SG24-4791-00

DB2 for MVS/ESA Version 4
Data Sharing Implementation

December 1996

International Technical Support Organization
San Jose Center

SG24-4791-00

International Technical Support Organization

DB2 for MVS/ESA Version 4 Data Sharing Implementation

December 1996

Take Note!

Before using this information and the product it supports, be sure to read the general information in Appendix B, “Special Notices” on page 237.

First Edition (December 1996)

This edition applies to DB2 Version 4, Program Number 5695-DB2.

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

Contents

Figures ix

Tables xiii

Preface xv
How This Redbook Is Organized xv
The Team That Wrote This Redbook xvi
Comments Welcome xviii

Chapter 1. Introduction 1
1.1 What Is DB2 Data Sharing? 1
1.2 Advantages of Data Sharing 4
1.3 Why Choose Data Sharing? 5

Chapter 2. Workload Distribution in a Data Sharing Environment 7
2.1 Application Workload Distribution 7
2.2 Data Sharing Overhead 8
2.3 Reducing Locking Activity and Contention 9
2.4 Estimating Data Sharing Overhead for an Application 10
2.5 Workload Distribution Using CICS 11
2.5.1 CICS/DB2 Configurations before Data Sharing 11
2.5.2 Connecting CICS Regions to DB2 in a Data Sharing Environment 13
2.5.3 Static Workload Distribution 13
2.5.4 Dynamic Workload Distribution 16
2.5.5 Implementing Dynamic Workload Distribution 17
2.5.6 CICS Transaction Affinities 22
2.6 IMS and DB2 in a Data Sharing Environment 22
2.7 Designing Databases and Applications to Exploit Data Sharing 24
2.7.1 Data Partitioning for Concurrent Utility Operations 25
2.7.2 Data Partitioning for Distinct Key Range Processing 25
2.7.3 Using GBPCACHE ALL 25

Chapter 3. Planning for DB2 Data Sharing 27
3.1 Capacity Planning 27
3.1.1 Additional CPU Capacity to Run Your Workload 27
3.1.2 Processing Capacity for the Coupling Facility 30
3.1.3 Multiple Coupling Facilities 30
3.1.4 Storage Capacity for the Coupling Facility 30
3.1.5 An Alternative Way of Sizing Coupling Facility Structures 34
3.1.6 Capacity and Coupling Facility Channels 36
3.2 Hardware Planning 37

3.2.1 Processors in the Parallel Sysplex 37
3.2.2 Sysplex Timer 38
3.2.3 Coupling Facility 40
3.2.4 Coupling Facility Channels 42
3.2.5 ESCON Directors and ESCON Control Units 42
3.3 MVS Software Planning 43
3.3.1 MVS Software Prerequisites 44
3.3.2 I/O Definitions 45
3.3.3 MVS Couple Data Sets 45
3.4 Network Planning 46
3.5 Software Maintenance Considerations 46

Chapter 4. DB2 Planning 49
4.1 Naming Convention 49
4.1.1 Procedure to Change DB2 Naming Convention 50
4.2 Implementation Checklists 58

Chapter 5. Global Locking and Buffer Pool Management 63
5.1 Global Locking 63
5.1.1 P-Locks 64
5.1.2 L-Locks 65
5.1.3 Group Buffer Pool Dependency 66
5.2 Locking Components 67
5.2.1 Types of Contention 69
5.2.2 Managing Contention 72
5.2.3 Page Set Physical Lock Negotiation Scenario 76
5.2.4 Page Physical Lock Negotiation Scenario 76
5.2.5 Retained Locks 78
5.2.6 Lock Avoidance in a Data Sharing Environment 79
5.3 Group Buffer Pools 79
5.3.1 Test for Page Validity 80
5.3.2 Page Registration and Deregistration 80
5.3.3 Page Refresh from Group Buffer Pool 81
5.3.4 Force at Commit Processing 81
5.3.5 Cross-System Invalidation 81
5.3.6 Getpage in a Data Sharing Environment 82
5.3.7 Group Buffer Pool Checkpoint 83
5.3.8 Group Buffer Pool Castout 83

Chapter 6. Installing DB2 Data Sharing 85
6.1 Naming Conventions Used 87
6.2 Enabling an Existing DB2 V4 Non-Data-Sharing System 87
6.3 Add Members to the Data Sharing Group 92
6.4 Install a New Data Sharing Group 98

6.5 Installation Verification 104
6.5.1 Test Group Buffer Pool Caching 104
6.5.2 Test Global Lock Serialization 106
6.5.3 Test Concurrency 107
6.6 Removing Members from the Data Sharing Group 107
6.7 Disabling Data Sharing 109
6.7.1 Backup Strategy 112
6.7.2 Lost Log Analysis 112
6.8 Reenabling Data Sharing 113

Chapter 7. Merging Existing DB2 Data into One Data Sharing Group 117
7.1 How to Move the Data 118
7.2 Choosing a DB2 Subsystem As the Originating Member 119
7.3 Creating Data Objects on the Target DB2 120
7.4 Copying Data with DSN1COPY and OBID Translation 124
7.5 Moving Data With REPAIR Utility 125
7.6 Moving Indexes 129
7.7 DSN1COPY with Incremental Image Copies 130
7.8 Summary 130

Chapter 8. Commands and Operations 133
8.1 Operations from an MVS Console 133
8.1.1 Routing a DB2 Command 133
8.1.2 Command Scope 134
8.1.3 Receiving Messages 135
8.2 Submitting Work to Be Processed 136
8.2.1 Running CICS and IMS Applications 136
8.2.2 TSO, Batch, and CAF Applications and Utilities 137
8.3 XCF Commands 139
8.3.1 Activating and Deactivating a CFRM Policy 139
8.3.2 XCF Display Command with Various Options 139
8.3.3 Displaying Information about Structures 143
8.4 Displaying Information about Structure Use 146
8.4.1 Lock Structure and SCA Structure 146
8.4.2 Displaying Global Lock Information 147
8.4.3 Group Buffer Pool Structures 148
8.5 Displaying Group Buffer Pool Dependencies 150
8.6 Changing Coupling Facility Structures 150
8.6.1 Adding a Structure to a Coupling Facility 151
8.6.2 Moving a Structure from One Coupling Facility to Another 151
8.6.3 Changing Characteristics of Structures 152
8.7 Shutting Down the Coupling Facility 153
8.8 Starting a Sysplex Failure Management Policy 154
8.8.1 Using Automatic Restart Management 155

8.9 Messages . . . 156
8.9.1 Group Buffer Pool Exceptions . . . 156
8.9.2 Connectivity Exceptions . . . 157
8.9.3 Subsystem Failure . . . 157
8.9.4 CPC Failure . . . 157
8.9.5 Timer Failure . . . 158
8.9.6 Changes of Policy . . . 158

Chapter 9. Recovery in a Data Sharing Environment . . . 159
9.1 Data Sharing Recovery Environment . . . 159
9.1.1 Logging Considerations . . . 159
9.1.2 Log Record Sequence Number . . . 162
9.1.3 Changes to Support the Data Sharing Environment . . . 163
9.1.4 Running the RECOVER Utility . . . 167
9.2 Recovery from Sysplex Failures . . . 170
9.2.1 DB2 Subsystem Failure . . . 170
9.2.2 Coupling Facility Failure . . . 172
9.3 Deleting Structures and Connections . . . 177
9.3.1 Deleting Failed-Persistent Connectors . . . 178
9.3.2 Deleting Persistent Structures . . . 179
9.4 Disaster Backup and Recovery . . . 180
9.4.1 Disaster Recovery for a Non-Data-Sharing DB2 System . . . 180
9.4.2 Disaster Recovery Backup . . . 181
9.4.3 Disaster Recovery Flow . . . 183
9.4.4 Disaster Recovery Preparation . . . 188
9.4.5 Data Sharing and Non-Data-Sharing Disaster Recovery . . . 189

Chapter 10. Performance and Tuning Considerations . . . 191
10.1 Factors Influencing Data Sharing Performance . . . 191
10.2 Performance Objectives . . . 192
10.3 Monitoring Tools . . . 193
10.4 Verifying the Performance of a Parallel Sysplex . . . 193
10.5 Tuning DB2 Locking . . . 195
10.5.1 Global Lock Contention . . . 198
10.6 Tuning Group Buffer Pools . . . 199
10.6.1 Tune the GBP Structure Size . . . 202
10.6.2 Tune the Ratio between Directory Entry and Data Pages in the GBP . . . 203
10.6.3 Tune the GBP Checkpoint Frequency and Castout Thresholds . . . 203
10.7 Transaction Run-Time Criteria: Data Sharing and Non-Data-Sharing . . . 204

Chapter 11. Distributed Processing in a Data Sharing Environment . . . 207
11.1 Overview of Changes . . . 207
11.2 Connectivity Alternatives . . . 207
11.3 Communication Database Configuration . . . 210

11.3.1 SYSIBM.SYSLUNAMES . . . 211
11.3.2 Updating CDB Values . . . 214
11.3.3 Resetting Generic LU Information . . . 214
11.3.4 Using the Generic LU Name . . . 217
11.4 Sample Cases . . . 218
11.4.1 Case 1 . . . 219
11.4.2 Case 2 . . . 222
11.4.3 Case 3 . . . 223
11.4.4 Case 4 . . . 225
11.4.5 Case 5 . . . 226
11.4.6 Case 6 . . . 227
11.4.7 Case 7 . . . 228
11.5 Replacing DDF Connections by Data Sharing . . . 230

Appendix A. Sample CFRM Policy . . . 233

Appendix B. Special Notices . . . 237

Appendix C. Related Publications . . . 239
C.1 International Technical Support Organization Publications . . . 239
C.2 Redbooks on CD-ROMs . . . 239
C.3 Other Publications . . . 239

How To Get ITSO Redbooks . . . 241
How IBM Employees Can Get ITSO Redbooks . . . 241
How Customers Can Get ITSO Redbooks . . . 242
IBM Redbook Order Form . . . 243

Index . . . 245

Figures

1. DB2 Data Sharing Group Structure . . . 2
2. CICS/DB2: Configuration 1—Before Data Sharing . . . 12
3. CICS/DB2: Configuration 2—Before Data Sharing . . . 13
4. Static Workload Distribution: Reconfiguring Configuration 2 for a Data Sharing Environment . . . 15
5. Dynamic Workload Balancing: Reconfiguring Configuration 1 for a Data Sharing Environment . . . 19
6. Dynamic Workload Balancing: Reconfiguring Configuration 2 for a Data Sharing Environment . . . 21
7. Static Workload Distribution with IMS Version 5 . . . 23
8. Dynamic Workload Distribution with IMS Version 6 . . . 24
9. Projected Capacity Loss with Degree of Data Sharing 50% and 100% . . . 29
10. DISPLAY GROUP . . . 35
11. DISPLAY GROUPBUFFERPOOL . . . 36
12. Locking Components in a Data Sharing Environment . . . 68
13. Lock Table . . . 70
14. Lock Information Stored by XES . . . 71
15. What Happens When a Lock Request Is Sent to XES . . . 73
16. Contention Resolution in a Data Sharing Environment . . . 74
17. Buffer Manager Getpage with Group Buffer Pool . . . 82
18. Type 2 Index Query . . . 88
19. Install Enable: Panel DSNTIPA1 . . . 88
20. Install Enable: Panel DSNTPK . . . 89
21. Install Enable: Panel DSNTIPR . . . 90
22. Parameters of DSNZDBC1 to Enable Data Sharing . . . 91
23. Change Log Inventory to Enable Data Sharing . . . 92
24. Output of DBC1MSTR . . . 92
25. Output of DBC1IRLM . . . 92
26. Install Member: Panel DSNTIPA1 . . . 93
27. Install Member: Panel DSNTIPK . . . 94
28. Install Member: Panel DSNTIPR . . . 95
29. Contents of Originating Member BSDS . . . 97
30. Install Group: Panel DSNTIPA1 . . . 98
31. Install Group: Panel DSNTIPK . . . 99
32. Install Group: Panel DSNTIPR . . . 100
33. Contents of BSDS before the First Start of the New DB2 Subsystem . . . 102
34. Contents of BSDS after the First Start of the New DB2 Subsystem . . . 103
35. DISPLAY DATABASE Command Showing Inter-DB2 Read-Write Interest . . . 105
36. Detailed Statistics Section of DISPLAY GBPOOL Command . . . 106
37. Contents of the BSDS for the Surviving Member . . . 111
38. Contents of the BSDS after Data Sharing Has Been Reenabled . . . 115

39. BSDS Listing: Checking the Highest RBA Written 119
40. Sample Query: Catalog Information about Table Spaces 121
41. Sample Query: Catalog Information about Indexes 121
42. Sample Query: Catalog Information about Tables 122
43. Sample CREATE Statements 123
44. Sample DSN1COPY Utility 124
45. DSN1PRNT of Page Set Header Page of a Nonpartitioned Table Space 126
46. REPAIR Utility 126
47. SPUFI Output of SYSIBM.SYSTABLESPACE 128
48. DSN1PRNT of Page Set Header Page of a Partitioned Table Space 129
49. REPAIR Utility 129
50. Identification of Messages in a Sysplex 136
51. Display of MVS Images in the Parallel Sysplex 140
52. Display of a Coupling Facility 141
53. Display of the Active Policies 142
54. Display of CFRM Couple Data Sets 143
55. Display a List of the Defined Structures 143
56. Display of Coupling Facility Information 144
57. Display of Structure Information 145
58. DB2 DISPLAY GROUP Command Showing Subsystems Running 146
59. Display of Page Set P-Locks and L-Locks 147
60. Output of DB2 DISPLAY GROUPBUFFERPOOL Command 149
61. Output of DB2 DISPLAY BUFFERPOOL Command 150
62. Display Coupling Facility Structures 151
63. Rebuild a Structure 152
64. Message IXC602I 155
65. REPORT Utility Output: Log Record Sequence Number 163
66. REPORT Utility Output: Recover to Current Time 167
67. REPORT Utility Output: Recover to a Point-in-Time 169
68. Display of Retained Locks 171
69. MODIFY IRLMPROC 171
70. Message DSNB250E 174
71. Message DSNB32xI 175
72. Display of GRECP and LPL 176
73. Deleting Failed-Persistent Connectors 179
74. Deleting Persistent Structures 179
75. Message IXC360I: Structure in Transition 180
76. Coupling Facility Structure Activity Report 194
77. Data Sharing Locking Block of DB2 PM Statistics Report 196
78. -DISPLAY GROUPBUFFERPOOL Output Status Information 200
79. -DISPLAY GROUPBUFFERPOOL Output Group Detail Dynamic Information 201
80. -DISPLAY GROUPBUFFERPOOL Output Member Detail Dynamic Information . . . 202
81. Locking Block and Data Sharing Block of DB2 PM Accounting Report 205
82. BPx Block and Group BPx Block of DB2 PM Accounting Report 205

83. Class 3 Suspension Time Block of DB2 PM Accounting Report 206
84. Workstation Application Requester Configuration 221
85. Suggested Workstation Application Requester Configuration 222
86. DDF Access to DB2 Data on Another MVS Image 230
87. DDF Access Replaced by Enabling Two-Way Data Sharing 231

Tables

1. Recommendations for DB2 SCA Structure Size 33
2. Initial Structure Size Estimates 35
3. MVS Software Requirements 44
4. Activities before Implementation of One-Way Data Sharing 58
5. Activities during Implementation of Data Sharing 62
6. Activities after Implementation of One-Way Data Sharing 62
7. Determining Group Buffer Pool Dependency 67
8. Installation Options for Data Sharing 86
9. Sample Naming Convention for a Data Sharing Group 87
10. Results of Sample Queries: Source DB2 122
11. Results of Sample Queries: Target DB2 123
12. Number of High-Order Bits Used to Specify the Partition Number 127
13. Synchronous Service Times to Access DB2 Structures 195
14. Connectivity Options: Data Sharing Group As Application Server 209
15. Test Cases 210
16. Using the RESET GENERICLU Command 217
17. SYSIBM.SYSLUNAMES Application Requester: Case 1 219
18. SYSIBM.SYSLOCATIONS Application Requester: Case 1 220
19. SYSIBM.SYSLUNAMES Application Server: Case 1 220
20. SYSIBM.SYSLOCATIONS Application Server: Case 1 220
21. SYSIBM.SYSLUNAMES Application Requester: Case 2 222
22. SYSIBM.SYSLOCATIONS Application Requester: Case 2 222
23. SYSIBM.SYSLUNAMES Application Server: Case 2 223
24. SYSIBM.SYSLOCATIONS Application Server: Case 2 223
25. SYSIBM.SYSLUNAMES Application Requester: Case 3 224
26. SYSIBM.SYSLOCATIONS Application Requester: Case 3 224
27. SYSIBM.SYSLULIST Application Requester: Case 3 224
28. SYSIBM.SYSLUNAMES Application Server: Case 3 224
29. SYSIBM.SYSLOCATIONS Application Server: Case 3 224
30. SYSIBM.SYSLUNAMES Application Requester: Case 4 225
31. SYSIBM.SYSLOCATIONS Application Requester: Case 4 225
32. SYSIBM.SYSLUNAMES Application Server: Case 4 225
33. SYSIBM.SYSLOCATIONS Application Server: Case 4 225
34. SYSIBM.SYSLUNAMES Application Requester: Case 5 226
35. SYSIBM.SYSLOCATIONS Application Requester: Case 5 226
36. SYSIBM.SYSLUNAMES Application Server: Case 5 226
37. SYSIBM.SYSLOCATIONS Application Server: Case 5 227
38. SYSIBM.SYSLUNAMES Application Requester: Case 6 227
39. SYSIBM.SYSLOCATIONS Application Requester: Case 6 227
40. SYSIBM.SYSLUNAMES Application Server: Case 6 228
41. SYSIBM.SYSLOCATIONS Application Server: case 6 228

42. SYSIBM.SYSLUNAMES Application Requester: Case 7 228
43. SYSIBM.SYSLOCATIONS Application Requester: Case 7 229
44. SYSIBM.SYSLULIST Application Requester: Case 7 229
45. SYSIBM.SYSLUNAMES Application Server: Case 7 229
46. SYSIBM.SYSLOCATIONS Application Server: Case 7 229

Preface

This redbook is unique in its detailed coverage of DB2 data sharing, with a particular focus on implementation aspects. It provides information about planning for DB2 data sharing, global locking and buffer pool management, considerations in installing data sharing and merging existing DB2 data into a data sharing group, commands and operational procedures that should be used in a data sharing environment, aspects of recovery, performance, and tuning, and distributed processing in a data sharing environment.

This redbook provides practical "how to" tips and serves as a complementary document to IBM DATABASE 2 for MVS/ESA Version 4 Data Sharing: Planning and Administration

This redbook is written for database administrators, system programmers, and anyone involved in DB2 data sharing implementation.

Several practical examples are presented to demonstrate procedures to be followed in implementing DB2 data sharing. Scenarios to successfully recover from failures are discussed. Some knowledge of DB2 data sharing concepts is assumed.

How This Redbook Is Organized

This redbook contains 250 pages. It is organized as follows:

- Chapter 1, "Introduction"

  Provides introduction to DB2 data sharing, lists advantages in this environment, and discusses why you may choose data sharing.

- Chapter 2, "Workload Distribution in a Data Sharing Environment"

  Discusses workload distribution in the DB2 data sharing environment.

- Chapter 3, "Planning for DB2 Data Sharing"

  Discusses capacity planning for DB2 data sharing, hardware planning, MVS software planning, network planning, and software maintenance considerations.

- Chapter 4, "DB2 Planning"

  Explains the importance of establishing proper naming conventions in the DB2 data sharing environment.

- Chapter 5, "Global Locking and Buffer Pool Management"

  Describes how locking and buffering are managed in the DB2 data sharing environment.

- Chapter 6, "Installing DB2 Data Sharing"

Discusses considerations for creating a data sharing group, and provides implementation guidelines to supplement the information contained in IBM DATABASE 2 for MVS/ESA Version 4 Data Sharing: Planning and Administration

- Chapter 7, “Merging Existing DB2 Data into One Data Sharing Group”

  Discusses reasons for merging data and outlines procedures to successfully merge existing data from multiple DB2 subsystems into a data sharing group.

- Chapter 8, “Commands and Operations”

  Describes new aspects of operations in a Parallel Sysplex, discusses commands that enable you to display and control coupling facilities as well as to display and monitor DB2 structures, and explains new messages related to DB2 data sharing.

- Chapter 9, “Recovery in a Data Sharing Environment”

  Describes what changes have been made to DB2 to support application recovery in a data sharing environment. From a systems perspective, logging considerations and different detailed recovery scenarios in a Parallel Sysplex environment are discussed. Disaster recovery considerations in a data sharing environment and steps that should be followed to carry out disaster recovery are also documented.

- Chapter 10, “Performance and Tuning Considerations”

  Offers some guidance as to what the main performance factors are and how you can control them.

- Chapter 11, “Distributed Processing in a Data Sharing Environment”

  Describes distributed processing in a data sharing environment. Member-specific and Group-generic methods that can be used by application requesters to connect to a data sharing group are described.

- Appendix A, “Sample CFRM Policy”

  Shows an example of CFRM policy.

The Team That Wrote This Redbook

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

Ravi Kumar is a Senior ITSO Specialist for DB2 for System 390 and MVS/ESA at the Application Development and Data Management ITSO Center, San Jose. He writes extensively and teaches IBM classes worldwide on all areas of DB2 for System 390 and MVS/ESA. Before joining the ITSO two years ago, Ravi worked in IBM Australia as a DB2 specialist.

Silvio Podcameni is a Senior ITSO Specialist for DRDA and DB2 for System 390 and MVS/ESA at the Application Development and Data Management ITSO Center, San Jose.

He conducts projects related to DRDA and DB2 for System 390 and MVS/ESA, delivers workshops on connectivity for workstations, and contributes to redbooks on DRDA security, stored procedures, and connectivity. Silvio comes from IBM Brazil, where he has worked for 25 years.

Guy Delaporte is a specialist in DB2 for MVS/ESA performance, tuning, and database design. He has been involved in customers′ big projects with the Systems Integration French organization since 1987 to 1994. Guy then joined the Field Support Center to provide support to DB2 V4 Data Sharing customers.

Peter Giesecke was born in Stuttgart, Germany and studied Mathematics and Physics at the University of Heidelberg. He has worked for over 17 years on the administration and system programming of MVS database systems for Deutsche Bank AG. He started with DB2 Version 1.3 and still loves it.

Mark Gladstone currently works as a DB2 Technical Specialist for Barclays Bank in the UK, where he is part of the team implementing DB2 data sharing. Over the past 12 years Mark has worked as an MVS, CICS, IMS and DB2 systems programmer.

Karen Lloyd works at Standard Bank of South Africa, Ltd as a DB2 Technical Consultant. She has 11 years of IT experience, 8 of which are in the DB2 arena. Karen was the Technical Project Manager for the DB2 Data Sharing Implementation Project at Standard Bank of South Africa.

Michael Parbs is a Senior Database Administrator, currently working for the Health Insurance Commission in Australia. He has over 15 years experience in data processing, the last 7 years working closely with DB2. His experience includes data modeling, database design, and DB2 installation, performance, and tuning. Michael has also presented at conferences on DB2.

Piero Proietti has been working at IBM since 1987 as a DB2 and IMS specialist. Currently, Piero is working in the IBM Availability Center of Rome supporting customers in the implementation of DB2 and in identifying and resolving implementation defects as well as recovering from failures.

Maryela Weihrauch is a systems consultant at IBM Germany. She has 5 years of experience in DB/DC environment. In the last 2 years, she has been responsible for introducing DB2 data sharing in different customer projects.

Barbara Zastrozynski is a DB2 system programmer at Victoria Vers. AG with 5 years experience in a large data center. In the last year Barbara worked mainly in the field of DB2 data sharing.

Special thanks are due to the following people for the invaluable advice and guidance they provided in the production of this document:

- Roy Cornford IBM United Kingdom
- Jeff Josten IBM Development, Santa Teresa
- Bryce Krohn IBM Education System Center, San Jose
- Roger Miller IBM Development, Santa Teresa
- Chris Mills IBM Development, Santa Teresa
- Nick Nomm IBM Development, Santa Teresa
- Fred Orosco IBM Development, Santa Teresa
- Jim Teng IBM Development, Santa Teresa
- Sam Wang IBM Development, Santa Teresa
- Kathy Zeidenstein IBM Development, Santa Teresa

Thanks also to Maggie Cutler for her editorial assistance.

Comments Welcome

We want our redbooks to be as helpful as possible. Should you have any comments about this or other redbooks, please send us a note at the following address:

redbook@vnet.ibm.com

Your comments are important to us!

Chapter 1. Introduction

IBM DATABASE 2 for MVS/ESA Version 4 (DB2 V4) introduces many enhancements in the areas of client/server, processing, availability, performance, and user productivity. Data sharing, a major feature of DB2 V4, facilitates a number of these enhancements.

Preparing for DB2 data sharing is complex. Database administrators, systems programmers, and personnel in operations, communications, and capacity management must understand the concepts of data sharing as implemented in DB2 V4. They have to communicate with one another to ensure that DB2 data sharing is properly implemented and managed.

As DB2 data sharing is introduced in your environment, you must review many of the procedures you currently have for managing DB2. The many operational, recovery, and disaster recovery issues that data sharing introduces may force you to change the way you currently manage DB2.

This book is written to help you plan for and implement data sharing in your environment. It also introduces some of the operational and recovery activities you should include in your procedures to successfully manage a data sharing group.

The book is written to supplement the planning considerations and installation procedures that are documented in *IBM DATABASE 2 for MVS/ESA Version 4 Data Sharing: Planning and Administration* and *IBM DATABASE 2 for MVS/ESA Version 4 Installation Guide*.

1.1 What Is DB2 Data Sharing?

Before DB2 V4, only the distributed function of DB2 allowed for read and write access among multiple DB2 subsystems. Data sharing implemented in DB2 V4 now allows for read and write access to DB2 data from different DB2 subsystems residing on multiple central processor complexes (CPCs) in a Parallel Sysplex.

A *Sysplex* is simply a number of MVS systems communicating and cooperating with one another through specialized hardware and software components. These MVS systems are connected and synchronized through a Sysplex Timer and enterprise systems connection (ESCON) channels.

A *Parallel Sysplex* is a Sysplex that is also using one or more coupling facilities. A coupling facility is a hardware and software solution that provides high-speed caching, list processing, and lock processing services for any MVS applications on the Sysplex.

A DB2 data sharing *group* is a collection of one or more DB2 subsystems accessing shared DB2 data. Each DB2 subsystem in a data sharing group is referred to as a *member* of that group. All members of the group use the same shared catalog and directory and must reside in the same MVS Parallel Sysplex.

Figure 1 shows the major components of a DB2 data sharing group:

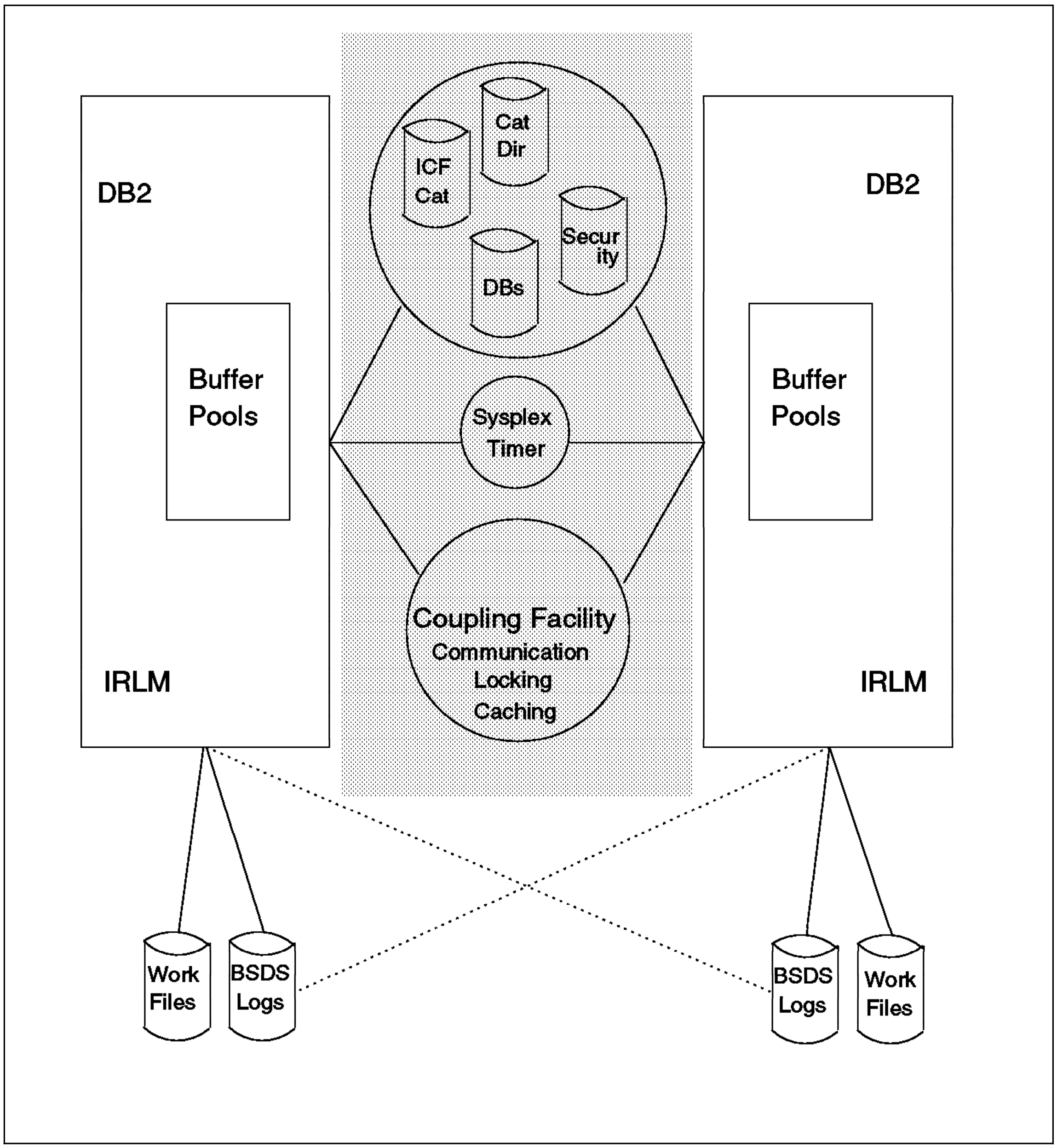


Figure 1. DB2 Data Sharing Group Structure

The two DB2 subsystems in Figure 1 are members of the same DB2 data sharing group. They can be running on the same MVS system or different MVS systems in the same Parallel Sysplex.

All members of the DB2 data sharing group can read and update the same tables at the same time. Therefore all data that different members of the data sharing group may access must reside on *shared DASD*.

Members of the DB2 data sharing group use the following coupling facility services to communicate and move data between the individual members:

- The *coupling facility list structure* contains the DB2 *shared communications area (SCA)*.

  Data sharing member names from DSNZPARM are used to CONNECT members to the coupling facility at DB2 startup. The first connector causes the structure to be allocated in a coupling facility based on the preference list in the active Coupling Facility Resource Management (CFRM) policy.

  Each DB2 member uses the SCA to pass control information throughout the data sharing group. The SCA contains all database exception status conditions and other information necessary for recovery of the data sharing group.

- The *coupling facility lock structure* is used to serialize on resources such as table spaces and pages. It protects shared DB2 resources and allows concurrency.

  The system lock manager (SLM), a name used informally to refer to a component of MVS Cross System Extended Services (XES), presents global lock information to the lock structure on behalf of each member′s Integrated Resource Lock Manager (IRLM).

  The lock structure consists of two parts; a coupling facility lock list table, commonly called the *modified resource* list, and a coupling facility lock hash table, commonly called the *lock table*. The modified resource list is used to record locks that protect changed data, thereby protecting the data in case of failure. The lock table contains the lock status information and the owning members of those locks. It is used to provide global lock serialization.

- One or more *coupling facility cache structures*, which are the *group buffer pools* (GBPs). They cache shared data pages.

  Data sharing, as implemented in DB2 V4, synchronously forces changed pages belonging to GBP-dependent page sets or partitions to be written to the GBPs before they are written to DASD. A write to the coupling facility is at electronic speed, so the I/O time is not a concern. However, the I/O itself is a concern because eventually these pages have to be cast out to their permanent home on DASD.

  Shared data pages are registered in a GBP directory in each cache structure, thus enabling the Coupling Facility Control Program in the coupling facility to cross-system invalidate the copies of data pages that are held in individual DB2 member buffer pools. The XI signals are processed entirely within the coupling facility channels without causing processor interrupts. Each DB2 member must then reread the pages from either the GBP or DASD if it needs to reference the pages again.

1.2 Advantages of Data Sharing

Data sharing offers a number of advantages that are well documented. Nevertheless, it is worth reviewing some of the major advantages:

- Capacity

  Additional capacity and scalability are the reasons why most of the customers have moved toward data sharing.

- Improves availability

  Data sharing helps to improve the availability of your data by providing access to the data during both planned and unplanned outages.

- Flexibility in growth

  Data sharing improves price performance by allowing you to run complex business applications on one or more smaller, less expensive S/390 microprocessors. As these applications grow, you can easily install a new DB2 subsystem into an existing data sharing group. The new subsystem can handle the increased workload by running on a different MVS in the Parallel Sysplex.

- Work flow management

  Instances of DB2 subsystems can be regularly started and stopped in the data sharing group. They can run on different MVS images in the Parallel Sysplex to handle variations in workload.

  Workload balancing techniques can also be used to dynamically distribute your current workload across multiple MVS systems.

- An alternative to distributed data

  Data sharing also offers better performance, with no VTAM overhead, and provides higher availability than a distributed data solution.

- Application interface is unchanged

  Existing SQL interfaces and attachments remain unchanged with data sharing. Your applications can therefore easily run in a data sharing environment with no change to application code in most cases.

  It is possible to bind a package or plan on one member and then run that package or plan on any other member in that data sharing group.

1.3 Why Choose Data Sharing?

For many installations, the move to data sharing is an important decision. The cost of hardware and software needed to support a Parallel Sysplex and data sharing is an important factor in making this decision.

In addition to the points noted in 1.2, "Advantages of Data Sharing" on page 4, here are some ways you may choose to use data sharing in your environment:

- Flexibility in scheduling existing workloads

  A CICS application that uses DB2 data, for example, is currently restricted to running on one MVS. Data sharing enables you to clone the CICS region on another MVS. The application can then run concurrently on both systems, increasing its overall throughput.

- Reduce the need for distributed processing

  When a number of applications run on different DB2 subsystems and have to communicate with one another through Distributed Relational Database Architecture (DRDA), an important consideration to be aware of is the DRDA overhead. These applications can now be merged into one data sharing group and still run on different DB2 subsystems. They can then be recoded to communicate with each other directly without using DRDA.

Make sure you decide to implement DB2 data sharing for the right reasons. You must understand how you plan to use DB2 data sharing and what you hope to gain from implementing it before you commit to any DB2 data sharing implementation plans.

Your organization may not be quite ready for DB2 data sharing. You may have a lot of work to do to set up the hardware and software environment where DB2 data sharing can be installed. In these cases, you must be committed to understanding how to manage a Parallel Sysplex environment and you must become familiar with other DB2 V4 features before using DB2 data sharing.

It is possible for both the affinity and nonaffinity workloads to share the same data sharing group. From an application′s point of view, however, some applications derive more benefit from data sharing than others. For example, if you have an application with CICS region affinities, that application can only be scheduled in the CICS region that is connected to that DB2 system. In this case it is not easy to reap the benefits from dynamic workload balancing. Similarly, applications with a high distributed workload can be collapsed into a data sharing group. However, if the location names are hard coded in the application, you may have to recode the application to remove the distributed code, to reap the performance benefits of running the same application in a nondistributed way.

In addition, moving an application into a data sharing environment where the location name changes implies that you must be committed to changing the application "up front" if the location names are hard coded in the application. If you are designing an application that is distributed now but may at some time in the future run in a data sharing environment, AVOID HARD CODING LOCATION NAMES in your application. You may, for example, want to implement a subroutine that looks at a location details table and dynamically connects the application to the right location. Then changing DB2 locations where applications are to connect is merely a case of changing "rows in tables" and rebinding the application on the new location.

You must review applications that have components that do not issue commits frequently and be committed to changing those applications before you can realize all of the benefits of DB2 data sharing.

To summarize, you can move ANY application into a data sharing environment, but some applications may reap more benefits from data sharing than others. To maximize the benefits of running your application in a data sharing environment, you may in some cases have to change the application. In short, planning and preparing for DB2 data sharing are major tasks, and they should not be rushed.

Chapter 2. Workload Distribution in a Data Sharing Environment

To exploit such data sharing benefits as improved availability and better processor usage, you have to distribute your DB2 workload across more than one member in a data sharing group. In this chapter we discuss different ways of distributing your application workload. We focus on the amount of data sharing overhead the different distribution methods place on an application′s workload and explain what you can do from an application perspective to reduce the overhead.

In a data sharing environment your main objective should be to maximize the benefits you get from data sharing while minimizing the cost (overhead) to your applications. Therefore in this chapter we also document some design considerations for applications running in a data sharing environment to ensure that once data sharing has been implemented at your site, your applications can fully exploit its capabilities.

2.1 Application Workload Distribution

Planning how you are going to distribute your application workload in a data sharing environment is an important task that you should do up front, in the planning phase of your data sharing implementation project. Your application workload distribution will affect a number of the activities that have to take place during the implementation of data sharing. For example, the amount of workload and how it is distributed in your data sharing environment will determine the size of the structures on the coupling facility. The structure sizes will in turn affect certain aspects of the hardware configuration.

Here are some questions you might consider:

- What are your site′s primary objectives for implementing data sharing and how are you going to distribute your workload to meet these objectives?
- Will all of your applications reside on a data sharing platform, or will some of them reside on non-data-sharing subsystems?

Implementing data sharing provides you with the flexibility to move pieces of DB2 application workload around the processors in the Sysplex, as now all of these processors have efficient, direct access to the same DB2 data.

Once data sharing is implemented, you can start distributing your DB2 workload across the members in the data sharing group. There are a number of ways of distributing workload. Initially you could distribute workload using relatively simple methods, such as:

- Moving CICS regions between processors
- Running batch jobs on different processors
- Running utilities on different processors

These distribution methods are quite easy to implement, and using them has the advantage of enabling you to test out your data sharing setup on a limited basis before starting to process larger volumes of shared workload. Moving workload around in these ways is an easy way of using the new CMOS processor in your Sysplex, thereby taking workload off processors that might have capacity constraints.

Implementing CICSPlex SM to dynamically distribute your workload across the Sysplex is another way of achieving workload distribution. CICSPlex SM implementation is a complex task that can take a while to complete, depending on the configuration of your existing CICS environment. For applications that have high availability requirements, however, you have to implement the kind of dynamic workload distribution you can achieve with CICSPlex SM. Refer to 2.5, "Workload Distribution Using CICS" on page 11 for further details on workload distribution using CICS and CICSPlex SM.

IMS V6 will also have dynamic workload distribution capabilities.

2.2 Data Sharing Overhead

Usually we talk about *coupling efficiency* (CE) and *coupling overhead* (CO), both of which take into account the fixed MVS multisystem management overhead as well as the variable DB2 data sharing overhead:

```
Nway ITR = N * 1Way ITR * CE  (ITR = Internal Throughput Rate)
CO = 100 - CE

CO for data sharing benchmarks (e.g., IBM Relational Warehouse Workload)
is approximately 17% + .5% * (N-2)

CO for production workloads is typically 10%.
```

How you distribute your workload will determine the amount of overhead DB2 data sharing places on your DB2 applications. Overhead is incurred in a data sharing environment when you start actively sharing data, that is, concurrently accessing page sets from multiple DB2 members in the data sharing group (where at least one of the members has write interest). When lock contention occurs between the members in the group, the overhead incurred increases further. (When there is no active sharing of data by members in the data sharing group, the amount overhead will be small.)

Data sharing overhead is determined by the number of accesses to the structures on the coupling facility and the amount of communication that occurs directly between members when contention has to be resolved. Data sharing overhead therefore consists of:

- Number of messages and/or requests passed between members

  Most of the messaging is due to global contention for locks. All the other DB2 data sharing messaging is usually insignificant.

- Number of lock requests propagated to the coupling facility lock structure
- Number of GBP requests
- Number of SCA requests

The number of GBP requests is affected mainly by how you distribute your workload—in other words, whether you choose to move your workload among the members in the group in such a way that there is concurrent access to data through more than one member. (Refer to 5.3, “Group Buffer Pools” on page 79 for details.) SCA requests are mostly made when the status of an object changes, for example, a table space status goes from RW to UT,UT. The number of shared communication area requests is insignificant when compared to GBP and lock requests.

The area that has the most scope for minimizing overhead, from an application perspective, is locking. A lot can be done at the application level to reduce both the number of lock requests propagated to the coupling facility lock structure and the amount of lock contention, as contention increases the amount of communication required between the members in the group.

2.3 Reducing Locking Activity and Contention

In this section we discuss how you can reduce locking and contention in your environment to ensure that the data sharing overhead incurred by an application is kept to a minimum.

When reading this section, remember that our recommendations are of particular importance where an application workload has been distributed in such a way that it causes high inter-DB2 read-write interest to occur. Even in a non-data-sharing environment, however, the recommendations will improve overall application performance.

Preferably, you should implement the recommendations before implementing data sharing, to facilitate a smooth transition from a non-data-sharing to a data sharing environment.

Reduce Locking Activity: Most of the recommendations listed below for reducing the number and duration of locks have been around for a long time and predate data sharing. However, the importance of implementing them increases in a data sharing environment. The recommendations are:

- Use Type 2 indexes for data-only locking.
- Consider the use of the ISOLATION(UR) bind parameter.
- Use ISOLATION(CS) wherever possible for lock avoidance.
- For maximum lock avoidance, use CURRENTDATA(NO) in conjunction with ISOLATION(CS).

- If you have applications that use ISOLATION(RR), investigate ISOLATION(RS), that is, read stability isolation. RS is similar to RR but slightly less restrictive, as it allows insert activity to take place.
- Reduce the duration of locks by issuing frequent commits.
- Bind plans and packages with RELEASE(DEALLOCATE) to ensure that table space L-locks are not continually released and reacquired and that global table space L-locks are held across transactions in the case of thread reuse (as in the case of CICS protected threads, for example).
- Carefully evaluate the LOCKSIZE parameter to ensure that the biggest lock size possible is used without causing contention.

Reduce the Amount of Contention: Most of the recommendations listed below for reducing the amount of contention have also been around for a long time:

- Increase PCTFREE, after giving due consideration to the use of row locking.
- Carefully evaluate the LOCKSIZE parameter to ensure that the biggest lock size possible is used without causing contention. In cases of severe contention, you might want to consider the use of LOCKSIZE ROW.
- Delay updates until just before commit.
- Issue frequent commits.
- Design new applications and tune existing applications to ensure low elapsed time for each unit of work.

2.4 Estimating Data Sharing Overhead for an Application

If at all possible, simulate an application workload running in a data sharing environment. This simulation will enable you to gain an understanding of how much overhead the application will incur when its workload is distributed in a data sharing environment.

Use a transaction generation tool such as TPNS, which records the application transactions that run in your production non-data-sharing environment and then replays them in a test data sharing environment. If, for example, you record CICS transactions running in one CICS region in a non-data-sharing production environment, you can replay them on a test data sharing environment that has been set up to execute the transactions through CICSPlex SM. To predict the amount of overhead the application will incur in a data sharing environment, monitor the transactions being played back through one CICS region and compare the performance against the same transactions being executed in a data sharing environment through multiple CICS regions, using CICSPlex SM.

Refer to 10.7, “Transaction Run-Time Criteria: Data Sharing and Non-Data-Sharing” on page 204 for details on how to monitor the overhead.

2.5 Workload Distribution Using CICS

In this section we describe some ways to distribute your DB2 data sharing workload by using CICS and CICSPlex SM.

There are two basic types of workload distribution, static and dynamic. With static workload distribution, you define where the workload runs, whereas with dynamic workload distribution you let certain system components choose where the workload will run according to criteria such as availability and capacity.

Before we discuss ways of distributing workload in a data sharing environment, let us first look at CICS/DB2 Configurations before data sharing.

2.5.1 CICS/DB2 Configurations before Data Sharing

The number of ways of configuring a CICS/DB2 environment is infinite. In this section we discuss two examples of commonly used CICS/DB2 configurations that predate data sharing. But first let us give you very basic definitions of CICS MRO and non-MRO configurations:

- In a multi region operation (MRO) configuration, specific functions are allocated to different CICS regions. For example, a CICS region is assigned to be a terminal owning region (TOR) or an application owning region (AOR). The TORs and AORs work together to process a CICS transaction.
- In a non-MRO configuration, one CICS region performs all functions and does not have to interact with other CICS regions to process a transaction.

Figure 2 on page 12 shows an MRO configuration where groups of AORs execute particular subsets of CICS transactions accessing your DB2 data. On MVS1 the TOR distributes the transactions among AOR1, AOR2, and AOR3. This distribution is fixed, and the transaction will be defined to run on a specific AOR. On MVS2 the configuration is identical, but the subset of transactions being executed and the DB2 data being accessed are different. A distributed data facility (DDF) is not used, so transactions running on one MVS image cannot access the DB2 data on the other MVS image.

Note: The AORs run certain subsets of CICS transactions:

- AOR1, AOR2, and AOR3 run subset 1 of CICS transactions
- AOR4 and AOR5 run subset 2 of CICS transactions

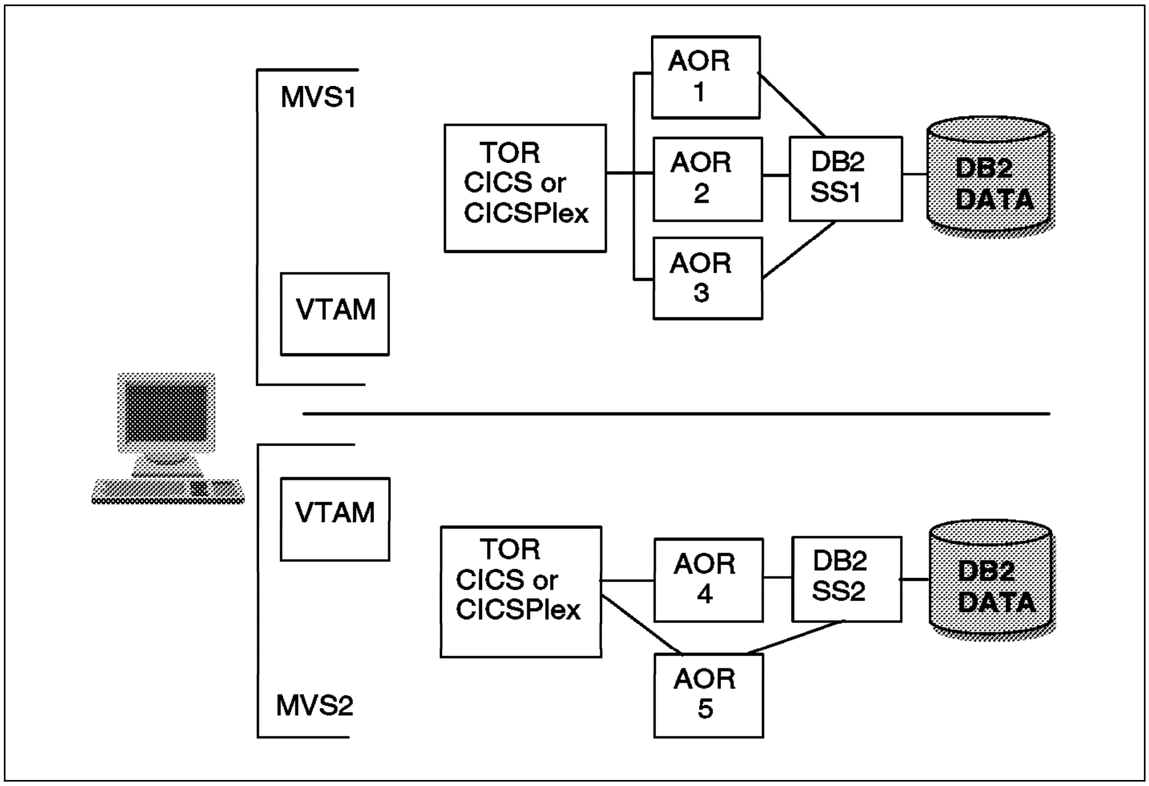


Figure 2. CICS/DB2: Configuration 1—Before Data Sharing

Figure 3 on page 13 shows a non-MRO configuration with CICS regions running application-specific subsets of CICS transactions. DDF is used when a CICS transaction on one processor has to access DB2 data on another processor.

Note: The CICS regions run application-specific subsets of CICS transactions:

- Region 1 runs application 1′s transactions
- Region 2 runs application 2′s transactions
- Region 3 runs application 3′s transactions
- Region 4 runs application 4′s transactions

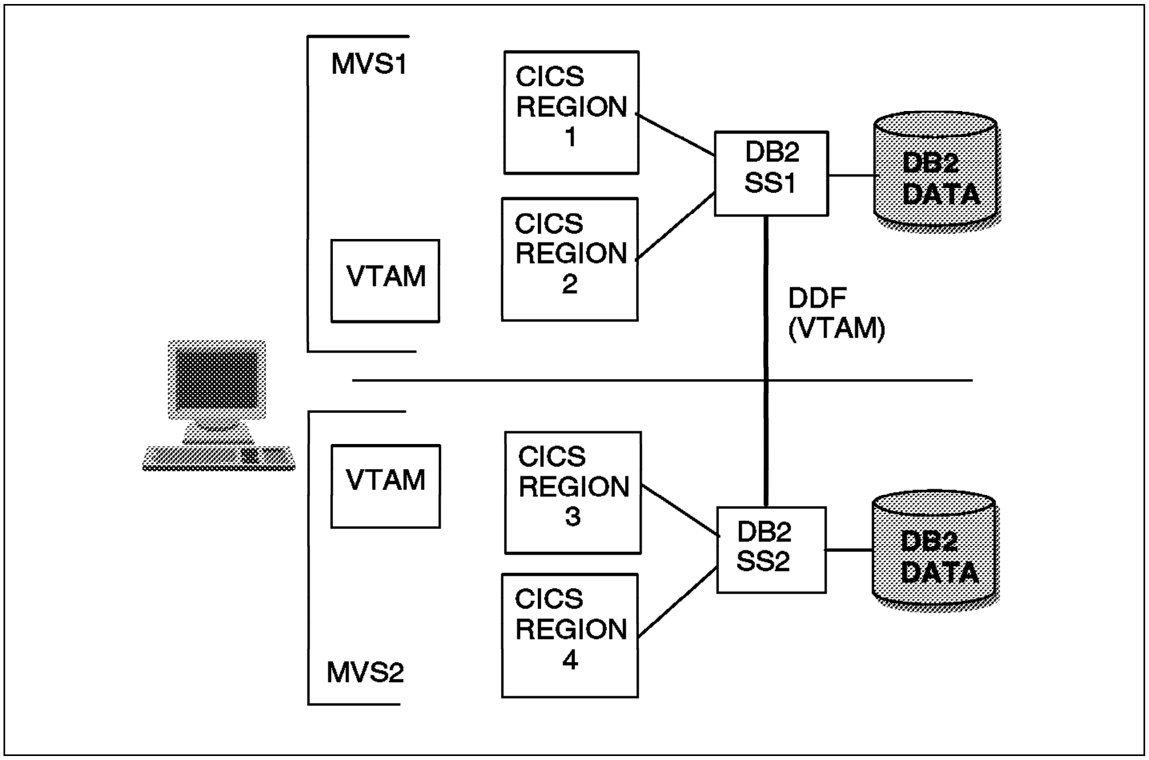


Figure 3. CICS/DB2: Configuration 2—Before Data Sharing

Below we discuss how the CICS/DB2 configurations can be reconfigured to distribute the work statically and then dynamically, once data sharing has been implemented.

2.5.2 Connecting CICS Regions to DB2 in a Data Sharing Environment

A CICS region is connected to DB2 by specifying the DB2 subsystem in the CICS region control task (RCT). It is not possible to define the data sharing group name to a CICS region. A region connects to a specific DB2 member.

2.5.3 Static Workload Distribution

Figure 4 on page 15 shows the changes that have been made to configuration 2 in Figure 3. Workload has been statically distributed across the Sysplex by moving CICS region 3 to the new MVS, MVS3. This distribution is possible because the third DB2 subsystem was created on the new processor as a member of the data sharing group. The processor now has efficient, direct access to the DB2 data, which used to be accessible only through the subsystem on MVS1.

Here is a summary of the actions required to reconfigure configuration 2 for a static data workload distribution environment:

- DB2 subsystems 1 and 2 were merged and are now both members of a data sharing group. A catalog merge was used. See Chapter 7, "Merging Existing DB2 Data into One Data Sharing Group" on page 117 for details.
- MVS3 was created on a new CMOS processor.
- A third DB2 subsystem, subsystem 3, was created on MVS3 as the third member of the data sharing group.
- CICS region 3 was moved off MVS2 and onto MVS3.
- DDF is no longer used, because DB2 subsystem 2 now has direct access to the DB2 data.

Static distribution can improve your processor usage. If capacity on a processor becomes constrained, you can move a CICS region off that processor and onto another with relative ease, thereby reducing the workload on the constrained processor. Static workload distribution is the quickest method of moving workload off your main processors and onto the new, less expensive CMOS processors.

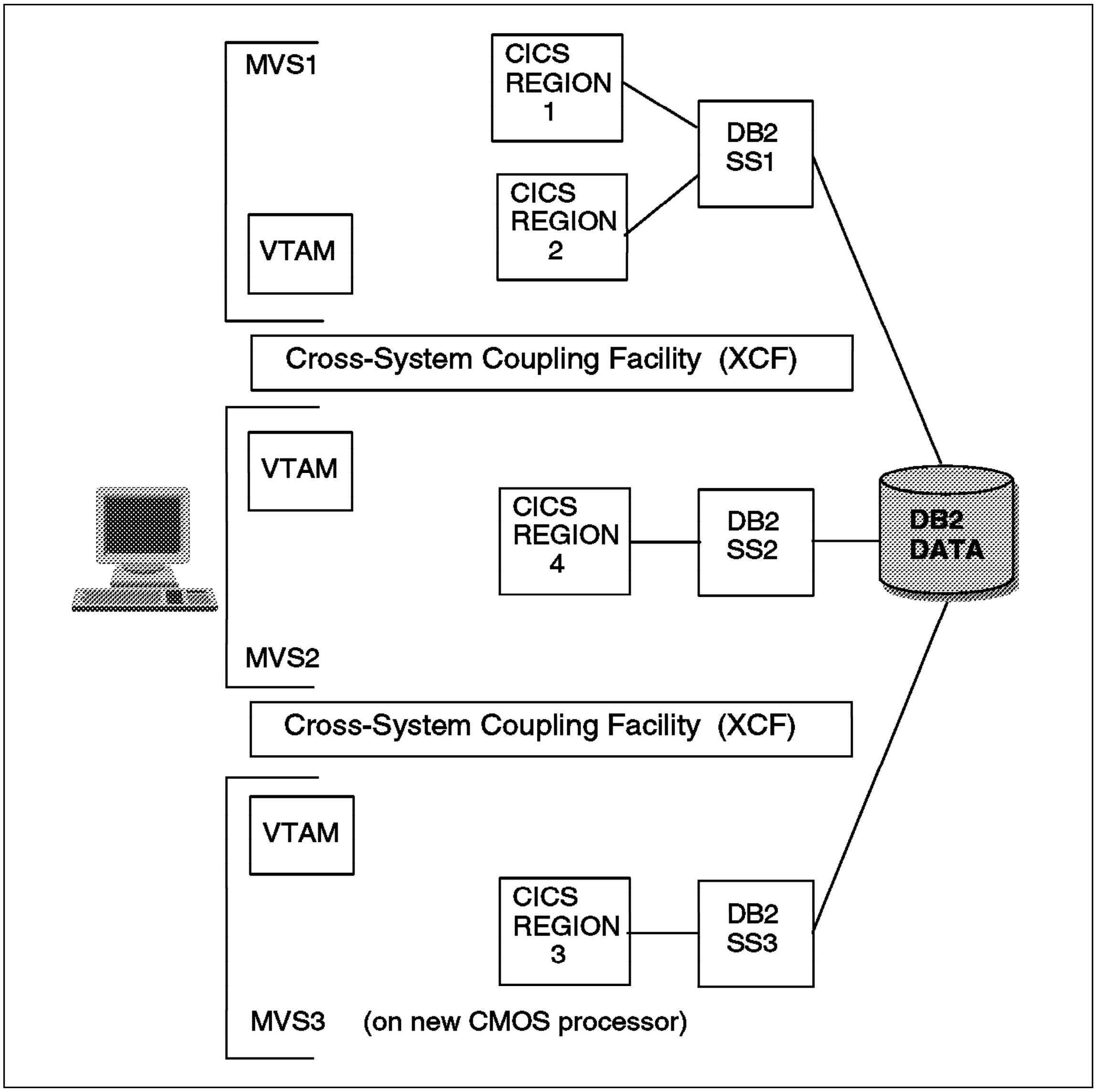


Figure 4. Static Workload Distribution: Reconfiguring Configuration 2 for a Data Sharing Environment

Static distribution does not improve system availability. Your workload is defined to run on specific software and hardware components, for example, a processor or a particular CICS region. As it is today, an application cannot function if any of the software or hardware components it uses are unavailable. For example, it cannot use another CICS region if the region it is currently using abends. It has to wait for that region to become available.

To reduce the overhead of data sharing in your environment for applications that do not have high availability requirements, consider the following implementation of static distribution, sometimes also referred to as *forced affinity routing*: Route all CICS transactions that access a certain subset of DB2 data through one CICS region. When all accesses come through one CICS region, there will be no inter-DB2 interest as all access to the data is done through one DB2 subsystem. Therefore the transactions will incur the minimum amount of data sharing overhead. You can implement this kind of transaction routing, using a technique called *CICS workload separation*. In a CICS workload separation scenario, transactions are routed from a TOR among a set of AORs, but the requirements of users, terminals, and the transactions themselves can influence which AOR or AOR set is used. In other words, workload separation is a way of applying some conditions of your own to workload balancing. Workload separation can be implemented by using CICSPlex SM′s dynamic transaction routing (DTR) program. For details on how the DTR program implements workload balancing, see 2.5.5.2, "CICSPlex SM′s Dynamic Transaction Routing Program" on page 17

2.5.4 Dynamic Workload Distribution

With dynamic workload distribution, you let certain system components choose where your workload will run according to such criteria as availability and capacity. Dynamic workload distribution enables a CICS transaction entering the Sysplex to be executed on any of the processors in the Sysplex. Dynamic workload distribution removes all application dependency on particular software and hardware components.

Dynamic workload distribution provides high systems availability and improved processor usage as workload is balanced out across available processors and CICS regions in the Sysplex. Refer to 2.5.5, "Implementing Dynamic Workload Distribution" on page 17 for details on how to configure your environment for dynamic workload distribution.

Dynamic workload distribution places the highest data sharing overhead on your environment, as it increases the degree of sharing across the data sharing group. Perhaps many DB2 subsystems are accessing the same data concurrently, so there may be a high level of inter-DB2 interest. See 2.2, "Data Sharing Overhead" on page 8 for a full discussion of what causes data sharing overhead and how to minimize it.

Your CICS workload cannot be dynamically distributed if your CICS transactions have affinities between them. Refer to 2.5.6, "CICS Transaction Affinities" on page 22 for a description of CICS transaction affinities.

Other factors that can inhibit dynamic workload distribution are:

- Availability of software components to all processors. Sometimes a piece of software is only licensed to run on a particular processor only. If an application uses this software, either it has to run on the processor on which the software is licensed, or the software license has to be upgraded to allow the software to run on multiple processors.

- Availability of hardware components to all processors. Hardware components have to be made available to all processors in the Sysplex. For example, optical disks that are currently accessible from only one processor have to be linked to all processors in the Sysplex.

2.5.5 Implementing Dynamic Workload Distribution

Dynamically distributing your DB2 workload across the members in a data sharing group improves the availability of your end-user applications.

2.5.5.1 CICS Session Balancing: If you use VTAM generic resources, your CICS sessions can be dynamically balanced across the Sysplex. At CICS session initiation time, VTAM routes the session to one of the available CICS regions in the Sysplex. VTAM interacts with the MVS workload manager (WLM) component (supported by MVS/ESA 5.1 and higher releases) to ensure that current processor usage is considered when choosing the TOR with which to establish the session. Figure 5 on page 19 shows an environment where VTAM has a choice of three TORs to establish new sessions with. These TORs should be on separate processors if high system availability is required, as this separation will isolate the applications running through these TORs from an outage if one of the processors becomes unavailable. In the event of a processor outage, existing sessions would abend, but new sessions could be established through TORs on different processors. For details on VTAM generic resources refer to the *VTAM V4 R3 for MVS/ESA Network Implementation Guide*.

Once a terminal has established a session with a TOR, new transactions initiated from that terminal will be routed to that TOR without rechecking how busy the processor where the TOR resides is.

Once the session has been established with a CICS region and transactions are being routed to that region, CICSPlex SM′s dynamic transaction routing program can be invoked to further distribute the workload through multiple AORs.

2.5.5.2 CICSPlex SM′s Dynamic Transaction Routing Program: For dynamic routing, CICSPlex SM′s DTR program distributes CICS workload among target AORs whose candidacy has been determined by their availability and activity levels. The DTR program can be set up to use different algorithms for distributing the workload among the candidate AORs. See *CICS/ESA 4.1 Application Programming Guide* for details on these algorithms.

Figure 5 on page 19 shows the evolution of configuration 1 in Figure 2 on page 12 to a dynamic workload balancing environment. The TORs on each MVS are linked to the AORs on their MVS and the AORs on the other MVSs. This gives CICSPlex SM the flexibility to further balance workload across available processors. In most cases CICSPlex SM will favors routing the incoming transaction to an AOR on the same MVS as the TOR. However, when the processor on which the TOR resides does not provide sufficiently good transaction performance, CICSPlex SM may route the transaction to an AOR on another processor where it knows transaction throughput is better.

Here is a summary of the main changes that have been made to reconfigure configuration 1 for a dynamic workload balancing environment:

- The CICS sessions are distributed among the three TORs using VTAM generic resources.
- Once the session is established with a TOR, CICSPlex SM′s DTR program further distributes the workload among the available AORs.
- The TORs on each MVS are linked to the AORs on their MVS and the AORs on the other MVSs.

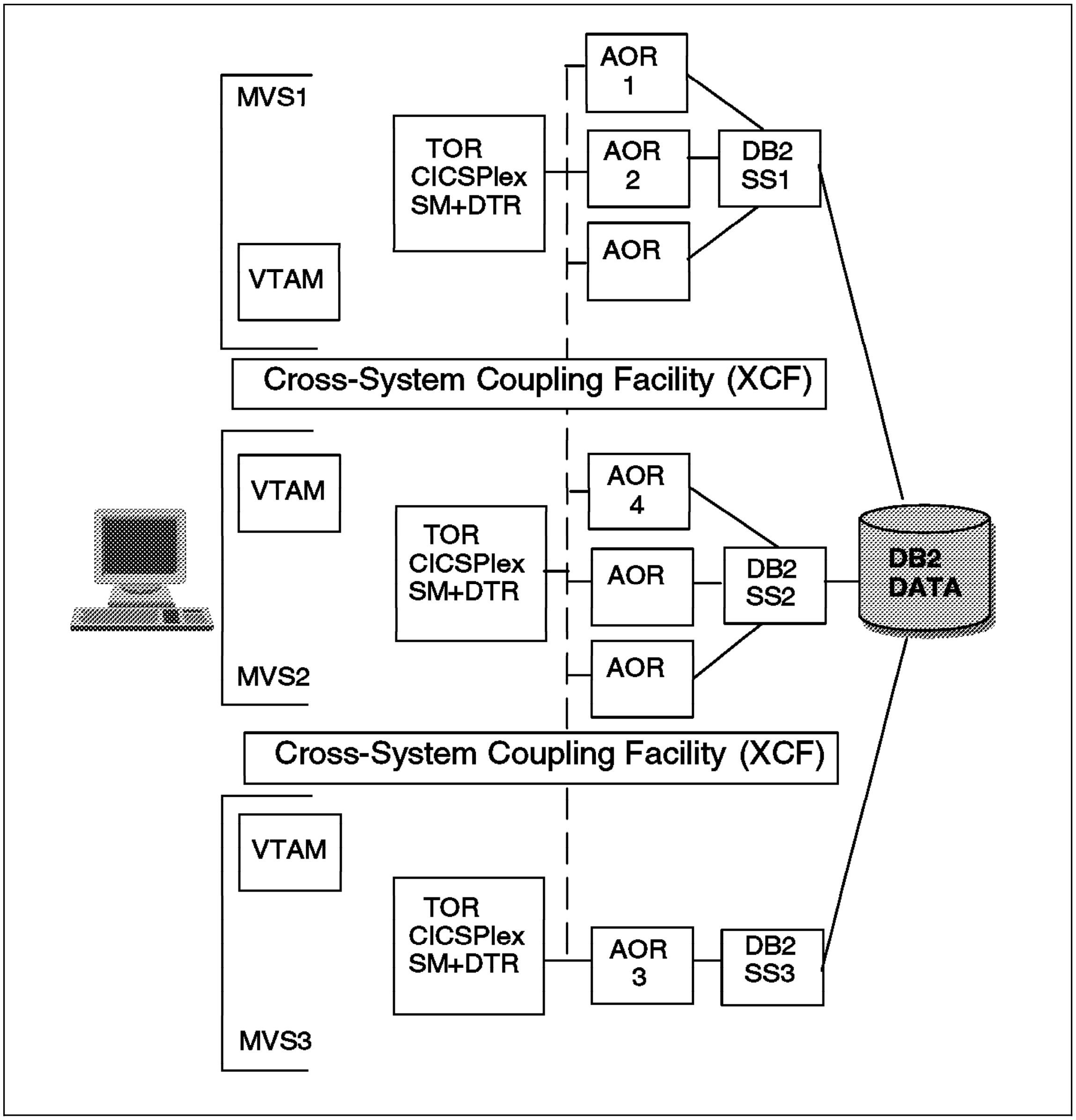


Figure 5. Dynamic Workload Balancing: Reconfiguring Configuration 1 for a Data Sharing Environment

Figure 6 on page 21 shows the evolution of configuration 2 to a dynamic workload balancing environment. All of application 1′s transactions that used to be executed through CICS region 1 are now under the control of CICSPlex SM. The other CICS regions have not yet been put under the control of CICSPlex SM.

The DTR program routes transactions away from busy regions and from those that are failing or likely to fail. This routine improves throughput and conceals problems from end users, thereby improving availability.

Here is a summary of the main configuration changes made to reconfigure configuration 2 for dynamic workload distribution:

- Application 1′s CICS transactions have had all transaction affinities removed.
- MRO and CICSPlex SM have been implemented and CICS region 1′s transactions have been moved into that environment (CICS region 1 is now referred to as *AOR1*)
- The CICS sessions for region 1 are distributed among the three new TORs using VTAM generic resources

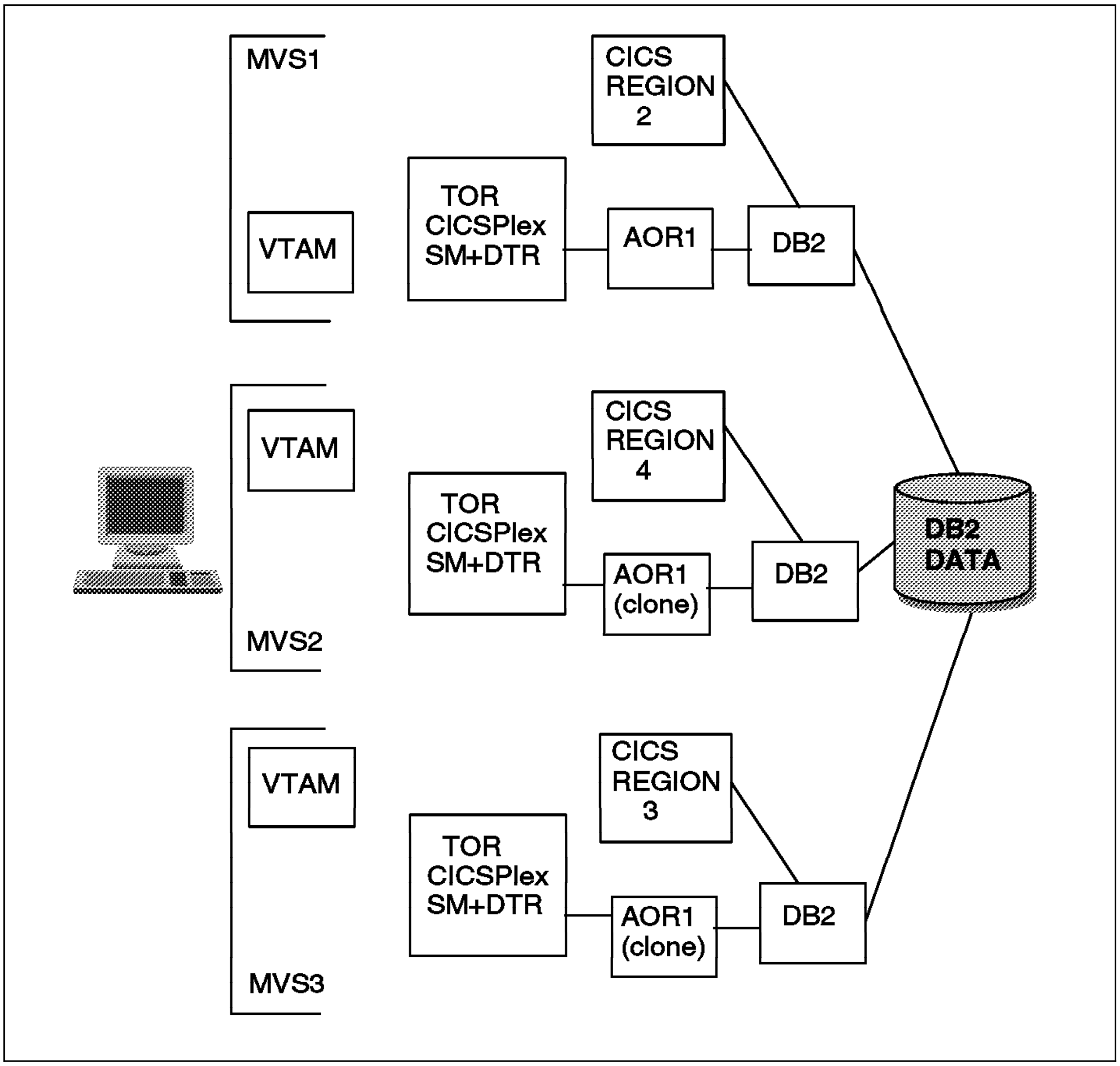


Figure 6. Dynamic Workload Balancing: Reconfiguring Configuration 2 for a Data Sharing Environment

MVS WLM: MVS WLM works at the CICS region level when trying to satisfy the performance goals you have defined to it for your CICS transactions. For example, if a CICS transaction has a performance goal of less than 1 second for 90% of the time, and the goal is not being met, MVS WLM will try to facilitate meeting the goal by raising the priority of the CICS region in which the transaction is running. It therefore makes sense to have CICS transactions that belong to high priority applications run in separate CICSPlexes from transactions that belong to less important applications.

So the next step in the evolution of configuration 2 would be to move all of CICS region 4′s transactions under the control of CICSPlex SM. If the performance goals of region 4′s transactions are not as high as the performance goals of region 1′s, region 4′s transactions could be moved into a separate CICSPlex rather than sharing the same CICSPlex region 1′s transactions use.

The Storm Drain Effect: When transactions appear to be completing rapidly through a CICS region, CICS starts routing more transactions through that region because it appears from a CICS perspective that that region is processing the most efficiently. This is known as the *storm drain effect*. The storm drain effect can occur during dynamic workload distribution if your site′s application programs are written in such a way that they end normally when encountering a DB2 resource unavailable condition. It can also occur when your CICS attachment facility is down and you are using the CICS INQUIRE EXITPROGRAM command.

An exit can be coded to avoid the storm drain effect caused by a resource unavailable SQLCODE -904, but the exit cannot cater for the INQUIRE EXITPROGRAM problem. CICS Transaction Server for OS/390 Release 1 caters for both scenarios that cause the storm drain effect, thereby removing this problem. Refer to *Applications Using CICSPlex SM* in the *IBM DATABASE 2 for MVS/ESA Version 4 Data Sharing: Planning and Administration* for details.

2.5.6 CICS Transaction Affinities

CICS transactions use many different techniques to exchange data. Some of these techniques require that the transactions exchanging data must execute in the same CICS region. Having to execute in the same region places restrictions on the dynamic routing of transactions because where these transactions execute is based on where other transactions have executed. If transactions exchange data in ways that impose such restrictions, there is said to be an affinity between them.

For the ways of identifying the affinities that exist in your applications, refer to the *CICS/ESA 4.1 Application Programming Guide*.

Your CICS workload cannot be dynamically balanced until all CICS transaction affinities have been eliminated. Quite often affinity elimination requires application design and coding changes or changes to CICS region configurations. Such changes typically require a high level of involvement of your project development teams and can take a long time to be implemented. Therefore focus on eliminating any affinities at the beginning of your data sharing implementation project if you intend to use CICSPlex SM to dynamically balance your workload.

2.6 IMS and DB2 in a Data Sharing Environment

The current release of IMS (V5) does not support VTAM generic resources, so there is no system-supported way of balancing sessions across your IMS subsystems. You must therefore divide your IMS network among your IMS subsystems to reduce the impact of an outage if an IMS subsystem should fail. Figure 7 on page 23 shows an IMS V5 static workload distribution configuration.

To balance your IMS transaction workload, use multiple system coupling (MSC) so that you can spread your workload across the IMS subsystems in a Sysplex. You can define an IMS transaction so that, no matter which IMS subsystem originates the transaction, it always runs on a particular IMS subsystem. Using MSC to spread IMS transactions among multiple IMS subsystems will limit the impact of an IMS subsystem failure as only a subset of your IMS transactions will be affected by the outage.

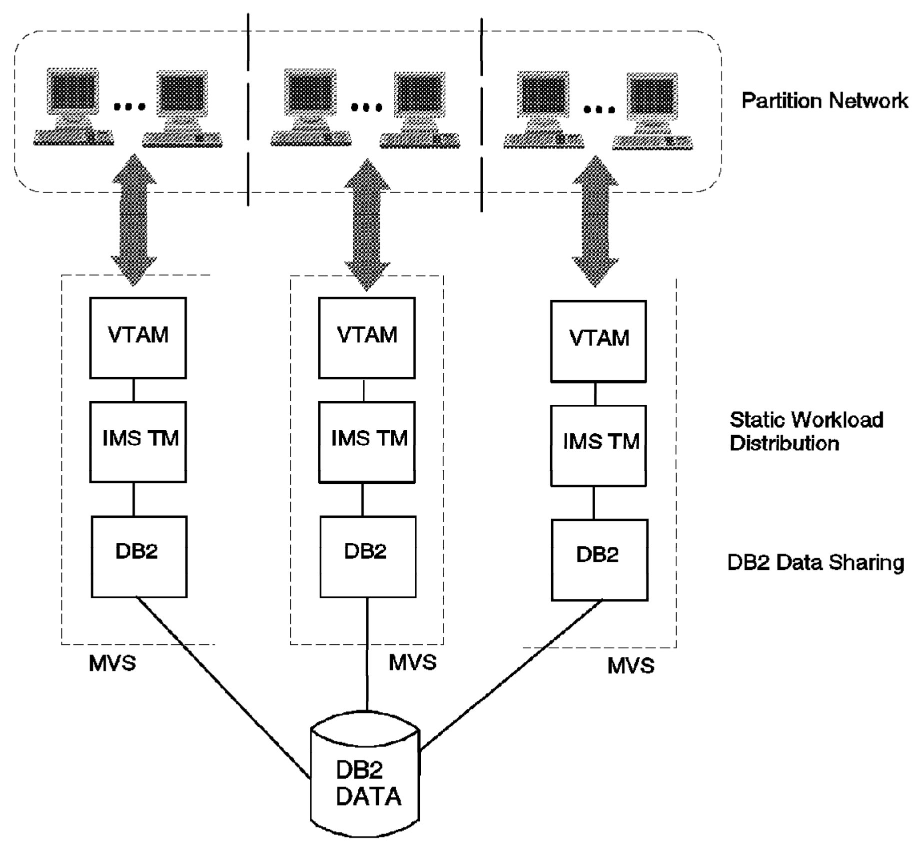


Figure 7. Static Workload Distribution with IMS Version 5

You cannot do dynamic workload distribution with IMS V5. IMS V6, however, will support VTAM generic resources. Thus it will be possible to distribute transactions across multiple

IMS control regions and thereby dynamically balance the IMS workload. Figure 8 on page 24 shows dynamic workload distribution with IMS V6.

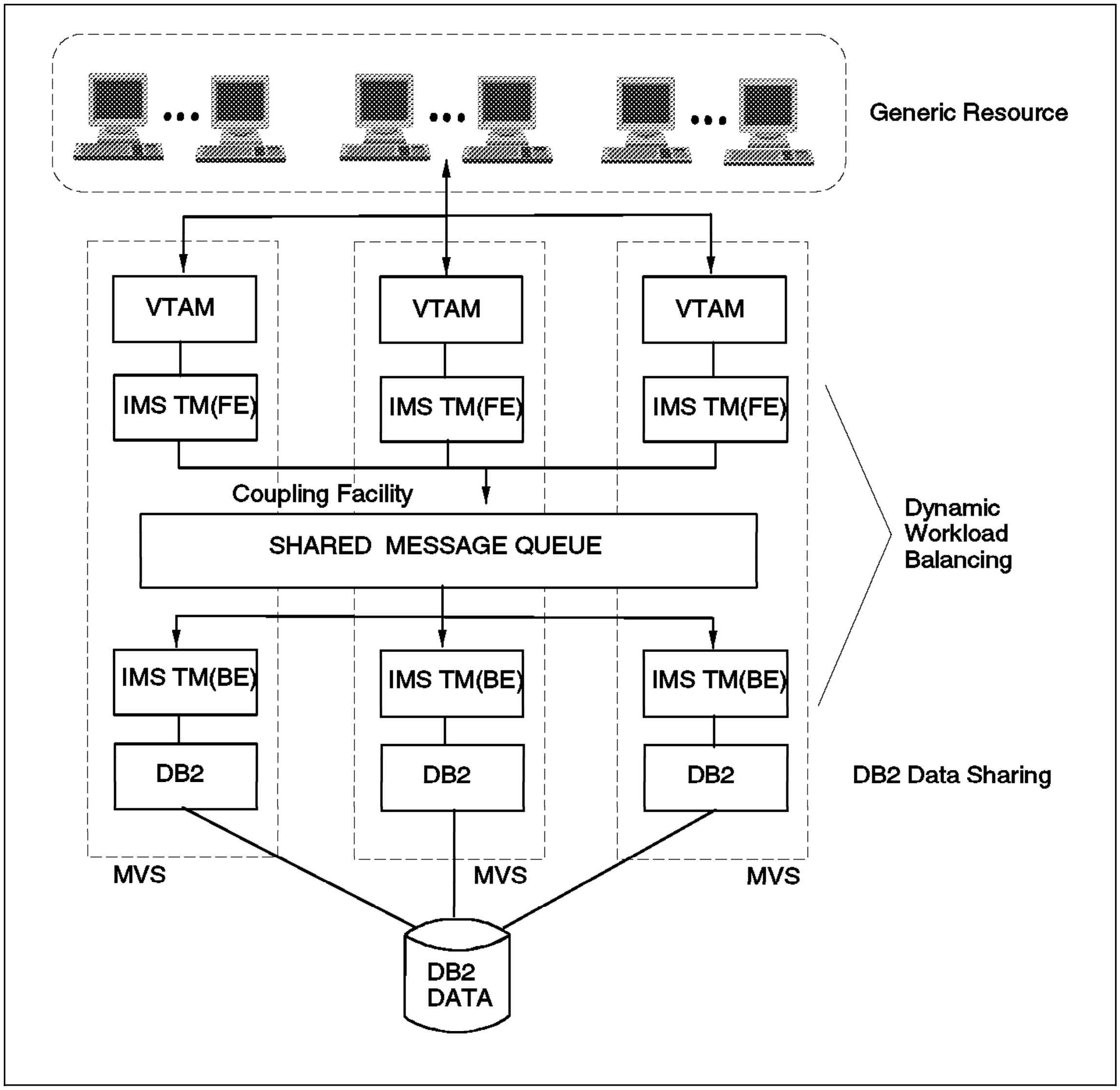


Figure 8. Dynamic Workload Distribution with IMS Version 6

2.7 Designing Databases and Applications to Exploit Data Sharing

When designing a new application, you have to ensure that the design you create facilitates exploitation of the data sharing environment. In this section we discuss some design techniques that can be used to great effect in a data sharing environment.

2.7.1 Data Partitioning for Concurrent Utility Operations

Partitioned designs work especially well in a data sharing environment. DB2 utilities can be split up to run against different partitions of a table space through multiple subsystems. For example, if you have a three-way data sharing environment with a 30-partition table space and you are executing 30 concurrent LOAD utilities, each LOAD against a different partition, you could run 10 LOAD jobs through each member of the data sharing group. In this way you will achieve a better throughput rate than you would if you ran all of the LOAD jobs through one subsystem, as you are effectively utilizing the resources of three subsystems (and potentially the capacity of three processors, if the subsystems are on different processors) to load your data.

If your partitioned tables have nonpartitioned indexes defined on them, be careful. In the example explained above, any nonpartitioned indexes would become GBP-dependent because they are being built from multiple subsystems concurrently. This dependency would significantly impact the performance of the LOAD jobs and negate the performance benefits gained by running the utilities through multiple subsystems. In addition to degrading performance, during the LOAD jobs nonpartitioned indexes would not be built cleanly and would therefore not provide optimal performance when they are traversed on completion of the LOAD jobs. Therefore when you run multiple LOAD or REORG utilities concurrently against partitions in one table space, we recommend that you drop the nonpartitioned indexes before you run the utilities and then create them separately once the concurrent execution of the utilities is complete. Of course you have to evaluate on a case-by-case basis whether you should follow this recommendation in your specific environment.

2.7.2 Data Partitioning for Distinct Key Range Processing

Data partitioning for distinct key range processing is a design technique that predates the advent of data sharing. Multiple batch jobs are executed concurrently against different partitions in one table space. With data sharing the design is improved by running the multiple batch jobs through multiple members in a data sharing group. As with concurrent utility operations, beware of any nonpartitioned indexes becoming GBP-dependent and affecting the overall performance of the concurrent jobs.

2.7.3 Using GBPCACHE ALL

The GBPCACHE parameter is a new parameter on the CREATE and ALTER TABLESPACE and INDEX statements.

The default for this parameter is CHANGED. GBPCACHE CHANGED specifies to DB2 that only pages that are changed will be cached in the GBP. GBPCACHE ALL tells DB2 to cache clean pages into the GBP, so that when a page is read into a members′ local buffer pool, the page is also cached and registered in the GBP.

Page sets or partitions where there is a high degree of inter-DB2 read interest are good candidates for GBPCACHE ALL. If the pages are in the GBP, several different members do not have to read the same page in from DASD. To prevent double buffering of clean pages, hiperpools are not used for page sets or partitions that are defined with GBPCACHE ALL.

When there is only one member in the data sharing group that has exclusive read-write interest, GBPCACHE ALL has no effect. Pages are not cached or registered in the GBP. As only one member is using the page set, there is no reason for the pages to be cached in the GBP also. As soon as another member in the group shows interest in the page set or partition, it becomes GBP-dependent; changed pages are moved to the GBP, and all pages are registered. The clean pages in the original members′ local buffer pool are not moved to the GBP, but any clean pages read in from DASD subsequently are placed in the GBP.

When considering the use of GBPCACHE ALL, remember that it increases the workload and resource consumption on the coupling facility.

Chapter 3. Planning for DB2 Data Sharing

Implementing DB2 data sharing is a major undertaking that requires the coordination of system hardware and software groups. In this chapter we provide DB2 systems programmers and database administrators with background information about the hardware and software areas that a DB2 data sharing implementation can affect. In each of these areas, we emphasize the considerations that can affect the function, availability, performance, and recoverability of a DB2 data sharing system and the underlying MVS Parallel Sysplex environment that supports it.

3.1 Capacity Planning

When you plan the capacity of the Parallel Sysplex for your DB2 data sharing workload, you have to consider:

- The additional CPU capacity to run your workload because of data sharing overhead
- The number of coupling facilities required for your workload and to meet your availability requirements
- The storage capacity of each coupling facility
- The number of coupling facility channels that attach each processor to each coupling facility

The main factors that determine both the overhead of data sharing and the capacity of the coupling facility are the degree of data sharing and the access rates to the coupling facility that result. These factors are highly dependent on your actual workload. The guidelines given below are based on measurements of a particular workload (the IBM Relational Warehouse Workload; for more details see *DB2 for MVS/ESA Version 4 Data Sharing Performance Topics*). Your workload may differ significantly from the IBM Relational Warehouse Workload, and it may be necessary for you to adjust the cost factors in the formula according to the measurements of your own workload.

3.1.1 Additional CPU Capacity to Run Your Workload

Ideally, as more identical processors are added to a processing complex, the transaction rate that DB2 could achieve would be the transaction rate achievable on a single processor, times the number of processors in the processing complex. However, the need for global locking and buffer management to ensure data consistency incurs additional DB2 and IRLM processing costs, thereby reducing the theoretical transaction rate that can be achieved. The processing costs vary with the degree of data sharing and are influenced by:

- Locking
  - Index type
  - Lock size
  - ISOLATION parameter

- CURRENTDATA option
  - RELEASE option
  - Commit frequency
  - Lock structure size
- Hardware and software configuration
  - Relative processor speed of processors running MVS and those running the coupling facility control code (CFCC).
  - Coupling facility structure size
  - Coupling facility channel configuration
  - Hardware and software levels
  - Number of members in the group
- Workload characteristics and effects
  - Effectiveness of lock avoidance
  - Amount of real lock contention
  - Amount of false lock contention
  - Amount of XES lock contention
  - Locking intensity
  - Buffer read-write ratios
  - Processor and DASD device contention
  - Workload variations over the day

3.1.1.1 Capacity Loss: In 2.2, "Data Sharing Overhead" on page 8 we discuss the concept of data sharing overhead. In this section, we discuss a related concept, that of capacity loss.

Three cost factors are associated with the performance overhead in a Parallel Sysplex environment:

The multisystem management cost
A fixed cost, measured at around 3%, associated with migrating from a non-Sysplex to a Sysplex environment. It is the MVS overhead associated with cross-system coupling facility (XCF) processing, GRS processing, shared JES2 checkpoints, and other miscellaneous items.

The initial cost
The cost for the second member to join the DB2 data sharing group. For the IBM Relational Warehouse Workload, the initial cost was 13.29% (Type 2 indexes, page locking, and degree of data sharing 100%).

The additional cost
The cost for each additional member to join the DB2 data sharing group. For the IBM Relational Warehouse Workload the additional cost was 0.26% for the third member.

The cost for adding second and subsequent members to the DB2 data sharing group also depends on the degree of data sharing, defined as the percentage of the total processing capacity that is dedicated to processing applications using data sharing.

Capacity Loss Formula: The loss in capacity, compared with the ideal, required for a Sysplex with *n* members and *d* degree of sharing can be expressed with the following formula:

```
Capacity Loss (%) = 1 + d * (2 + (3 * (n - 2))) / 100

    1 = Multisystem management cost
    2 = Initial data sharing cost
    3 = Additional data sharing cost
```

Factor **1** is fixed at an estimated 3%. The values for the other two factors (**2** and **3**) were obtained by performance tests on the IBM Relational Warehouse Workload run at the IBM Santa Teresa Laboratory. Substituting them into the formula above results in:

```
Capacity Loss (%) = 3 + d * (14 + (0.5 * (n - 2))) / 100
```

Using this formula as a guide, we derived the graph in Figure 9 for varying numbers of members in the data sharing group.

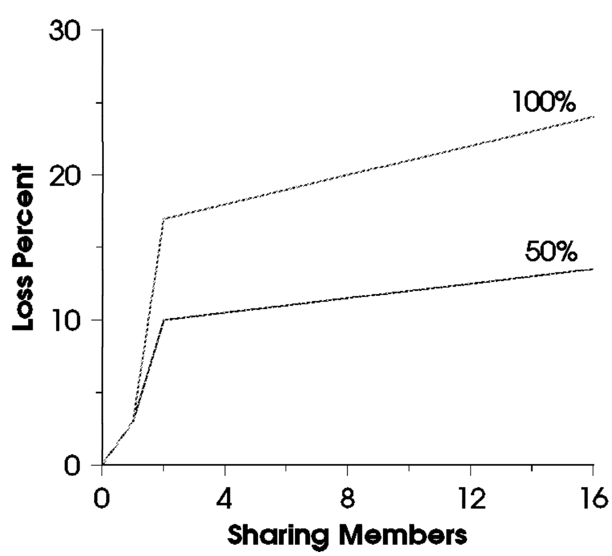


Figure 9. Projected Capacity Loss with Degree of Data Sharing 50% and 100%

3.1.2 Processing Capacity for the Coupling Facility

The number of processors required for the coupling facility depends on the speed and number of the sending processors and the degree to which the sending processors use coupling facility functions.

The *System/390 MVS Parallel Sysplex Performance* redbook gives the following formula to determine the minimum number of processors needed for a coupling facility:

$$MinimumCPs = \frac{1}{12} \times \sum_{n} \frac{Large\ Systems\ Performance\ Reference_{n}}{Large\ Systems\ Performance\ Reference_{cf}} \times CP_{n}$$

This number should be rounded up to the next whole number. The LSPR ratios can be found in the *Large Systems Performance Reference*.

The above formula is only a general guideline; it can be adjusted up or down depending on the number of functions using structures in the coupling facility. It can also be adjusted for the percentage of the sending processors that are actually using the coupling facility.

3.1.3 Multiple Coupling Facilities

The number of CPs calculated above can be divided among multiple coupling facilities for availability and recovery reasons. You can then distribute coupling facility structures among different coupling facilities. Such distribution allows the coupling facility structures to be rebuilt in another coupling facility should a structure or the coupling facility itself fail.

The coupling facility can be run on either dedicated CPCs or under a PR/SM logical partition with other logical partitions running MVS. For a dedicated coupling facility, one or more of the processors in a CPC are dedicated to the coupling facility logical partition. This configuration offers the best performance, as dedicated processors require less logically partitioned mode (LPAR) dispatching overhead.

For a production Parallel Sysplex, we recommend having at least one dedicated coupling facility. You should also avoid, where possible, running a coupling facility in a logical partition on a processor with other logical partitions running MVS images that also rely on the coupling facility. A processor failure would mean the loss of both the coupling facility and the MVS images as well as a difficult recovery process.

3.1.4 Storage Capacity for the Coupling Facility

Within a coupling facility, storage is dynamically allocated in areas called *structures*. You define the structures in an MVS CFRM policy and the subsystems use XES to manipulate data within the structures.

There are three types of structures: cache structures, list structures, and lock structures. DB2 data sharing uses all three structures to provide the following functions:

Global lock management
: IRLM uses a single lock structure per DB2 data sharing group to control locking across all members of the group.

Communication among group members
: An SCA uses a single list structure per DB2 data sharing group to allow efficient access to essential information by all members of the group. The SCA contains information about databases with exception conditions, and other information.

Group buffer pools
: A DB2 data sharing group can use up to 60 cache structures as GBPs (fifty 4KB GBPs and ten 32KB GBPs). DB2 uses a GBP to cache data that is of interest to more than one member in the data sharing group. A separate GBP is used for each correspondingly named DB2 buffer pool. For example, data associated with buffer pool 8 (BP8) would use GBP8, if that data is to be shared. You must therefore define a GBP cache structure for each DB2 buffer pool associated with table spaces whose data is to be shared.

The additional storage that a coupling facility needs for DB2 data sharing is the sum of all of its structures, that is, the sum of the space required for the lock structure, the SCA, and the GBP structures. Remember you will probably have more than one coupling facility.

3.1.4.1 The IRLM Lock Structure: The space in the lock structure is divided into a lock table, which is used to detect contentions, and a modified resource list (MRL), which saves information on modify locks. Unfortunately, you cannot tell IRLM the size of the lock table; it is inferred from other parameters in the IRLM procedure. IRLM attempts to use approximately one-half of the structure space for the lock table. However, the size of the lock table entries depends on the number of IRLMs in the data sharing group, and the number of entries must be a power of 2.

A lock table must be of sufficient size to minimize false contentions. However, choosing a lock table size that will achieve a reasonable level of false contentions (less than 0.1%) is highly workload dependent. For example, a larger lock table is required with increasing multiprogramming levels, longer duration units of work, and heavy update intensity. There must also be sufficient space in the modified resource list to hold the modify locks. IRLM reserves 10% of the modified resource list space for ″must complete″ functions that are performed during phase 2 commit or rollback processing. As the modified resource list fills up and the remaining 90% of the space becomes exhausted, transactions requesting modify locks will abend.

Use the following method to derive an initial estimate for the space required for the lock structure:

1. Estimate the number of lock entries required.

Sum the IRLM virtual storage used for holding locks (specified in MAXIMUM ECSA on the DSNTIPJ DB2 installation panel) for each data sharing member and divide the result by the space IRLM requires for an individual lock (about 250 bytes). Round up the result to a power of 2.

Note: In a DB2 data sharing environment, additional virtual storage requires IRLM for physical locks, which are held on open page sets and objects in the environmental description manager (EDM) pool.

2. Determine the space for the lock table.

   Multiply the number of lock entries required (the result of step 1) by the lock entry size. The size of a lock entry is dependent on the number of IRLMs connected to the group. For 1 - 7 IRLMs, each entry requires 2 bytes; for 8 - 23 IRLMs, 4 bytes; and for 24 - 32 IRLMs, 8 bytes. The initial lock table entry size is determined by the MAXUSRS parameter of the first IRLM that connects to the lock structure. MAXUSRS defaults to 7, which is probably the number you want, unless you plan to immediately create a data sharing group of more than 7 members. By using the default for MAXUSRS, you can maximize the lock table size for a given lock structure size.

   Note: Should the number of IRLMs connecting to the lock structure exceed the value of MAXUSRS, IRLM automatically rebuilds the lock structure to increase the size of the lock table entries. This action reduces the number of lock table entries and may result in increased false contentions.

3. Determine the space for the lock structure.

   Having determined the space required for the lock table, you calculate the optimum space for the lock structure by multiplying the lock table size by 2. This calculation will ensure that approximately one-half the space in the lock structure is available for the modify locks. If you choose to specify less space than this for the lock structure, the number of modify locks that can be acquired will be reduced. See the tables in INFOSYS entry (Q662530) for the lower and upper bounds of the modify locks when different storage sizes are chosen.

3.1.4.2 The SCA List Structure: DB2 allocates a single list structure for the SCA. Its size is dependent on:

- The number of databases
- The number of tables
- The amount of exception information

Table 1 on page 33 gives recommendations for SCA structure sizes on the basis of varying values for the above entities.

Table 1. Recommendations for DB2 SCA Structure Size			
Size of DB2 System	**Number of Databases**	**Number of Tables**	**Size of SCA (KB)**
Small	50	500	8192
Medium	200	2000	16384
Large	400	4000	32768
Very Large	600	6000	49152

3.1.4.3 Group Buffer Pool Cache Structures: DB2 supports up to fifty 4KB GBPs and ten 32KB GBPs. One cache structure is allocated for each of the GBPs used by the DB2 group.

DB2 GBP cache structures consist of two parts:

Directory entries These specify the location and status of an image of a data or index page somewhere in the data sharing group, whether the image appears in the GBP or in one or more of the member buffer pools.

Data entries These are where the actual data pages are stored. They can be either 4KB or 32KB in size depending on the size of the data page.

The ratio of directory entries to data entries is automatically defined for each GBP when the first member of the DB2 group is installed. The default value used is five directory entries per data entry. You can change this ratio after installation, using the ALTER GROUPBUFFERPOOL command. However, the change does not take effect until the next time the GBP is allocated.

Use “buffer updates” as your estimate. The buffer updates will be closer to the number of pages pushed to the coupling facility (as all changed pages are synchronously moved to the coupling facility at commit time). The number of buffer updates may be slightly higher than the number of pages pushed to the coupling facility if the same page is updated more once in the same unit of work before it is externalized to the coupling facility. It is best to err on the side of safety, and we are only talking about rough estimates anyway. When estimating the storage required for a GBP, consider:

- The total number of items that must be tracked in the local buffer pool of each DB2 member
- The amount of data that has to be cached

  The amount depends on whether DB2 is writing only changed pages to the GBP or all pages. The type of caching DB2 performs is controlled at a partition and index level by the GBPCACHE parameter setting; the default is CHANGED, that is, cache only changed pages.

Use the following method to derive an initial estimate for the size of a GBP that is caching only changed pages:

1. Determine the level of write activity to the local buffer pool as a percentage of the overall activity to the local buffer pool.

2. Sum the local buffer pool storage (both virtual buffer pool and hiperpool storage) for all DB2 members in the group.

3. Multiply the two values obtained in steps 1 and 2 to give the estimated space for the GBP.

When caching read-only pages and changed pages, the estimated space for the GBP can be obtained by multiplying the sum of the local buffer pool storage (only the virtual buffer pool, not the hiperpool storage) for all DB2 members in the group by a factor depending on the number of page sets that have GBPCACHE ALL specified. This factor ranges from 50%, if only a few page sets specify GBPCACHE ALL, to 100% if most page sets specify GBPCACHE ALL.

For example: Assume that there are eight members in a DB2 group, and that each member has a virtual buffer pool of 30 MB and a hiperpool of 70 MB. Also assume that the overall level of write activity to the buffer pool is 20%. Then the sum of their buffer pool sizes is:

8 x (30 MB + 70 MB) = 800 MB

And the estimated storage for their GBP is:

800 MB x 20% = 160 MB

If one-half the page sets in the above example were specified with GBPCACHE ALL, the estimated storage for the GBP would be:

(8 x 30 MB) x 75% = 180 MB

3.1.4.4 Placement of Coupling Facility Structures: You must also consider the placement of the coupling facility structures for availability and performance reasons. For instance, in a heavy data sharing system, you might have to place the DB2 lock structure in a different coupling facility from the DB2 GBPs to better balance the number of requests to the coupling facilities. For availability, you must leave sufficient unallocated space in your coupling facilities to rebuild structures if either a structure or a complete coupling facility should fail. This structure rebuild facility is controlled by a system failure management (SFM) policy.

3.1.5 An Alternative Way of Sizing Coupling Facility Structures

There are two main steps:

1. Estimate initial structure sizes for the definition in the CFRM policy.

2. Check actual structure use and resize.

3.1.5.1 Estimate Initial Structure Sizes for the Definition: Before you can start DB2 in data sharing mode you must define one lock structure, one SCA structure, and a GBP structure associated with each local buffer pool in a data sharing group.

With MVS 5.1 you can only define a maximum size, with MVS 5.2 and later the structures can be dynamically resized from an initial size, INITSIZE, up to the maximum size SIZE. The resizing requires you to execute the MVS SETXCF command. Table 2 gives provides information for estimating the initial size for the structures.

Prefer a larger value to avoid problems caused by structures that are too small!

Table 2. Initial Structure Size Estimates

SCA structure	DSNDSGC_SCA	8 MB - small DBMS 16 MB - medium DBMS 32 MB - large DBMS 48 MB - extra large DBMS
Lock structure	DSNDSGC_LOCK1	(MAX ECSA in DSNTIPJ * # member) / 2 Round up to a 4 MB multiple
GBPx structure	DSNDSGC_GBP0 DSNDSGC_GBP1 ...	CACHE=CHANGE: SUM(BPx) of all members * 20% CACHE=ALL: SUM(BPx) of all members * 100%

3.1.5.2 Check Actual Structure Use and Resize: You have two options for checking actual structure use:

- Control actual use of the structure, using the -DISPLAY GROUP and -DISPLAY GROUPBUFFERPOOL DB2 commands, as shown in Figure 10 and Figure 11 on page 36.

```
.
=DBB1 DIS GROUP
DSN7100I =DBB1 DSN7GCMD 182
*** BEGIN DISPLAY OF GROUP(DSNDSGB )
-------------------------------------------------------------------------
DB2                                         DB2   SYSTEM  IRLM
MEMBER    ID   SUBSYS  CMDPREF   STATUS    LVL  NAME    SUBSYS  IRLMPROC
--------  ---  ----    --------  --------  ---  ------  ------  --------
DBB1        1  DBB1    =DBB1     ACTIVE    410  SC48    IRB1    DBB1IRLM
DBB2        2  DBB2    =DBB2     ACTIVE    410  SC53    IRB2    DBB2IRLM
-------------------------------------------------------------------------
SCA   STRUCTURE SIZE:     4096 KB, STATUS= AC,   SCA IN USE:     1 %
LOCK1 STRUCTURE SIZE:    16128 KB,             LOCK1 IN USE: <   1 %
NUMBER  LOCK ENTRIES:     4194304
NUMBER  LIST ENTRIES:       54486, LIST ENTRIES  IN USE:           5
*** END DISPLAY OF GROUP(DSNDSGB )
DSN9022I =DBB1 DSN7GCMD 'DISPLAY GROUP ' NORMAL COMPLETION
```

Figure 10. DISPLAY GROUP

```
DSNB750I =DBB1 DISPLAY FOR GROUP BUFFER POOL GBP0 FOLLOWS
DSNB755I =DBB1 DB2 GROUP BUFFER POOL STATUS
             CONNECTED                                 = YES
             CURRENT DIRECTORY TO DATA RATIO           = 5
             PENDING DIRECTORY TO DATA RATIO           = 5
DSNB756I =DBB1   CLASS CASTOUT THRESHOLD                   = 10%
             GROUP BUFFER POOL CASTOUT THRESHOLD       = 50%
             GROUP BUFFER POOL CHECKPOINT INTERVAL     = 8 MINUTES
             RECOVERY STATUS                           = NORMAL
DSNB757I =DBB1 MVS CFRM POLICY STATUS FOR DSNDSGB_GBP0     = NORMAL
             MAX SIZE INDICATED IN POLICY              = 16000 KB
             ALLOCATED                                 = YES
DSNB758I =DBB1     ALLOCATED SIZE                          = 8192 KB
               VOLATILITY STATUS                       = VOLATILE
DSNB759I =DBB1     NUMBER OF DIRECTORY ENTRIES             = 8066
               NUMBER OF DATA PAGES                    = 1610
               NUMBER OF CONNECTIONS                   = 2
DSNB782I =DBB1 INCREMENTAL GROUP DETAIL STATISTICS SINCE 14:59:55 JUN
DSNB784I =DBB1 GROUP DETAIL STATISTICS
             READS
               DATA RETURNED                           = 129
DSNB785I =DBB1     DATA NOT RETURNED
                 DIRECTORY ENTRY EXISTED               = 107
                 DIRECTORY ENTRY CREATED               = 451
                 DIRECTORY ENTRY NOT CREATED           = 12, 0
DSNB786I =DBB1   WRITES
               CHANGED PAGES                           = 262
               CLEAN PAGES                             = 24
               FAILED DUE TO LACK OF STORAGE           = 0
             CHANGED PAGES SNAPSHOT VALUE              = 7
DSNB787I =DBB1   RECLAIMS
               FOR DIRECTORY ENTRIES                   = 0
               FOR DATA ENTRIES                        = 0
             CASTOUTS                                  = 176
DSNB788I =DBB1   CROSS INVALIDATIONS
               DUE TO DIRECTORY RECLAIMS               = 0
               DUE TO WRITES                           = 155
DSNB790I =DBB1 DISPLAY FOR GROUP BUFFER POOL GBP0 IS COMPLETE
DSN9022I =DBB1 DSNB1CMD '-DIS GBPOOL' NORMAL COMPLETION
```

Figure 11. DISPLAY GROUPBUFFERPOOL

These commands should be executed in a regular time interval.

- Control use of the structure, using the RMF Coupling Facility Report and DB2 PM Statistics Report.

3.1.6 Capacity and Coupling Facility Channels

In planning coupling facility channel capacity, you have to consider the speed of the channels, their distance, and the number of channels required.

3.1.6.1 Link Speed and Distance: IBM currently provides two types of fiber optic links for coupling facility channels to support different distance requirements. The first is a multimode fiber with a speed of 50 MB/sec for a distance of up to 1 km. The second is a singlemode fiber with a speed of 100 MB/sec for a distance of up to 10 km. The length of the fiber optic link between the sending processor and the coupling facility can also affect the performance of the sending processor, by increasing the response time the sending processor sees from the coupling facility. The increase in response time of the coupling facility caused by the length of the coupling facility channels is about 10 µsec per kilometer.

For DB2 data sharing, this additional response time would mainly affect the performance of DB2 lock requests, which are synchronous coupling facility operations. GBP operations (4 KB) are also mostly synchronous coupling facility operations.

3.1.6.2 Number of Coupling Facility Channels: The number of coupling facility channels required is primarily determined by the rate of requests from the sending processor to the coupling facility. This rate is dependent on the speed of the sending processor and the level of data sharing among the DB2 members.

System/390 MVS Parallel Sysplex Performance provides some guidelines on the number of coupling facility channels to use. These guidelines are based on performance measurements of specific workloads.

Depending on your coupling facility configuration, you may also require additional coupling facility channels for availability. As a minimum, you should plan for two coupling facility channels from each sending processor.

3.2 Hardware Planning

In this section we describe the hardware components and other factors you have to consider when planning a Parallel Sysplex configuration to support DB2 data sharing. The hardware components in a Parallel Sysplex include:

- The processors that will be involved in data sharing. These processors must be capable of being attached to a coupling facility.
- Sysplex Timers, which coordinate the time-of-day clocks for the processors in the Sysplex.
- Coupling facilities, which provide the data sharing functions for systems in the Sysplex.
- Coupling facility channels, which connect the processors to the coupling facilities.
- ESCON Directors, which control the switching of point-to-point connections between processors and control units.
- Control units for devices such as DASD, tape drives, printers, consoles, and workstations.

3.2.1 Processors in the Parallel Sysplex

The following processors are capable of linking to the coupling facility in a Parallel Sysplex:

- 9021 711-based models with SEC 228270

- 9121 511-based models with SEC C35954
- S/390 microprocessor CPCs

Connectivity Be aware of the following connectivity considerations:

- You must install coupling facility channels to connect these processors to the coupling facility. You can order the channels as field upgrades for existing processors or as a new factory shipment.

  Note: The installation of coupling facility channels in an existing processor requires an outage of that processor.

- You can use ESCON channels to connect the processor to ESCON Directors or ESCON control units. For connection to control units that require parallel channels, you can use an ESCON Converter Model 1.
- You can install and use parallel channels as needed.
- You can use the Integrated Coupling Migration Facility (ICMF) on a processor to define a coupling facility logical partition and test coupling facility functions among MVS images on that processor. The ICMF emulates the coupling facility channels, so that real coupling facility channel hardware does not have to be installed for testing.
- You can use the ESCON multiple image facility(EMIF) to share ESCON and coupling facility channels among logical partitions.

Limitations: Be aware of the following limitations:

- For 9021 711-based models and 9121 511-based models, up to eight coupling facility channels can be installed. For an S/390 microprocessor CPC, up to 24 coupling facility channels can be installed. However, the number of coupling facility channels that you install can reduce the number of other kinds of available channels.
- Because of the ICMF′s extra overheads, consider using it for testing only. Also, as the MVS images using the ICMF must reside on the same processors as the coupling facility logical partition, a failure of the processor will cause extended recovery times, because both the MVS images and the coupling facility on which they depend will have failed.

More detailed information can be found in *System/390 MVS Sysplex Hardware and Software Migration*.

3.2.2 Sysplex Timer

At least one Sysplex Timer must be installed to synchronize the time-of-day clocks for the MVS systems that run on separate processors in the Sysplex. The time-of-day clocks on all processors in the Sysplex must be synchronized to ensure that subsystems and applications have a mechanism for determining the order of events, regardless of which processor in the Sysplex performed them.

For instance, DB2 relies on being able to determine the order of updates to its databases so that it can recover from subsystem or database failures. When only a single DB2 system can update a database, DB2 can use its own mechanism—the log relative byte address (RBA)— to determine the order of events. However, in a data sharing environment, where multiple DB2 systems may be updating the same database from different processors, DB2 requires a means for determining the order of events among these multiple systems. Thus DB2 generates a value, the log record sequence number (LRSN), based on the time-of-day clock of the processor, which, because of the Sysplex Timer, can be guaranteed to be consistent among all processors in the Sysplex. DB2 can then use the LRSN during recovery to determine the order of updates performed by the data sharing DB2 systems.

Daylight Saving

It has been a recommendation that in those countries which have daylight saving time for part of the year, that the TOD clock is not changed for a non-Parallel-Sysplex environment. As the Sysplex Timer controls the TOD clock in a Parallel Sysplex environment, IBM recommends that, to avoid problems, you set the Sysplex Timer (and hence TOD clock) to Greenwich Mean Time (GMT).

If the TOD clock is set back, log records can be invalidated as a result of sequencing problems, meaning recoveries cannot take place. This has been a consideration in IMS for some time, but is new to DB2.

In addition, within a Parallel Sysplex setting the TOD clock back may cause timestamps to be duplicated in the system couple data set. If a Sysplex member is attempted to be brought up in the period of duplicate time, then because the highest timestamp may be greater than the current timestamp, the new member will fail initialization.

If the TOD clock is set forward, then a problem can be caused by MVS which uses the failure-detection interval specified in the COUPLExx PARMLIB member to determine system failure. When the TOD clock is set forward, the difference between a timestamp previously recorded in the couple data set and a timestamp obtained from a recent TOD clock could exceed the specified failure detection interval. In this case, MVS would initiate a ″status update missing″ condition, which in turn could cause the system to be removed from the Sysplex.

Local time can be changed dynamically using the SET TIME command to receive the correct local time on displays and reports, if supported by the applications.

If you plan to run your Parallel Sysplex on a single S/390 processor (for example, as a test DB2 data sharing system), a Sysplex Timer is not required; the time-of-day clock of the processor can be used.

Availability Considerations: The following availability considerations are important:

- Connection of a Sysplex Timer to a processor is by a fiber optic link and does not require a processor outage. However, to start using the Sysplex Timer as your time reference, you have to update the CLOCKxx member of PARMLIB and perform an IPL of MVS.
- The Sysplex Timer is a critical component in the MVS Sysplex. If an MVS system loses connectivity to the Sysplex Timer, for instance, because of the failure of the fiber optic link, it enters a nonrestartable wait state and is removed from the Sysplex. If the Sysplex Timer itself fails, all MVS systems in the Sysplex are placed in nonrestartable wait states.

  To prevent the outages that the above failures would cause, we recommend installing two Sysplex Timers with the expanded availability feature and duplexing the connections between the Sysplex Timers and each processor. For high availability, you should also connect each Sysplex Timer to a separate power source or use a continuous power supply.

Further information about the Sysplex Timer can be found in *IBM 9037 Sysplex Timer and System/390 Time Management*.

3.2.3 Coupling Facility

At least one coupling facility must be installed and defined to MVS before you can run DB2 with data sharing capability.

The coupling facility provides shared storage and shared storage management functions, such as high-speed caching, list processing, and locking for the Sysplex. Subsystems, such as DB2, running on MVS images in the Sysplex define the shared structures (areas of storage) used in the coupling facility. Thus the subsystems can efficiently share data and process their workload in parallel across the MVS systems in the Sysplex.

A coupling facility is a special logical partition that runs the CFCC, which is Licensed Internal Code. You automatically load the CFCC from the processor controller or support element when you activate the coupling facility logical partition.

Support for coupling facility logical partitions is provided by the following processors:

- 9021 711-based models
- 9121 511-based models, only when ICMF is enabled on the processor
- S/390 microprocessor CPCs

Depending on the processor model, you can define one or more coupling facility logical partitions on the processor.

Three types of coupling facilities can be defined:

Dedicated Coupling Facility: A dedicated coupling facility is a CPC that runs only the coupling facility in a logical partition with dedicated processor resources. There are two kinds of dedicated coupling facilities:

- A stand-alone coupling facility, that can run only the coupling facility
- Other CPCs that enable you to dedicate the CPC to the coupling facility

The 9674 is a special S/390 microprocessor CPC that can run only as a coupling facility. It also has an optional expanded area for additional coupling facility channels.

Coupling Facility Logical Partition Shared with Other Logical Partitions on the Same Processor: The 9021 711-based models and the S/390 microprocessor CPCs allow a coupling facility logical partition to be run with other logical partitions on the same CPC. You can then use coupling facility channels to connect the coupling facility logical partition to other MVS images on the same processor or to other processors capable of connecting to a coupling facility. The 9121 511-based models must use the ICMF on the processor to define a logical partition in which to run a coupling facility; they cannot use real coupling facility channel hardware. Anyway, ICMF is not meant to be used in a production environment.

Coupling Facility Logical Partition and the Integrated Coupling Migration Facility: All processors that are coupling-facility-capable (except the 9674 stand-alone coupling facility) can define coupling facility logical partitions, together with other logical partitions, on the same processor and use the ICMF to emulate the coupling facility channels. Thus other MVS images on the same processor can connect to the coupling facility logical partition. However, you cannot connect the coupling facility logical partition to MVS images on other processors.

3.2.3.1 Coupling Facilities for a Production Environment: To ensure good DB2 data sharing performance, you must ensure that the coupling facilities have adequate processor capacity, processor storage, and connectivity to each of the MVS systems in the Parallel Sysplex. Section 3.1, “Capacity Planning” on page 27 discusses ways of estimating coupling facility capacity.

Also, although several different types of coupling facilities can be defined, not all are suitable for running a coupling facility in a production data sharing environment. For a production data sharing environment we recommend installing at least one dedicated coupling facility for these reasons:

- Failure of the coupling facility and a connected MVS image, which is a more likely possibility when they execute on the same CPC, can lead to extended recovery times.
- Better performance can be obtained by using dedicated coupling facility hardware, for instance, a 9674 coupling facility.
- Better connectivity to the coupling facility can be achieved by using dedicated coupling facility hardware, such as a 9674 Coupling Facility.

Having multiple coupling facilities in a production Sysplex makes recovering from a failed MVS image or coupling facility relatively easy. The remaining processors in the Sysplex can continue to process new work, and, depending on how you define your coupling facility structures in the CFRM policy and on the application itself, the application may be able to rebuild its structures on the other coupling facilities.

3.2.3.2 Coupling Facilities for Testing: A shared coupling facility, as defined above, may be a viable option for testing data sharing or providing a lower-cost backup coupling facility, for instance for disaster recovery. *WSC Flash 9609 MVS/ESA Parallel Sysplex Performance LPAR Performance Considerations for Parallel Sysplex Environments* discusses the performance implications of the various options in detail.

3.2.3.3 Other Considerations: There are three other matters to consider:

- The coupling facility does not support dynamic I/O configuration or dynamic storage configuration. To redefine storage for the coupling facility logical partition, you must first deactivate the logical partition, make the changes, and then reactivate the logical partition. To reconfigure the coupling facility channels for the coupling facility, you must perform a power-on-reset of the processor that runs the coupling facility.
- You do not have to connect the Sysplex Timer to the coupling facility.
- If you are using the ICMF, you can define up to two coupling facility logical partitions on the processor. An MVS image that uses the ICMF cannot be connected to another coupling facility through coupling facility channels.

3.2.4 Coupling Facility Channels

Coupling facility channels require a direct point-to-point link between the processor and the coupling facility. The coupling facility channel on the processor side must be defined as a TYPE=CFS channel and that on the coupling facility side must be defined as a TYPE=CFR channel. For configurations that have multiple MVS images defined on a processor, you can use the ESCON multiple image facility to share TYPE=CFS channels. However, you cannot use the ESCON multiple image facility to share TYPE=CFR channels.

3.2.5 ESCON Directors and ESCON Control Units

An ESCON Director performs dynamic switching for point-to-point connections between processors and control units. It can connect channels from a processor to another processor or to control units for storage or communication control.

ESCON control units control access to DASD, tape devices, or communication equipment such as terminals and workstations and provide improved connectivity and performance.

3.2.5.1 Connectivity: ESCON Directors simplify the connectivity problems of connecting the potential many (up to 32) MVS images in a Sysplex to peripheral control units. Each MVS image has four channel paths to the ESCON Director and there are four channel paths from the ESCON Director to the 3990 ESCON control unit. The ESCON Director can dynamically switch any of the channels from the MVS images to any of the channels attached to the 3990 control unit. A configuration without an ESCON Director would require the 16 channel paths from the four MVS images to be directly connected to the 3990 control unit.

3.2.5.2 Availability: For high availability you should, where possible, connect two or more ESCON Directors to each processor and each control unit. (Some ESCON control units support connection to only one ESCON Director.)

3.2.5.3 Performance: For optimum performance,

- Use DASD control units that have a high logical path capacity, such as:
  - 934x control units with up to 64 logical paths
  - 3990 Storage Control Model 6 with up to 128 logical paths.
- Use cache DASD control units to improve data sharing among multiple systems.
- For automatic tape switching, which allows multiple systems in the Sysplex to switch to shared tape devices without operator intervention, use 3480 or 3490 control units.

3.2.5.4 Other Considerations: Two other matters need consideration:

- Coupling facility channels cannot be connected to the ESCON director.
- Use ESCON manager software to control I/O configurations in the Sysplex. The ESCON manager:
  - Controls ESCON and non-ESCON I/O devices
  - Provides a single point of control for issuing commands to vary devices online and offline and for configuring channels online and offline
  - Obtains status information for all I/O resources on all systems in the Sysplex, in a single, combined view

3.3 MVS Software Planning

In this section we provide a high-level overview of the software levels required to run DB2 data sharing and discuss the specific changes required to MVS to support DB2 data sharing.

3.3.1 MVS Software Prerequisites

Table 3 on page 44 shows the recommended and minimum software levels required to support DB2 data sharing. The table also includes information about the additional functions that software levels above the minimum required provide.

Table 3. MVS Software Requirements

Recommended Release of Product	Additional Function	Minimum Release Required or Tolerated
MVS/ESA SP 5.2.0 or above	This release added a number of significant enhancements for DB2 data sharing: • Prefetch and lock processing enhancements. These require a coupling facility to be at CFLEVEL=2 and MVS authorized program analysis reports (APARs) OW15587 and OW15590. • The ability to automatically restart a DB2 subsystem after a failure, using the MVS Automatic Restart Management (ARM) facility. • The ability to use the MVS WLM for dynamic routing of DDF transactions.	MVS/ESA SP 5.1
JES2 SP 5.2.0 or above	This release of JES2 is required if you want to use the ARM to perform cross-system restarts.	JES2 SP 4.2 with PTFs
JES3 SP 5.2.1	This release of JES3 is required if you want to use the ARM to perform cross-system restarts.	JES3 SP 3.1.3
DFSMS/MVS 1.2 or above	• System group name support allows you to specify a Sysplex name so that all MVS systems in the Sysplex are included in the SMS complex under a single name. • Support for IMS DB (local shared resources) for up to 32 systems to share data for IMS DB (OSAM and VSAM support).	• DFP 3.3 with PTFs • DFSMS/MVS 1.1 with PTFs
RACF 2.1 or above	• Provides data security for the Sysplex and can use the coupling facility to share RACF control data for security management for all systems. • Can propagate selected RACF commands for systems in the Sysplex to simplify security management.	RACF 1.9
VTAM 4.2 or above	This release is required if you want to use a VTAM generic name for DB2 DDF members.	VTAM 3.4.1 with PTF

For a comprehensive list of these and other software products and compatibility concerns in an MVS/ESA Version 5 environment, see *MVS/ESA SP V5 Planning: Installation and Migration with JES2* and *MVS/ESA SP V5 Planning: Installation and Migration with JES3*.

3.3.2 I/O Definitions

You have to use the hardware configuration definition (HCD) to define:

- The coupling facility logical partitions and the coupling facility channels they use.
- The coupling facility channels used by the sending processors.
- Any other hardware changes you have made to allow for the increased connectivity requirements of a Sysplex.

3.3.3 MVS Couple Data Sets

A Parallel Sysplex requires:

- A couple data set to store information about the systems in the Sysplex, the XCF groups and members defined, and general status information
- A CFRM couple data set to hold coupling facility policies, which define how MVS is to manage coupling facility resources.
- The SFM policy is highly recommended for DB2 data sharing for proper handling of coupling facility and coupling facility link failures.

> **Note**
>
> If you have APAR OW19718 installed, CONNFAIL(YES) in the SFM policy is not a requirement.

Depending on the policies you define to help manage resources and workload for the Sysplex, you may have to define these additional couple data sets to hold these policies:

- An SFM couple data set, which enables you to define how system failures, signaling connectivity failures, and PR/SM reconfiguration actions are to be managed
- A WLM couple data set, which enables you to define service goals for workloads. This couple data set is required only when WLM is run in goal mode.
- An ARM couple data set, which enables you to define how MVS is to manage restarts for specific jobs and started tasks
- A system logger (LOGR) couple data set, which enables you to define log stream or structure definitions

The different types of policies can reside in the same couple data set or in different couple data sets. Before a policy can be defined and activated, you must first define and format a couple data set. MVS provides the IXCL1DSU utility in SYS1.MIGLIB, which you can use to format couple data sets. This utility is fully described in Appendix A, "Format Utility for Couple Data Sets," in *MVS/ESA Setting Up a Sysplex*.

3.3.3.1 Connectivity: As couple data sets contain information about the Sysplex, they must be defined on DASD that is accessible by all systems in the Sysplex.

3.3.3.2 Availability: Couple data sets are critical to the operation of a Sysplex. Therefore we recommend that you create an alternate couple data set on a different volume, control unit, and channel from the primary couple data set. MVS maintains information in both the primary and alternate couple data sets concurrently. If the primary data set fails, the Sysplex automatically makes the alternate data set the primary.

3.4 Network Planning

DB2 data sharing may in fact reduce the load on your network, as you may be able to combine some of your existing distributed applications into a data sharing group where they no longer have to be distributed. DB2 data sharing also does not use any VTAM resources.

You may want to implement VTAM generic resources, however, if you use CICS or DB2 DDF to access your DB2 data sharing group. VTAM generic resources can increase the availability of your DB2 data for CICS or DDF applications by enabling you to use a common name to access your CICS TORs or DB2 members. With this common name, if a CICS TOR or DB2 member fails, only a subset of your network is affected. End users can log on again immediately and continue processing, not knowing that they are using a different CICS TOR or DB2 member.

Before you can implement VTAM generic resources, you must implement Advanced Peer-to-Peer Networking (APPN). At least one VTAM in the Sysplex must be defined as an APPN network node. To ensure that you can still use VTAM generic resources if the VTAM being used as the APPN network node fails, define at least two VTAMs to act as APPN network nodes in your Sysplex.

Getting your network ready to use VTAM generic resources could take some time. You should therefore allow for this time in any schedule you are developing.

3.5 Software Maintenance Considerations

The following rules for release coexistence in the data sharing environment are now enforced by APAR PN84642:

1. Only two releases of DB2 can coexist in the data sharing group.
2. After the data sharing group reaches release N, it stays as release N.
3. If the data sharing group is at release N, the N-1 release of DB2 must have the small programming enhancement (SPE) applied before it can start.
4. All active DB2 members at the N-1 release must have the SPE applied.

In a non-data-sharing environment, after a DB2 subsystem is started with release N, the subsequent restart of the same DB2 subsystem with release N-1 (the fallback scenario) must have the proper SPE applied.

Chapter 4. DB2 Planning

In this chapter we highlight some areas that you should pay particular attention to when planning your data sharing implementation project.

We discuss some points to consider when drawing up a naming convention for the data sharing group and its members, and we include some checklists that you can use to ensure that all activities required to implement data sharing are in your own project plans.

The checklist activities are based on customer experiences. The importance of each activity is site specific, but the two areas we believe should get a high degree of focus in your project are performance and recovery. Both areas become complex in a data sharing environment and can easily be neglected in the push to get data sharing implemented.

Another area we would like to draw your attention to is that of technical team interaction and exchange of skills. The successful implementation of data sharing to meet your company′s business objectives involves coordination of the efforts of many technical teams, such as the MVS team, CICS team, VTAM team, operations, DB2, and application development teams. To coordinate the efforts of these teams, an overall project manager should be appointed who would ensure that project interdependencies are tracked and managed and that change windows are farmed out among the different areas.

4.1 Naming Convention

Carefully consider your naming convention for all DB2 components of your data sharing group. Choosing names that have flexibility in terms of the number and placement of DB2 subsystems is important because after installation some names, such as the group name and member names, cannot be changed.

If you already have DB2, your naming convention will be influenced by the names of your existing DB2 subsystems. In some cases, however, you may want to "start from scratch." For a procedure to rename an existing DB2 subsystem, refer to 4.1.1, "Procedure to Change DB2 Naming Convention" on page 50.

Chapter 3 of *IBM DATABASE 2 for MVS/ESA Version 4 Data Sharing: Planning and Administration* has a detailed discussion on naming the various components of your data sharing group. It also contains a suggested naming convention, which you might want to consider using.

When choosing your group and member names, consider the impact that changing these names will have on the development teams that use DB2. To minimize the impact, consider promoting the originating member subsystem name to the group name. If this name change is performed as part of the actual upgrade procedure itself, application JCL changes are required, as effectively now all application JCL references the data sharing group′s name, and your DB2 developers can continue to reference the old familiar subsystem name, which is actually now the group name.

You can promote the originating member subsystem name to the group name with or without changing the originating member subsystem name. In other words, you could end up having the same name for both the group and one of the members in the group. Having the group and a member using the same name has some complications associated with it; please refer to the discussion of the group attachment function in 9.1.3, "Changes to Support the Data Sharing Environment" on page 163.

4.1.1 Procedure to Change DB2 Naming Convention

In this section, we describe a procedure that can be used to change the naming convention for DB2 system data sets, user databases, and DB2 and IRLM subsystem names. For a more detailed discussion on changing the high-level qualifiers for DB2 data sets, please refer to Chapter 2, ″Changing the High-Level Qualifier for DB2 Data Sets,″ in the *IBM DATABASE 2 for MVS/ESA Version 4 Administration Guide*.

The procedure to change your DB2 naming convention takes time to execute, and DB2 is unavailable during some of the steps. If you intend to change the naming convention of your existing DB2 subsystems, plan in advance when you can tolerate a DB2 outage.

Here are the steps:

1. Define new ICF VSAM aliases and optionally new ICF user catalogs where these new aliases will reside.
2. Define new RACF rules to protect the new aliases. Define new RACF rules for the renamed DB2 and IRLM address spaces.
3. If you plan to change the DB2 LUNAME used by the DDF, define the new LUNAME to VTAM.
4. Change DSNZPARM to specify the new high-level qualifier for the DB2 catalog, directory, and archive log data sets.
   - macro DSN6SPRM
     - change CATALOG — new catalog and directory data set high-level qualifier
     - change IRLMPRC — new IRLM JCL procedure name
     - change IRLMSID — new IRLM subsystem name
     - change SYSADM — new install SYSADM (optional)
     - change SYSADM2 — new install SYSADM2 (optional)
   - macro DSN6ARVP

- change ARCPFX1 — new archive log data set high-level qualifier (optional)
- change ARCPFX2 — new archive log data set high-level qualifier (optional)

- macro DSN6SYSP
  - change STORPROC — new stored procedure address space JCL procedure name
- macro DSNHDECM
  - change SSID — new default DB2 subsystem name (optional)

5. Quiesce the DB2 subsystem.

   Make sure all activity has been properly quiesced for the DB2 subsystem before continuing.

   To ensure this, run the Print Log Map utility (DSNJU004) against the quiesced BSDS, and then run the DSN1LOGP utility specifying SUMMARY(ONLY) or SUMMARY(YES) with a STARTRBA and ENDRBA of the last DB2 checkpoint taken from the output of the DSNJU004 execution. Verify there are no outstanding URs or pending writes by examining the Restart Summary portion of the report. Do not continue until all outstanding activity has been resolved.

   Here is a sample job to execute the DSN1LOGP utility:

```
//STEP1 EXEC PGM=DSN1LOGP
//SYSPRINT DD SYSOUT=*
//SYSSUMRY DD SYSOUT=*
//SYSABEND DD SYSOUT=*
//BSDS     DD DSN=VSDB2PD1.BSDS01,DISP=SHR
//SYSIN    DD *
  RBASTART (0A033BB8706E) RBAEND (0A033BB88F33)
  SUMMARY(YES)
/*
```

6. Using IDCAMS, rename the DB2 system data sets specifying the new high-level qualifier:

   - Rename the DB2 catalog and directory data sets.
   - Rename the BSDSs.
   - Rename application table space and index data sets.
   - Delete and define all the DB2 active log data sets found with a status of ″REUSABLE″ in the Print Log Map (DSNJU004) output.

   We show below a sample SQL statement to generate the appropriate VSAM ALTER statement for table spaces. A similar SQL statement can be coded for indexes:

```
SELECT
' ALTER ''VSDB2PD.DSNDBC.'||DBNAME||'.'||TSNAME||
'.I0001.A001'' - NEWNAME (''DSNDSGA.DSNDBC.'
||DBNAME||'.'||TSNAME||
'.I0001.A001'')',
' ALTER ''VSDB2PD.DSNDBD.'||DBNAME||'.'||TSNAME||
'.I0001.A001'' - NEWNAME (''DSNDSGA.DSNDBD.'
||DBNAME||'.'||TSNAME||
'.I0001.A001'')'
FROM SYSIBM.SYSTABLEPART
WHERE DBNAME IN
('DSNDB04','DSNDB06','DSNDDF','DSNRGFDB','DSNRLST')
AND PARTITION=0
UNION
SELECT
' ALTER ''VSDB2PD.DSNDBC.'||DBNAME||'.'||TSNAME||
'.I0001.A'||SUBSTR(DIGITS(PARTITION),3,3)||
''' - NEWNAME (''DSNDSGA.DSNDBC.'||DBNAME||'.'||TSNAME||
'.I0001.A'||SUBSTR(DIGITS(PARTITION),3,3)||''')',
' ALTER ''VSDB2PD.DSNDBD.'||DBNAME||'.'||TSNAME||
'.I0001.A'||SUBSTR(DIGITS(PARTITION),3,3)||
''' - NEWNAME (''DSNDSGA.DSNDBD.'||DBNAME||'.'||TSNAME||
'.I0001.A'||SUBSTR(DIGITS(PARTITION),3,3)||''')'
FROM SYSIBM.SYSTABLEPART
WHERE DBNAME IN
('DSNDB04','DSNDB06','DSNDDF','DSNRGFDB','DSNRLST')
AND NOT PARTITION = 0
```

Alternatively you can use a number of generic IDCAMS ALTER statements.

```
 ALTER VSDB2PD.DSNDBC.dbname.*.I0001.A001  -
    NEWNAME DSNDSGA.DSNDBC.dbname.*.I0001.A001
 ALTER VSDB2PD.DSNDBD.dbname.*.I0001.A001  -
    NEWNAME DSNDSGA.DSNDBD.dbname.*.I0001.A001
```

7. Update the BSDS to reflect the new high-level qualifier:

   - Specify the new catalog and directory alias to use.
   - Delete the old active log data set names from the BSDS and specify the new active log data sets to use.
   - Optionally change the DDF location and LUNAME.

   Here is a sample job to update the BSDS.

```
//DSNTLOG EXEC PGM=DSNJU003
//STEPLIB  DD  DSN=SYS1.DSNLOAD,DISP=SHR
//SYSUT1   DD  DISP=OLD,DSN=DSNDSGA1.DBA1.BSDS01
//SYSUT2   DD  DISP=OLD,DSN=DSNDSGA2.DBA1.BSDS02
//SYSPRINT DD  SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//SYSIN    DD  *
    NEWCAT VSAMCAT=DSNDSGA
    DELETE DSNAME=VSDB2PD1.LOGCOPY1.DS02
    DELETE DSNAME=VSDB2PD2.LOGCOPY2.DS02
    DELETE DSNAME=VSDB2PD1.LOGCOPY1.DS03
    DELETE DSNAME=VSDB2PD2.LOGCOPY2.DS03
    DELETE DSNAME=VSDB2PD1.LOGCOPY1.DS04
    DELETE DSNAME=VSDB2PD2.LOGCOPY2.DS04
    NEWLOG DSNAME=DSNDSGA1.LOGCOPY1.DS02,COPY1
    NEWLOG DSNAME=DSNDSGA2.LOGCOPY2.DS02,COPY2
    NEWLOG DSNAME=DSNDSGA1.LOGCOPY1.DS03,COPY1
    NEWLOG DSNAME=DSNDSGA2.LOGCOPY2.DS03,COPY2
    NEWLOG DSNAME=DSNDSGA1.LOGCOPY1.DS04,COPY1
    NEWLOG DSNAME=DSNDSGA2.LOGCOPY2.DS04,COPY2
    DDF    LOCATION=DSGA,LUNAME=DB2DSGA
//*
//LIST    EXEC PGM=DSNJU004
//STEPLIB  DD  DSN=SYS1.DSNLOAD,DISP=SHR
//SYSUT1   DD  DISP=OLD,DSN=DSNDSGA1.DBA1.BSDS01
//SYSUT2   DD  DISP=OLD,DSN=DSNDSGA2.DBA1.BSDS02
//SYSPRINT DD  SYSOUT=*
//SYSUDUMP DD  SYSOUT=*
```

8. Change the DB2 and IRLM JCL procedures to reflect the new DB2 and IRLM subsystem names:
   - Rename all DB2 and IRLM address space JCL procedures to reflect the new subsystem names.
   - Update the BSDS names in DB2 MSTR address space JCL.
   - Update the SUBSYS parameter in the DB2 SPAS address space JCL.
   - Update the IRLMNM parameter in the IRLM address space JCL.
9. Update the MVS PARMLIB member IEFSSNxx with the new DB2 command prefix and the new DB2 and IRLM subsystem names.

   If you have installed MVS V5.2, you can use the MVS SETSSI command. Otherwise you must implement this change by scheduling an IPL.
10. Start DB2 with ACCESS(MAINT), using the new DSNZPARM member.
11. Redefine the sort work table spaces to use the new VSAM high-level qualifier.
    a. Stop database DSNDB07, using the following command:

```
-STOP DATABASE(DSNDB07) SPACENAM(*)
```

b. Drop and re-create all DSNDB07 table spaces using the new VSAM high-level qualifier. Do not forget to grant the use of these table spaces to PUBLIC.

   Use installation job DSNTIJTM as a model for this activity.

c. Start database DSNDB07

12. To change the high-level qualifier for user-defined table spaces and indexes:

    a. Stop all the user-defined table spaces and indexes. Use the following query to identify these objects:

```
SELECT '-STOP DATABASE('||DBNAME||') SPACENAM(*)'
FROM SYSIBM.SYSTABLEPART
WHERE STORTYPE='E' AND PARTITION=0
AND NOT DBNAME IN('DSNDB06','DSNDB07')
UNION
SELECT '-STOP DATABASE('||DBNAME||') SPACENAM(*)'
FROM SYSIBM.SYSTABLEPART
WHERE STORTYPE='E' AND PARTITION>0
UNION
SELECT '-STOP DATABASE('||B.DBNAME||') SPACENAM(*)'
FROM SYSIBM.SYSINDEXPART A,
     SYSIBM.SYSINDEXES B
WHERE NAME=IXNAME AND CREATOR=IXCREATOR
AND   STORTYPE='E' AND PARTITION=0
AND NOT B.DBNAME IN('DSNDB06','DSNDB07')
UNION
SELECT '-STOP DATABASE('||B.DBNAME||') SPACENAM(*)'
FROM SYSIBM.SYSINDEXPART A,
     SYSIBM.SYSINDEXES B
WHERE NAME=IXNAME AND CREATOR=IXCREATOR
AND   STORTYPE='E' AND PARTITION>0
```

    b. Change all these table spaces and indexes to use the new high-level qualifier. Use the following query to generate the DB2 ALTER statements:

```
SELECT
'ALTER TABLESPACE '||DBNAME||'.'||TSNAME||' USING VCAT newvcat;'
FROM SYSIBM.SYSTABLEPART
WHERE STORTYPE='E' AND PARTITION=0
AND DBNAME NOT IN ('DSNDB06','DSNDB07')
UNION
SELECT
'ALTER TABLESPACE '||DBNAME||'.'||TSNAME||' PART '||DIGITS(PARTITION)||
' USING VCAT newvcat;'
FROM SYSIBM.SYSTABLEPART
WHERE STORTYPE='E' AND PARTITION>0
UNION
```

```
SELECT
'ALTER INDEX '||IXCREATOR||'.'||IXNAME||' USING VCAT newvcat;'
FROM SYSIBM.SYSINDEXPART A,
     SYSIBM.SYSINDEXES B
WHERE NAME=IXNAME AND CREATOR=IXCREATOR
AND   STORTYPE='E' AND PARTITION=0
AND   DBNAME NOT IN ('DSNDB06','DSNDB07')
UNION
SELECT
'ALTER INDEX '||IXCREATOR||'.'||IXNAME||' PART '||DIGITS(PARTITION)||
' USING VCAT newvcat;'
FROM SYSIBM.SYSINDEXPART A,
     SYSIBM.SYSINDEXES B
WHERE NAME=IXNAME AND CREATOR=IXCREATOR
AND   STORTYPE='E' AND PARTITION>0
```

You can code a similar query to generate the VSAM ALTER commands to rename the VSAM data sets.

c. Start all table spaces and indexes once again.

You can modify the output from the first query to start all objects that were stopped in step 12a.

13. To change the DB2 VCAT used by the DB2 STOGROUP defined table spaces and indexes:

a. Create new DB2 STOGROUPs using the new VCAT name.

b. Stop all DB2 STOGROUP defined table spaces and indexes. Use the following query to identify these objects:

```
SELECT '-STOP DATABASE('||DBNAME||') SPACENAM(*)'
FROM SYSIBM.SYSTABLEPART
WHERE STORTYPE='I' AND PARTITION=0
AND NOT DBNAME IN('DSNDB06','DSNDB07')
UNION
SELECT '-STOP DATABASE('||DBNAME||') SPACENAM(*)'
FROM SYSIBM.SYSTABLEPART
WHERE STORTYPE='I' AND PARTITION>0
UNION
SELECT '-STOP DATABASE('||B.DBNAME||') SPACENAM(*)'
FROM SYSIBM.SYSINDEXPART A,
     SYSIBM.SYSINDEXES B
WHERE NAME=IXNAME AND CREATOR=IXCREATOR
AND   STORTYPE='I' AND PARTITION=0
AND NOT DBNAME IN('DSNDB06','DSNDB07')
UNION
SELECT '-STOP DATABASE('||B.DBNAME||') SPACENAM(*)'
FROM SYSIBM.SYSINDEXPART A,
```

```
     SYSIBM.SYSINDEXES B
WHERE NAME=IXNAME AND CREATOR=IXCREATOR
AND   STORTYPE='I' AND PARTITION>0
```

c. Change all these table spaces and indexes to use the new DB2 STOGROUPs. You can use the following query to generate the DB2 ALTER statements:

```
SELECT
'ALTER TABLESPACE '||DBNAME||'.'||TSNAME||' USING STOGROUP newstgrp;'
FROM SYSIBM.SYSTABLEPART
WHERE STORTYPE='I' AND PARTITION=0
AND DBNAME NOT IN ('DSNDB06','DSNDB07')
UNION
SELECT
'ALTER TABLESPACE '||DBNAME||'.'||TSNAME||' PART '||
DIGITS(PARTITION)||
' USING STOGROUP newstgrp;'
FROM SYSIBM.SYSTABLEPART
WHERE STORTYPE='I' AND PARTITION>0
UNION
SELECT
'ALTER INDEX '||IXCREATOR||'.'||IXNAME||' USING STOGROUP newstgrp;'
FROM SYSIBM.SYSINDEXPART A,
     SYSIBM.SYSINDEXES B
WHERE NAME=IXNAME AND CREATOR=IXCREATOR
AND   STORTYPE='I' AND PARTITION=0
AND   DBNAME NOT IN ('DSNDB06','DSNDB07')
UNION
SELECT
'ALTER INDEX '||IXCREATOR||'.'||IXNAME||' PART '||
DIGITS(PARTITION)||
' USING STOGROUP newstgrp;'
FROM SYSIBM.SYSINDEXPART A,
     SYSIBM.SYSINDEXES B
WHERE NAME=IXNAME AND CREATOR=IXCREATOR
AND   STORTYPE='I' AND PARTITION>0
```

d. Start all table spaces and indexes once again.

You can modify the output from the first query to start all the objects that were stopped in step 13b.

Use this procedure once again to migrate all of the DB2 objects back to the old DB2 STOGROUP names if you wish.

14. Drop all of the old DB2 STOGROUPs. This is a good test to see whether all objects are using the new VCAT.

Use the following query to generate the appropriate DB2 DROP STOGROUP statements:

```
SELECT
'DROP STOGROUP '||NAME||';'
FROM SYSIBM.SYSSTOGROUP
WHERE NAME NOT IN ('newstgrp')
```

15. Run the DB2 installation verification procedure (IVP) to check all is OK.

16. Reassemble any CICS and IMS attachments for the new DB2 subsystem names.

17. If you have changed the DB2 location and/or VTAM LUNAME that this DB2 subsystem uses, change the communication definitions in all the application requesters that communicate with this DB2 subsystem.

18. Issue an ARCHIVE LOG command to cause the last remaining active log data sets to become ″REUSABLE″.

19. Quiesce the DB2 subsystem once again.

20. Verify the DB2 subsystem is quiesced normally leaving no outstanding work as before.

21. Run the Print Log Map utility (DSNJU004) again to verify the last remaining active log data sets are ″REUSABLE″.

22. Delete and define the last remaining DB2 active log data sets found with a status of ″REUSABLE″ in the Print Log Map (DSNJU004) output.

23. Update the BSDS to define the new log data set names.

```
//DSNTLOG EXEC PGM=DSNJU003
//STEPLIB  DD  DSN=SYS1.DSNLOAD,DISP=SHR
//SYSUT1   DD  DISP=OLD,DSN=DSNDSGA1.DBA1.BSDS01
//SYSUT2   DD  DISP=OLD,DSN=DSNDSGA2.DBA1.BSDS02
//SYSPRINT DD  SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//SYSIN    DD  *
    DELETE DSNAME=VSDB2PD1.LOGCOPY1.DS01
    DELETE DSNAME=VSDB2PD2.LOGCOPY2.DS01
    NEWLOG DSNAME=DSNDSGA1.LOGCOPY1.DS01,COPY1
    NEWLOG DSNAME=DSNDSGA2.LOGCOPY2.DS01,COPY2
//*
//LIST    EXEC PGM=DSNJU004
//STEPLIB  DD  DSN=SYS1.DSNLOAD,DISP=SHR
//SYSUT1   DD  DISP=OLD,DSN=DSNDSGA1.DBA1.BSDS01
//SYSUT2   DD  DISP=OLD,DSN=DSNDSGA2.DBA1.BSDS02
//SYSPRINT DD  SYSOUT=*
//SYSUDUMP DD  SYSOUT=*
```

24. Restart DB2 and make it available for general use.

25. Alter all application procedures and batch jobs to use the new naming convention and DB2 subsystem names.

26. Update the input member for the DSNTINST CLIST. Otherwise, the next time you run this CLIST, it will generate all of the jobs using the old naming convention.

27. Alter any automated procedures to use the new DB2 subsystem names and command prefixes.

4.2 Implementation Checklists

The checklists in this section will help you to remember all of the activities that have to take place before (Table 4), during (Table 5 on page 62), and after (Table 6 on page 62) the implementation of one-way data sharing.

The activities are listed in the checklists in the sequence in which we believe they should occur; however, the sequence does not imply any dependencies. Some of the activities may be applicable at your site.

Table 4 (Page 1 of 4). Activities before Implementation of One-Way Data Sharing

Activity	**Document Reference or Command**
Understand your company's business objectives for implementing data sharing to ensure that your project meets them	
Draft a workload distribution plan that meets the business objectives. For example, if the objective is to reduce the workload on an existing processor, identify which workload you are targeting to move off the processor, where it is going to be moved, and how.	
Check all prerequisite hardware installed.	• Program Directory • *System/390 MVS Sysplex Hardware and Software Migration* • *IBM DATABASE 2 for MVS/ESA Version 4 Data Sharing: Planning and Administration*, Chapter 2 • *DB2 for MVS/ESA Version 4 Release Guide*, Appendix E

Table 4 (Page 2 of 4). Activities before Implementation of One-Way Data Sharing	
Activity	**Document Reference or Command**
Check all prerequisite software installed.	• Program Directory • *System/390 MVS Sysplex Hardware and Software Migration* • *IBM DATABASE 2 for MVS/ESA Version 4 Data Sharing: Planning and Administration*, Chapter 2 • *DB2 for MVS/ESA Version 4 Release Guide*, Appendix E
If you want to use the new eight-character format of the subsystem command recognition characters, now called the *command prefix*, wait until V4 is stable in your environment before implementing, to avoid an IPL on fallback.	
If you change your command prefix, make sure that the change is reflected in any automated operations tools you might have.	
Decide on group and member naming convention.	• *IBM DATABASE 2 for MVS/ESA Version 4 Data Sharing: Planning and Administration*, Chapter 3
Decide how often will you apply DB2 maintenance.	
Understand which aspects of application design affect data sharing overhead and take steps to change existing applications to ensure that the overhead is kept to a minimum.	• *DB2 for MVS/ESA Version 4 Data Sharing Performance Topics*
Understand which needs to be done to exploit data sharing once it is enabled (dynamic and static workload balancing using CICS or IMS).	• *DB2 for MVS/ESA Version 4 Data Sharing Performance Topics*
Once V4 is stable in your environment convert all indexes to Type 2. (Note that Type 1 indexes with subpages 1 are also supported in a data sharing environment. However, converting all of your indexes to Type 2 allows you to take advantage of a number of new V4 features not supported by Type 1 indexes.)	• *IBM DATABASE 2 for MVS/ESA Version 4 Data Sharing: Planning and Administration*, Appendix E
If you want distributed functions, decide on whether to use a group-generic or member-specific setup.	• *IBM DATABASE 2 for MVS/ESA Version 4 Data Sharing: Planning and Administration*, Chapter 4

Table 4 (Page 3 of 4). Activities before Implementation of One-Way Data Sharing	
Activity	**Document Reference or Command**
Review active log data set sizes and archiving strategy to ensure acceptable recovery times for your data. Recovery in a data sharing environment could potentially require more tape mounts if your archive logs are on tape and are needed for the recovery.	
Size coupling facility structures.	• *IBM DATABASE 2 for MVS/ESA Version 4 Data Sharing: Planning and Administration*, Chapter 3
Consider the placement of coupling facility structures. Placement depends on whether you have a duplexed coupling facility; If you do, you may want to put the GBP structures on one coupling facility and your lock and list structures on the other because of the dynamic rebuild capability of the lock and list structures.	
Ensure that the CFRM policy is set up correctly.	• *IBM DATABASE 2 for MVS/ESA Version 4 Data Sharing: Planning and Administration*, Chapter 3 • MVS/ESA Setting up a Sysplex
Ensure that SFM policy is set up to allow for dynamic rebuild of lock and list structures in the event of a coupling facility failure. Ensure that there is sufficient storage on the alternative coupling facility specified in the SFM policy.	• *IBM DATABASE 2 for MVS/ESA Version 4 Data Sharing: Planning and Administration*, Chapter 3 • MVS/ESA Setting up a Sysplex
Decide whether you should use the ARM. If so, set up required policies.	• *IBM DATABASE 2 for MVS/ESA Version 4 Data Sharing: Planning and Administration*, Chapter 3
If you plan to use generic resources, make sure that VTAM is set up to support generic resource definition.	• VTAM V4 R3 for MVS/ESA Network Implementation Guide
Set up required RACF definitions	• *IBM DATABASE 2 for MVS/ESA Version 4 Administration Guide*, Chapter 3
If you are altering the name of your DB2 subsystems, make the necessary SSN changes.	

Table 4 (Page 4 of 4). Activities before Implementation of One-Way Data Sharing	
Activity	**Document Reference or Command**
If required, change the subsystem and/or IRLM names, either before implementing data sharing or as part of the upgrade itself.	• *IBM DATABASE 2 for MVS/ESA Version 4 Data Sharing: Planning and Administration*, Chapter 2
Update existing V3 and/or V4 disaster recovery procedures to incorporate data-sharing-specific tasks.	• *IBM DATABASE 2 for MVS/ESA Version 4 Data Sharing: Planning and Administration*, Chapter 5
Update standards, procedures, and guidelines to encompass new V4 and data sharing function.	
Ensure that the hardware configuration at your disaster recovery site can support a Parallel Sysplex configuration.	
Schedule and execute a disaster recovery exercise on a test data sharing group.	
Document a procedure to disable data sharing. As disabling data sharing requires you to cold start DB2, the procedure should incorporate a strategy to ensure recoverability of all user data.	• *IBM DATABASE 2 for MVS/ESA Version 4 Data Sharing: Planning and Administration*, Chapter 3
Test and time disablement of data sharing on a test data sharing group.	
Document a performance monitoring strategy for data sharing. Collect performance data throughout the project. Plan for performance measuring at the beginning of the project—What are you going to measure, and how? How are you going to measure data sharing overhead on applications?	• *IBM DATABASE 2 for MVS/ESA Version 4 Data Sharing: Planning and Administration*, Chapter 6
Make sure that your SSNs (DB2, CICS, IMS and VTAM) are defined on all MVS images. If a processor is unavailable, you will be able to bring up a member on another MVS image—which is especially important if the member is holding retained locks.	

Table 5. Activities during Implementation of Data Sharing	
Activity	**Document Reference or Command**
Make sure no objects have a pending status against them.	• DISPLAY DATABASE(*) SPACENAM(*) RESTRICT
Image copy critical table spaces.	
Check that there are no utilities outstanding.	• DISPLAY UTILITY(*)
Check that all catalog indexes are Type 2.	

Table 6. Activities after Implementation of One-Way Data Sharing	
Activity	**Document Reference or Command**
Start new DB2 subsystems as members of your new data sharing group.	
Monitor coupling facility use to ensure that structures have been sized correctly.	
Monitor coupling link use to ensure that it is not adversely impacting performance of the data sharing group.	
If you plan to start a DB2 member on a new system where no DB2 subsystems have been started before, review WLM policies.	
If your JCL still references an individual subsystem name, change it to reference the group name to facilitate workload balancing across the group.	
If you plan to merge another existing subsystem that contains user data into this data sharing group, check that there are no duplicate user object names in the two catalogs before doing the merge.	

Chapter 5. Global Locking and Buffer Pool Management

In this chapter we describe how locking and buffering are managed in a data sharing environment, so that you understand how much additional processing the management of these activities requires at a group level when data is shared.

5.1 Global Locking

Locking in a data sharing environment is now referred to as *global locking*. Conceptually all locks taken in a data sharing environment are global locks, that is, they are effective groupwide. All locks do not have to be propagated to the lock structure on the coupling facility, however. A number of optimizations are in place to limit the number of locks that get propagated to the lock structure.

All lock information that members in the group have to share is kept in the lock structure on the coupling facility. The number of locks that are propagated to the lock structure for a page set or partition is determined by the number of DB2 members interested in the page set and whether their interest is read or write. The lock structure consists of two parts, which work completely independently of each other:

- Lock table

  The lock table is sometimes also referred to as the *hash table*. The lock table is where members register their interest in a resource. The lock table is made up of many lock table entries, sometimes also referred to as *hash classes*. When a lock required by a member needs to be made known to other members in the group, it is registered in the lock table.

- MRL

  The IRLMs in the group use the modified resource list (MRL) to store run-time information about the modify locks owned by their DB2 members. The information stored here is used to create retained locks when a subsystem abends. Therefore, if a lock is to be preserved across a subsystem abend, it must be stored in the modified resource list.

DB2 data sharing uses two types of global locks:

- Physical locks (P-locks)
  - Page set physical locks are used to track the level of interest of the members in the group in a particular page set or partition.
  - Page physical locks are used in a similar way to latches. They are used to protect a page while the physical page structure is being modified.

– Other types of P-locks include DBD, castout, GBP structure, index tree, and repeatable read tracking.

- Logical locks (L-locks)

  – Also referred to as *transaction locks*, L-locks are used to serialize access to data to ensure data consistency.

P-locks and L-locks work independently of each other, although information about them is stored in common places. The lock table in the lock structure holds lock table entries for both P-locks and L-locks although the lock table entry for the P-locks held on a resource is hashed to a different lock table entry from the L-locks held on the same resource. The modified resource list also holds information about P-locks and L-locks.

5.1.1 P-Locks

There are two kinds of P-locks, page set and page. Page set P-locks are used to track inter-DB2 read-write interest, and the page P-locks are used to ensure the physical consistency of a page when it is modified.

5.1.1.1 Page Set Physical Locks: Page set P-locks are used to track inter-DB2 read-write interest, thereby determining when a page set has to become GBP-dependent.

When access is required to a page set or partition through a member in the data sharing group, a page set P-lock is taken. This lock is always propagated to the lock table on the coupling facility and is owned by the member. No matter how many times the resource is accessed through the member, there will always be only one page set P-lock for that resource for a particular member. This lock will have different modes depending on the level (read or write) of interest the member has in the resource.

The first member to acquire a page set P-lock on a resource takes out the most restrictive mode of lock possible, that is, an S page set P-lock for read or an X page set P-lock for write interest. An X page set P-lock indicates that the member is the only member with interest (read or write) in the resource. Once another member becomes interested in the resource, the page set P-lock mode can be negotiated, that is, it can be made less restrictive if the existing page set P-lock is incompatible with the new page set P-lock request. The negotiation always allows the new page set P-lock request to be granted, except when there is a retained X page set P-lock. A retained P-lock cannot be negotiated. Page set P-lock negotiation signifies the start of GBP dependence for the resource. Refer to 5.1.3, "Group Buffer Pool Dependency" on page 66 for a description of what happens when a page set or partition becomes GBP-dependent.

Although it may seem strange that a lock mode can be negotiated, remember that page set P-locks do not serialize access to a resource, they are used to track which members have interest in a resource and for determining when a resource must become GBP-dependent. Refer to 5.2.3, "Page Set Physical Lock Negotiation Scenario" on page 76 for a description of page set P-lock negotiation.

Page set P-locks are released when a page set or partition data set is closed. The mode of page set P-locks is downgraded from R/W to R/O when the page set or partition is not updated within an installation-specified time period or number of checkpoints. When page set P-locks are released or downgraded, GBP dependency is reevaluated.

5.1.1.2 Page Physical Locks: Page P-locks are used to ensure the physical consistency of a page across members of a group in much the same manner as latches do in a non-data-sharing environment. A page P-lock protects the page while the structure is being modified. Page P-locks are used not only when row locking is in effect, but also used in other ways, for example, when changes are being made to GBP-dependent space map pages.

Page P-locks can be negotiated. Refer to 5.2.4, "Page Physical Lock Negotiation Scenario" on page 76 for a description of page P-lock negotiation.

5.1.2 L-Locks

DB2 uses L-locks (also known as *transaction locks*) to ensure data consistency. L-locks are owned by a transaction, and the lock duration is controlled by the transaction. For example, the lock will be held from the time the application issues an update until the time it issues a commit. These locks are controlled locally per member by each member′s IRLM.

Within IRLM, a hierarchy now exists between certain types of L-locks, where a parent L-lock is the lock on a page set and a child L-lock is the lock held on either the table, data page, or row within that page set. Refer to "Explicit Hierarchical Locking" in the *IBM DATABASE 2 for MVS/ESA Version 4 Data Sharing: Planning and Administration* for additional details.

Explicit hierarchical locking has been introduced to reduce the number of locks that have to be propagated to the coupling facility. A parent L-lock is propagated only if it is more restrictive than the current state that XES knows about for this resource from this member. For example, if an IX lock has already been propagated to XES for this member, a subsequent IX lock does not have to be sent. It can be serviced locally. Parent L-locks are released either when the transaction commits or when the thread terminates, depending on the value you have specified for the RELEASE parameter on the bind. Child L-locks are propagated to the lock table only when there is inter-DB2 read-write interest for the page set.

IRLM propagates only the most restrictive locks held by the transactions executing through one member. For example, transactions A and B are running in one member:

- If transaction A has an IS L-lock, the IS L-lock gets propagated to the lock structure on the coupling facility. If transaction B has an IX L-lock, the IX L-lock gets propagated to the lock structure on the coupling facility.

- If transaction A has an IX L-lock, the IX L-lock gets propagated to the lock structure on the coupling facility. If transaction B has an IS L-lock, the IS L-lock does not get propagated to the lock structure on the coupling facility; that is, the resultant state does not change.
- If transaction A has an S L-lock, the S L-lock gets propagated to the lock structure on the coupling facility. If transaction B has an IX L-lock, the SIX L-lock gets propagated to the lock structure on the coupling facility.
- If transaction A has an IX L-lock, the IX L-lock gets propagated to the lock structure on the coupling facility. If transaction B has an S L-lock, the SIX L-lock gets propagated to the lock structure on the coupling facility.

5.1.3 Group Buffer Pool Dependency

When more than one member in a group accesses a page set or partition, and at least one of the members has write interest, there is said to be inter-DB2 read-write interest for the page set or partition. The page set or partition now becomes GBP-dependent.

When a page set or partition becomes GBP-dependent, all changed pages in the local buffer pool are moved, synchronously, into the GBP. All pages, clean and changed, are registered in the directory in the GBP.

GBP dependency is tracked through the use of page set P-locks. Refer to 5.1.1, "P-Locks" on page 64 for a description of when page set P-locks are acquired and released, as it is at this time that GBP dependency is reevaluated.

The pages of GBP-dependent page sets and partitions require extra administration to ensure that each DB2 member in the group that is using the GBP-dependent page set or partition has valid copies of the pages in their local buffer pools. Refer to 5.3, "Group Buffer Pools" on page 79 for a description of these administrative functions.

Locking and contention are now managed at a group level for GBP-dependent page sets and partitions. Refer to 5.1, "Global Locking" on page 63 for a description of locking in a data sharing environment.

A member's interest goes from none to R/O at physical open, R/O to R/W at pseudo open, R/W to R/O at pseudo close, and any to none at physical close. Page sets and partitions go in and out of GBP dependency through the P-locking mechanism (P-lock negotiation through the P-lock exit).

Table 7 on page 67 shows you how to determine when your page set is GBP-dependent.

Table 7. Determining Group Buffer Pool Dependency

<table>
<tr><th>My DB2′s Interest</th><th>Other DB2′s Interest</th><th>Is Page Set GBP-Dependent?</th></tr>
<tr><td>R/O</td><td>None, R/O</td><td>No</td></tr>
<tr><td>R/O</td><td>R/W</td><td>Yes</td></tr>
<tr><td>R/W</td><td>None</td><td>No</td></tr>
<tr><td colspan="3">Exception: If the other DB2 previously had R/O interest and then physically closed the page set, the page set remains GBP-dependent even though there is no inter-DB2 read-write interest on the page set or partition. It remains GBP-dependent until the last updating DB2 switches the page set or partition to read-only.</td></tr>
<tr><td>R/W</td><td>R/O</td><td>Yes</td></tr>
<tr><td>R/W</td><td>R/W</td><td>Yes</td></tr>
</table>

Beware of the exception condition mentioned in Table 7. Do not automatically assume that because only one member is accessing a page set that it is not still marked as being GBP-dependent. Pay special attention, when your processing moves from the online day into batch, that batch jobs do not suffer the overhead of the additional processing required for a GBP-dependent page set when in fact the page set is only being accessed through one member.

5.2 Locking Components

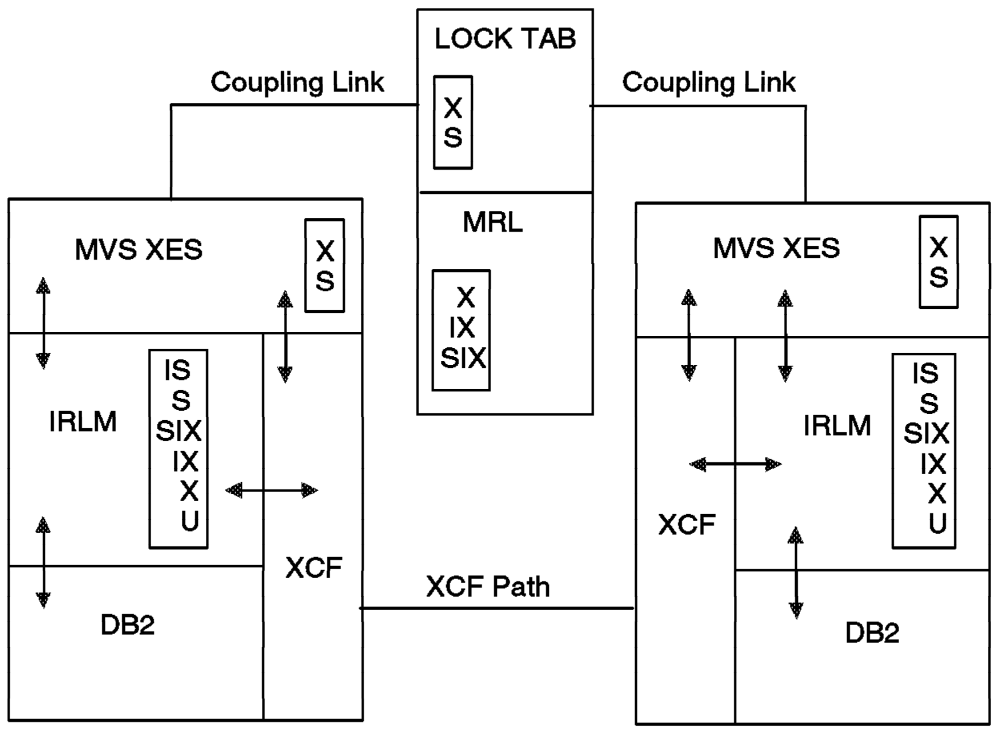
Figure 12 on page 68 shows the components used for locking in a data sharing environment. Lock information is held in three different components:

- IRLM
- SLM component of XES
- Lock structure on the coupling facility
  - Lock table
  - Modified resource list (MRL)

From here on we refer to the SLM component of XES as simply XES to be consistent with other publications.

When the DISPLAY DATABASE LOCKS command is used, the DB2 member through which the command was issued asks its IRLM about the locks that it knows about and then, using XCF, asks the other DB2s about theirs. The other DB2s in turn ask their respective IRLMs and relay the information back to the requesting DB2.

Retained lock information for an inactive member will also be displayed. The retained lock information of inactive members is stored locally in each active member′s IRLM. The technique used by the DISPLAY DATABASE LOCKS command ensures that all locks on a resource can be seen regardless of which member the display command was issued through.



Note: Diagram shows lock granularity supported by each component.

Figure 12. Locking Components in a Data Sharing Environment

The types of lock granularity supported by each component differ. IRLM contains the most detailed lock information whereas XES and the lock table on the coupling facility recognize only two types of locks—S and X. The modified resource list is used by the IRLMs in the group to store run-time information about the modify locks owned by their DB2 member. These locks are stored here regardless of whether the resource is GBP-dependent or not. A modify lock is an active X-type (X,IX,SIX) P-lock or L-lock. The modified resource list information is used to create retained locks in the event that a data sharing group member ends abnormally.

XES is an important locking component in a data sharing environment. Information about locks that IRLM has passed to XES is stored in XES. When contention occurs (false, XES, or real contention), one of the XESs is assigned to be the global lock manager to resolve the contention. This resolution involves all of the other XESs in the group that have locks which have been assigned to the lock table entry, passing their lock information to the global lock manager XES. This global lock manager XES can then drive resolution of the contention. Refer to 5.2.2, “Managing Contention” on page 72 for a detailed description of how locks are stored in the lock table and how XES manages contention.

XCF is a component of MVS that provides communication services used in a data sharing environment. For example, IRLMs use XCF to communicate with each other.

5.2.1 Types of Contention

In a data sharing environment, you can have three types of contention: real, false, and XES.

XES is used to resolve all three types of contention. False contention requires the least time to resolve.

For in-depth descriptions and examples of how locking works in a data sharing environment, refer to Chapter 4, “Locking in a Data Sharing Group,” in the *DB2 for MVS/ESA Version 4 Data Sharing Performance Topics* redbook.

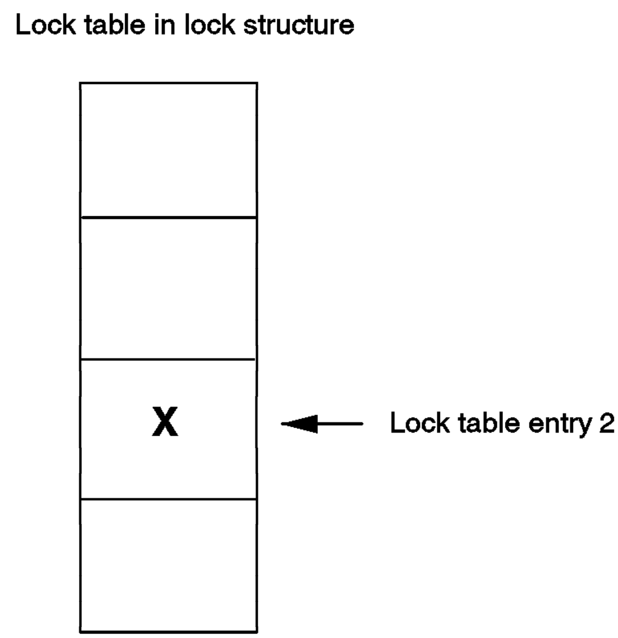
5.2.1.1 Real Contention: Real contention is caused by normal IRLM lock incompatibility between two members. For example, two transactions may try to update the same resource at the same time. Real contention requires the most time to resolve.

DB2 PM reports real contentions as IRLM contentions.

5.2.1.2 False Contention: An individual DB2 member calculates a hash value, which is used to calculate to which lock table entry a resource will be assigned in the lock table on the coupling facility. This hash value is calculated by using the name and other relevant information that uniquely define the resource to be locked. The hash value is a 32-bit value passed through IRLM to XES. XES uses a module for the hash value based on the maximum number of lock table entries available to assign the resource to a particular lock table entry. This assignment registers a members′ interest in a lock table entry.

Each lock table entry should represent only one resource. However, false contention occurs when the same lock table entry is used to represent more than one resource.

Figure 13 on page 70 shows a lock table that contains a maximum of four lock table entries. Therefore XES uses four modules to convert the hash value to a lock table entry position. Note that in this example, if the hashing algorithm results in a resource name mapping to hash value 6, which is a value outside the range of available lock table entries, XES will ″wraparound″ (using module 4), and the resource will be mapped to lock table entry 2.



Note: Lock table entries are positional. In this example there are four entries, 0, 1, 2 & 3.

Figure 13. Lock Table

Therefore the fewer lock table entries there are in your lock table, the more likely it is that false contention will occur because this wrap-around effect will occur more frequently.

Once potential contention has been identified, an XES is nominated as the global lock manager. The other XESs in the group are instructed to send the lock information that they hold pertaining to this lock table entry to the global lock manager XES so that it can determine whether the contention is real or false. Resolution is possible as each XES holds information about locks that its IRLM has passed to it. This information includes the resource name (RN), hash value (HV), and lock table entry (LTE) position, as shown in Figure 14 on page 71.

Two different resources have been assigned to the same LTE, so this is an example of false contention.

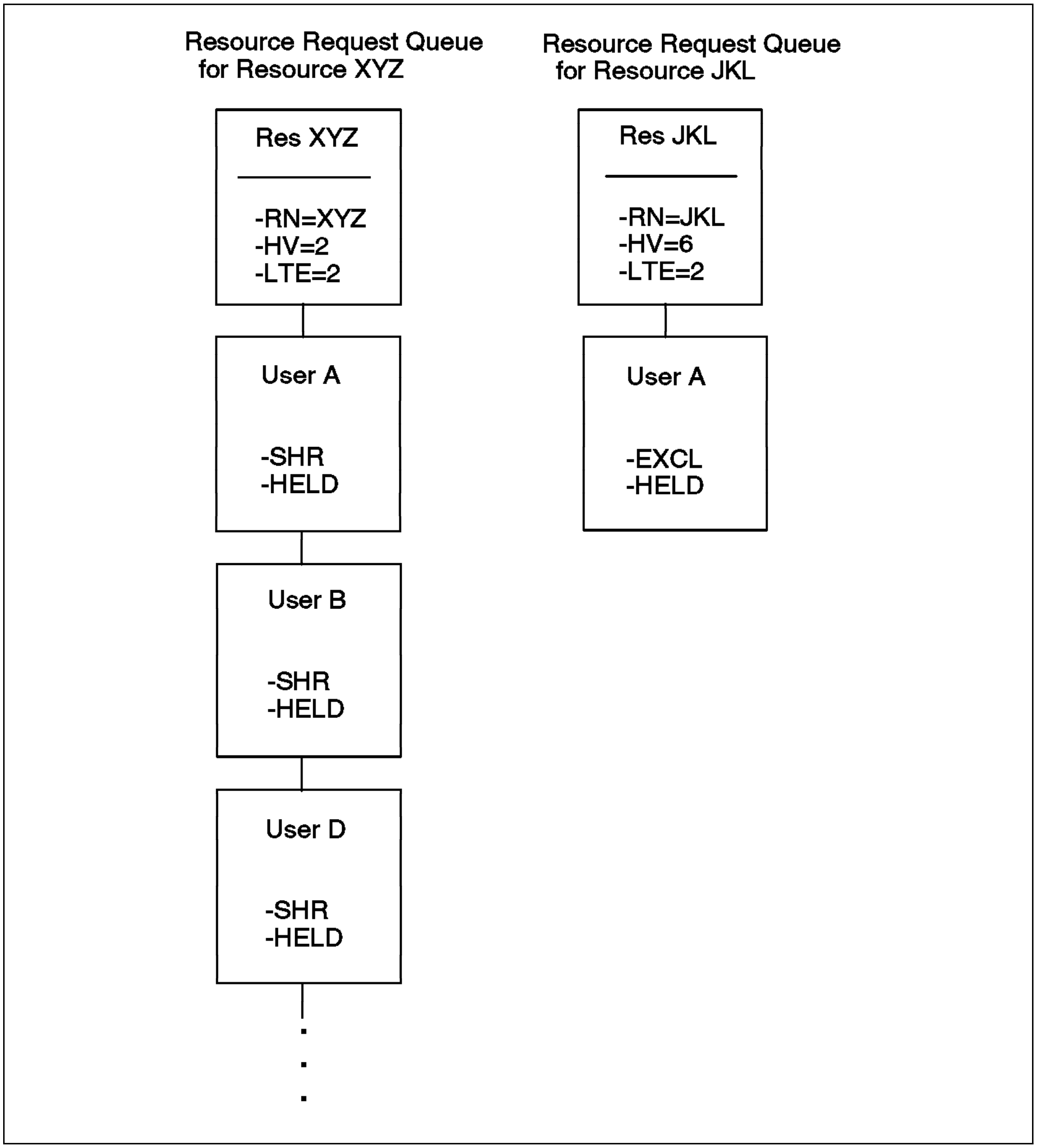


Figure 14. Lock Information Stored by XES

The global lock manager XES gets information from all of the other XESs in the group that have locks on the resource and can determine if the contention is false by checking to see whether different resources have been assigned to the same lock table entry. Figure 14 is an example of false contention because two different resources have been assigned to the same lock table entry.

The number of lock table entries available is determined by the size of the lock table and the size of the lock table entries. The lock table is allocated half of the space that has been allocated to the lock structure. The size of the lock table entries is determined by how many members there are in the group. Initially the number of members in the group is determined by the MAXUSRS parameter of the first IRLM to join the group. However, if this parameter value is smaller than the actual number of group members, IRLM automatically rebuilds the structure as a new member joins the group.

Make sure that the MAXUSRS parameter value is not too high. A high value means that fewer lock table entries are available, thereby potentially increasing the instances of false contention. For additional details, refer to Chapter 6, "Performance Monitoring and Tuning," in the *IBM DATABASE 2 for MVS/ESA Version 4 Data Sharing: Planning and Administration*.

5.2.1.3 XES Contention: If XES determines that false contention does not exist, it has to go and check for XES contention. As you can see in Figure 12 on page 68, the lock table on the coupling facility and XES recognize only two types of lock modes: X (exclusive) mode and S (share) mode. IRLM supports many more granular lock modes. This causes the lock mode to be abbreviated when it gets passed from IRLM to XES. For example, an IX lock in IRLM is passed to XES as an X lock, as are SIX and X locks.

If an XES tries to register an X-mode lock and X mode interest is already registered by another member, it has to check with the IRLMs to determine exactly the types of locks they require, as it could turn out that all of the IRLMs need IX locks, which are actually compatible. XES passes control to the IRLM contention exit to resolve the contention. The exit checks the actual mode of each IRLM lock. If the contention is not real, it is called *XES contention* and the requested lock can be granted.

5.2.2 Managing Contention

Lock contention in a data sharing environment is monitored through the use of the lock table in the coupling facility and through information stored locally in the XES components.

Let us look at how contention is managed in a data sharing environment by stepping through what happens to a lock request that is sent to XES from IRLM (Figure 15 on page 73) and how a contention is resolved (Figure 16 on page 74).

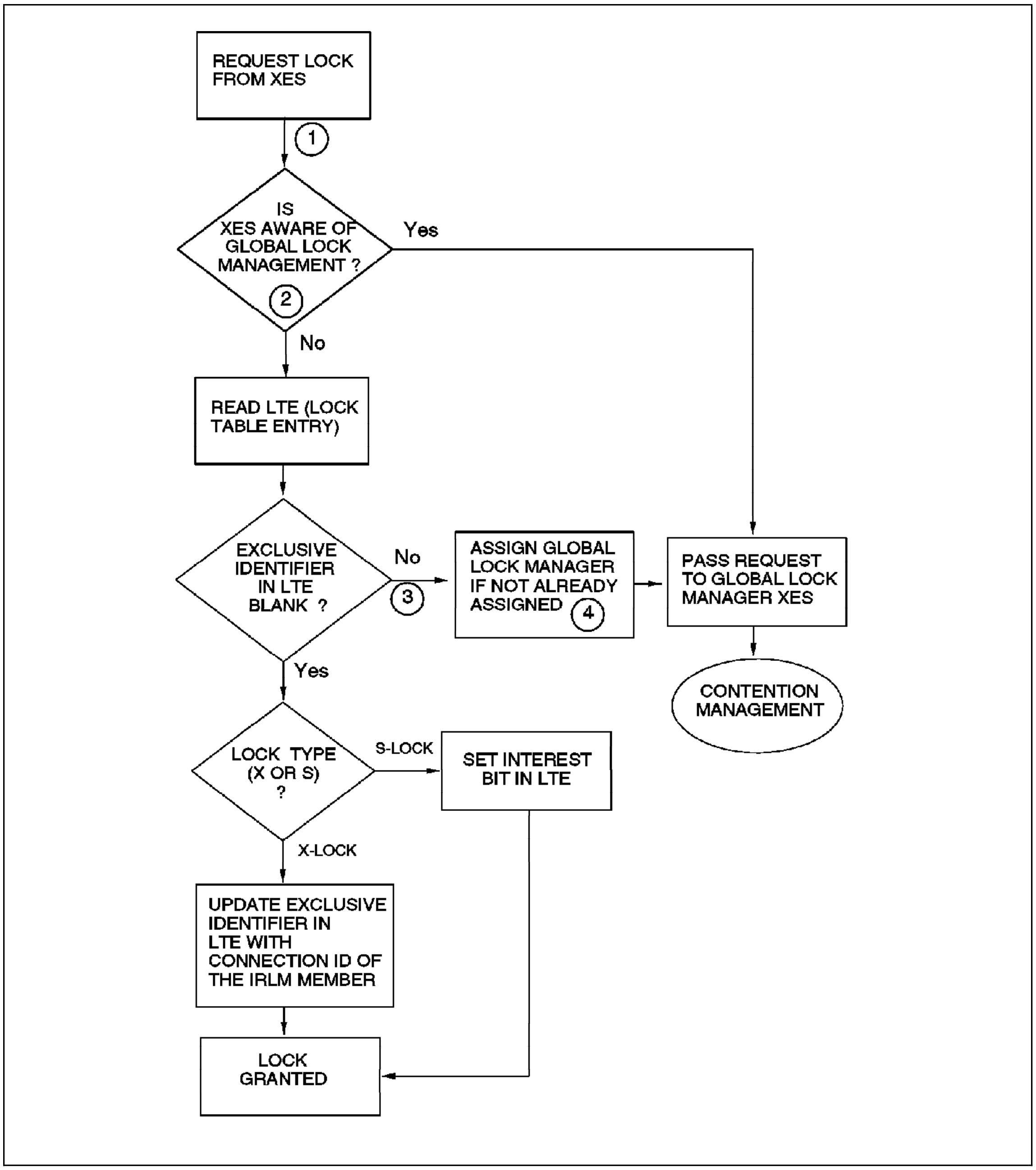


Figure 15. What Happens When a Lock Request Is Sent to XES

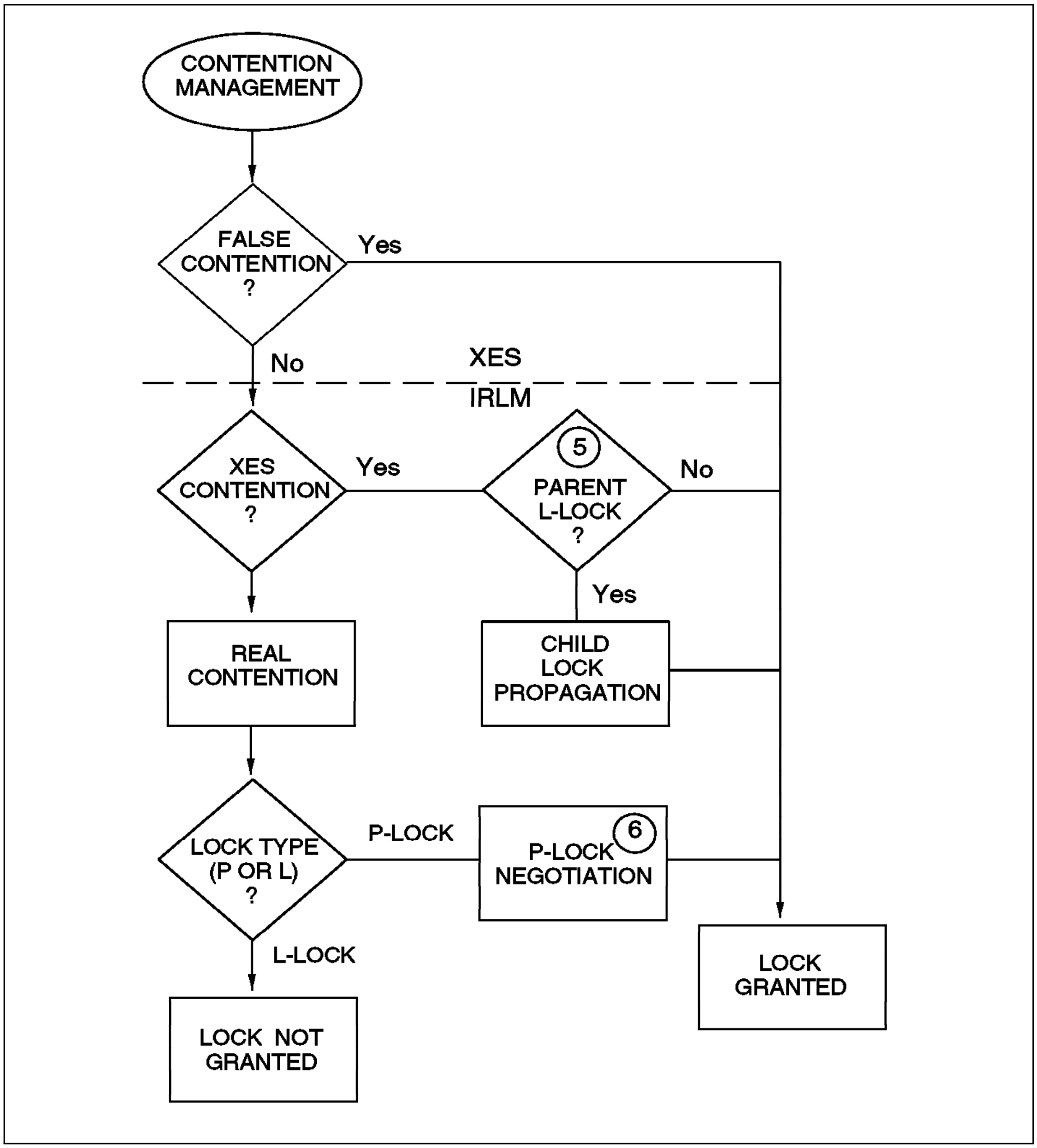


Figure 16. Contention Resolution in a Data Sharing Environment

Here is what happens when a lock request is sent to XES and XES contention is detected (refer to Figure 15 on page 73 and Figure 16):

1. When a lock is needed on a resource:

- DB2 calculates a hash value for the resource to be locked. This 32-bit hash value is calculated using information that uniquely defines the resource to be locked.
- The hash value and the resource name are passed through IRLM to XES.

2. When the request gets to XES:
   - XES uses a modulus of the hash value based on the maximum number of lock table entries to calculate to which lock table entry the resource is to be assigned.
   - If global lock management has already been initiated for the lock table entry, this member′s XES component will be aware of the fact if it had an interest in the lock table entry at the time global lock management was initiated. In this case global lock management is immediately invoked and there is no communication with the lock table.
3. If there is a member registered as having X mode interest, global lock management is initiated.
4. Assigning the global lock manager:
   - The XES component of the member that was registered in the lock table as having X mode interest first in the lock table entry is appointed as the global lock manager.
   - When global lock management is initiated, all XESs that have an interest in the lock table entry are notified to send their information pertaining to the lock table entry to the global lock manager XES.
   - All XESs that have an interest in the lock table entry store the fact that global lock management is now in place for the lock table entry and store which XES is the global lock manager. Any further lock requests for the lock table entry do not go to the lock table, they are sent directly to the global lock manager XES.
   - New lock requests coming from members that did not have any interest registered for the lock table entry at the time global lock management was initiated will not be aware that global lock management is in place and will go to the lock table before communicating with the global lock manager XES.
5. If the lock is a parent L-lock, IRLM will trigger child lock propagation to the lock structure for child L-locks below the parent L-lock.
6. If the lock is a page set P-Lock, the negotiation process marks the start of GBP dependence for the page set or partition. See 5.2.3, “Page Set Physical Lock Negotiation Scenario” on page 76 for a page set P-Lock negotiation scenario.

Remember that once global lock management has been initiated for a lock table entry, the information stored in the entry is no longer sufficient to grant a lock. Therefore, the only purpose the lock table entry serves at this time is to indicate that global lock management is in place for the lock table entry. All lock requests are directed to the global lock manager XES to resolve using the information that it maintains about the resources hashed to the lock table entry.

Once global lock management is established, it is not deescalated until the last X-type lock is released for that lock table entry. After deescalation, members again start using the lock table to register their interest.

5.2.3 Page Set Physical Lock Negotiation Scenario

Page set P-locks are used to track inter-DB2 interest in a resource. Knowing which members in the group are using the resource and what their level of interest is (read or write) determines whether or not a page set is to be GBP-dependent.

When DB2A opens a page set, an S mode page set P-lock is propagated to the lock table. When DB2B opens the same page set, it also has an S mode page set P-lock propagated to the same lock table entry on the lock table. These S-mode locks are compatible and indicate that the members have, as yet, no write (update) interest in the page set. Therefore the page set does not have to be GBP-dependent. As soon as DB2A wants to update the data, it has to propagate an X mode P-lock to the lock table. This mode of lock is incompatible with the S mode lock of DB2B, and this incompatibility is what triggers negotiation for page set P-locks.

The global lock manager XES cannot resolve the contention and so control is passed back to IRLM. DB2A′s IRLM, IRLMA, cannot grant the X mode lock, so it has to communicate using XCF with IRLMB. This communication will drive IRLMB′s contention exit, which will grant the lock and make the page set GBP-dependent.

5.2.4 Page Physical Lock Negotiation Scenario

In this page P-lock negotiation scenario, row locking is used.

The page set is GBP-dependent, and DB2A is updating row R1 on page P.

1. DB2A obtains an X mode page physical lock to ensure the integrity of the page.
2. The page P-lock is propagated to the coupling facility.
3. DB2A obtains an X mode L-lock for row R1.
4. The L-lock, which is obtained before the P-lock, is also propagated to the coupling facility.
5. DB2A changes the row on page P to R2.
6. DB2A creates page P2.

Now DB2B needs page P to update another row, S1. DB2A has not committed its update yet. Here are the steps to provide data integrity:

1. DB2B requests an X mode P-lock on page P. That lock cannot be granted as it is already held by DB2A. So the IRLMs of the two members negotiate for the P-lock. DB2A has to give up its page P-lock for page P.

2. To give it up, DB2A has to write the changed page P to the GBP and release the X mode P lock on page P. The log is forced to the page log RBA value before writing the page to GBP or disk.
3. When DB2A writes the changed page P2 to the GBP, the local copy of page P in DB2B′s local buffer pool will be invalidated.
4. The page in DB2A′s local buffer pool is still valid even after the X mode page P-lock is released by DB2A. It will remain valid until DB2B writes a new version of the page to the GBP.
5. DB2A keeps the X mode L-lock on row R2.
6. Now DB2B owns the X mode page P-lock on page P2.
7. DB2B obtains the X mode L-lock for row S1.
8. DB2B reads page P2 from the GBP.
9. DB2B updates row S1 to row S2 and creates page P3.
10. DB2B writes updated page P3 to the GBP. This write causes DB2A′s version of the page, P2, to be invalidated.

If DB2A issues a commit (remember DB2B got the page before DB2A had committed its changes to row R), it does not need the page.

However, what happens if DB2A performs a rollback? It has to renegotiate for the page P-lock so that it can change the row from R2 back to R1. Here is how it works:

1. DB2A negotiates for the page P-lock.
2. DB2B gives up the X mode page P-lock.
3. DB2A acquires the X mode page P-lock on page P3.
4. DB2A reads page P3 in from the GBP.
5. DB2A rolls back the row to R1 on page P3 and creates page P4.
6. DB2A writes page P4 to the GBP.
7. The page in DB2B′s local buffer pool is invalidated.
8. DB2A releases its L-lock on row R.

As you can see from this scenario, while all logical locking is done at the row level, the physical locking is done at the page level. Be careful when using row locking in a data sharing environment. When there is activity as described in the scenario above, where different members are updating rows on the same page, the amount of work that negotiating page P-locks, moving pages into the GBP, and invalidating copies of pages the local buffer pools can generate may have an adverse impact on performance.

5.2.5 Retained Locks

The IRLMs in the group use the MRL to store run-time information about the modify locks owned by their DB2 member. These locks are stored here regardless of whether the resource is GBP-dependent. A modify lock is an active X-type (X,IX,SIX) P-lock or L-lock. At subsystem abend time, this information is used to create retained locks.

Not all modify locks are stored in the MRL. IRLM uses explicit hierarchical locking techniques to determine which modify L-locks are propagated to the MRL. Refer to 5.1.2, “L-Locks” on page 65 for a description of explicit hierarchical locking. Child L-locks will not be propagated to the MRL until there is inter-DB2 read-write interest on the page set. Therefore at subsystem abend time, a page set IX L-lock could be converted to an X retained L-lock. This happens when X child L-locks existed on pages within the page set but had not been propagated to the MRL. The page set IX L-lock therefore has to be upgraded to an X retained page set L-lock to protect the integrity of the pages within the page set. However, it should be very rare that the parent IX lock gets retained as an X lock because IRLM, during its shutdown processing, propagates any child modify locks that have not yet been propagated before terminating.

At subsystem abend time any retained locks that have been created are propagated to the other IRLMs in the group. Thus, in the event that the lock structure is lost before the failed IRLM is restarted, retained lock information is available for the dynamic rebuild of the lock structure.

We therefore recommend that a coupling facility not be kept in an LPAR on the same machine as any of the group members. A machinewide failure would mean that an IRLM and a lock structure would be lost simultaneously, and the dynamic lock structure rebuild would fail, as there would be no way of obtaining the retained lock information for the failed member. A group restart would then be required to rebuild the lock structure.

Remember that both P-locks and L-locks can become retained locks.

The RETLWAIT parameter in DSNZPARM specifies to DB2 how to manage incompatible lock requests made against resources that are protected by retained locks. RETLWAIT=NO (the default) indicates that the lock request will be immediately rejected and a ″resource unavailable″ condition returned to the application. Specifying RETLWAIT=YES causes DB2 to suspend the application for the normal timeout period to wait for the lock to become available. If the request times out, a ″timeout″ condition is returned to the application.

Consider using RETLWAIT=YES if you have automatic restart or some other restart automation that quickly restarts failed DB2s.

5.2.6 Lock Avoidance in a Data Sharing Environment

A Global Commit Log Sequence Number (GCLSN) has been introduced so that lock avoidance in a data sharing environment can be managed at a group level.

In DB2 V4, for non-GBP-dependent page sets or partitions, each member maintains a local CLSN per page set or partition, which is the lowest uncommitted LRSN for all active transactions associated with that page set or partition.

For GBP-dependent page sets and partitions, a single GCLSN value is maintained and used for lock avoidance at a group level. The single GCLSN value is derived from each member′s CLSN values. The CLSN value provided by the individual members to create the GCLSN is maintained at **the DB2 subsystem level, NOT at the page set or partition level**. The CLSN value provided by the individual members to create the GCLSN is the LRSN taken from the beginning of the oldest current unit of recovery (UR) per DB2 member.

The GCLSN value will not be affected by a read-only member. For a read-only member that does not have any UR, the current LRSN value is used (that is, derived from the current STCK value) as the member′s CLSN value. The value is updated on a timer basis.

Because lock avoidance for GBP-dependent page sets and partitions does not use a CLSN value per page set or partition, more care has to be taken to ensure effective lock avoidance for GBP-dependent page sets and partitions. For example, a long-running batch job that does not issue frequent commits will cause fewer pages to qualify for lock avoidance as the CLSN passed by the member through which the job is running will be relatively old.

5.3 Group Buffer Pools

GBPs are cache structures in the coupling facility. Every local buffer pool (a local buffer pool includes a virtual buffer pool and hiperpool) needs a corresponding GBP if page sets or partitions that use the local buffer pool are going to be shared among members in the group.

Pages are cached into the GBP when:

- Two or more members in the group are sharing the same page set or partition and at least one of the members has write interest. When this happens, the page set or partition becomes GBP-dependent and changed pages are cached into the GBP.
- The GBPCACHE ALL parameter is used for a table space or index. This parameter causes clean (unchanged) pages to be cached in the GBP when they are read into the local buffer pool from DASD by the members in the group.

A number of activities have to take place to administer shared pages in a data sharing environment:

- Test for page validity
- Page registration and deregistration
- Page refresh from GBP
- Force at commit processing
- Cross-system invalidation

There are also a number of activities associated with GBP administration:

- Getpage
- GBP checkpoint
- GBP castout.

In this section we discuss each of these activities in detail.

5.3.1 Test for Page Validity

Before a page can be used in a local buffer pool it must be tested for validity as it could have been invalidated by the cross-system invalidation mechanism (see 5.3.5, "Cross-System Invalidation" on page 81). If a page is found to be invalid, a new copy of the page must be brought into the local buffer pool from the GBP or DASD.

5.3.2 Page Registration and Deregistration

The page directory in the GBP records the location of pages in a data sharing environment for GBP-dependent page sets and partitions and for page sets and partitions that have the GBPCACHE ALL option.

For GBP-dependent page sets and partitions, both dirty (changed) and clean pages are registered. (Only dirty pages are cached in the GBP.)

Page directory entries are used to check whether a copy of a page exists in the GBP. They are also used to determine which members are to be sent cross-system invalidation messages.

If a copy of page P1 is stored in member 1′s local buffer pool, member 2′s local buffer pool, and the GBP, only one directory entry is used in the page directory to track the three copies of the same page. The theoretical limit of directory entries would therefore be the sum of all members′ virtual buffer pools, hiperpools, and GBP data entries for that GBP, assuming that every buffer has a different data page.

Whenever a DB2 member reads a new page into the local virtual buffer pool from the GBP or DASD for a GBP-dependent page set, the interest in that page for that member must be registered in the page directory. This is also true for pages of GBP-dependent page sets being moved into and out of hiperpools.

By making use of the new ″Register-Name-List″ request function, the DB2 prefetch code can register a set of up to 32 pages with a single coupling facility interaction. This is available if the coupling facility where the GBP resides is at CFLEVEL=2. If the function is not available, a separate ″read-and-register″ for each page is used instead. For CFLEVEL=2 support, your ESA coupling facility must have APARS OW15587 and OW15590 or equivalent MVS maintenance applied.

When a buffer is stolen from a local buffer pool that currently contains a page from a GBP-dependent page set or partition, the page is deregistered in the GBP. If the buffer is stolen for a page of another GBP-dependent page set or partition, one call is made to deregister the old page and register the new page in the GBP.

Pages must also be deregistered when an existing directory entry must be reclaimed to handle new work. Even though the data page has not actually changed, it is still marked as invalid everywhere there is a copy, as it is no longer registered in the page directory. This invalidation requires that the page be refreshed when next needed from DASD, and the refresh can degrade the performance of the system.

When there is no more inter-DB2 read-write interest in the page set, the page set becomes non-GBP-dependednt. The pages are deregistered from the directory. Dirty pages are cast out to DASD and purged from the GBP. The directory entry is now available to be reused.

5.3.3 Page Refresh from Group Buffer Pool

When a member registers a page in the page directory, a check is made to see whether the page is registered in the GBP. If the page is registered in the GBP, it will be transferred to the member′s local buffer pool through XES. To optimize accesses to the coupling facility, the page registration and subsequent page acquisition from the GBP can be performed in one XES function.

5.3.4 Force at Commit Processing

In a DB2 data sharing environment, the changed pages for a GBP-dependent resource must be written to the GBP before or during transaction commit processing. The locks protecting the changed pages will not be released until the pages are written to the GBP, thus ensuring that the GBP always has the most up to date version of the data page.

If a GBP is unavailable at the time the transaction commits, and the pages cannot be written to the GBP, DB2 places those updated pages on the logical page list (LPL).

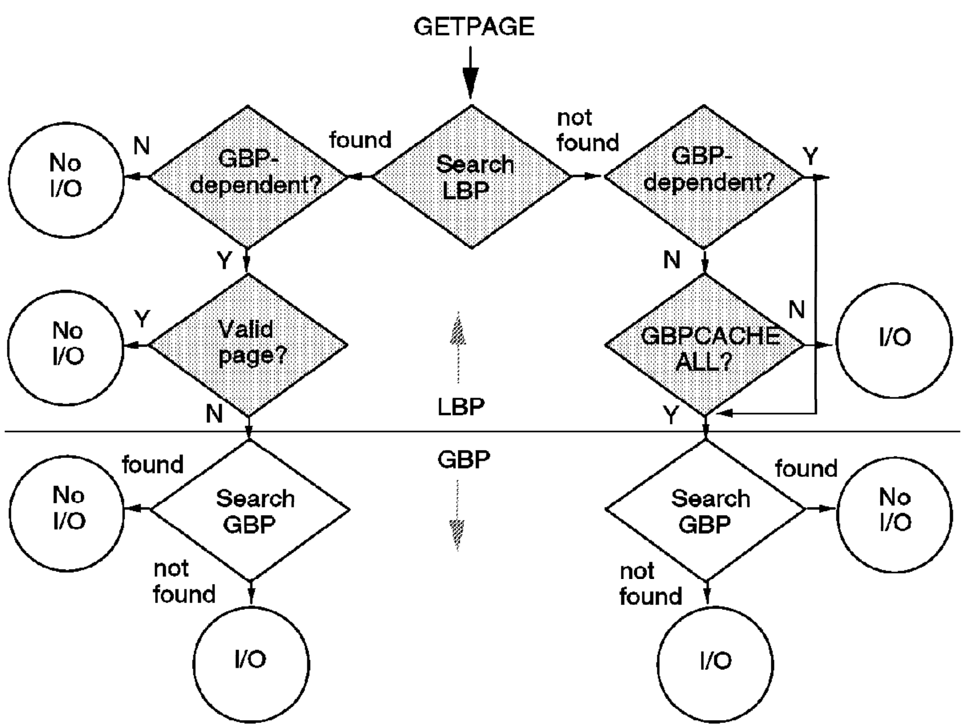
5.3.5 Cross-System Invalidation

When a data page is updated through a member in the data sharing group cross-system invalidation is used to flag any other copies of the page that exist in any of the other member′s local buffer pools as invalid. The next time the page is required by a member that has an old, invalid copy in its local buffer pool, the page will be reread in from the GBP or DASD.

Cross-system invalidation is also used when an existing directory entry must be reclaimed to handle new work. Even though the data page has not changed, it is still marked as invalid as it is no longer registered in the page directory. This invalidation requires that the page be refreshed when next needed from DASD, which can degrade the performance of the system.

5.3.6 Getpage in a Data Sharing Environment

Figure 17 shows you where a page will be searched for when a getpage is issued. The amount of I/O required is reduced if the page is cached in the GBP.



Note:

- LBP — Local Buffer Pool
- GBP — Group Buffer Pool

Figure 17. Buffer Manager Getpage with Group Buffer Pool

5.3.7 Group Buffer Pool Checkpoint

A GBP checkpoint triggers the writing of all changed pages in the GBP to DASD. The purpose of the GBP checkpoint is to reduce the amount of time needed to recover data in the event of a GBP failure.

When a GBP is damaged, all changed data that was stored in the GBP but not yet written out to DASD must be recovered from the DB2 logs. Writing changed pages out to DASD on a regular basis therefore reduces the data recovery time for the GBP as it ensures that the number of log records that have to be read for recovery purposes is relatively low.

At GBP checkpoint, the GBP page directory is read to identify which pages have to be cast out. The page directory is also read to determine the LRSN of the oldest changed page currently in the GBP. This LRSN is recorded in the member BSDSs and SCA and is the LRSN from which GBP recovery would have to take place. The LRSN of the oldest changed page is used because castout of changed pages is done asynchronously, and there is no guarantee that all changed pages in this checkpoint will make it out onto DASD.

The GBP checkpoint is triggered by the first DB2 member that causes the allocation of the GBP. If this original member disconnects from the GBP, another member in the group currently using the GBP will be assigned the task of triggering GBP checkpoints.

The default checkpoint frequency is 8 min. You can change this default by using the ALTER GROUPBUFFERPOOL command.

5.3.8 Group Buffer Pool Castout

The process of writing modified pages from the GBP to DASD is called *group buffer pool castout*. There is no physical connection between the GBP and DASD, so the castout process involves reading the page from the GBP into a group member′s private buffer (not part of the member′s buffer pool storage) and writing the page from the private buffer to DASD.

Castout is triggered when:

- A GBP checkpoint is taken
- The GBP castout threshold is reached
- The class castout threshold is reached
- GBP dependency is removed from a page set

Within a GBP there are a number of castout classes; the number of classes is an internal value set by DB2. Data sets (DB2 page sets or partitions) using the GBP are mapped to a specific castout class. DB2 will preferably have only one data set assigned to a particular castout class, although it is possible to have more than one data set mapped into the same castout class, depending on how many data sets are using the GBP concurrently.

Castout classes are used to limit the number of changed pages a data set can have in the GBP at any one time, thereby limiting the amount of I/O to the data set at castout time. (Large amounts of I/O could cause DASD contention.) This limitation is achieved through the use of the castout class threshold. The default of the castout class threshold parameter is 10, which means that castout is initiated for a particular class when 10% of the GBP contains pages for that class or, if only one data set is assigned to that class, when 10 percent of the GBP contains pages for that data set. The castout class threshold applies to all castout classes. You can change the castout class threshold by using the ALTER GROUPBUFFERPOOL command.

Data sets have a GBP castout owner assigned to them. The GBP castout owner is the first member to express write interest in the data set. After castout ownership is assigned, subsequent updating DB2 subsystems become backup owners. One of the backup owners becomes the castout owner when the original castout owner no longer has read-write interest in the page set or partition. At castout time the castout owner is responsible for performing the actual castout process for all changed pages for the data set.

How much is cast out?

- At GBP checkpoint, all changed pages across all data sets are cast out. The pages remain cached in the GBP, but the buffers are now available for stealing.
- When the GBP castout threshold is reached, castout is triggered for selected castout classes, until 10% of the changed pages in the GBP are written out. For example, if your GBP castout threshold is 50%, pages are written until the number of changed pages is 40% or less. The pages remain cached in the GBP, but the buffers are now available for stealing.
- When the class castout threshold is reached, the page write that triggered threshold detection determines which data set pages will be cast out. For example, if during a page write for table A the threshold is detected, up to a maximum of 128 pages are written to DASD. Table B, in the same cast out class, does not have any of its changed pages written to DASD. The pages remain cached in the GBP, but the buffers are now available for stealing.
- When GBP dependency is removed from a page set, all pages for the page set are cast out. The pages are purged from the GBP.

Chapter 6. Installing DB2 Data Sharing

Before creating a data sharing group you must first determine what you will use it for and which workloads you will run in it. Having that information will help you decide the best way to install data sharing in your environment.

You may want to enable data sharing for an existing DB2 subsystem or install a new data sharing group. Each method has a number of advantages and disadvantages which may vary from installation to installation.

A typical installation may choose to enable data sharing using both methods. A data sharing group for software testing can be created as a new installation. A data sharing group for development or testing can also be created as a new installation. Current production workloads can be moved to data sharing by enabling data sharing in existing DB2 subsystems.

It is very important to introduce a meaningful naming convention. With your installation in mind you could decide to introduce a new naming convention by enabling data sharing as a new install. Using this approach you can keep the existing subsystems with their naming convention unchanged and move applications and data step by step from an existing non-data-sharing system to the new data sharing group. In this way the naming convention you choose for data sharing need not be compromised by existing naming conventions that may not work well for what you plan to do with data sharing. Alternatively, this approach may not be viable if you plan to move large applications into data sharing that have large amounts of data that must be moved. Enabling data sharing using the existing DB2 subsystem may work better for you in these cases.

Enabling data sharing as a new install allows you to “clean up” your existing subsystems by selectively moving only applications and data that are active in your current system. You may be able to gradually increase the workload in your data sharing group by selectively moving applications. You then have the opportunity to monitor the performance of your applications in the new data sharing group and also learn about tuning a data sharing group. This is an advantage, if you have no possibilities to simulate a production workload in your test environment.

If you plan to migrate an application that performs a lot of distributed processing with other subsystems into a newly installed data sharing group, the DB2 location name that the application uses for communication between the data sharing group and other existing DB2 subsystems may have to change. This can be costly, as the application may also need to change for any of the new location names (unless the application uses some form of dynamic lookup table). An alternative implementation plan may be to use the existing DB2 location name as the data sharing group location name and enable an existing DB2 subsystem for data sharing.

For applications that have many batch processors where a lot of JCL and/or procedures may have to change, use the existing DB2 attach name for the data sharing group attach name and enable the existing DB2 subsystem for data sharing.

Remember, DB2 does not provide an automatic way of merging catalogs and resolving naming conflicts. This fact alone may be the major determinant of the method you choose to implement data sharing in your environment. **Whichever method you choose to enable data sharing, we strongly recommend that you become familiar with and use other Version 4 features before moving to data sharing**. This approach will enable you to test other Version 4 functions, without the added complexity of data sharing and coupling facility structures. If you plan to enable data sharing in an existing DB2, subsystem, migrating to Version 4 before enabling data sharing provides the opportunity to convert all indexes to Type 2 and reduces the likelihood of having to fall back to Version 3 after enabling data sharing.

Table 8 summarizes the various DSNTINST CLIST installation options you can choose to enable data sharing.

Table 8. Installation Options for Data Sharing

From ...	To ...	Choose
A Version 3 subsystem	A Version 4 subsystem	`INSTALL TYPE ===> migrate` `DATA SHARING FUNCTION ===> none`
No system	Version 4 non-data-sharing	`INSTALL TYPE ===> install` `DATA SHARING FUNCTION ===> none`

No system	Version 4 data sharing	`INSTALL TYPE ===> install`
		`DATA SHARING FUNCTION ===> group`
		`GROUP NAME ===> group name`
		`MEMBER NAME ===> originating member name`

A Version 4 non-data-sharing subsystem	A Version 4 data sharing subsystem	`INSTALL TYPE ===> install`
		`DATA SHARING FUNCTION ===> enable`
		`GROUP NAME ===> group name`
		`MEMBER NAME ===> originating member name`

One-member data sharing group	Add a new member	`INSTALL TYPE ===> install`
		`DATA SHARING FUNCTION ===> member`
		`GROUP NAME ===> group name`
		`MEMBER NAME ===> new member name`

In the rest of this chapter we present implementation guidelines to supplement the information in the *IBM DATABASE 2 for MVS/ESA Version 4 Data Sharing: Planning and Administration* manual and *IBM DATABASE 2 for MVS/ESA Version 4 Installation Guide*.

6.1 Naming Conventions Used

There are various ways to install data sharing in your environment.

The examples are based on a data sharing group with the naming convention described in Table 9.

Table 9. Sample Naming Convention for a Data Sharing Group

Group name	DSNDSGC
Group attach name	DSGC
Member names	DBC1, DBC2
Member command prefixes	=DBC1, =DBC2
Member subsystem names	DBC1, DBC2
IRLM group name	DXRDSGC
IRLM subsystem names	IRC1, IRC2
Member parameter modules	DSNZDBC1, DSNZDBC2
Work databases	WORKDBC1, WORKDBC2
Data set names	DSNDSGC.DBC#.**

6.2 Enabling an Existing DB2 V4 Non-Data-Sharing System

In this section we describe how to enable an existing DB2 V4 non-data-sharing system for data sharing. In terms of the DB2 installation this procedure is called *creating a data sharing group*. To enable data sharing for an existing DB2 subsystem, you must first decide which existing DB2 V4 subsystem to use. This subsystem will become the *originating* member for the data sharing group. The existing catalog and directory will become the catalog and directory of the data sharing group. After the data sharing group is verified, you can add new members as the need arises.

Before you begin to enable data sharing, you must be sure that all of your indexes have been converted to Type 2 indexes (or at least Type 1 with subpage 1). Although Type 1 indexes with subpage 1 can be used in a data sharing environment, they are not recommended as you will not be able to use other DB2 Version 4 features in your data sharing group

You can have Type 1 indexes with subpage > 1 in a data sharing group, but they cannot be shared. DB2 generates a warning SQLCODE (-2000) when you create a Type 1 index with subpages > 1 in a data sharing group. DB2 returns a resource unavailable SQLCODE with a DB2 reason code of 00C900C2 if you try to use the index in such a way that DB2 must place it in GBP-dependent status. The table can still become GBP-dependent, however.

You can query the catalog to find out whether you have converted all of your indexes to Type 2. Figure 18 on page 88 shows the query to run and the results using SPUFI.

```
-- To test the catalog whether there are still Type 1 indexes       --
--                                                                  --
   SELECT CREATOR, NAME, INDEXSPACE, INDEXTYPE
     FROM SYSIBM.SYSINDEXES
      WHERE INDEXTYPE <> '2'
      ORDER BY NAME, CREATOR;
```

Figure 18. Type 2 Index Query

The Type 2 index query works for the catalog and user databases. However for the directory database (DSNDB01), you cannot obtain information about its indexes from the catalog. So, make sure you run the CATMAINT utility with the CONVERT option to convert the directory indexes to Type 2.

Using the ENABLE data sharing option you enable data sharing in an existing subsystem which then becomes the originating member of the data sharing group.

To enable data sharing in an existing DB2 V4 subsystem, follow these steps:

1. Invoke the installation CLIST, DSNTINST to get to the DSNTIPA panel (see Figure 19).

```
DSNTIPA1            INSTALL, UPDATE, AND MIGRATE DB2 - MAIN PANEL
===>

Check parameters and reenter to change:

 1  INSTALL Type          ===> INSTALL   Install, Update, or Migrate
 2  DATA SHARING FUNCTION ===> ENABLE    None, Group, Member, or Enable

Enter the following value for migration only:
 3  DATA SET NAME(MEMBER) ===>

Enter name of your input data sets (SDSNLOAD, SDSNMACS, SDSNSAMP, SDSNCLST):
 4  PREFIX                ===> DSN410
 5  SUFFIX                ===>

Enter to set or save panel values (by reading or writing the named members):
 6  INPUT MEMBER NAME     ===> DSNTIDXA  Enter to read old panel values
 7  OUTPUT MEMBER NAME    ===> DSNTIDC1     Enter to write new panel va



PRESS:  ENTER to continue   RETURN to exit   HELP for more information
```

Figure 19. Install Enable: Panel DSNTIPA1

On the DSNTIPAI panel, specify:

- **INSTALL** for INSTALL Type

- **ENABLE** for DATA SHARING FUNCTION.
- An OUTPUT MEMBER NAME for the parameter values of the installation panel. You have to specify this member as the INPUT MEMBER NAME when you run the installation CLIST again to add new members to the data sharing group.

2. Press ENTER to get to the DSNTIPK panel (Figure 20).

```
DSNTPK          INSTALL DB2 - DEFINE GROUP OR MEMBER
===>
DSNT514I Fields 2 and 3 must be unique for each new member installed
Check parameters and reenter to change:

 1  GROUP NAME    ===> DSNDSGC      Name of the DB2 group
 2  MEMBER NAME   ===> DBC1         Name of DB2 member in group
 3  WORK FILE DB  ===> WORKDBC1     Work file database name for this
 4  GROUP ATTACH  ===> DSGC         Group attach name for TSO, batch













PRESS:  ENTER to continue   RETURN to exit   HELP for more information
```

Figure 20. Install Enable: Panel DSNTPK

On the DSNTPK panel specify:

- **DSNDSGC** for GROUP NAME
- **DBC1** for MEMBER NAME
- **WORKDBC1** for WORK FILE DB
- **DSGC** for GROUP ATTACH

3. Follow the installation panels as described in the *IBM DATABASE 2 for MVS/ESA Version 4 Installation Guide* until prompted to specify DDF information.
4. Specify the DDF information on the DSNTIPR panel (Figure 21 on page 90).

```
DSNTIPR                 INSTALL DB2 - DISTRIBUTED DATA FACILITY
 ===>
 DSNT512I Field 3 must be unique for each new member installed
 Enter data below:

  1 DDF STARTUP OPTION   ===> AUTO      NO, AUTO, or COMMAND
  2 DB2 LOCATION NAME    ===> DSGC              The name other DB2s
                                                refer to this DB2
  3 DB2 NETWORK LUNAME   ===> SCDBC1   The name VTAM uses to refer t
  4 DB2 NETWORK PASSWORD ===>           Password for DB2's VTAM application
  5 RLST ACCESS ERROR    ===> NOLIMIT   NOLIMIT, NORUN, or 1-5000000
  6 RESYNC INTERVAL      ===> 2         Minutes between resynchronization period
  7 DDF THREADS          ===> ACTIVE    (ACTIVE or INACTIVE) Status of a
                                        database access thread that commits or
                                        rolls back and holds no database locks
                                        or cursors
  8 DB2 GENERIC LUNAME   ===> SCDSGC    Generic VTAM LU name for this DB2
                                        subsystem or data sharing group
  9 IDLE THREAD TIMEOUT  ===> 0         0 or seconds until dormant server ACTIVE
                                        thread will be terminated (0-9999)


 PRESS:  ENTER to continue   RETURN to exit   HELP for more information
```

Figure 21. Install Enable: Panel DSNTIPR

Note the following about the DSNTIPR panel:

- A data sharing group can have only one location name. Therefore, the DB2 LOCATION NAME specifies the location name of the entire data sharing group, not the DB2 member. Specify **DSGC** for the DB2 LOCATION NAME.
- The DB2 NETWORK LUNAME is the LU name specific to this member. Specify **SCDBC1** as the LU name for member DBC1.
- The DB2 GENERIC LUNAME specifies the generic LU name used for this data sharing group. You have to specify the same value for each member of the data sharing group. However if you do not have your VTAM configured to support generic LUs, do not specify any value here. If you do specify a value, DB2 ignores it and issues a message indicating that it has ignored it.

5. Complete the installation panels as described in the *IBM DATABASE 2 for MVS/ESA Version 4 Installation Guide*.

6. Define the CFRM policy in the coupling facility.

   Chapter 3 in the *IBM DATABASE 2 for MVS/ESA Version 4 Data Sharing: Planning and Administration* contains a sample CFRM policy.

7. If a RACF profile is defined for the structure resource, you must define rules to allow DB2 and IRLM the proper security authorization facility (SAF) authorization to connect to the SCA, lock structure, and GBPs. (MVS checks SAF authorization only when a profile exists. If there is no profile, any authorized user can connect to the structure.)

If this is not done DB2 will fail on start with DB2 reason code 00F70602 (for SCA) or 00E30806 (for lock structure) or resource unavailable with DB2 reason code 00C20204 (for group buffer pool).

For example, if SAF authorization is not defined, DB2 will fail when you try to start it because it cannot access the SCA. In this case you will find messages similar to these:

```
IXL012I IXLCONN REQUEST FAILED, RETCOSE: 00000008, RSNCODE: 0201084C

DSN706A CONNECTION TO THE SCA STRUCTURE strname FAILED.
        MVS IXLCONN RETURN CODE = 00000008,
        MVS IXLCONN REASON CODE = 0201084C.
```

The user ID associated with the DB2 address spaces must be granted ALTER access over the IXLSTR.strname facility-class resource. Because IRLM is the connector for the lock structure, the user ID associated with the IRLM address space must have ALTER access over IXLSTR.lockstrname.

8. Complete the installation steps as defined in Chapter 2 of *IBM DATABASE 2 for MVS/ESA Version 4 Installation Guide*.

9. Complete the data sharing installation verification procedure as defined in Chapter 3 of *IBM DATABASE 2 for MVS/ESA Version 4 Data Sharing: Planning and Administration*.

Job DSNTIJUZ creates the member's parameter module, DSNZDBC1; the data-only module, DSNHDECP; and a change log inventory utility step to enable data sharing.

Figure 22 shows the new DSNZDBC1 parameters that are related to data sharing.

```
          .
          .
DSN6GRP    DSHARE=YES,
          GRPNAME=DSNDSGC,
          MEMBNAME=DBC1
```

Figure 22. Parameters of DSNZDBC1 to Enable Data Sharing

Figure 23 on page 92 shows the change log inventory utility that must be run to enable data sharing. The change log inventory utility updates the BSDS with the records for data sharing.

```
//DSNTLOG EXEC PGM=DSNJU003,COND=(4,LT)
//STEPLIB  DD  DISP=SHR,DSN=DSN410.SDSNLOAD
//SYSUT1   DD  DISP=OLD,DSN=DSNDSGC.DBC1.BSDS01
//SYSUT2   DD  DISP=OLD,DSN=DSNDSGC.DBC1.BSDS02
//SYSPRINT DD  SYSOUT=*
//SYSUDUMP DD  SYSOUT=*
//SYSIN    DD  *
 DATASHR ENABLE
 GROUP GROUPNAM=DSNDSGC,GROUPMEM=DBC1,MEMBERID=1
```

Figure 23. Change Log Inventory to Enable Data Sharing

After you have completed the installation jobs and successfully started the DB2 subsystem, a number of new messages will appear in the MSTR address space. Figure 24 shows the sample output of DBCIMSTR. In this output message DSN75071 indicates the allocation of the SCA structure.

```
1                 J E S 2  J O B  L O G  --  S Y S T E M  S C 4 8  --  N O D E  W T S C P L X 1
0
 13.20.12 STC25475  IEF695I START DBC1MSTR WITH JOBNAME DBC1MSTR IS ASSIGNED TO USER STC     , GROUP SYS1
 13.20.12 STC25475  $HASP373 DBC1MSTR STARTED
 13.20.12 STC25475  IEF403I DBC1MSTR - STARTED - TIME=13.20.12
 13.20.13 STC25475  DSNZ002I =DBC1 DSNZINIT SUBSYSTEM DBB1 SYSTEM PARAMETERS LOAD MODULE NAME IS DSNZDBC1
 13.20.15 STC25475  DSN7507I =DBC1 DSN7LSTK
                    SCA STRUCTURE DSNDSGC_SCA IS ALLOCATED IN A VOLATILE STRUCTURE.
 13.20.15 STC25475  S DBC1IRLM
 13.20.27 STC25475  DSNY001I =DBC1 SUBSYSTEM STARTING
 13.20.29 STC25475  DSNJ127I =DBC1 SYSTEM TIMESTAMP FOR BSDS= 96.163 13:1
```

Figure 24. Output of DBC1MSTR

In the output of the IRLM address space (see Figure 25) message DXR141E shows the allocation of the lock structure and message DXR132I shows that the IRLM has successfully joined the data sharing lock structure.

```
1                 J E S 2  J O B  L O G  --  S Y S T E M  S C 4 8  --  N O D E  W T S C P L X 1
0
 13.20.15 STC25476  IEF695I START DBC1IRLM WITH JOBNAME DBC1IRLM IS ASSIGNED TO USER STC     , GROUP SYS1
 13.20.15 STC25476  $HASP373 DBC1IRLM STARTED
 13.20.15 STC25476  IEF403I DBC1IRLM - STARTED - TIME=13.20.15
 13.20.15 STC25476  DXR117I IRC1 INITIALIZATION COMPLETE
 13.20.26 STC25476  DXR141E IRC1 THE LOCK TABLE DSNDSGC_LOCK1  WAS ALLOCATED IN A VOLATILE FACILITY
 13.20.27 STC25476  DXR132I IRC1 SUCCESSFULLY JOINED THE DATA SHARING GROUP, GLOBAL INITIALIZATION IS COMPLETE
```

Figure 25. Output of DBC1IRLM

From now on you can only specify the TOLOGPOINT keyword in the RECOVER utility if you want to recover an object to a point-in-time after data sharing was enabled. You can still specify the TOLOGPOINT or TORBA keywords in the RECOVER utility if you want to recover an object to a point-in-time before data sharing was enabled.

6.3 Add Members to the Data Sharing Group

As your data sharing workload grows, you may want to add new members to the data sharing group. This is quite a simple process. In this section we describe how to define a new DB2 member and start it in the data sharing group. After installing a new data sharing group you can add new members to it.

Using the MEMBER data sharing option, you define a new DB2 subsystem that will be started in the data sharing group. Refer to Section 2 in the *IBM DATABASE 2 for MVS/ESA Version 4 Data Sharing: Planning and Administration* for a discussion of the installation procedure.

To add a new member to the data sharing group, follow these steps.

1. Invoke the installation CLIST, DSNTINST, to get to the DSNTIPAI panel (see Figure 26).

```
DSNTIPA1            INSTALL, UPDATE, AND MIGRATE DB2 - MAIN PANEL
===>
Check parameters and reenter to change:

 1  INSTALL TYPE          ===> INSTALL   Install, Update, or Migrate
 2  DATA SHARING FUNCTION ===> MEMBER    None, Group, Member, or Enable

Enter the following value for migration only:
 3  DATA SET NAME(MEMBER) ===>

Enter name of your input data sets (SDSNLOAD, SDSNMACS, SDSNSAMP, SDSNCLST):
 4  PREFIX                ===> DSN410
 5  SUFFIX                ===>

Enter to set or save panel values (by reading or writing the named members):
 6  INPUT MEMBER NAME     ===> DSNTIDC1  Enter to read old panel values
 7  OUTPUT MEMBER NAME    ===> DSNTIDC2  Enter to write new panel values



PRESS:  ENTER to continue   RETURN to exit   HELP for more information
```

Figure 26. Install Member: Panel DSNTIPA1

On the DSNTIPAI panel, specify:

- **INSTALL** for INSTALL TYPE
- **MEMBER** for DATA SHARING FUNCTION
- An OUTPUT MEMBER NAME. You must use the OUTPUT MEMBER NAME that was used in the DSNTINST CLIST for the originating data sharing group DB2 member as the INPUT MEMBER NAME when you install a new DB2 subsystem into an existing data sharing group. If you do not follow this rule, the installation CLIST does not generate the jobs correctly, because you can only overwrite member-specific values. You cannot change any group-generic specification.

2. Press ENTER to get to the DSNTIPK panel (Figure 27 on page 94).

```
DSNTIPK         INSTALL DB2 - DEFINE GROUP OR MEMBER
===>

Check parameters and reenter to change:

 1  GROUP NAME    ===> DSNDSGC      Name of the DB2 group
 2  MEMBER NAME   ===> DBC2         Name of DB2 member in group
 3  WORK FILE DB  ===> WORKDBC2     Work file database name for this
 4  GROUP ATTACH  ===> DSGC         Group attach name for TSO, batch









PRESS:  ENTER to continue   RETURN to exit   HELP for more information
```

Figure 27. Install Member: Panel DSNTIPK

On the DSNTIPK panel, specify:

- **DSNDSGC** for GROUP NAME
- **DBC2** for MEMBER NAME
- **WORKDBC2** for WORK FILE DB
- **DSGC** for GROUP ATTACH

3. Follow the installation panels as described in the *IBM DATABASE 2 for MVS/ESA Version 4 Installation Guide* until prompted to specify DDF information.

4. Specify the DDF information on the DSNTIPR panel (see Figure 28 on page 95).

```
DSNTIPR                 INSTALL DB2 - DISTRIBUTED DATA FACILITY
 ===>
 DSNT512I Field 3 must be unique for each new member installed
 Enter data below:

  1 DDF STARTUP OPTION   ===> AUTO      NO, AUTO, or COMMAND
  2 DB2 LOCATION NAME    ===> DSGC              The name other DB2s
                                                refer to this DB2
  3 DB2 NETWORK LUNAME   ===> SCDBC2   The name VTAM uses to refer t
  4 DB2 NETWORK PASSWORD ===>           Password for DB2's VTAM application
  5 RLST ACCESS ERROR    ===> NOLIMIT   NOLIMIT, NORUN, or 1-5000000
  6 RESYNC INTERVAL      ===> 2         Minutes between resynchronization period
  7 DDF THREADS          ===> ACTIVE    (ACTIVE or INACTIVE) Status of a
                                        database access thread that commits or
                                        rolls back and holds no database locks
                                        or cursors
  8 DB2 GENERIC LUNAME   ===> SCDSGC    Generic VTAM LU name for this DB2
                                        subsystem or data sharing group
  9 IDLE THREAD TIMEOUT  ===> 0         0 or seconds until dormant server ACTIVE
                                        thread will be terminated (0-9999)


 PRESS:  ENTER to continue   RETURN to exit   HELP for more information
```

Figure 28. Install Member: Panel DSNTIPR

Note the following about the DSNTIPR panel:

- This panel is a little unclear when you are installing a new DB2 member into an existing data sharing group. A data sharing group can have only one location name. Therefore, the DB2 LOCATION NAME specifies the location name of the entire data sharing group, not the DB2 member. Specify **DSGC** for the DB2 LOCATION NAME.
- The DB2 NETWORK LUNAME is the LU name specific to this member. Specify **SCDBC2** as the LU Name for member DBC2.
- The DB2 GENERIC LUNAME specifies the generic LU name used for this data sharing group. You have to specify the same value for each member of the data sharing group. However, if you do not have your VTAM configured to support generic LUs, do not specify any value here. DDF will fail if generic LUs are not available on your system. Generic LUs are stored in the DB2 member's BSDS and can be updated at any time.

5. Complete the installation panels as described in the *IBM DATABASE 2 for MVS/ESA Version 4 Installation Guide*.

6. Review the CFRM policy in the coupling facility, as you may want to change some structure sizes in preparation for some increased data sharing workload.

7. Review RACF rules to make sure the new DB2 member has the proper SAF authorization to connect to the SCA, lock structure, and GBPs in the coupling facility.

If a RACF profile is defined for the structure resource, you must define rules to allow DB2 and IRLM the proper SAF authorization to connect to the SCA, lock structure, and GBPs. (MVS checks SAF authorization only when a profile exists. If there is no profile, any authorized user can connect to the structure.) If this is not done DB2 will fail on start with DB2 reason code 00F70602 (for SCA) or 00E30806 (for lock structure) or resource unavailable with DB2 reason code 00C20204 (for group buffer pool).

For example, if no SAF authorization is defined, DB2 will fail when you try to start it because it cannot access the SCA. In this case you will find messages similar to these:

```
IXL012I IXLCONN REQUEST FAILED, RETCOSE: 00000008, RSNCODE: 0201084C

DSN706A CONNECTION TO THE SCA STRUCTURE strname FAILED.
        MVS IXLCONN RETURN CODE = 00000008,
        MVS IXLCONN REASON CODE = 0201084C.
```

The user ID associated with the DB2 address spaces must be granted ALTER access over the IXLSTR.strname facility-class resource. Because IRLM is the connector for the lock structure, the user ID associated with the IRLM address space must have ALTER access over IXLSTR lockstrname.

8. Complete the installation steps as defined in Chapter 2 of the *IBM DATABASE 2 for MVS/ESA Version 4 Installation Guide*.

9. Complete the installation verification procedure as defined in Chapter 2 of the *IBM DATABASE 2 for MVS/ESA Version 4 Installation Guide*.

10. Complete the data sharing installation verification procedure as defined in Chapter 3 of *IBM DATABASE 2 for MVS/ESA Version 4 Data Sharing: Planning and Administration*,

Remember that there is a single data-only module DSNHDECP for a data sharing group. The application programming defaults contained in the DSNHDECP are considered global for the whole data sharing group. The DSNHDECP module is created during the installation of the data sharing group. During the procedure of adding new members to the group, it is not modified and therefore does not appear in job DSNTIJUZ.

Figure 29 on page 97 shows the contents of the BSDS for member DBC1 after member DBC2 was successfully introduced into the data sharing group.

```
********************************************************************************
*
*              LOG MAP OF THE BSDS DATA SET BELONGING TO MEMBER 'DBC1    ' OF
*
********************************************************************************
RELEASE LEVEL OF BSDS -  ACTIVE=2.3 AND ABOVE   ARCHIVE=2.3 AND ABOVE   DDNAME=
   LOG MAP OF BSDS DATA SET COPY 1, DSN=DSNDSGC.DBC1.BSDS01
   LTIME INDICATES LOCAL TIME, ALL OTHER TIMES ARE GMT.
         DATA SHARING MODE IS ON
         SYSTEM TIMESTAMP   - DATE=1996.162  LTIME= 7:05:36.79
         UTILITY TIMESTAMP  - DATE=1996.158  LTIME=16:45:27.18
         VSAM CATALOG NAME=DSNDSGC
         HIGHEST RBA WRITTEN       000000667DD0  1996.162  07:10:49.0
         HIGHEST RBA OFFLOADED     00000065EFFF
         RBA WHEN CONVERTED TO V4  000000000000
         MAX RBA FOR TORBA         000000000000
         MIN RBA FOR TORBA         000000000000
         STCK TO LRSN DELTA        000000000000
       THIS BSDS HAS MEMBER RECORDS FOR THE FOLLOWING MEMBERS:
         HOST MEMBER NAME:            DBC1
           MEMBER ID:                 1
           GROUP NAME:                DSNDSGC
           BSDS COPY 1 DATA SET NAME: DSNDSGC.DBC1.BSDS01
           BSDS COPY 2 DATA SET NAME: DSNDSGC.DBC1.BSDS02
 1       HOST MEMBER NAME:            DBC2
           MEMBER ID:                 2
           GROUP NAME:                DSNDSGC
           BSDS COPY 1 DATA SET NAME: DSNDSGC.DBC2.BSDS01
           BSDS COPY 2 DATA SET NAME: DSNDSGC.DBC2.BSDS02
ACTIVE LOG COPY 1 DATA SETS
  START RBA/LRSN/TIME   END RBA/LRSN/TIME      DATE    LTIME  DATA SET INFORMATION
 --------------------  --------------------  --------  -----  -----------------
                 EMPTY DATA SET              1996.158  16:18  DSN=DSNDSGC.DBC1.
     000000000000          000000000000                       STATUS=NEW, REUSABLE
 0000.000  00:00:00.0  0000.000  00:00:00.0
   000000000000          00000065EFFF        1996.158  16:18  DSN=DSNDSGC.DBC1.
     ACFAB5BD0604          ACFAD1E3F8A1                       STATUS=TRUNCATED,
 1996.158  21:17:33.8  1996.158  23:23:30.8
   00000065F000          00000281EFFF        1996.158  16:18  DSN=DSNDSGC.DBC1.
     ACFAD1E3F8A2          ............                       STATUS=NOTREUSABLE
       .     .     .     .     .     .     .     .     .     .
```

Figure 29. Contents of Originating Member BSDS

Note:

1 The BSDS for member DBC1 now contains information about member DBC2. A print log map utility listing of the second member's BSDS will also show member records for both DBC1 and DBC2.

Before the current status rebuild (CSR) phase of DB2 restart, each DB2 member updates its own BSDS with any new log information it has read from the SCA. Before the end of restart, the startup member copies all peer members' information from the SCA to its own BSDS and notifies all active peer members to copy its information to their BSDS.

6.4 Install a New Data Sharing Group

To create a new data sharing group as a new installation, you install a new DB2 subsystem and enable data sharing in one step. In this section we describe the steps required, although you will probably never have a need for this.

Using the GROUP data sharing option, you install a new subsystem that becomes the originating member of the data sharing group.

To install a new DB2 subsystem and enable data sharing in one step, follow these steps.

1. Follow the steps in Chapter 2 of *IBM DATABASE 2 for MVS/ESA Version 4 Installation Guide* to load the DB2 libraries and make the DB2 ISPF libraries available to TSO.

2. Invoke the installation CLIST, DSNTINST, to set to the DSNTIPAI panel (Figure 30).

```
DSNTIPA1            INSTALL, UPDATE, AND MIGRATE DB2 - MAIN PANEL
===>
Check parameters and reenter to change:

 1  INSTALL TYPE          ===> INSTALL   Install, Update, or Migrate
 2  DATA SHARING FUNCTION ===> GROUP    None, Group, Member, or Enab

Enter the following value for migration only:
 3  DATA SET NAME(MEMBER) ===>

Enter name of your input data sets (SDSNLOAD, SDSNMACS, SDSNSAMP, SDSNCLST):
 4  PREFIX                ===> DSN410
 5  SUFFIX                ===>

Enter to set or save panel values (by reading or writing the named members):
 6  INPUT MEMBER NAME     ===> DSNTIDXA  Enter to read old panel values
 7  OUTPUT MEMBER NAME    ===> DSNTIDC1  Enter to write new panel values



PRESS:  ENTER to continue   RETURN to exit   HELP for more information
```

Figure 30. Install Group: Panel DSNTIPA1

On the DSNTIPAI panel, specify:

- **INSTALL** for INSTALL TYPE
- **GROUP** for DATA SHARING FUNCTION
- An OUTPUT MEMBER NAME for the parameter values of the installation panel. You have to specify this member as the INPUT MEMBER NAME when you run the installation CLIST again to add new members to the data sharing group.

3. Press ENTER to get to the DSNTIPK panel (Figure 31 on page 99).

```
DSNTIPK         INSTALL DB2 - DEFINE GROUP OR MEMBER
===>

Check parameters and reenter to change:

 1  GROUP NAME    ===> DSNDSGC      Name of the DB2 group
 2  MEMBER NAME   ===> DBC1         Name of DB2 member in group
 3  WORK FILE DB  ===> WORKDBC1     Work file database name for this
 4  GROUP ATTACH  ===> DSGC         Group attach name for TSO, batch











PRESS:  ENTER to continue   RETURN to exit   HELP for more information
```

Figure 31. Install Group: Panel DSNTIPK

On the DSNTIPK panel, specify:

- **DSNDSGC** for GROUP NAME
- **DBC1** for MEMBER NAME
- **WORKDBC1** for WORK FILE DB
- **DSGC** for GROUP ATTACH

4. Follow the installation panels as described in the *IBM DATABASE 2 for MVS/ESA Version 4 Installation Guide* until prompted to specify DDF information.
5. Specify the DDF information on the DSNTIPR panel (see Figure 32 on page 100).

```
DSNTIPR                 INSTALL DB2 - DISTRIBUTED DATA FACILITY
 ===>
 DSNT512I Field 3 must be unique for each new member installed
 Enter data below:

  1 DDF STARTUP OPTION   ===> AUTO      NO, AUTO, or COMMAND
  2 DB2 LOCATION NAME    ===> DSGC              The name other DB2s
                                                refer to this DB2
  3 DB2 NETWORK LUNAME   ===> SCDBC1   The name VTAM uses to refer t
  4 DB2 NETWORK PASSWORD ===>           Password for DB2's VTAM application
  5 RLST ACCESS ERROR    ===> NOLIMIT   NOLIMIT, NORUN, or 1-5000000
  6 RESYNC INTERVAL      ===> 2         Minutes between resynchronization period
  7 DDF THREADS          ===> ACTIVE    (ACTIVE or INACTIVE) Status of a
                                        database access thread that commits or
                                        rolls back and holds no database locks
                                        or cursors
  8 DB2 GENERIC LUNAME   ===> SCDSGC    Generic VTAM LU name for this DB2
                                        subsystem or data sharing group
  9 IDLE THREAD TIMEOUT  ===> 0         0 or seconds until dormant server ACTIVE
                                        thread will be terminated (0-9999)


 PRESS:  ENTER to continue   RETURN to exit   HELP for more information
```

Figure 32. Install Group: Panel DSNTIPR

Note the following about the DSNTIPR panel:

- This panel is a little unclear when you are creating a new data sharing group. A data sharing group can have only one location name. Therefore, the DB2 LOCATION NAME specifies the location name of the entire data sharing group, not the DB2 member. Specify **DSGC** for the DB2 LOCATION NAME.
- The DB2 NETWORK LUNAME is the LU name specific to this member. Specify **SCDBC1** as the LU name for member DBC1.
- The DB2 GENERIC LUNAME specifies the generic LU name used for this data sharing group. You have to specify the same value for each member of the data sharing group. However, if you do not have your VTAM configured to support generic LUs, do not specify any value here. DDF will fail if generic LUs are not available on your system. Generic LUs are stored in the DB2 member's BSDS and can be updated at any time.

6. Complete the installation panels as described in the *IBM DATABASE 2 for MVS/ESA Version 4 Installation Guide*.
7. Define the CFRM policy in the coupling facility.

   Chapter 3 in the *IBM DATABASE 2 for MVS/ESA Version 4 Data Sharing: Planning and Administration* contains a sample CFRM policy.
8. If a RACF profile is defined for the structure resource, you must define rules to allow DB2 and IRLM the proper SAF authorization to connect to the SCA, lock structure, and

GBPs. (MVS checks SAF authorization only when a profile exists. If there is no profile, any authorized user can connect to the structure.)

If this is not done, DB2 will fail on start with DB2 reason code 00F70602 (for SCA) or 00E30806 (for lock structure) or resource unavailable with DB2 reason code 00C20204 (for group buffer pool).

For example, if no SAF authorization is defined, DB2 will fail when you try to start it because it cannot access the SCA. In this case you will find messages similar to these:

```
IXL012I IXLCONN REQUEST FAILED, RETCOSE: 00000008, RSNCODE: 0201084C

DSN706A CONNECTION TO THE SCA STRUCTURE strname FAILED.
        MVS IXLCONN RETURN CODE = 00000008,
        MVS IXLCONN REASON CODE = 0201084C.
```

The user ID associated with the DB2 address spaces must be granted ALTER access over the IXLSTR.strname facility-class resource. Because IRLM is the connector for the lock structure, the user ID associated with the IRLM address space must have ALTER access over IXLSTR.lockstrname.

9. Complete the installation steps as defined in Chapter 2 of the *IBM DATABASE 2 for MVS/ESA Version 4 Installation Guide*.

10. Complete the installation verification procedure as defined in Chapter 2 of the *IBM DATABASE 2 for MVS/ESA Version 4 Installation Guide*.

11. Complete the data sharing installation verification procedure as defined in Chapter 3 of *IBM DATABASE 2 for MVS/ESA Version 4 Data Sharing: Planning and Administration*.

Figure 33 on page 102 shows the contents of the BSDS before starting the DB2 subsystem for the first time.

```
********************************************************************************
*
*              LOG MAP OF THE BSDS DATA SET BELONGING TO MEMBER 'DBC1    ' OF
*
********************************************************************************
RELEASE LEVEL OF BSDS -  ACTIVE=2.3 AND ABOVE   ARCHIVE=2.3 AND ABOVE   DDNAME=
 1 WHEN THIS DB2 SUBSYSTEM IS RESTARTED, DATA SHARING WILL BE ENABLED.
     THIS BSDS HAS NOT BEEN SCHEDULED FOR CONVERSION TO VERSION 4.
   LOG MAP OF BSDS DATA SET COPY 0, DSN=DSNDSGC.DBC1.BSDS01
   LTIME INDICATES LOCAL TIME, ALL OTHER TIMES ARE GMT.
 2       DATA SHARING MODE IS OFF
         SYSTEM TIMESTAMP   - DATE=(NULL)  LTIME=(NULL)
         UTILITY TIMESTAMP  - DATE=1996.158  LTIME=16:45:27.18
         VSAM CATALOG NAME=DSNDSGC
 3       HIGHEST RBA WRITTEN       000000000000  0000.000  00:00:00.0
         HIGHEST RBA OFFLOADED     000000000000
         RBA WHEN CONVERTED TO V4  000000000000
         MAX RBA FOR TORBA         000000000000
         MIN RBA FOR TORBA         000000000000
         STCK TO LRSN DELTA        000000000000
ACTIVE LOG COPY 1 DATA SETS
    START RBA/TIME         END RBA/TIME        DATE    LTIME  DATA SET INFO
 --------------------  --------------------  --------  -----  -------------
                 EMPTY DATA SET              1996.158  16:18  DSN=DSNDSGC.D
 0000.000  00:00:00.0  0000.000  00:00:00.0                   STATUS=NEW, R
                 EMPTY DATA SET              1996.158  16:18  DSN=DSNDSGC.D
 0000.000  00:00:00.0  0000.000  00:00:00.0                   STATUS=NEW, R
      .     .     .     .     .     .     .     .     .     .
```

Figure 33. Contents of BSDS before the First Start of the New DB2 Subsystem

Notes:

1 Data sharing will be enabled the next time this DB2 subsystem is started.

2 Data sharing is currently disabled for this DB2 subsystem.

3 The highest RBA written is zero, indicating that the DB2 subsystem has never been started.

Each DB2 member records in its BSDS information about the BSDS data set names that are used by all of the DB2 members in the data sharing group. As this DB2 has never been started in data sharing mode, none of this information is recorded in the BSDS yet.

Figure 34 on page 103 shows the contents of the BSDS after the DB2 subsystem has been started.

```
*****************************************************************************
*
*              LOG MAP OF THE BSDS DATA SET BELONGING TO MEMBER 'DBC1    ' OF
*
*****************************************************************************
RELEASE LEVEL OF BSDS -  ACTIVE=2.3 AND ABOVE   ARCHIVE=2.3 AND ABOVE   DDNAME=
   LOG MAP OF BSDS DATA SET COPY 1, DSN=DSNDSGC.DBC1.BSDS01
   LTIME INDICATES LOCAL TIME, ALL OTHER TIMES ARE GMT.
 1       DATA SHARING MODE IS ON
         SYSTEM TIMESTAMP   - DATE=1996.158  LTIME=17:42:04.21
         UTILITY TIMESTAMP  - DATE=1996.158  LTIME=16:45:27.18
         VSAM CATALOG NAME=DSNDSGC
         HIGHEST RBA WRITTEN       0000000686F5  1996.158  21:42:01.8
         HIGHEST RBA OFFLOADED     000000000000
         RBA WHEN CONVERTED TO V4  000000000000
         MAX RBA FOR TORBA         000000000000
         MIN RBA FOR TORBA         000000000000
         STCK TO LRSN DELTA        000000000000
 2       THIS BSDS HAS MEMBER RECORDS FOR THE FOLLOWING MEMBERS:
         HOST MEMBER NAME:            DBC1
 3         MEMBER ID:                 1
           GROUP NAME:                DSNDSGC
           BSDS COPY 1 DATA SET NAME: DSNDSGC.DBC1.BSDS01
           BSDS COPY 2 DATA SET NAME: DSNDSGC.DBC1.BSDS02
ACTIVE LOG COPY 1 DATA SETS
  START RBA/LRSN/TIME   END RBA/LRSN/TIME      DATE    LTIME  DATA SET INFORM
 --------------------  --------------------  --------  -----  ---------------
                 EMPTY DATA SET              1996.158  16:18  DSN=DSNDSGC.DBC
 4  000000000000         000000000000                            STATUS=NEW, REU
 0000.000  00:00:00.0  0000.000  00:00:00.0
                 EMPTY DATA SET              1996.158  16:18  DSN=DSNDSGC.DBC
    000000000000         000000000000                         STATUS=NEW, REU
 0000.000  00:00:00.0  0000.000  00:00:00.0
  000000000000         0000021BFFFF       1996.158  16:18  DSN=DSNDSGC.DBC
    ACFAB5BD0604         ............                         STATUS=NOTREUSA
 1996.158  21:17:33.8  ........  ..........
      .     .     .     .     .     .     .     .     .     .
```

Figure 34. Contents of BSDS after the First Start of the New DB2 Subsystem

Notes:

1 Data sharing is now enabled.

2 Now information about all members of the data sharing group has been recorded in the BSDS. Because there is only one member in this data sharing group, there is information only for this member (DBC1).

Before the CSR phase of DB2 restart, each DB2 member will update its own BSDS with any new log information it has read from the SCA. The log information may be a new BSDS data set name for an existing DB2 member or knowledge of a new DB2 member. At the End-Restart phase of DB2 startup, each DB2 member will update the SCA with its new log information (in this case its BSDS data set name). DB2 then sends XCF notify messages to all active members (if there are any) to read the

SCA. The other members will update their own BSDS to include this new log information. DB2 members that are not started during this process will update their own BSDSs with the revised SCA information when they restart.

3 This is the *originating* member of the data sharing group.

The member ID is assigned when a new member joins the data sharing group. It is stored in the SYSLGRNX table to identify entries that belong to a specific DB2 member. Basically the member ID is a shorthand way of identifying a particular DB2 member, instead of using the 8-byte member name. The member ID is also stored in the SCA and BSDS for identifying member-related information.

A member ID of 1 will always be the originating member of the data sharing group (except when a multimember data sharing group was disabled to non-data-sharing and then data sharing was reenabled using a different member).

DB2 must keep track of the originating member of a data sharing group in order to know which DB2 member's logs to use in the event of a recovery that needs log data from before data sharing was enabled.

4 The BSDS has an expanded format to store the RBA and the LRSN.

6.5 Installation Verification

Once you have created a data sharing group and added a second member, you can test GBP caching, global lock serialization, and concurrency to ensure that the data sharing group is functioning correctly.

To perform these tests, you can use the sample objects that were created in job DSNTEJ1 of the DB2 installation verification process, together with member DSNTESD in *prefix*.SDSNSAMP, which contains SQL that refers to these objects.

6.5.1 Test Group Buffer Pool Caching

Use this procedure to test connectivity to the GBP and ensure that the GBP is operating correctly:

1. Execute the SQL in DSNTESD on one DB2 member, using SPUFI. Specify AUTOCOMMIT=YES on the SPUFI panel.

   Member DSNTESD contains the following SQL:

   a. Count the number of rows in the DSN8410.PARTS table.

   b. Insert five rows into the DSN8410.PARTS table.

c. Select from the DSN8410.PARTS table using an ORDER BY clause to make sure the DB2 member has access to the workfile database.

d. Count the number of rows again in the DSN8410.PARTS table. The number of rows should increase by 5 each time the SQL is run.

2. Using SPUFI, execute the SQL in DSNTESD on a different member in the data sharing group.

   The SQL must be executed on the second member as soon as possible after it was executed on the first member to avoid DB2 having to logically close the page set because of infrequent updates. (The default time between updates before DB2 switches the page set from read-write to read-only is 10 minutes.)

   This will cause DB2 to detect inter-DB2 read-write interest on the table space and index and use the group buffer pool.

   Each time you run the SQL, either on the same member or different members, the ITEM_COUNT should increase by 5.

3. After the SQL has executed on more than one member, issue the following command to see whether the table space and index are using the GBP:

```
=DBGC DISPLAY DATABASE(DSN8D41A) SPACENAM(DSN8S41S,XPARTS) LOCKS
```

   If the P-Lock state is IX or SIX, the page set and index are GBP-dependent. Figure 35 shows the output from the DISPLAY DATABASE command after the SQL was run on each member of a data sharing group.

```
DSNT360I =DBC1 ***********************************
DSNT361I =DBC1 *  DISPLAY DATABASE SUMMARY
               *    GLOBAL LOCKS
DSNT360I =DBC1 ***********************************
DSNT362I =DBC1     DATABASE = DSN8D41A  STATUS = RW
                   DBD LENGTH = 12104
DSNT397I =DBC1
NAME     TYPE PART STATUS            CONNID   CORRID       LOCKINFO
-------- ---- ---- ------------------ -------- ------------ ---------
DSN8S41S TS        RW                                      H(IX,PP,I)
   -                   MEMBER NAME DBC1     (CO)           2
DSN8S41S TS        RW          1                           H(IX,PP,I)
   -                   MEMBER NAME DBC2
XPARTS   IX        RW                                      H(IX,PP,I)
   -                   MEMBER NAME DBC1     (CO)
XPARTS   IX        RW                                      H(IX,PP,I)
   -                   MEMBER NAME DBC2
******* DISPLAY OF DATABASE DSN8D41A ENDED      **********************
```

Figure 35. DISPLAY DATABASE Command Showing Inter-DB2 Read-Write Interest

Notes:

1 The DB2 member holding the lock is shown here. CO indicates that this member is the coordinator for this page set and is responsible for castout processing for the changed pages.

2 The LOCKINFO column indicates the type and duration of the locks being held. An IX (shown by IX) page set P-Lock (shown by PP,I) is currently being held (shown by H) by the indicated member of the data sharing group. (An R in place of the H in the LOCKINFO column indicates that the lock is a retained lock and the DB2 member holding the lock has failed.)

4. Issue the following command to display the status of the GBP:

```
=DBGC DISPLAY GBPOOL(GBP0) GDETAIL
```

Figure 35 on page 105 shows the detailed statistics section of the DISPLAY GBPOOL command after the SQL was run on each member of a data sharing group.

```
DSNB784I =DBB1 GROUP DETAIL STATISTICS
             READS 1
               DATA RETURNED                        = 9
DSNB785I =DBB1     DATA NOT RETURNED 091
                 DIRECTORY ENTRY EXISTED            = 19
                 DIRECTORY ENTRY CREATED            = 22
                 DIRECTORY ENTRY NOT CREATED        = 0, 0
DSNB786I =DBB1   WRITES 1
               CHANGED PAGES                        = 17
               CLEAN PAGES                          = 0
               FAILED DUE TO LACK OF STORAGE        = 0
             CHANGED PAGES SNAPSHOT VALUE           = 0
DSNB787I =DBB1   RECLAIMS
               FOR DIRECTORY ENTRIES                = 0
               FOR DATA ENTRIES                     = 0
             CASTOUTS                               = 12
DSNB788I =DBB1   CROSS INVALIDATIONS
               DUE TO DIRECTORY RECLAIMS            = 0
               DUE TO WRITES                        = 12
DSNB790I =DBB1 DISPLAY FOR GROUP BUFFER POOL GBP0 IS COMPLETE
```

Figure 36. Detailed Statistics Section of DISPLAY GBPOOL Command

Note:

1 Nonzero values in the READS and WRITES values of the display indicate that DB2 is using the GBP successfully for caching.

6.5.2 Test Global Lock Serialization

Use this procedure to verify that locks are acquired and released correctly across the DB2 members in the data sharing group:

1. Execute the SQL in DSNTESD on one member of the data sharing group, using SPUFI. Specify AUTOCOMMIT=NO on the SPUFI panel.

Because you have inserted data into the DSN8410.PARTS table but have not committed, this DB2 member will be holding global locks on the page set and index.

2. Execute the SQL in DSNTESD on a second member of the data sharing group, using SPUFI. Specify AUTOCOMMIT=NO on the SPUFI panel.

   Because the first member holds global locks on the page set and index, the second member must wait to perform the insert.

3. While the second member is waiting, issue a commit on the first member. If you wait too long before the commit, the waiting SPUFI execution on the second member suffers a lock timeout. (The default time a DB2 member will wait for locks before the IRLM will detect a lock timeout is about 60 sec. This time is controlled by the IRLMRWT DSNZPARM parameter for each DB2 subsystem.)

   The SPUFI execution on the second member should now complete, and the ITEM_COUNT will increase by 5.

6.5.3 Test Concurrency

Use the procedure below to test concurrency within the data sharing group. You may need the help of one of your colleagues to complete the procedure.

1. Execute the SQL in DSNTESD at the same time on different members of the data sharing group, using SPUFI. Specify AUTOCOMMIT=YES on the SPUFI panel for each execution.

   Each member of the data sharing group coordinates the inserts to the DSN8410.PARTS table with the other members through global locks.

2. Each time the SQL completes successfully, either on the same member or different members, the ITEM_COUNT should increase by 5.

6.6 Removing Members from the Data Sharing Group

One of the important features of DB2 data sharing is incremental growth, the ability to easily add members to an existing data sharing group. It is also a simple process to remove members from the data sharing group, either permanently or temporarily.

To remove a member from the data sharing group temporarily you merely quiesce the member. The member becomes *dormant* until you restart it again.

The same principle applies when you want to *remove* a member from a data sharing group forever. You only have to quiesce the unwanted member.

You must remember to keep the BSDS for as long as the data sharing group is needed. However, you can delete the log data sets when you are sure they are no longer needed for recovery.

When performing group restart, other DB2 members will have to access the quiesced member′s BSDS. If the quiesced member′s BSDS has been deleted, the restarting DB2 members will not be able to access the BSDS for group restart. The restarting DB2 members will respond with message DSNR020I, indicating that for group restart to continue, the other members must be restarted. You must reply “QUIESCED” to this message. Group restart will continue without using the logs of the quiesced member. You must also reply “YES” to the subsequent message, DSNR030I, confirming that group restart will continue without using all of the DB2 members′ logs.

To review, if you have permanently quiesced a DB2 member from a data sharing group and deleted its BSDS data sets, you must perform the above procedure for a successful group restart. You may be able to reinitialize the BSDS and reinstall the quiesced member into the data sharing group. This procedure will rebuild the BSDS, which can be retained when the DB2 member is once again permanently quiesced.

The logs can safely be deleted only when you are sure they are no longer needed for recovery. The logs may contain updates other members may need for recovery.

To permanently remove a member from a data sharing group, follow these steps:

1. Stop the DB2 subsystem you plan to remove from the data sharing group.
2. From another member in the data sharing group, enter the following commands to ensure that the member has quiesced properly and has not left any outstanding work:

   ```
   DISPLAY GROUP
   DISPLAY UTILITY(*) MEMBER(member-name)
   DISPLAY DATABASE(*) RESTRICT
   ```

   This is very important because knowledge of this member never actually gets removed from the data sharing group. If a DB2 member has to perform group restart to recover the SCA or lock structure, it must read all nonquiesced members′ BSDSs and logs. Therefore if this member is not successfully quiesced, its BSDSs and logs can never be deleted.

3. If there is any unresolved work, you must restart the member with ACCESS(MAINT) and resolve any remaining activity for the member. You may want to optionally create an archive log at this time, which can be sent to your recovery site:

   ```
   ARCHIVE LOG MODE(QUIESCE)
   ```

4. Quiesce the DB2 member again.

   ```
   STOP DB2 MODE(QUIESCE)
   ```

5. You may now optionally dismantle the member.

   Remember to delete the log data sets only when you are certain the data sharing group does not have to read them to recover any DB2 objects.

You can delete the quiesced DB2 member′s workfiles by issuing the DROP command from another member in the data sharing group.

For distributed communications, do not forget to remove entries in any application requester′s definitions for the DB2 member you are removing. For member-specific connections using DB2 V4 remove rows from the SYSIBM.SYSLULIST table. For group-generic connections remove the LU name from generic LU list in VTAM.

Remember that after you have removed a member from the data sharing group and deleted all its data sets, knowledge of that member still remains in the data sharing group. A print of the BSDS for any other member in the data sharing group will show the member record for the removed DB2 member. It will also appear in the DISPLAY GROUP command output forever, with a status of QUIESCED.

The member ID that was allocated to the DB2 member when it first entered the data sharing group will also never be used by any other member. The member ID is recorded in SYSLGRNX and is used to determine which DB2 member the SYSLGRNX entry is for, and therefore which DB2 member′s logs to search. It is also stored in the BSDS of all the members.

6.7 Disabling Data Sharing

In this section we detail some points you have to consider when you disable data sharing. For more information on the actual process of disabling data sharing, we suggest that you refer to Chapter 3, "Disabling DB2 Data Sharing," in *IBM DATABASE 2 for MVS/ESA Version 4 Data Sharing: Planning and Administration* .

Note

Disabling data sharing is a complex procedure. Therefore:

- Do not make a disabling procedure part of your contingency.
- Plan for handling recovery situations.
- For temporary bypasses to data sharing problems, you are much better off moving to one-way data sharing.
- Some situations can be resolved by doing a group restart.

The disabling procedure is included in *IBM DATABASE 2 for MVS/ESA Version 4 Data Sharing: Planning and Administration* in the event you have made a strategic decision to move off data sharing or have a situation in which one-way data sharing is not working. Do not attempt to disable data sharing without a thorough understanding of the procedure.

Before starting to disable data sharing, you have to choose the surviving member, which can be any one of the DB2 members in the data sharing group. Also, save jobs used to migrate to data sharing; they will be needed whether or not you reenable the data sharing group.

When you disable data sharing, you have to cold start DB2. Disabling data sharing makes all logs written before data sharing was disabled unavailable to DB2. Both the logs of the surviving member and the logs of the other members in the data sharing group are unavailable. The surviving member′s logs are unavailable because the information they contain is potentially incomplete. The logs only contain information about activity that took place through the surviving member, so from a DB2 recovery perspective, they are unusable. In a data sharing environment all recovery information is LRSN based. In a non-data-sharing environment, recovery is based on the RBA.

DB2 still requires a cold start even if you have never had more than one member in your data sharing group.

Because all DB2 logs are unavailable, forward recovery using them is not possible, and so we recommend that you include, as part of the testing and timing of your data sharing disablement procedure, a step to ensure the recoverability of your data once data sharing has been disabled. Depending on how much application data you have and the method you are going to use to back it up, this step could take a very long time.

At DB2 restart, after data sharing has been disabled, the last active log will be truncated, and the corresponding LRSN at that time will be used as the value of start RBA because you must be sure you are using an RBA value greater than any LRSN value registered in any page header. The conditional restart control record in the BSDS will show this value as the cold start value.

Figure 37 on page 111 shows the contents of the BSDS after data sharing has been disabled.

```
********************************************************************************
*
*               LOG MAP OF THE BSDS DATA SET BELONGING TO MEMBER 'DBC1    ' OF
*
********************************************************************************
RELEASE LEVEL OF BSDS - ACTIVE=2.3 AND ABOVE   ARCHIVE=2.3 AND ABOVE   DDNAME=
   LOG MAP OF BSDS DATA SET COPY 1, DSN=DSNDSGC.DBC1.BSDS01
   LTIME INDICATES LOCAL TIME, ALL OTHER TIMES ARE GMT.
 1       DATA SHARING MODE IS OFF
         SYSTEM TIMESTAMP   - DATE=1996.164  LTIME=14:17:52.12
         UTILITY TIMESTAMP  - DATE=1996.164  LTIME=14:06:48.27
         VSAM CATALOG NAME=DSNDSGC
 2       HIGHEST RBA WRITTEN       AD0218C39FFF  0000.000  00:00:00.0
         HIGHEST RBA OFFLOADED     00000140DFFF
         RBA WHEN CONVERTED TO V4  0000013A9BA2
 3       MAX RBA FOR TORBA         FFFFFFFFFFFF
 4       MIN RBA FOR TORBA         AD0218C3A000
         STCK TO LRSN DELTA        000000000000
       THIS BSDS HAS MEMBER RECORDS FOR THE FOLLOWING MEMBERS:
         HOST MEMBER NAME:            DBC1
           MEMBER ID:                 1
           GROUP NAME:                DSNDSGC
           BSDS COPY 1 DATA SET NAME: DSNDSGC.DBC1.BSDS01
           BSDS COPY 2 DATA SET NAME: DSNDSGC.DBC1.BSDS02
         HOST MEMBER NAME:            DBC2
           MEMBER ID:                 2
           GROUP NAME:                DSNDSGC
           BSDS COPY 1 DATA SET NAME: DSNDSGC.DBC2.BSDS01
           BSDS COPY 2 DATA SET NAME: DSNDSGC.DBC2.BSDS02
     .     .     .     .     .     .     .     .     .     .
```

Figure 37. Contents of the BSDS for the Surviving Member

Notes:

1 Data sharing is now disabled.

2 The HIGHEST RBA WRITTEN now looks like an LRSN but it is actually an RBA value.

3 The MAX RBA FOR TORBA value now is set to x'FFFFFFFFFFFF'.

4 The MIN RBA FOR TORBA value now equals the RBA used to restart DB2 after data sharing was disabled.

The contents of the BSDSs for the other quiesced members of the now disabled data sharing group remain unchanged.

After the data sharing group is disabled, the group attachment function will be still usable if you do not change the IEFSSNxx to remove the group name. Thus you do not have to change any JCL after disabling data sharing if your JCL is already using the group attach name.

Attention

At this point the other quiesced members of the data sharing group will still be able to start up. If they restart, they will start with data sharing active and use the same catalog, directory, and user page sets as the surviving member after data sharing was disabled. Therefore to avoid the possibility of data corruption, ensure that the quiesced members for the former data sharing group cannot be started. For example, edit the startup procedures JCL to make them inoperative.

6.7.1 Backup Strategy

The allowable window for disabling data sharing and how critical your user data is from a business perspective will determine which of the following backup strategies you choose:

1. Back up all system and user data, using image copy (FULL or INCREMENTAL) or non-DB2 backup. This option is the safest and ensures complete recoverability of all data. However, the timing of it will be determined by the amount of user data you have.

   If your backup procedures are not coordinated by a central team, this might not be a viable option, unless you have a tool that can generate backup procedures dynamically.

2. Back up system data but not all user data. Using image copy (FULL or INCREMENTAL) with SHRLEVEL REFERENCE or non-DB2 backup allows for complete recoverability of the system data. Application data that has not been backed up can be recovered to a point in time (before disabling data sharing), provided you have a full image copy or a non-DB2 backup. The image copy entries are still registered in SYSCOPY, so you can execute RECOVER TOCOPY, but no log records can be processed.

   Backing up system data but not all user data decreases the time required for the backup but limits the scope of recoverability. The only data that you should consider not backing up is data that can be re-created from another source or is static (does not change) and can be recovered to the last available image copy.

   Remember to take referential integrity into consideration when deciding what to back up; always back up tables that are related through referential integrity.

Consider having more than one copy generated by the COPY utility for all data at your site in case primary image copy is not available.

6.7.2 Lost Log Analysis

If you need access to the data on logs that are now unavailable to DB2, you can use a log analysis tool or program. The log analysis tool or program could generate the SQL statements recorded on the log, which you could then use to update your user data. Consider this option only in extreme circumstances as there are a number of associated pitfalls:

- Generation of complete SQL statements is possible only if a table has the DATA CAPTURE CHANGES option active or your application is doing insert processing.

- If you are in more than one-way data sharing, you might, depending on how sophisticated your log analysis tool is, have to decide on the sequence in which the SQL generated from the different members′ logs should be applied.
- If the table you are analyzing was updated in the same unit of work as other tables, you have to ensure some kind of data integrity across all of the tables.

6.8 Reenabling Data Sharing

To reenable data sharing for a DB2 subsystem where data sharing was previously disabled, you have to perform a subset of the procedure you used to originally enable data sharing.

You can only reenable data sharing for the surviving member of a previously disabled data sharing group. After data sharing has been reenabled, you can restart other quiesced members or install new DB2 subsystems into the data sharing group.

You cannot use the ENABLE process of the installation CLIST to reenable data sharing. The coupling facility definitions and the jobs used to originally enable data sharing remain intact and do not have to be re-created.

To reenable data sharing for a surviving DB2 subsystem, follow these steps:

1. Edit and run job DSNTIJGF.

   This job runs the change log inventory utility (DSNJU003) to insert a data sharing reenable record for the surviving member.

2. Change the IRLM JCL procedure to specify SCOPE=GLOBAL.

3. Ensure that there are no data sharing structures in the coupling facility that are remaining from any previous execution of this data sharing group.

   This is very important as DB2 will not allow you to start DB2 if an old SCA structure still exists in the coupling facility.

4. Start the surviving DB2 subsystem, using the DSNZPARM member that was used when data sharing was previously enabled.

   Although it is not mandatory to use the same DSNZPARM member that was used when data sharing was previously disabled, the DSNZPARM member that you do use to start DB2 must specify that data sharing is enabled. Other parameters such as the CATALOG ALIAS must also remain the same.

   During the startup you will be asked to start all other members that were not quiesced at the time you disabled data sharing. You must start all these members because DB2 must perform a group restart to rebuild the coupling facility structures. At reenable time, you force all other nonquiesced members to start up because you force them to do cold start.

5. Edit and run job DSNTIJFT to re-create the workfile database.

6. Verify that the data sharing group is functioning correctly.

Data sharing is now reenabled for the surviving member, and DB2 will use the LRSNs again instead of the log RBAs.

The surviving DB2 member becomes the *originating* member for the data sharing group. However, the member ID for this member is the same as that allocated to it at the time it joined the data sharing group. For example, if the member ID of the surviving member was 2 and data sharing was reenabled, member 2 will be the originating member of the reenabled data sharing group.

This is not a problem as you cannot use any DB2 logs to recover to a point-in-time when data sharing was previously enabled, even when data sharing is now disabled or even reenabled again. In these cases you can only recover using the TOCOPY keyword.

DB2 does not have a direct way of telling you who is the originating member of a data sharing group when member 1 has been removed from the data sharing group and data sharing has been reenabled using another member. The REPORT utility can indirectly tell you whether the table space or partition was updated by the surviving member when data sharing was disabled. The report output gives every member′s SYSLGRNX records and the member that has an SYSLGRNX record with member ID = 0 is the originating member.

However, the member ID is only 0 when data sharing has been disabled. At that time, there is no notion of member outside of the surviving member. Once you have reenabled data sharing, the originating member is the member that survived the disable.

Any new member entering the reenabled data sharing group will be allocated a new member ID just as before.

If a member was quiesced and removed from the data sharing group in the original data sharing group, it will still be listed in the DISPLAY GROUP command output now that data sharing has been reenabled with a different member as the surviving member.

Figure 38 on page 115 shows the contents of the BSDS after the DB2 subsystem has been reenabled for data sharing and successfully restarted.

```
*****************************************************************************
*
*              LOG MAP OF THE BSDS DATA SET BELONGING TO MEMBER 'DBC2    ' OF
*
*****************************************************************************
RELEASE LEVEL OF BSDS -  ACTIVE=2.3 AND ABOVE   ARCHIVE=2.3 AND ABOVE   DDNAME=
   LOG MAP OF BSDS DATA SET COPY 1, DSN=DSNDSGC.DBC2.BSDS01
   LTIME INDICATES LOCAL TIME, ALL OTHER TIMES ARE GMT.
 1       DATA SHARING MODE IS ON
         SYSTEM TIMESTAMP   - DATE=1996.193  LTIME=15:27:23.77
         UTILITY TIMESTAMP  - DATE=1996.193  LTIME=15:11:13.40
         VSAM CATALOG NAME=DSNDSGC
         HIGHEST RBA WRITTEN       AD26902AB9EA  1996.193  19:01:48.6
         HIGHEST RBA OFFLOADED     0000002D7FFF
 2       RBA WHEN CONVERTED TO V4  000000EBED16
 3       MAX RBA FOR TORBA         AD26902AB9EA
 4       MIN RBA FOR TORBA         AD2690291000
 5       STCK TO LRSN DELTA        000000000000
       THIS BSDS HAS MEMBER RECORDS FOR THE FOLLOWING MEMBERS:
 6       HOST MEMBER NAME:            DBC2
           MEMBER ID:                 2
           GROUP NAME:                DSNDSGC
           BSDS COPY 1 DATA SET NAME: DSNDSGC.DBC2.BSDS01
           BSDS COPY 2 DATA SET NAME: DSNDSGC.DBC2.BSDS02
 6       HOST MEMBER NAME:            DBC1
           MEMBER ID:                 1
           GROUP NAME:                DSNDSGC
           BSDS COPY 1 DATA SET NAME: DSNDSGC.DBC1.BSDS01
           BSDS COPY 2 DATA SET NAME: DSNDSGC.DBC1.BSDS02
       .     .     .     .     .     .     .     .     .     .
```

Figure 38. Contents of the BSDS after Data Sharing Has Been Reenabled

Notes:

1 Data sharing has now been reenabled.

2 The RBA WHEN CONVERTED TO V4 field is set when a DB2 V3 subsystem is migrated to V4, or when a DB2 V4 subsystem is installed.

The value is never changed after that and has nothing to do with data sharing. It just happens to be reported along with all of the other RBA values in the BSDS that are related to data sharing.

DB2 V4 tracks modifications to pages differently than in V3 for incremental image copy. This RBA indicates when the change occurred with respect to the log. The change allows DB2 to improve the performance of both full image copy and incremental image copy significantly.

3 The MAX RBA FOR TORBA field is the RBA of the originating member at the time data sharing was enabled or reenabled.

The recover TORBA keyword cannot be used to recover beyond this point. The recover TOLOGPOINT keyword must be used to perform a point-in-time recovery beyond this point. Both the TOLOGPOINT and TORBA keywords can be used before this point.

Any value that is specified in the recover utility TOLOGPOINT keyword that is lower than the MAX RBA FOR TORBA value will be considered as an RBA for a point-in-time recovery. Similarly, any value specified that is greater than the MAX RBA FOR TORBA value will be considered as an LRSN for a point-in-time recovery.

4 The MIN RBA FOR TORBA field is the RBA at which DB2 continued to log after the cold start was performed to disable data sharing.

The recover utility cannot recover any DB2 objects before this RBA value. However, the recover TORBA and TOLOGPOINT keywords can be used to perform a point-in-time recovery beyond this point. You can only specify the TOCOPY keyword for point-in-time recoveries before this RBA value.

5 The STCK TO LRSN DELTA field is only used when the RBA values before data sharing was enabled or reenabled are higher than the LRSN that will be used when data sharing will be enabled or reenabled. In this case a delta value must be used to step up the LRSN values.

As of today, no customer has an RBA like this even if it has reenabling data sharing. Unless you can generate log records faster than 16 nanoseconds, the log RBA will not grow faster than the Sysplex Timer.

The delta is kept in the BSDS and the SCA for each DB2 member. All members will keep the same delta because the LRSN must be kept in synch for the entire data sharing group.

6 In this case member DBC1 was quiesced and data sharing was disabled from member DBC2. Data sharing was then reenabled for member DBC2.

The BSDS still holds information for members DBC1 and DBC2. Each member still has the same member IDs assigned although member DBC2 is now the originating member. When data sharing is reenabled, the fields in the BSDS are only updated for the originating member. Member DBC1′s BSDS is still the same as it was when DBC1 was quiesced. It will be updated when DBC1 is successfully restarted.

Chapter 7. Merging Existing DB2 Data into One Data Sharing Group

A common situation in customer environments is that the production data is split over several DB2 subsystems. If one of the DB2 subsystems requires data from another DB2 subsystem, the two subsystems communicate with each other through the DDF. In a data sharing environment you can merge the existing DB2 production data of different subsystems into one data sharing group.

The concept of merging DB2 subsystems is simple:

1. You choose one of the existing subsystems as the originating member of the data sharing group. This implies that you use the catalog and directory of this subsystem as the catalog and directory of the data sharing group.
2. Then you move the data definitions and the data from the other subsystems to the originating member.
3. Finally you add a new member to the data sharing group, using the procedure to add new members to a group. The applications that ran on the stand-alone DB2 should now run on the new member.

The procedure of merging subsystems requires careful planning because DB2 does not provide direct support to merge existing DB2 data or subsystems into a data sharing group.

In this chapter we describe three procedures for merging data. The procedure you choose depends mainly on the amount of data and the number of database objects that have to be merged. In our description of these procedures, we assume that all existing DB2 subsystems are non-data-sharing V4 subsystems.

Note that the procedures we describe can be used to copy all table spaces within a database from one DB2 to another DB2 without changing the table space and table definitions. If you want to merge data of tables from different DB2 subsystems, these procedures do not apply.

To simplify our explanation we use the term *source* when referring to the stand-alone DB2 from which you are moving data and the term *target* for the data sharing group to which you are moving data.

Independent of the procedure you use for merging the subsystems, you have to move the catalog definitions of the data objects from the source DB2 to the catalog of the target DB2.

DB2 does not provide an automated way of moving catalog definitions from one DB2 to another DB2 or to a data sharing group. If you have procedures or tools to move catalog definitions for data objects from your test environment to your production environment, you can use them to move the desired catalog definitions from the source to the catalog of the target. If you do not use such procedures or tools, you have to query the catalog of the source DB2 to find out all of the necessary information about the data objects.

7.1 How to Move the Data

As explained in 7.3, "Creating Data Objects on the Target DB2" on page 120, each table space has internal identifiers (object identifiers, or OBIDs) assigned by DB2 when you create the objects. OBIDs that are common for the table space are stored in the table space header page. OBIDs for tables are stored in the row header. When moving data from one DB2 subsystem to another, generally these OBIDs are different.

You can use the DSN1COPY utility with OBID translation to change all OBIDs. That seems to be the simplest way. Using DSN1COPY implies that you make a copy of all of the data you want to merge. You have to provide DASD space for the copies, and you use CPU time for the copy jobs. In addition, the translation of OBIDs is an I/O-intensive process because DSN1COPY must translate the OBID in the header of every row. Thus the procedure using the DSN1COPY utility and translating OBIDs is practicable when the amount of data that has to be merged is of manageable size.

If you want to use DSN1COPY with translation of OBIDs and you have too much to copy all at once, it might be possible to use incremental image copies as input for the DSN1COPY utility.

Independent of the input data set that is used for the DSN1COPY utility, make sure to assign sufficient utility buffers. In Version 4 the default number of utility buffers was increased from 8 to 20. For that reason we had to increase the region size on the utility job statement. You can speed up the DSN1COPY utility a lot if you increase the number of utility buffers to a number greater than 20. Make sure to increase the region size as well.

One other procedure to move data avoids using the resource-intensive DSN1COPY utility. Most of the time, when you create a table, you can force DB2 to use a table identifier specified by you. In this case, you can use the existing copy of the data and execute the REPAIR utility to update the header page. With this procedure you use the existing data sets (no copies) of the source DB2. You have to use the REPAIR utility to change the database ID (DBID) and the OBIDs for table spaces and indexes in the header page of the page sets. You also use the REPAIR utility to change the LEVELID of the page sets before restarting them on the target DB2.

The REPAIR utility procedure is not suitable for Type 2 indexes because of their physical structure. In contrast to the Type 1 index where the DBID and OBID of the indexes are located in the index page set header page, the OBID of the Type 2 index is located in the index page set header page as well in each index page. Thus you have to change the OBID using the REPAIR utility on the index page set header page and on each index page, which seems to be a ridiculous expense. At the moment there is no actual technical reason why each index page of a Type 2 index contains the OBID.

Once you have planned how to move the table spaces and Type 1 indexes, you have to decide how to move the Type 2 indexes:

- You can copy the Type 2 indexes with the DSN1COPY utility and translate the OBID.
- You can alter the Type 2 indexes to Type 1 indexes subpage 1 and recover them. Using Type 1 indexes, you can change the OBID with the REPAIR utility.
- You can create the Type 2 index on the target DB2 after you have moved the table space.

7.2 Choosing a DB2 Subsystem As the Originating Member

Before you can start merging the systems, you must choose one subsystem′s catalog as the originating catalog of the data sharing group. Analyze all existing stand-alone subsystems and choose as the originating member the DB2 subsystem that has the most database objects, to minimize the amount of database objects that have to be moved. If you are using DSN1COPY, you must also consider the size of the objects. In this case you should choose as the originating member the DB2 subsystem that has the most data. If all of your stand-alone DB2 systems have almost the same number of database objects or the same amount of data, you can choose any of the systems as your originating member.

Consider the values of the log RBAs of the existing subsystems. Use the print log map utility (DSNJU004) to find out the values (refer to the *Utility Guide and Reference* for a description of this utility). Figure 39 shows an output of the BSDS of a DB2 subsystem. You can choose:

- The DB2 subsystem that has the highest RBA as the originating member
- Any of the stand-alone DB2s as the originating member. In this case you must use DSN1COPY with the RESET option to reset the log RBA in each data and index page to zero when you move databases from other DB2 subsystems to the data sharing group. This is the log RBA of the last update to the page.

The safest way is to choose the DB2 subsystem that has the highest RBA as the originating member.

```
LOG MAP OF BSDS DATA SET COPY 1, DSN=DSNDSGA.DBA1.BSDS01
LTIME INDICATES LOCAL TIME, ALL OTHER TIMES ARE GMT.
      SYSTEM TIMESTAMP   - DATE=1996.165  LTIME=14:40:54.30
      UTILITY TIMESTAMP  - DATE=1996.151  LTIME=20:08:36.88
      VSAM CATALOG NAME=DSNDSGA
      HIGHEST RBA WRITTEN       0000014AC49C  1996.165  18:39:05.7
      HIGHEST RBA OFFLOADED     000001460FFF
      RBA WHEN CONVERTED TO V4  000000EBED16
      MAX RBA FOR TORBA         000000EBED16
      MIN RBA FOR TORBA         000000000000
```

Figure 39. BSDS Listing: Checking the Highest RBA Written

7.3 Creating Data Objects on the Target DB2

After one of the DB2 subsystems becomes the originating member called the target DB2, you have to make sure that there are no naming conflicts among the objects on the target DB2. Immediately after defining the objects to the target DB2, you must leave them in an unavailable state, to prevent updating to them on the target DB2. You can use the STOP command to make the database or the table space unavailable. You must leave the objects in the unavailable state until you actually move the data to the target DB2.

After moving the objects to the target DB2 you must provide the required authorizations in the target DB2.

Before creating the data objects on the target DB2 you must be aware of how the internal identifiers are used. These are the internal identifiers of data objects that are important for the merging procedure:

- The DBID is an internal identifier for a database. The DBID is unique within a DB2 subsystem. The DBID is assigned by DB2 when you create a database.
- The OBID is an internal identifier for an object in one database. It must be unique in the database. The object can be a table space, a table, an index, or a referential constraint relationship.
- The PSID (page set ID) is the unique identifier for the page set. You can find it in the PSID column of SYSIBM.SYSTABLESPACE. Note that there is a column named OBID in the SYSIBM.SYSTABLESPACE catalog table. That column contains the OBID of a file descriptor and must not be confused with the PSID.
- The ISOBID (index page set identifier) is the unique identifier for the index page set. You can find it in the ISOBID column of SYSIBM.SYSINDEXES. Note that there is a column named OBID in the SYSIBM.SYSINDEXES. That column contains the identifier of a fan set descriptor and must not be confused with the ISOBID.
- The LEVELID is an option of the REPAIR utility (see the *Utility Guide and Reference* for a description). This option sets the level identifier of the named table space or partition to a new identifier. You must use the LEVELID option to accept the use of a down-level data set that has been moved from another DB2.

In summary, if you create a database on your target DB2, that database is identified on the target DB2 with a DBID that, in most cases, is not the same DBID with which the database was identified in the source DB2. If you create a table space, it gets a unique OBID and PSID within this database.

To find out the DBID, the PSID of the table spaces, and the ISOBID of the indexes in the source DB2, you have to query the catalog of the source DB2.

Figure 40 on page 121 shows a sample query to get the DBID, OBID, and PSID of a table space.

```
---------+---------+---------+---------+---------+---------+-----
-- query to find out catalog information about table space
--
SELECT NAME,CREATOR,DBID,OBID,PSID
    FROM SYSIBM.SYSTABLESPACE
    WHERE DBNAME LIKE 'RESTDB%'
    ORDER BY NAME;
---------+---------+---------+---------+---------+---------+-----
NAME      CREATOR     DBID    OBID    PSID
---------+---------+---------+---------+---------+---------+-----
RESTTS01  DB2RES       263       1       2
DSNE610I NUMBER OF ROWS DISPLAYED IS 1
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
---------+---------+---------+---------+---------+---------+-----
--
---------+---------+---------+---------+---------+---------+-----
DSNE617I COMMIT PERFORMED, SQLCODE IS 0
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 0
```

Figure 40. Sample Query: Catalog Information about Table Spaces

Figure 41 shows a sample query to get the DBID, OBID, and ISOBID of indexes.

```
---------+---------+---------+---------+---------+---------+---------
-- To test the catalog to find out index information
--
   SELECT CREATOR, NAME, INDEXSPACE, DBID, OBID, ISOBID
     FROM SYSIBM.SYSINDEXES
      WHERE DBNAME LIKE 'RES%'
      ORDER BY NAME, CREATOR;
---------+---------+---------+---------+---------+---------+---------
CREATOR   NAME              INDEXSPACE    DBID    OBID  ISOBID
---------+---------+---------+---------+---------+---------+---------
DB2RES    XRESDEPT1         XRESDEPT       263       6       7
DB2RES    XRESDEPT2         XRES1FYC       263       8       9
DB2RES    XRESDEPT3         XRES1Q0F       263      10      11
DSNE610I NUMBER OF ROWS DISPLAYED IS 3
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
---------+---------+---------+---------+---------+---------+---------
--
---------+---------+---------+---------+---------+---------+---------
DSNE617I COMMIT PERFORMED, SQLCODE IS 0
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 0
```

Figure 41. Sample Query: Catalog Information about Indexes

In addition you must know the OBIDs of the tables in the existing table space in the source DB2. You can query the catalog of the source DB2 to find out these OBIDs. Figure 42 on page 122 shows a sample query to get the OBIDs of tables.

```
---------+---------+---------+---------+---------+---------+---
--
SELECT NAME,CREATOR,DBID,OBID
  FROM SYSIBM.SYSTABLES
    WHERE TSNAME LIKE 'RESTTS%'
     ORDER BY NAME;
---------+---------+---------+---------+---------+---------+---
NAME                CREATOR     DBID    OBID
---------+---------+---------+---------+---------+---------+---
RESDEPT             DB2RES       263       5
DSNE610I NUMBER OF ROWS DISPLAYED IS 1
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
---------+---------+---------+---------+---------+---------+---
---------+---------+---------+---------+---------+---------+---
DSNE617I COMMIT PERFORMED, SQLCODE IS 0
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 0
```

Figure 42. Sample Query: Catalog Information about Tables

Table 10 shows the summarized results of the sample queries:

Table 10. Results of Sample Queries: Source DB2

Name	Object	DBID	OBID	PSID/ISOBID
RESTDB01	Database	263 (x'107')	-	-
RESTTS01	Table space	263	1	2
RESDEPT	Table	263	5	-
XRESDEPT1	Index	263	6	7
XRESDEPT2	Index	263	8	9
XRESDEPT3	Index	263	10	11

The OBIDs of the objects within a database are assigned by DB2 in ascending order according to the order in which you create them.

If the number of data objects is of manageable size and you have decided to use the DSN1COPY utility with OBID translation, you can enter the CREATE statements for the data objects on the target DB2 in any order. DB2 assigns new OBIDs for all of the objects you create. During the copy process, the DSN1COPY utility translates the source OBIDs into the target OBIDs.

If you have decided to use DSN1COPY without OBID translation (you still have to read the entire page set) or the REPAIR utility to update the header page, you must ensure that tables on the target DB2 have the same OBID they have on the source DB2. Therefore you must specify the OBID clause in the CREATE statement when you create the tables in the target DB2. Nevertheless, the OBIDs you want to assign to the tables must be available within the database. Because other objects in a database, such as table spaces, indexes, and constraints, have their individual OBIDs assigned, the order of the CREATE statements for the objects is very important. We recommend the following order:

- Create the database.
- Create the table space, if an explicit table space name is specified.
- Create all tables within this table space, using the OBID clause.
- Create indexes and referential constraints if necessary.

In our example we create the database, table space, table, and indexes and assign the table OBIDs the same value as they have in the source DB2 (refer to Figure 43).

```
SET CURRENT SQLID = 'DB2RES';
CREATE DATABASE RESTDB01;
CREATE TABLESPACE RESTTS01 IN RESTDB01
       USING STOGROUP stogroup name;
CREATE TABLE RESDEPT
       (column1,
        column2,..)
IN RESTDB01.RESTTS01
OBID 5;
CREATE UNIQUE INDEX DB2RES.XRESDEPT1
                 ON DB2RES.RESDEPT....;
CREATE INDEX DB2RES.XRESDEPT2
                 ON DB2RES.RESDEPT ...;
CREATE TYPE 2 INDEX DB2RES.XRESDEPT3
                 ON DB2RES.RESDEPT ...;
```

Figure 43. Sample CREATE Statements

Table 11 shows the result of querying the catalog of the target DB2 using the sample queries shown in Figure 40 on page 121 , Figure 41 on page 121 , and Figure 42 on page 122.

Table 11. Results of Sample Queries: Target DB2

Name	**Object**	**DBID**	**OBID**	**PSID/ISOBID**
RESTDB01	Database	264 (X'108')	-	-
RESTTS01	Table space	264	3	4
RESDEPT	Table	264	5	-
XRESDEPT1	Index	264	6	7
XRESDEPT2	Index	264	8	9
XRESDEPT3	Index	264	10	11

If you use the REPAIR utility, it is very important that the database and table space names on the target DB2 match the names on the source DB2 because of the special way DB2 builds up the VSAM data set names from the catalog definitions.

We suggest that you execute the REORG utility on the table spaces of the source system before moving them to the target system. If you have to make changes in the table definition, such as adding a column, you *must* execute the REORG utility on those table spaces, because, if you add a column, only the description of the table changes. The data still does not match the new description.

7.4 Copying Data with DSN1COPY and OBID Translation

You can use the DSN1COPY utility to copy a DB2 VSAM data set (such as a table space) to other DB2 VSAM data sets (such as another table space). Note that the output data set is a page-for-page copy of the input data set. If the intended use of DSN1COPY is to move or restore data, the data definitions for the source and target table spaces, tables, and indexes must be identical. Refer to Chapter 3 of *Utility Guide and Reference* for a description of the DSN1COPY utility. To use DSN1COPY to copy data, follow these steps:

1. Stop the table space on the source and target DB2.

   This forces the pages to disk on the source DB2 and prevents updates to the table space.

2. Use the DSN1COPY utility with the OBIDXLAT option to translate the OBIDs (database, table space, tables) from the OBIDs on the source DB2 to the OBIDs on the target DB2. Use the RESET option to reset the log RBA of the last update to the page in each data or index page to 0.

   You must use the RESET option to be able to process the output file from the DSN1COPY operation (the table space and/or index spaces of the source DB2) on a DB2 subsystem with a different recovery log. If you do not specify the RESET option, you can get abends during subsequent update activity. The abend reason code of *00C200C1* indicates that the specified RBA value is outside the valid range of the recovery log of the DB2 subsystem.

Figure 44 shows the example we used in our project.

```
//EXECUTE EXEC PGM=DSN1COPY,PARM='OBIDXLAT,RESET'
//STEPLIB  DD  DISP=SHR,DSN=DSN410.SDSNLOAD
//SYSPRINT DD  SYSOUT=*
//SYSUT1   DD  DSN=DSNDSGA.DSNDBC.RESTDB01.RESTTS01.I0001.A001  1
//             DISP=OLD
//SYSUT2   DD  DSN=DSNDSGB.DSNDBC.RESTDB01.RESTTS01.I0001.A001  2
//             DISP=OLD
//SYSXLAT  DD  *
263,264  3
2,4  4
5,5 5
/*
```

Figure 44. Sample DSN1COPY Utility

Notes:

1 The input data set is the table space on the source DB2.

2 The output data set is the table space on the target DB2.

3 263/264 is the DBID of the database on the source/target DB2.

4 2/4 is the PSID of the table space on the source/target DB2.

5 5 is the table OBID on both the source and target DB2s.

When you have restored a table space to a certain point, using DSN1COPY, there are two ways of restoring the indexes. The easiest and safest way is to use the RECOVER INDEX utility, which reconstructs the indexes from the data. However, for table spaces with many rows, the RECOVER INDEX utility might take a long time.

The alternative is to use DSN1COPY for the indexes. If you use the OBIDXLAT option for the data, you must also use it for the indexes. Table 10 on page 122 (source DB2) and Table 11 on page 123 (target DB2) show the OBIDs of our sample indexes. Also, you must copy the indexes at the same time as the data; otherwise, inconsistencies may be present.

After the copy operations you can start the table space on the target DB2 for R/W access.

We strongly recommend that you take full image copies of all copied data, because this is the earliest time to which you can recover after the merge.

To complete the merge procedure, on the target DB2 execute the RUNSTATS utility and bind all plans and packages that were bound on the source DB2.

7.5 Moving Data With REPAIR Utility

Here are the steps that you have to follow to move data with the REPAIR utility.

1. Stop the database on the source and target DB2.
2. If the data objects you have created on the target DB2 are storage group defined, you have to delete the data sets that DB2 allocated.
3. Rename the source data sets to the high-level qualifier of the target DB2.

   You can use the ALTER option of IDCAMS to rename the data sets. Make sure that you rename both the cluster and the data part of the VSAM data set.

   Note that ALTER is not an uncatalog or catalog type of operation. It is essentially a rename inside the catalog. It does not cause the cluster to be cataloged in another catalog. Therfore you cannot rename the high-level qualifier of data sets when the alias of the old high-level qualifier points to a different user catalog from the new catalog on the target system.

If you have different user catalogs, you must copy the source data set, to the target data set, using techniques such as IDCAMS REPRO or a DFDSS copy.

4. For our example, the new DBID is 264 (X′108′), and the new PSID is 4(X′4′) for the table space on the target DB2 (Table 11 on page 123).
5. Use the REPAIR utility to change those identifiers in the page set header page to match the new identifiers on the target DB2, and specify the LEVELID option.

First we consider the procedure for nonpartitioned table spaces. Figure 45 shows the page set header page of the sample table space printed with the DSN1PRNT utility in hexadecimal format.

```
 ¢  >  SYSUT1   SHR¢  DSNDSGA.DSNDBC.RESTDB01.RESTTS01.I0001.A001
DSN1999I START OF DSN1PRNT FOR JOB DB2RES2J RUNPRNT
DSN1989I DSN1PRNT IS PROCESSED WITH THE FOLLOWING OPTIONS:
          4K/NO IMAGECOPY  /NUMPARTS =   0/NO FORMAT/NO EXPAND/NO SWONLY/   PRINT/NO VALUE
DSN1998I INPUT  DSNAME = DSNDSGA.DSNDBC.RESTDB01.RESTTS01.I0001.A001 , VSAM

***  BEGINNING OF PAGE NUMBER 000000 ***
0000 10AD089D D0550C02 00000018 01070002   00000001 00C70000 00000000 00000001  *....................G..........*
0020 01010000 00000000 C4C2C1F2 00011000   00000000 00000000 00010005 00000000  *........DBA2....................*
0040 00440005 00000000 19960617 18445636   1655D9C5 E2E3E2C7 F0F1C4E2 D5C4E2C7  *.................RESTSG01DSNDSG*
0060 C140AD08 9EC998AE 00000000 00000000   00000000 00000000 00000000 00000000  *A ...I..........................*
     LINES ARE ALL ZERO.
0160 00000000 00000000 AD089EC9 98AEAD08   9EC5CFBF AD089DD0 33C80000 00000000  *...........I.....E.......H......*
     LINES ARE ALL ZERO.
0FE0 00000000 00000000 00000000 00000000   00000000 00000000 00000000 000000D5  *..............................N*

DSN1994I DSN1PRNT COMPLETED SUCCESSFULLY,  00000001 PAGES PROCESSED
```

Figure 45. DSN1PRNT of Page Set Header Page of a Nonpartitioned Table Space

The 2-byte fields start at offset X′0C′:

- HPDDBID is the internal DBID of the database in which the page set is defined
- HPGPSID is the internal OBID within the DBID, also known as the PSID.

You want to change the source DBID (263 / X′107′) and PSID (2 / X′2′) to the target DBID (264 / X′108′) and PSID (4 / X′4′). Figure 46 shows an example of the REPAIR utility.

```
//DB2RES2J JOB (999,POK),NOTIFY=DB2RES2,
//       CLASS=A,MSGCLASS=T,MSGLEVEL=(1,1),TIME=1440
//*-------------------------------------------------------------*//
//*  SAMPLE FORMAT :                                             *//
//*  REPAIR LOCATE TABLESPACE RESTDB01.RESTTS01 PAGE X'00'       *//
//*    VERIFY OFFSET X'0C' DATA X'DBIDPSID'  OLD DESCRIPTOR      *//
//*    REPLACE OFFSET X'0C' DATA X'DBIDPSID' NEW DESCRIPTOR      *//
//*-------------------------------------------------------------*//
//UTIL EXEC DSNUPROC,SYSTEM=DBB1,UID=REPTS,
//        LIB='DSN410.SDSNLOAD'
//DSNUPROC.SYSIN    DD  *
REPAIR LOCATE TABLESPACE RESTDB01.RESTTS01  PAGE X'00'
  VERIFY OFFSET X'0C'  DATA X'01070002'
   REPLACE OFFSET X'0C'  DATA X'01080004'
   DUMP  LENGTH 100
```

Figure 46. REPAIR Utility

For partitioned table spaces the procedure is slightly different. A partitioned table space has one header page in each partition. The header page for each partition is page number 0. The internal identifiers are located at offset X′0C′ in each header page, as they are for nonpartitioned table spaces.

When using the REPAIR utility to move data, you have to update the header page of all partitions. For this utility, if the table space is not partitioned, the PAGE option of the LOCATE block specifies only the page number. If the table space is partitioned, the PAGE option specifies a partition number and the page number in the partition.

The PAGE option is specified as a string of 3 bytes specified as a hexadecimal number. The partition number is specified in certain high-order bits and the page number is specified in the low-order bits.

The number of high-order bits to represent the partition number depends on the number of partitions and on table space page size as shown in Table 12.

Table 12. Number of High-Order Bits Used to Specify the Partition Number

Number of Partitions	**For 4 KB Page Size**	**For 32 KB Page Size**
1 to 16	4	7
17 to 32	5	8
33 to 64	6	9

As you always want to update page 0 of each partition, the low-order bits are always 0.

For the REPAIR utility, the first partition is coded as 0, the second partition is coded as 1, the third partition is coded as 2, and so on.

For example, if your table space has 15 partitions with 4 KB pages and you need to specify partition number 13 (the 14th partition), you use the four high-order bits to specify the partition number and the remaining low-order bits to specify the page number within the partition (which is 0 in our case). The following example shows the bit configuration:

```
1101 0000 0000 0000 0000 0000
\__/      page number
 |
partition number
```

All low-order bits should be 0 because you want to update page 0. The final value for the PAGE option specification in hexadecimal representation is:

```
PAGE X'D00000'
```

If your table space has 40 partitions with 4 KB pages and you need to specify partition number 13 (the 14th partition), you use the six high-order bits to specify the partition number and the remaining low-order bits to specify the page number within the partition (which is 0 in our case). The following example shows the bit configuration:

```
0011 0100 0000 0000 0000 0000
\_____/      page number
   |
 partition number
```

All low-order bits should be 0 because you want to update page 0. The final value for the PAGE option specification in hexadecimal representation is:

```
PAGE X'340000'
```

So it is important to know the number of many partitions and the page size for the table space. Figure 47 shows the query from the SYSIBM.SYSTABLESPACE catalog table to obtain this information.

```
---------+---------+---------+---------+---------+---------+---------+-
-- query to find out catalog information about table space           --
--                                                                   --
SELECT NAME,CREATOR,DBID,OBID,PSID,partitions,pgsize
    FROM SYSIBM.SYSTABLESPACE
    WHERE DBNAME LIKE 'RESTDB%'
    ORDER BY NAME;
---------+---------+---------+---------+---------+---------+---------+-
NAME      CREATOR     DBID    OBID    PSID  PARTITIONS  PGSIZE
---------+---------+---------+---------+---------+---------+---------+-
RESTTP01  DB2RES       264       1       2           4       4
RESTTS01  DB2RES       263       1       2           0       4
RESTTS02  DB2RES       263       3       4           0       4
DSNE610I NUMBER OF ROWS DISPLAYED IS 3
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
---------+---------+---------+---------+---------+---------+---------+-
--                                                                   --
---------+---------+---------+---------+---------+---------+---------+-
DSNE617I COMMIT PERFORMED, SQLCODE IS 0
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 0
```

Figure 47. SPUFI Output of SYSIBM.SYSTABLESPACE

Note that if you intend to use REPAIR to update a page other than page 0, the PAGE option specification has another algorithm which is described in the *Utility Guide and Reference*.

Figure 48 on page 129 shows a DSN1PRNT output of the page set header page of the fourth partition of a table space.

```
DSN1999I START OF DSN1PRNT FOR JOB DB2RES2J RUNPRNT
DSN1989I DSN1PRNT IS PROCESSED WITH THE FOLLOWING OPTIONS:
         4K/NO IMAGECOPY  /NUMPARTS = 004/NO FORMAT/NO EXPAND/NO SWONLY/   PRINT/NO VALUE
DSN1998I INPUT  DSNAME = DSNDSGA.DSNDBC.RESTDB02.RESTTP01.I0001.A004 , VSAM

***  BEGINNING OF PAGE NUMBER 300000 ***
0000 00000000 20089102 30000018 01080002   00300001 00C70000 00000000 00000001  *....................G..........*
0020 01010000 00000000 C4C2C1F2 00011000   00000004 00000000 00010003 00000000  *........DBA2...................*
0040 0059000E 00000000 19960618 16291667   8740D9C5 E2E3E2C7 F0F1C4E2 D5C4E2C7  *................ RESTSG01DSNDSG*
0060 C140AD09 C183B04B 00000000 00000000   00000000 00000000 00000000 00000000  *A ..A...........................*
     LINES ARE ALL ZERO.
0160 00000000 00000000 AD09C183 B04BAD09   C16C10D8 AD09C191 B81C0000 00000000  *..........A.....A%.Q..A.........*
     LINES ARE ALL ZERO.
0FE0 00000000 00000000 00000000 00000000   00000000 00000000 00000000 000000C5  *..............................E*

DSN1994I DSN1PRNT COMPLETED SUCCESSFULLY,  00000001 PAGES PROCESSED
```

Figure 48. DSN1PRNT of Page Set Header Page of a Partitioned Table Space

Figure 49 shows an example of the REPAIR utility for the fourth partition.

```
//DB2RES2J JOB (999,POK),NOTIFY=DB2RES2,
//       CLASS=A,MSGCLASS=T,MSGLEVEL=(1,1),TIME=1440
//*-------------------------------------------------------------*//
//*  SAMPLE FORMAT :                                             *//
//*  REPAIR LOCATE TABLESPACE RESTDB02.RESTTP01 PAGE X'300000'   *//
//*    VERIFY OFFSET X'0C' DATA X'DBIDPSID'  OLD DESCRIPTOR      *//
//*    REPLACE OFFSET X'0C' DATA X'DBIDPSID' NEW DESCRIPTOR      *//
//*-------------------------------------------------------------*//
//UTIL EXEC DSNUPROC,SYSTEM=DBB1,UID=REPTS,
//        LIB='DSN410.SDSNLOAD'
//DSNUPROC.SYSIN    DD  *
REPAIR LOCATE TABLESPACE RESTDB02.RESTTP01  PAGE X'300000'
  VERIFY OFFSET X'0C' DATA X'01080002'
   REPLACE OFFSET X'0C' DATA X'01090004'
   DUMP  LENGTH 100
```

Figure 49. REPAIR Utility

Note that the PAGE option specification is X'300000', indicating the fourth (3+1) partition and page 0.

7.6 Moving Indexes

The procedure for Type 1 indexes is the same as described for table spaces. If you print a Type 1 index page set header page with the DSN1PRNT utility, you find the DBID and OBID / ISOBID at the same offset, X'0C,' as described for table spaces. As mentioned before, you can not use that procedure for Type 2 indexes.

If you do not want to use the REPAIR utility for indexes, you can still recover the indexes from the data on the target DB2.

Complete the merge procedure as described in 7.4, "Copying Data with DSN1COPY and OBID Translation" on page 124. We recommend that you take full image copies for the moved table spaces, run RUNSTATS on the target DB2, and bind all plans and packages that were bound on the source DB2.

7.7 DSN1COPY with Incremental Image Copies

If you have a large amount of data you have to move, and you do not want to use the REPAIR utility, you might have a problem finding a convenient time frame in which to copy the data with DSN1COPY and the OBIDXLAT option.

If there is not much update activity on the table spaces you want to move, that is, only a couple of pages are affected by the update activity, incremental image copies could be a solution. This procedure uses DSN1COPY with an image copy taken in the source DB2 as input instead of the source table space. The advantage of this procedure is that you can prepare offline a copy of the table space to be used at the target system, although the copy does not reflect the last updates. Meanwhile, the table space is still available on the source DB2 subsystem for updates.

To move data by using DSNICOPY with incremental image copies follow these steps:

1. Stop the table space on the target DB2 to prevent it from being updated on the target DB2.
2. Using the last image copy taken on the source DB2 as input and the table space on the target DB2 as output, execute DSN1COPY with the OBIDXLAT option.
3. Take incremental image copies of the table spaces on the source DB2.
4. When you are ready to move the data, stop the table space for update processing on the source DB2 and take a last incremental image copy (SHRLEVEL REFERENCE).
5. Use the MERGE utility if you have more than one incremental image copy.
6. Apply the merged incremental image copy to the table space created in step 2, using DSN1COPY with OBID translation.
7. Recover the indexes on the target DB2.
8. Start the table spaces on the target DB2.

Using this procedure there is still the disadvantage that you have to keep the data twice: There is one copy on the source DB2 and another copy on the target DB2. Therefore you have to provide the DASD space for both copies.

7.8 Summary

In this chapter we describe several procedures that should help you in merging DB2 subsystems. Various aspects such as the hardware environment, amount of data, structure of your data objects, and time frame influence the decision of which procedure to follow. Unfortunately no one procedure fits the environment of every customer. Thus the safest way is to copy all data, using the DSN1COPY utility with the OBIDXLAT option, and recover all indexes on the target DB2. If you cannot use that procedure, use the REPAIR utility on the table spaces and recover the indexes on the target DB2.

Chapter 8. Commands and Operations

In this chapter we describe:

- New aspects of operations in a Sysplex
- Commands to display and control coupling facilities
- Commands to display and monitor DB2 structures
- New messages related to DB2 data sharing

8.1 Operations from an MVS Console

In a Parallel Sysplex all MVS consoles are connected to all MVS images. On one MVS console you can route commands to a specific MVS image in the Sysplex, using the ROUTE or RO command:

```
ROUTE sysname,text
example: ROUTE sc53,=DBA2 START DB2
```

In this example, the command routed is sc53, an MVS image in the Sysplex, to start the DBA2. subsystem

8.1.1 Routing a DB2 Command

If you have an active data sharing group, operations on an individual member can be controlled from any MVS console by entering commands prefixed by the appropriate command prefix.

For example, you can start a DB2 performance trace on member DBA1 by entering this command on any console in the Parallel Sysplex:

```
=DBA1 START TRACE (PERFM)
```

This routing of commands requires that the command prefix scope be registered as *S* or *X* on the IEFSSNxx PARMLIB member. For example:

```
SUBSYS SUBNAME(DBA1)
  INITRTN(DSN3INI)
  INITPARM('DSN3EP,=DBA1,S,DSGA')
```

Scope S in INITPARM enables you to start this member on any MVS of the Sysplex.

See "Using Command Prefix" in Chapter 3 of *IBM DATABASE 2 for MVS/ESA Version 4 Data Sharing: Planning and Administration* for more information.

8.1.2 Command Scope

The breadth of a command′s impact is called the scope of that command. There are member-scope, group-scope, and member- or group-scope commands:

- The **Member-scope commands** are commands used in a data sharing environment that affect only the DB2 for which they are issued:
  - ALTER BUFFERPOOL
  - CANCEL THREAD
  - DISPLAY BUFFERPOOL
  - DISPLAY PROCEDURE
  - DISPLAY TRACE
  - DSNC DISPLAY
  - DSNC START
  - START DB2
  - STOP DB2
  - TERM UTILITY
  - ....

  IBM DATABASE 2 for MVS/ESA Version 4 Data Sharing: Planning and Administration contains a complete list of member-scope commands.

- **Group-scope commands** are commands that affect only the group for which they are issued:
  - ALTER GROUPBUFFERPOOL
  - BIND PACKAGE
  - BIND PLAN
  - DCLGEN
  - DISPLAY DATABASE
  - DISPLAY GROUP
  - DISPLAY GROUPBUFFERPOOL
  - DISPLAY UTILITY
  - FREE PACKAGE
  - FREE PLAN
  - REBIND PACKAGE
  - REBIND PLAN

- START DATABASE
- STOP DATABASE.

- **Member or group-scope commands** are commands whose options enable them to affect either a single member or an entire group:

  - ARCHIVE LOG SCOPE(MEMBER/GROUP)

    The use of MODE(QUIESCE) in the ARCHIVE LOG command implies a group scope. The update activities on the entire group stop so that a groupwide point of consistency can be taken.

    If you use the default options, you do an OFFLOAD on the member and switch your active log even though there are update activities. A point of consistency is not taken.

  - MODIFY IRLMPROC,STATUS,ALLD/ALLI

    This command enables you to see all DB2 subsystems in a data sharing group that are or have been connected to one IRLM. The STATUS can be UP, DOWN, or SYSFAIL. The RET_LKS column indicates the number of retained locks held by a subsystem that failed or was running on an IRLM that failed.

```
MODIFY DBA1IRLM,STATUS,ALLD
DXR102I IRA1 STATUS IRLMID=001 737
SUBSYSTEMS IDENTIFIED                PT01
NAME     STATUS    RET_LKS   IRLMID  IRLM_NAME
DBA2     UP              0    002      IRA2
DBA1     UP              0    001      IRA1
```

8.1.3 Receiving Messages

In a Sysplex, each system has its own SYSLOG and logrec data sets. You can receive messages from all members at a single MVS console.

If you are running MVS 5.1, you use the JES2 Multiple Access Spool (MAS) functions to operate from an SDSF console. The SDSF command, SYSID sysname, enables you to switch the log facility from one system to another.

If you are running MVS 5.2 and SDSF 1.5, use the OPERLOG facilities. The log data is in a log stream that resides on a coupling facility; as the coupling facility fills, older data is offloaded to archival log stream data sets on DASD.

The system logger has its own required data set, LOGR, which contains information that the coupling facility can use to manage system logger resources. A specific coupling facility structure is dedicated to the system logger.

The MVS system name that issues a message appears at the beginning of a message. In Figure 50 on page 136 , the commands DISPLAY THREAD were issued from the same console and routed to the correct subsystem because of the command prefix. The answers came identified by MVS sysname in the first column (SC48, SC53). Subsystem DBA1 with command prefix =DBA1 is running on MVS image SC48, and subsystem DBA2 with command prefix =DBA2 is running on MVS image SC53.

```
SC48 DB2RES7   =DBA1 DIS THREAD(*)
SC48 STC25215  DSNV401I =DBA1 DISPLAY THREAD REPORT FOLLOWS -
SC48 STC25215  DSNV402I =DBA1 ACTIVE THREADS - 871
          871  NAME     ST A   REQ ID           AUTHID   PLAN     ASID
          871  TSO      T        4 DB2RES7      DB2RES7  DSNESPCS 0054
          871  DISPLAY ACTIVE REPORT COMPLETE
SC48 STC25215  DSN9022I =DBA1 DSNVDT '-DIS THREAD' NORMAL COMPLETION
SC53 DB2RES7   =DBA2 DIS THREAD(*)
SC53 STC25288  DSNV401I =DBA2 DISPLAY THREAD REPORT FOLLOWS -
SC53 STC25288  DSNV402I =DBA2 ACTIVE THREADS - 373
          373  NAME     ST A   REQ ID           AUTHID   PLAN     ASID
          373  TSO      N        1              DB2RES3           004B
          373  DISPLAY ACTIVE REPORT COMPLETE
SC53 STC25288  DSN9022I =DBA2 DSNVDT '-DIS THREAD' NORMAL COMPLETION
```

Figure 50. Identification of Messages in a Sysplex

Notice that messages DSNV401I and DSNV402I contain the command prefix (=DBA1,=DBA2) of the subsystem issuing the message.

If one system is no longer connected to the OPERLOG structure, the messages coming from this system are no longer displayed. To display the state of the connection use this command:

```
D XCF,STR,STRNM=SYSTEM_OPERLOG
```

and issue the following MVS command to reuse the OPERLOG:

```
V OPERLOG,HARDCPY
```

8.2 Submitting Work to Be Processed

In this section we describe the use of the group attachment name and the ways to run DB2 applications in a DB2 data sharing environment.

8.2.1 Running CICS and IMS Applications

There is no change in the process of running CICS and IMS applications for data sharing. You have to attach CICS or IMS to a specific DB2 member using the subsystem name. It is not possible for CICS and IMS to use the group attachment name, because they must be aware of the particular DB2 subsystem to which they are attached so they can resolve in-doubt units of recovery in the event of a failure.

If you want to restart a failed DB2 subsystem onto another MVS image, you have to restart the transaction manager and IRLM there too. There must be a one-to-one relationship between a DB2 and its IRLM. The target DB2 cannot be restarted on another system by connecting to an IRLM that another DB2 on that system is using.

8.2.2 TSO, Batch, and CAF Applications and Utilities

TSO, batch, CAF applications, and utilities have two methods for specifying the DB2 to which they want to connect. For utilities the group attachment name is usable only if the PTF for PN78818 is operational on the DB2 subsystem.

- Group attachment name

  The group attachment name acts as a generic name for the DB2 subsystem in a data sharing group. It is replaced by a DB2 subsystem name running on the MVS where the job is started.

  The group attachment name is specified at DB2 installation on the DSNTIPK panel. It is placed in the IEFSSNxx member and in the DSNHDECP load module used by the group. In case of implicit connection with the CAF attachment, DB2 uses the group attachment name from the DSNHDECP module. The recommended method is to explicitly use the group attachment name as it allows you to keep a single version of the JCLs that can run on any system. As a group attachment name, use the name of the DB2 subsystem in your production environment so that you do not have to change the subsystem name in your JCLs.

- Subsystem name

  Using the subsystem name enables you to specify one member name and prevent another member from executing the job. This method could be useful in a nonsymmetrical Sysplex where you want specific long-running batch jobs to run on the more powerful CEC. An alternative would be to route those batch jobs with the JES2 classes (a specific class is started on only one MVS image of the Sysplex) and keep the group attachment name in the JCLs.

If you have two members running on the same MVS and you use the group attachment name, the job will be attached to the first subsystem described in the IEFSSNxx member of SYS1.PARMLIB. There is one exception. Should you have defined a DB2 subsystem with the same name as the group attachment name, that subsystem will be used. First the subsystem name is searched. If none is found, the group attachment name is searched. Here is an example:

```
IEFSSNxx definition:
        DSGA
        DBA1 ... ,DSGA
        DBA2 ... ,DSGA
        DBA3 ... ,DSGA
application call:
        DSN(DSGA)   --> you will go to subsystem DSGA.
```

It is probably not good practice to use the same name for a DB2 subsystem and a DB2 group attachment name at the same time. The following algorithm helps to qualify our recommendation:

Algorithm

```
Attempt to identify to a DB2 subsystem with the specified name.
 If successful, then RETURN
 If (RC00F3006 (ssid not known) OR
     RC00F3011 (ssid not active AND
                group attach is specified in ssid entry)
    )
    AND (CAF or TSO or Utility)
    AND (MVS 4.2 or higher) then
 For RC00F3006; see if the name is defined as a group attach name.
  If it is not a group attach name, then
   RETURN with (Identify failed)
  If it is a group attach name, then
   Attempt to identify to each subsystem that specified the group
   attach name until we are successful or exaust the list.
  If we exaust the list then RETURN with (identify failed)

Two points must be noted here;

1. If you have defined a ssid with the same name as a group but the
   ssid definition has not specified a group name
   Example: DBGA
            DBA1...DBGA
            DBA2...DBGA
            then you cannot connect to the data sharing group DBGA
            (The group attach list will not be searched)

2. If in the above example the ssid DBGA has another group attach name
   defined, then MVS will only try DB2 members associated with that
   particular group attach name.
```

8.3 XCF Commands

XCF provides a set of commands that get you a lot of information about the status of the coupling facilities, the coupling facility links, and the Parallel Sysplex.

In this section we describe:

- Activating and deactivating a CFRM policy
- XCF display command with various options
- Displaying information about structures

8.3.1 Activating and Deactivating a CFRM Policy

Before you start DB2 data sharing, you must define coupling facility structures such as the lock structure, SCA structure, and GBP. The size and the location of the structures are determined in the coupling facility resource management (CFRM) policy. The MVS utility, IXCMIAPU, enables you to define DB2 structures. See Appendix A for an example of a CFRM policy.

To **activate a CFRM policy**, you have to issue the following command:

```
SETXCF START,POLICY,TYPE=CFRM,POLNAME=cfrm_polname
```

If a CFRM policy is active, when you issue the SETXCF START,POLICY command, the MVS system starts the transition to using the new policy. If you change structure definitions in the new CFRM policy:

- The change takes effect immediately for unconnected structures.
- The change remains pending for connected structures. If the structure is rebuildable (SCA or lock structure), you might use the operator command SETXCF START,REBUILD to rebuild the structure with the new characteristics. If the structure is not rebuildable (GBP), you have to disconnect and connect from the structure to get the changes active.

The DISPLAY XCF,STR command provides specific information about a structure and any pending policy changes.

To **deactivate a CFRM policy**, issue:

```
SETXCF STOP,POLICY,TYPE=CFRM
```

8.3.2 XCF Display Command with Various Options

The XCF DISPLAY command helps you obtain information about the Sysplex and the coupling facility.

8.3.2.1 D XCF: The D XCF command shows the names of the MVS images that are connected to the Parallel Sysplex (see Figure 51 on page 140).

```
D XCF
IXC334I  19.15.04  DISPLAY XCF 938
   SYSPLEX WTSCPLX1:   SC42           SC43           SC47
                       SC48           SC49           SC50
                       SC52           SC53           SC54
```

Figure 51. Display of MVS Images in the Parallel Sysplex

8.3.2.2 D CF: The D CF command shows the hardware characteristics of the coupling facility Figure 52 on page 141 shows an example of the command output.

```
D CF
IXL150I  19.23.01  DISPLAY CF 944
COUPLING FACILITY 009672.IBM.02.000000040104
                  PARTITION: 1  CPCID: 01
                  CONTROL UNIT ID: FFFD
NAMED [1] CF02
COUPLING FACILITY SPACE UTILIZATION
 ALLOCATED SPACE                 DUMP SPACE UTILIZATION
  STRUCTURES: [2] 56320 K       STRUCTURE DUMP TABLES:        0 K
  DUMP SPACE:       2048 K                  TABLE COUNT:        0
 FREE SPACE: [3]  443904 K      FREE DUMP SPACE:           2048 K
TOTAL SPACE:      502272 K      TOTAL DUMP SPACE:           2048 K
                               MAX REQUESTED DUMP SPACE:       0 K
   VOLATILE:         YES        STORAGE INCREMENT SIZE:      256 K
    CFLEVEL: [4]  2

COUPLING FACILITY SPACE CONFIGURATION
                          IN USE         FREE         TOTAL
CONTROL SPACE:           58368 K      443904 K      502272 K
NON-CONTROL SPACE:           0 K           0 K           0 K

SENDER PATH        PHYSICAL     LOGICAL     STATUS
     [5] 71        ONLINE       ONLINE      VALID
        73         ONLINE       ONLINE      VALID

COUPLING FACILITY DEVICE     SUBCHANNEL     STATUS
                   FFF8         184E        OPERATIONAL/IN USE
                   FFF9         184F        OPERATIONAL/IN USE
                   FFFA         1850        OPERATIONAL/IN USE
                   FFFB         1851        OPERATIONAL/IN USE
```

Figure 52. Display of a Coupling Facility

The information displayed includes:

1 Name of the coupling facility (here CF2)

2 Utilization of memory by structures

3 Free space

4 Level of coupling facility microcode

5 Device, subchannel, and CHPID status

Use the DISPLAY CF command to show how the coupling facility and structures are defined.

8.3.2.3 D XCF,POLICY: Use the D XCF,POLICY command to determine which policies are active for CFRM, SFM, and ARM (see Figure 53 on page 142).

```
D XCF,POLICY
IXC364I  13.28.31  DISPLAY XCF 564
TYPE: ARM
     POLNAME:       ARMPOL01
     STARTED:       01/31/96 12:53:53
     LAST UPDATED:  01/31/96 12:52:09
TYPE: CFRM
     POLNAME:       CFRM09
     STARTED:       06/10/96 13:13:04
     LAST UPDATED:  06/10/96 13:10:39
TYPE: LOGR
     NOT SUPPORTED BY DISPLAY XCF,POLICY
TYPE: SFM
     POLNAME:       SFM01
     STARTED:       01/02/96 11:00:29
     LAST UPDATED:  03/20/95 12:45:32

SYSPLEX FAILURE MANAGEMENT IS ACTIVE
TYPE: WLM
     NOT SUPPORTED BY DISPLAY XCF,POLICY
```

Figure 53. Display of the Active Policies

This command displays information about the CFRM policies in use in the Sysplex. The information includes:

- Name of the active CFRM policy
- Status of the policy
- Date and time the policy was started
- Date and time the policy was last updated

8.3.2.4 D XCF,COUPLE,TYPE=CFRM: The D XCF,COUPLE,TYPE=CFRM command displays information about CFRM couple data sets in use. The information displayed (see Figure 54 on page 143) is:

- Physical attributes of the CFRM couple data set such as name, volume serial number, device address, and time the data set was formatted.
- Maximum number of systems that the primary CFRM couple data set can support

```
D XCF,COUPLE,TYPE=CFRM
IXC358I  17.28.12  DISPLAY XCF 195
CFRM COUPLE DATA SETS
PRIMARY    DSN: SYS1.XCF.CFRM00
           VOLSER: TOTDS0     DEVN: 0CEE
           FORMAT TOD         MAXSYSTEM
           01/02/96 10:34:58         16
ALTERNATE  DSN: SYS1.XCF.CFRM01
           VOLSER: TOTDS1     DEVN: 0FEE
           FORMAT TOD         MAXSYSTEM
           01/02/96 10:36:39         16
CFRM IN USE BY ALL SYSTEMS
```

Figure 54. Display of CFRM Couple Data Sets

8.3.3 Displaying Information about Structures

The D XCF,STR command shows you status information about the coupling facility structures. You can specify difference options to see more detailed information.

8.3.3.1 D XCF,STR: The D XCF,STR command shows status information about the coupling facility structures in the policy. If specified without further qualification, summary information will be displayed about all coupling facility structures that are in the policy. See Figure 55.

```
D XCF,STR
IXC359I  19.28.03  DISPLAY XCF 976
STRNAME           ALLOCATION TIME  STATUS
DSNDSGA_GBP0     06/07/96 15:39:40 ALLOCATED
DSNDSGA_GBP1     06/07/96 21:44:07 ALLOCATED
DSNDSGA_LOCK1    05/30/96 20:00:42 ALLOCATED
DSNDSGA_SCA      06/07/96 17:59:43 ALLOCATED
DSNDSGB_GBP0        --       --    NOT ALLOCATED
DSNDSGB_LOCK1       --       --    NOT ALLOCATED
DSNDSGB_SCA         --       --    NOT ALLOCATED
ISTGENERIC       05/20/96 16:14:25 ALLOCATED
IXC_DEFAULT_1    06/04/96 08:22:34 ALLOCATED
IXC_GRS_1           --       --    NOT ALLOCATED
```

Figure 55. Display a List of the Defined Structures

8.3.3.2 D XCF,CF,CFNAME=cfname: Use this command to display information about the coupling facility in the policy. See Figure 56 on page 144.

```
D XCF,CF,CFNAME=CF01
IXC362I  18.03.32  DISPLAY XCF 566
CFNAME: CF01
  COUPLING FACILITY     :   009672.IBM.02.000000040104
                            PARTITION: 1   CPCID: 00
  POLICY DUMP SPACE SIZE:   2048 K
  ACTUAL DUMP SPACE SIZE:   2048 K
  STORAGE INCREMENT SIZE:   256 K

  CONNECTED SYSTEMS:
  SC42     SC43     SC47     SC48     SC49     SC50     SC52
  SC53     SC55

  STRUCTURES:
  DSNDSGA_GBP0     DSNDSGA_GBP1     IEFAUTOS         IGWLOCK00
  IXC_DEFAULT_1    IXC_DEFAULT_2    SYSTEM_LOGREC
```

Figure 56. Display of Coupling Facility Information

8.3.3.3 D XCF,STR,STRNAME=strname: The system displays detailed information for one or more named coupling facility structures. You can specify ALL to request information for all coupling facility structures. Wild card (*) suffixes are allowed. See Figure 57 on page 145.

```
D XCF,STR,STRNAME=DSNDSGA_GBP1
IXC360I  19.52.05  DISPLAY XCF 138
STRNAME: DSNDSGA_GBP1
 STATUS: ALLOCATED
  POLICY SIZE    : 16000 K
  POLICY INITSIZE: 8000 K
  REBUILD PERCENT: N/A
  PREFERENCE LIST: CF01     CF02
  EXCLUSION LIST IS EMPTY

 ACTIVE STRUCTURE
 ----------------
  ALLOCATION TIME: 06/07/96 21:44:07
  CFNAME         : CF01  1
  COUPLING FACILITY: 009672.IBM.02.000000040104
                     PARTITION: 1   CPCID: 00
  ACTUAL SIZE    : 8192 K  2
  STORAGE INCREMENT SIZE: 256 K
  VERSION        : ACFC332F 9992A405
  DISPOSITION    : DELETE
  ACCESS TIME    : 0
  MAX CONNECTIONS: 32
  # CONNECTIONS  : 2  3

 CONNECTION NAME  ID VERSION  SYSNAME  JOBNAME  ASID STATE
  ---------------- -- -------- -------- -------- ---- ----------------
  DB2_DBA1         02 00020002 SC48     DBA1DBM1 004E ACTIVE  4
  DB2_DBA2         01 00010002 SC53     DBA2DBM1 00FD ACTIVE
```

Figure 57. Display of Structure Information

Notes:

1 CF1 is the coupling facility where the DSNDSGA_GBP1 structure is actually allocated.

2 The effective size of the structure is 8192 KB.

3 The number of members using the structure is two.

4 The ACTIVE state is the normal state. When one subsystem is stopped, the state becomes FAILED-PERSISTENT for the corresponding connection name for the lock structure.

8.4 Displaying Information about Structure Use

The DB2 commands DISPLAY GROUP and DISPLAY DATABASE with the LOCK option give you information about the use of the lock structure and on which page sets the locks are kept. DISPLAY GROUPBUFFERPOOL gives you information about the actual use of the GBP structures.

8.4.1 Lock Structure and SCA Structure

The DB2 DISPLAY GROUP command displays information about the data sharing group to which a DB2 subsystem belongs. Figure 58 shows the command output.

```
=DBA1 DIS GROUP
DSN7100I =DBA1 DSN7GCMD 432
*** BEGIN DISPLAY OF GROUP(DSNDSGA ) GROUPLEVEL(410)
-------------------------------------------------------------------------
DB2                                       DB2  SYSTEM  IRLM
MEMBER    ID   SUBSYS  CMDPREF   STATUS   LVL  NAME    SUBSYS  IRLMPROC
--------  ---  ----    --------  -------  ---  ------  ------  --------
DBA1        1  DBA1    =DBA1     ACTIVE   410  SC43    IRA1    DBA1IRLM
DBA2        2  DBA2    =DBA2     ACTIVE   410  SC53    IRA2    DBA2IRLM
-----------------------------------------------------------------------
SCA   STRUCTURE SIZE:     4096 KB, STATUS= AC,   SCA IN USE:     1 %
LOCK1 STRUCTURE SIZE:    16128 KB,             LOCK1 IN USE:   < 1 %
NUMBER  LOCK ENTRIES:     4194304
NUMBER  LIST ENTRIES:       54486, LIST ENTRIES  IN USE:          5
*** END DISPLAY OF GROUP(DSNDSGA )
```

Figure 58. DB2 DISPLAY GROUP Command Showing Subsystems Running

Figure 58 shows that DBA1 is running on MVS system SC43 with IRLM IRA1.

SCA shows the size in kilobytes of the SCA structure and the percentage currently in use.

LOCK1 shows the size in kilobytes of the lock structure.

The display also shows the following:

- The maximum number of lock entries possible for the lock table
- The maximum number of modify resource list entries and how many of them are currently in use

8.4.2 Displaying Global Lock Information

With the DISPLAY DATABASE LOCKS command, you can display the global locks at the page set level (table space or partition or index or index partition). To get information about page or row locks, you need a performance tool like DB2 PM or its equivalent.

To display all page set P-locks and L-locks (see Figure 59) issue this command:

```
-DISPLAY DATABASE(dbname) SPACE(tsname) LOCKS
```

scale="0.8" width=column.

```
DSNT361I =DBA1 *  DISPLAY DATABASE SUMMARY
           *    GLOBAL LOCKS
DSNT360I =DBA1 ***********************************
DSNT362I =DBA1     DATABASE = DSN8D41A  STATUS = RW
               DBD LENGTH = 12104
DSNT397I =DBA1
NAME     TYPE PART STATUS             CONNID   CORRID       LOCKINFO
-------- ---- ---- ------------------ -------- ------------ ---------
DSN8S41D TS        RW
DSN8S41E TS    01  RW
DSN8S41E TS    02  RW                                       H(IX,PP,I)
    -                  MEMBER NAME DBA2                    1
DSN8S41E TS    02  RW                                       H(IX,PP,I)
    -                  MEMBER NAME DBA1  (CO)    2
DSN8S41E TS    03  RW
DSN8S41E TS    04  RW
DSN8S41P TS        RW
DSN8S41R TS        RW
DSN8S41S TS        RW
DSN8S41X TS        RW                                       H(X,PP,I)
    -                  MEMBER NAME DBA1          3
DSN8S41X TS        RW                 TSO      DB2RES7      H(X,S,C)
    -                  MEMBER NAME DBA1                      4
XACT1    IX        RW
XACT2    IX        RW
XDEPT1   IX        RW                                       H(S,PP,I)
    -                  MEMBER NAME DBA2
XDEPT1   IX        RW                                       H(S,PP,I)
    -                  MEMBER NAME DBA1
```

Figure 59. Display of Page Set P-Locks and L-Locks

Notes:

1 DBA2 has a P-lock (IX) on partition 2 of DSN8S41E.

2 DBA1 has a P-lock (IX) on partition 2 of DSN8S41E.

When any P-lock has a lock state of NSU, SIX, or IX, the identified page set is GBP-dependent. Here partition 2 of DSN8S41E is GBP-dependent.

3 DBA1 has a P-lock (X) on table space DSN8S41X.

4 DBA1 has an L-lock (X) on DSN8S41X for user DB2RES7.

Information about P-Locks can be obtained from the statistics and accounting traces, along with information about transaction locking. IFCID 259 enables you to monitor page P-locks without having to turn a full DB2 lock trace. IFCID 251 contains information about P-lock negotiation.

8.4.3 Group Buffer Pool Structures

You can use many commands and tools to get information about GBP use. One such command is the DB2 DISPLAY GROUPBUFFERPOOL command. The command output shows you information about directory usage that you do not see in other reports. An easy way of collecting interval statistics for performance analysis is to create a batch job that issues the following command periodically:

```
-DISPLAY GROUPBUFFERPOOL(GBP1) GDETAIL(*)
```

```
DSNB750I *DB1G DISPLAY FOR GROUP BUFFER POOL GBP1 FOLLOWS
DSNB755I *DB1G DB2 GROUP BUFFER POOL STATUS
             CONNECTED                                = YES
             CURRENT DIRECTORY TO DATA RATIO          = 5
             PENDING DIRECTORY TO DATA RATIO          = 5
DSNB756I *DB1G   CLASS CASTOUT THRESHOLD                  = 10%
             GROUP BUFFER POOL CASTOUT THRESHOLD      = 50%
             GROUP BUFFER POOL CHECKPOINT INTERVAL    = 8 MINUTES
             RECOVERY STATUS                          = NORMAL
DSNB757I *DB1G MVS CFRM POLICY STATUS FOR DSNDB0G_GBP1    = NORMAL
             MAX SIZE INDICATED IN POLICY             = 1024 KB
             ALLOCATED                                = YES
DSNB758I *DB1G     ALLOCATED SIZE                         = 1024 KB
               VOLATILITY STATUS                      = VOLATILE
DSNB759I *DB1G     NUMBER OF DIRECTORY ENTRIES            = 942
               NUMBER OF DATA PAGES                   = 187
               NUMBER OF CONNECTIONS                  = 2
DSNB783I *DB1G CUMULATIVE GROUP DETAIL STATISTICS SINCE 09:37:49 JUL
15, 1996
DSNB784I *DB1G GROUP DETAIL STATISTICS
             READS
               DATA RETURNED                          = 1802
DSNB785I *DB1G     DATA NOT RETURNED
                 DIRECTORY ENTRY EXISTED              = 2624
                 DIRECTORY ENTRY CREATED              = 11507
                 DIRECTORY ENTRY NOT CREATED          = 2688, 0
DSNB786I *DB1G   WRITES
               CHANGED PAGES                          = 12761
               CLEAN PAGES                            = 0
               FAILED DUE TO LACK OF STORAGE          = 0
             CHANGED PAGES SNAPSHOT VALUE             = 27
DSNB787I *DB1G   RECLAIMS
               FOR DIRECTORY ENTRIES                  = 1582
               FOR DATA ENTRIES                       = 5532
             CASTOUTS                                 = 11994
DSNB788I *DB1G   CROSS INVALIDATIONS
               DUE TO DIRECTORY RECLAIMS              = 1357
               DUE TO WRITES                          = 3132
DSNB790I *DB1G DISPLAY FOR GROUP BUFFER POOL GBP1 IS COMPLETE
```

Figure 60. Output of DB2 DISPLAY GROUPBUFFERPOOL Command

Figure 60 shows the output of the DB2 DISPLAY GROUPBUFFERPOOL command. You can use the output to ensure that your GBPs are correctly sized and have the correct ratio of directory entries to data pages. See Chapter 10, “Performance and Tuning Considerations” on page 191, and Chapter 8 of *Data Sharing Performance Topics* for an explanation of the various counters displayed.

8.5 Displaying Group Buffer Pool Dependencies

To determine which table spaces or indexes are GBP-dependent, issue the following command:

```
-DISPLAY BUFFERPOOL(BP1) LIST(*)
```

Figure 61 shows the output of the DB2 DISPLAY BUFFERPOOL command.

```
DSNB401I =DBA1 BUFFERPOOL NAME BP1, BUFFERPOOL ID 1, USE COUNT 3
DSNB402I =DBA1 VIRTUAL BUFFERPOOL SIZE = 1000 BUFFERS
             ALLOCATED       =     1000   TO BE DELETED   =       0
             IN-USE/UPDATED  =        0
DSNB403I =DBA1 HIPERPOOL SIZE = 0 BUFFERS, CASTOUT = YES
             ALLOCATED       =        0   TO BE DELETED   =       0
             BACKED BY ES    =        0
DSNB404I =DBA1 THRESHOLDS -
             VP SEQUENTIAL         = 80   HP SEQUENTIAL         = 80
             DEFERRED WRITE        = 50   VERTICAL DEFERRED WRT = 10
             PARALLEL SEQUENTIAL   = 50
DSNB451I =DBA1 INDEXSPACE = DSN8D41A.XEMP2, USE COUNT = 0, GBP-DEP = Y
DSNB451I =DBA1 INDEXSPACE = DSN8D41A.XEMP1, USE COUNT = 0, GBP-DEP = Y
DSNB450I =DBA1 TABLESPACE = DSN8D41A.DSN8S41E, USE COUNT = 0, GBP-DEP
= Y
DSN9022I =DBA1 DSNB1CMD '-DISPLAY BUFFERPOOL' NORMAL COMPLETION
```

Figure 61. Output of DB2 DISPLAY BUFFERPOOL Command

The table spaces and indexes mentioned in messages DSNB450I and DSNB451I are currently open and associated with this buffer pool. USE COUNT indicates the number of applications accessing the page set. GBP-DEP=Y indicates that the page set is GBP-dependent. If the table space is partitioned, Y indicates that at least one of the partitions is GBP-dependent.

8.6 Changing Coupling Facility Structures

In this section we discuss some modifications you may have to make to coupling facility structures when DB2 is active.

8.6.1 Adding a Structure to a Coupling Facility

Create a new CFRM policy (or update an existing CFRM policy) to establish the new structure definition. Activate the new CFRM policy, using the SETXCF START,POLICY command. This operation is required if you are adding a new GBP. It can be done without stopping any activity, provided there is enough storage in the coupling facility for the new structure.

8.6.2 Moving a Structure from One Coupling Facility to Another

This procedure could be useful if you have to reconfigure the coupling facility. We look at two cases:

- Lock and SCA structures

  You can dynamically rebuild these structures:

  1. If necessary, create a new CFRM policy (or update an existing CFRM policy) to add additional coupling facilities to the preference list for the structure that is to be moved. The structure can be allocated in a different coupling facility only if that coupling facility is in the structure′s preference list. It may also be necessary or desirable to remove from the structure′ preference list the coupling facility in which the structure is currently allocated.
  2. Activate the new CFRM policy, using the SETXCF START,POLICY command. After the START, you might observe a POLICY CHANGE PENDING status. See Figure 62.

```
D XCF,STR
IXC359I  16.38.53  DISPLAY XCF 124
STRNAME            ALLOCATION TIME  STATUS
DSNDSGA_GBP0     06/11/96 13:55:49 ALLOCATED
DSNDSGA_GBP1     06/11/96 13:56:41 ALLOCATED
                                   POLICY CHANGE PENDING
DSNDSGA_LOCK1    06/11/96 10:51:37 ALLOCATED
                                   POLICY CHANGE PENDING
DSNDSGA_SCA      06/07/96 17:59:43 ALLOCATED
                                   POLICY CHANGE PENDING
```

Figure 62. Display Coupling Facility Structures

  3. To dynamically rebuild the structure, issue this command (see Figure 63 on page 152):

     SETXCF START,REBUILD,STRNM=DSNDSGA_LOCK1,LOCATION=OTHER

     LOCATION=OTHER causes the structure to be rebuilt into a different coupling facility from the one in which it is currently allocated, in accordance with the current structure preference list. There must be at least one other usable coupling facility in the structure′s preference list in order for this command to be processed successfully.

```
SETXCF START,REBUILD,STRNM=DSNDSGA_LOCK1,LOCATION=OTHER
IXC367I THE SETXCF START REBUILD REQUEST FOR STRUCTURE 186
DSNDSGA_LOCK1 WAS ACCEPTED.
DXR145I IRA1 REBUILDING LOCK STRUCTURE AT OPERATORS REQUEST
DXR145I IRA2 REBUILDING LOCK STRUCTURE AT OPERATORS REQUEST
DXR146I IRA1 REBUILD OF LOCK STRUCTURE COMPLETED SUCCESSFULLY
```

Figure 63. Rebuild a Structure

- GBP structures

  You cannot change the coupling facility where a group buffer pool is allocated while there is an active connection to one or many DB2 members. You have to:

  1. Stop databases that use this GBP.
  2. Change the CFRM policy.
  3. A new GBP will be automatically allocated and connected at the point when the first page set goes into inter-DB2 read-write interest

  Stopping the activity with GBP0 (and SYSLGRNX) results in stopping all members of the group. See “Tuning Group Buffer Pool” in Chapter 6 of *IBM DATABASE 2 for MVS/ESA Version 4 Data Sharing: Planning and Administration* for more information.

8.6.3 Changing Characteristics of Structures

You can dynamically change the main characteristics of the GBP:

- Size

  SETXCF START,ALTER,STRNM=DSNDSGA_GBP1,SIZE=1536

  This change increases the GBP size to 1536 KB, assuming that this size is less than the MAXSIZE defined in the CFRM policy.

- Castout threshold values

  Use ALTER GROUPBUFFERPOOL to change the GBP castout thresholds:

  =DBA1 ALTER GROUPBUFFERPOOL(GBP1) CLASST(15) GPOOLT(55)

  These changes take effect immediately.

- Checkpoint frequency

  Use ALTER GROUPBUFFERPOOL to change the time between two GBP checkpoints:

  =DBA1 ALTER GROUPBUFFERPOOL(GBP1) GBPCHKPT(3)

With this command, the GBP checkpoints will occur every 3 minutes.

You can dynamically change the size of the lock structure if all of the following conditions are true:

- All members of the group are running with MVS 5.2 or above.
- The lock structure is allocated in a coupling facility with CFLEVEL greater than zero.
- The new SIZE is less than the maximum defined in the CFRM policy.

To change the size, enter the following command:

```
SETXCF START,ALTER,STRNM=DSNDSGA_LOCK1,SIZE=newsize
```

If you want to change the maximum size in the CFRM policy, you have to change the CFRM policy and then rebuild the lock structure.

8.7 Shutting Down the Coupling Facility

You have to shut down a coupling facility to apply maintenance or perform some other type of reconfiguration. For the least amount of disruption, move all of your structures to another coupling facility before shutting it down.

Follow these steps:

1. Modify the preference list in the CFRM policy to drive the next allocation of structures to the remaining coupling facility and adapt the size parameters.
2. Activate the new CFRM policy, using this command:

   ```
   SETXCF START,POLICY,TYPE=CFRM,POLNAME=polname
   ```

3. Move the SCA and lock structures to another coupling facility, using this command:

   ```
   SETXCF START,REBUILD,STRNAME=strname,LOC=OTHER
   ```

   Alternatively, you can use:

   ```
   SETXCF START,REBUILD,CFNAME=cfname,LOC=OTHER
   ```

   This command rebuilds all structures that enable rebuild onto the alternate coupling facility.
4. Deallocate any GBP allocated in the coupling facility by one of the methods described in “Tuning the Group Buffer Pool ” in Chapter 6 of *IBM DATABASE 2 for MVS/ESA Version 4 Data Sharing: Planning and Administration*.
5. Remove the target coupling facility.
6. When the GBPs are accessed again, they will be reallocated in the alternate coupling facility according to the preference list of the new CFRM policy.

7. Apply maintenance to the coupling facility.

8.8 Starting a Sysplex Failure Management Policy

SFM enables you to define a sysplexwide policy that specifies the actions that MVS is to take when certain failures occur in the Sysplex. A number of situations might occur during the operation of a Sysplex when one or more systems has to be removed so that the remaining Sysplex members can continue to do work. The goal of SFM is to enable these reconfiguration decisions to be made and carried out with little or no operator involvement.

DB2 takes advantage of the SFM policy with:

- The SCA and lock structure

  With an active SFM policy and the REBUILDPERCENT specification in the CFRM policy, DB2 will trigger an automatic REBUILD of the SCA and lock structures on the alternate coupling facility if there is a coupling facility or connectivity failure.

- GBP

  The automatic REBUILD of a GBP is not available in DB2 4.1, but DB2 GBP error handling can take advantage of the new information that MVS provides to DB2 event exits that indicates the scope of the loss of coupling facility connectivity. If MVS indicates that the scope of the loss of coupling facility connectivity is 100% (for example, the coupling facility containing the GBP has failed), and there is an active SFM policy, DB2 immediately triggers damage assessment for the affected GBPs and automatically cleans up all active and failed-persistent connections so that the GBPs can be reallocated into an alternate coupling facility. This function requires DB2 APAR PN76191 and MVS APARs OW16069, OW16071, and OW16072. If the MVS APARs are not applied, or there is no active SFM policy with CONNFAIL(YES), DB2 will handle 100% of the loss of coupling facility connectivity as described below, regardless of whether this APAR (PN76191) is applied or not.

  When a coupling facility or an attachment (for example, a coupling facility link or channel) to a coupling facility that contains one or more GBPs fails and no SFM policy is active, DB2 may hang as a result of trying to process the loss of coupling facility connectivity. The result is that the affected DB2 members do not disconnect from the GBP. Each DB2 member′s connection to the GBP goes to ″failed-persistent,″ and the GBP structure remains ″allocated″ to MVS. Before recovery can be initiated for the pages that go into the LPL, the user must first use the SETXCF FORCE command to manually delete the failed-persistent connections so that DB2 can initiate the damage assessment process to determine which DB2 objects may have had changed pages in the failed GBP.

The recommendation is to use the SFM policy to avoid the manual operations.

To start an SFM policy (POLICY1, for example) that is defined in the SFM couple data set, issue this command:

SETXCF START,POLICY,POLNAME=POLICY1,TYPE=SFM

Figure 64 shows the output of the command.

```
SETXCF START,POLICY,POLNM=SFM01,TYPE=SFM
IXC602I SFM POLICY SFM01 INDICATES FOR SYSTEM SC48 A STATUS 859
UPDATE MISSING ACTION OF ISOLATE AND AN INTERVAL OF 30 SECONDS.
THE ACTION IS THE POLICY DEFAULT.
IXC609I SFM POLICY SFM01 INDICATES FOR SYSTEM SC48 A SYSTEM WEIGHT OF 2
SPECIFIED BY POLICY DEFAULT
IXC601I SFM POLICY SFM01 HAS BEEN STARTED BY SYSTEM SC48
IXC602I SFM POLICY SFM01 INDICATES FOR SYSTEM SC53 A STATUS 839
UPDATE MISSING ACTION OF ISOLATE AND AN INTERVAL OF 30 SECONDS.
THE ACTION IS THE POLICY DEFAULT.
IXC609I SFM POLICY SFM01 INDICATES FOR SYSTEM SC53 A SYSTEM WEIGHT OF 5
SPECIFIED BY SPECIFIC POLICY ENTRY
IXC601I SFM POLICY SFM01 HAS BEEN MADE CURRENT ON SYSTEM SC53
```

Figure 64. Message IXC602I. The Output of SETXCF START, POLICY, POLNAME=POLICY1, TYPE=SFM.

Message IXC609I indicates the weight of the system for the SFM policy.

To stop an SFM policy, issue this command:

SETXCF STOP,POLICY,TYPE=SFM

8.8.1 Using Automatic Restart Management

ARM is an MVS recovery function that can improve the availability of specific batch jobs or started tasks. When a job or task fails, or the system on which the job is running fails, ARM can restart the job or task without operator intervention.

The goals of SFM and ARM are complementary. Whereas SFM keeps the Sysplex running, ARM keeps specific work in the Sysplex running. If a job or task fails, ARM restarts it on the same system on which it was running at the time of the failure. If a system fails, ARM restarts the work on other systems in the Sysplex; this is called a *cross-system restart*.

IBM provides policy defaults for ARM.

To implement the ARM policy, you have to:

- Format an ARM couple data set.
- Start the ARM policy in the ARM couple data set. Once the policy is started, all systems connected to the ARM couple data set will use the same active policy.

  To start ARM with the policy defaults, issue this command:

  SETXCF START,POLICY,TYPE=ARM

  To start your own ARM policy (mypol, for example), issue this command:

  SETXCF START,POLICY,TYPE=ARM,POLNAME=mypol

  Use the SETXCF STOP command to disable automatic restarts:

  SETXCF STOP,POLICY,TYPE=ARM

  For more information, see *MVS/ESA Setting Up a Sysplex*.

ARM is useful for restarting a failing DB2 member in case of a hardware failure or abend. Effectively, the failed subsystem is likely to own locks that become retained locks and can cause partial unavailability of the applications.

If you use the MVS 5.2 ARM or have other automation to ensure that failed DB2 members are restarted quickly in case of a failure, we recommend that you use RETLWAIT=YES. If you do not have automation in place to ensure that failed DB2 members are restarted quickly, we recommend that you use RETLWAIT=NO (the default).

8.9 Messages

In data sharing environment, new situations and new messages can occur. In this section we explain some of the new messages that could help you update your operations manual or change the automation procedures.

8.9.1 Group Buffer Pool Exceptions

The following messages concern the GBP:

- DSNB314I Damage assessment triggered for group buffer pool gbpname REASON=reason

  The main reason for this condition is a coupling facility failure or total loss of connectivity with this GBP.

  The damage assessment process uses some restrictive conditions conditions—group buffer pool recovery pending (GRECP) or LPL—to protect data integrity. See DSNB32xI and DSNB250E.

- DSNB250E A page range was added to the logical page list db=xx sp=yy

  When a ″must complete″ operation to the GBP fails (write at commit or rollback, read for rollback or restart), the page is put in the LPL and becomes unavailable.

To recover pages from the LPL, use the START DATABASE command.

- DSNB32xI (x for 0,1,2,3)

  A page set (table space or index or partition) has GRECP status. This occurs during the damage assessment process (see DSNB314I). This page set is no longer available for applications.

  To clear the GRECP status, use the START DATABASE command.

- DSNB303I

  A loss of connectivity was detected to group buffer pool gbpname

  DB2 disconnects from the coupling facility structure.

  The transactions in this member get -904 resource not available type 701 (GBP) reason C20204

- DSNB228I group buffer pool gbpname cannot be accessed IXLCACHE reason xxxxxxxx

  The reason code is explained in *The Sysplex Services Reference*.

  Example: xxxx0C17 could be a storage shortage in the GBP because the castout is too slow compared to the pace of the GBP writes.

8.9.2 Connectivity Exceptions

IXC518I indicates that system xxxx is not using the coupling facility for yy reason and connectivity is lost.

This detection triggers an automatic rebuild of the SCA or lock structure if it resides on the coupling facility that is no longer connected, there is an active SFM policy, and the REBUILDPERCENT has been reached.

8.9.3 Subsystem Failure

DSNT378I indicates that a user has timed out due to a retained lock:

- RC00C900BE if it is a retained L-lock
- RC00C20264 if it is a retained P-lock

8.9.4 CPC Failure

DXR137I indicates that irlmnm group status has changed, and irlmx has been disconnected from the data sharing group.

ARM, if installed, restarts the failed subsystem on another CPC.

8.9.5 Timer Failure

If you lose one of the two connections between a CPC and its Sysplex timer, the following message appears:

IEA262I ETR PORT xx IS NOT OPERATIONAL

Processing continues normally. Once connectivity has been restored, you receive the following message:

IEA267I ETR PORT xx IS NOW AVAILABLE

Alternatively, if all connections are lost to the Sysplex timer, the CPC will enter a nonrestartable wait state. This may include all of the CPCs in the Sysplex.

In this case you must first determine the cause of the connectivity failure and correct it.

Once connectivity to the Sysplex Timer has been restored, you can restart any failed subsystems, using the normal restart procedures.

Note that a CPC can be started in "local" mode where it does not use the External Time Reference offered by the Sysplex Timer. However, a failed DB2 data sharing member cannot be restarted on this system.

8.9.6 Changes of Policy

The messages associated with stopping or starting policies could explain some reactions of the Sysplex.

- IXC511I Start administrative policy polname for CFRM accepted.
- IXC607I SFM policy has been stopped by system xxxx.
- IXC601I SFM policy polname has been made current.
- IXC806I ARM policy has been stopped by system xxxx.
- IXC805I ARM policy has been started by system xxxx policy named polname is now in effect.

Chapter 9. Recovery in a Data Sharing Environment

From an application recovery perspective, not much has changed from the previous releases of DB2 when it comes to recovering your data. However, behind the scenes recovery in a data sharing environment is now far more complex.

In this chapter we describe the changes that have been made to DB2 to support application recovery in a data sharing environment. From a systems perspective we discuss logging considerations and detail different recovery scenarios in a Sysplex environment, stepping you through the messages to expect and the actions to take. We also document disaster recovery considerations in a data sharing environment and list the steps you have need to take when carrying out a disaster recovery.

9.1 Data Sharing Recovery Environment

In a data sharing environment, each DB2 member maintains its active and archive logs and records them in its own BSDS. But, as a recovery or restart process could need access to all logs for all members in the data sharing group, it is essential that all BSDSs and all log data sets be accessible to all members. In this scenario, DB2 uses a groupwide identifier, the LRSN, to merge the log records in the right order and to apply changes to data.

Let us have a look at the global recovery environment:

- Each DB2 member maintains:
  - Its own active and archive log data sets
  - Its own BSDS
- Each DB2 member needs:
  - Access to logs of all other members
  - Access to the BSDSs of all other members
- All logs may be needed for recovery

  Logs are merged by LRSN for recovery.
- The BSDS of each member contains information about the BSDSs of all the other members.

9.1.1 Logging Considerations

In a data sharing environment, DB2′s recovery process might need log records from other DB2 members logs that have updated data. Recovery process performance depends on whether logs have been archived or not, and where they have been archived (DASD or tape).

9.1.1.1 Avoid Archive Log to Be Used by Recovery: As in the past, the recovery cycle for a table space is delimited by the frequency of image copies. A RECOVER job will use the last full image copy, the incremental image copies (if available), and the log records written since the last copy. A RECOVER job will not use archived log if the log records it needs are still in the active log data sets.

Using archived log in the recovery process will increase recovery time and might have an impact on operational activities because of the need to have many unit tapes allocated.

Consider the following to minimize the need to use archived log:

- Increase the total active log space

  The total amount of active log space is the number of active log data sets multiplied by its size. Currently, DB2 limits the maximum number of active log data sets to 31. Because each DB2 member can have up to 31 active log data sets, the total number of active log data sets is effectively increased by the number of DB2 members in a data sharing group.

  The size of an active log data set is up to 2 GB but is often sized not to exceed a tape cartridge. Most installations prefer not to have an archived data set on more than one volume. With the ever-increasing capacity of the new tape devices, the size of the active log can also increase. However, some of the increased capacity is due to a tape compression algorithm. We do not recommend using tape compression for the DB2 archive log, because DB2 has to read the log backwards for backout operations. Performance for backout can be severely degraded if there is compression.

  This tape compression is not to be confused with DB2 data compression, which compresses the data portion of a DB2 log record. With DB2 data compression, the log record header is not compressed and causes no extra performance degradation for backward scans.

- Increase the frequency of image copies

  Because only the log records generated since the last image copy are needed for recovery, the more often you make copies, the less chance there is that archive log records will be needed. Of course, this consideration must be weighed against the time it takes to make the image copies and their effects on SQL transactions. Consider using SHRLEVEL CHANGE if image copy time is a concern.

- Consider using COPY (incremental) and MERGECOPY. The performance of the COPY utility is improved in Version 4.

9.1.1.2 Archiving Logs in a Data Sharing Environment: As we stated before, the RECOVER job depends on how the log has been archived. The following considerations are also valid in a non-data-sharing environment; in a data sharing environment they will be emphasized by the number of DB2 members involved in a recovery.

Basically, we have two ways of archiving logs:

- Archive to TAPE
- Archive to DASD
  - without using DFSMShsm
  - using DFSMShsm to migrate to tape

Archive to Tape: The RECOVER job needs at least one tape unit for each DB2 member whose archived log records are to be merged. (More might be needed if you run parallel recovery jobs.) Therefore, do not archive logs from more than one system to the same tape.

Archiving to tape is not recommended because there can be negative consequences to not having enough tape units allocated: If there are not enough tape units to do the recovery, DB2 could deadlock. If DB2 deadlocks happens, use the SET ARCHIVE command to increase the number of tape units that can be used.

If you must archive to tape, make sure that the value for READ TAPE UNITS on installation panel DSNTIPA for each member is high enough to handle anticipated recovery work. For example, if you have eight members, each member should specify at least eight tape drives. You will need more drives if you run more than one recovery job at the same time on a given member, or if multiple members run recovery jobs at the same time. Adjust the default set in the installation panel, using the SET ARCHIVE command (the changes are temporary; at restart, DB2 again uses the values set during installation).

Remember that in a data sharing environment the premount (look-ahead) function available since DB2 V3 is not used because the next log tape could be in use by another DB2 member. Only one recovery job can read an archive tape at a time.

Archive to DASD without Using DFSMShsm: There is no major impact on recovery performance but you have to carefully handle the archive logs on DASD to avoid running out of space. Remember that DASDs have to be shared by all the members in the data sharing group.

Archive to DASD Using DFSMShsm: This is probably the best solution whether in data sharing group environment or not. DFSMShsm can do automatic space and data availability management among storage devices in a system. But, a particular RECOVER job needs only one tape unit to recall migrated archive data sets. If the archive data sets have been migrated, they are done one at a time from the member running the RECOVER job. For example, a RECOVER job started on member DB1G might need log data sets of members DB1G and DB2G. DB1G sends the recall requests to DFSMShsm one at a time for the tapes needed for recovery.

You can run the REPORT RECOVERY utility to find out which log data sets recovery will need and then issue recall for those data sets before the recovery job needs them.

Consider archiving to DASD the LOGCOPY1 and manage it with DFSMShsm and direct LOGCOPY2 to tape, as LOGCOPY2 is rarely used.

9.1.2 Log Record Sequence Number

In a data sharing environment, RBAs can no longer be used to sequence the log records from multiple members in the data sharing group. Each DB2 system has its own log, and the RBA range that one DB2 system is using has no relationship to another DB2′s log RBAs.

The LRSN is used instead. It is derived from the STORE CLOCK instruction, based on time of day. It is a 6-byte value that is equal to or greater than the time stamp value truncated to 6 bytes. This value also appears in the page header, whereas in a non-data-sharing environment, we have an RBA value.

A delta may have to be added to the STCK value to cover the remote possibility that the member′s RBA at the time of joining the data sharing group is higher than the current time stamp from which the LRSN is derived. If that happens and DB2 is not using the delta, you have page log RBA values greater than the current LRSN value.

The delta is kept in the BSDS and SCA. All members will keep the same delta because LRSN must be in synch for the entire group.

The LRSN introduces the need for time synchronization across all DB2s in the data sharing group. This function is handled by the Sysplex Timer.

Even if all of the logs written on behalf of a UR are in the member′s DB2 log where the UR runs, you might have URs from multiple members in the data sharing group updating the same data. The LRSN is used to merge updates from multiple log data sets in the time sequence in which they occurred.

The LRSN will be unique and increasing for an object (for example, a page) but may be identical for different objects.

When reading a log from DASD by RBA, DB2 knows exactly where in the log data set the record of interest resides, because the record′s RBA is really an offset from the beginning of the data set to the record. When using an LRSN, however, it is not possible to predict which record corresponds to a particular LRSN because more log records may be written to a log data set during a given time interval in one part of the log than in another part of the same log data set. Therefore, when reading a DASD log data set by LRSN, it is necessary to do a binary search of the log blocks (control intervals) to locate the record containing a particular LRSN.

However, DB2 always tries to use the RBA if possible. The SYSLGRNX record has corresponding RBAs and LRSNs in pairs (STARTRBA, STARTLRSN and ENDRBA, ENDLRSN). For recovery, unless the image copy′s LRSN falls in the middle of a log range (that is, when you run image copy SHRLEVEL CHANGE), the Log Manager will use STARTRBA to READ the log directly. For all subsequent log ranges, the Log Manager will always use STARTRBA instead of using STRTLRSN. In the worst case, reading by LRSN is only done once per recover per member. After the first record is read, the log is read using RBAs.

During recovery, DB2 compares the LRSN in the log record with the LRSN in the page header before applying changes to DASD. If the LRSN in the log record is larger than the LRSN on the data page, the change is applied.

One of the implications of using LRSN is that a point-in-time recovery must use an LRSN rather than an RBA, that is, it must use the TOLOGPOINT keyword.

Figure 65 shows the REPORT RECOVERY utility output.

```
DSNU582I =DBA2 DSNUPPCP - REPORT RECOVERY TABLESPACE DB1.TS2 SYSCOPY ROWS
TIMESTAMP = 1996-07-01-16.46.27.098076,  IC TYPE =  F ,  SHR LVL = R,  DSNUM    = 0000,  START LRSN =AD1A1D6946D9
DEV TYPE  = 3390    ,                    IC BACK =   ,   STYPE   =  ,  FILE SEQ = 0000,  PIT LRSN  = 000000000000
DSNAME    = DSN410.IMAGCOPY.TS2.COPY5                        ,   MEMBER NAME = DBA1

TIMESTAMP = 1996-07-01-16.50.31.651021,  IC TYPE =  F ,  SHR LVL = C,  DSNUM    = 0000,  START LRSN =AD1A1E51ED7F
DEV TYPE  = 3390    ,                    IC BACK =   ,   STYPE   =  ,  FILE SEQ = 0000,  PIT LRSN  = 000000000000
DSNAME    = DSN410.IMAGCOPY.TS2.COPY6                        ,   MEMBER NAME = DBA2

DSNU592I =DBA2 DSNUPREC - REPORT RECOVERY INFORMATION FOR DATA SHARING MEMBER : DBA1
DSNU583I =DBA2 DSNUPPLR - SYSLGRNX ROWS FROM REPORT RECOVERY FOR TABLESPACE DB1.TS2
 UCDATE     UCTIME      START RBA      STOP  RBA     START LRSN     STOP  LRSN    PARTITION   MEMBER ID
 070196    15260658   0000017B4A5F   0000017B72AB   AD19D5CEF0A5   AD19D5EDD131     0000         0001
 070196    15281165   0000017B8EA1   0000017BE7F4   AD19D646724F   AD19DA4C8CB9     0000         0001
 070196    20464562   0000017C27A9   000000000000   AD1A1D7AF0FB   000000000000     0000         0001

DSNU592I =DBA2 DSNUPREC - REPORT RECOVERY INFORMATION FOR DATA SHARING MEMBER : DBA2
DSNU583I =DBA2 DSNUPPLR - SYSLGRNX ROWS FROM REPORT RECOVERY FOR TABLESPACE DB1.TS2
 UCDATE     UCTIME      START RBA      STOP  RBA     START LRSN     STOP  LRSN    PARTITION   MEMBER ID
 070196    20481090   000000268F00   000000000000   AD1A1DCBBA37   000000000000     0000         0002
```

Figure 65. REPORT Utility Output: Log Record Sequence Number

In this example, if we run RECOVER utility to currency, DB2 will restore the last image copy data set (DSN410.IMAGECOP.TS2.COPY6). Then, as the image copy′s LRSN (AD1A1E51ED7F) is in the middle of the SYSLGRNX ranges that have to be applied, DB2 will do a binary search on both members′ logs to position on the correct RBA record.

9.1.3 Changes to Support the Data Sharing Environment

To support the data sharing environment some changes have been made to the catalog and directory table spaces, BSDSs and, utilities.

9.1.3.1 Catalog and Directory:

SYSCOPY, SYSLGRNX, and SYSUTILX have been modified to reference the member name, member ID, and LRSN.

- **SYSCOPY Changes**
  - **GROUP_MEMBER (new column)** The DB2 data sharing member name of the DB2 subsystem that performed the operation. This column is blank if the DB2 subsystem was not in a DB2 data sharing environment at the time the operation was performed.
  - **PIT_RBA (changed)** When ICTYPE=P, this field contains the LRSN for the point in the DB2 log. (The LRSN is the RBA in a non-data-sharing environment.) For other ICTYPEs, this field is X′000000000000′.

    When ICTYPE=P, this field indicates the stop location of a point-in-time recovery. If a record contains ICTYPE=P and PIT_RBA=X′000000000000′, the copy pending state is active, and a full image copy is required. If such a record is encountered during fallback processing of RECOVER, the recover job fails, and a point-in-time recovery is required. PIT_RBA can be zero if the point-in-time recovery is completed by the fall-back processing of RECOVER, or if ICTYPE=P from a previous release of DB2.
  - **START_RBA (changed)** A 48-bit positive integer containing the LRSN of a point in the DB2 recovery log. (The LRSN is the RBA in a non-data-sharing environment.) The indicated point is:
    - For ICTYPE I or F, the starting point for all updates since the image copy was taken
    - For ICTYPE P, the point after the log-apply phase of point-in-time recovery
    - For ICTYPE Q, the point after all data sets have been successfully quiesced
    - For ICTYPE R or S, the end of the log before the start of the LOAD utility and before any data is changed
    - For ICTYPE T, the end of the log when the utility is terminated
    - For other values of ICTYPE, the end of the log before the start of the RELOAD phase of the LOAD or REORG utility
- **SYSLGRNX Changes**
  - **LGRSLRSN** Starting LRSN of update log records for data sharing. X′000000000000′ otherwise. For non-data-sharing, this column contains the same value as LGRERBA.
  - **LGRRELRSN** Ending LRSN of update log records for data sharing. X′000000000000′ otherwise. For non-data-sharing, this column contains the RBA value.
  - **LGRMEMBR(ID)** Data sharing member ID of the modifying DB2 subsystem. X′0000′ for non-data-sharing.

- **SYSUTILX** Changes
  - **USUMEMBR (new)** Executing DB2 subsystem′s member name for data sharing. Otherwise blank.

9.1.3.2 BSDS: The BSDS has been modified to contain the additional data sharing information:

- **The LRSN ranges for each active and archive log**. Each active and archive log entry has both an RBA and an LRSN range.
- **Member record**
  - The DB2 group name
  - The member name for this DB2
- **The member names for other DB2s**
- **The member ID for all the members** The member ID is assigned by DB2 to a member the first time it joins the data sharing group. It cannot be changed. This value is also found in SYSLGRNX records.
- **RBA when converted to V4**. DB2 uses this RBA to manage certain utility operations.
- **MAX RBA for TORBA**. This is the RBA of the originating member at the time data sharing was enabled. You cannot use the TORBA keyword on the RECOVER utility for any point on the log above this value.
- **MIN RBA for TORBA**. In case you disable data sharing, this is the log RBA of the surviving member. Recovery to a previous point-in-time cannot go before this point.
- **STCK to LRSN delta**. This number is only nonzero if DB2 could not use the exact store clock value to calculate the LRSN.

9.1.3.3 Utilities:: The COPY, LOAD, REORG, QUIESCE, RECOVER, and MERGECOPY utilities now place LRSN values in the SYSCOPY record and DB2 member IDs as well as the beginning and ending LRSN value in the SYSLGRNX record in addition to the RBA values.

The RECOVER utility has a new keyword, TOLOGPOINT.

The REPORT utility includes the LRSN information in its report. It also extracts from the BSDSs of other members of the data sharing group the volume serial numbers of their archived log data sets. This information is required for recovery operation of the specified table space.

The QUIESCE utility returns the LRSN value of the quiesce point. If WRITE(YES) is specified, all changed pages of listed page sets are castout to DASD.

In a data sharing group, the QUIESCE utility quiesces an object for the entire group and records an LRSN that can be used by RECOVER TOLOGPOINT.

Group Attachment Function: This function is available with some DB2 PTFs installed, and it can be used to execute online utilities in the data sharing environment. With it you can specify the group name in the SYSTEM PARM when running a utility. The utility will identify the DB2 subsystem that resides in the MVS where the job is running. If you have more than one DB2 member in the same data sharing group in the same MVS, the function will be attached to the first member in the IEFSSNxx. However, if you have a subsystem with the same name as the group attachment name, this will be used.

For example, if on the same MVS, you have two subsystems named DBA1 with a group name of DSGA and another subsystem named DSGA, if you use the group attach, the DSGA subsystem will be attached. In case it is not active the utility job will fail. But, if you also specify a group name of DSGA for the DSGA subsystem, and it is not active, DB2 will attach DBA1.

In summary, if a subsystem ID match is found, no attempt is made to search the list of group attachment names, unless a group name is specified for this subsystem.

We do not recommend having a DB2 subsystem with the same name as a data sharing group name, not because of the way it works, but because you need to have a clear naming convention.

Consider using some MVS mechanism for controlling where jobs run. For example, in JES2 MAS systems, use the following JCL statement:

```
/*JOBPARM SYSAFF=cccc
```

where cccc is the JES2 name. You can specify an asterisk (SYSAFF=*) to indicate that the job should run on the system from which it was submitted.

Following are some considerations about submitting online utilities:

- A utility job can be executed on any DB2 member in a data sharing group There are no restrictions on where utilities can be executed. For example, a recover utility job can run on a different MVS from the MVS where the image copy was run.
- **Only** utilities in STOPPED status can be restarted on any other DB2 in the data sharing group, regardless of whether you use the group attachment function.
- If DB2 fails while a utility is running, you cannot restart it on another DB2. The status when displayed through other members in the group is ACTIVE. When the failing DB2 comes up, it changes this status from ACTIVE to STOPPED. Once this happens, the utility can be restarted on any member of the data sharing group, regardless of whether you are using the group attachment function.

  Just as with previous releases of DB2, the same utility ID (UID) must be used to restart the utility.

- Utilities can be stopped only in the original DB2 subsystem. In each case a utility can be stopped (with the TERM utility command) only on the DB2 where it is running, regardless of group attachment.

  Depending on the maintenance level of your DB2, if the utility is in STOPPED status you can use the TERM utility command from any member of the data sharing group

Note

You may want to read the informational APAR II09377 related to the group attachment function.

9.1.3.4 Stand-Alone Utilities: Stand-alone utilities have no direct connection to DB2 services. As in previous DB2 releases, stand-alone utilities run as MVS jobs without attaching DB2. Therefore, a DB2 system has no indication that one of these utilities is running.

When running stand-alone utilities, consider that most recently updated pages might be not written to DASD yet, and that data could be in RECP or group buffer pool recovery pending GRECP status or have pages in the LPL. Use DISPLAY DATABASE with the RESTRICT option to find out whether there are exception statuses.

9.1.4 Running the RECOVER Utility

The RECOVER utility can be used to recover to the current time or a point-in-time and to apply only the logs to a previously restored version of data. All recovery scenarios operate in a data sharing environment as in non-data-sharing, a part of LRSN in a data sharing environment will be used to merge the logs and to verify if the change has to be applied.

```
DSNU582I =DBA2 DSNUPPCP - REPORT RECOVERY TABLESPACE DB1.TS2 SYSCOPY ROWS
TIMESTAMP = 1996-07-01-16.46.27.098076,  IC TYPE =  F ,  SHR LVL = R,  DSNUM    = 0000,  START LRSN =AD1A1D6946D9
DEV TYPE  = 3390    ,                    IC BACK =    ,   STYPE   =  ,  FILE SEQ = 0000,  PIT LRSN  = 000000000000
DSNAME    = DSN410.IMAGCOPY.TS2.COPY5                         ,   MEMBER NAME = DBA1

TIMESTAMP = 1996-07-01-16.50.31.651021,  IC TYPE =  F ,  SHR LVL = C,  DSNUM    = 0000,  START LRSN =AD1A1E51ED7F
DEV TYPE  = 3390    ,                    IC BACK =    ,   STYPE   =  ,  FILE SEQ = 0000,  PIT LRSN  = 000000000000
DSNAME    = DSN410.IMAGCOPY.TS2.COPY6                         ,   MEMBER NAME = DBA2

DSNU592I =DBA2 DSNUPREC - REPORT RECOVERY INFORMATION FOR DATA SHARING MEMBER : DBA1
DSNU583I =DBA2 DSNUPPLR - SYSLGRNX ROWS FROM REPORT RECOVERY FOR TABLESPACE DB1.TS2
 UCDATE     UCTIME      START RBA      STOP  RBA     START LRSN     STOP  LRSN    PARTITION   MEMBER ID
 070196    15260658   0000017B4A5F   0000017B72AB   AD19D5CEF0A5   AD19D5EDD131     0000         0001
 070196    15281165   0000017B8EA1   0000017BE7F4   AD19D646724F   AD19DA4C8CB9     0000         0001
 070196    20464562   0000017C27A9   000000000000   AD1A1D7AF0FB   000000000000     0000         0001

DSNU592I =DBA2 DSNUPREC - REPORT RECOVERY INFORMATION FOR DATA SHARING MEMBER : DBA2
DSNU583I =DBA2 DSNUPPLR - SYSLGRNX ROWS FROM REPORT RECOVERY FOR TABLESPACE DB1.TS2
 UCDATE     UCTIME      START RBA      STOP  RBA     START LRSN     STOP  LRSN    PARTITION   MEMBER ID
 070196    20481090   000000268F00   000000000000   AD1A1DCBBA37   000000000000     0000         0002
```

Figure 66. REPORT Utility Output: Recover to Current Time

Figure 66 shows the REPORT utility output. If you run the RECOVER utility to currency, DB2 will restore the last image copy data set (DSN410.IMAGECOP.TS2.copy6). Then, as the image copy′s LRSN (AD1A1E51ED7F) is in the middle of the SYSLGRNXL ranges that have to be applied, DB2 will do a binary search on both members′ log to position on the right RBA record.

The RECOVER utility has a new keyword, TOLOGPOINT, which any DB2 subsystem (sharing or not) can use to recover to a point-in-time after the start of Version 4. If you use the TOLOGPOINT keyword in a non-data-sharing environment, the value you specify is an RBA. In a data sharing environment, the keyword must be used to recover to a point-in-time after the originating member has joined the data sharing group. If you want to recover to a point-in-time before the originating member has joined the data sharing group, you can use either the TORBA or TOLOGPOINT keyword (with an RBA value). The RECOVER job can be run on any system of the data sharing group.

The maximum RBA value you can use for a point-in-time recovery is available from the output of REPORT and PRINT LOG MAP utilities. Figure 67 on page 169 shows the REPORT utility output.

If you want to recover to a point-in-time, say, RBA AD26902AB000 (an RBA within the MIN and MAX RBA range), you can use either the TORBA or TOLOGPOINT keyword with a value of AD26902AB000.

Even in a data sharing environment, as in the past, you can use the TOCOPY keyword to recover data to the values contained in an image copy without subsequent application of log changes.

```
DSNU593I =DBA1 DSNUPREC -  REPORT RECOVERY ENVIRONMENT RECORD:
'             MINIMUM   RBA: AD2690291000
'             MAXIMUM   RBA: AD26902AB9EA
'             MIGRATING RBA: 000000EBED16
DSNU582I =DBA1 DSNUPPCP - REPORT RECOVERY TABLESPACE DSNDB04.TS2 SYSCOPY ROWS
TIMESTAMP = 1996-07-11-14.50.39.568839,  IC TYPE =  F ,  SHR LVL = R,  DSNUM    = 0000,  START LRSN =AD269029E6F6
DEV TYPE  = 3390    ,                    IC BACK =   ,   STYPE   =  ,  FILE SEQ = 0000,  PIT LRSN  = 000000000000
DSNAME    = DSN410.IMAGCOPY.TS2.COPY1                     ,   MEMBER NAME =

DSNU586I =DBA1 DSNUPSUM - REPORT RECOVERY TABLESPACE DSNDB04.TS2 SUMMARY
DSNU588I =DBA1 DSNUPSUM - NO DATA TO BE REPORTED

DSNU592I =DBA1 DSNUPREC - REPORT RECOVERY INFORMATION FOR DATA SHARING MEMBER : DBA1
DSNU583I =DBA1 DSNUPPLR - SYSLGRNX ROWS FROM REPORT RECOVERY FOR TABLESPACE DSNDB04.TS2
 UCDATE     UCTIME      START RBA      STOP  RBA     START LRSN     STOP  LRSN    PARTITION   MEMBER ID
 071196    19532340   00000181AE5B   000000000000   AD26A433AD18   000000000000     0000        0001

DSNU584I =DBA1 DSNUPPBS - REPORT RECOVERY TABLESPACE DSNDB04.TS2 BSDS VOLUMES
DSNU588I =DBA1 DSNUPPBS - NO DATA TO BE REPORTED

DSNU586I =DBA1 DSNUPSUM - REPORT RECOVERY TABLESPACE DSNDB04.TS2 SUMMARY
DSNU588I =DBA1 DSNUPSUM - NO DATA TO BE REPORTED

DSNU592I =DBA1 DSNUPREC - REPORT RECOVERY INFORMATION FOR DATA SHARING MEMBER : DBA2
DSNU583I =DBA1 DSNUPPLR - SYSLGRNX ROWS FROM REPORT RECOVERY FOR TABLESPACE DSNDB04.TS2
 UCDATE     UCTIME      START RBA      STOP  RBA     START LRSN     STOP  LRSN    PARTITION   MEMBER ID
 071196    18563748   AD26902A7DA2   000000000000   AD26902A7DA2   000000000000     0000        0000

DSNU584I =DBA1 DSNUPPBS - REPORT RECOVERY TABLESPACE DSNDB04.TS2 BSDS VOLUMES
DSNU588I =DBA1 DSNUPPBS - NO DATA TO BE REPORTED

DSNU586I =DBA1 DSNUPSUM - REPORT RECOVERY TABLESPACE DSNDB04.TS2 SUMMARY
DSNU588I =DBA1 DSNUPSUM - NO DATA TO BE REPORTED

DSNU597I =DBA1 DSNUPREC - INVALID SYSLGRNX INFORMATION FROMREPORT RECOVERY
DSNU583I =DBA1 DSNUPPLR - SYSLGRNX ROWS FROM REPORT RECOVERY FOR TABLESPACE DSNDB04.TS2
 UCDATE     UCTIME      START RBA      STOP  RBA     START LRSN     STOP  LRSN    PARTITION   MEMBER ID
 062896    18440846   000001786603   000001787CF9   AD163C7A3A14   AD163C7E2D74     0000        0001
 062896    18472490   00000178E394   0000017938D7   AD163D3591D5   AD163ECFF96F     0000        0001
 062896    19112960   0000017A7E92   0000017A8D2B   AD16429757D9   AD1645DAB95A     0000        0001
 062996    00243500   0000017AE1D8   0000017B0831   AD1688927F71   AD1688A78797     0000        0001
 070196    15260658   0000017B4A5F   0000017B72AB   AD19D5CEF0A5   AD19D5EDD131     0000        0001
 070196    15281165   0000017B8EA1   0000017BE7F4   AD19D646724F   AD19DA4C8CB9     0000        0001
 070196    20464562   0000017C27A9   0000017C4000   AD1A1D7AF0FB   AD1A20B71A39     0000        0001
 070196    20481090   000000268F00   00000026F826   AD1A1DCBBA37   AD1A20F05B6B     0000        0002
 070196    21494742   00000027149B   000000272E53   AD1A2B918CBB   AD1A2E5CE618     0000        0002
 070196    22091363   0000002731EA   000000274B41   AD1A2FE9BC43   AD1A2FFF73B1     0000        0002
 070196    22524194   00000027B48D   00000027D0C0   AD1A39A13610   AD1A3CE7B6C8     0000        0002
 070296    00144396   00000027D4C8   0000002900F2   AD1A4BF735FF   AD1A4FE81CAC     0000        0002
 070296    00164642   0000017C4CF4   0000017C5ECB   AD1A4C6BC3C6   AD1A4FA82803     0000        0001

DSNU589I =DBA1 DSNUPREC - REPORT RECOVERY TABLESPACE DSNDB04.TS2 COMPLETE
```

Figure 67. REPORT Utility Output: Recover to a Point-in-Time

9.2 Recovery from Sysplex Failures

In this section we detail some recovery scenarios relating to the loss or failure of:

- DB2 subsystem
- Coupling facility structures
  - Lock structure
  - SCA
  - GBP

We also describe how to delete structures and connections to the structures as this is sometimes necessary to ensure a clean rebuild of a structure during recovery.

9.2.1 DB2 Subsystem Failure

In the event of a DB2 subsystem failure, all modified global locks owned by the subsystem are converted to retained locks. The purpose of these locks is to protect the corresponding resources from any change from other members of the Sysplex, until the failed subsystem is again running.

When restarting the failed DB2 member on the same or another MVS, you back out or commit the pending threads and release the retained locks.

On one of the active systems, issue the following commands to determine how many retained locks you have:

- DISPLAY DATABASE(dbname) SPACE(spacename) LOCKS

  Figure 68 on page 171 shows the display when this command is executed.

```
DSNT360I =DBA1 ***********************************
DSNT361I =DBA1 *  DISPLAY DATABASE SUMMARY
           *    GLOBAL LOCKS
DSNT360I =DBA1 ***********************************
DSNT362I =DBA1     DATABASE = DSN8D41A  STATUS = RW
               DBD LENGTH = 12104
DSNT397I =DBA1
NAME     TYPE PART STATUS            CONNID   CORRID       LOCKINFO
-------- ---- ---- ------------------ -------- ------------ ---------
DSN8S41D TS        RW
DSN8S41E TS    01  RW                                      H(S,PP,I)
    -                  MEMBER NAME DBA1
DSN8S41E TS    01  RW                                      R(X,P)
    -                  MEMBER NAME DBA2
DSN8S41E TS    02  RW,LPL                                  R(IX,PP)
    -                  MEMBER NAME DBA2
DSN8S41E TS    02  RW,LPL                                  H(IS,PP,I)
    -                  MEMBER NAME DBA1
DSN8S41E TS    02  RW,LPL                                  R(X,P)
    -                  MEMBER NAME DBA2
DSN8S41E TS    03  RW                                      H(S,PP,I)
    -                  MEMBER NAME DBA1
DSN8S41E TS    03  RW                                      R(X,P)
    -                  MEMBER NAME DBA2
DSN8S41E TS    04  RW                                      H(S,PP,I)
    -                  MEMBER NAME DBA1
DSN8S41E TS    04  RW                                      R(X,P)
```

Figure 68. Display of Retained Locks

The page set retained locks are identified by an R in the first position of the LOCKINFO in Figure 68. The page (or row) L-locks and the page P-locks are not displayed.

- MODIFY IRLMPROC,STATUS,ALLD

Figure 69 shows the display when this command is executed.

```
F DBA1IRLM,STATUS,ALLD
DXR102I IRA1 STATUS IRLMID=001 022
        SUBSYSTEMS IDENTIFIED                PT01
        NAME     STATUS    RET_LKS   IRLMID  IRLM_NAME
        DBA2     SFAIL          15    001      IRA2
        DBA1     UP              0    001      IRA1
```

Figure 69. MODIFY IRLMPROC

The STATUS can be UP, DOWN, or SFAIL.

RET_LKS shows the number of retained locks held by a subsystem that failed or was running on an IRLM that failed.

You have to release these retained locks, restarting the failed DB2 on the same system or another, in the shortest time.

New parameter in DSNZZPARM

A new DSNZPARM option, RETLWAIT YES or NO (wait for retained locks option) enables you to specify that on an IRLM lock or change request, if the request is incompatible with a retained lock on that resource, DB2 should wait for the retained lock to become available instead of immediately rejecting the request. If the user timeout period expires before the retained lock becomes available, an IRLM timeout condition is returned (usual SQLCODE 911 or 913).

NO, the default, indicates that a request for a resource that has an incompatible retained lock is immediately rejected, and a ″resource unavailable″ condition is returned to the requesting application. YES indicates that DB2 suspends the application for the normal timeout period to wait for the lock to become available.

For RETLWAIT=YES to take effect, you must have IRLM APAR PN79938 installed. If you do not have APAR PN79938 installed, the RETLWAIT option is ignored; all IRLM lock or change requests that are incompatible with retained locks would be treated as if RETLWAIT=NO.

If you use the MVS 5.2 ARM, or you have other automation to ensure that failed DB2 members are restarted quickly in case of a failure, we recommend that you use RETLWAIT=YES. If you do not have automation in place to ensure that failed DB2 members are restarted quickly, we recommend that you use RETLWAIT=NO (the default).

9.2.2 Coupling Facility Failure

Coupling facility failures can mean serious outages for users. Not having a lock structure or SCA can cause an entire group to come down abnormally. GBP failure does not cause the group to come down, but it can still mean loss of availability for applications depending on the data in that GBP.

Careful preparation for structure and connectivity failures can greatly reduce the impact of coupling facility outages on your end users. To best prepare yourself for both types of failures, you must have the following:

- An alternate coupling facility
- An active SFM policy and system weights specified

- Alternative coupling facility information provided on the preference list of each of the structures in the CFRM policy
- Adequate storage in an alternate coupling facility to rebuild or reallocate structures as needed.

The cases of hardware failure of the coupling facility and the loss of 100% connectivity of one coupling facility with all members of the DB2 Sysplex have the same consequences. The details depend on the type of structures that were allocated in the failing coupling facility .

9.2.2.1 Lock Structure Failure: The lock structure can be automatically rebuilt on another coupling facility, provided you have defined the other coupling facility in the preference list and the SFM policy is active. The rebuild can proceed even if there are some failed DB2 members. This operation should be transparent for applications, as it generally takes only a few seconds. You receive these messages:

- DXR143I irlmnm Rebuilding lock structure because it has failed or an IRLM lost connection to it
- DXR146I irlmnm Rebuild of the lock structure completed successfully

A rebuild of the lock structure may fail because:

- There is no alternate coupling facility defined in the CFRM policy
- There is not enough storage in the alternate coupling facility
- A connectivity error was found accessing the alternate coupling facility
- The lock structure fails at the same time an IRLM fails (case of a CEC failure containing one member and one coupling facility in an LPAR).

If the rebuild of the lock structure fails and the old lock structure is not usable, all DB2 members will fail, and a group restart must be performed to recover the lost information from the logs.

9.2.2.2 SCA Failure: The SCA structure can be automatically rebuilt on another coupling facility , provided you have defined the other coupling facility in the preference list and the SFM policy is active. The rebuild can proceed even if there are some failed DB2 members. This operation should be transparent for applications as it generally takes only a few seconds. You receive these messages:

- DSN7502I SCA structure strnm failure, attempt to rebuild is in progress
- DSN7503I SCA structure strnm rebuild successful

A rebuild of the SCA structure may fail (MSG DSN7504I) because:

- There is no alternate coupling facility defined in the CFRM policy

- There is not enough storage in the alternate coupling facility
- A connectivity error was found accessing the alternate coupling facility
- The SCA structure fails at the same time an IRLM fails (case of a CEC failure containing one member and one coupling facility in an LPAR).

If the rebuild of the SCA structure fails and the old SCA structure is not usable, all DB2 members will fail, and a group restart must be performed to recover the lost information from the logs.

9.2.2.3 Group Buffer Pool Failure: If one or many GBPs are lost, you receive several messages:

- Message DSNB304I: structure set to ″damage assessment pending″

  DB2 does a damage assessment to identify the modified pages that might have been in the GBP at the time of the failure.

- Message DSNB250E: pages are added in LPL if write operations to the group buffer pool "must complete" (like commit, rollback, restart) have been interrupted.

  DB2 marks entries in the LPL (see Figure 70).

```
DSNB250E =DBA2 DSNKUNR2 A PAGE RANGE WAS ADDED TO THE 186
LOGICAL PAGE LIST
          DATABASE NAME=DSN8D41A
          SPACE NAME=XEMP2
          DATA SET NUMBER=1
          PAGE RANGE X′000000′ TO X′FFFFFF′
          START LRSN=X′AD0C204054DB′
          END LRSN=X′AD0C204054DB′
          START RBA=X′00000021B9B2′
```

Figure 70. Message DSNB250E

- Message DSNB32xI: page sets are set to GRECP status. DB2 marks the affected table spaces, indexes, or partitions as GRECP (see Figure 71 on page 175).

```
DSNB228I =DBA2 DSNB5SCO GROUP BUFFER POOL GBP1 179
          CANNOT BE ACCESSED FOR READ_STGSTATS
          MVS IXLCACHE REASON CODE=X'0C1C0C25'
DSNB314I =DBA2 DSNB1REE DAMAGE ASSESSMENT TO BE 180
          TRIGGERED FOR GROUP BUFFER POOL GBP1
          REASON=STRFAIL
DSNB304I =DBA2 DSNB1DAO GROUP BUFFER POOL GBP1 WAS 181
SET
          'DAMAGE ASSESSMENT PENDING' STATUS
DSNB321I =DBA1 DSNB1DA1 DBNAME DSN8D41A SPACE NAME 125
XEMP2 IN GROUP BUFFERPOOL GBP1
          IS IN GROUP BUFFERPOOL RECOVERY PENDING STATE
DSNB320I =DBA1 DSNB1DA1 DBNAME DSN8D41A SPACE NAME 126
XEMP1  PARTITION 2 IN GROUP BUFFERPOOL GBP1
          IS IN GROUP BUFFERPOOL RECOVERY PENDING STATE
DSNB320I =DBA1 DSNB1DA1 DBNAME DSN8D41A SPACE NAME 127
DSN8S41E  PARTITION 2 IN GROUP BUFFERPOOL GBP1
          IS IN GROUP BUFFERPOOL RECOVERY PENDING STATE
DSNB305I =DBA2 DSNB1DAO 'DAMAGE ASSESSMENT PENDING' 182
STATUS WAS
          CLEARED FOR GROUP BUFFER POOL GBP1
```

Figure 71. Message DSNB32xI

- Message DSNB305I indicates that damage assessment is cleared.

Displaying the GRECP and LPL: You can issue one of these commands to display the restricted page sets:

- -DISPLAY DATABASE(*) SPACE(*) LPL LIMIT(*)
- -DISPLAY DATABASE(*) SPACE(*) RESTRICT LIMIT(*)
- -DISPLAY DATABASE(*) SPACE(*) LOCKS LIMIT(*).

Figure 72 on page 176 shows the result of the DISPLAY ... LPL ... command.

```
DSNT360I =DBA1 ***********************************
DSNT361I =DBA1 *  DISPLAY DATABASE SUMMARY
           *    GLOBAL LPL
DSNT360I =DBA1 ***********************************
DSNT362I =DBA1     DATABASE = DSN8D41A  STATUS = RW
               DBD LENGTH = 12104
DSNT397I =DBA1
NAME     TYPE PART STATUS            LPL PAGES
-------- ---- ---- ------------------ ------------------
DSN8S41D TS        RW
DSN8S41E TS    01  RW
DSN8S41E TS    02  RW,LPL,GRECP       100001-100002
DSN8S41E TS    03  RW
DSN8S41E TS    04  RW
DSN8S41P TS        RW
DSN8S41R TS        RW
DSN8S41S TS        RW
DSN8S41X TS        RW
XACT1    IX        RW
XACT2    IX        RW
XDEPT1   IX        RW
XDEPT2   IX        RW
XDEPT3   IX        RW
XEMPPROJ IX        RW
XEMP1    IX    01  RW
XEMP1    IX    02  RW,LPL,GRECP       000000-FFFFFF
XEMP1    IX    03  RW
XEMP1    IX    04  RW
XEMP1MLS IX        RW
XEMP2    IX        RW,LPL,GRECP       000000-FFFFFF,000000-FFFFFF
XPARTS   IX        RW
XPROJAC1 IX        RW
XPROJ1   IX        RW
XPROJ2   IX        RW
```

Figure 72. Display of GRECP and LPL

Recovering: To remove the LPL and GRECP statuses, you have to issue this command:

-START DATABASE(dbname/*) SPACE(tsname/*)

This is the recommended way of recovering the availability to the page set.

DB2 has to read the log since the last GBP checkpoint and the data or index pages from DASD and apply the changes.

Multiple START DATABASE(dbname) SPACE(*) commands running in parallel should complete faster than one START DATABASE(*) SPACE(*). Spread out the START commands on many DB2 members to finish earlier.

If the catalog or directory page sets are GRECP, start with this order:

-START DATABASE(DSNDB01) SPACE(*)

-START DATABASE(DSNDB06) SPACE(*)

This START performs a global drain on the object and can go into timeout if the ″STARTED″ object is not available (Msg DSNI005I). This process is asynchronous and sends a DSNI021I message when the recovery is complete.

You can also remove the GRECP and LPL statuses by executing:

- RECOVER utility on the object

  This method is suitable in case of a previous point-in-time recovery.

  The only exception is when a logical partition of a Type 2 nonpartitioned index has both LPL and RECP status. If you want to recover the logical partition using RECOVER INDEX with the PART keyword, you must first use the START DATABASE command to clear the LPL pages.

- LOAD utility with the REPLACE option on the object

  This method is suitable for a previous point-in-time recovery.

- REPAIR utility using SET with NORCVRPEND

  This method can leave your data in an inconsistent state (**not recommended**).

- START DATABASE ACCESS(FORCE) command

  This method can leave your data in an inconsistent state (**not recommended**).

Remove Retained Locks First

None of the items in the above list works if retained locks are held on the object. You must restart any failed DB2 that is holding retained locks.

9.3 Deleting Structures and Connections

Sometimes you have to get rid of old structures that are still allocated in order to start with new structures.

Structures on the failed coupling facility become persistent and/or structure connectors become failed-persistent. This may happen for a number of different reasons, such as:

- The structure REBUILD failed.
- An alternate coupling facility did not exist.
- The structure REBUILD is not supported by the owning application.
- A subsystem failure occurred.

If the application that owns a persistent structure or connector is restarted while the coupling facility remains unavailable, its attempt to connect to the structure will not succeed. MVS maintains a recovery bind, represented by the persistent structure or failed-persistent connection, to the instance of the structure that was in use at the time of the failure. Thus, if you want to restart the application and have it allocate another instance of the structure in an alternate coupling facility, you first must break the recovery bind to the old instance of the structure by deleting the persistent connectors and/or the structure before structure allocation will succeed.

To determine which structures remain allocated on the failed coupling facility, issue the following command:

```
D XCF,CF,CFNAME=failed_cfname
```

To delete the allocated structures, display the structure information for each structure in the failed coupling facility:

```
D XCF,STR,STRNM=strname
```

The structure may have failed-persistent connectors, or the structure itself may be persistent, or both. If the structure has failed-persistent connectors, delete all failed persistent connectors as described in 9.3.1, “Deleting Failed-Persistent Connectors.” If the structure is persistent (DISPOSITION: KEEP in IXC360I), delete the structure as described in 9.3.2, “Deleting Persistent Structures” on page 179

9.3.1 Deleting Failed-Persistent Connectors

Issue SETXCF FORCE to delete all failed-persistent connectors:

```
SETXCF FORCE,CON,CONNM=ALL,STRNM=strname
```

Because the MVS system does not have connectivity to the failed coupling facility, you will receive the message shown in Figure 73 on page 179.

```
IXC354I THE SETXCF FORCE REQUEST FOR CONNECTION
conname IN STRUCTURE strname WAS ACCEPTED:
REQUEST WILL BE PROCESSED ASYNCHRONOUSLY
```

Figure 73. Deleting Failed-Persistent Connectors

Display the structure again:

D XCF,STR,STRNM=strname

The number of connectors to the structure should now be zero.

9.3.2 Deleting Persistent Structures

The number of connectors in message IXC360I must be zero before proceeding. If ACTIVE connectors exist (see connection state in IXC360I), invoke recovery procedures for the connector, or CANCEL the connector′s address space to make the connector disconnect from the structure. Then, if the connector enters a failed-persistent state, force the connector using the SETXCF FORCE,CON command as described above.

Issue SETXCF FORCE to delete the structure:

SETXCF FORCE,STR,STRNM=strname

Because the MVS system does not have connectivity to the failed coupling facility, you will receive the message shown in Figure 74.

```
IXC353I THE SETXCF FORCE REQUEST FOR STRUCTURE
strname WAS ACCEPTED:
REQUEST WILL BE PROCESSED ASYNCHRONOUSLY
```

Figure 74. Deleting Persistent Structures

Once all failed-persistent connectors are forced and/or the structure is forced, the structure should enter an ″in transition″ state. If you display the structure again, message IXC360I should contain the statements shown in Figure 75 on page 180.

```
STRUCTURE IN TRANSITION
-----------------------
REASON IN TRANSITION: CONNECT OR DISCONNECT IN PROGRESS
```

Figure 75. Message IXC360I: Structure in Transition

When the structure enters an ″in transition″ state, it is targeted for deletion by the coupling facility resource manager. At this point, a new instance of the structure can be allocated in another coupling facility.

9.4 Disaster Backup and Recovery

In this section we describe the disaster backup and recovery considerations for a DB2 data sharing group. These considerations differ from those for the non-data-sharing case. To highlight these differences we give a short overview of the disaster recovery procedure for a non-data-sharing DB2 system and then cover these topics for a data sharing group:

- Disaster recovery backup
- Disaster recovery flow
- Disaster recovery preparation

We conclude the chapter by comparing disaster recovery for a DB2 data sharing group and a non-data-sharing DB2 system.

9.4.1 Disaster Recovery for a Non-Data-Sharing DB2 System

The procedure for disaster recovery for a non-data-sharing DB2 system is described in the *IBM DATABASE 2 for MVS/ESA Version 4 Administration Guide* The procedure can be summarized as follows:

1. Set up the environment (for example, ICF catalogs, DB2 libraries).
2. Recover the BSDSs.
3. Run DSN1LOGP to analyze which transactions were in process at the end of the last archive log and whether there were utilities inflight at that moment.
4. Create a conditional restart control record, using the change log inventory utility DSNJU003:

   ```
   CRESTART CREATE,ENDRBA=nnnnnnnnn000,FORWARD=YES,BACKOUT=YES
   ```

   where nnnnnnnnn000 equals a value one more than the ENDRBA of the latest archive log.
5. Conditionally restart DB2.

6. Resolve the indoubt units of work.
7. Recover, recover, recover ..... and take care of utilities.

9.4.2 Disaster Recovery Backup

For the disaster recovery of a DB2 data sharing group you cannot use the procedure described in 9.4.1, "Disaster Recovery for a Non-Data-Sharing DB2 System" on page 180 for each member of the group. Because the RBAs used for the conditional restart of the members are completely independent, from a data sharing point of view the consistency of the data would be thoroughly corrupted. Therefore a groupwide point of consistency (GPOC) must be built during the disaster backup process for a DB2 data sharing group.

9.4.2.1 Using the ARCHIVE LOG Command: To create GPOC, issue the following command:

```
ARCHIVE LOG MODE(QUIESCE)
```

This command tries to quiesce each active member of the data sharing group. In each active member, it stops all new update activity for a specified period of time and attempts to bring all existing users to a point of data consistency after a commit or rollback.

If any DB2 member is in a failed state, fails during quiesce processing, or is stopped with outstanding units of work, the ARCHIVE LOG command fails, and the remaining active members allow update activity to proceed.

If no indoubt units of work are left on all quiesced members, active or inactive, for each active member this point of consistency is captured in the current active log data set. The RBA of this point of consistency—the so-called quiesce RBA—is written to the BSDS. Then the archive operation can continue for each active member in the group: the current active log data set is truncated, and another log data set in the inventory becomes current.

If for any member a systemwide point of consistency cannot be reached during the quiesce period, which is a length of time you can control, the execution of the ARCHIVE LOG command fails, and an error message is issued.

9.4.2.2 Disaster Backup Procedure: These are the basic steps of a disaster backup procedure for a DB2 data sharing group:

1. Check for utilities.

   Use the DISPLAY UTIL command.

2. Check the status of all objects.

   Use the DISPLAY DATABASE command with the RESTRICT parameter.

3. Ensure that all members are up. (Otherwise there is log data for which no offload is triggered.)

Use the DISPLAY GROUP command.

4. Ensure that all objects are recoverable.

   This check can be done by appropriate SQL statements against the DB2 catalog. Use a user-written program or DSNTEP2 for this purpose.

5. Issue ARCHIVE LOG MODE(QUIESCE) WAIT(YES) TIME(n)

   Verify that the ARCHIVE command completes successfully. For each member check that:

   - The quiesce has been successful.
   - The triggered offload has completed successfully.

   Use the print log map Utility DSNJU003 for this verification.

6. Obtain the information about the quiesce RBAs and the offloads, and store it in a secure location.

   There are several ways of obtaining the information:

   - Save the print log map output.
   - Trigger a second groupwide offload by ARCHIVE LOG SCOPE(GROUP).
   - Run the IDCAMS REPRO utility against the BSDSs.

9.4.2.3 Automating the Procedure: DB2 disaster backup is only one part of the disaster backup for a whole MVS system or a whole Sysplex. The activities for DB2 disaster backup have to be coordinated with other backup activities. Therefore the regular execution of DB2 disaster backup must run under the control of your job scheduling system (for example, OPC/ESA). All of the steps in 9.4.2.2, "Disaster Backup Procedure" on page 181 can be embedded in a REXX EXEC. For example,

- The DB2 commands can be easily executed from REXX because DSN is a TSO command.
- The print log map utility, DSNJU003, can be invoked from REXX too. REXX is powerful for analyzing the print log map output.
- Because there is no REXX DB2 interface for TSO and batch, the execution of SQL statements against the DB2 catalog is rather awkward but possible. You can call DSNTEP2, for example.

You have to set up a strategy or policy for DB2 disaster backup that covers the following questions and points:

- How many GPOCs per day does your business need?
- At what points in time do you want to establish the GPOCs?
- Is it possible to establish a GPOC at these points in time? Perhaps you are in a dilemma: On the one hand you cannot increase the quiesce period because your users

do not accept the potential increase of the transaction response time at these points in time; on the other hand the ARCHIVE LOG MODE(QUIESCE) fails because of outstanding units of works. You may have to change application programs to increase the commit frequency. Unfortunately the failed quiesce only gives the number of outstanding units of work. There is no information that identifies these units of work directly.

- How many nonrecoverable objects do you accept? Do you want the GPOC for the whole DB2 data sharing group to fail because there is one—perhaps not so important—object that is not recoverable? During the implementation phase of a new application there may be a lot of nonrecoverable objects. Perhaps you have to build an exclude list of objects, which should not disturb a GPOC. Similar considerations apply for utilities.
- If there are updates in the DB2 data sharing group to be quiesced that come, for example, from IMS transactions that update DL1 databases, CICS transactions that update VSAM data, or distributed units of work, the scope of the consistency of your data exceeds the scope of your DB2 data. Therefore in the case of a disaster recovery your data may be inconsistent although the data in the DB2 data sharing group is consistent. This consideration is not specific to data sharing; it applies to the non-data-sharing too.

9.4.3 Disaster Recovery Flow

In this section we describe the general flow of the disaster recovery based on the disaster backup described in 9.4.2.2, “Disaster Backup Procedure” on page 181.

1. Set up the environment.

   We mention only those points that are specific for the data sharing case.

   There must be a fully functional parallel Sysplex in your remote location. You may use a different configuration and different hardware technology, but you must not change the names of the DB2 members and the names of the DB2 structures. You may adapt the size of the DB2 structures to your needs.

2. Allocate the data sets of all members.

   Use IDCAMS DEFINE CLUSTER for:

   - Catalog and directory
   - Active logs and BSDSs (for each member).

3. Restore the BSDSs for all members.

   Use copies that contain the information about the offload triggered by the ARCHIVE command. The copies of the BSDSs generated by the offload triggered by ARCHIVE LOG MODE(QUIESCE) do not contain information about the archive logs generated by that offload. Therefore we recommend forcing a second offload (ARCHIVE LOG SCOPE(GROUP)) after the successful GPOC and using these copies of the BSDSs for the restores.

4. Delete/add active logs for all members.

   You have to expect a return code of 4 because there are nonreusable active log data sets: those that are offloaded and the new, active data sets at the point in time when the offload has taken place.

5. Add archive logs for all members.

   This step is required only if the BSDS are restored from copies that do not contain the information about the offload triggered by the ARCHIVE command. Basically this step is superfluous and error prone.

6. Create the conditional restart control records (CRCRs) for all members.

   The change log inventory utility (DSNJU003) control statement to create the CRCR needed must be of the following form:

   ```
   CRESTART CREATE,ENDRBA=nnnnnnnnn000,FORWARD=YES,BACKOUT=YES
   ```

   where nnnnnnnnn000 equals a value one more than the quiesce RBA created by the ARCHIVE command.

   The normal restart processing is not changed (FORWARD=YES, BACKOUT=YES). Only truncation of the log takes place (ENDRBA=nnnnnnnnn000).

7. Modify the DSNZPARM modules for all members:

   - DEFER ALL

     Prevent DB2 from accessing the nonexistent spaces.

   - DDF=NO, RLF=NO

     The communications database (CDB) and the resource limit facility database do not exist. Therefore it does not make sense to start the DDF and resource limit facility (RLF) during DB2 startup. You would see superfluous messages.

   - (SITETYP=RECOVERYSITE)

     Use RECOVERYSITE if the image copies to be used for disaster recovery are backup copies.

     If you use RECOVERYSITE, whole table spaces instead of pages are put into the LPL at restart.

8. Clean up the coupling facilities

   All DB2 structures used by the members to be conditionally restarted must be deallocated; otherwise incorrect information is taken from these structures during restart.

   To deallocate a structure, all failed-persistent connections have to be removed:

   ```
   SETXCF FORCE,CON,STRNM=strname,CONNM=ALL
   ```

   and the structures have to be deleted:

```
SETXCF FORCE,STR,STRNM=strname
```

9. Restart all DB2 members.

   Start one DB2 member with ACCESS(MAINT). During initialization this member detects the CRCR in the BSDS (message DSNJ245I). To accept the truncation of the log and make DB2 continue the startup, you have to reply “Y” to that WTOR.

   After the completion of the CSR, DB2 tries a peer CSR. During that process it detects for each member that the log environment of the peer member might have changed because the system time stamp in the BSDS of the member is lower than the utility time stamp set by the change log inventory utility that inserted the CRCR. Therefore DB2 cannot perform the peer restart for the peer members, so it requests the manual startup of each member. For each member a write to operator with reply (WTOR) message (DSNR020I) appears:

   DSNR020I csect-name START MEMBER member, OR REPLY ′NO′ OR ′QUIESCED′

   Explanation: During a group restart, the group member issuing this message attempted to access the BSDS for a member and either was unable to access the BSDS or discovered that the BSDS was modified by the change log inventory (DSNJU003) utility since that member was last started. In either case, this DB2 cannot perform peer restart for the member.

   For group restart to continue, the identified member must be started. If it is not possible to start the other member, and you want to terminate this group restart, you can reply “NO” to this message. Replying “NO” causes this DB2 to terminate with a reason code of 00D900E1. You then have to reply “NO” to this message for all remaining members participating in this group restart.

   If you are sure that the identified member was quiesced the last time it was started, that is, it was stopped with a -STOP DB2 MODE(QUIESCE) command and it stopped with no indoubt URs and no distributed resynchronization responsibility, you can reply “QUIESCED” to this message. Replying “QUIESCED” causes message DSNR030I to be issued as a reminder that group restart will continue without using the log from the identified member.

   System Action: This group member waits for the identified member to be started or for a valid reply to this message:

   - If the identified member is started, group restart continues.
   - If the reply is “NO,” this DB2 terminates with a reason code of 00D900E1.
   - If the reply is “QUIESCED,” group restart continues without using the log from the identified member.

   Before you reply “QUIESCED,” take note of the fact that logs that contain image copy records of those catalog and directory table spaces that do not have SYSCOPY entries are required.

Typically each member must be restarted. If you reply “QUIESCED,” you must be very sure not to cause inconsistency of data.

Start the other members with ACCESS(MAINT). Reply “Y” to the WTOR (message DSNJ245I).

After the completion of the CSR of the peer members, the first member finishes its CSR. All members now start their forward log recovery phase. Because of DEFER ALL in the DSNZPARM module, those pages necessary for the restart of an object are put in the LPL.

All members synchronize at the end of the forward log recovery phase. Then the back recovery phase starts. A lot of resource unavailable conditions may occur.

10. (Quiesce some DB2 members.)

    After the successful restart of all members, some members may be quiesced. If the Sysplex configuration at your remote location is not as large as it is at your local site, you may decide to use fewer members for the offsite workload. The following recoveries do not need all DB2 members to be up.

11. Recover the catalog and directory.

```
//JOBLIB   DD DISP=SHR,DSN=DSN410.SDSNLOAD
//REC#CD1  EXEC PGM=DSNUTILB,PARM='DSGC,REC#CD1'
//SYSPRINT DD SYSOUT=*
//SYSIN    DD  *
  RECOVER TABLESPACE DSNDB01.SYSUTILX
//REC#CD2  EXEC PGM=DSNUTILB,PARM='DSGC,REC#CD2'
//SYSPRINT DD SYSOUT=*
//SYSIN    DD  *
  RECOVER TABLESPACE DSNDB01.DBD01
//REC#CD3  EXEC PGM=DSNUTILB,PARM='DSGC,REC#CD3'
//SYSPRINT DD SYSOUT=*
//UTPRINT  DD SYSOUT=*
//SYSIN    DD  *
  RECOVER INDEX (ALL) TABLESPACE DSNDB01.SYSUTILX
//REC#CD4  EXEC PGM=DSNUTILB,PARM='DSGC,REC#CD4'
//SYSPRINT DD SYSOUT=*
//UTPRINT  DD SYSOUT=*
//SYSIN    DD  *
  RECOVER TABLESPACE DSNDB06.SYSCOPY
  RECOVER INDEX (ALL) TABLESPACE DSNDB06.SYSCOPY
  RECOVER TABLESPACE DSNDB01.SYSLGRNX
  RECOVER INDEX (ALL) TABLESPACE DSNDB01.SYSLGRNX
  RECOVER TABLESPACE DSNDB06.SYSDBAUT
  RECOVER INDEX (ALL) TABLESPACE DSNDB06.SYSDBAUT
  RECOVER TABLESPACE DSNDB06.SYSUSER
  RECOVER TABLESPACE DSNDB06.SYSDBASE
  RECOVER INDEX (ALL) TABLESPACE DSNDB06.SYSDBASE
  RECOVER INDEX (ALL) TABLESPACE DSNDB06.SYSUSER
  RECOVER TABLESPACE DSNDB06.SYSGROUP
  RECOVER INDEX (ALL) TABLESPACE DSNDB06.SYSGROUP
  RECOVER TABLESPACE DSNDB06.SYSGPAUT
  RECOVER INDEX (ALL) TABLESPACE DSNDB06.SYSGPAUT
  RECOVER TABLESPACE DSNDB06.SYSPLAN
  RECOVER INDEX (ALL) TABLESPACE DSNDB06.SYSPLAN
  RECOVER TABLESPACE DSNDB06.SYSPKAGE
  RECOVER INDEX (ALL) TABLESPACE DSNDB06.SYSPKAGE
  RECOVER TABLESPACE DSNDB06.SYSSTATS
  RECOVER INDEX (ALL) TABLESPACE DSNDB06.SYSSTATS
  RECOVER TABLESPACE DSNDB06.SYSSTR
```

```
RECOVER INDEX (ALL) TABLESPACE DSNDB06.SYSSTR
RECOVER TABLESPACE DSNDB06.SYSVIEWS
RECOVER INDEX (ALL) TABLESPACE DSNDB06.SYSVIEWS
RECOVER TABLESPACE DSNDB01.SCT02
RECOVER INDEX (ALL) TABLESPACE DSNDB01.SCT02
RECOVER TABLESPACE DSNDB01.SPT01
RECOVER INDEX (ALL) TABLESPACE DSNDB01.SPT01
```

The first step can be replaced by an initialization of SYSUTILX because utilities cannot be restarted in a remote location.

12. Image copy the catalog and directory.

13. Perform integrity checks for the catalog and directory:

    - DSN1CHKR

      DSN1CHKR should not be run on table spaces DSNDB06.SYSCOPY, DSNDB06.SYSGPAUT, DSNDB06.SYSPKAGE, DSNDB06.SYSSTATS, DSNDB06.SYSSTR, DSNDB06.SYSUSER, DSNDB01.SCT02, DSNDB01.SPT01, DSNDB01.SYSLGRNX, and DSNDB01.SYSUTILX.

      DSN1CHKR can use image copy data sets as input.

    - DSN1COPY, DSN1PRNT

      DSN1COPY can use image copy data sets as input.

    - CHECK DATA utility

    - CHECK INDEX utility

14. Set up workfile database for all active members.

15. Recover other spaces.

    Improve the performance of the recovery process by maximizing the degree of parallelism. This degree and the performance of the recovery process are most probably limited by I/O constraints.

    - Adjust the number of recover jobs to run in parallel.

    - Choose which objects are to be recovered in parallel.

    - Choose which table spaces are to be recovered together within one RECOVER statement.

      Remark: If you recover several table spaces by one RECOVER TABLESPACE statement, then:

      – The restores of the table spaces from their image copies are processed successively.

      – There is only one log apply phase.

      – One restriction applies: you cannot use the DSNUM keyword.

      By this, the total number of log apply phases can be reduced.

- Put all archive logs needed on DASD.

Your strategy depends on:

- The amount of log data to applied
- The size and number of objects to be recovered
- The configuration and availability of DASD
- The availability of tape units
- The distribution of the table spaces and index spaces across the volumes.

Avoid parallel recover jobs to have common target volumes.

If you start to develop an optimal recovery strategy on the day the disaster recovery takes place, it is definitely too late.

16. Image copy other spaces.
17. STOP DB2.
18. Restore original DSNZPARM modules.
19. START DB2.

9.4.4 Disaster Recovery Preparation

As seen in section 9.4.3, "Disaster Recovery Flow" on page 183, a number of steps have to be executed during the process of disaster recovery:

1. Define cluster for the catalog and directory.
2. Define cluster for the BSDSs.
3. Define cluster for the active logs.
4. Define cluster for user-defined spaces.
5. Restore the BSDSs.
6. Declare new logs (change log inventory utility).
7. Create a CRCR.
8. Assemble and link DSNZPARM.
9. Initialize SYSUTILX.
10. Recover the catalog and directory.
11. Integrity check the catalog and directory.
12. Image copy the catalog and directory.
13. Create the workfile database.
14. Recover system spaces (CDB, RLFDB, ...).

15. Image copy system spaces (CDB, RLFDB, ...).

16. Alter of STOGROUPS.

17. Recover user spaces.

18. Image copy user spaces.

An actual version of these jobs, source, DDL, IDCAMS control statements, utility control statements, and so on, should be available at the moment of disaster recovery. They should be prepared in advance and be part of the backup process to have them available at the recovery site.

Some of these components are stable and they change only seldom, for example, each time a new DB2 release is implemented. Some of these components may be very volatile; for example, the DEFINE CLUSTER statements may not reflect the actual space requirements or the correct volume serial number. Therefore you can also gather information from:

ICF catalogs
  IDCAMS LISTCAT

DB2 catalog and directory
  SQL SELECT (DSNTEP2)
  REPORT utility

BSDSs
  print log map utility DSNJU004

during disaster backup and send it as you do image copies and archive logs to a secure location. In the case of a disaster recovery, this information can be used, for example, to:

- Generate actual DEFINE CLUSTER statements.
- Build optimized recover jobs.
- Determine the amount of log data needed.

REXX is a powerful tool for gathering this information at the local site as well as for processing it at the remote site.

Some changes may be necessary in the case of a disaster recovery. For example, if the DASD environment at the remote site differs from that at your local site, you should have available some prepared generic jobs for the alteration of storage groups.

9.4.5 Data Sharing and Non-Data-Sharing Disaster Recovery

The main difference between data-sharing and non-data-sharing disaster recovery is that for a DB2 data sharing group, you must use a GPOC, and for a non-data-sharing system you can use a systemwide point of consistency. Therefore, it makes sense to differentiate between a disaster backup and recovery approach that is based on a systemwide point of consistency and a GPOC and an approach that is not. Refer to the procedure described in Chapters 4-7 “Recovery Scenarios: Remote Site Recovery” in *IBM DATABASE 2 for MVS/ESA Version 4 Administration Guide.*

For a disaster recovery approach that is not based on a point-of-consistency, the disaster recovery is more complicated. You have to analyze which transactions were in process at the point of the disaster, and you may have to manually resolve indoubt URs.

For an approach that is based on a point-of-consistency, the disaster recovery is more complicated. It may be very difficult to find (enough) points in time when you can execute the ARCHIVE LOG MODE(QUIESCE) command to produce a systemwide point-of-consistency and/or a GPOC. consistency.

Chapter 10. Performance and Tuning Considerations

The DB2 data sharing architecture makes very efficient use of the S/390 Parallel Sysplex environment. The coupling facility is used as a fast accessible shared memory to detect global lock contention and provide for data buffer coherence. Local optimization mechanisms like hierarchical locking are used wherever possible to limit the additional capacity required for data sharing. However, after implementation of data sharing, you should take some actions to tune your data sharing environment.

Lack of tuning in a non-data-sharing environment can lead to serious performance problems in a data sharing environment because of increased complexity.

In this chapter we offer some guidance on the main performance factors in a data sharing environment and how you can control them. We cover the following topics:

- Factors influencing data sharing performance
- Performance objectives
- Monitoring tools
- Verifying the performance of a parallel sysplex
- Tuning DB2 locking
- Tuning GBPs
- Transaction run-time criteria: data sharing and non-data-sharing

For detailed performance analyses of the data sharing system, refer to the redbook entitled *DB2 for MVS/ESA Version 4 Data Sharing Performance Topics*.

10.1 Factors Influencing Data Sharing Performance

Many factors influence the degree of data sharing and the additional CPU capacity delta required for data sharing. Mileage is influenced heavily by the number of data objects where DB2 determines there is intersystem read-write interest. The design of DB2 data sharing enables DB2 to take advantage of natural or forced process affinity routing. Forced transaction routing—that is, isolating a given application and its private data onto a single DB2 system—might be considered where best performance is the exclusive concern or the degree of contention is excessive.

The degree of data sharing is an abstract term used to cover the three factors that interplay in determining the capacity delta required for DB2 data sharing:

- The percentage of total CPU processing capacity across the systems accessing DB2 shared data

- Within the subset of overall CPU processing capacity defined above, the ratio of SQL processing against shared data
- The data intensity of the SQL workload in terms of driving the coupling facility; for example, the number of IRLM requests going to the lock structure, or the number of getpages going to the associated GBP.

It is very important to control global lock contention. Global lock contention requires global management and incurs the relatively expensive overhead of message passing. Global lock contention, which is the sum of real and false contention, should be as low as possible.

An important factor determining the CPU data sharing overhead is the relative single-engine processor speed of the sender CPC running MVS and the receiver CPC running the CFCC. Because the majority of coupling facility accesses are performed synchronously, the number of instructions lost while waiting for a coupling facility response will be higher if the difference between the single-engine processor speeds is high.

It is almost impossible to determine in advance the overhead caused by data sharing in your environment because this overhead depends mostly on your workload profile, degree of data sharing, physical database design, bind options, and hardware configuration. The mileage achieved in practice in your environment will vary. The point is that many variables affect performance, and no single rule-of-thumb can be prescribed for the additional CPU processing capacity delta required for data sharing that applies in each and every environment.

10.2 Performance Objectives

The design point for the data sharing solution balances good performance with random scheduling of transactions across a data sharing group. There are three main performance objectives:

- A DB2 V4 non-data-sharing system and a DB2 V4 one-way data sharing system should have near equivalent performance values for the same workload in a Parallel Sysplex environment.
- In a DB2 V4 two-way sharing configuration and a 100% shared read-write interest, the data sharing overhead should be equal to or less than 17%.

  The second objective translates to an additional CPU capacity delta of 17% to achieve equivalent throughput.

  The data sharing overhead of 17% includes a fixed portion of approximately 3% for MVS Sysplex overhead and up to 14% DB2 data sharing overhead. The MVS overhead is required for inter-MVS communication. DB2 data sharing is a function required as needed for global locking protocols and data buffer coherence controls.
- As each additional member is added to the group, the additional data sharing overhead should be about 0.5%.

Considerable effort has been made to reduce the costs of global locking, the volume of global lock contention, and data buffer coherence control to meet the performance objectives explained above.

We strongly recommend monitoring the behavior of your system, especially with regard to the workload during migration to data sharing.

10.3 Monitoring Tools

Several tools enable you to access information about distribution of response time, storage allocation, and other performance-relevant information. In a data sharing environment you need additional information about the structures that DB2 uses and a global view of the performance-relevant information. In this section we briefly describe some tools that you can use to get all of the information required to tune your Sysplex.

- Resource Measurement Facility (RMF)

  The RMF provides you with both a single-system and a Sysplex view. The RMF Coupling Facility Report contains information about the activities of DB2 structures.

- DB2 Performance Monitor (DB2 PM)

  Use the DB2 PM to monitor group and member activities of the DB2 data sharing group. It gives you detailed information about the DB2 subsystem and the transaction activities in DB2. You can use the DB2 PM in online or batch mode. In batch mode you can produce group level reports and member level reports.

- DB2 DISPLAY commands

  The DB2 DISPLAY commands show you a snapshot of DB2 activities at the time the commands are executed.

- XCF DISPLAY commands

  The XCF DISPLAY commands provide useful information about the state of the coupling facility and its structures.

In the sections that follow we use the reports of these monitoring tools to describe key performance values.

10.4 Verifying the Performance of a Parallel Sysplex

It is important to review the performance of the Parallel Sysplex. The different hardware configurations and processor speeds of CPCs running in a Parallel Sysplex result in different service times needed for access to the structures using the coupling facility links. Because the coupling facility links are fast channels, such configurations as a coupling facility running in LPAR mode could have a big impact.

The RMF Coupling Facility Report (Figure 76 on page 194) describes the activity of all structures in the coupling facility for a given time period (default 15 min). It also shows you the service times needed to access every structure. To get the report you have to start SMF classes 70 to 79.

```
------------------------------------------------------------------------------------------------------------------------------
                                                C O U P L I N G   F A C I L I T Y   S T R U C T U R E   A C T I V I T Y
    --------------------------------------------------------------------------------------------------------------------------


     STRUCTURE NAME = DSNDSGB_LOCK1     TYPE = LOCK
               # REQ    -------------- REQUESTS -------------   -------------- QUEUED REQUESTS --------------
     SYSTEM    TOTAL            #      % OF  1  -SERV TIME(MIC)-           (ARRIVAL  RATE) --- QUEUE TIME(MIC)--         REQUE
T
     NAME      AVG/SEC         REQ    ALL      AVG   STD_DEV          MIN   AVG   MAX          AVG  STD_DEV       CONTENTIONS

     SC48        250K   SYNC    250K  50.2%   147.0     73.6    HPRIO     0   0.0     0  NO SCH   0.0     0.0  # REQ
   323K
               416.3    ASYNC     0    0.0%     0.0      0.0    LPRIO     0   0.0     0                        # REQ DELAYED  895
                        CHNGD     0    0.0%  INCLUDED IN ASYNC                                                   -CONT        997
                                                                                                                 -FALSE CONT  305
          .  .  .  .
     -------------------------------------------------------------------------------------------------------------------------
     TOTAL       498K   SYNC    498K   100%   147.5     74.1                              NO SCH   0.0     0.0  # REQ         650K
               829.4    ASYNC     0    0.0%     0.0      0.0                                                    # REQ DELAYED 1977
                        CHNGD     0    0.0%                                                                      -CONT       2318


     STRUCTURE NAME = DSNDSGB_SCA       TYPE = LIST
               # REQ    -------------- REQUESTS -------------   -------------- QUEUED REQUESTS --------------
     SYSTEM    TOTAL            #      % OF  1  -SERV TIME(MIC)-           (ARRIVAL  RATE) --- QUEUE TIME(MIC)--
     NAME      AVG/SEC         REQ    ALL      AVG   STD_DEV          MIN   AVG   MAX          AVG  STD_DEV

     SC48         254   SYNC     220  42.7%   415.1     85.4    HPRIO     0   0.1     2  NO SCH  1113   672.6
                0.42    ASYNC      2   0.4%  3957.5   1233.9    LPRIO     0   0.0     0
                        CHNGD     32   6.2%  INCLUDED IN ASYNC  DUMP      0   0.0     0  DUMP     0.0     0.0
        .  .  .  .
     -------------------------------------------------------------------------------------------------------------------------
     TOTAL        515   SYNC     479  93.0%   413.5     87.7                              NO SCH  1067   679.3
                0.86    ASYNC      2   0.4%  3973.3   1244.9                              DUMP     0.0     0.0
                        CHNGD     34   6.6%


    STRUCTURE NAME = DSNDSGB_GBP0      TYPE = CACHE
               # REQ    -------------- REQUESTS -------------   -------------- QUEUED REQUESTS --------------
     SYSTEM    TOTAL            #      % OF  1  -SERV TIME(MIC)-           (ARRIVAL  RATE) --- QUEUE TIME(MIC)--
     NAME      AVG/SEC         REQ    ALL      AVG   STD_DEV          MIN   AVG   MAX          AVG  STD_DEV

     SC48        6400   SYNC    6221  48.7%   336.0     78.1    HPRIO     0   0.0     0  NO SCH 11595    8122
               10.67    ASYNC      0   0.0%  3512.7   1185.7    LPRIO     0   0.3    22
                        CHNGD    179   1.4%  INCLUDED IN ASYNC  DUMP      0   0.0     0  DUMP     0.0     0.0
        .  .  .  .
     -------------------------------------------------------------------------------------------------------------------------
     TOTAL      12776   SYNC     13K  98.4%   335.6     85.7                              NO SCH 10470    8149
               21.29    ASYNC      0   0.0%  3499.6   1177.0                              DUMP     0.0     0.0
                        CHNGD    208   1.6%
```

Figure 76. Coupling Facility Structure Activity Report

Note: **1** Service time (measured in microseconds) is the time needed to access the structure from the MVS image.

The service times required vary from structure to structure. Table 13 on page 195 shows the service times for each coupling facility structure.

Table 13. Synchronous Service Times to Access DB2 Structures	
Structure	**Service Time**
Lock structure	between 120 and 180 µsec
SCA structure	350 and 400 µsec
GBPx structure	250 and 350 µsec

The service times are more configuration dependent than workload dependent. They represent the MVS wait time for the coupling facility. They do not include additional DB2, IRLM, and MVS software time to perform the coupling facility access. Table 13 is more useful for checking out the health of the coupling facility than it is for capacity planning.

If the synchronous service time in your production environment differs from the value given in the table, talk with your MVS system programmer to find out the reason for the difference. Some possible reasons could be:

- Running the coupling facility in LPAR mode

  The CFCC that runs in a coupling facility logical partition uses a polling loop to find new work. The “active wait” polling algorithm used by the CFCC does not fit very well into an environment that wants to share coupling facility resources. Because most of the coupling facility requests are synchronous, the LPAR dispatching time could have a big impact on the service time. *The PR/SM Planning Guide* lists guidelines for setting up a coupling facility.

- Not enough coupling facility links between an MVS image and the coupling facility

  You should have at least two coupling facility links between each MVS image and the coupling facility, not only for availability reasons but also for performance reasons.

- Using the coupling facility links in EMIF mode

  The EMIF mode overhead has a very small impact on your existing I/O environment where the times are measured in milliseconds. Nevertheless, the EMIF mode overhead could be visible in the service time of a coupling facility measured in microseconds.

10.5 Tuning DB2 Locking

It is very important to control locking rates and global locking to achieve best performance. A number of physical database design, application design, and application bind options have a direct impact on global locking. Monitor the global lock contention rate. The objectives are to:

- Keep the total contention (real and false) to less than 2% of the total number of lock requests.

Real contention is a function of application and database design and workload dynamics. If the real contention is a problem, take the necessary actions as per local real contention.

- Keep the amount of false contention less than 50% of the total contention.

  False contention occurs when lock requests against different resources are hashed to the same lock list entry.

To control these objectives you can use either the RMF Coupling Facility Report or the DB2 PM Statistics Report. Execute this DB2 command:

```
-START TRACE(STAT) CLASS(1)
```

to get the input data for the DB2 PM Statistics Report.

Below we explain where you locate the right values in the report and show you how to use the DB2 PM Statistics report to calculate the total amount of global contention and the percentage of false contention.

Refer to the Data Sharing Locking block in the DB2 PM Statistics Report as shown in Figure 77. *IBM DATABASE 2 for MVS/ESA Version 4 Data Sharing: Planning and Administration* presents an example of using the values provided by the RMF Coupling Facility Report.

```
DATA SHARING LOCKING        QUANTITY  /MINUTE  /THREAD  /COMMIT
--------------------------- --------  -------  -------  -------
LOCK REQUESTS (P-LOCKS)     26367.00  2635.57  1464.83     1.46
UNLOCK REQUESTS (P-LOCKS)   26055.00  2604.39  1447.50     1.44
CHANGE REQUESTS (P-LOCKS)     293.00    29.29    16.28     0.02

 1
SYNCH.XES - LOCK REQUESTS     138.4K    13.8K  7687.56     7.66
SYNCH.XES - CHANGE REQUESTS 64013.00  6398.57  3556.28     3.55
SYNCH.XES - UNLOCK REQUESTS 35075.00  3506.00  1948.61     1.94

ASYNCH.XES - RESOURCES          0.00     0.00     0.00     0.00

 2 SUSPENDS - IRLM GLOBAL CONT    625.00    62.47    34.72     0.03
 3 SUSPENDS - XES GLOBAL CONT.      0.00     0.00     0.00     0.00
 4 SUSPENDS - FALSE CONTENTION     71.00     7.10     3.94     0.00
INCOMPATIBLE RETAINED LOCK      0.00     0.00     0.00     0.00

NOTIFY MESSAGES SENT           76.00     7.60     4.22     0.00
NOTIFY MESSAGES RECEIVED     1608.00   160.73    89.33     0.09
P-LOCK/NOTIFY EXITS ENGINES   500.00      N/A      N/A      N/A
P-LCK/NFY EX.ENGINE UNAVAIL     0.00     0.00     0.00     0.00

PSET/PART P-LCK NEGOTIATION     0.00     0.00     0.00     0.00
PAGE P-LOCK NEGOTIATION       428.00    42.78    23.78     0.02
OTHER P-LOCK NEGOTIATION      121.00    12.09     6.72     0.01
P-LOCK CHANGE DURING NEG.     549.00    54.88    30.50     0.03
```

Figure 77. Data Sharing Locking Block of DB2 PM Statistics Report

Explanation of Fields:

1 The total number of lock requests that are propagated to XES synchronously.

2 The number of real contentions, as detected by IRLM

3 The number of real contentions, as detected by XES, that were not IRLM-level contentions. IRLM knows more lock types than XES, so often resolves contentions that XES cannot.

4 The number of false contentions

You can calculate the global contention percentages as:

```
the total number of suspends because of contention ( 2 + 3 + 4 )
        divided by
the total number of request that went to XES (excluding asynchronous requests)
((three fields under 1 ) + 2 + 3 + 4 )
        multiplied by 100.
```

So for our example:

```
(696 / 237488) * 100 = 0.29%
```

giving a contention rate of approximately 0.29 %. If you have such a contention rate, you probably do not have to calculate the false contention percentages. The percentage of global contention should be less than 2%. Here is the calculation to use for false contention percentages:

```
the number of false contentions ( 4 )
        divided by
the total number of contentions ( 2 + 3 + 4 )
        multiplied by 100.
```

So for our example:

```
71 / 696 * 100 = 10.2 %
```

The approximate rate of false contention to real contention is 10.2% (this is a very good figure). The rate of false contention should be less than 50%.

If the number of false contentions is too high, you have two options to minimize them:

- Increase the coupling facility lock structure size.

  The number of false contentions is a function of the size of the hash table, which is one-half the size of the lock structure in the coupling facility. The number of false contentions should track very closely to being inversely proportional to the size of the hash table. Increasing the size of the lock structure should reduce the number of false contentions.

- Ensure that the MAXUSRS value in IRLMPROC is not oversized.

  The size of the MAXUSRS parameter on IRLMPROC and the number of DB2 members started in the group determine the size of each entry and how many entries are available in the hash table. If MAXUSRS is:

  – 7 or less, the initial size of the hash entry is 2 bytes

  – between 7 and 24, the initial size of the hash entry is 4 bytes

  – greater than 23, the initial size of the hash entry is 8 bytes

  If you oversize the MAXUSRS parameter, you unnecessarily decrease the available number of hash entries in the hash table.

You can change the lock structure size dynamically if all members of the group are running with MVS 5.2 or above and the new size is less than the maximum size defined in the CFRM policy. Use the following command:

```
SETXCF START,ALTER,STRNAME=DSNDSGB_LOCK1,SIZE=newsize
```

Note that this command does not affect the size of the hash table. It affects only the MRL. Otherwise you have to increase the storage for the lock structure in the CFRM policy to the next power of 2 and execute the following two commands to start the updated policy and rebuild the structure to the new size:

```
SETXCF START,POLICY
SETXCF START,REBUILD,STRNAME=DSNDSGB_LOCK1
```

You can execute the first command in a running Parallel Sysplex and an active data sharing environment. All changes covering structures to which active connections exist go into a pending state. All others will be active at the next connection time. The second command resolves the pending state of the lock structure and makes the change active.

10.5.1 Global Lock Contention

With data sharing you have to consider false contention, XES-level contention, and P-lock negotiation.

Following are some tasks that you might perform to minimize global lock contention:

- Analyze your workload, and avoid table spaces becoming GBP-dependent.
- For batch applications, submit all updating and read-only jobs from the same DB2 member. You can also try to serialize the use of table spaces by each member.
- For online transactions, avoid different transactions that update the same table space running on different TP monitors connected to different DB2 members in the group.
- Whenever possible use UR isolation level.
- To benefit from lock avoidance, specify ISOLATION(CS) and CURRENTDATA(NO) whenever possible.

- Remember, if all DB2 members are read-only, you still can have data sharing and prevent a table space from becoming GBP-dependent.

10.6 Tuning Group Buffer Pools

DB2 GBPs are used as ″store in″ cache for efficient data buffer coherence control and faster transfer of ownership with ″hot″ data. Pages that are read into the local buffer pool from DASD or a GBP are registered in the GBP for that member. Updated pages are forced out to the GBP usually at commit time, and cross-invalidate signals are sent to the other systems that have interest in that particular page. Remember that cross-invalidation signaling does not cause a processing interrupt on the subject system. The other system now has down-level pages. The next time a particular system touches a page, a check is made to see whether the page is down-level or not. If the page is down-level, an attempt is made to refresh the page from the GBP. Otherwise the page is refreshed from DASD.

It is very important to monitor and tune each GBP to keep the number of accesses to GBP as low as possible. Even though an access to GBP is much faster than an access to DASD, it needs more time compared to finding the page in the local buffer pool. Here are some general rules that make the use of GBP more effective:

1. Separate the DB2 catalog and directory to facilitate GBP structure changes for application and user data
2. Separate DB2 workfiles that cannot be shared across members. You do not need an associated GBP.
3. Separate most GBP-dependent page sets from non-GBP-dependent page sets and index from data.

In the third step, control:

- The GBP structure size
- The ratio between directory entries and data pages in the GBP
- The GBP checkpoint frequency and castout thresholds

To control these in a more general way, use the -DISPLAY GBPOOL command. Execute it at regular intervals, for example, every half hour. The command output gives you information about the GBP that you do not get in any other tool. Figure 78 on page 200, Figure 79 on page 201, and Figure 80 on page 202 show the output of executing the following DB2 command:

```
-DIS GBPOOL(GBP0) GDETAIL(INTERVAL) MDETAIL(INTERVAL)
```

```
-DIS GBPOOL(GBP0) GDETAIL(INTERVAL) MDETAIL(INTERVAL)
DSNB750I *DBB1 DISPLAY FOR GROUP BUFFER POOL GBP0 FOLLOWS
DSNB755I *DBB1 DB2 GROUP BUFFER POOL STATUS
             CONNECTED                                 = YES
             1 CURRENT DIRECTORY TO DATA RATIO            = 2
             PENDING DIRECTORY TO DATA RATIO           = 2
DSNB756I *DBB1    A CLASS CASTOUT THRESHOLD                    = 10%
             B GROUP BUFFER POOL CASTOUT THRESHOLD        = 50%
             C GROUP BUFFER POOL CHECKPOINT INTERVAL      = 10 MINUTES
             RECOVERY STATUS                           = NORMAL
DSNB757I *DBB1 MVS CFRM POLICY STATUS FOR DSNDFOG_GBP0     = NORMAL
             MAX SIZE INDICATED IN POLICY              = 2640 KB
             ALLOCATED                                 = YES
DSNB758I *DBB1      2 ALLOCATED SIZE                          = 2816 KB
               VOLATILITY STATUS                       = NON-VOLATILE
DSNB759I *DBB1      3 NUMBER OF DIRECTORY ENTRIES             = 1247
              4  NUMBER OF DATA PAGES                     = 620
               NUMBER OF CONNECTIONS                   = 2
```

Figure 78. -DISPLAY GROUPBUFFERPOOL Output Status Information

Notes:

1 CURRENT DIRECTORY TO DATA RATIO is the number of directory entries per data page in the GBP.

A CLASS CASTOUT THRESHOLD limits the space that the changed pages from a single class can occupy in the GBP.

B GROUP BUFFER POOL CASTOUT THRESHOLD is the percentage of changed pages that can be kept in the GBP without castout. If the number of changed pages is higher, a GBP castout is triggered.

C GROUP BUFFER POOL CHECKPOINT INTERVAL is used by the GBP structure castout owner to write a GBP checkpoint. This checkpoint triggers writing of changed pages from the coupling facility to DASD.

2 ALLOCATED SIZE is the size allocated for the GBP at the time of first use.

3 NUMBER OF DIRECTORY ENTRIES is given by **2** and **1**.

4 NUMBER OF DATA PAGES is given by **2** and **1**.

```
 DSNB782I *DBB1 INCREMENTAL GROUP DETAIL STATISTICS SINCE 12:20:13 JUN
 15, 1996
 DSNB784I *DBB1 GROUP DETAIL STATISTICS
              READS
                DATA RETURNED                          = 4336
 DSNB785I *DBB1      DATA NOT RETURNED
                  DIRECTORY ENTRY EXISTED              = 0
                  DIRECTORY ENTRY CREATED              = 0
                  DIRECTORY ENTRY NOT CREATED          = 0, 0
 DSNB786I *DBB1   WRITES
                CHANGED PAGES                          = 8290
                CLEAN PAGES                            = 0
                5 FAILED DUE TO LACK OF STORAGE           = 0
              CHANGED PAGES SNAPSHOT VALUE             = 151
 DSNB787I *DBB1   RECLAIMS
                6 FOR DIRECTORY ENTRIES                   = 0
                FOR DATA ENTRIES                       = 114
1             CASTOUTS                                 = 187
 DSNB788I *DBB1   CROSS INVALIDATIONS
                7 DUE TO DIRECTORY RECLAIMS               = 0
                DUE TO WRITES                          = 4283
```

Figure 79. -DISPLAY GROUPBUFFERPOOL Output Group Detail Dynamic Information

Notes:

5 Writes FAILED DUE TO LACK OF STORAGE. DB2 tries to write data pages for that GBP but cannot complete the operation because the GBP does not have enough storage. On write failure, the affected DB2 member waits for only 3 sec after triggering the castout, retries up to four times, and then the pages go to the LPLs requiring recovery action.

6 RECLAIM FOR DIRECTORY ENTRIES. Only those directory entries that have no associated data or have associated data that is clean can be reclaimed.

7 CROSS INVALIDATIONS DUE TO DIRECTORY RECLAIMS. This field shows how many pages are cross-invalidated because of **6**.

```
DSNB771I *DBB1 INCREMENTAL MEMBER DETAIL STATISTICS SINCE 12:20:15
JUN 15, 1996
DSNB773I *DBB1 MEMBER DETAIL STATISTICS
             SYNCHRONOUS READS
               DUE TO BUFFER INVALIDATION
                 DATA RETURNED                       = 2133
                 DATA NOT RETURNED                   = 0, 0
DSNB774I *DBB1     DUE TO DATA PAGE NOT IN BUFFER POOL
                 DATA RETURNED                       = 0
                 DATA NOT RETURNED                   = 0, 0
DSNB775I *DBB1   PREFETCH READS
               DATA NOT RETURNED                     = 29, 0
               REGISTER PAGE LIST NOT AVAILABLE
                 DATA RETURNED                       = 0
DSNB776I *DBB1   SYNCHRONOUS WRITES
              D CHANGED PAGES                           = 4132
               CLEAN PAGES                           = 0
DSNB777I *DBB1   ASYNCHRONOUS WRITES
               CHANGED PAGES                         = 8
               CLEAN PAGES                           = 0
             FAILED WRITES DUE TO LACK OF STORAGE    = 0
DSNB778I *DBB1   CASTOUT THRESHOLDS DETECTED
              E FOR CLASSES                             = 4
              F FOR GROUP BUFFER POOL                   = 0
            G CASTOUTS                                  = 187
DSNB779I *DBB1   ENGINES NOT AVAILABLE
               FOR CASTOUT                           = 0
```

Figure 80. -DISPLAY GROUPBUFFERPOOL Output Member Detail Dynamic Information

Notes:

D SYNCHRONOUS WRITES CHANGED PAGES are changed and committed pages written out to the coupling facility at force-at-commit time from this member.

E CASTOUT THRESHOLDS DETECTED FOR CLASSES is the number of times a class threshold is detected by DB2.

F CASTOUT THRESHOLDS DETECTED FOR GROUP BUFFER POOL is the number of times a GBP threshold is detected.

G CASTOUTS is the total count of pages cast out in the measurement period.

10.6.1 Tune the GBP Structure Size

One of the critical tuning factors in a DB2 data sharing configuration is the size of the GBPs. If the GBP is too small, DB2 data sharing performance degrades as follows:

1. The castout threshold for changed pages is reached more frequently. DB2 castout processing is triggered more often. This increases the processing load on the coupling facility and drives up CPU and I/O resource consumption.

You would see that the CASTOUTS (**G**) count is nearly as large as the SYNCHRONOUS WRITES CHANGED PAGES (**D**) count in the member detail information part of the -DISPLAY GROUPBUFFERPOOL output.

2. The residence time for changed pages once clean may be short. This reduces the GBP hit ratio when a down-level page is detected in the local buffer pool and increases the number of DASD I/Os required for refreshes.

3. In an extreme situation where the GBP is saturated with changed pages, write failures may occur possibly with pages going to LPLs requiring recovery action.

   You would see the WRITES FAILED DUE TO LACK OF STORAGE (**5**) counter as nonzero in the member detail information part of the -DISPLAY GROUPBUFFER POOL output.

10.6.2 Tune the Ratio between Directory Entry and Data Pages in the GBP

DB2 performance improves with more buffers in the local buffer and hiperpool up to a certain point. If the sum of the local buffers across the members in a group associated with a certain GBP is larger than the number of directory elements in the coupling facility, directory entries may be stolen from the clean pages registered. This will result in pages stored locally being cross-invalidated. When page cross-invalidation is detected, the page must be reread from DASD.

The -DISPLAY GROUPBUFFERPOOL output gives you two values: RECLAIM FOR DIRECTORY ENTRIES (**6**) and CROSS INVALIDATIONS DUE TO DIRECTORY RECLAIMS (**7**). Both values should be near zero. Increase the size of the GBP structure and/or increase the ratio of directory elements to data pages.

An easy way of avoiding directory reclaims and resulting expensive cross-invalidation is to take the sum of all buffer pools and hiperpools across the member and the data pages in the GBP as the value for directory entries. This covers the worst case, where every page everywhere has a directory entry.

For example:

```
DBB1 has defined BP0=10000 and HP0=20000
DBB2 has defined BP0=12000 and HP0=25000
GBP0 has 20000 data pages,

10000 + 20000 + 12000 + 25000 + 20000 = 87000 directory entries
```

10.6.3 Tune the GBP Checkpoint Frequency and Castout Thresholds

Make sure that castout processing can keep up with ″force-at-commit″ processing pushing changed data out into the coupling facility. If castout processing is triggered too often, the processing load on the coupling facility and CPU and I/O resource consumption will increase. Therefore, increase the size of the GBP structure and/or adjust GROUP BUFFER POOL CHECKPOINT INTERVAL (**C**), **A** CLASS CASTOUT THRESHOLD (**A**), and GROUP BUFFER POOL CASTOUT THRESHOLD (**B**).

10.7 Transaction Run-Time Criteria: Data Sharing and Non-Data-Sharing

Ideally, a DB2 subsystem using many identical central processors would generate a throughput equal to the number of central processors times the transaction rate achieved by each central processor, with the response time of each transaction remaining unchanged. However, because of increased processing cost per transaction and increased DB2 and IRLM system overhead, throughput may be reduced and the response times may increase.

After you have tuned your data sharing group, you should control the transaction run-time criteria of the most critical transactions, such as CPU and I/O time. To evaluate the correct data sharing overhead of a special workload (special transaction), you need a separate test environment where you run the workload and collect performance data. The data sharing overhead as described in 10.2, "Performance Objectives" on page 192 is very workload dependent and the additional processing cost is collected in different places. If you produce a DB2 PM Statistics Report, you will see an increase in CPU time for the DB2 address spaces (especially for DBM1) and the IRLM address space because the CPU time needed for requests to the coupling facility structures is collected there.

Here is a more practical way of controlling your transactions in your production environment, for example:

1. Collect SMF data of certain comparable time intervals, which are typical for your installation **starting before data sharing is enabled**. Use the following DB2 command:

   ```
   -START TRACE(ACCTG) CLASS(1,2,3,7,8)
   ```

2. Produce DB2 PM Accounting short reports of times before enabling data sharing, one-member data sharing, two-member data sharing, and compare:

   a. The number of SQL statements and the number of getpages per transaction, to ensure that the transaction runs are comparable

   b. The DB2 class 2 elapsed times. The DB2 class 2 elapsed time includes the CPU time that the transaction spent in DB2 and the suspension time inside DB2. If this time is in an acceptable range for you, your data sharing system is running well. If not, you have to analyze the reasons for the difference.

3. Produce DB2 PM Accounting long reports for the selected transactions. You will see a Locking block and a Data Sharing block, as shown in Figure 81 on page 205. Compare the number of IRLM lock requests (**1**) propagated to the lock structure in the coupling facility (**2**) to get an idea of how much data sharing is involved in this transaction run.

```
LOCKING          AVERAGE    TOTAL   DATA SHARING         AVERAGE
-------------   --------  --------  -------------------  --------
TIMEOUTS           0.00         0   LOCK REQ   - PLOCKS      1.00
DEADLOCKS          0.00         0   UNLOCK REQ - PLOCKS      0.00
ESCAL.(SHARED)     0.00         0   CHANGE REQ - PLOCKS      0.00
ESCAL.(EXCLUS)     0.00         0   2 LOCK REQ   - XES          9.74
MAX LOCKS HELD     0.50         1   UNLOCK REQ - XES         1.58
1 LOCK REQUEST       12.74      3350   CHANGE REQ - XES         0.00
UNLOCK REQUEST     2.58       678   SUSPENDS   - IRLM        0.00
QUERY REQUEST      0.00         0   SUSPENDS   - XES         0.00
CHANGE REQUEST     0.00         0   SUSPENDS   - FALSE       0.00
OTHER REQUEST      0.00         0   INCOMPATIBLE LOCKS       0.00
LOCK SUSPENS.      0.01         2   NOTIFY MSGS SENT         0.00
LATCH SUSPENS.     0.04        11
OTHER SUSPENS.     0.00         0
TOTAL SUSPENS.     0.05        13
```

Figure 81. Locking Block and Data Sharing Block of DB2 PM Accounting Report

You will also see a BPx block and the associated Group BPx block for every buffer pool that used the transaction as shown in Figure 82. Compare how many getpage requests (3) cause access to the GBP (six fields behind 4).

```
BP3                     AVERAGE     TOTAL    GROUP BP3           AVERAGE     TOTAL
  ---------------------  --------  --------   4-------------------  --------  --------
  EXPANSIONS                0.00         0    READ(XI)-DATA RETUR      0.03         7
  3 GETPAGES                   3.24       851    READ(XI)-R/W INTER.      0.00         0
  BUFFER UPDATES            0.00         0    READ(XI)-NO R/W INT      0.00         0
  SYNCHRONOUS WRITE         0.00         0    READ(NF)-DATA RETUR      1.34       353
  SYNCHRONOUS READ          0.00         1    READ(NF)-R/W INTER.      0.00         1
  SEQ. PREFETCH REQS        1.68       443    READ(NF)-NO R/W INT      0.00         0
  LIST PREFETCH REQS        0.00         0    CLEAN PAGES WRITTEN      0.00         0
  DYN. PREFETCH REQS        0.00         0    CHANGED PAGES WRTN       0.00         0
  PAGES READ ASYNCHR.      34.55      9086
  HPOOL WRITES              0.00         0
  HPOOL WRITES-FAILED       0.00         0
  PAGES READ ASYN-HPOOL     0.00         0
  HPOOL READS               0.00         0
  HPOOL READS FAILED        0.00         0
```

Figure 82. BPx Block and Group BPx Block of DB2 PM Accounting Report

Finally in the DB2 PM Accounting long report, you will find the Class 3 SUSP. block, as shown in Figure 83 on page 206. If a lot of data sharing is involved in the selected transaction run, global contention (5) could cause a longer repose time. This example shows no additional suspension time attributed to global contention.

```
CLASS 3 SUSP.   AVERAGE TIME  AV.EVENT
--------------  ------------  --------
LOCK/LATCH          0.001133      1.64
SYNCHRON. I/O       0.054263      1.96
OTHER READ I/O      2.199771     14.66
OTHER WRTE I/O      0.000000      0.00
SER.TASK SWTCH      0.000000      0.00
ARC.LOG(QUIES)      0.000000      0.00
ARC.LOG READ        0.000000      0.00
DRAIN LOCK          0.000000      0.00
CLAIM RELEASE       0.000000      0.00
PAGE LATCH          0.000103      0.01
STORED PROC.        0.000000      0.00
NOTIFY MSGS.        0.000000      0.00
 5 GLOBAL CONT.         0.000000      0.00
TOTAL CLASS 3       2.255269     18.27
NOT NULL                 263
```

Figure 83. Class 3 Suspension Time Block of DB2 PM Accounting Report

Refer to *IBM DATABASE 2 for MVS/ESA Version 4 Data Sharing: Planning and Administration* and *DB2 for MVS/ESA Version 4 Data Sharing Performance Topics* for more detailed information.

Chapter 11. Distributed Processing in a Data Sharing Environment

In this chapter we describe two new methods that application requesters can use to connect to a data sharing group: the group-generic method and the member-specific method. We also describe scenarios where a member of a data sharing group is an application requester. We provide some sample communications database (CDB) configurations to help you when setting up your environment.

11.1 Overview of Changes

VTAM 4.2 has introduced support for generic LU names, enabling multiple real LU names to map to one common generic LU name.

Generic LU names are used in a data sharing environment. Each member in a group has its own real LU name, but there can also be one generic LU name to which all real LU names map. The generic name is stored in the BSDSs of the members in a group, and it must be the same for all members in one group. If you want to use a generic name for your data sharing group, you must include the generic name in the DB2 GENERIC NAME field on installation panel DSNTIPR.

The members in a data sharing group share a common CDB, and all members in the group are identified by one common location name. The location name is also stored in the BSDSs of the members in the group and, like the generic LU name, must be the same for all members in one group.

Now let us look at how the generic LU name is used in a data sharing environment and also at how another new connectivity option, member-specific access, can be used.

11.2 Connectivity Alternatives

Distributed access to a data sharing group can be set up in three different ways. Requesters can connect by:

- Connecting to a specific member in the data sharing group by using a member′s specific LU name.

  There is nothing new or different about this method. It is set up in the same way as connecting to a non-data-sharing subsystem.

- Using the group-generic method.

  The requester can specify a generic LU name that represents all of the members in a data sharing group. VTAM or a user exit then chooses one of the members in the group and establishes a session with that member.

For any application requester to connect to an application server data sharing group, using the generic LU name of the application server data sharing group, it is enough that the application requester specifies the generic LU name for that server. When DB2 MVS is the application requester, this specification is done in the application requester DB2′s CDB. When the application requester uses the application server data sharing group′s generic LU name, the method being used is *group-generic*.

Connections to a group are spread out among the members in the group. Workload balancing, however, is at the session level, that is, when the requester establishes a session with the server. These sessions are never redistributed to other servers.

- Using the member-specific method.

  The requester (currently DB2 MVS V4 is the only requester that can connect using member-specific access) invokes the DB2 Sysplex transaction program (DB2 STP) on the server. The DB2 STP interacts with the MVS WLM and calculates a weighting for each member that is also passed to the requester. The weighting indicates the percentage of work (conversations) the requester should pass to each member. The weighting is based on available capacity. The DB2 STP returns a list of available members to the requester.

  – Workload is balanced on the basis of available capacity. The requester can connect to multiple members in the group. The DB2 STP can be invoked by the requester as required, thereby ensuring that member capacity can be continually reevaluated.

In this chapter, we discuss the benefits of each connection method and provide you with sample CDB configurations for each method. But first, some high-level definitions can aid understanding:

- LU: A source of requests entering the network and a receptor of replies from the network. A particular DB2 is an LU, for example.
- Session: A logical connection between two LUs. A single session can support multiple conversations but only one at a time:

  – A session is established when the first connect is made from the requester to the server.

  – A session is terminated when the DDF is stopped.

- Conversation: A dialog that uses a session to transfer information between transaction programs, such as DB2 to DB2:

  – A conversation is initiated when the requester connects to the server.

  – A conversation is terminated when the requester disconnects from the server.

Table 14 compares the connectivity options when a data sharing group is the application server.

Table 14 (Page 1 of 2). Connectivity Options: Data Sharing Group As Application Server			
	Specific LU	**Group-Generic**	**Member-Specific**
How many links to the data sharing group?	One link to a specific member in the data sharing group	One link to a member in the data sharing group. Member chosen by VTAM or user exit. Requesters cannot link to multiple members concurrently.	Possibility of many links to different members in the data sharing group concurrently.
Workload balancing?	No workload balancing	Workload balancing done by VTAM or user exit. In the case of VTAM the type of algorithm used for balancing depends on release of VTAM. Balancing is done at a session level—that is, when an LU establishes a session—and is not reevaluated unless the session is terminated and reinitiated. Workload balancing decided at the server.	Workload balancing based on available capacity. Information about members in the group passed to requester from DB2 Sysplex transaction program (DB2 STP). DB2 STP interacts with MVS WLM to get capacity information. Distribution of work is done at the conversation level and can be continually reevaluated by invoking the DB2 STP. Workload balancing decided at the requester.
Reconnection after a communication failure—requesters active at time of failure.	Requesters must connect to the specific member in the group.	Requesters that had a session with two-phase commit support enabled must connect to the specific member in the group with which a session had been established before the failure (irrespective of the request type—select or update).	If resynchronization must be performed, the request is routed to the DB2 member with which the requester had a session at the time of the failure.

Table 14 (Page 2 of 2). Connectivity Options: Data Sharing Group As Application Server			
	Specific LU	**Group-Generic**	**Member-Specific**
Special software requirements?	None	None	Requester and server must implement support for the DB2 SPT

11.3 Communication Database Configuration

In this section we present some test cases and for each the minimum configuration of the CDB in DB2 for MVS/ESA.

Table 15 summarizes all cases and the method the application requester uses to connect to the application server.

Table 15. Test Cases

Case	Application Requester	Application Server	Method
1	DB2 for MVS/ESA (V2.3, V3, or V4) or other DBMS	Data sharing group	Group-generic
2	DB2 for MVS/ESA (V2.3, V3, or V4) or other DBMS	Data sharing group	Using specific LU
3	DB2 for MVS/ESA V4 non-data sharing	Data sharing group	Member-specific
4	Data sharing group	DB2 for MVS/ESA (V2.3, V3)	Using specific LU
5	Data sharing group	DB2 for MVS/ESA V4 or other DBMS	Using specific LU
6	Data sharing group	Data sharing group	Group-generic
7	Data sharing group	Data sharing group	Member-specific

Note: The member-specific method is supported only when the requester and server both implement support for the DB2 STP. DB2 MVS V4 is currently the only product that has implemented support for DB2 STP and therefore the number of cases we show using the member-specific method is limited.

Before discussing these cases in more detail, let us review some basic concepts about the CDB. DB2 uses the CDB to store information about how to communicate with other DB2s or with remote locations that support DRDA. The CDB is created during installation and is a user-maintained collection of tables. The DDF uses the CDB to map DB2 location names to

VTAM LU names. The CDB also handles security translation requirements and communication requirements.

If you plan to use DB2 only as a server, you do not have to populate the CDB; you can use the default blank row to allow any requester access. For security reasons, if you would prefer to control which requesters are allowed access, you have to delete the default blank row from the SYSIBM.SYSLUNAMES table of the server and insert the LU names of the requesters you want to have access to the server.

If you intend to request data, however, you have to insert one row for each remote system into each of two tables: SYSIBM.SYSLUNAMES and SYSIBM.SYSLOCATIONS. There are some other tables in the CDB that you can use to associate some applications with a specific VTAM mode or for controlling requests from remote applications. Because our discussion is limited to what is required to connect to DB2, we cover the following tables only, which are the minimum requirements to connect to a remote DB2.

- SYSIBM.SYSLUNAMES
- SYSIBM.SYSLOCATIONS
- SYSIBM.SYSLULIST (new in DB2 V4)

11.3.1 SYSIBM.SYSLUNAMES

SYSLUNAMES defines the security and mode requirements for conversations with other systems. Decisions about how to populate this table depend on how you intend to use DB2:

- If you use DB2 system only as a server, it can use a default row, in which the LUNAME column is blank. DB2 uses the values in that row as defaults for requesting LUs that are not explicitly defined in SYSLUNAMES.

  The DSNTIJSG installation job creates the default row in SYSLUNAMES. You can change the values for the default row any time before starting the DDF. If you created the CDB using your own SQL, you might not have the default row, in which case you can add the row, changing the defaults to the values you want.

- If this DB2 requests data from other systems, you have to provide LU names for those systems.

11.3.1.1 SYSLUNAMES Columns: SYLUNAMES has the following columns:

LUNAME CHAR(8) The LU name of the remote system. The default of eight blanks indicates that this row is used for serving the requests of any system that is not specifically listed in the SYSLUNAMES table. However, you must provide LU names for any remote system that uses values different from the defaults.

This column is the primary key. All other tables in the CDB have foreign keys that refer to this column.

SYSMODENAME CHAR(8) The mode used to establish system-to-system conversations for system-directed access (private protocol). This column is ignored for application-directed access (DRDA) conversations.

USERSECURITY CHAR(1) Defines the security that this subsystem requires from the remote system

ENCRYPTPSWDS CHAR(1) A *Y* in this column indicates the use of password encryption.

MODESELECT CHAR(1) Determines whether to use the default mode or to choose a mode from the SYSMODESELECT table

USERNAMES CHAR(1) This column is used for:

- Outbound requests, to determine the security level of those requests (whether a password is sent with the request)
- Inbound and outbound requests, to control authorization ID translation.

GENERIC CHAR(1) (new with DB2 V4) This column applies only to an application requester. A *Y* in this column indicates that the generic LU name for this data sharing group application requester, is to be used for change number of sessions (CNOS) processing. You can also specify this column for a DB2 V4 not joined to a data sharing group, but probably it will be in the time you are preparing to move to a data sharing group

11.3.1.2 How the GENERIC Column in SYSLUNAMES Works: For an application requester member of a data sharing group to use its group′s generic LU when connecting to any application server, it is enough to set the GENERIC column to *Y* in the application requester′s CDB definitions for the server. This does not define which of the three connection options (group-generic, member-specific, or real LU) will be used when connecting to the application server.

Use the GENERIC column of the SYSIBM.SYSLUNAMES table to specify whether the VTAM generic LU name or the real DB2 LU name is used by an application requester DB2 to identify itself to a given remote server. This column is used only when DB2 is initiating contact with a given partner LU. The generic name does not have to be defined to VTAM as an APPL.

The GENERIC column is used only when DB2 is the application requester; it is ignored when DB2 is an application server. Let us look at an example when DB2 is the application requester and GENERIC=Y has been specified next to the LU name of a particular server. When the DB2 application requester accesses this server, it tells VTAM to use the generic LU name for this data sharing group application requester when connecting to the server.

The GENERIC column is ignored if the DB2 GENERIC NAME field on installation panel DSNTIPR is blank.

There are some restrictions when using GENERIC=Y:

- In a data sharing environment, where multiple LU names can be represented by one generic LU, only one member of the group can connect to a given remote LU with the GENERIC name. VTAM automatically switches to the real LU name if it believes one of the other members has already used the generic name to link to the same LU.
- Even if the generic LU name, at the requester, is passed to the server, the real LU name is checked by the server DB2 in SYSIBM.SYSLUNAMES.

Because of these restrictions we do not see any benefit in specifying GENERIC=Y.

11.3.1.3 SYSIBM.SYSLOCATIONS: The SYSLOCATIONS table has the following purpose: The location name specified in the connect statement must match a value in the LOCATION column, which maps the location name to the VTAM LU name and, if necessary, the transaction program name (TPN).

SYSLOCATIONS has the following columns:

LOCATION CHAR(16) The unique network location name, or RDB_NAME, assigned to a system, remote or local. You must provide location names for any systems from which you request data. This column is the primary key for this table.

LOCTYPE CHAR(1) This column is reserved and must be blank.

LINKNAME CHAR(8) The LU name for this location. This column is a foreign key, referring to the LUNAME column of SYSLUNAMES, with a delete rule of cascade. Because the SYSLOCATIONS table is used for outbound requests, you must provide an LU name or your requests fail. Do not enter blanks in this column.

LINKATTR VARCHAR(64) This column is used to enter a TPN for non-DB2 MVS/ESA systems. Use this column only if you are sending requests to systems using non-default TPNs.

11.3.1.4 SYSIBM.SYSLULIST: This new DB2 MVS V4 table is used on the application requester to support member-specific access to a DB2 data sharing group. It allows you to use a symbolic name on the application requester that represents a list of the application server data sharing group members′ real LU names.

Remember that member-specific access requires that both the server and requester implement support for the DB2 STP. DB2 MVS V4 is currently the only product that has implemented support for DB2 STP.

The SYSLULIST table is used by a DB2 requester as follows:

- When access is requested to a particular remote location, the SYSIBM.SYSLOCATIONS table on the application requester is searched to find a matching row for the remote location specified. If member-specific access is being used, the LINKNAME corresponding to the remote location name in SYSIBM.SYSLOCATIONS will be a symbolic name.

- For member-specific access, there will be one or more corresponding rows in the SYSIBM.SYSLULIST table for this symbolic name. These rows will contain the LUNAME values of the real LU names associated with the location. When the location is a data sharing group, the LU names are the real LU names of the members in the group.
- There will be a row in SYSIBM.SYSLUNAMES for the symbolic name, and the values in this row will apply to all of the real LU names.
- If no matching row for the LINKNAME exists in SYSIBM.SYSLULIST, member-specific access is not being used and the SYSIBM.SYSLUNAMES table is read to find an LU name associated with the LINKNAME (as in previous releases of DB2).

  SYSLULIST has the following columns:

  LINKNAME CHAR(8) This column contains a symbolic name. There is a corresponding row in SYSIBM.SYSLOCATIONS for this symbolic name. There is also a row in the SYSIBM.SYSLUNAMES table where the LUNAME contains this symbolic name. The values in this row on SYSIBM.SYSLUNAMES will apply to all of the real LU names associated with the symbolic name.

  LUNAME CHAR(8) The real VTAM LU name of a remote server. This LUNAME must not exist in the LUNAME column of SYSIBM.SYSLUNAMES.

11.3.2 Updating CDB Values

Any table in the CDB can be updated while the DDF is active. Changes to SYSLUNAMES and SYSLOCATIONS take effect as follows:

- If the DDF has not yet tried to communicate with a particular remote location, rows added to SYSLUNAMES and SYSLOCATIONS take effect when the DDF attempts to communicate with that location.
- If the DDF has already attempted communication with a particular location, rows added to SYSLUNAMES and SYSLOCATIONS take effect the next time the DDF is started.

These updates do not affect existing conversations; that is, existing conversations continue to operate as the table was before the update.

11.3.3 Resetting Generic LU Information

If you are using a generic LU name to connect to a member of the data sharing group with two-phase commit support enabled on both application requester and application server, VTAM records information in the coupling facility about which member of the DB2 group was involved in the communication. This information is required to guarantee that future VTAM sessions are always directed to the same DB2 group member, making it possible to provide access to the correct DB2 subsystem log for resolution of indoubt threads.

The type of access you are using—distributed unit of work or remote unit of work— and the type of statements you are issuing (select, update) at the server are not considered by VTAM when establishing this relationship between the requester and server. In other words, this relationship will be recorded (and preserved in the event of an abend) even if the type of access you are using does not require access to the DB2 subsystem log in the event of an abend and subsequent reconnection.

To enable two-phase commit support in DB2 for MVS/ESA, specify SYNCLVL=SYNCPOINT in VTAM for the DB2 LU APPL definition.

There might be times when you have to break the affinity between the group member and the other system. You would have to do this, for example, if you want to start using the member-specific method, or if you want to remove a member from the data sharing group. To break the affinity, use the RESET GENERICLU command. The command must be issued from the DB2 member with the VTAM affinity to the particular partner LU whose information you are purging. Therefore if you have connected to a data sharing group application server using the data sharing group application server generic LU name, and you are not using the GENERIC=Y column in SYSIBM.SYSLUNAMES in the application requester, the RESET GENERICLU command must be specified from the application server specifying the real LU name of the requester. For example, if you have an application requester requesting data from a data sharing group using the generic LU name DSGA, it will be connected to a member of the data sharing group, say, DBA1. If then you want to purge this affinity, you have to issue the following command:

```
-DBA1 RESET GENERICLU(netid.luname)
```

where

- netid is the net ID of the application requester.
- luname is the real LU name of the application requester.

If the application requester is passing its generic LU name (using the GENERIC=Y column in SYSIBM.SYSLUNAMES in the application requester), you have to specify the generic LU name instead of the real LU name.

Let us go through an example. Data sharing group A is the application requester and has GENERIC=Y specified in its CDB. Data sharing group B is the application server. Data sharing group A uses the group-generic access method when connecting to data sharing group B. To remove all affinities between these two members, you have to issue the following commands:

-
  ```
  -DBA1 RESET GENERICLU(netid.luname)
  ```

  where

  - netid is the net ID of the data sharing group B (the application requester)
  - luname is the generic LU name of data sharing group B (the application requester)

•

```
-DBB1 RESET GENERICLU(netid.luname)
```

where

- netid is the net ID of the data sharing group A (the application server).
- luname is the generic LU name of data sharing group A (the application server).

Great care should be taken when using the RESET GENERIC LU command, because it can potentially cause the specified partner LUs to connect to different DB2 subsystems on later sessions. This can cause operational problems if indoubt threads exist at a partner LU when this command is issued. The command must be issued from the DB2 that has the affinity to the particular LU whose information you are purging.

The following conditions must be satisfied for the RESET GENERICLU command to be successful:

- The DDF must be started.
- No VTAM sessions can be active to the partner LU specified on the command.
- DB2 must not have any indoubt thread resolution information associated with the specified partner LU.

In summary, Table 16 on page 217 shows where to issue the RESET GENERICLU command and which LU name to use when issuing the command. Assume that:

- Data sharing group A has a generic LU name of GENA and a real LU name of REALA
- Data sharing group B has a generic LU name of GENB and a real LU name of REALB

Note

When you connect to a data sharing group, you are actually connected to a member of that data sharing group (each DB2 member has its own real LU name, but the data sharing group has only one DB2 location name, which in turn can be associated with one VTAM generic LU name).

When you are connected to a member of the data sharing group, VTAM uses either the real LU name for that member or the generic LU name for the data sharing group.

In the context of this discussion, the real LU name implies the LU name of the DB2 member to which you are connected.

- Data sharing group A is the application requester
- Data sharing group B is the application server

Table 16. Using the RESET GENERICLU Command

LU name	**Generic**	**Issue RESET on application requester for**	**Issue RESET on application server for**
DSGB	' '		REALA
DSGB	'Y'	GENB(*)	GENA(*)
REALB	' '		
REALB	'Y'	REALB	

Note: *For this entry generally you should issue the RESET using the generic LU name. However, VTAM could have switched from using the generic LU name to using the real LU name. Therefore if, when you issue the RESET and are told that the connection cannot be found, reissue the command, using the real LU name.

11.3.4 Using the Generic LU Name

If a DB2 MVS application requester is using a generic LU name to connect to a data sharing group, all threads for a single LU will be directed to one of the members in the data sharing group. It will be the first member connected to after DDF startup. After all sessions are terminated, VTAM can choose a different DB2 member in the group as a server for future communications, unless the sessions have two-phase commit support. Where sessions have two-phase commit support, the same DB2 server must be used, and so VTAM records this information in the coupling facility. To clean up the sessions you can stop the DDF or deactivate the LU, using VTAM commands. Note that if the sessions used two-phase commit support, the VTAM affinity between a member LU name and generic LU name persists even after the DDF has been stopped. Therefore, if you want to connect to another member, you have to clean up the generic LU information from the VTAM structure in the coupling facility, which you can do by issuing the RESET GENERICLU DB2 command.

This configuration offers some flexibility because the member you will be connected to is not fixed. If you have to bring down a system and want to route the workload to another member, you can do it easily if your sessions did not use a protocol with two-phase commit support. You have to stop the DDF, and then the next requests will go to another member of the data sharing group. If the sessions used two-phase commit, you have to do the following:

- STOP DDF to clean up the sessions
- START DDF
- RESET GENERICLU to reset the affinity
- STOP DDF

Now the next sessions will be directed to another member.

If there is a communication failure, for example, the DB2 member comes down, if the session did not use a protocol with two-phase commit support, the next request from the application requester will be directed to any DB2 LU in the group. If the sessions used a protocol with two-phase commit support, VTAM must always connect the requester to the DB2 LU chosen during the first session established by VTAM. If that DB2 LU is unavailable, a communication error is returned with VTAM sense code 08570003.

11.4 Sample Cases

For these cases we used two data sharing groups, one DB2 V3 and one DB2 V4. Here are the DB2 LOCATION name, LU name, and GENERIC name definitions:

- Data sharing group A
  - Member DBA1
    - LOCATION=DSGA
    - LU name=LUDBA1
    - GENERIC=LUDSGA
  - Member DBA2
    - LOCATION=DSGA
    - LU name=LUDBA2
    - GENERIC=LUDSGA
- Data sharing group B
  - Member DBB1
    - LOCATION=DSGB
    - LU name=LUDBB1

- GENERIC=LUDSGB

– Member DBB2

  - LOCATION=DSGB
  - LU name=LUDBB2
  - GENERIC=LUDSGB

- V4 system

  – LOCATION=DB41

  – LU name=LUDB41

- V3 system

  – LOCATION=DB31

  – LU name=LUDB31

The above definitions are contained in the BSDS for each DB2 subsystem.

We want to remind you that the location name of each member in a data sharing group must be the same.

For the cases described here, we always used a blank in the GENERIC column of SYSIBM.SYSLUNAMES, on both the application requester and the application server.

11.4.1 Case 1

The application requester can be a DB2 for MVS/ESA (V2.3, V3, or V4) or any other DBMS supporting DRDA. The application server is data sharing group A, and the method used to connect is group-generic.

The configuration of the CDB only applies to DB2 for MVS/ESA; for other DBMSs refer to the appropriate documentation. Table 17, Table 18 on page 220, Table 19 on page 220, and Table 20 on page 220 show the appropriate entries in SYSIBM.SYSLUNAMES and SYSIBM.SYSLOCATIONS for both the application requester and application server.

Table 17. SYSIBM.SYSLUNAMES Application Requester: Case 1

LUNAME	GENERIC
LUDSGA	.....
.....	.....
.....	.....

Table 18. SYSIBM.SYSLOCATIONS Application Requester: Case 1	
LOCATION	**LINKNAME**
DSGA	LUDSGA
.....	.....
.....	.....

Table 19. SYSIBM.SYSLUNAMES Application Server: Case 1	
LUNAME	**GENERIC**
.....	.....
.....	.....
.....	.....
Note: The blank row in this table allows any requester to access this server.	

Table 20. SYSIBM.SYSLOCATIONS Application Server: Case 1	
LOCATION	**LINKNAME**
......	......
.....	.....
.....	.....

As the AS is a data sharing group connected to by using its generic LU name, all of the considerations in 11.3.4, "Using the Generic LU Name" on page 217 apply. If you are using workstations as application requesters, you could have two different scenarios (see Figure 84 on page 221). In the first scenario, you have all workstations using a Distributed Database Connection Services (DDCS) machine as a gateway to the DB2 data sharing group. You have to consider that all the workstations will be connected using the DDCS LU name. Thus all of the connections will be directed to the same DB2 member.

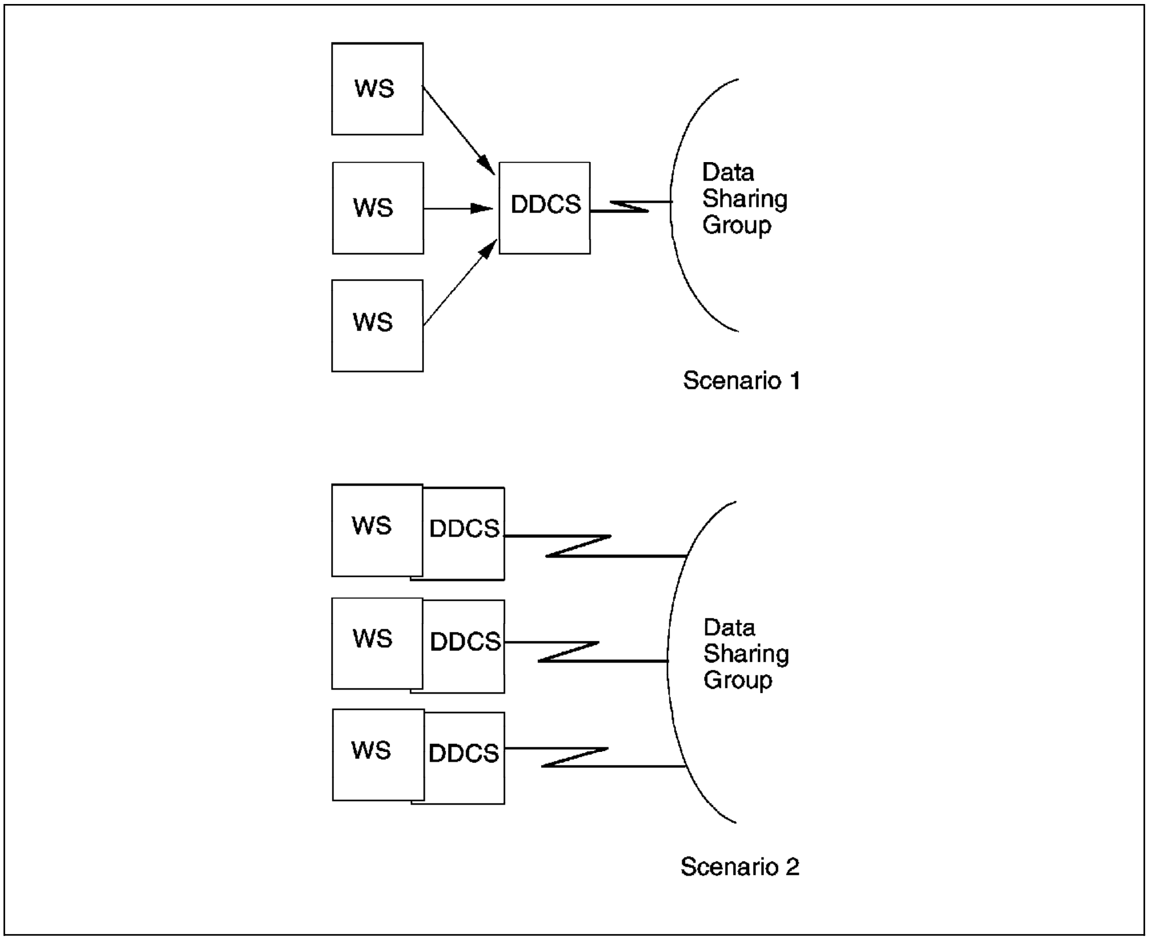


Figure 84. Workstation Application Requester Configuration

In the second scenario, each workstation has DDCS capability. Therefore each workstation will be connected using its own LU name. This configuration provides improved workload balancing, as the connections from the workstations are balanced out among the members in the application server data sharing group, as opposed to scenario 1 where all the workstations connect to one member in the data sharing group.

The two configurations shown in Figure 84 are extremes; generally a configuration such as that shown in Figure 85 on page 222 is what we would recommend. Groups of workstations share DDCS gateways. This configuration allows for workload balancing but also limits the number of DDCS gateways required.

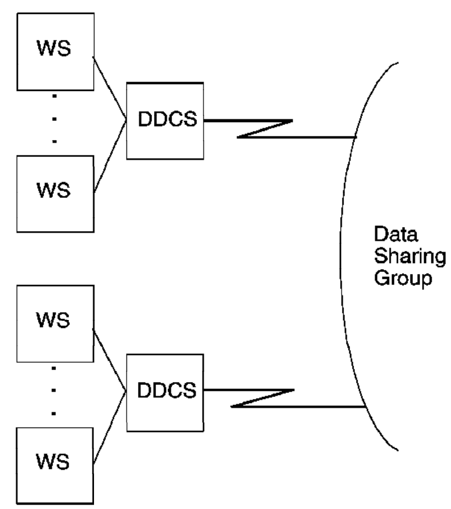


Figure 85. Suggested Workstation Application Requester Configuration

11.4.2 Case 2

The application requester can be a DB2 for MVS/ESA (V2.3, V3, or V4) or any other DBMS supporting DRDA. The application server is data sharing group A, and the method of connecting is by using the specific LU.

The configuration of the CDB applies only to DB2 for MVS/ESA, for other DBMSs refer to the appropriate documentation. Table 21, Table 22, Table 23 on page 223, and Table 24 on page 223 show the appropriate entries in SYSIBM.SYSLUNAMES and SYSIBM.SYSLOCATIONS for both the application requester and the application server.

Table 21. SYSIBM.SYSLUNAMES Application Requester: Case 2

LUNAME	**GENERIC**
LUDBA1 or LUDBA2	......
......	.....
......	.....

Table 22 (Page 1 of 2). SYSIBM.SYSLOCATIONS Application Requester: Case 2

LOCATION	**LINKNAME**
DSGA	LUDBA1 or LUDBA2
.....	.....

Table 22 (Page 2 of 2). SYSIBM.SYSLOCATIONS Application Requester: Case 2	
LOCATION	**LINKNAME**
.....	.....

Table 23. SYSIBM.SYSLUNAMES Application Server: Case 2	
LUNAME	**GENERIC**
.....	.....
.....	.....
.....	.....
Note: The blank row in this table allows any requester to access this server.	

Table 24. SYSIBM.SYSLOCATIONS Application Server: Case 2	
LOCATION	**LINKNAME**
......	.....
.....	.....
.....	.....

In this configuration each request must be directed to one of the available DB2 LU names to gain access to the DB2 server. If the requester is DB2 for MVS/ESA, all sessions must be directed to the same member of the DB2 group, as the LOCATION specified in the SYSIBM.SYSLOCATIONS table must be unique. However, other types of requesters, for example, an OS/2 workstation, may allow for multiple concurrent sessions with multiple members in the group. If the chosen LU name is not available, a communication failure is returned.

This configuration does not offer the same flexibility as Case 1, because you cannot switch to another member of the data sharing group without changing the CDB.

11.4.3 Case 3

In this case the application requester is a non-data-sharing DB2 for MVS/ESA V4. The application server is data sharing group A, and the method used to connect is member-specific.

Table 25 on page 224, Table 26 on page 224, Table 27 on page 224, Table 28 on page 224, and Table 29 on page 224 show the appropriate entries in SYSIBM.SYSLUNAMES and SYSIBM.SYSLOCATIONS for both the application requester and application server, and in SYSIBM.SYSLULIST for the application requester.

Table 25. SYSIBM.SYSLUNAMES Application Requester: Case 3	
LUNAME	**GENERIC**
LIST	.....
.....	.....
.....	.....

Table 26. SYSIBM.SYSLOCATIONS Application Requester: Case 3	
LOCATION	**LINKNAME**
DSGA	LIST
.....	.....
.....	.....

Table 27. SYSIBM.SYSLULIST Application Requester: Case 3	
LINKNAME	**LUNAME**
LIST	LUDBA1
LIST	LUDBA2
.....	.....

Table 28. SYSIBM.SYSLUNAMES Application Server: Case 3	
LUNAME	**GENERIC**
.....	.....
.....	.....
.....	.....
Note: The blank row in this table allows any requester to access this server.	

Table 29. SYSIBM.SYSLOCATIONS Application Server: Case 3	
LOCATION	**LINKNAME**
......	.....
.....	.....
.....	.....

This configuration offers more flexibility and is more efficient in terms of workload balancing. Using the member-specific method, each requester can establish concurrent sessions with multiple DB2 subsystems in the data sharing group. The threads need not be directed to a single member of the data sharing group. After a communication failure, the requester can connect to any LU in the group. If resynchronization must be performed because of a previous communication or system failure, the resynchronization request is routed to the DB2 LU involved in the failure. This activity is performed asynchronously and does not prevent new sessions from being established with other members of the DB2 group.

11.4.4 Case 4

The application requester is data sharing group B. The application server can be a DB2 for MVS/ESA V2.3 or V3. The method of connecting is by using a specific LU.

The configuration of the CDB applies only to DB2 for MVS/ESA; for other DBMS refer to the appropriate documentation. Table 30, Table 31, Table 32, and Table 33 show the appropriate entries in SYSIBM.SYSLUNAMES and SYSIBM.SYSLOCATIONS for both the application requester and application server.

Table 30. SYSIBM.SYSLUNAMES Application Requester: Case 4

LUNAME	**GENERIC**
LUDB31	......
......	.....
......	.....

Table 31. SYSIBM.SYSLOCATIONS Application Requester: Case 4

LOCATION	**LINKNAME**
DB31	LUDB31
.....	.....
.....	.....

Table 32. SYSIBM.SYSLUNAMES Application Server: Case 4

LUNAME	**GENERIC**
.....	.....
.....	.....
.....	.....
Note: The blank row in this table allows any requester to access this server.	

Table 33 (Page 1 of 2). SYSIBM.SYSLOCATIONS Application Server: Case 4

LOCATION	**LINKNAME**
......	.....

Table 33 (Page 2 of 2). SYSIBM.SYSLOCATIONS Application Server: Case 4	
LOCATION	**LINKNAME**
.....	.....
.....	.....

As the server is a DB2 for MVS/ESA V2.3 or V3, only one LU in the DB2 group is allowed to send SQL to the DB2 server. If two or more DB2 subsystems in the data sharing group require access to the remote data, the installation must have some mechanism for ensuring that all applications requiring access to the server run on a single member of the DB2 group.

11.4.5 Case 5

The application requester is data sharing group B, the application server is a DB2 for MVS/EA V4 or any other DBMS supporting DRDA and able to act as an application server. The method of connecting is by using the specific LU.

The configuration of the CDB applies only to DB2 for MVS/ESA; for other DBMSs refer to the appropriate documentation. Table 34, Table 35, Table 36, and Table 37 on page 227 show the appropriate entries in SYSIBM.SYSLUNAMES and SYSIBM.SYSLOCATIONS for both the application requester and application server.

Table 34. SYSIBM.SYSLUNAMES Application Requester: Case 5	
LUNAME	**GENERIC**
LUDB41	......
......	.....
......	.....

Table 35. SYSIBM.SYSLOCATIONS Application Requester: Case 5	
LOCATION	**LINKNAME**
DB41	LUDB41
.....	.....
.....	.....

Table 36 (Page 1 of 2). SYSIBM.SYSLUNAMES Application Server: Case 5	
LUNAME	**GENERIC**
.....	.....
.....	.....

Table 36 (Page 2 of 2). SYSIBM.SYSLUNAMES Application Server: Case 5	
LUNAME	**GENERIC**
.....	.....
Note: The blank row in this table allows any requester to access this server.	

Table 37. SYSIBM.SYSLOCATIONS Application Server: Case 5	
LOCATION	**LINKNAME**
......	.....
.....	.....
.....	.....

In this case, as the application server is a DB2 for MVS/ESA V4, or any other DBMS supporting DRDA, all members of the data sharing group can have concurrent connections to the application server.

11.4.6 Case 6

The application requester is data sharing group B. The application server is data sharing group A, and the method used to connect is group-generic. Table 38, Table 39, Table 40 on page 228, and Table 41 on page 228 show the appropriate entries in SYSIBM.SYSLUNAMES and SYSIBM.SYSLOCATIONS for both the application requester and application server.

Table 38. SYSIBM.SYSLUNAMES Application Requester: Case 6	
LUNAME	**GENERIC**
LUDSGA	.....
.....	.....
.....	.....

Table 39. SYSIBM.SYSLOCATIONS Application Requester: Case 6	
LOCATION	**LINKNAME**
DSGA	LUDSGA
.....	.....
.....	.....

Table 40. SYSIBM.SYSLUNAMES Application Server: Case 6	
LUNAME	**GENERIC**
.....	.....
.....	.....
.....	.....
Note: The blank row in this table allows any requester to access this server.	

Table 41. SYSIBM.SYSLOCATIONS Application Server: case 6	
LOCATION	**LINKNAME**
......	.....
.....	.....
.....	.....

Case 6 is similar to case 1; however, the application requester is a data sharing group that consists of many subsystems, unlike case 1 where the application requester is only one subsystem. A member in the requester group can be linked to only one member in the server group at any one time. Members in the server group can be connected to by more than one member in the requester group at the same time. If you are using a protocol that has two-phase commit support, the VTAM affinity will persist even if all sessions are terminated.

As the application server is a data sharing group connected by using its generic LU name, all of the considerations in 11.3.4, "Using the Generic LU Name" on page 217 apply

11.4.7 Case 7

The application requester is data sharing group B. The application server is data sharing group A and the method used to connect is member-specific. Table 42, Table 43 on page 229, Table 44 on page 229, Table 45 on page 229, and Table 46 on page 229 show the appropriate entries in SYSIBM.SYSLUNAMES and SYSIBM.SYSLOCATIONS for both the application requester and application server, and also SYSIBM.SYSLULIST for the application requester.

Table 42. SYSIBM.SYSLUNAMES Application Requester: Case 7	
LUNAME	**GENERIC**
LIST	.....
.....	.....
.....	.....

Table 43. SYSIBM.SYSLOCATIONS Application Requester: Case 7

LOCATION	**LINKNAME**
DSGA	LIST
.....	.....
.....	.....

Table 44. SYSIBM.SYSLULIST Application Requester: Case 7

LINKNAME	**LUNAME**
LIST	LUDBA1
LIST	LUDBA2
.....	.....

Table 45. SYSIBM.SYSLUNAMES Application Server: Case 7

LUNAME	**GENERIC**
.....	.....
.....	.....
.....	.....
Note: The blank row in this table allows any requester to access this server.	

Table 46. SYSIBM.SYSLOCATIONS Application Server: Case 7

LOCATION	**LINKNAME**
......	.....
.....	.....
.....	.....

Case 7 is similar to case 3. The difference is that in case 3 the application requester is only one subsystem whereas in case 7 the application requester is a data sharing group consisting of multiple subsystems. As the data sharing group application server is connected by using the member-specific method, each member of the application requester data sharing group can establish concurrent sessions with one or more members in the application server data sharing group. Even if one of the members in the server group fails, the application requester can still connect to any of the other members in the application server data sharing group. If resynchronization is required because of a communication or system failure, the resynchronization request is routed to the DB2 member involved in the failure.

11.5 Replacing DDF Connections by Data Sharing

Before data sharing, the DDF was used to enable access to DB2 data on other MVS systems. Figure 86 on page 230 shows a program (DB2B) running on MVS2 and using the DDF to gain access to DB2A′s data on MVS1.

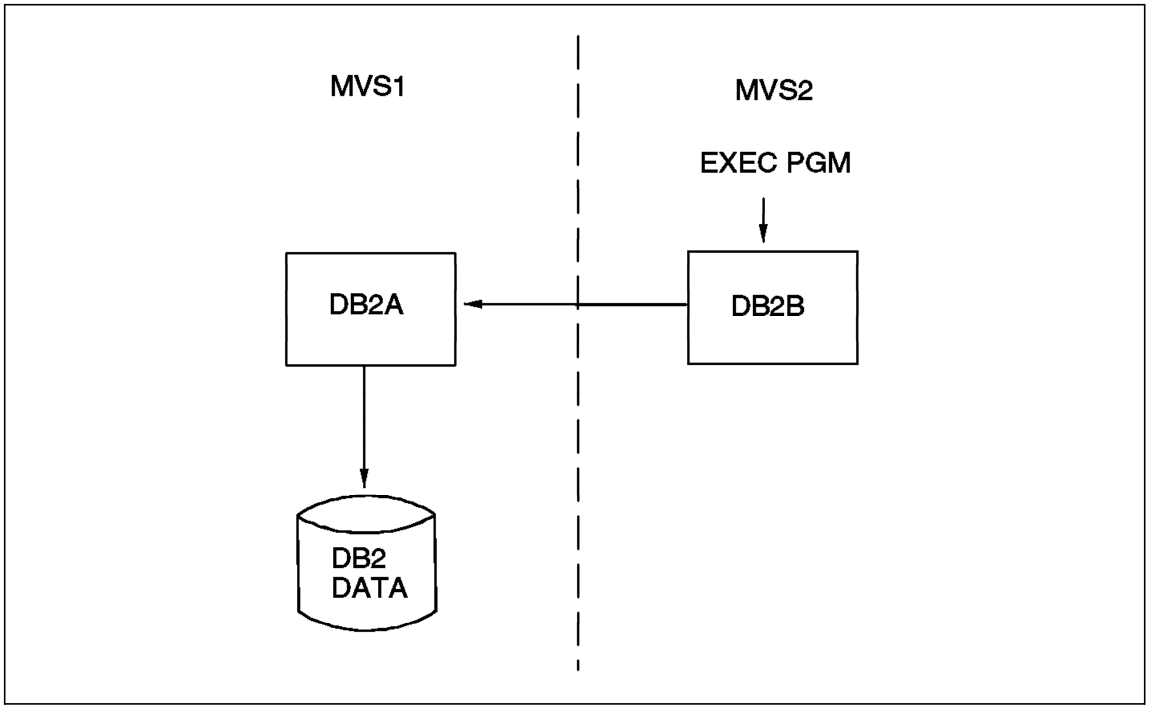


Figure 86. DDF Access to DB2 Data on Another MVS Image

With data sharing, DB2B can access DB2A′s data directly (see Figure 87 on page 231):

- Enable one-way data sharing, using DB2A as the originating member.
- If DB2B has any local user data, move the data to DB2A by performing a catalog merge.
- Delete DB2B.
- Create a DB2 subsystem on MVS2 as the second member of the data sharing group.

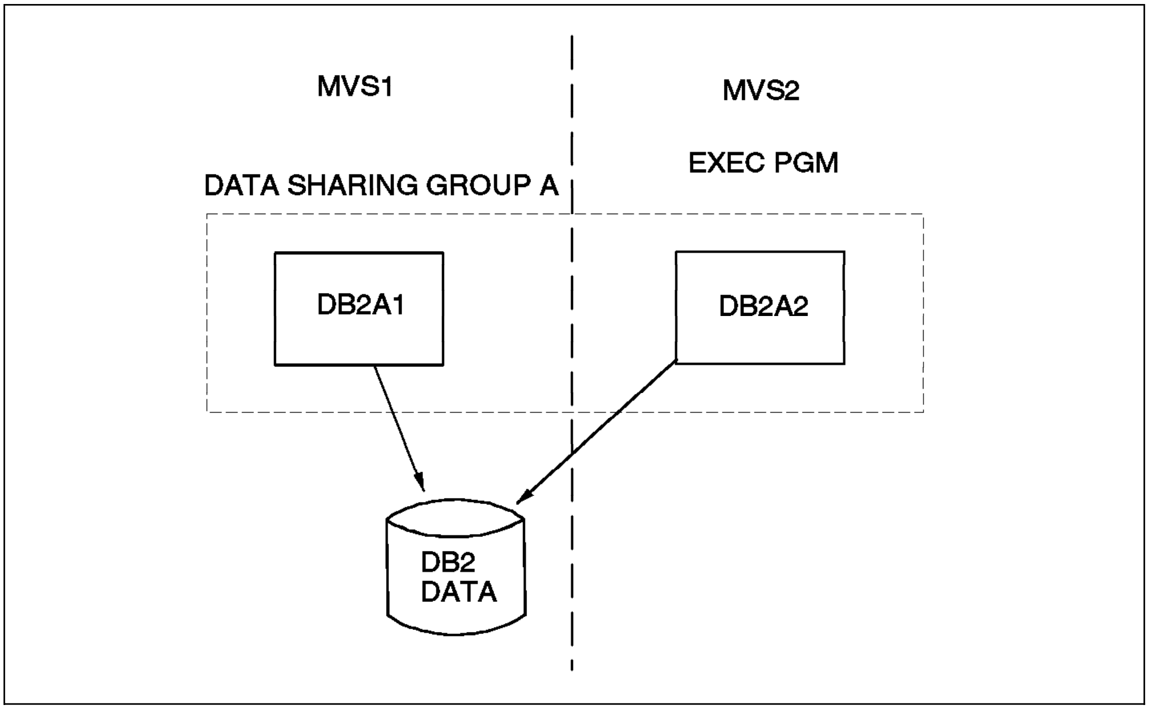


Figure 87. DDF Access Replaced by Enabling Two-Way Data Sharing

Now programs running on MVS2 have efficient direct access to DB2A′s data. The response times of applications will improve, as the path to the data is much shorter.

Depending on the type of distributed access—system directed or application directed—that was being used program changes might be required. For example, if your programs contained CONNECT statements that were used to direct access to the other DB2 subsystem, these statements would have to be removed.

In some cases where non-DB2 data and DB2 data were processed together, the non-DB2 data was transmitted to the processor where the DB2 data resided. These file transmissions are now no longer necessary as the DB2 data is accessible from both processors in our example.

Appendix A. Sample CFRM Policy

The following is a sample CFRM policy, where the DB2 structures and the structures of other subsystems like XCF are defined:

```
//*
//* JCL TO DEFINE THE ADMINISTRATIVE POLICY DATA IN THE
//* COUPLE DATA SET FOR COUPLING FACILITY RESOURCE MANAGER (CFRM)
//*
//*****************************************************************
//*****************************************************************
//*
//*  MULTIPLE STRUCTURES FOR SIGNALLING AND 2 COUPLING FACILITIES
//*
//*****************************************************************
//STEP1    EXEC PGM=IXCMIAPU
//SYSPRINT DD   SYSOUT=*
//SYSABEND DD   SYSOUT=*
//SYSIN    DD   *

     DATA TYPE(CFRM) REPORT(YES)

     DEFINE POLICY NAME(CFRM14) REPLACE(YES)
        CF NAME(CF01)
               TYPE(009672)
               MFG(IBM)
               PLANT(02)
               SEQUENCE(000000040104)
               PARTITION(1)
               CPCID(00)
               DUMPSPACE(2048)

        CF NAME(CF02)
               TYPE(009672)
               MFG(IBM)
               PLANT(02)
               SEQUENCE(000000040104)
               PARTITION(1)
               CPCID(01)
               DUMPSPACE(2048)
        STRUCTURE NAME(IXC_GRS_1)
               SIZE(16128)
               PREFLIST(CF01,CF02)
               EXCLLIST(IXC_GRS_2)

        STRUCTURE NAME(IXC_DEFAULT_1)
               SIZE(16128)
               PREFLIST(CF02,CF01)
               EXCLLIST(IXC_DEFAULT_2)

        STRUCTURE NAME(ISTGENERIC)
               SIZE(326)
               PREFLIST(CF02,CF01)

        STRUCTURE NAME(SYSTEM_OPERLOG)
               SIZE(1024)
               PREFLIST(CF02,CF01)

  /*----------------------------------------------------*/
  /* DB2 DATA SHARING GROUP: DSNDSGA / LOCK STRUCTURE    */
  /*----------------------------------------------------*/
        STRUCTURE NAME(DSNDSGA_LOCK1)
               INITSIZE(16000)
               SIZE(32000)
               PREFLIST(CF02,CF01)
               REBUILDPERCENT(5)

  /*----------------------------------------------------*/
  /* DB2 DATA SHARING GROUP: DSNDSGA / LIST STRUCTURE    */
  /*----------------------------------------------------*/
        STRUCTURE NAME(DSNDSGA_SCA)
               INITSIZE(4000)
               SIZE(10000)
               PREFLIST(CF02,CF01)
               REBUILDPERCENT(5)
```

```
  /*-----------------------------------------------------*/
  /* DB2 DATA SHARING GROUP: DSNDSGA / CACHE STRUCTURE(S)*/
  /*-----------------------------------------------------*/
        STRUCTURE NAME(DSNDSGA_GBP0)
               INITSIZE(8000)
               SIZE(16000)
               PREFLIST(CF02,CF01)
               REBUILDPERCENT(5)

        STRUCTURE NAME(DSNDSGA_GBP1)
               INITSIZE(8000)
               SIZE(16000)
               PREFLIST(CF02,CF01)
               REBUILDPERCENT(5)

  /*-----------------------------------------------------*/
  /* DB2 DATA SHARING GROUP: DSNDSGB / LOCK STRUCTURE    */
  /*-----------------------------------------------------*/
        STRUCTURE NAME(DSNDSGB_LOCK1)
               INITSIZE(16000)
               SIZE(32000)
               PREFLIST(CF02,CF01)

  /*-----------------------------------------------------*/
  /* DB2 DATA SHARING GROUP: DSNDSGB / LIST STRUCTURE    */
  /*-----------------------------------------------------*/
        STRUCTURE NAME(DSNDSGB_SCA)
               INITSIZE(4000)
               SIZE(10000)
               PREFLIST(CF01,CF02)

  /*-----------------------------------------------------*/
  /* DB2 DATA SHARING GROUP: DSNDSGB / CACHE STRUCTURE(S)*/
  /*-----------------------------------------------------*/
        STRUCTURE NAME(DSNDSGB_GBP0)
               INITSIZE(8000)
               SIZE(16000)
               PREFLIST(CF02,CF01)

/*
```

Appendix B. Special Notices

This publication is intended to help database administrators and systems programmers to implement DB2 data sharing in the Sysplex environment. The information in this publication is not intended as the specification of any programming interfaces that are provided by DB2 for MVS/ESA Version 4. See the PUBLICATIONS section of the IBM Programming Announcement for the DB2 product for more information about what publications are considered to be product documentation.

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

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

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to the IBM Director of Licensing, IBM Corporation, 500 Columbus Avenue, Thornwood, NY 10594 USA.

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

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

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

Any 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:

CICS	CICS/ESA
DATABASE 2	DB2
DFSMS	DFSMS/MVS
DFSMSdss	DFSORT
DRDA	IBM
IMS	IMS/ESA
MVS/ESA	OS/2
RACF	RMF
S/390	Sysplex Timer
System/390	VTAM
3090	

The following terms are trademarks of other companies:

C-bus is a trademark of Corollary, Inc.

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

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

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

Other trademarks are trademarks of their respective companies.

Appendix C. Related Publications

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

C.1 International Technical Support Organization Publications

For information on ordering these ITSO publications see "How To Get ITSO Redbooks" on page 241.

- *OS/390 MVS Parallel Sysplex Capacity Planning*, SG24-4680
- *DB2 V4 Data Sharing Performance Topics*, SG24-4611
- *MVS/ESA SP Version 5 Sysplex Migration Guide*, SG24-4581
- *DB2 V4 Non-Data-Sharing Performance Topics*, SG24-4562
- *System/390 MVS Parallel Sysplex Performance*, SG24-4356

C.2 Redbooks on CD-ROMs

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

CD-ROM Title	Subscription Number	Collection Kit Number
System/390 Redbooks Collection	SBOF-7201	SK2T-2177
Networking and Systems Management Redbooks Collection	SBOF-7370	SK2T-6022
Transaction Processing and Data Management Redbook	SBOF-7240	SK2T-8038
AS/400 Redbooks Collection	SBOF-7270	SK2T-2849
RISC System/6000 Redbooks Collection (HTML, BkMgr)	SBOF-7230	SK2T-8040
RISC System/6000 Redbooks Collection (PostScript)	SBOF-7205	SK2T-8041
Application Development Redbooks Collection (available soon)	SBOF-7290	SK2T-8037
Personal Systems Redbooks Collection (available soon)	SBOF-7250	SK2T-8042

C.3 Other Publications

These publications are also relevant as further information sources.

- *IBM DATABASE 2 for MVS/ESA Version 4 Data Sharing: Planning and Administration*, SC26-3269-01
- *IBM DATABASE 2 for MVS/ESA Version 4 Administration Guide*, SC26-3265
- *IBM DATABASE 2 for MVS/ESA Version 4 Utility Guide and Reference*, SC26-3395
- *IBM DATABASE 2 for MVS/ESA Version 4 Messages and Codes*, SC26-3268

- *IBM DATABASE 2 Performance Monitor for MVS (DB2 PM) Version 4 Online Monitor User's Guide*, SH12-6165
- *IBM DATABASE 2 Performance Monitor for MVS (DB2 PM) Version 4 Batch User's Guide*, SH12-6164
- *IBM DATABASE 2 Performance Monitor for MVS (DB2 PM) Version 4 Report Reference, Volume 1*, SH12-6163
- *IBM DATABASE 2 Performance Monitor for MVS (DB2 PM) Version 4 Report Reference, Volume 2*, SH12-6166
- *System/390 MVS Sysplex Overview*, GC28-1208
- *System/390 MVS Sysplex Systems Management*, GC28-1209-01
- *System/390 MVS Sysplex Hardware and Software Migration*, GC28-1210-01
- *System/390 MVS Sysplex Application Migration*, GC28-1211
- *MVS/ESA Setting Up a Sysplex*, GC28-1449-02
- *MVS/ESA SP V5 Programming: Sysplex Services Guide*, GC28-1495-02
- *MVS/ESA SP V5 Programming: Sysplex Services Reference*, GC28-1496-02
- *MVS/ESA SP V5 Planning: Installation and Migration with JES 2*, GC28-1428-02
- *MVS/ESA SP V5 Planning: Installation and Migration with JES 3*, GC28-1429-03
- *Large Systems Performance Reference*, SC28-1187-02
- *IBM 9037 Sysplex Timer and System/390 Time Management*, GG66-3264
- *VTAM V4 R3 for MVS/ESA Network Implementation Guide*, SC31-6548
- *CICS/ESA 4.1 Application Programming Guide*, SC33-1169

How To Get ITSO Redbooks

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

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

How IBM Employees Can Get ITSO Redbooks

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

- **PUBORDER** — to order hardcopies in United States
- **GOPHER link to the Internet** - type GOPHER.WTSCPOK.ITSO.IBM.COM
- **Tools disks**

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

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

  To get lists of redbooks:

  ```
  TOOLS SENDTO WTSCPOK TOOLS REDBOOKS GET REDBOOKS CATALOG
  TOOLS SENDTO USDIST MKTTOOLS MKTTOOLS GET ITSOCAT TXT
  TOOLS SENDTO USDIST MKTTOOLS MKTTOOLS GET LISTSERV PACKAGE
  ```

  To register for information on workshops, residencies, and redbooks:

  ```
  TOOLS SENDTO WTSCPOK TOOLS ZDISK GET ITSOREGI 1996
  ```

  For a list of product area specialists in the ITSO:

  ```
  TOOLS SENDTO WTSCPOK TOOLS ZDISK GET ORGCARD PACKAGE
  ```

- **Redbooks Home Page on the World Wide Web**

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

- **IBM Direct Publications Catalog on the World Wide Web**

  http://www.elink.ibmlink.ibm.com/pbl/pbl

  IBM employees may obtain LIST3820s of redbooks from this page.

- **ITSO4USA category on INEWS**
- **Online** — send orders to: USIB6FPL at IBMMAIL or DKIBMBSH at IBMMAIL
- **Internet Listserver**

  With an Internet E-mail address, anyone can subscribe to an IBM Announcement Listserver. To initiate the service, send an E-mail note to announce@webster.ibmlink.ibm.com with the keyword subscribe in the body of the note (leave the subject line blank). A category form and detailed instructions will be sent to you.

How Customers Can Get ITSO Redbooks

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

- **Online Orders** (Do not send credit card information over the Internet) — send orders to:

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

- **Telephone orders**

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

- **Mail Orders** — send orders to:

IBM Publications	IBM Publications	IBM Direct Services
Publications Customer Support	144-4th Avenue, S.W.	Sortemosevej 21
P.O. Box 29570	Calgary, Alberta T2P 3N5	DK-3450 AllerT3d
Raleigh, NC 27626-0570	Canada	Denmark
USA		

- **Fax** — send orders to:

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

- **1-800-IBM-4FAX (United States)** or **(+1) 415 855 43 29 (Outside USA)** — ask for:

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

- **Direct Services** - send note to `softwareshop@vnet.ibm.com`

- **On the World Wide Web**

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

- **Internet Listserver**

  With an Internet E-mail address, anyone can subscribe to an IBM Announcement Listserver. To initiate the service, send an E-mail note to `announce@webster.ibmlink.ibm.com` with the keyword `subscribe` in the body of the note (leave the subject line blank).

IBM Redbook Order Form

Please send me the following:

Title	Order Number	Quantity

- **Please put me on the mailing list for updated versions of the IBM Redbook Catalog.**

First name Last name

Company

Address

City Postal code Country

Telephone number Telefax number VAT number

- Invoice to customer number
- Credit card number

Credit card expiration date Card issued to Signature

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

DO NOT SEND CREDIT CARD INFORMATION OVER THE INTERNET.

Index

Special Characters

/*JOBPARM 166

Numerics

3480 control unit 43
3490 control unit 43
3990 Storage Control 43
9021 37, 40
9121 38, 40
934x control unit 43
9674 41

A

abend
  ARM 156
  generic LU 215
  lock preservation 63
  RESET option 124
  retained locks 78
  session 17
accounting traces 148
active log
  activity checklist 60
  ARCHIVE LOG command 181
  changing naming convention 51
  recovery 159, 184
  space 160
Advanced Peer-to-Peer Networking
  See APPN
affinity
  CICS 16, 20, 22
  performance 191
  VTAM 215, 217
  workload 5
ALTER access 91, 96, 101
ALTER GROUPBUFFERPOOL command 33, 83, 84, 152
ALTER option 125
ALTER TABLESPACE 25
AOR 11, 16, 17
APAR
  II09377 167
  OW15587 44, 81
  OW15590 44, 81
  OW16069 154
  OW16071 154
  OW19718 45
  PN76191 154
  PN79938 172
  PN84642 46
APPL 212
application owning region
  See AOR 11
application requester
  CDB 208
  data sharing group case 4 225
  data sharing group case 5 226
  data sharing group case 6 227
  data sharing group case 7 228
  DB2 for MVS/ESA (V2.3, V3, or V4) or other DBMS
    case 1 219
    case 2 222
  GENERIC column 212
  methods to connect 207
  non-data-sharing DB2 for MVS/ESA V4 - case 3 223
  RESET GENERICLU command 215
  resetting generic LU information 214
  SYSLULIST 213
  workstation 220
application server
  connectivity options 209
  data sharing group case 1 219
  data sharing group case 2 222
  data sharing group case 3 223
  data sharing group case 6 227
  data sharing group case 7 228
  DB2 for MVS/ESA V2.3 or V3 - case 4 225
  DB2 for MVS/ESA V4 or any other DBMS supporting DRDA - case 5 226
  GENERIC column 212
  group-generic method 208

application server *(continued)*
RESET GENERICLU command 215
resetting generic LU information 214
SYSLULIST 213
application-directed access 212, 231
APPN 46
archive log
activity checklist 60
data sharing environment 160
disaster recovery 180, 184, 188
naming convention 50
recovery 159, 160
remove member 108
volume serial number 165
ARCHIVE LOG command 135, 181
ARM
activity checklist 60
change of policy 158
couple data set 45
CPC failure 157
D XCF command 142
DB2 subsystem failure 172
planning for 44
using 155
attachment 4, 22, 111
authorization 120, 212
automated operation 59
Automatic Restart Management
See ARM
automatic space management 161
availability
advantages of data sharing 4
capacity planning 27
couple data set 46
distribution 7, 8
dynamic workload distribution 16
ESCON 43
structure placement 34
Sysplex Timer 40
workload distribution 15

B

backup 112, 181
backup castout owners 84
batch
ARM 155
batch *(continued)*
automation of disaster recovery 182
data partitioning 25
DB2 PM 193
distributing workload 7
global lock contention 198
lock avoidance 79
overhead 67
submitting work to be processed 137
bibliography 239
binary search 162
bind 4, 192
BSDS
adding a member 96
ARCHIVE LOG command 181
changes 165
changing naming convention 51
delta 116, 162
disaster recovery 180, 183, 185
GBP checkpoint 83
generic LU name 95, 100, 207
installation 91
member ID 104
merging existing DB2 subsystems 119
new installation 101
recover 159
reenabling data sharing 114
remove member 107, 109
buffer pool 3, 31, 33

C

cache structure 3, 31, 33, 79
caching 1
CAF 137
capacity
advantages of data sharing 4
dynamic workload distribution 16
loss 28
member-specific method 208
planning 27
service time 195
static workload distribution 14
castout
GBP checkpoint 83
I/O 3
owner 84, 200

castout *(continued)*
P-lock 64
page deregistration 81
processing 83
QUIESCE utility 165
threshold 152, 199, 203
catalog
automation of disaster recovery 182
disabling data sharing 112
enabling a data sharing group 87
GBP tuning 199
GRECP 177
merging existing DB2 subsystems 14, 117, 119
naming convention 50
recover 164, 186
sharing 2
SYSCOPY 185
CATMAINT utility 88
CDB
application requester 208, 212
case 1 219
case 2 222
case 4 225
case 5 226
data sharing group 207
RESET GENERICLU command 215
test cases 210
updating 214
CE 8
CEC 137, 173
central processor complex
See CPC
CFCC 3, 28, 40, 192, 195
CFLEVEL 44, 81, 153
CFRM policy
activate and deactivate 139
activity checklist 60
adding a member 95
adding a structure 151
changing 139, 158
couple data set 45
coupling facility failure 173
D XCF command 142
generic LU name 100
installation 90
lock structure size 198
CFRM policy *(continued)*
multiple coupling facilities 30
preference list 3
production environment 42
rebuild failure 173
structure size 34
change log inventory utility 91, 180
changed page 66, 165
channel 3, 27, 36, 42
checkpoint 65, 83
child L-lock 65, 75
CHPID 141
CICS
activity checklist 59
affinities 22
connecting to DB2 13
data sharing group 5
distribution 7
estimating overhead 10
group attachment 136
network planning 46
protected thread 10
session balancing 17
submitting work to be processed 136
WLM 21
workload balancing 5, 16
workload distribution 11
CICSPlex SM 8, 10, 11, 17, 21
class 2 elapsed time 204
class castout 83
clean page 26, 66, 79, 80
client/server 1
CLOCKxx member 40
CLSN 79
CMOS 8, 14
CNOS 212
CO 8
cold start 61, 110, 113
command prefix 59, 133, 136
command recognition characters 59
command scope 134
commands 133
commit
ARCHIVE LOG command 181
DB2 subsystem failure 170
force processing 81
GBP checkpoint 203

commit *(continued)*
GBP failure 174
GBP pages 33
GPOC 183
L-lock 65
lock avoidance 79
phase 2 31
processing cost 28
reduce contention 10
reduce duration of lock 10
review applications 6
row lock 76
synchronous write 202
two-phase 209, 214, 217, 228
communications database
See CDB 184
compression 160
concurrency 3, 25, 107
conditional restart control record 110
See CRCR
CONNECT statement 231
connection
deleting 177
persistent 179
connectivity
alternatives for distributed access 207
considerations 38
coupling facility failure 172
ESCON Director 43
exceptions 157
CONNFAIL 45, 154
contention
DB2 PM 69, 197
exit 72
global lock 192
lock structure 31
managing 66, 72
overhead 8
real 69
tuning 195
types 69
XES 68
continuous power supply 40
control regions 24
control unit 37, 42
conversation 208, 211
copy pending state 164
COPY utility 112, 160
couple data set 39, 45, 142, 156
COUPLExx member 39
coupling efficiency
See CE
coupling facility
accesses 192
activity checklist 60
adding a structure 151
capacity planning 27, 30
castout 204
channel 27
D CF command information 140, 141
dedicated 41
disaster recovery 184
failure 172
GBP structure size 202
GBPCACHE 26
generic LU name 217
lock propagation 63
LPAR 78
moving a structure 151
overhead 8
page refresh 81
page registration 81
page update 33
persistent structure 179
placement 34
production environment 41
propagation 76
rebuild failure 173
reenabling data sharing 113
reports 193
service time 194
services provided 3
shared with other logical partitions on same processor 41
shutting down procedure 153
sizing 7, 34
speed 37
storage planning 30
Sysplex 1
testing environment 42
XCF commands 139

Coupling Facility Control Program
  See CFCC
Coupling Facility Resource Management
  See CFRM policy
coupling overhead
  See CO
CPC
  DB2 subsystems 1
  failure 157
  nonrestartable wait state 158
  processor speed 192
CPU
  capacity planning 27
  castout 204
  class 2 elapsed time 204
  delta 191
  DSN1COPY utility 118
  GBP structure size 202
  processor speed 192
CRCR 180, 184, 185
CREATE INDEX 25
CREATE statements 122
CREATE TABLESPACE 25
Cross System Extended Services
  See XES 3
cross-system coupling facility
  See XCF 28
cross-system invalidation 3, 80, 81, 199, 201
cross-system restart 155
CSR
  adding a member 97
  BSDS 185
  new installation 103
current status rebuild
  See CSR
CURRENTDATA option 9, 28

D

D CF command 140
D XCF command 140, 142, 144, 178
damage assessment 154, 156, 157, 174
DASD
  archive log 159, 161
  cash control unit 43
  castout 81, 84
  changed pages 3
DASD *(continued)*
  connection to GBP 83
  contention 28, 84
  couple data set 46
  cross-system invalidation 199
  disaster recovery 188
  GBPCACHE 26
  merging existing DB2 subsystems 118, 130
  page invalidation 81
  shared 3
  utilities 167
DATA CAPTURE CHANGES option 112
data consistency 64, 65, 181
data definition 117
data integrity 76, 113, 156
data partitioning 25
data sharing
  disabling 61, 104, 109
data sharing group
  adding a member 92
  application requester 212
  application server 209
  backup 181
  breaking VTAM affinity 215
  communication among members 31
  CPC failure 157
  data partitioning 25
  disabling data sharing 110
  disaster recovery 181
  DSNHDECP 96
  dynamic workload distribution 16
  enabling 85, 87
  GBPCACHE 26
  generic LU name 207
  group attachment name 137
  install 4, 85
  location name 90
  member ID 104
  merging existing DB2 subsystems 117, 119
  MODIFY command 135
  naming convention 49
  network planning 46
  new installation 98
  operating 133
  performance 192
  QUIESCE utility 165
  RCT 13

data sharing group *(continued)*
real LU names 214
recovery 3, 159
reenabling 104, 113
remove member 107
resetting generic LU information 214
structure 3
test global lock serialization 106
database
design 24, 192, 195
exception status 3, 31
merging existing DB2 subsystems 117, 120, 123
naming convention 50
recover 39
workfile 187
daylight saving 39
DB2 PM 69, 147, 193, 196, 204
DBD 64
DBID 120, 126
DDCS 220
DDF
adding a member 94
CDB 210
disaster recovery 184
dynamic routing 44
generic LU name 218
installation 89
merging existing DB2 subsystems 117
naming convention 50, 52
network planning 46
new installation 99
replacing by data sharing 230
RESET GENERICLU command 216
session termination 208
static workload distribution 14
SYSLUNAMES 211
updating CDB 214
deadlocks 161
degree of sharing 16, 28, 30, 191, 192
delta 162, 165
deregistration 80
DFDSS 126
DFP 44
DFSMS/MVS 44
DFSMShsm 161
directory
cross-system invalidation 82
disabling data sharing 112
displaying information 148
enabling a data sharing group 87
GBP checkpoint 83
GBP tuning 199
GRECP 177
merging existing DB2 subsystems 117
naming convention 50
page registration 66, 80
recover 164, 186
sharing 2
SYSCOPY 185
disaster recovery
activity checklist 61
backup 180
testing 42
DISPLAY DATABASE command 67, 105, 170, 175
DISPLAY GBPOOL command 106, 199
DISPLAY GROUP command 35, 114, 146
DISPLAY GROUPBUFFERPOOL command 35, 146
DISPLAY THREAD command 136
DISPLAY UTIL command 181
DISPLAY XCF command 139
distributed data 4
distributed data facility
See DDF
distributed processing 5
Distributed Relational Database Architecture
See DRDA
distributed unit of work 214
dormant 107
down-level data set 120
DRDA 5, 210, 212, 219, 222, 226
DSN1CHKR utility 187
DSN1COPY utility 118, 122, 124, 130
DSN1LOGP utility 51, 180
DSN1PRNT utility 126
DSN6ARVP macro 50
DSN6SPRM macro 50
DSN6SYSP macro 51

DSN7502I message 173
DSN7503I message 173
DSN7504I message 173
DSN75071 message 92
DSNB228I message 157
DSNB250E message 156, 174
DSNB303I message 157
DSNB304I message 174
DSNB305I message 175
DSNB314I message 156
DSNB32xI message 157, 174
DSNDB01 database 88
DSNDB07 database 53
DSNHDECM macro 51
DSNHDECP module 91, 96, 137
DSNI021I message 177
DSNJ245I message 185, 186
DSNJU004 utility 51
DSNR020I message 108, 185
DSNR030I message 108
DSNT378I message 157
DSNTEJ1 job 104
DSNTIJFT job 113
DSNTIJGF job 113
DSNTIJSG job 211
DSNTIJTM job 54
DSNTIJUZ job 91, 96
DSNTINST CLIST 58, 86, 88, 93, 98
DSNTIPA panel 88, 161
DSNTIPAI panel 93, 98
DSNTIPJ panel 32
DSNTIPK panel 93, 98, 137
DSNTIPR panel 89, 94, 99, 207, 212
DSNUM keyword 187
DSNZPARM
  DEFER ALL 186
  disaster recovery 184
  IRLMRWT parameter 107
  member names 3
  naming convention 50, 53
  reenabling data sharing 113
  retained lock 78
  RETLWAIT option 172
DTR 16, 17, 20
DXR132I message 92
DXR137I message 157
DXR141E message 92
DXR143I message 173
DXR146I message 173
dynamic transaction routing
  See DTR

E

ECSA 32
EDM pool 32
EMIF mode 195
encryption 212
ENCRYPTPSWDS column 212
ESCON 1, 37, 42
exit 22, 66, 72, 209
expanded availability feature 40

F

failed-persistent connection
  deleting 178
  disaster recovery 184
  GBP 154
  lock structure 145
  reasons 177
failure
  communication 209, 218, 225, 229
  coupling facility 172
  CPC 157
  DB2 subsystem 170
  GBP 174
  lack of storage 201
  SCA 173
  subsystem 157
  Sysplex 158, 170
failure detection interval 39
fallback 47, 59, 86, 164
false contention
  detection 69
  lock table 31, 32
  performance 192, 195
  XES 28
fiber optic link 36, 40
forced affinity routing 16
foreign key 211, 213

forward log recovery phase 186

G

GBP
activity checklist 60
allocation 83, 152
cache structure 3
cache testing 104
castout 83, 203
changing 152
checkpoint 83, 84, 176, 199, 203
connection to DASD 83
coupling facility failure 172
data entries 33
data pages 199
dependency 66
description 79
directory 3, 33, 199
DISPLAY GBPOOL command 106
DISPLAY GROUPBUFFERPOOL command 146
displaying dependencies 150
failure 174
GBP0 152
getpage 82
loss of connectivity 157
messages 156
number of requests 9
page refresh 81
page registration 80
page sets 3
partitions 3
RACF 90
ratio 203
size 33, 199, 202
storage 31, 33
tuning 199
GBP-dependent page set
buffer steal 81
changed pages 3, 79
data partitioning 25
displaying lock information 148
force at commit processing 81
global lock contention 198
MRL 68
P-lock 64, 76
GBP-dependent page set *(continued)*
page registration 80
retained lock 78
row lock 76
Type 1 index 87
utilities 25
GBPCACHE 25, 26, 79
GCLSN 79
GENERIC column 212, 215
generic resource
installation 90
LU name
adding a member 95
application requester 217
CNOS 212
installation 100
mapping 207
planning 46
resetting LU information 214
resources 17, 18, 46, 60
getpage 82, 192, 204
global commit log sequence number
See GCLSN 79
global contention 8
global drain 177
global lock
capacity planning 27
contention 192, 198
data sharing 63
DB2 subsystem failure 170
displaying information 147
manager 68, 75, 76
negotiation 198
serialization 3
tuning 195
XES 3
global serialization 106
GMT 39
goal mode 45
GPOC 181, 182, 189
GRECP
displaying information 175
DSNB32xI message 157, 174
messages 156
removing 176
retained locks 177
utilities considerations 167

Greenwich Mean Time
See GMT
group attachment 111, 136, 166
group buffer pool
See GBP
group buffer pool recovery pending status
See GRECP 167
group name 49, 62
GROUP option 98
group restart
disaster recovery 185
lock structure rebuild failure 173
reenabling data sharing 113
remove member 108
retained lock 78
SCA structure rebuild failure 174
group-generic method
activity checklist 59
case 1 219
connectivity option 207, 209, 210
data sharing group case 6 227
GENERIC column 212
remove member 109
group-scope commands 134
GROUP_MEMBER column 164
groupwide point of consistency 135
See GPOC 181
GRS 28

H

hardware
activity checklist 58, 61
ARM 156
capacity planning 28
cost 5
coupling facility 140, 173
disaster recovery 183
planning 37
setup 5
structure size 7
workload distribution 15, 16
hardware configuration definition
See HCD
hash classes 63
hash table 63, 197
HCD 45
header page 118, 122, 126, 128
hierarchical locking 65, 78, 191
high-level qualifier 125
hiperpool 26, 79, 80, 203
HPDDBID 126
HPGPSID 126

I

I/O
castout 202, 204
contention 84
DSN1COPY 118
dynamic configuration 42
EMIF 195
GBP 82
HCD 45
performance 187
ICF 50
ICMF 38, 40, 41, 42
ICTYPE column 164
identifiers 118, 120
IEA262I message 158
IEA267I message 158
IEFSSNxx member 53, 111, 133, 137, 166
IFCID 259 148
II09377 167
image copy
activity checklist 62
disabling data sharing 112
DSN1CHKR utility 187
frequency 160
merging existing DB2 subsystems 125, 130
recovery cycle 160
RECOVERYSITE 184
reenabling data sharing 115
SYSCOPY 185
SYSLGRNGX 163
IMS
activity checklist 59
daylight savings 39
dynamic workload distribution 8
generic resources 22
group attachment 136
local shared resources 44
submitting work to be processed 136

inbound requests 212
incremental image copy
merging existing DB2 subsystems 118, 130
recovery cycle 160
reenabling data sharing 115
index
activity checklist 59, 62
capacity planning 27
displaying GBP dependencies 150
displaying lock information 147
DSNB32xI message 157
enabling a data sharing group 87
GBP 33
GBPCACHE 33, 79
GRECP 174
installing data sharing group 86
ISOBID 120
merging existing DB2 subsystems 118, 120, 123, 125, 129
nonpartitioning 177
P-lock 64
page set header 118
Type 1 118
Type 2 9, 59, 118
indoubt unit of work 137, 181, 185, 214, 216
INITSIZE parameter 35
INQUIRE EXITPROGRAM command 22
installation verification 104
Integrated Coupling Migration Facility
See ICMF
Integrated Resource Lock Manager
See IRLM 3
inter-DB2 interest
allocating a new GBP 152
CICS 16
dynamic workload distribution 16
GBP 66
L-lock 65
P-lock 64, 76
page deregistration 81
propagation 78
read-write 9
testing 105
internal identifiers 118, 120
interrupt 4, 199
invalidation 77, 81
IPL 40, 53, 59
IRLM
capacity planning 27
contention 72, 76
DXR132I message 92
DXR141E message 92
failure 172
global locking 63
L-lock 65
lock structure 31
MODIFY command 135
MRL 78
naming convention 50
overhead 204
RACF 100
real contention 197
rebuild failure 173, 174
reenabling data sharing 113
row lock 76
SLM 3
IRLMRWT parameter 107
ISOBID 120
ISOLATION parameter 9, 27
IXC353I message 179
IXC354I message 179
IXC360I message 180
IXC511I message 158
IXC518I message 157
IXC601I message 158
IXC602I message 155
IXC607I message 158
IXC609I message 155
IXC805I message 158
IXC806I message 158
IXCL1DSU utility 45
IXCMIAPU utility 139

J

JES2 28, 44, 135, 166
JES3 44

L

L-lock
description 65

L-lock *(continued)*
DISPLAY DATABASE command 171
displaying information 147
parent 65, 75
propagation 75, 78
reducing 10
retained 78
row lock 76
subsystem failure 157
latch 63, 65
LEVELID option 118, 120, 126
LGRMEMBR column 164
LGRRELRSN column 164
LGRSLRSN column 164
Licensed Internal Code 40
link 36, 139, 209
LINKATTR column 213
LINKNAME column 213, 214
list processing 1
list structure
activity checklist 60
capacity 31
SCA 3
LOAD utility 25, 177
local buffer pool
buffer steal 81
cross-system invalidation 81
GBP 79
page refresh 81
page registration 26, 80
performance 203
row lock 77
validation 80
local mode 158
local shared resources 44
local time 39
LOCATE block 127
LOCATION column 213
location name
adding a member 95
application 5
CDB 210
data sharing group 207
generic LU name 217
installation 90
new installation 100
SYSLOCATIONS 213
lock
avoidance 9, 28, 79, 198
capacity planning 27
components 67
contention 8, 28, 192, 195, 198
DB2 PM 69
DB2 subsystem failure 170
DISPLAY DATABASE command 106
duration 9, 65
explicit hierarchical 65, 78
false contention 31
force at commit processing 81
global locking 63
granularity 68
hash table 3
list entry 196
list table 3
modes 64
processing enhancements 44
propagation 9
reducing activity and contention 9
release 65, 170
retained 61, 78, 135, 157
row 10, 65, 76, 147
structure 31
test serialization 106
lock structure
activity checklist 60
capacity 31
changing 153
contention 31
coupling facility failure 172
DISPLAY GROUP command 146
DXR132I message 92
DXR141E message 92
failed-persistent 145
failure 173
global lock 63
initial space 31
IRLM 31
propagation 204
RACF 90, 100
rebuild 78, 151
serialize 3
size 28, 197

lock table
contention 69
hash 69
L-lock 64
P-lock 64
register interest 63
size 72
structure 31
LOCKINFO column 106, 171
LOCKSIZE parameter 10
LOCTYPE column 213
log
abend 215
apply phase 187
data sets needed to recover 162
disabling data sharing 110
disaster recovery 185
GBP checkpoint 83
lost analysis 112
member ID 104
rebuild failure 173, 174
record header 160
recovery considerations 159
reenabling data sharing 114
remove member 107
removing LPL and GRECP states 176
row lock 77
TOD 39
UR 162
log record sequence number
See LRSN
logical close 105
logical lock
See L-lock
logical page list
See LPL 81
logical partition 30, 40, 42, 177
logical paths 43
logically partitioned mode
See LPAR 30
LOGR 45, 135
LPAR 30, 173, 174, 193, 195
LPL
DEFER ALL 186
displaying information 175
force at commit processing 81
GBP failure 154, 174
LPL *(continued)*
GBP structure size 203
message 156
RECOVERYSITE 184
removing 176
retained locks 177
utilities considerations 167
write failure 201
LRSN
binary search 162
delta 116
disabling data sharing 110
GBP checkpoint 83
lock avoidance 79
recover 159, 162
reenabling data sharing 114, 116
SYSLGRNX 164
Sysplex Timer 39
LSPR ratio 30
LU name
adding a member 95
DDCS 221
generic 212, 217
installation 90
mapping 207
new installation 100
real 207, 213, 214
remove member 109
RESET GENERICLU command 216
resetting generic information 214
SYSLULIST 213
LUNAME column 211, 213, 214

M

maintenance 59, 153
MAXUSRS parameter 32, 72, 198
member ID
assignment 104, 114
BSDS 165
catalog and directory changes 164
LGRMEMBR column 164
reenabling data sharing 114
remove member 109
utility 165
member name 49, 164, 165

member-scope commands 134
member-specific method
  activity checklist 59
  breaking affinity 215
  case 3 224
  connectivity 208
  connectivity option 209, 210
  data sharing group case 3 223
  data sharing group case 7 228
  GENERIC column 212
  remove member 109
  SYSLULIST 213, 214
merge 5, 62
MERGE utility 130
MODESELECT column 212
modified resource list 3
  See MRL
MODIFY IRLMPROC command 135, 171
modify lock
  MRL 63, 68
  retained 78
monitoring tools 193
MRL
  change size 198
  lock components 67
  lock structure 31
  lock table size 31
  retained lock 78
  run-time information 63
MRO 11
MSC 23
multiple system coupling
  See MSC
must complete operation 31, 156, 174
MVS
  APAR 81, 154
  application requester 208
  ARM 155
  CFRM policy 30
  console 133
  IPL 40
  multiple images 42
  overhead 8, 28, 192
  prerequisites 44
  RACF 90
  receiving messages 135
  recovery bind 178
MVS *(continued)*
  software planning 43
  Sysplex 1
  TOD 39
  utilities 166, 167
  wait time 195

N

naming convention 49, 59, 85, 166
negotiation
  global lock 198
  P-lock 64
  row lock 76
  scenario 76
netid 215
network 22, 46, 213
nonpartitioned index 25
nonrestartable wait state 40, 158

O

OBID 118, 120
OBIDXLAT option 124, 125, 130
OPC/ESA 182
operations 133
OPERLOG facilities 135
optical disks 17
originating member
  data sharing group 87
  ENABLE data sharing option 88
  member ID 104
  merging existing DB2 subsystems 117, 119, 120
  naming convention 49
  new installation 98
  point-in-time recovery 168
  replacing DDF 230
  surviving member 114
  SYSLGRNX 114
OSAM 44
outage 4, 17, 172
outbound requests 212, 213
overhead
  activity checklist 59
  capacity planning 27
  DB2 204

overhead (continued)
  distribution 7
  EMIF mode 195
  estimating 10
  GBP-dependent 67
  IRLM 204
  reducing lock activity and contention 9
  workload distribution 16
OW15587 44, 81
OW15590 44, 81
OW16069 154
OW16071 154
OW16072 154
OW19718 45

P

P-lock
  DISPLAY DATABASE command 105, 171
  displaying information 147
  exit 66
  global lock 63
  negotiation 198
  page 65
  page set 76
  retained 78
  row lock 76
  subsystem failure 157
  types 64
package 4, 10, 125
page
  header 162
  invalidation 81
  P-lock 63
  registration 80
  validity 80
PAGE option 127
page set
  buffer steal 81
  castout 83, 165
  changed pages 3
  CLSN 79
  concurrent access 8
  displaying lock information 147
  DSNB32xI message 157
  GBP 79
  GBPCACHE 26
page set (continued)
  GCLSN 79
  header page 126, 128
  LEVELID 118
  lock avoidance 79
  lock propagation 63
  logical close 105
  P-lock 63, 75, 76
  page registration 80
  propagation 78
  retained lock 171
Parallel Sysplex
  See Sysplex
parent L-lock 65, 75
PARMLIB 39
partition
  buffer steal 81
  castout 83
  changed pages 3
  CLSN 79
  concurrency 25
  displaying GBP dependencies 150
  displaying lock information 147
  DSNB32xI message 157
  GBP 79
  GBPCACHE 26, 33
  GCLSN 79
  GRECP 174
  header page 127
  lock avoidance 79
  lock propagation 63
  logical 177
  page registration 80
  reenabling data sharing 114
partner LU 212, 215
password 212
PCTFREE 10
peer restart 185
pending status 62, 198
pending writes 51
performance
  activity checklist 61
  advantages of data sharing 4
  capacity loss 28
  COPY utility 160
  cross-system invalidation 82
  disaster recovery 187

performance *(continued)*
DISPLAY GROUPBUFFERPOOL command 148
ESCON 43
GBP structure size 202
goals 21
objectives 192
page invalidation 81
row lock 77
structure placement 34
tape compression 160
utilities 25
persistent structure 179
physical lock
See P-lock
physical open 66
PIT_RBA column 164
plan 4, 10, 32, 125
PN76191 154
PN78818 137
PN79938 172
PN84642 46
point-in-time recovery
disabling data sharing 112
GRECP 177
ICTYPE 164
LPL 177
originating member 168
reenabling data sharing 114
surviving member 165
TOLOGPOINT 163
TORBA 116
POLICY CHANGE PENDING status 151
polling loop 195
power source 40
PR/SM 30, 45
preference list 3, 151, 173
prefetch 44, 81
premount function 161
prerequisite MVS software 44
primary key 211, 213
print log map utility 51, 168, 182
priority 21
private protocol 212
processor 28, 37
propagation 64, 78, 204
pseudo open 66
PSID 120, 126

Q

Q662530 32
quiesce
ARCHIVE LOG command 181
changing naming convention 51
group restart 185
remove member 107
QUIESCE utility 165

R

RACF
activity checklist 60
adding a member 95
generic LU name 100
installation 90
level 44
naming convention 50
RBA
ARCHIVE LOG command 181
BSDS 165
conditional restart 181
delta 116
disabling data sharing 110
highest written 102
maximum to point-in-time recovery 168
merging existing DB2 subsystems 119
order of events 39
quiesce 182
recover 162
reenabling data sharing 114, 115
RESET option 124
row lock 77
RCT 13
RDB 213
read-and-register request 81
read-only switch 67
real lock contention 28, 68, 69, 192, 195
rebuild
activity checklist 60
failure 173
lock structure 173

rebuild *(continued)*
propagation 78
SCA 173
structure placement 34
REBUILDPERCENT 154
recall 161
receiving messages 135
reconnection 209
recover
activity checklist 60
application 159
catalog and directory 186
data sharing group 3
disabling 109
disaster
automation 182
backup 181
non-data-sharing 180, 189
preparation 188
DSNTIPA panel 161
environment 159
GBP checkpoint 83
member ID 104
merging existing DB2 subsystems 125, 129
persistent structure 179
point-in-time 92, 112, 116
reenabling data sharing 114
remove member 108
Sysplex Timer 39
tape compression 160
TOLOGPOINT keyword 92
TORBA keyword 92
RECOVER INDEX utility 125
RECOVER utility 92, 163, 167, 177
recovery pending status
See RECP 167
RECP status 167
reenable record 113
reenabling data sharing 104, 113
referential constraints 123
referential integrity 112
register-name-list request 81
registration 26, 80
relative byte address
See RBA
release coexistence 46
RELEASE parameter 10, 28, 65
remote unit of work 214
REORG utility 25, 124
REPAIR utility 118, 122, 123, 125, 130, 177
repeatable read 64
REPORT utility 114, 162, 163, 168
requests 212
RESET GENERICLU command 215, 217
RESET option 119, 124
response time 37, 204, 231
restart
adding a member 97
automatic 44
conditional 181
DB2 subsystem failure 170
disabling data sharing 110
disaster recovery 180, 185
GBP failure 174
must complete operation 156
new installation 103
recover 159
RECOVERYSITE 184
reenabling data sharing 113
retained lock 172
SET ARCHIVE command 161
RESTRICT option 167
resynchronization 209, 225, 229
RET_LKS column 135, 172
retained lock
activity checklist 61
DB2 subsystem failure 170
description 78
DISPLAY DATABASE LOCKS command 67
LPL and GRECP 177
MRL 63
negotiation 64
release 170
RET_LKS column 135
RETLWAIT parameter 78, 156
REXX EXEC 182
RLF 184
RMF 193
RO command 133
rollback 31, 77, 156, 174, 181

ROUTE command 133
routing a DB2 command 133
row header 118
row locking 10, 65
RUNSTATS utility 125

S

S/390 microprocessor 4, 38, 40
SAF 90, 95, 101
SCA
  adding a member 97
  coupling facility failure 172
  delta 116, 162
  DISPLAY GROUP command 146
  DSN75071 message 92
  failure 173
  GBP checkpoint 83
  list structure 3
  member ID 104
  number of requests 9
  RACF 90, 100
  rebuild 151, 173
  reenabling data sharing 113
  size 32
scheduling 5
security 44, 211, 212
security authorization facility
  See SAF 90
serialization 3, 106
service time 194
session
  balancing 17, 18
  definition 208
  group-generic method 207
  IMS 22
  two-phase commit 217
SET ARCHIVE command 161
SET TIME command 39
SETSSI command 53
SETXCF command 35
SETXCF FORCE command 178, 179
SETXCF START command 139, 151, 152, 155, 198
SETXCF STOP command 139, 155
SFM policy
  activity checklist 60
SFM policy *(continued)*
  change of policy 158
  couple data set 45
  coupling facility failure 172
  D XCF command 142
  rebuild 173
  starting 154
  structure placement 34
shared communications area
  See SCA 3
shared storage 40
SHRLEVEL 112, 130, 160, 163
shutdown 78
signaling 45, 199
sizing the GBP 34
SLM 3, 67
SMF 194, 204
SMS 44
software
  activity checklist 59
  capacity planning 28
  connectivity options 210
  cost 5
  maintenance 46
  setup 5
  workload distribution 15, 16
space map 65
SQLCODE 22, 87, 172
SSID 51
SSN 60
START DATABASE command 157, 177
START TRACE command 196, 204
START_RBA column 164
started task 45, 155
startup 3
static workload balancing 59
static workload distribution 11, 13
statistics report 196
statistics trace 148
status update missing 39
STCK value 162, 165
storage configuration 42
STORE CLOCK instruction 162
stored procedure 51
storm drain effect 22

STP 208, 209, 210, 213
structure
  adding to coupling facility 151
  changing 139, 150, 152
  D XCF command 143, 144
  deleting 177
  disaster recovery 183
  in transition state 179
  initial size 34
  moving to other coupling facility 151
  overhead 8
  pending state 198
  persistent 179
  recovery bind 178
  service times 194
  size 7, 139, 202
subpage 59, 87, 119
subroutine 6
subsystem name 53, 62, 137, 166
surviving member 110, 112, 113, 114, 165
suspension time 204, 205
SYNCLVL 215
SYS1.MIGLIB 45
SYSADM 50
SYSADM2 50
SYSCOPY 112, 164
SYSINDEXES 120
SYSLGRNX table
  GBP0 152
  LRSN 163
  member ID 104
  modifications 164
  originating member 114
  remove member 109
SYSLOCATIONS table 211, 213, 214
SYSLULIST table 109, 211, 213
SYSLUNAMES table 211, 212, 213, 214, 215
SYSMODESELECT table 212
Sysplex
  advantages of data sharing 4
  capacity planning 27
  change of policy 158
  consoles 133
  cost 5
  daylight saving 39
  DB2 subsystems 1
  disaster recovery 183
Sysplex *(continued)*
  distribution 7
  dynamic workload distribution 16
  failures 170
  hardware planning 37
  IMS 23
  manage 5
  one-way data sharing 192
  overhead 28, 192
  performance 193
  receiving messages 135
  remote location 186
  session balancing 17
  SFM 155
  static workload distribution 13
  XCF commands 139
Sysplex Timer 1, 37, 38, 40, 158, 162
Sysplex transaction program
  See STP
SYSTABLESPACE 120, 128
system data set 50, 51
system failure management policy
  See SFM policy
system lock manager
  See SLM
system logger 45, 135
system-directed access 212, 231
systemwide point of consistency 189
SYSUTILX 164, 165, 187

T

table space
  DBID 120
  disaster recovery 187
  displaying GBP dependencies 150
  displaying lock information 147
  DSNB32xI message 157
  GBPCACHE 79
  global lock contention 198
  GRECP 174
  header page 118
  merging existing DB2 subsystems 117, 123, 125
  nonpartitioned 126
  partitioned 127
  PSID 120

table space *(continued)*
recover 160
RECOVERYSITE 184
reenabling data sharing 114
tables 117, 123
tape 37, 42, 43, 60, 160, 161
terminal owning region
See TOR 11
thread 10, 65, 170, 217, 224
threshold 83, 84
throughput 25, 204
timeout 78, 107, 157, 172, 177
timestamp 39, 185
TOCOPY keyword 112, 114, 168
TOD 39
TOLOGPOINT keyword 92, 116, 168
TOR 11, 16, 17, 46
TORBA keyword 92, 116, 165, 168
TPN 213
TPNS 10
transaction lock 64
transaction program 208
transaction routing 16
TSO 137
tuning data sharing 191
Type 1 index 59, 87, 119, 129
Type 2 index
checklist 62
converting 87
DSNDB01 88
installing 86
lock 9, 28
logical partition 177
merging existing DB2 subsystems 129

U

UR isolation level 198
USERNAMES column 212
USERSECURITY column 212
USUMEMBR column 165
utility
activity checklist 62
buffer 118
concurrency 25
considerations 166
disaster recovery 180
utility *(continued)*
distribution 7
execution 166
ID 166
LRSN 165
remote location 187
stand-alone 167
submitting work to be processed 137

V

virtual storage 32
VSAM 44, 124
VTAM
activity checklist 60
adding a member 95
advantages of data sharing 4
affinity 228
APPL 212
CICS 17, 18
connectivity options 209
generic LU name 207, 217
IMS 22
installation 90
LU name 211
mode 211, 212
naming convention 50
new installation 100
planning 44, 46
remove member 109
resetting generic LU information 214, 216
SYNCLVL 215

W

weight 155, 172
WLM
activity checklist 62
CICS 21
couple data set 45
dynamic routing of DDF transactions. 44
STP 208
VTAM 17
work flow 4
workfile 109, 113, 187, 199
workload
activity checklist 58

workload *(continued)*
adding a member 92, 95
advantages of data sharing 4
balancing 5
capacity planning 27
CICS 11
connectivity options 209
DDCS 221
degree of sharing 192
distribution 7, 16
dynamic balancing 59
dynamic distribution 23
GBPCACHE 26
generic LU name 218
IMS 23
lock table size 31
remote location 186
scheduling 5
service time 195
session level 208
workload manager
See WLM 17
workoad
dynamic distribution 11
workstations 220

X

XCF
commands 139
component 69
couple data set 45
DISPLAY DATABASE LOCKS command 67
overhead 28
XES
component 3, 68
contention 28, 72, 76, 198
L-lock 65
multiple coupling facilities 30
page refresh 81
real contention 197
reports 197
SLM 67
XI signals 3

Printed in U.S.A.

SG24-4791-00