL/DS prep during installation.

B.2 SQL/DS Version 3 VMDSS Feature

This topic describes the concept of VMDSS, how it works, and how it can improve performance.

The VMDSS Feature was introduced with SQL/DS V3R3. The VMDSS Feature uses the VM Data Spaces Support, available with VM/ESA 1.0 and later.

NOTE

This feature has been updated for SQL/DS V3R5 and the new level **MUST** be used but the feature has functionally **NOT** changed.

B.2.1 What is VMDSS and How it Works

The SQL/DS VM Data Space Support Feature (VMDSS) is designed to significantly enhance the performance of SQL/DS in the VM/ESA environment, by exploiting the VM/ESA Data Spaces Support. VMDSS uses a high performance DASD I/O system which gives a more effective way of retrieving data than the standard IUCV *BLOCKIO system.

An increase in performance is expected in areas where large amounts of database I/O operations are involved.
For example during:

- Large complex queries
- Large scans
- Very frequently accessed tables

This feature affects how the SQL/DS product works with the base operating system. It does not affect how users or application programs interact with the SQL/DS product. This means that there is no need to change existing application programs or change the way users work with the SQL/DS product.

Most of the support provided by VMDSS is transparent to the users. It is important to realize however, that if a program is executing a large dbspace scan because of a lack of indexes or appropriate predicates, for example, **VMDSS will not change the efficiency of the program itself**. VMDSS does not modify the performance characteristics of existing applications. However, VMDSS can make the dbspace scan execute more efficiently. Further performance enhancements can be achieved by adjusting the applications.

VMDSS does require planning, tuning and monitoring in order to make optimum use of this feature. There are new tuning parameters and also new performance monitoring commands added to help do these tasks in a more effective way.

B.2.2 Expected and/or Potential Benefits

The following benefits are expected when using VMDSS:

- The number of instructions required to access a data space page is far less than the number required to perform an equivalent IUCV *BLOCKIO request.
- Multiple pages can be blocked into a single I/O, thus reducing the total number of I/Os.

- The VM REFPAGE macro allows VM users to request prefetching. The REFPAGE interface is used by VMDSS to inform VM of an anticipated sequence of reference. VM adjusts its reference pattern to match the anticipated reference and prefetches, considering other system loads.
- Page writes are completely asynchronous when using data spaces.
  SQL/DS can continue processing in parallel with write I/O even when only one agent is active.
- Checkpoint I/O can be more asynchronous.
  Because of the asynchronous saves between checkpoints, and at the start of checkpoints, saving the buffers to disk will be faster.
- It is possible to use the VM system paging DASD for internal dbspaces instead of DASD allocation, leaving more space for permanent data and also improving performance because of VM′s efficient paging subsystem.
- Data can be spread across DASD volumes in a more efficient way with the new allocation algorithm called *striping*. This provides less I/O contention on the DASD devices.

B.2.3 Basic Requirements for Using VMDSS

To make use of all functions provided by VMDSS, the following is required:

- ES/9000 family of processors
- VM/ESA Version 1 Release 2.1 or later
- Virtual machine running in XC mode
  This can be done by updating the MACHINE statement in the CP user directory.
- XCONFIG statements in CP user directory
  For more information on the MACHINE and XCONFIG directory statements, refer to the *VM/ESA CP Planning and Administration* manual.
- VMDSS feature code (5688-103-03) is installed
  Use the I5688VMD EXEC provided with the feature tape to load the VMDSS system files.
  VMDSS replaces some SQL/DS modules. Rebuild the database manager with the VMDSS Feature by link-editing the SQL/DS DBSS component. For a detailed description of using the ARISLKIT EXEC refer to the *SQL/DS Database Administration* manual.
- Reblocking of the database directory may be needed
  The SQL/DS database directory can be used with either data spaces support or the standard IUCV *BLOCKIO system.
  The directory disk (BDISK) must have a 4096 byte (4KB) blocksize to be mapped to a data space. The database manager will automatically use data spaces when it detects the 4KB blocks.
  The SQLCDBEX EXEC is updated for the VMDSS Feature, and now asks which blocksize is required for the SQL/DS directory.
  For more information on the standard SQLCDBEX EXEC refer to the *SQL/DS System Administration* manual.
- Create the VMDSS storage pool specifications
  These specifications turn data space support on and off, set storage residency priorities, and turn striping on and off for the individual storage pools.
- Provide the new VMDSS initialization parameters

These parameters set the application server's save interval, target working storage size, and whether mapped or unmapped internal dbspaces will be used.

B.3 SQL/DS Version 3 DataHub Feature

The objective of this topic is to describe DataHub, and the benefits it brings to the installation by providing integrated database systems management functions for IBM Systems Application Architecture (SAA) relational database management systems (RDBMSs).

NOTE

This feature has not changed for SQL/DS V3R5 and the version for SQL/DS V3R4 can be used.

B.3.1 What is DataHub

DataHub is a family of products designed to meet the challenges of managing complex database environments while increasing productivity and reducing costs. The DataHub family of products provides integrated database systems management functions for IBM Systems Application Architecture (SAA) relational database management systems (RDBMSs).

The DataHub user control point for database systems management is an OS/2 based personal computer workstation. This DataHub/2 workstation provides common services for working with DataHub Support components at the hosts and for the display of information in the DataHub/2 window. These common services are used by programmers writing database systems management tools that can be invoked from the DataHub/2 workstation.

Using the Platform feature of DataHub/2 it is possible at the workstation to:

- Display database objects in the managed databases
- Run files of JCL, SQL or AS/400 authorization statements
- Set options for the DataHub/2 sessions
- Manage the profile information.

Additional functions are provided by the DataHub/2 Tools feature, addressing database systems management tasks in the areas of operations, configuration, problem resolution and security management. The DataHub/2 Tools feature provides functions to:

- Copy database objects such as tables and views between and within RDBMSs
- Display the status of database activity
- Manage user and object authorizations
- Back up, recover and reorganize data.

The functions of the Platform and Tools features help simplify the tasks of database administrators, system administrators, help desk personnel, system programmers and application programmers in distributed and non-distributed database environments.

DataHub Release 1 supports the following IBM SAA RDBMSs:

- DATABASE 2 (DB2) on MVS
- SQL/DS on VM/ESA
- Operating System/400 (OS/400) database
- Database Manager for OS/2.

The DataHub family of products can be used for database systems management in distributed and non-distributed database environments. DataHub capabilities allow the user to move into the distributed database environment more easily and to manage that environment more effectively.

The DataHub approach to database systems management allows the user to more effectively:

- Distribute data across the enterprise by copying database objects from one RDBMS to another
- Perform database systems management tasks across multiple like or unlike RDBMSs
- Perform database systems management tasks across multiple sites
- Speed up problem detection and minimize down-time
- Reduce skill requirements by hiding most of the differences between IBM RDBMSs.

Figure 86 on page 194 shows the DataHub family of products in a distributed environment.

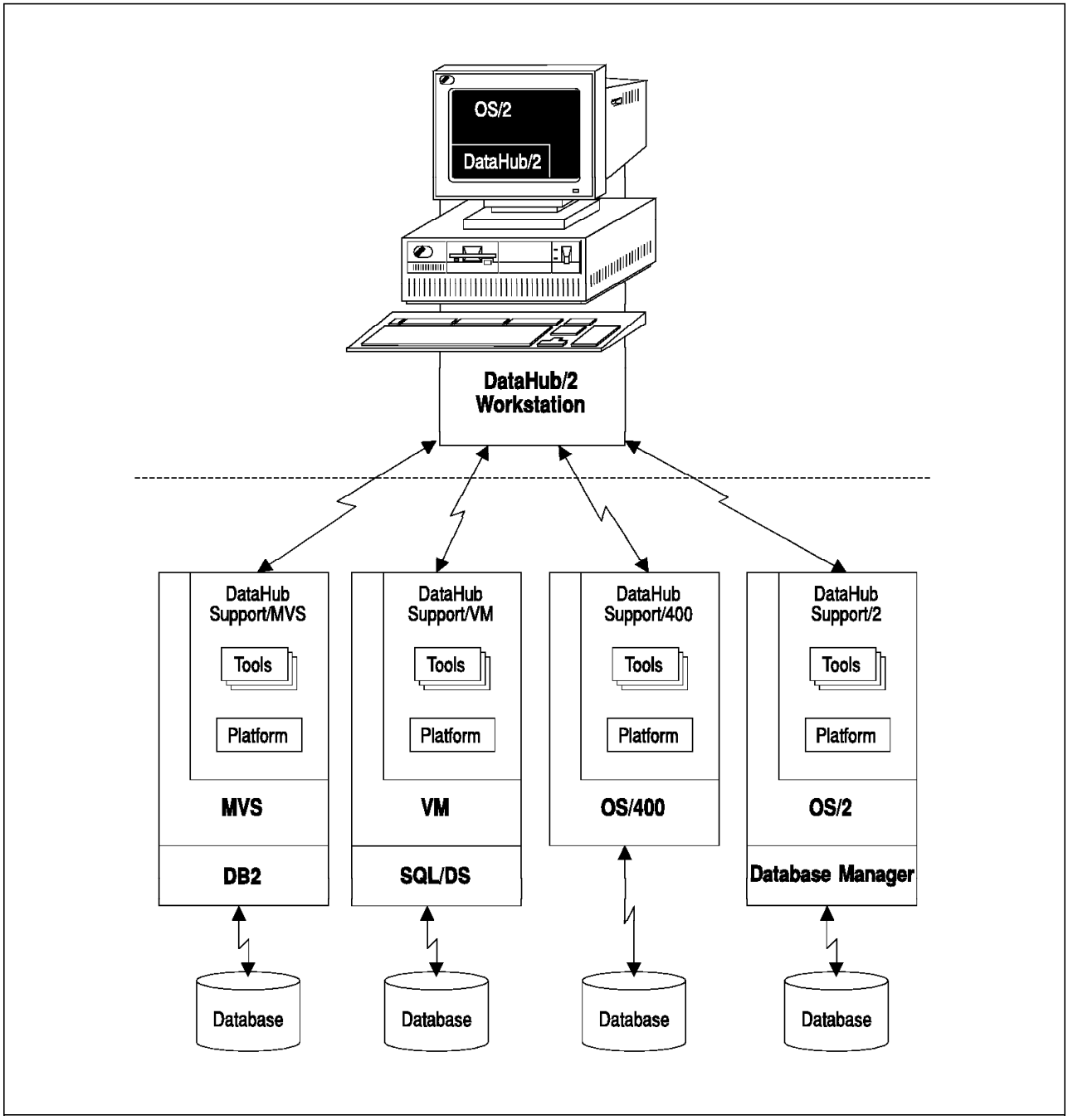


Figure 86. The DataHub Family of Products

The diagram above shows the DataHub family of products, installed on a DataHub/2 workstation and DataHub Support hosts.

DataHub/2 workstation

The DataHub/2 workstation is the point of control from which the user can perform database administration tasks. The DataHub/2 has the DataHub/2 Platform feature installed, and optionally, the DataHub/2 Tools feature.

The DataHub/2 Platform feature can be described in terms of two aspects of its function:

DataHub/2 window

The DataHub/2 window is the graphical user interface from which the user displays and accesses objects and uses database systems management tools. Menus, pop-up windows, messages, information reports and database objects are displayed in the DataHub/2 window.

DataHub/2 common services

The DataHub/2 common services is the software that provides communications and run-time function between the DataHub/2 workstation and the hosts. These common services facilitate the development, integration and execution of database systems management tools.

The *DataHub Tool Builder's Guide and Reference* describes how to use the DataHub common services when developing database systems management products.

DataHub Tools feature

The DataHub/2 Tools feature provides the database systems management functions such as copying objects and managing authorizations.

DataHub Support

This is a support component installed on each of the hosts. The DataHub host support component is part of DataHub's extended tools integration services and must be installed on each host managed by the DataHub/2 workstation. The host support components provide services for their respective relational database management systems as follows:

- **DataHub Support/MVS** for DB2 on MVS
- **DataHub Support/VM** for SQL/DS on VM/ESA
- **DataHub Support/400** for OS/400 database
- **DataHub Support/2** for Database Manager on OS/2.

There are Tools feature software components for both the workstation and the hosts.

It is possible to perform systems management tasks on relational data anywhere in the enterprise from a DataHub/2 workstation.

Appendix C. Field Procedures for Cultural Sorts

By default, string data is sorted based on the System/390 collating sequence. However, the collating sequence required for certain alphabets is different from the default System/390 collating sequence. Users expect that sorted data will match the order that is culturally correct for them, and that searches on data will return the result that is correct for the sorting sequence of their language. They are at ease with only one sort order, the one used in their dictionaries, telephone directories, book indices, and so on.

A way to accommodate special sorting requirements is to use Field Procedures. Field Procedures can be used to encode data being inserted into a column. The encoding effectively alters the collating sequence for the data in the column, enabling the special sorting requirements to be met by the System/390 collating sequence.

Two field procedures are provided. For a VM database, they are found on the database machine′s service disk. For a VSE database, they are supplied as A-type members.

The field procedures provided are:

- FP870L2 for Slovenia, Poland and Romania
- FP102CY for Russia, Bulgaria, Serbia and Montenegro

The field procedures are written in Assembler. In VM, the field procedure must be assembled and the corresponding module must be generated and placed on a disk that is accessible to the database manager when it is running. Note that in VM, the MACLIB FLDPROC (which is provided with SQL/DS) must be specified on the GLOBAL MACLIB statement in order to assemble the field procedure and generate the module. In VSE, the field procedure must be assembled and the corresponding phase must be generated and placed in a library that is accessible to the database manager when it is running.

Once the phase or module for the field procedure has been generated and made accessible to the database manager, it can be used by specifying its name in the FIELDPROC clause of the CREATE TABLE or ALTER TABLE statement.

For further information about implementing and using field procedures, refer to the *SQL/DS System Administration for IBM VM Systems*, or *SQL/DS System Administration for VSE* manual.

Appendix D. Database Directory Expansion Examples

In this example, the directory of an SQL/DS V3R4 database was expanded, and the directory of an SQL/DS V3R5 directory was expanded using the enhanced directory expansion. The directory expansion processes were run on both VM/ESA and VSE/ESA.

All the database machines were generated with the same definitions. A description of these machines can be found in 3.2.1, “Test Environment” on page 89.

In all cases, the directory disk was expanded from 40 cylinders to 60 cylinders, using 3380 DASD.

D.1 Directory Expansion on SQL/DS V3R4 under VM

Before expanding the directory, we issued the SHOW DBCONFIG command in order to compare the numbers available for logical and physical space. The results are shown in Figure 87.

```
show dbconfig
System identification at DB generation = SQL/DS VERSION 3

DBA specified the following at DB definition:
  Maximum pools =             999
  Maximum DBEXTENTS =         999
  Maximum DBSPACES =          10240

Computed:
  Total number of DBSPACE blocks =    25739
  Total amount of DBSPACE =            6589184 K
  Total number of physical pages =    1662976
  Total amount of physical space =     6651904 K
  Total number of directory blocks =  27362

  Number of DBSPACE blocks left =     631
ARI0065I SQL/DS operator command processing is complete.
```

Figure 87. SHOW DBCONFIG before Directory Expansion (SQL/DS V3R4)

After this, the database machine was stopped to run the SQLCDBEX EXEC from SQL/DS V3R4. The console log from this process is shown in Figure 88 on page 201.

```
sqlcdbex dbname(sqlvm340)
ARI0717I Start SQLCDBEX EXEC: 09/28/95 16:35:30 CET.
ARI0721I Get SQL/DS production minidisk WRITE access: SQLMACH 195.
DASD 0195 LINKED R/W; R/O BY     4 USERS

ARI6102A Enter DBEXTENT number (or LOGDSK1, LOGDSK2,
         or BDISK) to copy.
         (Enter a null response to end input or
         enter QUIT to exit.)
bdisk
ARI6188A Enter the output block size of the directory.
         Enter 512 or 4096,
         or a null response to use the original size,
         or 111(Quit) to exit.
512
ARI6103A Enter virtual address for new BDISK.
         (Enter a null response to end input or
         enter QUIT to exit.)
300
DMSACP112S C(300) device error
ARI6148I New disk 300 has not been formatted. Program will continue
         to format the disk before copying.
ARI6146D Are you expanding the SQL/DS directory?
         Enter 0(No), 1(Yes), or 111(Quit).
1
ARI6118I Formatting in progress. Please wait...
ARI6130I Disk 300 is formatted successfully.
ARI6131I Copying in progress. Please wait...
ARI0640I 4000 of
         41046 records copied to output disk.
ARI0640I 8000 of
         41046 records copied to output disk.
ARI0640I 12000 of
         41046 records copied to output disk.
ARI0640I 16000 of
         41046 records copied to output disk.
ARI0640I 20000 of
         41046 records copied to output disk.
ARI0640I 24000 of
         41046 records copied to output disk.
ARI0640I 28000 of
         41046 records copied to output disk.
ARI0640I 32000 of
         41046 records copied to output disk.
ARI0640I 36000 of
         41046 records copied to output disk.
ARI0640I 40000 of
         41046 records copied to output disk.
ARI6108I Minidisk copied successfully. The SQLVM340 SQLFDEF file
         will be updated.
ARI6109I SQLVM340 SQLFDEF file has been updated on the A disk.

ARI6102A Enter DBEXTENT number (or LOGDSK1, LOGDSK2,
         or BDISK) to copy.
         (Enter a null response to end input or
         enter QUIT to exit.)
ARI0620I SQLVM340 SQLFDEF file
         successfully copied to production disk.
ARI0673I All COPY DBEXTENT processing completed successfully.
ARI0796I End SQLCDBEX EXEC: 09/28/95 17:07:00 CET
ARI0721I Get SQL/DS production minidisk READ access: SQLMACH 195.
Ready; T=87.49/103.73 17:07:06
```

Figure 88. SQLCDBEX EXEC with SQL/DS V3R4

After successfully running the directory expansion, we issued the SHOW DBCONFIG command again, as shown in Figure 89 on page 202.

```
show dbconfig
System identification at DB generation = SQL/DS VERSION 3

DBA specified the following at DB definition:
  Maximum pools =             999
  Maximum DBEXTENTS =         999
  Maximum DBSPACES =          10240

Computed:
  Total number of DBSPACE blocks =    39415
  Total amount of DBSPACE =            10090240 K
  Total number of physical pages =    1662976
  Total amount of physical space =     6651904 K
  Total number of directory blocks =  41046

  Number of DBSPACE blocks left =     14307
ARI0065I SQL/DS operator command processing is complete.
```

Figure 89. SHOW DBCONFIG after Directory Expansion (SQL/DS V3R4)

Note, the number of dbspace blocks and total amount of dbspace have increased. The total number of directory blocks has increased. The total number of physical pages and amount of physical space have not increased.

D.2 Directory Expansion on SQL/DS V3R5 under VM

The next step was to execute the directory expansion process against our SQL/DS V3R5 database machine, this time using the enhanced directory expansion program.

We repeated the steps as in the previous SQL/DS V3R4 case. Before expanding the directory, we executed the SHOW DBCONFIG command in order to compare the numbers later.

The results from this command are shown in Figure 90.

```
show dbconfig
System identification at DB generation = SQL/DS VERSION 3

DBA specified the following at DB definition:
  Maximum pools =             999
  Maximum DBEXTENTS =         999
  Maximum DBSPACES =          10240

Computed:
  Total number of DBSPACE blocks =    25739
  Total amount of DBSPACE =            6589184 K
  Total number of physical pages =    1662976
  Total amount of physical space =     6651904 K
  Total number of directory blocks =  27362

  Number of DBSPACE blocks left =     951
ARI0065I SQL/DS operator command processing is complete.
```

Figure 90. SHOW DBCONFIG before Directory Expansion (SQL/DS V3R5)

After the command was issued, the SQL/DS V3R5 database machine was stopped and the enhanced SQLCDBEX EXEC was run. The console log of this execution is shown in Figure 91 on page 204.

```
sqlcdbex dbname(sqlvm350)
ARI0717I Start SQLCDBEX EXEC: 09/28/95 16:41:50 CET.
ARI0721I Get SQL/DS production minidisk WRITE access: SQLDB350 195.
DASD 0195 LINKED R/W; R/O BY     2 USERS

ARI6102A Enter DBEXTENT number (or LOGDSK1, LOGDSK2,
         or BDISK) to copy.
         (Enter a null response to end input or
         enter QUIT to exit.)
bdisk
ARI6188A Enter the output block size of the directory.
         Enter 512 or 4096,
         or a null response to use the original size,
         or 111(Quit) to exit.
512
ARI6103A Enter virtual address for new BDISK.
         (Enter a null response to end input or
         enter QUIT to exit.)
300
DMSACP112S C(300) device error
ARI6148I New disk 300 has not been formatted. Program will continue
         to format the disk before copying.
ARI6146D Are you expanding the SQL/DS directory?
         Enter 0(No), 1(Yes), or 111(Quit).
1
ARI6118I Formatting in progress. Please wait...
ARI6130I Disk 300 is formatted successfully.
ARI6200D You have requested to expand the directory.
         Enter 1 to expand the directory to hold more DBSPACE
         pages. Enter 2 to expand the directory to hold more
         DBSPACE and DBEXTENT pages. Enter 111 to quit.
         Enter 1,2 or 111(Quit).
2
ARI6198D Current maximum DBEXTENT pages: 1662976
         New maximum DBEXTENT pages: 2502656
         DBEXTENT pages added: 839680
         Current maximum DBSPACE pages: 3294592
         New maximum DBSPACE pages: 4964352
         DBSPACE pages added: 1669760
         Current directory size: 27362
         Current directory block size: 512
         New directory size: 41046
         New directory block size: 512
         Do you wish to continue with expanding the
         directory to allow the directory to hold
         additional DBEXTENT and DBSPACE pages ?
         Enter 0(No), 1(Yes) or 111(Quit)
1
```

Figure 91 (Part 1 of 2). SQLCDBEX EXEC with SQL/DS V3R5

```
ARI6131I Copying in progress. Please wait...
ARI0640I 4000 of
         41046 records copied to output disk.
ARI0640I 8000 of
         41046 records copied to output disk.
ARI0640I 12000 of
         41046 records copied to output disk.
ARI0640I 16000 of
         41046 records copied to output disk.
ARI0640I 20000 of
         41046 records copied to output disk.
ARI0640I 24000 of
         41046 records copied to output disk.
ARI0640I 28000 of
         41046 records copied to output disk.
ARI0640I 32000 of
         41046 records copied to output disk.
ARI0640I 36000 of
         41046 records copied to output disk.
ARI0640I 40000 of
         41046 records copied to output disk.
ARI6201I Directory expansion completed successfully.
         It is strongly recommended that a database
         archive be taken immediately.
ARI6108I Minidisk copied successfully. The SQLVM350 SQLFDEF file
         will be updated.
ARI6109I SQLVM350 SQLFDEF file has been updated on the A disk.

ARI6102A Enter DBEXTENT number (or LOGDSK1, LOGDSK2,
         or BDISK) to copy.
         (Enter a null response to end input or
         enter QUIT to exit.)
ARI0620I SQLVM350 SQLFDEF file
         successfully copied to production disk.
ARI0673I All COPY DBEXTENT processing completed successfully.
ARI0796I End SQLCDBEX EXEC: 09/28/95 17:11:35 CET
ARI0721I Get SQL/DS production minidisk READ access: SQLDB350 195.
Ready; T=90.80/106.77 17:11:40
```

Figure 91 (Part 2 of 2). SQLCDBEX EXEC with SQL/DS V3R5

Note that SQLCDBEX EXEC now issues two new messages: ARI6200D and ARI6198D. The first one, ARI6200D, lets the user decide what type of directory expansion will happen. The user can expand the directory to hold more dbspace pages or more dbspace and physical pages.

If the user decides to expand the directory to hold more dbspace and dbextent pages, then the message ARI6198D will show the current size and the new size, and will prompt the user to decide if the process should continue or to quit.

After the SQLCDBEX EXEC completed successfully, the database machine was restarted to issue the SHOW DBCONFIG command, the result of which is in Figure 92 on page 206.

```
show dbconfig
System identification at DB generation = SQL/DS VERSION 3

DBA specified the following at DB definition:
  Maximum pools =             999
  Maximum DBEXTENTS =         999
  Maximum DBSPACES =          10240

Computed:
  Total number of DBSPACE blocks =    38784
  Total amount of DBSPACE =            9928704 K
  Total number of physical pages =    2502656
  Total amount of physical space =     10010624 K
  Total number of directory blocks =  41046

  Number of DBSPACE blocks left =     13996
ARI0065I SQL/DS operator command processing is complete.
```

Figure 92. SHOW DBCONFIG after Directory Expansion (SQL/DS V3R5)

Note that after the SQL/DS V3R5 directory expansion, the total number of physical pages and total amount of physical space have increased. The total number of dbspace blocks has increased, as in SQL/DS V3R4, although there are slightly less, due to the expansion of the physical pages.

D.3 Directory Expansion on SQL/DS V3R4 under VSE

Before expanding the directory, we issued the SHOW DBCONFIG command in order to compare the numbers available for logical and physical space. The results are shown in Figure 93.

```
7 show dbconfig
F7 007 System identification at DB generation = SQL/DS VERSION 3
F7 007
F7 007 DBA specified the following at DB definition:
F7 007   Maximum pools =             999
F7 007   Maximum DBEXTENTS =         999
F7 007   Maximum DBSPACES =          10240
F7 007
F7 007 Computed:
F7 007   Total number of DBSPACE blocks =    23102
F7 007   Total amount of DBSPACE =            5914112 K
F7 007   Total number of physical pages =    1495040
F7 007   Total amount of physical space =     5980160 K
F7 007   Total number of directory blocks =  24600
F7 007
F7 007   Number of DBSPACE blocks left =     5770
F7 007 ARI0065I SQL/DS operator command processing is complete.
```

Figure 93. SHOW DBCONFIG before Directory Expansion (SQL/DS V3R4)

After this, the database machine was stopped to run the ARIMEXBD Job from SQL/DS V3R4. The example of the job control is shown in Figure 94 on page 208.

```
// JOB ARIMEXBD EXPAND SQL/DS V3R4 DIRECTORY TO 60 CYLINDERS
// SETPFIX LIMIT=1024K
// OPTION  NOFASTTR
* ********************************************************
* * START ARIMEXBD EXPAND DIRECTORY FROM 40 to 60 CYLINDERS *
* ********************************************************
// LIBDEF  PROC,SEARCH=(PRD2.SQL340)
// DLBL    IJSYSUC,¢SQLPCAT.USER.CATALOG¢,,VSAM
// DLBL    BDSKNEW,¢SQLVS340.BDISK.SQLDIR60¢,,VSAM,CAT=SQLPCAT
// EXEC    PROC=ARIS34SL *-- SQL/DS PRODUCTION LIBRARY ID PROC
// EXEC    PROC=SQLVS340 *-- SQL/DS SQLVS340 DATABASE ID PROC
// PROC CAT=¢SQLPCAT¢
// DLBL IJSYSUC,¢SQLPCAT.USER.CATALOG¢,,VSAM
* ********************************************************
* SQLVS340: SQL/DS DATABASE IDENTIFICATION
* ********************************************************
// DLBL BDISK,¢SQLVS340.BDISK.SQLDIR40¢,,VSAM,CAT=SQLPCAT
// DLBL LOGDSK1,¢SQLVS340.LOGDSK1.SQLLOG¢,,VSAM,CAT=SQLPCAT
// DLBL DDSK1,¢SQLVS340.DDSK1.POOL1¢,,VSAM,CAT=SQLPCAT
// DLBL DDSK2,¢SQLVS340.DDSK2.POOL1¢,,VSAM,CAT=SQLPCAT
// DLBL DDSK3,¢SQLVS340.DDSK3.POOL2¢,,VSAM,CAT=SQLPCAT
// DLBL DDSK4,¢SQLVS340.DDSK4.POOL2¢,,VSAM,CAT=SQLPCAT
// DLBL DDSK5,¢SQLVS340.DDSK5.POOL2¢,,VSAM,CAT=SQLPCAT
// DLBL DDSK6,¢SQLVS340.DDSK6.POOL3¢,,VSAM,CAT=SQLPCAT
// DLBL DDSK7,¢SQLVS340.DDSK7.POOL3¢,,VSAM,CAT=SQLPCAT
// DLBL DDSK8,¢SQLVS340.DDSK8.POOL3¢,,VSAM,CAT=SQLPCAT
// DLBL DDSK9,¢SQLVS340.DDSK9.POOL4¢,,VSAM,CAT=SQLPCAT
// DLBL DDSK10,¢SQLVS340.DDSK10.POOL4¢,,VSAM,CAT=SQLPCAT
// DLBL DDSK11,¢SQLVS340.DDSK11.POOL4¢,,VSAM,CAT=SQLPCAT
// EXEC    ARIMEXBD,SIZE=AUTO
EOJ ARIMEXBD  MAX.RETURN CODE=0000
```

Figure 94. ARIMEXBD Job for SQL/DS V3R4

The console log from this process is shown in Figure 95.

```
F7 007 // JOB SQLVS34X EXPAND DIRECTORY FROM 40 TO 60 CYLINDERS
   DATE 10/24/95,CLOCK 21/14/11
F7 007 * ********************************************************
F7 007 * * START SQLVS34X EXPAND DIRECTORY FROM 40 to 60 CYLINDERS *
F7 007 * ********************************************************
F7 007 * ********************************************************
F7 007 * SQLVS340: SQL/DS DATABASE IDENTIFICATION
F7 007 * ********************************************************
F7 007 ARI0943I Directory expansion was completed successfully.
F7 007 EOJ SQLVS34X  MAX.RETURN CODE=0000
```

Figure 95. ARIMEXBD Job with SQL/DS V3R4

After successfully having run the directory expansion, we issued again the SHOW DBCONFIG command as shown in Figure 96 on page 209.

```
7 show dbconfig
F7 007 System identification at DB generation = SQL/DS VERSION 3
F7 007
F7 007 DBA specified the following at DB definition:
F7 007   Maximum pools =             999
F7 007   Maximum DBEXTENTS =         999
F7 007   Maximum DBSPACES =          10240
F7 007
F7 007 Computed:
F7 007   Total number of DBSPACE blocks =    35396
F7 007   Total amount of DBSPACE =            9061376 K
F7 007   Total number of physical pages =    1495040
F7 007   Total amount of physical space =     5980160 K
F7 007   Total number of directory blocks =  36900
F7 007
F7 007   Number of DBSPACE blocks left =     18064
F7 007 ARI0065I SQL/DS operator command processing is complete.
```

Figure 96. SHOW DBCONFIG after Directory Expansion (SQL/DS V3R4)

Note, here the number of dbspace blocks and total amount of dbspace have increased. The total number of directory blocks has increased. The total number of physical pages and amount of physical space have not increased.

D.4 Directory Expansion on SQL/DS V3R5 under VSE

The next step was to execute the directory expansion process against our SQL/DS V3R5 VSE database machine, this time, using the enhanced directory expansion program.

We repeated the steps as in the previous SQL/DS V3R4 case. Before expanding the directory, we executed the SHOW DBCONFIG command in order to compare the numbers later.

The results from this command are shown in Figure 97.

```
8 show dbconfig
F8 008 System identification at DB generation = SQL/DS VERSION 3
F8 008
F8 008 DBA specified the following at DB definition:
F8 008   Maximum pools =             999
F8 008   Maximum DBEXTENTS =         999
F8 008   Maximum DBSPACES =          10240
F8 008
F8 008 Computed:
F8 008   Total number of DBSPACE blocks =    23102
F8 008   Total amount of DBSPACE =            5914112 K
F8 008   Total number of physical pages =    1495040
F8 008   Total amount of physical space =     5980160 K
F8 008   Total number of directory blocks =  24600
F8 008
F8 008   Number of DBSPACE blocks left =     5770
F8 008 ARI0065I SQL/DS operator command processing is complete.
```

Figure 97. SHOW DBCONFIG before Directory Expansion (SQL/DS V3R5)

After the command was issued, the SQL/DS V3R5 database machine was stopped and the enhanced ARIMEXBD Job was run. The example of the job control is shown in Figure 98 on page 211.

```
// JOB ARIMEXBD EXPAND SQL/DS V3R5 DIRECTORY TO 60 CYLINDERS
// SETPFIX LIMIT=1024K
// OPTION  NOFASTTR
* ********************************************************
* * START ARIMEXBD EXPAND DIRECTORY FROM 40 to 60 CYLINDERS *
* ********************************************************
// LIBDEF  PROC,SEARCH=(PRD2.SQL350)
// DLBL    IJSYSUC,¢SQLPCAT.USER.CATALOG¢,,VSAM
// DLBL    BDSKNEW,¢SQLVS350.BDISK.SQLDIR60¢,,VSAM,CAT=SQLPCAT
// EXEC    PROC=ARIS35SL *-- SQL/DS PRODUCTION LIBRARY ID PROC
// EXEC    PROC=SQLVS350 *-- SQL/DS SQLVS350 DATABASE ID PROC
// PROC CAT=¢SQLPCAT¢
// DLBL IJSYSUC,¢SQLPCAT.USER.CATALOG¢,,VSAM
* ********************************************************
* SQLVS350: SQL/DS DATABASE IDENTIFICATION
* ********************************************************
// DLBL BDISK,¢SQLVS350.BDISK.SQLDIR40¢,,VSAM,CAT=SQLPCAT
// DLBL LOGDSK1,¢SQLVS350.LOGDSK1.SQLLOG¢,,VSAM,CAT=SQLPCAT
// DLBL DDSK1,¢SQLVS350.DDSK1.POOL1¢,,VSAM,CAT=SQLPCAT
// DLBL DDSK2,¢SQLVS350.DDSK2.POOL1¢,,VSAM,CAT=SQLPCAT
// DLBL DDSK3,¢SQLVS350.DDSK3.POOL2¢,,VSAM,CAT=SQLPCAT
// DLBL DDSK4,¢SQLVS350.DDSK4.POOL2¢,,VSAM,CAT=SQLPCAT
// DLBL DDSK5,¢SQLVS350.DDSK5.POOL2¢,,VSAM,CAT=SQLPCAT
// DLBL DDSK6,¢SQLVS350.DDSK6.POOL3¢,,VSAM,CAT=SQLPCAT
// DLBL DDSK7,¢SQLVS350.DDSK7.POOL3¢,,VSAM,CAT=SQLPCAT
// DLBL DDSK8,¢SQLVS350.DDSK8.POOL3¢,,VSAM,CAT=SQLPCAT
// DLBL DDSK9,¢SQLVS350.DDSK9.POOL4¢,,VSAM,CAT=SQLPCAT
// DLBL DDSK10,¢SQLVS350.DDSK10.POOL4¢,,VSAM,CAT=SQLPCAT
// DLBL DDSK11,¢SQLVS350.DDSK11.POOL4¢,,VSAM,CAT=SQLPCAT
// EXEC    ARIMEXBD,SIZE=AUTO,PARM=¢EXPAND=ALL¢
EOJ ARIMEXBD  MAX.RETURN CODE=0000
```

Figure 98. ARIMEXBD Job for SQL/DS V3R5

The console log of this execution is shown in Figure 99.

```
F8 008 // JOB SQLVS35X EXPAND DIRECTORY FROM 40 TO 60 CYLINDERS
   DATE 10/24/95,CLOCK 21/24/10
F8 008 * ********************************************************
F8 008 * * START SQLVS35X EXPAND DIRECTORY FROM 40 to 60 CYLINDERS *
F8 008 * ********************************************************
F8 008 * ********************************************************
F8 008 * SQLVS350: SQL/DS DATABASE IDENTIFICATION
F8 008 * ********************************************************
F8 008 ARI0946I Expanded space will be used for DBSPACE and DBEXTENT
F8 008 ARI0947I CURRENT maximum DBEXTENT pages: 1495040
F8 008 ARI0947I NEW maximum DBEXTENT pages: 2248704
F8 008 ARI0948I DBEXTENT pages added: 753664
F8 008 ARI0947I CURRENT maximum DBSPACE pages: 2957056
F8 008 ARI0947I NEW maximum DBSPACE pages: 4458240
F8 008 ARI0948I DBSPACE pages added: 1501184
F8 008 ARI0949I CURRENT directory size: 24600
F8 008 ARI0949I CURRENT directory block size: 512
F8 008 ARI0949I NEW directory size: 36900
F8 008 ARI0949I NEW directory block size: 512
F8 008 ARI0943I Directory expansion was completed successfully.
F8 008 ARI0924I It is recommended that a database archive be taken.
F8 008 EOJ SQLVS35X  MAX.RETURN CODE=0000
```

Figure 99. ARIMEXBD Job with SQL/DS V3R5

Note that ARIMEXBD Job now gives details of the page usage before and after the expansion.

After the ARIMEXBD Job completed successfully, the database machine was restarted to issue the SHOW DBCONFIG command, the results of which are in Figure 100 on page 212.

```
8 show dbconfig
F8 008 System identification at DB generation = SQL/DS VERSION 3
F8 008
F8 008 DBA specified the following at DB definition:
F8 008   Maximum pools =              999
F8 008   Maximum DBEXTENTS =         999
F8 008   Maximum DBSPACES =          10240
F8 008
F8 008 Computed:
F8 008   Total number of DBSPACE blocks =    34830
F8 008   Total amount of DBSPACE =            8916480 K
F8 008   Total number of physical pages =    2248704
F8 008   Total amount of physical space =     8994816 K
F8 008   Total number of directory blocks =  36900
F8 008
F8 008   Number of DBSPACE blocks left =     17498
F8 008 ARI0065I SQL/DS operator command processing is complete.
```

Figure 100. SHOW DBCONFIG after Directory Expansion (SQL/DS V3R5)

Note that after the SQL/DS V3R5 directory expansion, the total number of physical pages and total amount of physical space have increased, as well as the dbspace blocks.

D.5 Conclusions

When using directory expansion with SQL/DS V3R4, only the dbspace pages can be increased. The dbextent pages cannot be increased. The user running SQL/DS V3R5 can obtain this same result, by selecting option 1 (VM) or option PARM='EXPAND=DBSPACE' (VSE) when running the Directory Expansion Program.

In SQL/DS V3R5, the enhanced SQLCDBEX EXEC (VM) and ARIMEXBD Job (VSE) allow the user to expand both dbspace and dbextent pages. As shown in the previous figures the total number dbspace pages is less in SQL/DS V3R5 compared with the results in SQL/DS V3R4. This is because in SQL/DS V3R5, the SQLCDBEX EXEC (VM) and ARIMEXBD Job (VSE) use the same algorithm as used in the database generation to determine what portion of the expanded directory should be used for the allocation bitmaps and what portion should be used for the page map table.

Being able to increase the number of dbextent pages in the directory, avoids the need to regenerate the database machine, when running short of physical space.

List of Abbreviations

AMODE	Addressing Mode
ANS	American National Standard
ANSI	American National Standard Institute
APAR	Authorized Program Analysis Report
APPC	Advanced Program to Program Communication
AR	Application Requester
AS	Application Server
CCSID	Coded Character Set IDentifier
CICS	Customer Information Control System
CDRA	Character Data Representation Architecture
CMS	Conversational Monitor System
CP	Control Program
CSL	Callable Services Library
DA	Data Administrator
DASD	Direct Access Storage Device
DBA	Database Administrator
DBCS	Double-Byte Character Set
DBMS	Database Management System
DBSU	Database Services Utility
DB2	IBM DATABASE 2
DCSS	Discontiguous Saved Segment
DDCS	Distributed Database Connection Services
DDL	Data Definition Language
DML	Data Manipulation Language
DRDA	Distributed Relational Database Architecture
ESA	Enterprise Systems Architecture
FIPS	Federal Information Processing Standard
HCAT	Hash Chain Anchor Table
IBM	International Business Machines Corporation
ICCF	Interactive Computing and Control Facility
ISO	International Standards Organization
ISQL	Interactive SQL
ITSO	International Technical Support Organization
IUCV	Inter-User Communication Vehicle
LI	Language Interface
LUW	Logical Unit of Work
MUM	Multiple User Mode
MVS	Multiple Virtual Storage
MVS/ESA	Multiple Virtual Storage/Enterprise Systems Architecture
NLS	National Language Support
PROFS	Professional Office System
PTF	Program Temporary Fix
QMF	Query Management System
RAS	Reliability / Availability / Serviceability
RDBMS	Relational Database Management System
RDS	Relational Data System
REXX	Restructured Extended Executive Language
RI	Referential Integrity
RM	Resource Manager
RMI	Resource Manager Interface
RMID	Resource Manager ID
RMODE	Residency Mode
RPG	Report Program Generator
RUOW	Remote Unit of Work
SAA	Systems Application Architecture
SBCS	Single-Byte Character Set
SQL	Structured Query Language
SQL/DS	Structured Query Language/Data System
SQLCA	SQL Communication Area
SUM	Single User Mode
TID	Tuple IDentifier
TSAF	Transparent Services Access Facility
VM	Virtual Machine

VM/ESA	Virtual Machine / Enterprise Systems Architecture
VSE	Virtual Storage Extended
VSE/ESA	Virtual Storage Extended / Enterprise Systems Architecture
VTAM	Virtual Telecommunication Access Method
XA	Extended Architecture
XC	Extended Configuration

Index

Numerics

A

B

C

D

E

F

H

I

L

M

N

O

P

R

S

T

U

V

W

X

Y

ITSO Technical Bulletin Evaluation RED000

International Technical Support Organization
SQL/DS Version 3 Release 5
Usage Guide
December 1995

Publication No. SG24-4647-00

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

- Mail it to the address on the back (postage paid in U.S. only)
- Give it to an IBM marketing representative for mailing
- Fax it to: Your International Access Code + 1 914 432 8246
- Send a note to REDBOOK@VNET.IBM.COM

Please rate on a scale of 1 to 5 the subjects below.
(1 = very good, 2 = good, 3 = average, 4 = poor, 5 = very poor)

Overall Satisfaction	____		
Organization of the book	____	Grammar/punctuation/spelling	____
Accuracy of the information	____	Ease of reading and understanding	____
Relevance of the information	____	Ease of finding information	____
Completeness of the information	____	Level of technical detail	____
Value of illustrations	____	Print quality	____

Please answer the following questions:

a) If you are an employee of IBM or its subsidiaries:

Do you provide billable services for 20% or more of your time? Yes____ No____

Are you in a Services Organization? Yes____ No____

b) Are you working in the USA? Yes____ No____

c) Was the Bulletin published in time for your needs? Yes____ No____

d) Did this Bulletin meet your needs? Yes____ No____

If no, please explain:

_____

_____

What other topics would you like to see in this Bulletin?

_____

_____

What other Technical Bulletins would you like to see published?

_____

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

Name	Address
Company or Organization	
Phone No.	

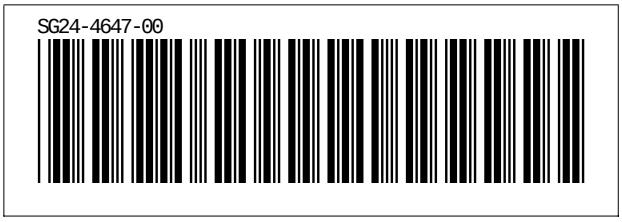
Fold and Tape **Please do not staple** Fold and Tape

PLACE
POSTAGE
STAMP
HERE

IBM International Technical Support Organization
Department 3222, Building 71032-02
POSTFACH 1380
71032 BOEBLINGEN
GERMANY

Fold and Tape **Please do not staple** Fold and Tape

Printed in U.S.A.



Artwork Definitions

id	File	Page	References
ITSLOGO	4647SU	i	i

Table Definitions

id	File	Page	References
MBBDBSH	4647CH05	124	124
MBBDBSR	4647CH05	124	124, 124, 124, 124
A01T010	4647AX01	167	167

Figures

C02F185	4647CH02	35	24	
C02F200	4647CH02	36	25	36
C02F210	4647CH02	37	26	36
C02F220	4647CH02	38	27	37
C02F230	4647CH02	39	28	38, 39
C02F240	4647CH02	40	29	39
C02F250	4647CH02	41	30	40
C02F215	4647CH02	41	31	41
C02F290	4647CH02	52	32	
C02F300	4647CH02	53	33	
C02F310	4647CH02	59	34	
C02F910	4647CH02	67	35	
C02F920	4647CH02	68	36	67
C02F930	4647CH02	68	37	68
C02F940	4647CH02	69	38	68
C02F950	4647CH02	69	39	69
C02F960	4647CH02	70	40	69
C02F970	4647CH02	70	41	70
C02F980	4647CH02	71	42	71
C02F350	4647CH02	77	43	
C02F360	4647CH02	77	44	
C02F370	4647CH02	77	45	
C02F380	4647CH02	77	46	
C02F390	4647CH02	78	47	
C02F400	4647CH02	78	48	
C02F410	4647CH02	78	49	
C02F420	4647CH02	79	50	
C03F010	4647CH03	90	51	
C03F020	4647CH03	90	52	
C03F030	4647CH03	91	53	
C03F040	4647CH03	91	54	
C03F050	4647CH03	92	55	
C03F060	4647CH03	92	56	
C03F070	4647CH03	92	57	
C03F080	4647CH03	92	58	
C04F010	4647CH04	97	59	
C04F020	4647CH04			

		97	60	97
C04F030	4647CH04	102	61	
C04F040	4647CH04	103	62	102
C04F050	4647CH04	104	63	103
C04F060	4647CH04	104	64	
C04F070	4647CH04	105	65	105
C04F080	4647CH04	106	66	
C04F090	4647CH04	106	67	
C04F100	4647CH04	106	68	
C04F110	4647CH04	107	69	107
C04F120	4647CH04	107	70	107
C04F130	4647CH04	108	71	107
C05F010	4647CH05	117	72	117
C05F020	4647CH05	118	73	118
C05F030	4647CH05	119	74	119
C05F040	4647CH05	120	75	119
C05F050	4647CH05	121	76	119, 121
C05F060	4647CH05	122	77	122
C06F010	4647CH06	139	78	139
C07F010	4647CH07	143	79	144
C07F020	4647CH07	144	80	144
C07F030	4647CH07	146	81	146
A01F010	4647AX01	161	82	
A01F020	4647AX01	170	83	
A01F030	4647AX01	170	84	
A02F010	4647AX02	187	85	186
BIGPIC	4647AX02	194	86	193
A04F010	4647AX04	200	87	200
A04F020	4647AX04	201	88	200
A04F030	4647AX04	202	89	202
A04F040	4647AX04	203	90	203
A04F050	4647AX04	204	91	203

Headings

Index Entries

Tables

id	File	Page	References
C02T010	4647CH02	17	1
C02T015	4647CH02	21	2
C02T020	4647CH02	29	3
C02T030	4647CH02	31	4
C02T035	4647CH02	33	5
C02T040	4647CH02	35	6
C02T050	4647CH02	43	7
C02T060	4647CH02	45	8
C02T070	4647CH02	47	9
C03T010	4647CH03	89	10
			89
C03T030	4647CH03	89	11
			89
C03T020	4647CH03	89	12
C03T040	4647CH03	90	13
C03T050	4647CH03	90	14
C03T060	4647CH03	91	15
C03T070	4647CH03	92	16
C03T080	4647CH03	93	17
C03T090	4647CH03	94	18
C05T040	4647CH05	114	19
C05T010	4647CH05	124	20
C07T010	4647CH07	145	25
			145
C07T050	4647CH07	149	26
			148
C07T060	4647CH07	149	27
			149
C07T070	4647CH07	150	28
			150
A01T020	4647AX01	167	29

Processing Options

Runtime values:

Document fileid	SG244647 SCRIPT
Document type	USERDOC
Document style	IBMXAGD
Profile	EDFPRF30
Service Level	0029
SCRIPT/VS Release	4.0.0
Date	95.11.20
Time	10:07:23
Device	3820A
Number of Passes	4
Index	YES
SYSVAR D	YES
SYSVAR G	INLINE
SYSVAR S	OFFSET
SYSVAR V	ITSCEVAL

Formatting values used:

Annotation	NO
Cross reference listing	YES
Cross reference head prefix only	NO
Dialog	LABEL
Duplex	YES
DVCF conditions file	(none)
DVCF value 1	(none)
DVCF value 2	(none)
DVCF value 3	(none)
DVCF value 4	(none)
DVCF value 5	(none)
DVCF value 6	(none)
DVCF value 7	(none)
DVCF value 8	(none)
DVCF value 9	(none)
Explode	NO
Figure list on new page	YES
Figure/table number separation	YES
Folio-by-chapter	NO
Head 0 body text	Part
Head 1 body text	Chapter
Head 1 appendix text	Appendix
Hyphenation	NO
Justification	NO
Language	ENGL
Layout	OFF
Leader dots	YES
Master index	(none)
Partial TOC (maximum level)	4
Partial TOC (new page after)	INLINE
Print example id's	NO
Print cross reference page numbers	YES
Process value	(none)
Punctuation move characters	.,
Read cross-reference file	(none)
Running heading/footing rule	NONE
Show index entries	NO
Table of Contents (maximum level)	3
Table list on new page	YES
Title page (draft) alignment	RIGHT
Write cross-reference file	(none)

Imbed Trace

Page 0	4647SU
Page 0	4647VARS
Page 0	4647FM
Page i	4647EDNO
Page ii	4647ABST
Page xiii	4647SPEC
Page xiii	4647TMKS
Page xiv	4647PREF
Page xix	4647ACKS
Page xx	4647CH01
Page 10	4647CH02
Page 85	4647CH03
Page 94	4647CH04
Page 108	4647CH05
Page 127	4647CH06
Page 141	4647CH07
Page 158	4647AX01
Page 183	4647AX02
Page 195	4647AX03
Page 197	4647AX04
Page 214	4647ABRV
Page 221	4647EVAL
Page 221	RCFADDR
Page 221	ITSCADDR FILE
Page 223	RCFADDR
Page 223	ITSCADDR FILE