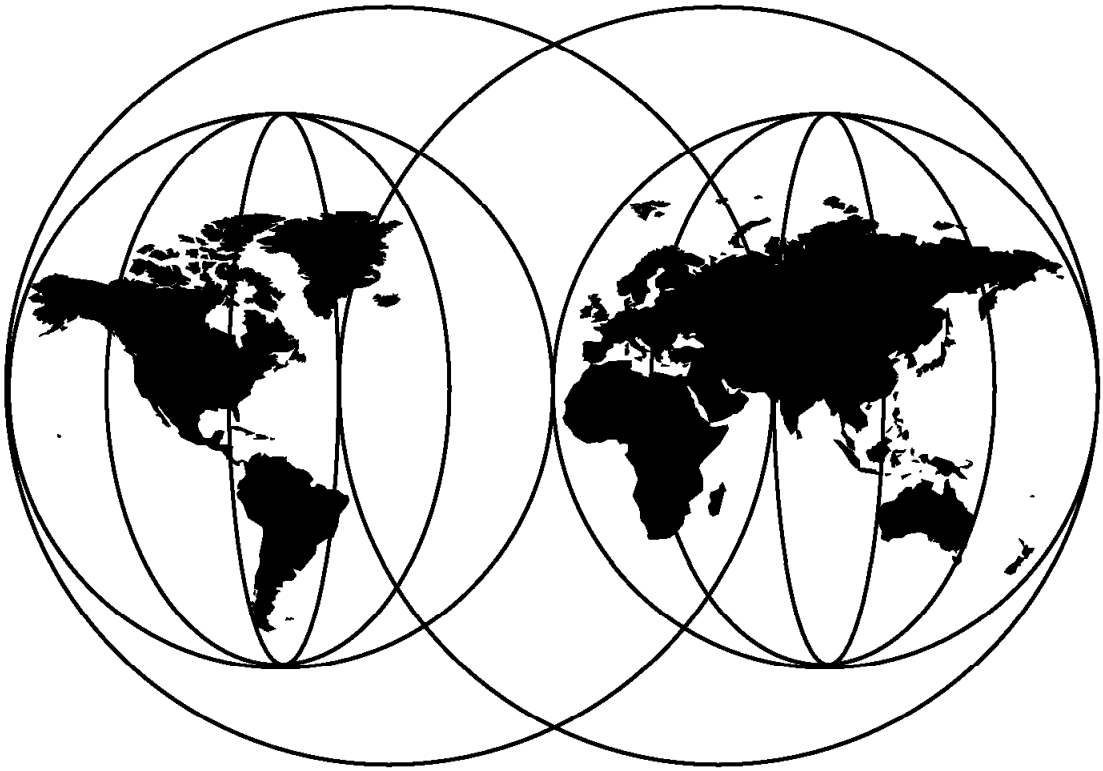
DB2 Performance Tuning on VSE and VM

Brummer Egeler-Brennenstuhl Foulds Hernandez Lange
Przytula Reimer Sarazola Wizemann Yang



International Technical Support Organization

http://www.redbooks.ibm.com

SG24-5146-00

International Technical Support Organization

SG24-5146-00

DB2 Performance Tuning on VSE and VM

May 1998

Take Note!

Before using this information and the product it supports, be sure to read the general information in Appendix C, “Special Notices” on page 203.

First Edition (May 1998)

This edition applies to Version 5, Release 1 Modification 0 of DB2 Server for VSE & VM, Program Number 5648-158, for use with VM/ESA or VSE/ESA.

Comments may be addressed to:
IBM Corporation, International Technical Support Organization
Dept. 3222 Building 71032-02
Postfach 1380
71032 Böblingen, Germany

Contents

Figures vii

Tables xi

Preface xiii
The Team That Wrote This Redbook xiii
Comments Welcome xiv

Chapter 1. Introduction 1
1.1 Organization of This Manual 1

Part 1. Enhancements 3

Chapter 2. Utilizing the New Functions 5
2.1 VSE Extended User Buffering 5
2.2 Isolation Level Uncommitted Read 5
2.2.1 Invocation 5
2.2.2 Usage 7
2.2.3 Locking 7
2.2.4 Storage Usage 8
2.2.5 Performance Implications 8
2.3 Faster Archiving 9
2.4 SET Initialization Parameters 9
2.4.1 Usage 9
2.4.2 Initialization Parameter Categories 10
2.5 SHOW Initialization Parameters 11
2.5.1 Usage 11
2.6 Display Statistics Initialization Parameter 11
2.6.1 Checkpoint Performance Information 12
2.6.2 Shutdown Counter Information 13
2.6.3 Checkpoint Statistics 13
2.6.4 Termination Statistics 15
2.7 Lock Wait Timeout 15
2.7.1 Usage 16
2.8 Increased Buffer Maximums 17
2.8.1 Positive and Negative Effects 18
2.8.2 Measurement and Tuning 19

Chapter 3. Disks and Caching 23
3.1 Minidisk Placement on RAMAC and Multiprise Internal Disk 23
3.2 Caching 23
3.2.1 Where Data can be Cached 24
3.2.2 What Data Should or Should Not be Cached 24
3.2.3 Measuring Cache Effectiveness 25
3.3 DB2 Server for VSE & VM Specific Caching Information 25
3.3.1 Caching Recommendations 26

Part 2. Tools 31

Chapter 4. Control Center Feature 33

4.1 Advantages of Using Control Center . . . 33
4.2 Control Center for VSE . . . 33
4.2.1 Description . . . 33
4.2.2 Database Monitor Tool . . . 35
4.2.3 Dbspace Analysis Tool . . . 37
4.2.4 Dbspace Reorganization . . . 38
4.3 Control Center for VM . . . 39
4.3.1 Environment . . . 39
4.3.2 Highlights and Capabilities . . . 40
4.3.3 Control Center Tools . . . 42
4.3.4 System Administration Tools . . . 43
4.3.5 Database Administration Tools . . . 45
4.3.6 Control Center Administration Tools . . . 49
4.4 Summary . . . 51

Chapter 5. SQL-Tune . . . 53
5.1 Installation of SQL-Tune on VSE . . . 53
5.1.1 Prepare Installation . . . 54
5.1.2 Install SQL-Tune from Product Tape . . . 54
5.1.3 Define VSAM Files . . . 56
5.1.4 Prepare CICS . . . 57
5.1.5 Acquire Password . . . 59
5.2 Installation of SQL-Tune on VM . . . 59
5.2.1 Define Resources . . . 59
5.2.2 Install SQL-Tune from Product Tape . . . 60
5.2.3 Generate Objects . . . 60
5.2.4 Acquire Password . . . 63
5.3 Using SQL-Tune on VSE and VM . . . 63
5.3.1 Performance Tuning . . . 63
5.3.2 SQL-Tune Utilities . . . 64

Chapter 6. VM:DB/Monitor . . . 85
6.1 Features and Benefits . . . 86
6.2 Installation of VM:DB/Monitor . . . 86
6.2.1 Prepare Installation . . . 86
6.2.2 Install the VM:DB/Monitor from Tape . . . 87
6.2.3 Customize and Run VM:DB/Monitor . . . 87
6.3 Components of VM:DB/Monitor . . . 90
6.3.1 DBMON . . . 91
6.3.2 DBMONVM . . . 91
6.3.3 IDBM . . . 94
6.3.4 DBMAP . . . 95
6.3.5 DBMONUP (Optional) . . . 102

Part 3. DB2 Tuning . . . 103

Chapter 7. Application Development . . . 105
7.1 Test Databases . . . 105
7.1.1 Defining a Simulation Test Environment . . . 105
7.2 Defining the Enterprise Rules . . . 108
7.3 Application Development Steps . . . 109
7.3.1 Overview . . . 109
7.3.2 Application Coding . . . 110
7.3.3 Application Testing . . . 111

7.3.4 Using SQL-Tune to See the Application Behavior . . . 111
7.3.5 Establishing Relationships Between Tables . . . 112
7.3.6 Estimating Space for Objects . . . 112
7.3.7 Defining Objects into Production . . . 112
7.3.8 Isolating CICS Applications in a VSE Environment . . . 113

Chapter 8. Application Tuning . . . 115
8.1 Tuning in General . . . 115
8.1.1 Table Growth . . . 115
8.1.2 Dbspace Statistics and History . . . 116
8.1.3 Index Statistics and History . . . 117
8.2 Tuning Your Current Applications . . . 118
8.3 How to Tune without Source Code . . . 122
8.3.1 Views on Read-Only Tables . . . 123
8.4 Tuning Dynamic Statements . . . 123

Chapter 9. Checklists and Recommendations . . . 125
9.1 Introduction . . . 125
9.1.1 Available Tools . . . 125
9.2 List of Questions to Users . . . 125
9.3 Performance Checklist for VSE and VM . . . 126
9.3.1 Applications . . . 126
9.3.2 Database . . . 129
9.3.3 System . . . 137
9.4 Performance Checklist for VSE . . . 138
9.5 Performance Checklist for VM . . . 138
9.6 Design Checklist . . . 141
9.6.1 Entities . . . 141
9.6.2 Attributes . . . 141
9.6.3 Normalization . . . 142
9.6.4 Data Types . . . 142
9.6.5 Physical Design . . . 143

Appendix A. Installed Environment . . . 145

Appendix B. Screens and Reports of VM:DB/Monitor . . . 147
B.1 IDBM Sample Screens . . . 147
B.2 DBMAP Activity Reports . . . 166
B.3 DBMAP Monitor Reports . . . 183
B.4 DBMAP Detail Reports . . . 198

Appendix C. Special Notices . . . 203

Appendix D. Related Publications . . . 207
D.1 International Technical Support Organization Publications . . . 207
D.2 Redbooks on CD-ROMs . . . 207
D.3 Other Publications . . . 207

How to Get ITSO Redbooks . . . 209
How IBM Employees Can Get ITSO Redbooks . . . 209
How Customers Can Get ITSO Redbooks . . . 210
IBM Redbook Order Form . . . 211

Glossary . . . 213

List of Abbreviations 217

Index 221

ITSO Redbook Evaluation 223

Figures

1. Syntax of the SELECT Command . . . 6
2. Syntax of the SET Operator Command . . . 9
3. Example of the SET Command . . . 10
4. Syntax of the SHOW INITPARM Operator Command . . . 11
5. Output from SHOW INITPARM . . . 11
6. Checkpoint Performance Information, Format 1 . . . 12
7. Checkpoint Performance Information, Format 2 . . . 12
8. Sample Result of the COUNTER * Command in VSE . . . 16
9. Sample Result of COUNTER POOL * Command in VM . . . 21
10. Output of 3990-6 CACHE Command . . . 25
11. Control Center for VSE Monitor Thresholds Screen . . . 36
12. Control Center for VSE Threshold Message Screen . . . 36
13. Control Center for VSE - Monitor Numbers . . . 36
14. Control Center for VSE Monitor Control Screen . . . 37
15. Control Center for VM Main Menu . . . 42
16. Control Center for VM Dbspace Reorganization Candidates Report . . . 48
17. Control Center for VM DBEXTENT-STORPOOL Mapping Report . . . 50
18. ISQL Command Sequence to Acquire Dbspace for SQL-Tune in VSE . . . 54
19. Sample JCL to Define Four VSAM Files for SQL-Tune in VSE . . . 56
20. Sample JCL to Initialize the VSAM Files for SQL-Tune in VSE . . . 56
21. DFHPCT Modifications for SQL-Tune in VSE . . . 57
22. DFHPPT Modifications for SQL-Tune in VSE . . . 57
23. DFHFCT Modifications for SQL-Tune in VSE . . . 57
24. DFHPLTPI Modifications for Statistics Capture on CICS Startup . . . 58
25. DFHPLTSD Modification to Stop Statistics Capture on CICS Shutdown . . . 58
26. Starting the CICS System for Use with SQL-Tune . . . 58
27. ISQL Command Sequence for Acquire Dbspace for SQL-Tune on VM . . . 60
28. Output of Q NSS for SQL-Tune in VM . . . 61
29. Output of VMFSETUP to Install SQL-Tune in VM . . . 61
30. Original ARIBLLLD EXEC . . . 62
31. ARIBLLLD EXEC Modifications for SQL-Tune in VM . . . 62
32. SQL-Tune Main Menu in VSE . . . 64
33. SQL-Tune Main Menu in VM . . . 65
34. SQL-Tune Development Function Overview . . . 66
35. SQL-Tune On-Line Statistics . . . 67
36. SQL-Tune Application Statistics . . . 68
37. SQL-Tune SQL Statement Tuning . . . 68
38. SQL-Tune Indentation of SQL Statement . . . 69
39. SQL-Tune SQL Access Path . . . 69
40. SQL-Tune Host Variables Specification . . . 70
41. SQL-Tune New Access Path . . . 71
42. SQL-Tune Execute Output . . . 71
43. SQL-Tune Display Output . . . 72
44. SQL-Tune Zoom Option . . . 72
45. SQL-Tune Data Base Scan . . . 73
46. SQL-Tune List of Dbspaces . . . 74
47. SQL-Tune List of Tables . . . 75
48. SQL-Tune Table Selection . . . 76
49. SQL-Tune Structure of Columns . . . 76
50. SQL-Tune List of Indexes . . . 77
51. SQL-Tune VSE - Sample JCL to Run XTS1MONA in a Separate Partition 78

52. SQL-Tune VSE - Monitoring the CICS SQL Activity . . . 79
53. SQL-Tune VSE - Detail of CICS SQL Activity . . . 80
54. SQL-Tune VSE - Detail of CICS SQL Statement . . . 81
55. SQL-Tune VSE - Monitoring the Batch SQL Activity . . . 81
56. SQL-Tune VSE - Detail of Batch SQL Activity . . . 82
57. SQL-Tune VM - Statistics Capture . . . 83
58. DBMON INITIAL File . . . 88
59. DBMONVM INITIAL File . . . 89
60. DBMAP Specification File . . . 90
61. Components of VM:DB/Monitor . . . 91
62. DBMONVM INITIAL File Setting Threshold and Notify . . . 92
63. DBMONVM Notify Message . . . 92
64. DBMONVM INITIAL File Setting Threshold and User Exit . . . 93
65. VM:DB/Monitor Sample Monitor Exit Routine . . . 93
66. VM:DB/Monitor Output of Monitor Exit Routine . . . 94
67. XDBMAP REXX Program Example . . . 101
68. SQL Statements for Statistics Capture . . . 107
69. SQL Statements for Statistics Update . . . 108
70. RDO CEDA EXPAND to Alter a Transaction Definition . . . 113
71. RDO Screen Showing the TClass Parameter . . . 113
72. CICS DFHPCT Table Showing the TClass Parameter . . . 114
73. CICS DFHSIT Showing the CMXT Parameter . . . 114
74. Sample Program for Table Growth Analysis . . . 115
75. Output Table Overview . . . 116
76. History Table Growth . . . 116
77. Output Dbspace Overview . . . 117
78. Index History Information . . . 118
79. Example 1: Statement from SQL-Tune . . . 119
80. Example 1: Access Path from SQL-Tune . . . 119
81. Example 1: Index . . . 119
82. Example 2: Index Definition . . . 120
83. Example 3: Statement with Dbspace Scan . . . 121
84. Example 3: Access Path from SQL-Tune . . . 121
85. Example 3: Index Structure . . . 121
86. Program Statistics . . . 122
87. Command to Display Index Usage . . . 122
88. Sample Index Usage Display . . . 123
89. Sample Program for Dynamic Statements . . . 124
90. Sample Output of the SHOW DBSPACE Command . . . 131
91. Basic Translation from Data Model to DB2 . . . 143
92. IDBM - Startup Screen . . . 147
93. PROFILE IDBM . . . 147
94. IDBM - Main Menu . . . 148
95. IDBM BUFFERS - Page Buffers by Dbspaceno . . . 149
96. IDBM COUNTERS - DB2 Counter Values . . . 150
97. IDBM DATABASE - Monitored Databases . . . 151
98. IDBM DBSTATS - Database Statistics . . . 152
99. IDBM DSCNTRS - VMDSS Counter Values . . . 153
100. IDBM EXTENTS - DB Extent I/O Activity . . . 154
101. IDBM OVERVIEW - Performance Overview . . . 155
102. IDBM PAGENTS - Pseudo-Agent Information . . . 156
103. IDBM PERFORM - Performance Summary . . . 157
104. IDBM POOLS - Storage Pool I/O Activity . . . 158
105. IDBM RAGENTS - Real Agent Information . . . 159
106. IDBM SQLLOG - SQL/DS Log Usage . . . 160

107. IDBM SQLLOG - SQL/DS Log Usage HISTORY 160
108. IDBM SYSSTATS - System Statistics 161
109. IDBM TOPUSERS - Top Users of Virtual CPU 162
110. IDBM USER - Detailed User Data 163
111. IDBM STTS - User Defined Panel 164
112. IDBM Sample Program for Screen Combination 165
113. DBMAP ACTDETL Activity Report 168
114. DBMAP ACTVTIME Activity Report 169
115. DBMAP DEADLOCK Activity Report 170
116. DBMAP DYNSQL Summary Activity Report 171
117. DBMAP LUWSTATS Summary Activity Report 172
118. DBMAP PACKAGED Summary Activity Report 174
119. DBMAP PACKAGES Summary Activity Report 175
120. DBMAP SECTIONS Summary Activity Report 176
121. DBMAP SECTSTAT Activity Report 177
122. DBMAP SESSIOND Summary Activity Report 178
123. DBMAP SESSIONS Activity Report 179
124. DBMAP TABNDXS Summary Activity Report 180
125. DBMAP TABNDXU Summary Activity Report 180
126. DBMAP TABLES Summary Activity Report 181
127. DBMAP USERRES Summary Activity Report 181
128. DBMAP USERTIME Summary Activity Report 182
129. DBMAP WAITLOCK Summary Activity Report 182
130. DBMAP LOG BUFFERS Summary Monitor Report 184
131. DBMAP COUNTERS Summary Monitor Report 185
132. DBMAP DBSTATS Summary Monitor Report 187
133. DBMAP DSCNTRS Summary Monitor Report 188
134. DBMAP EXTENTS Summary Monitor Report 189
135. DBMAP LOGDETL Summary Monitor Report 190
136. DBMAP PERFORM Summary Monitor Report 191
137. DBMAP POOLS Summary Monitor Report 192
138. DBMAP SQLLOG Summary Monitor Report 194
139. DBMAP SYSSTATS Summary Monitor Report 195
140. DBMAP THRESHOLD Summary Monitor Report 196
141. DBMAP TOPUSERS Summary Monitor Report 197
142. DBMDET DETAIL Report 198
143. DBMDET TABNDXD Report 199
144. DBMDET TRANSTAT Report 201

Tables

1. Valid Isolation Levels 8
2. CHKINTVL Multiple User Mode Initialization Parameter 14
3. Control Center Tools Summary 43

Preface

This redbook describes performance tuning of DB2 Server for VSE & VM Version 5 Release 1 from various aspects. It contains an overview of the performance relevant new functions and features, and how customers can benefit from them. Many details, techniques and recommendations about how to use these new facilities are provided as well as the typical steps for performance tuning regarding databases and applications. Checklists are provided to assist in analyzing performance problems, and recommendations are given for prevention. Sample tools are described that might help perform these tasks, and examples illustrate the versatility of these products.

This document is intended for use by database administrators, system programmers, application programmers and those who are involved in using DB2 Server for VSE & VM. It assumes a knowledge of DB2 database administration and basic knowledge of either VM/ESA or VSE/ESA.

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 Böblingen Center. It was created during several residencies in the ITSO Center Böblingen. These were conducted by:

Eberhard Lange, from the International Technical Support Organization, Böblingen Center

The authors of this document are:

Terence Foulds, IBM Toronto
Gys Brummer, IBM South Africa
Melanie Maria Egeler-Brennenstuhl, IBM Germany
Jose Hernandez, GBM Costa Rica
Angel Yang, IBM China
Hernan Sarazola, Ediguay S.A., Uruguay
Zbigniew Przytula, IBM Belgium
Carlos Henrique Reimer, TEKA Brazil
Andreas Wizemann, Fahrlehrerversicherung VaG, Germany

As well as writing this redbook, all residents contributed to the setup of demonstrations.

Special thanks also to

Dr. Wolfgang Krämer, IBM Laboratory Böblingen,

for his assistance, review and invaluable advice.

We appreciate the cooperation with

XT-Soft, Colombes, France
and
Sterling Software Inc., Germany

regarding their specific products. We thank them for the test license, support and feedback to the respective chapters.

Comments Welcome

Your comments are important to us!

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

- Fax the evaluation form found in “ITSO Redbook Evaluation” on page 223 to the fax number shown on the form.
- Use the electronic evaluation form found on the Redbooks Web sites:

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

- Send us a note at the following address:

  `redbook@us.ibm.com`

Chapter 1. Introduction

Performance has always been one of the most important issues in every data processing department. Performance relates to both hardware, such as CPU load and DASD I/O load, and software, such as user applications and database parameters. To identify the causes of a performance problem, each area must be analyzed carefully, and then the appropriate actions must be taken.

Optimizing the database can improve the performance, therefore this redbook describes new aspects since the previous redbook "*SQL/DS Version 3 Release 4 Performance Guide*." Also, implementing VMDSS, which is now a no-charge feature, on a VM database is worth the effort. You'll receive the best effects implementing it on top of a well tuned database.

The most important aspect, this means the effort promising the largest performance improvements from the invested time, is tuning the applications. Therefore, this redbook emphasizes this aspect and describes the steps for this both on VSE and VM.

Selected vendor programs have been used during the residencies writing this redbook; these programs can ease the tuning efforts and are described here as well. This comprises how these programs can be used during the various steps of tuning, but also what might be mentioned on top of the existing manuals regarding installation and handling.

1.1 Organization of This Manual

This manual is divided into three major parts:

1. **Enhancements**

   This part covers the performance aspects of the capabilities, features and tools that have been added in SQL/DS V3R5 and in DB2 Server for VSE & VM Version 5 Release 1, and in related hardware and software.

   Chapter 2, "Utilizing the New Functions" on page 5 describes the performance improvements of the new capabilities in SQL/DS V3R5 and DB2 Server for VSE & VM Version 5 Release 1.

   Chapter 3, "Disks and Caching" on page 23 describes the external influence on performance by the operating system and the I/O subsystem.

2. **Tools**

   This part covers the tools we used in the ITSO during the residencies writing this redbook. Control Center feature, and the vendor program SQL-Tune comprise both platforms, VSE and VM. VM:DB/Monitor is available for VM only.

   Chapter 4, "Control Center Feature" on page 33 gives an overview of the different tools available in Control Center and a summary of the advantages and benefits that can be expected by implementing Control Center in VSE or VM.

   Chapter 5, "SQL-Tune" on page 53 presents the installation and usage of SQL-Tune on VSE and VM. This tool can help you in locating a performance problem and gives you the possibility to change a statement on-line and see

the behavior of this statement after the modification. It will also warn you if some conditions occur that need special attention.

Chapter 6, "VM:DB/Monitor" on page 85 presents the installation and usage of VM:DB/Monitor. VM:DB/Monitor is a comprehensive facility for monitoring and reporting on the performance of DB2 Server for VM. It collects data about how both applications and users access and use the DB2 databases as well as the resource consumption.

3. **DB2 Tuning**

   This part offers practical hints and tips about how to tune a database and applications using the database. Also, comprehensive recommendation lists are offered and checklists to analyze performance problem causes.

   Chapter 7, "Application Development" on page 105 describes the steps typically done during development of a new SQL application. Additionally, it describes recommended procedures, test environments and responsibilities.

   Chapter 8, "Application Tuning" on page 115 describes some guidelines for performance tuning of existing applications, whether the source code still exists or not. It also describes some general techniques that can be helpful while tuning applications or the database.

   Chapter 9, "Checklists and Recommendations" on page 125 offers lists of items to be checked in case of a performance problem or during regular performance monitoring and tuning. Additionally, it contains lists of recommendations and considerations on the various subjects in order to help prevent performance problems from arising.

Appendix A, "Installed Environment" on page 145
gives an overview of the installed hardware and software at the ITSO.

Appendix B, "Screens and Reports of VM:DB/Monitor" on page 147
shows examples of all predefined panels and reports available with VM:DB/Monitor.

Part 1. Enhancements

This part covers the performance aspects of the capabilities, features and tools that have been added in SQL/DS V3R5 and in DB2 Server for VSE & VM Version 5 Release 1, and in related hardware and software.

Chapter 2. Utilizing the New Functions

This chapter describes the performance improvements of the new capabilities in SQL/DS V3R5 and DB2 Server for VSE & VM Version 5 Release 1.

2.1 VSE Extended User Buffering

User Buffering is an improvement on VSAM which reduces the number of I/O requests (interrupts) by bundling several requests into a single I/O request, so the applications can get better response times when they have to access data that is stored on VSAM files.

The following subsystems use VSAM User Buffering under VSE/ESA:

1. ADSM
2. DL/I
3. **DB2 Server for VSE & VM Version 5 Release 1**

To make DB2 Server for VSE Version 5 Release 1 use this new function and benefit from it, **APAR PQ11656** must be applied. This APAR needs VSAM **APAR DY44327** as a prerequisite.

2.2 Isolation Level Uncommitted Read

The isolation level Uncommitted Read (UR) gives users the ability to query data simultaneously by many applications while being updated by another application. This will prevent read-only applications from waiting on applications that have changed or may change the data about to be read. This provides greater concurrency and throughput.

The isolation level UR is defined as follows:

- An application can see but cannot update uncommitted changes made by other application processes.
- The re-execution of a statement can be affected by other application processes. This means, the results can differ.
- Accessed rows can be read and updated by other application processes.
- The current row of an update cursor cannot be changed by other application processes. An update automatically changes to the isolation level of Cursor Stability (CS).

2.2.1 Invocation

The isolation level UR can be specified in the following ways:

1. The SELECT statement has a WITH clause that allows the isolation level to be specified.

   The syntax is as follows:

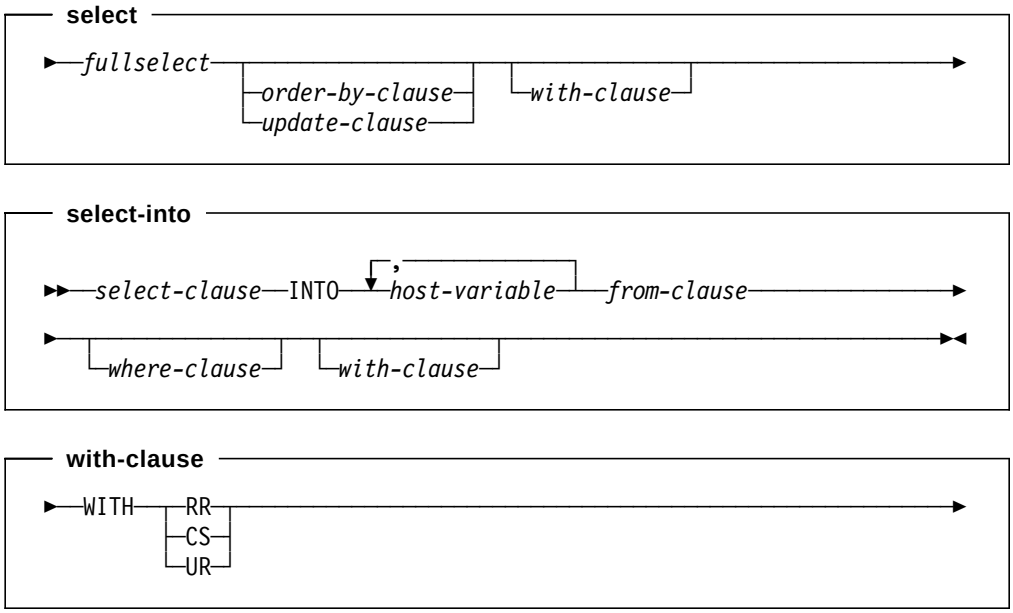


Figure 1. Syntax of the SELECT Command

The INSERT, UPDATE and DELETE statements also have the WITH clause, but will only be able to specify isolation level of Repeatable Read (RR) or Cursor Stability (CS).

2. When preprocessing an application program:
   - In VM, the SQLPREP statement has an ISOL parameter where the isolation level can be defined. Refer to “Using the SQLPREP EXEC Procedure” in Chapter 4 of the *DB2 for VM Application Programming* manual for more details.
   - In VSE, the ISOL parameter can be defined as an option of the PARM keyword of the job control EXEC statement. Refer to “Defining the Preprocessing Parameters” in Chapter 4 of the *DB2 for VSE Application Programming* manual for more details.

   An application program can even control its isolation level dynamically, depending on the processing requirements, if it is preprocessed with ISOL(USER).
3. In ISQL or DBSU using the SET ISOLATION UR statement.

2.2.2 Usage

Warning

The isolation level Uncommitted Read introduces data integrity issues. Users can see data that has changed but has not yet been committed. This data may be rolled back to its original state or it may contain just a part of a Logical Unit of Work (LUW), but the user reading the changed data is not aware of that.

This might be acceptable for queries giving an overview for statistical purposes only.

You must not use single data gained through reading with UR as input to another application or to a database that could then contain incorrect or invalid results. Therefore, when you read single data with the intent to do individual processing or to use it as input to a database, it is preferable to use either of the isolation levels RR or CS.

2.2.3 Locking

The isolation level UR is only available on data in public dbspaces with row or page level locking. If data is either in a public dbspace with dbspace level locking or in a private dbspace, isolation level RR will be used if UR has been requested.

Two additional lock modes have been introduced to handle the isolation level UR:

IN (Intent None) Acquired for read-only operations under isolation level UR for read of any data, even uncommitted data. Can be obtained at the dbspace, table, page or row level. This lock prevents index or data pages being split while reading.

Z (Super Exclusive) Acquired at dbspace, table, page or row level. No other applications can read or update an object locked in Z mode, even if using isolation level UR. This is used during index page splitting and during updating long fields.

Lock mode X has been modified:

X (Exclusive) Exclusive locks are acquired for DML operations only. Objects locked in X mode can be read by applications using isolation level UR.

Other changes include:

- DDL statements acquire a Z lock on the dbspace.
- LOCK DBSPACE ... IN EXCLUSIVE MODE acquires a Z lock on the dbspace.
- LOCK TABLE ... IN EXCLUSIVE MODE acquires a Z lock on the table and an IX lock on the dbspace.

The SHOW LOCK operator commands have been modified to display the two new lock modes.

Table 1. Valid Isolation Levels		
Isolation Level	**Lock Waits**	**Description**
UR (Uncommitted Read)	very few	Allows users or applications to read data from a public dbspace even if another user or application is writing to the same data.
CS (Cursor Stability)	YES	Applications using CS must wait on data until other applications are no longer updating it.
RR (Repeatable Read)	YES	Users or applications are completely isolated from interference by other applications during a Logical Unit of Work (LUW). Tables in private DBSPACES are always specified with isolation level RR.

The following lock waits can occur with isolation level CS and RR:

S (Share)
X (Exclusive)
Z (Super Exclusive)
IS (Intent Share)
IX (Intent Exclusive)
IN (Intent None)
SIX (Share with Intent Exclusive)
U (Update)

The durations of the locks can vary between instant, short, medium or long.

The following lock wait can occur with isolation level UR:

Z (Super Exclusive)

Note: Using the isolation level UR will not completely eliminate lock contention. Operations using this isolation level still need to wait for objects being processed under Z mode (see above). Deadlocks are still possible.

2.2.4 Storage Usage

Applications using isolation level UR perform less locking than other applications. It may be appropriate to review the NLRBS and NLRBU initialization parameters if a database supports mainly query applications using isolation level UR, as each agent requires fewer LRBs. Thus, less virtual storage is needed.

2.2.5 Performance Implications

In environments where multiple read-only operations are performed on a database, significant performance improvements for isolation level UR can be realized in two ways:

1. Applications using isolation level UR read the same data simultaneously, avoiding the lock waits which would occur with isolation level CS. In this type of environment, applications using isolation level CS must wait on data until other applications are no longer updating it.
2. The path length is shorter for applications using isolation level UR since fewer locks are required.

Additional locking is done for rows containing long fields to preserve data integrity. The reason is, although data might be rolled back or represent only a part of an LUW (inconsistent data - see the warning above), the data must be read correctly and completely. There is an instant Z lock on the base row when deleting or updating a long field, and a medium IN lock when reading the long field which is released after the long field has been read. The IN lock on the base row is not required if the long fields are not accessed.

Short IN locks are acquired on the index root and leaf pages to prevent an index page from being split while it is being read under UR. Conversely, an index page split requires a Z lock on the root and the predecessor page, to prevent a page being read under UR while it is being split concurrently.

2.3 Faster Archiving

With SQL/DS V3R5, archives and log archives have improved a lot; we measured an archive improvement of more than factor 5.

Additionally, the Data Restore feature was introduced. With DB2 Server for VSE & VM Version 5 Release 1 storage pool level recovery (SPLR) has been added into the Data Restore feature.

Although this influences performance a lot, especially during on-line archiving, or can increase the availability time of the database, this is not discussed here. The whole recovery strategy is affected and might be revised. This is described in detail in the redbook *DB2 Recovery on VSE and VM Using the Data Restore Feature*.

2.4 SET Initialization Parameters

Some initialization parameters can be dynamically modified from the operator console. The initialization parameters CHKINTVL, DISPBIAS, DSPSTATS, DUMPTYPE, and LTIMEOUT can be changed using the SET command. Previously, these initialization parameters could only be changed during database initialization.

2.4.1 Usage

SET commands can be issued from the operator console only.

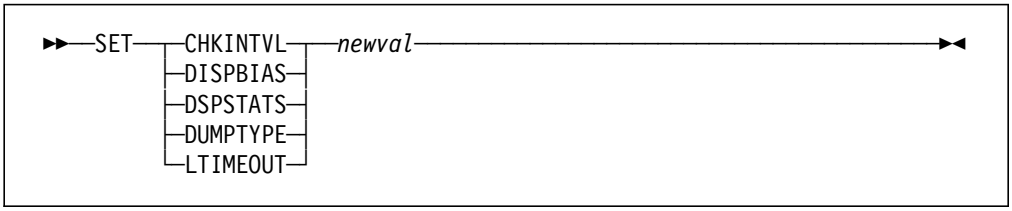


Figure 2. Syntax of the SET Operator Command

```
set ltimeout 15
ARIO065I SQL/DS operator command processing is complete.

set chkintvl 50
ARIO065I SQL/DS operator command processing is complete.
```

Figure 3. Example of the SET Command

Modifying **CHKINTVL** can assist in performance tuning of the DB2 Server. CHKINTVL is the maximum number of log pages written before a checkpoint occurs. Setting the CHKINTVL value higher will allow performance critical applications to take advantage of a larger gap between checkpoints, improving performance. When these applications are complete, the CHKINTVL value can be reset.

Changing **DISPBIAS** levels will allow the dispatcher to behave more like a priority dispatcher, or like a round-robin dispatcher. This could be used to manage different system loads.

Changing the **DSPSTATS** parameters can help to take a snapshot of checkpoint processing without having to spend the resources to run it all the time the database manager is up. Likewise, it is helpful to be able to turn on the final counter information if you detect the need for it while the database is running.

Changing the **DUMPTYPE** level will allow different levels of information to be captured for any problems that occur. This may be useful if a problem occurs and N (None) or P (Partial) has been specified and an F (Full) dump is necessary.

Modifying **LTIMEOUT** can improve performance in applications where lock contention has begun to affect performance and concurrency levels: It could initially be set to the maximum time a user wishes to wait for a lock, and tuned based on performance and concurrency levels.

This command is available in both VM and VSE.

2.4.2 Initialization Parameter Categories

The parameters described in the following subjects are called Multiple User Mode (MUM) **initialization parameters**.

The initialization parameter indicating MUM is the **SYSMODE=M** parameter. The initialization parameters can be split into the following groups:

- **Environment Parameters**

  for example DBname, AMODE, SYSMODE, STARTUP, DSPSTATS ...
- **Performance Parameters**

  for example NCUSERS, NPAGBUF, NDIRBUF, LTIMEOUT ...
- **Recovery Parameters**

  for example LOGMODE, CHKINTVL, ARCHPCT ...
- **Service Parameters**

  for example DUMPTYPE, TRACBUF ...

Some of these parameters are also valid in single user mode (SUM). There are parameters also outside the group of performance parameters that have quite some influence on performance, such as the CHKINTVL.

2.5 SHOW Initialization Parameters

Initialization parameters can be viewed using the new operator command SHOW INITPARM. Previously, these parameters could not be viewed from the database. They are stored in a file on the DB2 Server machine, and access to this file was required to determine which parameters were in effect.

This is especially important because they might have been changed with the SET command described in 2.4, “SET Initialization Parameters” on page 9.

2.5.1 Usage

SHOW INITPARM can be issued from the operator console, an ISQL session or by using the RXSQLOP EXEC.

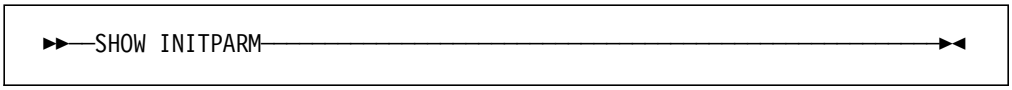


Figure 4. Syntax of the SHOW INITPARM Operator Command

```
Initialization parameters at 1997-11-07  10:40:30

AMODE    31            DUALLOG  N             PROTOCOL SQLDS
SYNCPNT  N             ACCOUNT  N             DBMODE   G
DUMPTYPE F             LOGMODE  Y             STARTUP  W
SYSMODE  M             EXTEND   N             CHARNAME INTERNATIONAL
DBNAME   ELDB2A                               RESID    ELDB2A
PARMID   ELDB2A        TRACDBSS 00000000000   TRACDSC  00
TRACRDS  0000000       TRACCONV 0             TRACSTG  0
TRACEBUF 0             ARCHPCT  80            CHKINTVL 250
NCSCANS  30            NCUSERS  20            NDIRBUF  500
NLRBS    10050         NLRBU    1000          NPACKAGE 10
NPACKPCT 30            NPAGBUF  1000          SLOGCUSH 90
SOSLEVEL 10            DISPBIAS 7             LTIMEOUT 2
DSPSTATS 21            MAPPING  L             SAVEINTV 10
SEPINTDB Y             TARGETWS 32

ARI0065I Operator command processing is complete.
```

Figure 5. Output from SHOW INITPARM

This command is available in both VM and VSE.

2.6 Display Statistics Initialization Parameter

The display statistics (DSPSTATS) parameter belongs to the category of environment parameters among the **initialization parameters**.

DSPSTATS represents two new initialization parameters. Both are also supported in single user mode. The positional parameters control respectively:

- checkpoint performance information
- shutdown counter information

If only one parameter is given, it is taken to be control checkpoint performance data, whereas the shutdown counter information setting remains unaffected. It is not possible to set shutdown counter information and leave checkpoint performance information unaffected. The latter is set although it may be the same value as before.

2.6.1 Checkpoint Performance Information

The first parameter indicates whether (and how) checkpoint performance information is displayed. If only one parameter is given, it is understood to be this one.

"**0**" means that no checkpoint performance information is displayed, as in previous releases.

"**1**" means that for each checkpoint a message is given showing that a checkpoint has occurred, and the point in time of checkpoint scheduling, start, and end.

```
ARI2052I A checkpoint was taken.
         Scheduled:  17:54:27 Started:  17:54:27 Ended:  17:54:27
```

Figure 6. Checkpoint Performance Information, Format 1

- By checking the scheduled timestamp you can get a feeling for how often a checkpoint is required.
- If the start time is significantly later than the scheduling, it had to wait for an LUW to finish.
- If the end timestamp is significantly later than the start, the checkpoint must have had a lot of work to do. This causes users to wait. Response time is perceived to be slow.

"**2**" means that in addition to the information given with "1," resource usage and other information is displayed:

```
ARI2052I A checkpoint was taken.  Type = Periodic
         Scheduled:  18:05:31 Started:  18:05:31 Ended:  18:05:31
         DIRREAD  = 0  DIRWRITE = 8
         PAGEREAD = 0  PAGWRITE = 0
         DSREAD   = 0  DSWRITE  = 0
         Time spent waiting to start:       0.002 seconds
         Time spent executing:              0.115 seconds

         Log pages filled since last checkpoint: 0
```

Figure 7. Checkpoint Performance Information, Format 2

- The type of the checkpoint is given.
- The number of directory reads and writes that were required by the checkpoint processing is given. DIRREAD, DIRWRITE, PAGEREAD and PAGWRITE counters are saved at checkpoint start, and at the end the difference is displayed. If dataspace support is installed (on VM only), dataspace reads and writes are also displayed. This gives an indication of how much work the checkpoint had to do.

- The number of log pages written is given as an indication of why the checkpoint occurred. If the number of log pages written is close to the CHKINTVL value, then the checkpoint was done because the CHKINTVL value was reached. If the number of log pages written is less than CHKINTVL, the checkpoint was scheduled for some other reason. This could be because of a DROP DBSPACE command or a commit of an update to a non-recoverable storage pool, for example. See the *Performance Tuning Handbook* (SC09-2402-00) for a complete list of when checkpoints occur.

This information offers a much better idea whether checkpoints are taken too often or too seldom, and thus helps in performance tuning.

Note: Avoid displaying checkpoint performance data constantly because it takes resources in itself. The DSPSTATS setting can be changed while the database is running.

2.6.2 Shutdown Counter Information

The second parameter, if given, indicates whether shutdown counter information is displayed or not.

0 means no display as in previous releases.

1 means, that a “COUNTER *” operator command is performed at shutdown time, be it an SQLEND or an abnormal end, even in single user mode. If dataspaces support is active “COUNTER POOL *” is performed also, which displays two sets of counters: One set is for storage pools, directory and unmapped internal dbspaces using VMDSS, the other is for pools and directory using the standard DASD I/O system.

This information is helpful because it gives you the counter information at the very end of the database manager.

2.6.3 Checkpoint Statistics

This information can help you to find the optimum value of CHKINTVL for your installation, or to locate performance problems if they are related to checkpoint processing. The checkpoint interval (CHKINTVL) parameter belongs to the category of recovery parameters among the **initialization parameters**, but has quite some influence on performance.

CHKINTVL is the maximum number of log pages written before a checkpoint occurs. If LOGMODE=Y, A or L is specified, the database manager writes to the log all affected data when you perform an INSERT, UPDATE or DELETE. The more modifications you make, the faster you will reach the checkpoint.

A checkpoint is an internal operation during which the database manager stops servicing users and copies a snapshot of the database to disk. This snapshot includes updates from the completed LUWs as well as from those that are still in progress and writes them to DASD. A checkpoint serves three important purposes:

1. Store the database to disk for recovery
2. Free storage pool space by releasing the shadow pages.
3. Free log space, depending on the LOGMODE:
   If LOGMODE=A or L is specified, the space is only freed when an archive is taken.

If LOGMODE=Y is specified a checkpoint frees the log space up to the beginning of the oldest LUW that was still active when the checkpoint is taken.

Checkpoints can occur for various reasons even before the CHKINTVL number of log pages has been filled, for example before and after an archive or log archive, or when an LUW has finished modifying data in a NONRECOVERABLE storage pool.

Table 2. CHKINTVL Multiple User Mode Initialization Parameter

Parameter	Default Setting	Minimum Value	Maximum Value
CHKINTVL=n	**10**	1	99 999 999

With the operator command **SET CHKINTVL newvalue** you can change the interval for taking a checkpoint without having to stop and restart the application server.

2.6.3.1 Performance Implications

A checkpoint has two performance implications:

1. It performs a high amount of I/O to DASD. It writes all the modified buffer pages and data space pages back to DASD, and updates the directory disk.
2. It holds up processing. User agents must wait until the checkpoint has finished before they can proceed.

2.6.3.2 Considerations

If you set the CHKINTVL parameter too low:

1. You increase the system overhead, because it is a very time-consuming operation. A checkpoint has a fixed amount of overhead, which is wasteful if you are taking checkpoints more often than necessary.
2. You interrupt users more often and increase their response time.

If you set the CHKINTVL parameter too high:

1. It takes longer to recover from a system failure.
2. You risk filling up the log (if you are running LOGMODE=Y).
3. You risk filling up the storage pools with shadow pages. This makes checkpoints happen.
4. More I/Os occur between checkpoints to free buffer space filled by too many shadow pages.
5. You increase the time required to complete a single checkpoint.

Not performing any checkpoints can cause performance problems in the long run - even more than are caused by checkpoints: Sooner or later you'll become short of storage. Whenever storage is needed, both data pages and shadow pages are stored to disk.

2.6.3.3 Recommendation

1. Your database should execute a checkpoint every 10 to 15 minutes.
2. The default value of 10 often is too low. Many installations find that the optimum is between **50 and 500**.
3. During high-peak periods it is recommended to set the checkpoint interval value very high and to have a tight control over the moment checkpoints are executed:

Hint

To force a checkpoint manually, `SHOW INITPARM` first, then `SET CHKINTVL 1` and after the checkpoint `SET CHKINTVL n` to the original value.

2.6.4 Termination Statistics

The statistics that are produced from the DB2 Server at shutdown when the second parameter of DSPSTATS is set to 1, provide a very convenient and consistent method of collecting information that may be used for performance evaluation and tuning. If the COUNTER values have not been reset during the operation of the DB2 Server, these values reflect the statistics during the period when the DB2 Server was active.

For a detailed description of the output and how it may be used for performance measurement and tuning, refer to:

- "COUNTER Operator Command" in Chapter 2, Measuring Performance of *DB2 Server for VSE & VM Performance Tuning Handbook* for interpretation of the counter values.
- "DB2 for VM COUNTER Operator Command" and "VMDSS COUNTER POOL Operator Command" in Chapter 4, Measuring Performance of *DB2 Server Data Spaces Support for VM/ESA* for interpretation of the counter values in a VMDSS environment.
- "COUNTER Command" in Chapter 16, Database Monitoring Tools of *SQL/DS Version 3 Release 4 Performance Guide* for additional interpretation and suggested actions.

2.7 Lock Wait Timeout

In DB2 Server for VSE & VM Version 5 Release 1, the initialization parameter `LTIMEOUT` was introduced. It enables an automatic timeout of an LUW being in LOCK WAIT state for a specified period of time.

This initialization parameter is available in both VM and VSE and can be changed with the operator command

`SET LTIMEOUT n`

. The LTIMEOUT value "n" is specified in seconds and has a range of 0 (no timeout) or 1 to 99,999 seconds, which is around 27.8 hours.

A zero LTIMEOUT value means there is no timeout, which means waiting for a lock indefinitely with the timeout capability switched off. This is the default and is the same as in previous releases.

A non-zero lock wait timeout value will cause any agents waiting for a lock to have their current LUW rolled back when the lock wait timeout period has expired. The agent will get an SQLCODE -911. A reason code will be returned to indicate whether the SQLCODE -911 was a result of a deadlock or lock wait timeout. Reason code 2 indicates a deadlock and reason code 68 is given for a lock wait timeout.

The lock wait timeout period begins when an agent requests a lock. If a user is in LOCK WAIT and the LTIMEOUT period is changed, the user will receive a timeout if they have been in LOCK WAIT longer than the new LTIMEOUT period. Their LUW will be rolled back.

2.7.1 Usage

Setting LTIMEOUT can improve response time in applications where lock contention has begun to affect performance and concurrency levels.

LTIMEOUT should be set where DB2 Server for VSE & VM is being used for Distributed Unit of Work (DUOW). It is the way to avoid global deadlocks for DUOW applications.

LTIMEOUT has been added to the list of parameters from the COUNTER command as shown in Figure 8. The LTIMEOUT line within the “COUNTER *” command shows the actual number of lock waits which have timed out, causing the LUWs to be rolled back.

The LTIMEOUT counter can also be RESET with the operator command “RESET LTIMEOUT.”

```
F7 0007 Counter values at  DATE='10-31-97'  TIME='16:05:10'
F7 0007 Calls to RDS                  RDSCALL :  100266
F7 0007 Calls to DBSS                 DBSSCALL:  5720105
F7 0007 LUWs started                  BEGINLUW:  25455
F7 0007 LUWs rolled back              ROLLBACK:  3665
F7 0007 System checkpoints taken      CHKPOINT:  7
F7 0007 Maximum locks exceeded        LOCKLMT :  0
F7 0007 Lock escalations              ESCALATE:  1
F7 0007 Waits for lock                WAITLOCK:  1196
F7 0007 Deadlocks detected            DEADLCK :  291
F7 0007 Looks in page buffer          LPAGBUFF:  5898410
F7 0007 DBSPACE page reads            PAGEREAD:  25869
F7 0007 DBSPACE page writes           PAGWRITE:  6750
F7 0007 Looks in directory buffer     LDIRBUFF:  74588
F7 0007 Directory block reads         DIRREAD :  2039
F7 0007 Directory block writes        DIRWRITE:  3343
F7 0007 Log page reads                LOGREAD :  3
F7 0007 Log page writes               LOGWRITE:  14477
F7 0007 Total DASD reads              DASDREAD:  27911
F7 0007 Total DASD writes             DASDWRIT:  24570
F7 0007 Total DASD I/O                DASDIO  :  52481
F7 0007 Lock timeouts detected        LTIMEOUT:  0
```

*Figure 8. Sample Result of the COUNTER * Command in VSE*

2.8 Increased Buffer Maximums

In SQL/DS V3R5 the maximum number of buffers in the page and directory buffer pools was substantially increased, to help keep data in memory. Here we are discussing various aspects of this enhancement.

The System/370 24-bit addressability with a maximum storage size of 16MB dictated the original buffer pool limits. These original limits restricted the performance of the database server, which can now use the 31-bit addressability with a storage size of up to 2GB.

The buffer pool limits were increased as follows:

- The number of directory buffers (.5kB) increased from 28,000 to 400,000.
- The number of page buffers (4kB) increased from 3,500 to 400,000.

If you specify both of the new maximums, you need approximately 1.8GB of virtual storage just for buffers.

In addition, improvements were made to the hashing algorithms used when searching for a page in the buffers. These are now scaled to the number of buffers, to prevent the search times from increasing proportionally to the number of buffers defined.

Remember that increasing the number of buffers does not guarantee increased performance by a reduced number of I/Os. You must measure and tune your buffers to see if performance improvements can be realized.

The improvements you may see are dependent on many variables. The following is not a complete list:

- The amount of main storage that is available to back up the increased number of buffers.

  Having not enough main storage leads to buffer pages being paged out. Both changed pages and shadow pages might be paged out. This means additional I/O without reducing the amount of checkpoint I/O. And, remember that page I/O can not be executed overlapped as opposed to file I/O. Therefore, you need to have enough main storage to back up the virtual storage defined. A recommended factor virtual/real in VSE/ESA is 3. For additional virtual storage to keep more buffers the factor should be no more than 2. This means, the additional virtual storage should be backed up by at least half this amount of additional real storage.

- The type and mix of workloads regarding page reuse.

  High reuse of pages reduces the number of I/Os. If you use mostly different pages, a large buffer does not help much.

- The ratio of update (read-write, R/W) to read-only (R/O) activity.

  High update activity causes you to have more checkpoints. If you have a high percentage of reused pages for update, increasing CHKINTVL can help. If there is not much reuse, increasing CHKINTVL does not help but makes each checkpoint last longer.

  Independent from CHKINTVL, if your read-only data is frequently re-used, having more buffers to keep it in may reduce your I/Os.

- The frequency of sorts and joins and their data sizes.

Large sorts or joins being performed as “merged scan” might easily fill the larger buffers and use them for the benefit of just one single user; nested loop joins do not use as much buffer space so that other users can also profit from the larger buffers.

If the majority of your sorts fit in the buffers, you will reduce the amount of I/O needed to internal dbspaces; if not, the sort will flush the buffers of other data and thus **not** reduce the number of I/Os.

2.8.1 Positive and Negative Effects

You must have sufficient main storage available to back up the virtual storage allocated to the buffers, to prevent an increase in database server paging. You must remember that a page fault stops the execution of the entire server partition or virtual machine.

Generally, performance improvements will be realized when you increase your number of buffers **if** your buffer hit ratios improve. See 2.8.2, “Measurement and Tuning” on page 19 to learn how you calculate your hit ratios. An improved hit ratio means that your data is being reused in the buffer more often and is read in from DASD less often, thus saving I/Os.

However, a large increase in the number of buffers might require more CPU time to maintain the buffer control blocks. This is the downside of applying data in memory. But remember that saving I/Os also means saving CPU time. Therefore, such an increase in CPU time might be acceptable **if** you really save I/Os. Alternately, if you do **not** save I/Os, your server performance might degrade due to an extra CPU cost of buffer control block maintenance. Increasing the number of buffers by a very large amount (for example, by two orders of magnitude) will noticeably increase the time required for the server to be initialized.

2.8.1.1 System Checkpoints

Between checkpoints, modified pages in the buffers are not written back to DASD unless the buffer is needed for a different page. Then the modified buffer is first written out and a different page is read into the buffer. All modified buffers **must** be written to DASD during a checkpoint. A very large number of buffers could increase the time for a checkpoint to complete, if a larger number of the buffers contain modified pages.

Increasing the number of buffers will not change the actual number of pages that must be saved on DASD, but, in some cases, it may actually decrease the number of I/Os required. Consider when a specific page is updated twice within a short period of time:

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

However, a larger number of buffers may cause many or **all** writes to be delayed until the next checkpoint, causing an increase in the elapsed time for the checkpoint, due to the larger number of writes that must occur while all users are waiting. While the users work may occur slightly faster (if there is less wait time for I/O completion because fewer buffers are written out between checkpoints), this possible performance improvement is very unlikely to be

noticeable to most users. Alternatively, many users may notice the increase in checkpoint elapsed time.

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

If you use VMDSS, you can use the SAVEINTV parameter to help reduce the cost of checkpoints. Setting the SAVEINTV value such that a few save intervals expire between checkpoints will reduce the amount of I/Os required when the next checkpoint occurs. See the VMDSS documentation for more details about SAVEINTV and checkpoint processing.

Again, it may be that the general performance improvement given by the increased number of buffers will outweigh this increase in checkpoint overhead. The new **DSPSTATS** initialization parameter can be useful for monitoring the frequency and cost of checkpoints (see 2.6.4, “Termination Statistics” on page 15).

2.8.1.2 Page Buffers versus Directory Buffers

Accessing a data page usually requires accessing a directory block as well. A large increase in the number of page buffers without some increase in the number of directory buffers may cause a decrease in the directory buffer hit ratio, indicating that the directory buffers have become a bottleneck. You may need to increase the number of directory buffers when you increase the number of page buffers. As usual, you must periodically examine the hit ratios of both types of buffers.

2.8.2 Measurement and Tuning

You measure and tune page and directory buffer pool sizes in the same way:

1. Measure the *looks* and the *reads* with the DB2 Server for VSE & VM `COUNTER` command (or, if you are using VMDSS, with the `COUNTER POOL` command).
2. Calculate the hit ratio using LPAGBUFF (the number of requests) and PAGEREAD (the number of misses). Divide the hits (LPAGBUFF - PAGEREAD) by the number of looks (LPAGBUFF):

   `(LPAGBUFF - PAGEREAD) / LPAGBUFF`
3. Increase the number of buffers.
4. After a comparable workload has been run, remeasure the looks and reads and recalculate the hit ratio.

For examples of the `COUNTER` and `COUNTER POOL` output, refer to Figure 8 on page 16 and Figure 9 on page 21.

As an example assume you have the following `COUNTER *` values:

```
Looks in the Page Buffers:          LPAGBUFF = 14876  (Requests)
DASD Page Reads:                    PAGEREAD =  1231  (Misses)

Looks minus reads: (LPAGBUFF - PAGEREAD )    = 13654  (Hits)

Hit Ratio: (Hits / Requests) = 13654 / 14876 = 0.917 or 91.7%
```

As a rule of thumb, anything below 90% is poor, and more buffers should be used.

Your hit ratio has improved if it has increased in value. You will be using less I/Os to do the same processing. You should repeat this process until your hit ratio no longer increases or until your system paging begins to increase.

2.8.2.1 Method 1

You should increase your number of buffers by a small amount at a time and monitor the change in hit ratio, repeating this until you find your peak hit ratio. If you increase your buffers by a large number, you may exceed the optimal number of buffers without knowing it. This is because you may see an increase in the hit ratio, but you will not know if a smaller number of buffers would give you the same increase in hit ratio. To be sure, you would need to start *decreasing* the number of buffers to see if the hit ratio decreases. If it does **not** decrease, you may still have more than the optimal number of buffers. If it does decrease, then you have less than the optimal number of buffers. It is better to have slightly more than the optimal number of buffers than slightly less and by increasing the number of buffers by small increments, you will probably be able to determine the optimal number of buffers more quickly.

2.8.2.2 Method 2

An alternative is the following, knowing that more buffers will almost always reduce database I/O:

1. Work out how much storage you can spend without hitting paging problems.
2. Give all that space to the database.
3. Define 500 directory buffers and give all the remaining space to page buffers. NPAGBUF should be at least 1000 to take advantage of blocking (Multi-Block *BLOCKIO in VM or Extended user buffering in VSE).
4. Monitor the COUNTER values for DASD I/O and calculate the buffer hit ratios.
5. Increase the directory buffers decreasing the page buffers until you get the best results, monitoring as described in step 4. Add eight directory buffers for every page buffer removed.
6. Decrease the page buffers until you notice that the results get worse. Thus you determine how much buffer space you really exploit.

```
Counter values at  DATE='10-27-97'  TIME='15:58:06'

Directory:  *BLOCKIO
Pages looked at in the buffer  LBUFLOOK:  750
Page reads                     PGREAD  :  825
Page writes                    PGWRITE :  199
IUCV *BLOCKIO I/O requests     IUCVBIO :  1024

Unmapped Internal Dbspaces:  Data Spaces
Pages looked at in the buffer  LBUFLOOK:  18
Pages moved from DS to buffer  DSREAD  :  0
Pages moved from buffer to DS  DSWRITE :  0
DS page fault notifications    DSFAULT :  0

Pool No. 1:  *BLOCKIO
Pages looked at in the buffer  LBUFLOOK:  2198
Page reads                     PGREAD  :  149
Page writes                    PGWRITE :  0
IUCV *BLOCKIO I/O requests     IUCVBIO :  149

Pool No. 2:  Data Spaces
Pages looked at in the buffer  LBUFLOOK:  0
Pages moved from DS to buffer  DSREAD  :  0
Pages moved from buffer to DS  DSWRITE :  0
DS page fault notifications    DSFAULT :  0

Pool No. 3:  *BLOCKIO
Pages looked at in the buffer  LBUFLOOK:  950
Page reads                     PGREAD  :  14
Page writes                    PGWRITE :  0
IUCV *BLOCKIO I/O requests     IUCVBIO :  14

   ...

Pool No. 15:  *BLOCKIO
Pages looked at in the buffer  LBUFLOOK:  0
Page reads                     PGREAD  :  0
Page writes                    PGWRITE :  0
IUCV *BLOCKIO I/O requests     IUCVBIO :  0

Total count for storage pools using data spaces:
Pages looked at in the buffer  LBUFLOOK:  0
Pages moved from DS to buffer  DSREAD  :  0
Pages moved from buffer to DS  DSWRITE :  0
DS page fault notifications    DSFAULT :  0

Total count for storage pools using *BLOCKIO:
Pages looked at in the buffer  LBUFLOOK:  3165
Page reads                     PGREAD  :  165
Page writes                    PGWRITE :  0
IUCV *BLOCKIO I/O requests     IUCVBIO :  165
ARIO065I Operator command processing is complete.
```

*Figure 9. Sample Result of COUNTER POOL * Command in VM*

Chapter 3. Disks and Caching

This chapter describes the external influence on performance by the operating system and the I/O subsystem.

3.1 Minidisk Placement on RAMAC and Multiprise Internal Disk

Previously, before the RAMAC family of DASD was available, under VM the placement of minidisks (and in VSE, the placement of VSAM clusters) on the physical DASD, if uncached, had some influence on data access time, considering the different DASD head movement times. With the RAMAC Array family this does no longer necessarily have any controllable influence.

The RAMAC Array Models 1, 2 and 3 consist of drawers. A drawer consists of actually four hard disk drives. Such a drawer can be defined as an emulated DASD, typically a 3380 or 3390 DASD. The data is stored on all four drives, striped by microcode.

On a RAMAC Array Subsystem, the physical drives are connected to logical volumes, but because of the flexible size of 3390-9 emulations it is not possible to detect where on a drive the logical volume resides.

RAMAC Virtual Array (RVA) uses data compression, and the logical volumes are not at all connected to physical drives; striping takes place controlled by microcode.

Therefore, on the whole RAMAC family no minidisk placement optimization can be done.

On a Multiprise Internal Disk, up to eight drives emulate one 3380 or 3390 logical volume. No disk space is reserved for internal code. A minidisk placement optimization is possible and necessary, considering the mapping of logical volumes to physical drives.

Summary: With Multiprise Internal Disk a minidisk placement optimization is still possible and needed, but not with the RAMAC family. In any case, the use of **small** logical volumes is recommended so the operating system can start as many I/Os in parallel as possible.

Also important is the caching behavior of these disks, described in 3.2, “Caching.”

3.2 Caching

While we will be concentrating here on the caching of data in DB2 Server for VSE & VM, remember that caching does not *only* apply to DB2 Server for VSE & VM data, but to all data on your system.

As you probably know, caching can yield significant performance improvements, when used and configured appropriately.

3.2.1 Where Data can be Cached

Caching can occur in several places:

- In DASD Controllers:

  Caching is controlled by the DASD Controller, normally in units of a track. With newer I/O subsystems the operating system has less and less control over what does or does not get cached, but the controller itself has its own LRU and adaptive algorithms.

- In Expanded Storage:

  Under VM, expanded storage can be used for caching both minidisk and system paging data. Minidisk caching (MDC) is controlled by VM at the volume or minidisk level, using CP commands and Directory statements. System paging data caching is controlled by CP commands and by CP itself. VSE does not use expanded storage.

- In Main Storage:

  Under VM, main storage can be used for the same caching as expanded storage: for minidisk and virtual storage data. Under VSE, main storage is used for caching virtual storage. Note that DB2 Server for VSE & VM caches data in local buffers, which is virtual storage from the operating system point of view.

Virtual disks are treated by both VM and VSE as a kind of virtual storage.

3.2.2 What Data Should or Should Not be Cached

The very best use of cache is to hold data which is often read and seldom or never written. This is fairly obvious, but there is another requirement your data must meet for effective cache use. This is the size of the data being cached. If the size of the data being used exceeds the size of the cache, you will flood the cache and will not use it effectively. When flooding occurs, the data that you want to re-use will not be there, having been pushed out to make room for other data. Of course, this applies to all types of cache.

Consider DASD Controller cache: it is usually safe to assume that VM paging and spooling data is cache unfriendly, that is exceeds this cache size, and this kind of data should be excluded from Controller caching. However, this may not always be true. With the very large main storage sizes of modern processors you may have a low enough paging rate so that caching page I/Os can be beneficial.

The same data size considerations apply to minidisk caching. Large minidisks containing data that is always used sequentially may flood the minidisk cache, and should be excluded. Remember that the VM default is to cache *all* minidisks, including minidisks of VSE Guest systems.

Of course, when we speak of data size, we do not mean the size of the minidisk. It is the size of the data that is *actually* used frequently.

This is why we recommend caching the CMS ″S″ and ″Y″ disks and similar high usage product and (local) tools minidisks/libraries/files, which are normally read only *and* are “frequently re-used.” While the minidisks may be quite large (hundreds of megabytes are not unusual), the size of the frequently used data is usually much smaller.

3.2.3 Measuring Cache Effectiveness

The only way to tell how effectively you are using your cache is to monitor the cache "hit ratio." A hit ratio is defined as the number of times data was actually found in the cache divided by the number of times data was requested from the cache (that is, *hits* divided by *requests*). Using this calculation, the closer the hit ratio is to 100%, the better is your cache effectiveness. If data requested is never in the cache, this ratio will be zero: your cache is giving you no benefit.

If you have a 3990 model 3 or 6 you can issue the CACHE command for each of your database volumes. The following is the result of this command in a customer installation:

```
CACHE UNIT=92A,REPORT
AR 0015 3990 SUBSYSTEM COUNTERS REPORT
AR 0015 VOLUME 'PROD2A' DEVICE ID=X'2A'
AR 0015                              CHANNEL OPERATIONS
AR 0015                 <----SEARCH/READ----> <-------------WRITE------------>
AR 0015                    TOTAL   CACHE-READ    TOTAL  CACHE-WRITE DASD-FAST
AR 0015 REQUESTS
AR 0015   NORMAL         206320814  194282721  108208673  108200666  108208673
AR 0015   SEQUENTIAL       1737801    1487021      39533      39533      39533
AR 0015   CACHE FAST WR          0          0          0          0        N/A
AR 0015
AR 0015 TOTALS           208058615  195769742  108248206  108240199  108248206
```

Figure 10. Output of 3990-6 CACHE Command

In order to calcule the hit ratios, use the following formulas:

Read hit ratio:
Cache-read totals/total totals = 195769742/208058615 = 94,09 %

Write hit ratio
Cache-write totals/total totals = 108240199/108248206 = 99,99 %

3.3 DB2 Server for VSE & VM Specific Caching Information

The DB2 Server for VSE & VM Log extents are almost always write only and they should *not* be cached anywhere, to reserve your cache for more appropriate data.

The DB2 Server for VSE & VM Directory extent will usually be a prime candidate for caching, as it usually sustains many more reads than writes. This is *not* always the case, especially if you have a large number of Directory Buffers defined *and* a large amount of data updating (including high internal dbspace usage). This can be determined by calculating your Directory Buffer Hit Ratio, from the DB2 Server for VSE & VM COUNTER * operator command.

A dbextent which is read only (or very infrequently updated) could fall in this category, as long as it has a high enough frequency of use and does not flood the cache.

The data size and usage of many dbextents will simply be too large for caching and should be excluded.

3.3.1 Caching Recommendations

Below are our recommendations and warnings about caching specific to DB2 Server for VSE & VM Version 5 Release 1, listed by type of cache. Naturally, all of these recommendations are subject to your data size considerations concerning cache flooding.

3.3.1.1 DASD Controller Caching

- In the following, we assume that you are using VSE/ESA Version 2 Release 1.2 & higher, VM/ESA Version 2 Release 1 & higher and the latest DASD Controller Licensed Internal Code (LIC, or microcode).
- The following does not always apply to all DASD Controllers, depending on type, model and features. It generally applies to later models of 3390 controllers and all RAMAC controllers, but you must check your particular system hardware and its setup.
- Both VSE/ESA and VM/ESA provide commands used to control the DASD controller cache at the volume level. In addition, VM/ESA provides control at the minidisk level, via the `MINIOPT` VM Directory statement. Refer to the appropriate operating system documentation.

  Warning: VM/ESA defaults to caching *all* minidisks (including second level VSE Guest volumes), which will frequently cause cache flooding.

- Basic Caching - Reads and Writes: This is available on all DASD controllers that have cache storage available. You can use this for DB2 Server for VSE & VM data.

  1. For DB2 Server for VSE Archives (including Log Archives), VSAM sets the sequential access flags, which will frequently cause the DASD controller to pre-load data into the cache before VSAM requests it.
  2. For non Archive DB2 Server for VSE I/Os, VSAM sets the Regular Data Format (RDF) and Record caching flags, which prevents the DASD controller from doing *rest of track* staging into the cache. This is more appropriate for the random I/O characteristics of DB2 Server for VSE.
  3. DB2 Server for VM uses the **BLOCKIO* system service of VM/ESA. When doing Database and Log Archive reads and some dbextent writes, DB2 Server for VM uses blocked *BLOCKIO requests, which can significantly speed up these I/Os. Some DASD controller adaptive caching algorithms may further improve these read I/Os, because VM/ESA always sets on the Regular Data Format (RDF) bit in the DEFINE EXTENT CCW.

- Extended Function Fast Write Caching:

  1. DASD Fast Write - This feature is available with DASD Controllers that have Non Volatile Storage (NVS). It allows the Controller to present Device End (DE) as soon as the data has been written into NVS, before the data has been destaged onto DASD. This makes the writes very fast. The data will be destaged to DASD later, when the Controller determines that the NVS is needed for more recent data. This is safe for DB2 Server for VSE & VM data because your data will remain in NVS for up to 48 hours, if a power failure occurs. As long as power is restored before battery failure occurs, your data will be destaged from NVS to DASD. DASD Fast Write is the hardware default for 3990 and RAMAC Controllers which have NVS and is applicable to both VSE/ESA and VM/ESA.

Warning: Battery failure before power is restored, a Special Subsystem IML, a VSE `CACHE SUBSYS=cuu,REINIT` command or the VM `SET DASDFW DEVICE FORCEOFF` and `DISCARD PINNED` commands *will* cause NVS data loss, *requiring* a database restore. Also, the VM `DESTAGE SUBSYSTEM` or `SET NVS OFF` commands can be used to force the destaging of NVS cached data.

2. Cache Fast Write - This feature is only applicable to temporary work files, which could only apply to DB2 Server for VSE & VM internal dbspaces. However, there is no VSE/VSAM or VM/ESA *BLOCKIO support for requesting the use of this feature, so it cannot be used.

3. Dual Copy - This is the automatic updating of a secondary physical volume when writes are made to the primary volume. If either volume suffers physical damage, the alternate volume will continue operations. When the damaged volume is repaired, the DASD controller will automatically bring the repaired volume up to date, as compared to the undamaged volume. Remember that Dual Copy is done at the volume level and, for it to be effective at preventing outages, *all* dbextents within a DB2 Server for VSE & VM storage pool must be on dualed volumes. Using Dual Copy just for the log disk volume instead of `DUAL LOG` might improve the performance. Refer to the operating system documentation for usage information.

- RAMAC Virtual Array Notes:

  1. The over commitment of virtual to real DASD space does not apply to DB2 Server for VSE & VM dbextents. This is because all dbextents (including the Log and the Directory) are pre-formatted (via VSAM or CMS FORMAT).

  2. The *Snapshot* capability is not supported by VSE/ESA and has limited VM/ESA support. Problems arise due to the duplicate VOLIDs that a snapshot produces and there are no facilities for backing up or restoring a snapshot (for example, for DB2 Server for VSE & VM User Archives).

3.3.1.2 Expanded Storage Caching (VM/ESA)

- VM uses expanded storage for caching in three ways:

  1. system paging (virtual storage pages)

  2. minidisk caching (MDC)

  3. dedicated to a virtual machine (for example, an OS/390 Guest)

- Minidisk caching now operates at the DASD track level and is no longer restricted to minidisks blocked at 4K. This means that virtually *all* minidisks can now be cached, including second level system minidisks (for example, of a VSE Guest) and the DB2 Server for VSE & VM Directory extent.

- You can control the use of expanded storage by using the CP RETAIN XSTORE command.

- You should not use MDC for the DB2 Server for VSE & VM Log extent, as this is mostly write only.

  Warning: VM defaults to minidisk caching *all* minidisks, which may cause cache flooding.

- You can use MDC for the DB2 Server for VSE & VM Directory extent and any dbextent, except when those extents are accessed via VMDSS and data spaces. All DB2 Server for VSE & VM minidisks accessed via data spaces should have MINIOPT NOMDC statements in the VM Directory. This is so you can eliminate the checking that CP must do to flush any MDC resident pages that may exist, when the same DASD page is updated via data spaces. As DB2 Server for VSE & VM never accesses the same page via data spaces and via *BLOCKIO simultaneously, this data integrity checking just wastes CPU time.
- You should treat all dbextents in a storage pool as a single unit, either caching all or none of the dbextents.
- Your DB2 Server for VM data, accessed by VMDSS, may also be cached in expanded storage, when CP caches data space pages in expanded storage instead of paging it to DASD.

  Note: VSE/ESA does not support expanded storage.

3.3.1.3 Main Storage Caching

- Main storage is the final level of caching, as data must be in main storage to be processed.
- VM/ESA now uses main storage, as well as expanded storage, for minidisk caching. This means that most of the minidisk caching comments in the previous Expanded Storage section also apply here. Of course, you have less control over how CP uses main storage for caching, compared to expanded storage.
- Main storage is also used for caching data space pages (which includes virtual disks). In VM, this is controlled by CP.
- Under VM/ESA any pageable data, such as “normal” virtual storage, data space storage and virtual disk storage, may be cached by CP in expanded storage.
- Finally, DB2 Server for VSE & VM local buffers can be considered as being cached in main storage, when they are not paged out to expanded storage or DASD by the operating system.

3.3.1.4 Miscellaneous Caching Notes

- It is usually better to use VMDSS with your DB2 Server for VM database and reserve your minidisk cache for other data. This is because the VM paging subsystem is more efficient, asynchronous and flexible than the CP *BLOCKIO system service. You can usually let CP control how much expanded storage is used for paging and how much is used for minidisk caching. You must also remember that VMDSS needs a relatively large amount of main storage to be effective. On the other hand, if you have lots of “spare” main storage, do not artificially restrict VMDSS by setting the TARGETWS parameter value too low.
- When running VSE under VM, it is usually more efficient to use a VSE virtual disk, instead of a VM virtual disk. However, a VM virtual disk can be shared between multiple second level VSE systems and will survive a VSE IPL. Remember, you must create and initialize a VM virtual disk *before* you IPL the VSE system.
- A few dbextents is usually better than one large dbextent because you have more control over dbextent placement (where possible) and easier relocation. Do not take this to an extreme, because the more dbextents you actively use, the more virtual storage and CPU overhead there is.

- Minidisk Cache changes in VM/ESA V1.2.2.

  Prior to version 1.2.2, VM/ESA minidisk caching was restricted to using only expanded storage and would only cache CMS minidisks formatted with a block size of 4K.

  As of version 1.2.2, these restriction have been lifted. Specifically, *any* minidisk, regardless of its block size or track format, can be cached. Note that this includes MVS or VSE minidisks.

  VM/ESA may now use main, as well as expanded, storage for minidisk caching. Also, minidisk caching now caches minidisk data on a track basis, instead of the previous 4K block basis. This provides some prefetching capabilities.

  As data is not stored on consecutive pages, there can be a slight overhead, to read a complete track instead of pages.

  APAR VM61045 was introduced, to have the possibility in VM/ESA V2.1.0. or up to specify if record based caching should be used instead of the default track based caching.

Part 2. Tools

This part covers the tools we used in the ITSO during the residencies writing this redbook. Control Center feature, and the vendor program SQL-Tune comprise both platforms, VSE and VM. VM:DB/Monitor is available for VM only.

Chapter 4. Control Center Feature

This chapter gives an overview on the different tools available in Control Center and a summary of the advantages and benefits that can be expected by implementing Control Center in VSE or VM.

IBM Control Center for VSE & VM is a charged feature of DB2 Server for VSE & VM, which works with the databases to automate many DBA functions. Before DB2 Version 5 Release 1.0, Control Center was offered as a separate program ("IBM SQL Master for VSE & VM," 5684-136).

4.1 Advantages of Using Control Center

The advantages of using Control Center are listed below. Control Center:

- Simplifies the task of supporting databases.
- Enables you to automate complex steps of DBA activities.

  This eliminates the need for manual support during a database event.

  The automated feature performs all these activities from start to completion.
- Allows to perform each function in a repeatable manner.
- Allows to initiate each function immediately or to schedule it for periods of low system usage.
- Offers a high degree of security and control.
- Improves the productivity of the entire system.
- Reduces the amount of support time.
- Allows to integrate new databases easily.
- Minimizes the workload impact of the administration of additional databases.

4.2 Control Center for VSE

Control Center for VSE is a program which helps automate many of the DBA tasks that normally would be carried out manually, such as dbspace backup, migration, reorganization, and analysis. It generates the complete set of Data Definition Language required to redefine any dbspace and its objects. It also keeps the catalog statistics updated.

4.2.1 Description

Control Center for VSE covers the following functions:

1. Operator commands

   Control Center lets you issue any SHOW or COUNTER command easily without having to use ISQL or be connected to a VSE console.

2. Dbspace reorganization

   This function offers four main functions:

   - DDL generation
   - Unload dbspace
   - Reload dbspace

- Reorganize dbspace

3. Dbspace analysis

   This tool evaluates your databases using built-in expertise and builds an on-line list of dbspaces that require maintenance. From this list you can select which dbspaces you want to apply maintenance on, and the utility will build and submit the appropriate batch job.

4. Work file label definition

   You can define your tape and SAM work area files where the dbspaces are going to be during the process.

5. CICS Report Controller

   This is a quick access to the jobs and reports that have been submitted. With this CICS facility, you can release, hold and or delete batch jobs, and browse, print or delete reports.

6. Help facility

   A menu lists a number of subjects about which the user can read detailed information. The subjects presented cover currently supported functions and also include text on individual panels within those functions. In addition, there may be other subjects present for viewing which may deal with hints, tips, problem circumvention, PTFs, and other topics deemed worthwhile for the user.

7. Package utility

   You can UNLOAD, RELOAD, rebind, or view the packages stored in your database. You can also migrate packages from one database to another.

8. Group authorization

   You can define groups of objects and groups of users and give them authorizations. This simplifies the management of controlling the access to database objects.

9. Monitor utility

   This utility records the database activity such as locking or log full and provides notification when the threshold you define has been exceeded. Monitor information is stored in tables that you can view on-line or print in a batch report.

4.2.1.1 Control Center Tools

The performance of a system seen as a whole can be affected by many factors, one of them is the database performance, and it is here where Control Center can be of great help.

The tools that a DBA can use for this purpose are:

1. Database monitoring
2. Dbspace analysis
3. Dbspace reorganization

4.2.2 Database Monitor Tool

This tool can be used when you are interested in tracking:

- User activity
- Locking
- Physical and logical space usage
- Database status

The following monitors can help a DBA to figure out what is happening in the database system:

- SHOW ACTIVE

  Monitors active database users. This monitor will optionally send a message to the VSE console when it detects an active checkpoint agent or a user agent in checkpoint, communication, or lock wait.

- SHOW LOCK

  Monitors database lock contention. In addition, this monitor will optionally send a message whenever it detects a lock holder not processing, locking due to a checkpoint, or any lock contention at all.

  With this information, you can determine which users are holding resources for long periods of time. Then you must find out the applications and improve them to free the resources as soon as possible.

- SHOW CONNECT

  Monitors users connected to the database. Optionally, this monitor will send a message to the VSE console whenever it detects an active user not processing or an inactive user.

- COUNTER *

  Monitors the occurrence of key events in the database.

Control Center for VSE provides also the following monitors which can be of great help, especially if thresholds are specified for them:

- SHOW DBEXTENT
- SHOW LOG
- SHOW DBSPACE

Note: For more information about these monitors, consult the *Control Center for VSE Installation and Operations Guide*.

Thresholds are specified when any of the monitors listed above are selected with Control Center for VSE, and for this purpose you will get the screen shown in Figure 11 on page 36.

```
     10/22/1997               CONTROL CENTER V5.1                    15:02:55
*-------------------------     MONITOR THRESHOLDS     -------------------------*
|                                                                              |
| DATABASE     => SQLVSE01                          MONITOR     => ACTIVE      |
|                                                                              |
| ***************************     SHOW ACTIVE     *************************** |
| CHECKPOINT WAIT                        => 1 (1=YES/2=NO)                     |
| USER WAIT                              => 1 (1=YES/2=NO)                     |
|                                                                              |
| ***************************    SHOW CONNECT     *************************** |
| AGENT NOT PROCESSING                   => 1 (1=YES/2=NO)                     |
| INACTIVE                               => 1 (1=YES/2=NO)                     |
|                                                                              |
| *****************************    SHOW LOCK    ***************************** |
| CHECKPOINT                             => 1 (1=YES/2=NO)                     |
| ANY LOCKING                            => 1 (1=YES/2=NO)                     |
|                                                                              |
| ***************         SHOW LOG AND SHOW DBEXTENT          *************** |
| PERCENT USED                           =>    %                               |
|                                                                              |
*------------------------------------------------------------------- SQC42 ----*

PRESS ENTER TO PROCESS
ENTER F1=HELP F3=EXIT F12=CANCEL
```

Figure 11. Control Center for VSE Monitor Thresholds Screen

The following is an example of the message sent to the operator console when a threshold is reached.

```
F4 0004 SQM0632 SQLMSTR  IS IN COMMUNICATION WAIT
F4 0004 SQM0318 DATABASE:      SQLVSE01
F4 0004 SQM0632 PRODCICS IS IN COMMUNICATION WAIT
F4 0004 SQM0318 DATABASE:      SQLVSE01
F4 0004 SQM0632 PRODCICS IS IN COMMUNICATION WAIT
F4 0004 SQM0318 DATABASE:      SQLVSE01
```

Figure 12. Control Center for VSE Threshold Message Screen

The general steps to define and activate a monitor are as follows:

1. From the Control Center for VSE main menu, choose the Monitor Utility option, specifying in the DATABASE field the database name to which you are defining the monitor.
2. Add the Monitor by:
   - Choosing option 4: Add Monitor
   - Supplying the monitor number you are adding as follows:

```
01 SHOW ACTIVE
02 SHOW LOCK
03 SHOW DBEXTENT
04 SHOW LOG
05 SHOW CONNECT
06 SHOW DBSPACE
07 COUNTER *
```

Figure 13. Control Center for VSE - Monitor Numbers

- Completing the information for the following variables shown in Figure 14 on page 37.
- Supplying the monitor thresholds using the screen shown in Figure 11 on page 36

3. Initiate the monitor by starting the Kernel using the option 1 from the Monitor Utility screen

```
DESCRIPTION       =>
ACTIVE?           => 2                    (1=YES/2=NO)
                     1 2 3 4 5 6 7
RUN DAYS          => 2 1 1 1 1 1 2        (1=YES/2=NO)
START TIME        => 0001                 (HHMM)
STOP TIME         => 2359                 (HHMM)
INTERVAL          => 0015                 (HHMM)
RESET DATA?       => 2                    (1=YES/2=NO)
RESET DAY         => 1                    (1-7)
PRINT REPORT?     => 2                    (1=YES/2=NO)

**********************      REPORT PARAMETERS      ****************

REPORT NAME       => MONPRNT              CLASS     => A
PRI               => 3                    DISP      => D (D,H,L,K)
```

Figure 14. Control Center for VSE Monitor Control Screen

4.2.3 Dbspace Analysis Tool

This tool allows a DBA to supply some selection options and criteria to determine **which dbspaces are candidates to receive maintenance**.

The DBA can select the dbspaces using the following parameters:

- Dbspace owner

  This is the name of the dbspace owner. Valid values are: ALL, PUBLIC, PRIVATE, a specific dbspace owner (such as CUST1), or a wildcard such as %CUST%. OWNER defaults to ALL.

- Dbspace name

  The name of the dbspace. Valid values are a specific name such as SAMPLE, or a wildcard such as %DB%.

- MIN(PAGES), MAX(PAGES)

  Specifies the minimum and maximum dbspace sizes. Used to limit the dbspaces selected for analysis. MINIMUM defaults to 0. MAXIMUM defaults to 5,146,999.

- DAYS SINCE

  Specifies the number of days since the last update. A value of 14 will select dbspaces which have not been reorganized or have not run UPDATE STATISTICS within the previous two weeks. A value of 0 would cause all dbspaces to be selected.

The DBA can specify additional criteria to be more selective about the dbspaces which need reorganization. For this purpose you can supply one of the following criteria:

1. CLUSTER RATIO < nnnn

   The dbspace is chosen as a candidate if the CLUSTERRATIO of any index in the dbspace is below the value specified. Valid values are 1 to 5,146.

2. UNCLUSTERED INDEX

   The dbspace will be selected as a candidate if it contains any indexes that are not clustered.

3. NOVERFLOW > nn%

   The dbspace is chosen as a candidate if the number of rows that have overflowed from their original page in storage to another page exceeds the specified percentage.

4. APPLY ALL CRITERIA

   A dbspace is selected as a candidate if it meets any one of the three reorganization criteria stated above.

4.2.4 Dbspace Reorganization

You can use this tool when you need to work with a **particular dbspace** in order to:

- Backup the dbspace to:
  - Make a security copy of the dbspace
  - Move it to another volume due to growth in data
  - Distribute dbspaces among different volumes according to their DASD usage
- Restore the dbspace to:
  - Restore a security backup
  - Restore it on a different volume
- Reorganize the dbspace
- Know the characteristics of a table and its related objects: columns, indexes, packages, grants and so on.

  This feature is very useful, because you get a complete description of the dbspace, so you need not have huge libraries to keep all the information related to dbspaces.

During the reorganization process, this tool gathers system catalog information about the specified dbspace and creates the DDL required to:

- Create the tables in the dbspace
- Add table and column comments
- Reload the tables
- Define referential integrity
- Create unique columns definitions
- Create indexes
- Set table grants
- Define views, grants, comments and labels
- Rebind packages

4.3 Control Center for VM

Control Center interacts with the databases to automate many DBA functions such as:

- Archive and Recovery
- Adding and deleting dbextents
- Adding dbspaces
- Startup and shutdown of a database
- Changing startup parameters
- Dbspace-level reorganization and redefinition
- Table-level reorganization and redefinition
- Catalog index reorganization and redefinition
- Database monitoring

It simplifies the tasks of supporting databases by automating the complex steps required to perform many DBA activities. These functions can be scheduled or performed immediately.

Hint: The major difference to Control Center for VSE is that Control Center for VM additionally comprises the functionality of archiving, recovery and also supports the Data Restore Feature.

4.3.1 Environment

Virtual Machines

- Code Owner
  Control Center code can be installed from within any user ID on a minidisk to which this user ID has write access. The owner of the minidisk will be referred to as *Code Owner*.
- Service Machine
  The user ID that is defined as secondary console of the database virtual machine and handling all messages from and commands to the database will be referred to as *Service Machine*. In most cases there will only be one Service Machine in each system. If databases are available on different VM systems, there can be a Service Machine on each system and these Service Machines communicate to each other by means of sending files to each other. This gives you the possibility to maintain databases on different VM systems from one user ID on one of the systems.
- Support Machine
  The user ID that will execute scheduled jobs against any database to free the Service Machine will be referred to as *Support Machine*. More Support Machines can be defined on each VM system, depending on the quantity of jobs to be executed and on the number of databases on this system.
- Data Restore Machine
  This special support machine is optional and needs to be defined if the Data Restore Feature is installed and needs to be controlled by the Service Machine. The Data Restore Machine can support more than one database on a VM system.

Minidisk

- Code Disk
  The minidisk keeping the Control Center code and parameter files needed by users will be called *Code Disk*. It belongs to the Code Owner Machine. Each user of Control Center, administrator, end-user or operator will need to link and access this code disk. The level of authority is defined by the administrator of Control Center.
- Work Disk
  The private minidisk owned by the Service Machine will be referred to as *Work Disk*. This minidisk will keep all settings of the controlled databases and also the history files keeping the console logs and output of jobs, running on this Service Machine.

4.3.2 Highlights and Capabilities

1. Access control

   Four authorization levels can be specified for controlling access.

2. Externalized database startup parameters

   All database startup parameters are maintained by Control Center and can be updated at any time by the DBA using its user interface. Modified parameters become operational during the next database startup.

3. Local and remote user interface capability

   The user interface is designed to communicate with multiple Service Machines.

4. Database operator command interface

   Provides the DBA with the capability to perform all database operator commands.

5. Group authorization tool

   This new tool assists users in managing the access to database objects. Simplifies the process of authorization.

6. Automated archiving and database recovery

   Archives are supported under all logmodes. Control Center supports multi-volume database archives.

   Recovery functions provide the DBA with all available recovery sets derived from database archive events. It enables the DBA to select which recovery set should be used.

7. Handling logmode changes

   All database logmode switching is automatically handled.

8. Database monitoring

   Many of the periodic monitoring activities performed by the DBA are automated with the database operator commands. It has a report generation capability which logs database monitor information each time a monitor runs.

9. Dbspace reorganization

   Run dbspace reorganizations with the PAUSE option.

10. Dbspace reorganization candidate selection

    Select dbspaces that need maintenance.

11. Table reorganization and re-definition

    Table-level reorganization with many options, such as dropping, recreating, moving and copying.

12. Index analysis and reorganization

    Automatically maintain indexes. This provides the options to define the scope of analysis.

13. Object search list

    This tool is used for searching and listing database objects.

14. Job scheduling

    This tool provides the capability to schedule concurrent or single-threaded processing events against one or more databases.

15. Data Restore interface

    Control Center for VM automates single and dual BACKUP and RECOVERY. All tapes are cataloged for later use. Control Center manages the translation of your archive tapes to BACKUP format and performs Data Restore dbspace UNLOAD and table RELOAD. It also manages the LISTLOG and APPLYLOG functions.

16. Year 2000 support

    All of the Control Center functions support the year 2000. This includes date processing in job scheduling. All dates are displayed with a four-digit year. In addition users may choose the NLS date and time format such as ISO, USA, EUR and JIS.

The following figure shows the main panel of Control Center for VM. It gives you a global overview of how the tools are grouped together.

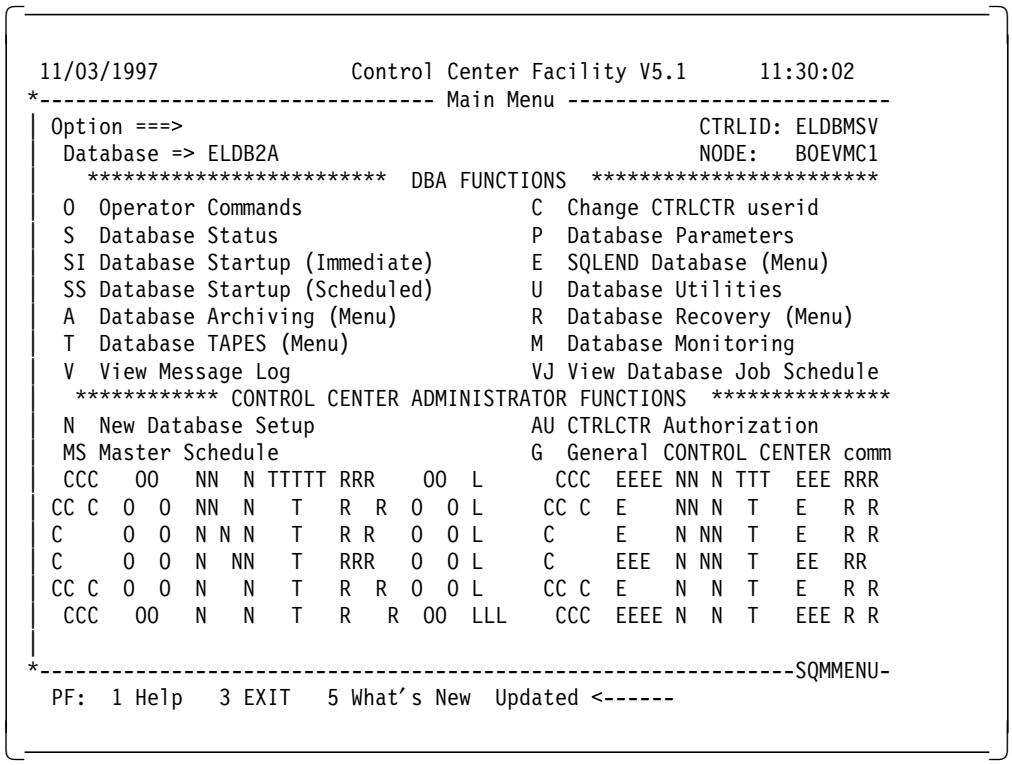


Figure 15. Control Center for VM Main Menu

4.3.3 Control Center Tools

There are two parts that group different types of tools:

1. **DBA Functions**
   - **System administration tools** work directly with the database.
   - **Database administration tools** work directly with the database.
2. **Control Center administration tools**
   - **Control Center Administration Tools** help to utilize the product as a whole.

The following table shows a summary of the Control Center tools.

Table 3. Control Center Tools Summary

Tools	Description
System administration tools	The system administration tools work with the database virtual machine console. They manage the operational needs of the database such as: • Database archiving and recovery • Starting and stopping a database • Adding, deleting and moving dbextents • Adding dbspaces • Database – Startup parameters – Virtual machine console – Monitoring – Reporting – Activities running on the database console • Interfacing the database with VM and tape manager. • Responding to each message generated on a database virtual machine console. All messages are routed from the database virtual machine to the Control Center Service Machine through the VM-provided Single Console Image Facility (SCIF). SCIF enables the Control Center Service Machine to become the database machine's virtual console and keyboard.
Database administration tools	Are database application programs. They require DBA authority. The programs connect to a database and perform queries while the database is up and running. They should be run under a Control Center Support Machine, never on a Control Center Service Machine. **The Control Center Support Machine must have DBA authority.**
Control Center administration tools	They work with the Control Center Service and Support Machines rather than with the database. These tools are invoked by the Control Center Service or Support Machine. They are accessed by the menu interface of the Control Center. Activities run with the Control Center administration tools: • Starting and stopping Control Center • Authorizing new users • Listing and reviewing control files

4.3.4 System Administration Tools

System administration tools are used to manage the operations of your database virtual machine consoles. These tools automate the activities of a database operator, logged on directly to a database. System administration tools run on the Service Machine of Control Center.

The following list summarizes the Control Center system administration tools:

- Database archiving
- Database recovery
- Database monitoring
- Database startup and termination
- Master database status

- Database operator command interface
- Single User Mode (SUM) operations

Database Archiving

All different types of archiving are supported:

- ARCHIVE, either on-line or initiated at database shutdown
- LOG ARCHIVE
- Data Restore Feature BACKUP
- Any type of USER ARCHIVE

Archives are supported for all logmodes and different output media such as tape or disk.

The information about the tapes being used for any archive is stored on the Service Machine work disk and can be displayed and modified by use of the panel interface.

Database Recovery

The Database Recovery tool automates database recovery events. The process of running a manual database restore requires stopping the database, identifying tapes and label definitions, initialization parameters modifications, tape mounts and so on. All these activities from the start to completion of the database recovery will be performed by the Database Recovery tool.

If the Data Restore Feature is installed, the Data Restore Machine will receive commands from the Service Machine to perform this recovery.

Database Monitoring

With the database monitoring tool you can specify operator commands to be executed in a specific time frame, and with a defined frequency. The monitor tool collects the results in form of reports. These reports can be analyzed to determine if any problem exists.

The following lists all types of monitors that are available:

- Special check monitor
  - Database log check monitor
  - Database up and running check
- User activity monitor
  - Users active
  - Users connected
  - User locking
- Space usage monitor
  - Dbspace usage
  - Pool usage
  - Dbextent usage
- Database counter monitor
  - Reset counters monitor
  - Display counters monitor
- VMDSS counter monitor
  - Internal monitors
  - Pool monitors

Database Startup and Termination

This utility is used to view and update the database initialization parameters as well as the VMDSS specific parameters. It also maintains the *dbname* ARISPOOL file, that defines which pool is eligible for dataspace mapping and use of striping or not.
The modified parameters will only be effective at the next startup of the database, because this is when the database requests a new copy of the initialization parameter and the VMDSS specific file.

Master Database Status

This panel provides information about the current status of all controlled databases, and gives you the possibility to start, stop or execute some operator commands against the database. Some information about date and time of the last status change and the last startup of the database is displayed.

Database Operator Command Interface

This utility provides an interface between your virtual machine and the database to perform database operator commands. It includes all SHOW, FORCE, COUNTER, RESET, TRACE commands and VMDSS specific operator commands. The advantages are:

- No real agent is being used to execute these operator commands as they are directed to the database console.
- From within one panel, you can execute operator commands directed to different databases without having to modify any setup.

Single User Mode Operations

These tools are used to invoke and manage utilities that run in single user mode (SUM):

- Add or delete dbextents
- Add dbspace
- Copy or expand the directory
- Copy or move log disks
- Copy or move data disks
- Coldlog
- Reorganize catalog indexes

Each SUM tool can be invoked to be run immediately or to be scheduled for later execution. These tools maintain individual history files recording the date, time, and other information to be able to retrace a specific modification of the database design.

4.3.5 Database Administration Tools

Database administration tools should run on a Support Machine because while these jobs are running, and some of these jobs can be running for a long time, messages from the database or other jobs can not be handled.

The database administration tools are the following:

- Dbspace reorganization
- Table reorganization and redefinition

- Automated dbspace maintenance
- Index reorganization and redefinition
- Object search and list
- Rebind package

4.3.5.1 Dbspace Reorganization - SQLREORG

This tool can perform:

- Multiple user mode dbspace reorganization
- Single user mode dbspace reorganization
- Multiple user mode UNLOAD DBSPACE
- Single user mode RELOAD DBSPACE

Dbspace Reorganization in Multiple User Mode

- SQLREORG should be used during non-peak hours to prevent locking contention with other users on the database.
- Running more than one SQLREORG against different dbspaces simultaneously within a database should not be done due to catalog contention.
- SQLREORG should be used whenever database statistics indicate that a dbspace needs to be reorganized.
- It should also be used when a large dbspace has to be moved to another storage pool with less I/O contention.

Dbspace Reorganization in Single User Mode

The SQLREORG utility provides a possibility to run both the UNLOAD and RELOAD process of a dbspace reorganization in single user mode (SUM).*:* Performing the reorganization in SUM, avoids:

- contention with interactive users
- logging, if the initialization parameter LOGMODE=N is specified for this job.

Dbspace Reload in Single User Mode

The SUM RELOAD DBSPACE process requires two separate steps:

1. Unload the dbspace in MUM, with the PAUSE and DISK option
2. Reload the dbspace in SUM

Dbspace Reorganization Driver - SQLREODR

The dbspace reorganization driver tool provides the capability to schedule a single job that will process reorganization of multiple dbspaces. This tool executes in MUM and should be scheduled to run on a Support Machine.*:* The tool consists of a:

- Panel driven dbspace candidate selection process
- Job scheduling function

SQLREODR may be used to:

- Reorganize a dbspace because of unclustered data
- Organize a table against a different index
- Take advantage of striping support of VMDSS

The SQLREODR tool gives the DBA a variety of selection criteria to select the dbspaces that meet your requirements. It also allows you to take an archive during the job when a specified log threshold is reached.

4.3.5.2 Table Reorganization and Redefinition - SQLTABLE

The table reorganization and redefinition tool provides the possibility to reorganize, migrate, backup, and redefine tables within databases. It provides full control and flexibility to manage database objects for better performance and to adapt to the changing needs.

SQLTABLE should be used during non-peak hours to prevent locking contention with other users on the database. Do not run more than one SQLTABLE job against different tables simultaneously within a single database, as it may cause catalog contention.

SQLTABLE has the following features:

- Redefine a table, for example add and delete columns, change datatype or attributes of columns
- Unload data from a table and append data to another table
- Move and copy a table to a different dbspace or database

4.3.5.3 Automated Dbspace Maintenance - SQLMAINT

SQLMAINT provides the following functions:

- Update statistics on selected dbspaces and keep track of this operation
- List information about statistics executed previously
- Create a list of dbspaces that apply to different criteria (see Figure 16 on page 48)
- Reorganize dbspaces from this list

These tools are designed for automatic execution on a scheduled basis on the Support Machine during periods of low system usage.

```
                DBSPACE REORGANIZATION CANDIDATES REPORT
  Database: BOEVMCT1.ELDB2A                    01/28/1998 11:15:23

             OPTIONS: Dbspaces: ALL.ALL, min=0, max=9999999,
                              days=14, seq=WEIGHT
  DBSPACES TO BE EXAMINED = 46
  ************************************************************
  CRITERIA 1 = DBSPACES with Tables with index type W
  CRITERIA 2 = DBSPACES with Tables with NOVERFLOW rows >= 10%
  CRITERIA 3 = DBSPACES with data or index pages > 70% used
  CRITERIA 4 = DBSPACES with index pages used > 40% and data pages < 40%
               -- OR -- with data pages used > 40% and index pages < 40%
  CRITERIA 5 = DBSPACES with data pages free space > 35%
               -- AND --  data pages used > 25%
  CRITERIA 6 = DBSPACES with empty data or index pages > 5%
                        of dbspace data pages or index pages
*****************************************************************************
PUBLIC.CTPLANOCONTAS         DBSPACENO: 40    LAST REORG: 0000000 00:00:00
       Criteria 1:  1 index unclustered
       Criteria 3:  FREEPCT: Old=15, Suggested=10
                    PCTINDX: Old=33, Suggested=31

                           NBR PAGES  OCCUPIED PAGES  %FREE  EMPTY PAGES
                    HEADER:        8        1 ( 12%)    94%
                    DATA:       5481     4518 ( 82%)    22%            0
                    INDEX:      2703     2044 ( 75%)     4%            0
            WEIGHT = 2,    DBSPACE PAGES = 8192
*****************************************************************************
PUBLIC.CTTABELASAUX          DBSPACENO: 38    LAST REORG: 0000000 00:00:00
       Criteria 4:  PCTINDX: Old=33, Suggested=28

                           NBR PAGES  OCCUPIED PAGES  %FREE  EMPTY PAGES
                    HEADER:        8        1 ( 12%)    91%
                    DATA:        336      207 ( 61%)    22%            0
                    INDEX:       168        4 (  2%)    45%            0
            WEIGHT = 1,    DBSPACE PAGES = 512
*****************************************************************************
```

Figure 16. Control Center for VM Dbspace Reorganization Candidates Report

4.3.5.4 Index Reorganization and Redefinition - SQLRINDX

The Index Reorganization tool automates index maintenance by analyzing the indexes in your databases and reorganizing those that require it.

This tool uses the system catalog to retrieve information about the indexes. It calculates the number of pages that each index should occupy. The total of the calculated index pages is compared to the actual number of index pages in the dbspace. The results are presented in a report. This report can be used as input to reorganize the listed indexes.

Note: System catalog table indexes are analyzed but not reorganized. Use the SQLCIREO utility to reorganize system catalog indexes.

4.3.5.5 Object Search and List - SQMUTIL

The Object Search and List tool (SQMUTIL) is a set of integrated applications that select data from the system catalogs and display information about dbspaces, tables, views indexes, packages and columns.

From within a level of hierarchy the information about a lower level can be requested, this means that while the information about dbspaces is displayed, the information about tables can be requested for a specific dbspace.

On each panel there are a number of functions available that can be performed against a selected object.

4.3.5.6 Rebind Package - SQLRBIND

Using the Rebind Package tool you can selectively rebind packages within the target database. This task can be scheduled or performed immediately.

4.3.6 Control Center Administration Tools

These tools are designed to minimize the usage of the consoles of the Service and Support Machines. The consoles should be kept free for the disposal of the System and Database Administration Tools. Use the Control Center administration tools to manage the database console related information and activities.

4.3.6.1 Job Scheduling

This tool can be used to schedule the execution of any job, part of Control Center or user-defined.

It provides the ability to schedule events during periods of low system usage under control of the Service or Support Machine. It also allows you to define various dependencies for each job which must be met prior to the job being automatically initiated by Control Center. Priorities can be specified for each job, to control which job should be favored if more than one job is scheduled to run in the same time frame.

4.3.6.2 Tape Management

Control Center supports environments with or without a tape management system. If there is no tape management system available, it sends tape mount requests directly to a defined tape operator user ID. It also provides support for the following tape management systems:

- VMTAPE
- AMMR (IBM Attachable Media Manager)
- EPIC

The sample file SQMOUNT $EXEC can be modified and should be renamed to SQMOUNT EXEC. This file keeps the commands to interface with each different tape management system. If no tape management system is available, this procedure will request tape mounts by the use of the CP MESSAGE command to the tape operator user ID. Modifications can be applied to support any other tape management system.

During a database archive, Control Center manages the tape mount requests to be sure you are using the correct tapes. Information about tape usage is maintained by the Service Machine. It can be reviewed and it will be used if recovery of a specific archive is requested.

4.3.6.3 Database Analysis

During the installation of Control Center there will be a number of files created that keep information about the environment and user defined retention periods of history files. Each time a new database is defined to Control Center there will some files created that keep specific information regarding this database.

These **control files** keep:

- Initialization parameters
- Tape information
- Authorizations
- Settings for backup procedures
- Minidisk owner and addresses of DB2 service and production disks

Mapping Dbextents - Storage Pools

This tool will create a report with information about storage pools and dbextents that belong to these pools. The next figure shows a sample output of this report.

```
                          Database: SQLDBA
                    DBEXTENT/STORPOOL Mapping
                           14 Feb 1998

                 ******** Dbextent Sequence ********

                     Virtual           Blocks/    DASD           Real
Dbextent  Storpool   Address   Pages  Cylinders   Type   Volid   Address
--------  --------  -------   -----  ---------  ------  ------  -------
  BDISK               200               35 CYL   3380   VMPRD2   03A5
 LOGDSK1              201               10 CYL   3380   VMPRD2   03A5
    1          1      202      5985     40 CYL   3380   VMPRD2   03A5
    2          2      203      4161     28 CYL   3380   VMPRD2   03A5
    3          3      204      5985     40 CYL   3380   VMPRD2   03A5


   ********* Pool Sequence *********

                     Virtual          Blocks/
Storpool  Dbextent   Address  Pages  Cylinders
--------  --------  -------  -----  ---------

    1          1     202      5985     40 CYL
    2          2     203      4161     28 CYL
    3          3     204      5985     40 CYL
```

Figure 17. Control Center for VM DBEXTENT-STORPOOL Mapping Report

View Message Log

All messages of the database console are received via SCIF (Single Console Image Facility) and logged in a console history file. All commands executed by the Service Machine on behalf of an administrator or user are also logged in this file. An administrator can use this history file to analyze problem situations and have an overview of requested information from the database operator console.

Information that is related to a specific database operation is kept in a separate file. This is the case for ADD or DELETE DBEXTENT, ADD DBSPACE and other database operations.

Database Commands

Use this facility if any CP command should be executed in the database virtual machine. CMS commands can only be executed when the database has been shut down, as there is no possibility to execute CMS commands from the database operator console.

4.3.6.4 Group Authorization Tools

These tools assist DBAs in maintaining the access to database objects. They simplify the process of authorization and shorten the amount of time needed to grant or revoke privileges.

A group can be defined to keep either:

- Users or
- Objects

An object group can be defined to keep either tables or packages. Authorities can be granted to or removed from user groups on object groups. A user added to a user group will receive all authorities, that were granted on all objects to this user group.

4.4 Summary

Control Center - on VSE and on VM - is a set of tools that help a DBA to automate many of his repetitive tasks. Especially, it can be used as a tool to monitor the database and help the DBA to keep it performing well for end users and applications. Scheduling reorganization tasks for periods of low system usage minimizes the impact of the maintenance to the users.

The DBA can define thresholds in the Monitor Tool such that, when they are reached, a message is sent to the operator console. With this information, the DBA has an idea where to look for possible performance problems, for example in a specific group of applications, database parameters, hardware or an area of the database structure.

Chapter 5. SQL-Tune

This chapter presents the installation and usage of SQL-Tune on VSE and VM.

SQL-Tune is designed to be used by application programmers in tuning their application programs performance. It can help you in locating a performance problem and gives you the possibility to change a statement on-line and see the behavior of this statement after the modification. It will also warn you if some conditions occur that need special attention.

SQL-Tune provides three groups of utilities:

- Application Development Utility to optimize SQL statements
- Database Administration Utility to analyze the status of a database
- System Monitoring Utility to monitor the SQL activity

SQL-Tune is a program offered

In France and Europe by:

XT-SOFT (Extended Software S.A.)
Centre d'Affaires Colombia
146, Boulevard de Valmy
92700 Colombes
France
Tel. +33-1-4769 5252
Fax. +33-1-4769 5251
email XTSOFT@AOL.COM

In the USA by:

The Paragon Collection, Ltd.
4676 Admiralty Way, Suite 300
Marina del Rey, CA 90292
USA
Tel. (USA +1-) 310-574-5370
Fax. (USA +1-) 310-574-5371
email earlech@aol.com

In the first part of this chapter, we discuss the installation procedures for SQL-Tune. In the second part of this chapter, we discuss how to use SQL-Tune utilities on VSE and VM.

5.1 Installation of SQL-Tune on VSE

This section describes the installation of SQL-Tune for VSE/ESA step by step. It is assumed that the reader is already familiar with VSE systems, VSE job control language, VSAM and the CICS system, and has some basic knowledge of DB2.

The installation consists of the following steps:

1. Prepare installation
2. Install SQL-Tune from tape
3. Define VSAM files
4. Prepare CICS
5. Acquire Password

5.1.1 Prepare Installation

Before installing SQL-Tune, you should determine the parameters that are needed during the installation steps and which also will affect the customization of the sample JCL provided with SQL-Tune:

- Define a VSE sublibrary for SQL-Tune. During installation, this library name will be needed as well as the library name of DB2.
- Determine the database name, on which the SQL-Tune will run.
- Determine in your DB2 database a user ID with DBA authority.

  During the installation, the DB2 user ID **SQLDBA** is required with password **SQLDBAPW.**
- Determine a storage pool for the PUBLIC dbspace XTSOFT1, if it doesn′t exist yet. 1024 pages are needed for this dbspace.

  To define the dbspace XTSOFT1, execute the following command sequence from ISQL.

  Alternatively, you can do this step with a batch job.

```
GRANT DBA TO SQLDBA IDENTIFIED BY SQLDBAPW
CONNECT SQLDBA IDENTIFIED BY SQLDBAPW
ACQUIRE PUBLIC DBSPACE NAMED XTSOFT1 ( PAGES=1024,STORPOOL=storpool )
COMMIT WORK
EXIT
```

Figure 18. ISQL Command Sequence to Acquire Dbspace for SQL-Tune in VSE

  storpool is the storage pool, into which you want to define this public dbspace XTSOFT1.
- Determine the DASD volume as well as the catalog name for the following four VSAM files which SQL-Tune needs:

  XTS1BAT (2 Cyls)
  XTS1ONL (5 Cyls)
  XTS1STM (1 Cyls)
  XTS1STB (1 Cyls)
- A tape drive is required for loading SQL-Tune.

5.1.2 Install SQL-Tune from Product Tape

Attach a tape drive with the address ***cuu***.

Enter the following commands from the VSE console:

1 **S RDR,*cuu***

2 **R RDR,XTS1LOAD**

Enter the **bold** statements after the pause statements:

// PAUSE ENTER COMMAND // SETPARM SQL=′SQL.BIB′
// SETPARM SQL=′xxx.xxx′ (DB2 library)

// PAUSE ENTER COMMAND // SETPARM TUNE=′SQLTUNE.BIB′
// SETPARM TUNE=′yyy.yyy′ (SQL-Tune library)

// PAUSE TO INSTALL FRENCH VERSION PRESS ENTER OTHERWISE TYPE CANCEL
Type *'Cancel'* to have the English Version of SQL-Tune, or press **ENTER** to have the French Version.

3 **R RDR, XTS1SRCE** (SRCE for SOURCE)

Specify the SQL-Tune library after the pause statement:

L146A ENTER THE LIBRARY.SUBLIBRARY SPECIFICATION TO BE PROCESSED - OR TYPE CANCEL.
yyy.yyy (SQL-Tune library)

4 **R RDR,XTS1CRE**

If the database, on which SQL-Tune will run, is not the default database of your system, perform the following steps:

a. Copy XTS1CRE from the RDR queue to your primary library.

b. Change every CONNECT statement in XTS1CRE to connect to the specific database (***dbname***) on which SQL-Tune will run:

CONNECT SQLDBA IDENTIFIED BY SQLDBAPW to ***dbname***;

CONNECT XTS1 IDENTIFIED BY XTS1PW to ***dbname***;

or modify the statement *//EXEC ARIDBS* to

`//EXEC ARIDBS PARM='DBNAME(dbname)'`

c. Submit the job from the ICCF library.

On the pause statement, enter the following command:

// LIBDEF PHASE,SEARCH=(xxx.xxx) DB2 library

5 **R RDR,XTS1PREP**

If the database, on which SQL-Tune will run, is not the default database of your system, perform the following steps:

a. Copy XTS1PREP from the RDR queue to your primary library.

b. Change every CONNECT statement in XTS1PREP to connect to the specific database (***dbname***) on which SQL-Tune will run.

CONNECT XTS1 IDENTIFIED BY XTS1PW ***to dbname***;

CONNECT XTSOFT IDENTIFIED BY XTSOFT ***to dbname***;

or modify the statement *//EXEC ARIDBS* to

`//EXEC ARIDBS PARM='DBNAME(dbname)'`

c. Submit the job from the ICCF library.

On the pause statement enter the following command:

// LIBDEF PHASE,SEARCH=(xxx.xxx) DB2 library

5.1.3 Define VSAM Files

- Punch XTS1DEF.A from the SQL-Tune library. Modify and submit it to define four VSAM files.

```
* $$ JOB JNM=XTS1DEF,DISP=D,CLASS=O
// JOB XTS1DEF
// DLBL IJSYSUC,'.........',,VSAM
// EXEC IDCAMS,SIZE=AUTO
 DELETE XTS1BAT PURGE CLUSTER
 DEFINE CLUSTER (NAME(XTS1BAT) VOLUME(vvvvv) NIXD SHR(4 3)-
                 CYL(002 01) RECSZ(4089 4089) REUSE )-
        DATA    (NAME(XTS1BAT.DATA) CISZ(4096))
 DELETE XTS1ONL PURGE CLUSTER
 DEFINE CLUSTER (NAME(XTS1ONL) VOLUME(vvvvv) NIXD SHR(2 3)-
                 CYL(005 01) RECSZ(4089 4089) REUSE )-
        DATA    (NAME(XTS1ONL.DATA) CISZ(4096))
 DELETE XTS1STM PURGE CLUSTER
 DEFINE CLUSTER (NAME(XTS1STM) VOLUME(vvvvv) NIXD SHR(2 3)-
                 CYL(001 01) RECSZ(8185 8185) REUSE )-
        DATA    (NAME(XTS1STM.DATA) CISZ(8192))
 DELETE XTS1STB PURGE CLUSTER
 DEFINE CLUSTER (NAME(XTS1STB) VOLUME(vvvvv) NIXD SHR(4 3)-
                   CYL(001 01) RECSZ(8185 8185) REUSE )-
        DATA    (NAME(XTS1STB.DATA) CISZ(8192))
 /*
 * $$ EOJ
```

Figure 19. Sample JCL to Define Four VSAM Files for SQL-Tune in VSE

Change *vvvvv* to the volume names for these VSAM files.

Replace '.....' with the catalog names for these VSAM files.

- Punch the XTS1INIT.A from your SQL-Tune library. Modify and submit it to initialize the VSAM files.

```
 * $$ JOB JNM=XTS1INIT,DISP=D,CLASS=O
 // JOB XTS1INIT
 // LIBDEF PHASE,SEARCH=(xxx.xxx, yyy.yyy)
 // DLBL XTS1ONL,'XTS1ONL',,VSAM,CAT=cccccc
 // DLBL XTS1STM,'XTS1STM',,VSAM,CAT=cccccc
 // EXEC XTS10003,SIZE=AUTO
    INIT XTS1ONL
 /*
 // DLBL XTS1BAT,'XTS1BAT',,VSAM,CAT=cccccc
 // DLBL XTS1STB,'XTS1STB',,VSAM,CAT=cccccc
 // EXEC XTS10003,SIZE=AUTO
    INIT XTS1BAT
 /*
 * $$ EOJ
```

Figure 20. Sample JCL to Initialize the VSAM Files for SQL-Tune in VSE

Replace *xxx.xxx* with the DB2 library name and *yyy.yyy* with the SQL-Tune library name.

Replace *cccccc* with the VSAM catalog name.

5.1.4 Prepare CICS

To prepare the CICS system for SQL-Tune, update the CICS control tables and modify the CICS startup job control.

5.1.4.1 CICS Table Entries

The following CICS tables need to be updated:

DFHPCT
DFHPPT
DFHFCT
DFHPLTPI
DFHPLTSD
DFHSIT

Since copy books have been loaded into the SQL-Tune library during installation of the product, you only need to add these copy books into the PPT, PCT and FCT tables. In addition, add the SQL-Tune library to the LIBDEF statement in these tables, and submit the jobs to compile.

DFHPCT Modifications

Add the following copy books into your PCT table:

```
COPY XTS1PCT
COPY XTSPPCT
```

Figure 21. DFHPCT Modifications for SQL-Tune in VSE

DFHPPT Modifications

Add the following copy books into your PCT table:

```
COPY XTS1PPT
COPY XTSPPPT
```

Figure 22. DFHPPT Modifications for SQL-Tune in VSE

You can also add the PCT entries and the PPT entries using the Resource Definition On-line (RDO) tool. To interactively define resources with RDO, access the CEDA CICS transaction. For more information on RDO and CEDA, see the CICS manuals.

DFHFCT Modifications

Add the following copy book into your FCT table:

```
COPY XTSPFCT7
```

Figure 23. DFHFCT Modifications for SQL-Tune in VSE

DFHPLTPI Modifications (Optional)

Add the following statement into your PLTPI table if you immediately want to start statistics capture when CICS begins:

```
    DFHPLT TYPE=ENTRY,PROGRAM=XTS1ON
```

Figure 24. DFHPLTPI Modifications for Statistics Capture on CICS Startup

DFHPLTSD Modifications (Optional)

Add the following statement into your PLTSD table if you want to stop statistics capture when CICS is shut down:

```
    DFHPLT TYPE=ENTRY,PROGRAM=XTS1OFF
```

Figure 25. DFHPLTSD Modification to Stop Statistics Capture on CICS Shutdown

DFHSIT Verification

Verify that **BFP=YES** is specified in your DFHSIT, otherwise modify it.

5.1.4.2 CICS Startup JCL

In the JCL add a LIBDEF for the SQL-Tune library and DLBLs for the VSAM files defined above.

```
// JOB CICS2 START
// LIBDEF *,SEARCH=(PRD2.DB2510, PRD2.XT1,......)
      .
      .
      .
// EXEC PROC=DTRCICS2                  LABELS FOR CICS FILES
// DLBL XTS1ONL,'XTS1ONL',,VSAM,CAT=VSESPUC
// DLBL XTS1STM,'XTS1STM',,VSAM,CAT=VSESPUC
// DLBL XTS1BAT,'XTS1BAT',,VSAM,CAT=VSESPUC
// DLBL XTS1STB,'XTS1STB',,VSAM,CAT=VSESPUC
      .
      .
      .
/*
/&
```

Figure 26. Starting the CICS System for Use with SQL-Tune

Shut down CICS and start it again with this new JCL.

5.1.5 Acquire Password

To get a valid product password for SQL-Tune, get your CPU number and call the vendor. After getting the product password, you can use the XTSP transaction to specify it.

XTSP also allows you to restrict the access of SQL-Tune by defining passwords of your own.

5.2 Installation of SQL-Tune on VM

This section describes the installation of SQL-Tune for VM/ESA step by step. It is assumed that the reader is already familiar with VM systems and has some basic knowledge of DB2.

The installation consists of the following steps:

1. Define resources
2. Install SQL-Tune from tape
3. Generate objects
4. Acquire Password

5.2.1 Define Resources

Before installing SQL-Tune, a user ID (XTS4TEST) should be defined that will own all minidisks for this product and capture the statistics.

- Define user ID XTS4TEST
  The recommendation for this user ID is to have classes BEG. Class B is needed to attach a tape unit. Class E is needed for the CP SAVESYS command.
- Define minidisks that keep all the runtime code. These sample entries are based on DASD device type 3380.

  ```
  MDISK 191 VOLUME 3380 xxxx 0015 MR
  MDISK 193 VOLUME 3380 xxxx 0065 MR
  MDISK 195 VOLUME 3380 xxxx 0030 MR
  MDISK 194 VOLUME 3380 xxxx 0015 MR
  ```

  The size of the 193 and 195 should be equal to the size of the database service disk (default 193) and production disk (default 195). Copy all files from the database service disk to the XTS4TEST 193 minidisk, and all files from the database production disk to the XTS4TEST 195 minidisk. This can be done by the use of copyfile or DDR, if the same device type is used for source and target minidisk.
- Determine the database name, on which the SQL-Tune will run.
- Determine in your DB2 database a user ID with DBA authority.

  During installation, the DB2 user ID **SQLDBA** is required with password **SQLDBAPW.**
- Determine a storage pool for the PUBLIC dbspace XTSOFT1, if it doesn′t exist yet. 1024 pages are needed for this dbspace.

  To define the dbspace XTSOFT1, execute the following command sequence from ISQL.

  Alternatively, you can do this step with a batch job.

```
GRANT DBA TO SQLDBA IDENTIFIED BY SQLDBAPW
CONNECT SQLDBA IDENTIFIED BY SQLDBAPW
ACQUIRE PUBLIC DBSPACE NAMED XTSOFT1 ( PAGES=1024,STORPOOL=storpool )
COMMIT WORK
EXIT
```

Figure 27. ISQL Command Sequence for Acquire Dbspace for SQL-Tune on VM

storpool is the storage pool, into which you want to define this public dbspace XTSOFT1.

- A tape drive is required for loading SQL-Tune.

5.2.2 Install SQL-Tune from Product Tape

Attach a tape drive with the address ***181***.

Enter the following command in XTS4TEST:

```
ACCESS 191 A
TAPE LOAD * * A
```

This will load all files to the minidisk defined as 191 in XTS4TEST.

5.2.3 Generate Objects

In this step, we will create the database objects and generate the runtime objects for the monitor and the end users.

1. Run **XTS4CRE** EXEC
   This will create all tables and indexes needed for the runtime. There will also be some grants executed on system tables to XTS4, the DB2 user ID (SQL-ID) that will capture the statistics.
2. Run **XTS4PREP** EXEC
   This will reload all packages, needed for the product, into the currently connected database. Grant run will be executed on these packages to authorize users to run these packages.
3. Run **XTS4GMOD** EXEC
   This will create all CMS modules to run the product.
4. Run **XTS4SEG** EXEC
   First define a shared segment that will keep objects, created by the monitor. The XTS4SEG exec will check the existence of the shared segment and load predefined code to this segment.

   ```
   DEFSEG XTS4SHR  hexpage1-hexpage2 SW
   ```

   The size for this segment should be at least 32 pages. The start and end address of this segment, indicated by hexpage1 and hexpage2, should be defined so that no overlap exists with other segments that are loaded at the same time. After the load, verify the segment by the use of the following CP command:

   ```
   Q NSS NAME XTS4SHR MAP
   ```

   This command will create an output as in next figure.

```
FILE FILENAME FILETYPE MINSIZE  BEGPAG ENDPAG TYPE CL #USERS PARMREGS
0158 XTS4SHR  DCSS       N/A    00900  009FF   SW  A  00002   N/A
```

Figure 28. Output of Q NSS for SQL-Tune in VM

5. Linkedit the resource manager with some new object code and generate a new loadlib ARISSQLD with this modified object code.

There are different ways to generate the new loadlib, depending on the DB2 version you are using.

5.2.3.1 SQL/DS Version 3:

For users, running version 3 of SQL/DS, the exec **XTS4OLRM** should be executed. This exec will modify the loadlist, containing the list of TEXT files, for building the resource manager. Then it will create a new ARISSQLD loadlib that will be kept on the 195 mdisk of XTS4TEST.

5.2.3.2 DB2 Version 5:

For users running version 5 of DB2, the procedure is different.
First log on to the product owner of DB2 Server for VM (default 564815A1)
Setup the environment by executing the command

```
VMFSETUP 564815A1 DB2VM
```

The following output will be produced.

```
VMFSET2760I VMFSETUP processing started for 564815A1 DB2VM
VMFUTL2205I Minidisk|Directory Assignments:
            String    Mode  Stat  Vdev  Label/Directory
VMFUTL2205I LOCALSAM  E      R/W  2C2   SQL2C2
VMFUTL2205I APPLY     F      R/W  2A6   SQL2A6
VMFUTL2205I           G      R/W  2A2   SQL2A2
VMFUTL2205I DELTA     H      R/W  2D2   SQL2D2
VMFUTL2205I BUILD2    I      R/O  195   SQL295
VMFUTL2205I BUILDO    J      R/O  193   SQL293
VMFUTL2205I BASE      K      R/W  2B2   SQL2B2
VMFUTL2205I SYSTEM    L      R/W  295   SQL295
VMFUTL2205I           M      R/W  293   SQL293
VMFUTL2205I --------  A      R/W  191   SQL191
VMFUTL2205I --------  B      R/O  5E5   MNT5E5
VMFUTL2205I --------  D      R/W  DIR   VMSYS:MAINT.51D
VMFUTL2205I --------  S      R/O  190   MNT190
VMFUTL2205I --------  Y/S    R/O  19E   MNT19E
VMFSET2760I VMFSETUP processing completed successfully
Ready; T=5.91/6.19 10:42:20
```

Figure 29. Output of VMFSETUP to Install SQL-Tune in VM

Keep an original copy of the buildlist ARIBLLLD EXEC that resides on the 2B2 mdisk, named BASE and accessed as K.

```
FILELIST ARIBLLLD EXEC K
COPYFILE ARIBLLLD EXEC K = OEXEC K (OLDD'
```

Link to the mdisk 191 of XTS4TEST and access this mdisk.

```
CP LINK XTS4TEST 191 194 RR
ACCESS 194 G
```

Copy the XTS4OLRM TEXT file to the target disk.

```
COPYFILE XTS4OLRM TEXT G = = K (OLDD
```

Modify the ARIBLLLD EXEC as follows. XEDIT the file and locate *OBJNAME. ARIRVMRM*

```
:OBJNAME. ARIRVMRM LEPARMS NCAL RENT AMODE 31 RMODE ANY
:PARTID. ARIRFSTC TXT
:PARTID. ARIRRTRC TXT
:PARTID. ARIRWUDC TXT
:PARTID. ARIRCTLC TXT
:PARTID. ARIRCONC TXT
:PARTID. ARIRDIRC TXT
:OPTIONS. LANGFUNC ARILDAR
:PARTID. ARIRLCNC TXT
:OPTIONS.
:PARTID. ARIRVRMC TXT
:OPTIONS. LANGFUNC ARILDAR
 ......
:OPTIONS. LANGFUNC ARILDARE
:PARTID. ARITKEY TXT
:OPTIONS. ENTRY ARIRVIR
:EOBJNAME.
```

Figure 30. Original ARIBLLLD EXEC

Add a record immediately after the located line and modify the line specifying the entry for the object, as indicated below.

Note: For the partid XTS4OLRM specify filetype **TEXT** instead of **TXT**.

```
:OBJNAME. ARIRVMRM LEPARMS NCAL RENT AMODE 31 RMODE ANY
:PARTID. XTS4OLRM TEXT
:PARTID. ARIRFSTC TXT
:PARTID. ARIRRTRC TXT
:PARTID. ARIRWUDC TXT
:PARTID. ARIRCTLC TXT
:PARTID. ARIRCONC TXT
:PARTID. ARIRDIRC TXT
:OPTIONS. LANGFUNC ARILDAR
:PARTID. ARIRLCNC TXT
:OPTIONS.
:PARTID. ARIRVRMC TXT
:OPTIONS. LANGFUNC ARILDAR
 .......
:OPTIONS. LANGFUNC ARILDARE
:PARTID. ARITKEY TXT
:OPTIONS. ENTRY XTS4OLRM
:EOBJNAME.
```

Figure 31. ARIBLLLD EXEC Modifications for SQL-Tune in VM

Finally execute the following command to recreate the ARISSQLD LOADLIB.

```
VMFBLD PPF 564815A1 DB2VM ARIBLLLD ARIRVMRM ( ALL
```

The newly created loadlib has to be copied to the 195 mdisk, owned by XTS4TEST.
This ends the build process of the resource manager.

DO NOT FORGET to restore the original buildlist ARIBLLLD EXEC which should be used for generating the unmodified resource manager if any maintenance was applied to the DB2 product code.

```
COPYFILE ARIBLLLD OEXEC K = EXEC K (OLDD REPLACE'
```

Note: The users for which you want to capture statistics, have to link and access this 195 mdisk of XTS4TEST instead of the regular production disk of the database server.

5.2.4 Acquire Password

To get a valid product password for SQL-Tune, look up your CPU number and call the vendor. After getting the product password, you can use the XTSP command to specify it.

XTSP also allows you to restrict the access to SQL-Tune by defining passwords.

5.3 Using SQL-Tune on VSE and VM

SQL-Tune is a powerful tool which describes the access path that was chosen by the optimizer together with some other useful information about the query. It also provides information about the structure and status of the database, such as storage pools, dbspaces, tables and indexes. It can switch between monitoring on-line and batch SQL activities, and it can gather statistics on both.

5.3.1 Performance Tuning

SQL-Tune can be used in performance analysis and tuning for the following purposes:

1. Evaluate data and application design

   For new applications, the design of tables, indexes and imbedded SQL DML statements in the application programs needs to be evaluated. SQL-Tune provides information which may help the user to identify areas for improvement. The following areas are considered as candidates for changes to improve performance:

   - The SQL statements may be altered using better predicates to improve the access path.
   - The table design may be modified by moving columns from one table to another, or splitting or joining tables.
   - The design of the indexes may be modified by adding or removing columns.
   - New indexes may be created.
   - Existing indexes may be modified or discarded.

2. Tune existing applications

   If there are applications which are not performing satisfactorily, the imbedded SQL statements can be analyzed by SQL-Tune.

   By using SQL-Tune, users can:

   - Identify the paths that the optimizer has chosen.
   - Change the SQL statements on-line and see the changes of the access path.

- Test their execution and see the statement execution duration.

5.3.2 SQL-Tune Utilities

The following figure shows the main panel of the SQL-Tune. It gives you a global overview of the three utilities of SQL-Tune:

- The first part is called Application **Development** Utility and is used to analyze and optimize the SQL statements.
- The second part is called Database **Administration** Utility and is used to explore the database and detect the critical issues of the database.
- The third part is called System **Monitoring** Utility and is used to monitor the on-line and batch SQL Activities in VSE or all SQL activity in VM.

You can activate SQL-Tune by entering:

- XTS1 as an on-line transaction name in VSE, or
- the CMS command XTS4 in VM

```
XTS1
Date: 11/28/97                                              Time:  9:07:13
                   Enter option:           Password:
               S Q L T U N E    Version 2.1

                PF1   Help

   DEVELOPMENT
                PF2   SQL Statement Tuning
                PF4   Statistics

   ADMINISTRATION
                PF5   Data Base Scan
                PF6   List of Dbspaces
                PF7   List of Tables
                PF8   List of Critical Dbspaces
                PF9   List of Critical Tables

   MONITORING
                PF10  CICS SQL Activity       PF13 BATCH SQL Activity
                PF11  CICS Statistics Capture PF14 BATCH Statistics Capture
                PF12  Exclusion/Inclusion


PF3: End     (General Function Keys PF1: Help)
```

Figure 32. SQL-Tune Main Menu in VSE

```
XTS4                                                    Base :ELDB2A
Date: 19/01/98    S Q L T U N E Version 2.1   Time 10:15

                    PF1   Help

   DEVELOPMENT
                    PF2   On-line Tuning
                    PF4   Statistics

   ADMINISTRATION
                    PF5   Base Scan
                    PF6   List of dbspaces
                    PF7   List of tables
                    PF8   List of critical dbspaces
                    PF9   List of critical tables

   MONITORING
                    PF10  SQL Activity
                    PF11  Capture statistics
                    PF12  Exclusion/Inclusion

                  Enter option :         Base        :
                                         password    :
PF3:End (General function keys PF1:help AP2:cms/subset
```

Figure 33. SQL-Tune Main Menu in VM

There are some minor differences between the main panel on VSE and VM.

The currently connected database name is displayed in VM.

In VM there is a possibility to connect to a different database by entering the database name.

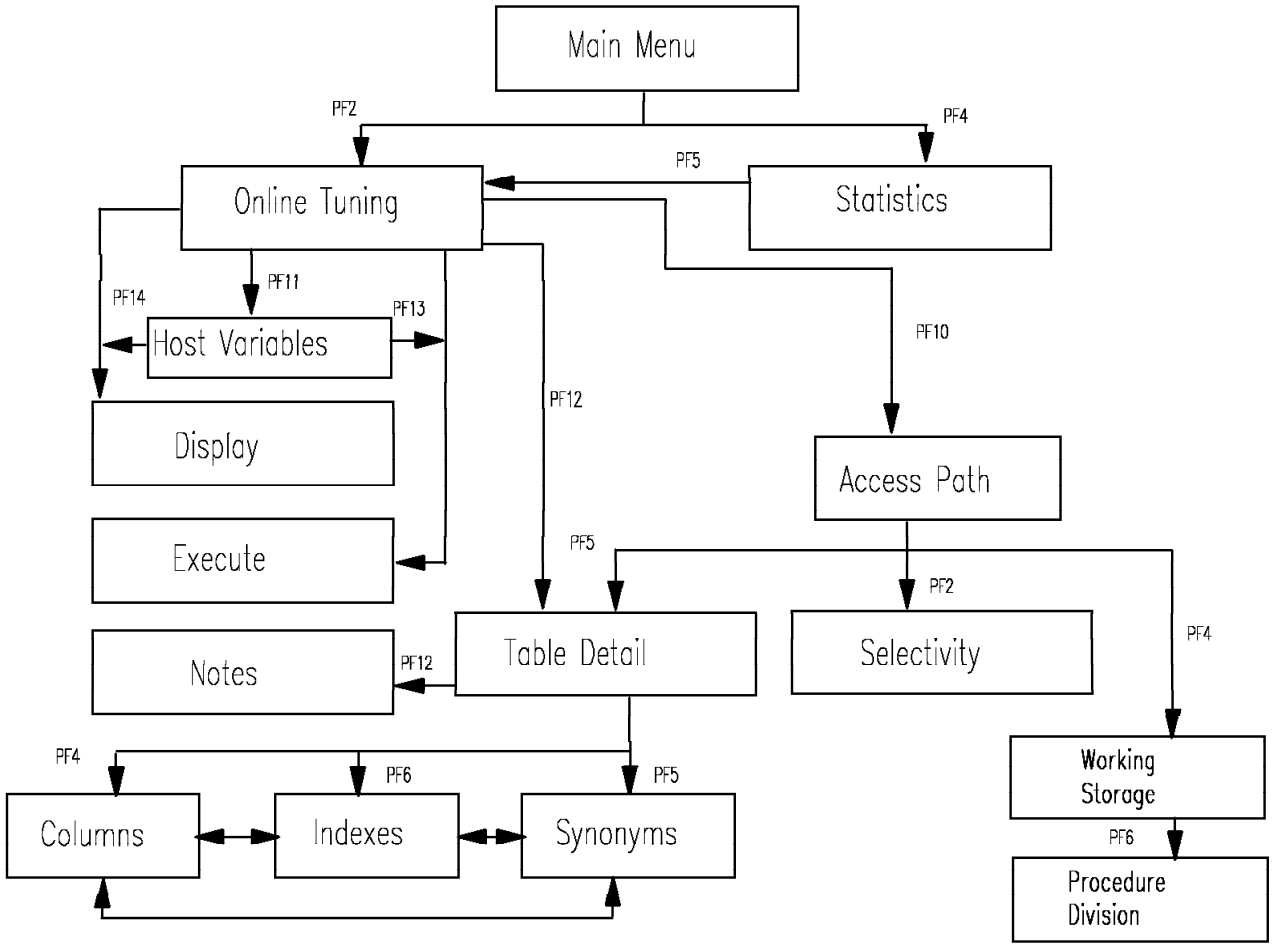


Figure 34. SQL-Tune Development Function Overview

5.3.2.1 Application Development Utility

This utility can be used to check the statistics of SQL activities. See Figure 34 for a schematic overview of the functions. The following is an example of typical steps to evaluate.

1 Choose STATISTICS option (PF4) from the Main Menu to display the statistics. It shows the statistics of on-line SQL statements which is split into their single atomic SQL requests.

At first, you are shown an empty frame. You can select specific statistics by giving the parameter to any one of the following items:

- Date
- Time
- SQL preprocessed package name
- Specific SQL statement
- Transaction name
- Terminal id

- Minimum duration of program (Cost Per Program)
- Minimum average unit cost of SQL statement

On the following example, we enter the date “11/24/97.” It shows the SQL statistics of Nov 24, 1997.

```
XTS1
Date: 12/02/97                                          Time: 11:03:28
                            ON/LINE STATISTICS
Date: 11/24/97 Time       :          SQL Prep:          SQL Stmt :
               Transaction:          Terminal:
        Cost per program  :          Average unit cost:

   SQL prep  date      time   tran term n¢  duration  no    average  req
   ARIISQL  11/24/97  9:55:24 CISQ A003 04     0,014     1    0,014 PREP
                                      02     0,010     1    0,010 RBACK
   ARIISQL  11/24/97  9:55:39 CISQ A003 01     0,001     1    0,001 COMIT
   ARIISQL  11/24/97  9:56:03 CISQ A003 01     0,002     1    0,002 COMIT
   ARIISQL  11/24/97 10:29:32 CISQ A003 04     0,013     1    0,013 PREP
                                      04     0,002     2    0,001 DESCR
                                      04     0,001     1    0,001 OPEN
                                      04     0,100   111    0,000 FETCH
                                      04     0,002     1    0,002 CLOSE
                                      01     0,004     1    0,004 COMIT
   ARIISQL  11/24/97 10:29:56 CISQ A003 04     0,007     1    0,007 PREP
                                      02     0,039     1    0,039 RBACK
   ARIISQL  11/24/97 10:30:08 CISQ A003 04     0,030     1    0,030 PREP
                                      04     0,002     2    0,001 DESCR

PF3:Return PF5:Tuning PF6:Verif. PF7:Page-1 PF8:Page+1 PF12:Batch Statistics
```

Figure 35. SQL-Tune On-Line Statistics

Pressing enter fills the screen with information fulfilling the given conditions.

The following information is displayed on the above figure:

- The preprocessed package name
- The execution date and time
- The transaction name
- The terminal id
- The number of the SQL statement in the package
- Total execution duration
- The number of times (how often) the SQL request was executed
- Average duration for each execution (duration/no)

The transaction name and terminal id are replaced by user ID if this panel is requested in VM.

The dynamic SQL statements are highlighted on the above screen.

To switch to batch statistics, press PF12.

2 To have an overview of the behavior of a selected SQL statement in a different run, position the cursor on the selected SQL statement and press PF06 (Verif). This will produce a panel similar to the next figure which shows an overview of statement 01 in program *PBPB998*.

```
XTS4                                                          Base : ELDB2A
Date: 22/01/98                                                Time : 13:52:34
                        APPLICATION STATISTICS
Date:          time:          Prep SQL: PBPB998      Stmt No. : 01
Userid: ELRES5      Cost per program  :           Average unit cost :


   prep SQL  date     time     userid  No duration No.times average  req
   PBPB998  22/01/98 13:45:05 ELRES5   01    14,304      1    14,304 OPEN
                                       01    22,267  23725     0,000 FETCH
                                       01     0,000      1     0,000 CLOSE
   PBPB998  22/01/98 14:03:46 ELRES5   01    15,303      1    15,303 OPEN
                                       01    22,027  23725     0,000 FETCH
                                       01     0,001      1     0,001 CLOSE
   PBPB998  22/01/98 15:15:41 ELRES5   01    15,045      1    15,045 OPEN
                                       01    21,988  23725     0,000 FETCH
                                       01     0,001      1     0,001 CLOSE

PF3: Return  PF5: On-line tuning  PF6: Verif.  PF7: Page-1  PF8: Page+1
```

Figure 36. SQL-Tune Application Statistics

3 On the On-line or Batch Statistics Menu, move the cursor to the line which you want to inspect and choose the Tuning option (PF5) to show the SQL statement as shown in the following figure, the same screen you would get when using PF2 from the main menu.

```
XTS1
Date: 12/02/97                                          Time: 11:01:57
                           ON-LINE TUNING

Creator: UTIL     Program: EXTFK    Statement: 01 User: UTIL     /


SELECT 'SQL/DS HELP' FROM SYSTEM.SYSTEXT2 WHERE
ITEM = :H001  OR  ITEM = :H002


PF 2:Indentation 3:Return 5:Next 7:Page-1 8:Page+1 9:Insert
PF 10:Access path 11:Host Variables 12:Table 13:Execute 14:Display
```

Figure 37. SQL-Tune SQL Statement Tuning

4 Press PF2 from the above screen to see the SQL statement in a more readable format with indentation.

```
XTS1
Date: 12/03/97                                              Time: 11:02:40
                              ON-LINE TUNING

Creator: UTIL     Program: EXTFK    Statement: 01 User: UTIL     /

*************************************************************************
SELECT   'SQL/DS HELP'
FROM     SYSTEM.SYSTEXT2
WHERE    ITEM=:H001
   OR    ITEM=:H002


*************************************************************************

PF 2:Indentation 3:Return 5:Next 7:Page-1 8:Page+1 9:Insert
PF 10:Access path 11:Host Variables 12:Table 13:Execute 14:Display
```

Figure 38. SQL-Tune Indentation of SQL Statement

5 On the upper right of the above figure, fill in the DB2 user ID and password. The password is only necessary if the display function (PF14) is used.

6 Press PF10 to see the access path chosen by the optimizer.

The following figure shows the access method together with some other useful information.

```
XTS4                                                        Base : ELDB2A
Date: 26/01/98                                              Time : 11:15:40
                              SQL Access Path

  Program  : ARIISQL  Stmt No.  : 11 No. blocks: 01  Total cost:       0,224

block: 01/00       Rows :        49       Cost:       0,224 *         1,0


     Table : SYSTEXT2              Creator : SQLDBA
                                   Method  : INDEX SCAN
                                   Index   : NO POSSIBLE USE

     Table :                       Creator :
                                   Method  :
                                   Index   :


PF2:Selectiv  PF3:Return  PF4:Dyn.Prog.  PF5:Table  PF7:Backward  PF8:Forward
```

Figure 39. SQL-Tune SQL Access Path

From the above figure, you can

- Press PF5 to show the table properties using the administration utilities described below:
  - The list of indexes (refer to Figure 50 on page 77)
  - The structure of columns (refer to Figure 49 on page 76)

- The synonyms

- Choose PF6 (Selectiv) to see the Filter Factors, which are values between 0 and 1 expressing the fraction of rows from the table that qualify for the predicate.
- Choose PF4, if *'**Dynamic SQL is recommended**'* appears on the above screen. Then, you can look at the application's:
  - working-storage section
  - procedure division

  Otherwise, Dynamic SQL is not required and the "Dyn. Prog" option can't be used.

7 If there are any host variables in the predicate, you can choose Host Variables Option (PF11) to display the specification of the host variables.

```
XTS1
Date: 12/03/97                                              Time: 11:07:16
                      Host-Variables Specification

               Program: EXTFK       Number of resulting rows: 00250


 Host_Variable       Type       N Len. Dec       Value
H001               SMALLINT     N     2      7000
H002               SMALLINT     N     2     30000


PF3: Return   PF7: Page-1   PF8: Page+1   PF13: Execute   PF14: Display
```

Figure 40. SQL-Tune Host Variables Specification

8 After supplying the value for the host variables, you can execute the SQL statement with PF13. This will show the duration of the request, number of executions and average unit cost. See Figure 42 on page 71 for sample output.

9 The Display option (PF14) will display the output of the SQL statement. See Figure 43 on page 72 for sample output from

```
SELECT * FROM SQLDBA.EMPLOYEE
```

The zoom option (PF02) on this panel will display only the row, where the cursor is currently located. See Figure 44 on page 72 for sample output.

Return to step 4 on page 68 and change your statement again. Repeat the steps until step 9. You can do this several times until you have found the best form of the SQL statement.

In this sample we replaced the *OR* predicate by an *IN* predicate.
The display for the new access path is shown in the next figure where we can see that the access path has changed from *INDEX SCAN* to *DIRECT ACCESS BY INDEX*

```
XTS4                                                        Base : ELDB2A
Date: 26/01/98                                              Time : 11:17:00
                          SQL Access Path

  Program  : ARIISQL  Stmt No.  : 11 No. blocks: 01  Total cost:       0,023

block: 01/00       Rows :        49       Cost:       0,023 *         1,0


     Table : SYSTEXT2              Creator : SQLDBA
                                   Method  : DIRECT ACCESS BY INDEX
                                   Index   : SYSTEXT2INDEX(1)

     Table :                       Creator :
                                   Method  :
                                   Index   :


PF2:Selectiv  PF3:Return  PF4:Dyn.Prog.  PF5:Table  PF7:Backward  PF8:Forward
```

Figure 41. SQL-Tune New Access Path

Do not forget to apply this change to the statement in your application source.

```
XTS4                                                  Base : ELDB2A
Date: 20/01/98                                        Time : 10:05:02
                          ON-LINE TUNING

Creator : SQLDBA   Program  : ARIISQL  SQL Stmt : 11 User: SQLDBA   /


SELECT 'SQL/DS HELP' FROM SQLDBA.SYSTEXT2 WHERE ITEM IN (:H001,:H002)

                              +--------------------------------------+
                              | Duration    No    Average    Request |
                              |     0,022      1      0,022   OPEN   |
                              |     0,156    250      0,000   FETCH  |
                              |     0,000      1      0,000   CLOSE  |
                              +--------------------------------------+


PF 2:Indentation 3:Return 5:Next 7:Page-1 8:Page+1 9:Insert
PF 10:Access path 11:Host Variables 12:Table 13:Execute 14:Display
```

Figure 42. SQL-Tune Execute Output

```
DATE : 20/01/98     LINE : 0001 : 0012    COL : 001 : 014  TIME : 14:16:37

      -EMPNO -FIRSTNME    -M-LASTNAME       -WOR-PHON-HIREDATE   |
=====  000010 CHRISTINE    I HAAS            A00 3978 1965-01-01
=====  000020 MICHAEL      L THOMPSON        B01 3476 1973-10-10
=====  000030 SALLY        A KWAN            C01 4738 1975-04-05
=====  000050 JOHN         B GEYER           E01 6789 1949-08-17
...............
=====  000270 MARIA        L PEREZ           D21 9001 1980-09-30
=====  000280 ETHEL        R SCHNEIDER       E11 8997 1967-03-24
=====  000290 JOHN         R PARKER          E11 4502 1980-05-30
=====  000300 PHILIP       X SMITH           E11 2095 1972-06-19
=====  000310 MAUDE        F SETRIGHT        E11 3332 1964-09-12
=====  000320 RAMLAL       V MEHTA           E21 9990 1965-07-07
=====  000330 WING           LEE             E21 2103 1976-02-23
=====  000340 JASON        R GOUNOT          E21 5698 1947-05-05
CLEA:END PF02:ZOOM PF03:EXIT PF04:SAV PF05:SEL PF06:ALL PF07:P-1 PF08:P+1 PF09
```

Figure 43. SQL-Tune Display Output

```
XTS8
DATE: 20/01/98   CREATOR:          TNAME:   TIME: 14:16:37

EMPNO              000010
FIRSTNME           CHRISTINE
MIDINIT            I
LASTNAME           HAAS
WORKDEPT           A00
PHONENO            3978
HIREDATE           1965-01-01
JOB                PRES
EDLEVEL                18
SEX                F
BIRTHDATE          1933-08-01
SALARY                52750,00
BONUS                  1000,00
COMM                   4220,00

PF02:ZOOM PF03:EXIT PF05:SEL PF07:P-1 PF08:P+1
```

Figure 44. SQL-Tune Zoom Option

5.3.2.2 Database Administration Utility

This displays the status of some aspects of the database environment such as:

- Storage Pools (“Date Base Scan”)
- Dbspaces
- Tables
  - Indexes
  - Columns
  - Synonyms
- Critical Tables
- Critical Indexes

Data Base Scan (PF5)

This function is used to retrieve information on the storage pools defined for a database. A storage pool is made up of one or more database extents (dbextents).

```
XTS1
Date: 12/04/97                                              Time: 11:02:07

          BASE: SQL/DS                              Number of STORPOOLS:  14

--------------------------- LIST OF STORPOOLS ---------------------------

number  backout     no. Dbspaces       no. Tables       no. Pages
                 total public private                 total   active
    1     YES        9     9      0         53       25216      1344
    2     YES        6     6      0         17       37120      2084
    3     YES        2     2      0         13        3072       362
    4     YES        1     1      0          1        3072         0
    5     YES        2     2      0          2         256         4
    6     YES        1     1      0          1       20480       782
    7     YES        1     1      0         12       12800      1355
    8     YES        1     1      0          1        4096      1112
    9     YES        5     5      0         28      174592      7214
   11     YES        2     2      0         25       21504         3
   12     YES        1     1      0          1       20480      1627
   13     YES        1     1      0          1       24064     11112
   14     YES        5     5      0         41      192000     26215
 ********************************************************************************
 PF3: Return   PF5: Detail   PF7: Backward   PF8: Forward
```

Figure 45. SQL-Tune Data Base Scan

For each storage pool displayed, the following information is provided:

- Storage pool number
- Backout, indicating whether the storage pool is recoverable or not
- The number of total dbspaces (public dbspaces + private dbspaces)
- The number of tables
- The total number of pages
- The number of active pages

Move the cursor to the line of the storage pool which you want to inspect in detail (PF5), then you can see the information about the dbspaces (tables, columns and indexes) of this storage pool in the same format as the function described next.

List of Dbspaces (PF6)

This function allows you to see the status of all dbspaces in the database.

```
XTS1
Date: 12/04/97                                              Time: 16:06:12

--------------------------- LIST OF DBSPACES ---------------------------

name               no  owner    type lock no tab no pages no pages %page %free
                                                    total    active   index space
ADMGROUP            23 PUBLIC   PUBL PAGE      5      256         0    10    10
ATDHHXSS            29 PUBLIC   PUBL PAGE      1     2048        57    33     5
CC_MONITOR          22 PUBLIC   PUBL PAGE      8      128         0    33     0
CONTABILIDAD_1      36 PUBLIC   PUBL PAGE     18     4096       255    33     0
CONTABILIDAD_11     49 PUBLIC   PUBL PAGE      1    25600     16250    33    15
CONTABILIDAD_13     41 PUBLIC   PUBL PAGE      1    25600      5279    33     0
CONTABILIDAD_2      37 PUBLIC   PUBL PAGE      0     4096         0    33     0
CONTABILIDAD_21     30 PUBLIC   PUBL PAGE      0    12800        -1    33     0
CONTABILIDAD_24     38 PUBLIC   PUBL PAGE      8    12800        13    33     0
CONTABILIDAD_25     46 PUBLIC   PUBL PAGE      7    12800       357    33     0
CONTABILIDAD_27     48 PUBLIC   PUBL PAGE      1    12800      8563    33     0
CONTABILIDAD_33     31 PUBLIC   PUBL PAGE      2    12800      1701    33     0
CONTABILIDAD_45     32 PUBLIC   PUBL PAGE     12    12800      1355    33     0
DATARFTR             6 PUBLIC   PUBL PAGE     11     6400        56    33     0
*******************************************************************************
PF3:Return  PF5:Detail  PF7:Backward  PF8:Forward  PF9:Selection  PF12:Notes
```

Figure 46. SQL-Tune List of Dbspaces

The following information is displayed:

- The name of the dbspace
- The number of the dbspace
- The owner of the dbspace
- The type of the dbspace (PUBLic or PRIVate)
- The lock mode of the dbspace (ROW, PAGE or DBSPACE)
- The number of tables in the dbspace
- The total number of pages that is allocated for the dbspace
- The number of active pages in the dbspace
- The percentage of dbspace pages reserved for the indexes
- The minimum percentage of free space reserved within each page (PCTFREE)

The highlighted lines on the above screens are the critical dbspaces, for which the same conditions apply as listed in “List of Critical Dbspaces (PF8)” on page 78.
You can press PF12 (Notes) to see the appropriate comments that apply to these dbspaces.

The Detail option (PF5) offers information about the tables (columns, indexes and synonyms) in the dbspace in the same format as described next.

List of Tables (PF7)

This function allows you to see the status of all tables in the database.

```
XTS1
Date: 12/04/97                                                Time: 18:04:57

------------------------------- LIST OF TABLES -------------------------

Name                Creator   syn  ins    no     pct            ROWS
                                          pages  pages  av.len    no   overflow
ACTIVITY            SQLDBA           I       1   100       31      18      0
APPL_GROUP_TAB      SQLMSTR          I       0     0        0       0      0
ATQHHXSS            SQLDBA           D      57   100       27    6484      0
BC_ACTIVIDADES      GENERAL          I       3   100       46     164      0
BC_AUTOCONSULTA     CTABLINT         D       1    33       14      46      0
BC_BANCOS           GENERAL          I       3    60      202      28      0
BC_CNTL_INDICADOR   CTABLINT         D       1    20       20      13      0
BC_COTIZ_ARB_HIST   GENERAL          I     667   100       60   39889      0
BC_DEPENDENCIAS     GENERAL          D       3     0      123      52      0
BC_IND_HIST         GENERAL          I       5     1       22     330      0
BC_INDICADORES      CTABLINT         D       1     0       50      14      0
BC_LOG_ACTIVIDADES  GENERAL          I       3     0       70     101      0
BC_MONEDAS          GENERAL          I       4     1      111      71      0
BC_OPERATIVAS       GENERAL          I       2     0       95       7      0
BC_PAISES           GENERAL          I       6     1       54     251      0
********************************************************************************
PF3:Return  PF5:Detail  PF7:Backward  PF8:Forward  PF9:Selection  PF12:Notes
```

Figure 47. SQL-Tune List of Tables

The following information is displayed:

- The table name and creator
- Whether synonyms for the table exist (YES)
- The way to insert a row into the table (I = using a clustering index, or D = default)
- The number of active pages
- The approximate percentage of the total active pages in the dbspace that have rows from this table on them (PCTPAGES)
- The average row length for the table in bytes
- The number of rows in the table
- The number of rows that needed overflow pages

The Selection option (PF09) gives you the possibility to specify a table creator, table name or both to get a reduced list of tables. This can be very useful if a table list is requested from a database with many tables.

```
XTS4
Date: 26/01/98                                              Time : 12:56:20

                              LIST  OF  TABLES

Name                Creator   syn  ins    No.     %             ROWS
                                       pages  pages avg.len  No.    overflow
ACTIVITY            SQLDBA           I       2    33      31       18      0
CL_SCHED            SQLDBA           D       1    17      23        3      0
CMD                 DATARFTR         I       0    -1      -1        0      0
COST_TABLE          ELRES3           I       0    -1      -1        0      0
COST_TABLE          ELRES5           I       0    -1      -1        0      0
COST_TABLE          SQLDBA           I       0    -1      -1        0      0
CSQESTAB            ELRES5           D       3   100     223       40      0
CUSTOMER            SQLDBA           I   11112   100     419   100000      0
DEPARTMENT          SQLD+-----------------------------------------------------+
DISTRICT            SQLD|                      TABLES                         |
DRIVER_TABLE        SQLD|                                                     |
ELOLANGUAGE         SQLD|   Name  :                      Owner :              |
ELOOPTIONS          SQLD|                                                     |
ELOTEXT1            SQLD+-----------------------------------------------------+
ELOTEXT2            SQLDBA           I      43    93      76     2270      0

PF3:Return PF5:Detail PF7:Backward PF8:Forward PF9:Selection PF12:Notes
```

Figure 48. SQL-Tune Table Selection

The highlighted lines on the above screen are considered as critical tables, for which the same conditions apply as listed in “List of Critical Tables (PF9)” on page 78. With the cursor positioned on a highlighted line you can press PF12, to see the comments that apply to this table.

Move the cursor to the line of the table which you want to inspect in detail and press PF5, to display the columns’ structure.

```
XTS1
Date: 12/08/97                                                  Time: 15:
     STORPOOL:   1         DBSPACE: SAMPLE                   backout: YES
                                 T A B L E
        Name : ACTIVITY                          Creator : SQLDBA
        ins  : I             no pages:       1   % pages :  100
ROW---- average len:    31   no.     :      18   overflow:    0
---------------------------- STRUCTURE OF COLUMNS -----------------------

Name                  n¢     type    length  null   columns   Orderfield
                                                    count
ACTNO                  1   SMALLINT          NO         18      YES
ACTKWD                 2   CHAR           6 NO         -1
ACTDESC                3   VARCHAR       20 NO         -1


************************************************************************
PF3:Return  PF5:Synon.  PF6:Index  PF7:Backward  PF8:Forward
```

Figure 49. SQL-Tune Structure of Columns

The above figure shows the following information of the columns:

- The name of the column
- The order number of the column
- The data type of the column
- The length of the column in bytes
- Whether NULLs are allowed or not
- Number of different values in this column, retrieved from the statistics. ′-1′ means no statistics exist
- Whether the column belongs to a single-column index or not

From this figure, you can see the synonyms of this table (PF5) and the structure of the index (PF6).

```
XTS1
Date: 12/09/97                                                Time: 12:04:11
     STORPOOL:   1         DBSPACE: SAMPLE                  backout: YES
                                  T A B L E
        Name : ACTIVITY                          Creator : SQLDBA
        ins  : I             no pages:       1   % pages :  100
ROW---- average len:    31   no.     :      18   overflow:    0
------------------------------ LIST OF INDEXES -------------------------

Index               Creator   seq  clust  unique  Lock   first   full  c.ratio
                                                   mode    key     key
Columns of index and sequence
PKEYCHQ2VS8QDSKD    SQLDBA    YES   YES    YES    PAGE      18      18  100,00
ACTNO(ASC)


*****************************************************************************
PF3:Return  PF4:Struct.  PF5:Synon.  PF7:Backward  PF8:Forward  PF12:Notes
```

Figure 50. SQL-Tune List of Indexes

The list of indexes gives us the following information about each index:

- The index name and creator
- Whether the index determines the insertion (clustering index)
- Clustered or not (= needs reorganization)
- Unique or duplicate (= multiple rows with same key are allowed)
- Lockmode of the index (Page or Key)
- The number of different values in the first column of the index
- The number of different values in the index
- Percentage of the index clusterisation (cluster ratio)

Indexes can be highlighted if the index duplicates part of another index on the same table:

```
INDEX1 on table ( COLA, COLB, COLC )
INDEX2 on table ( COLA )
```

In this case, the index called INDEX2 would be highlighted.

List of Critical Dbspaces (PF8)

This function displays all those dbspaces for which one of the following conditions apply:

- Statistics for this dbspace have never been updated
- This dbspace contains more than one table, and at least one table does not have an index on it

This list has the same presentation as Figure 46 on page 74.

List of Critical Tables (PF9)

This function displays all those tables for which one of the following conditions apply:

- Statistics have never been updated for this table
- No index exists on this table and there is more than one table in this dbspace
- The column NOVERFLOW of SYSTEM.SYSCATALOG indicates there are overflow pages (NOVERFLOW>0)
- There is no clustering index available on this table. This could happen if the clustering index would have been dropped.

This list has the same presentation as Figure 47 on page 75.

5.3.2.3 Database Monitoring Utility on VSE

This utility is used to monitor on-line and batch SQL activities and control the statistics capture.

Initiate Batch SQL Statistics Capture: Before you use the batch monitoring function, you must run program XTS1MONA to initiate statistics capture at first. There are two ways to execute XTS1MONA:

1. Execute XTS1MONA in a separate partition.
2. Execute XTS1MONA in the VTAM partition.

To capture the statistics of a batch application, a LIBDEF for the SQL-Tune library should be added to that JCL.

Run XTS1MONA in a Separate Partition

- To start batch statistics capture, submit the job to execute XTS1MONA in ***Fx*** or in a dynamic partition.

```
* $$ JOB JNM=XTS1MONA,DISP=L,CLASS=6
// JOB XTS1MONA
// LIBDEF *,SEARCH=(PRD2.XT1)    <----- SQL-Tune Library
// DLBL XTS1BAT,'XTS1BAT',,VSAM,CAT=VSESPUC
// DLBL XTS1STB,'XTS1STB',,VSAM,CAT=VSESPUC
// EXEC XTS1MONA,SIZE=AUTO
/*
* $$ EOJ
```

Figure 51. SQL-Tune VSE - Sample JCL to Run XTS1MONA in a Separate Partition

- To stop batch statistics capture, issue command ***MSG Fx*** on the console. This will terminate the job.

Execute XTS1MONA in the VTAM Partition

- Add a LIBDEF for SQL-Tune library in the VTAM startup job.
- Add DLBLs for the VSAM files XTS1BAT and XTS1STB in the VTAM startup job.
- Have at least 128K partition GETVIS available to execute XTS1MONA.
- To start batch statistics capture, issue the following command from the console:

  F NET, ATTACH, ID=XTS1MONA

- To stop batch statistics capture, issue the following command from the console:

  F NET, DETACH, ID=XTS1MONA

Monitor CICS SQL Activity (PF10)

Using this function, you can monitor (see the snapshot of) the on-line SQL activity. The following figure shows a sample of ongoing SQL activity:

```
XTS1
Date: 12/11/97                                          Time:  9:05:45
                          SQL  ACTIVITY


**************************************************************************
**  Taskno Tran  Term Act   Prep     Stmt    Time      User     Req.   **
**  00129             C                                SQLDBA          **
**  00148  CISQ  A006  S  ARIISQL        4    16,378   EL01       OPEN **
**                                                                     **
**                                                                     **
**                                                                     **
**************************************************************************
PF3:Return   PF5: Select
```

Figure 52. SQL-Tune VSE - Monitoring the CICS SQL Activity

The following information is shown:

- Execution date and time

for each transaction:

- CICS task number
- Transaction name
- Terminal id
- Type of activity (C: CICS; S: SQL)
- Name of the preprocessed package
- Statement number of preprocessed package
- Execution time so far

  Each time you press ENTER, it will be refreshed.

- DB2 user ID
- SQL request type currently used

Move the cursor to the line of an SQL request, and press PF5 to see the details. No details are provided for CICS activities (“C”).

```
XTS1
Date: 12/11/97                                              Time:  9:06:45
                          SQL  ACTIVITY
           User: EL01                              Taskno: 00148
           Tran: CISQ                              Term  : A006
*************************************************************************
**  Prep        Stmt   Req     No times     Duration  Dyn       Time   **
**  ARIISQL        4   PREP          1        0,138    YES              **
**  ARIISQL        4   DESCR         2        0,003    NO               **
**  ARIISQL        4   OPEN          1                 NO        76,870 **
**                                                                     **
**                                                                     **
**                                                                     **
**                                                                     **
*************************************************************************
PF3:Return   PF5: Select
```

Figure 53. SQL-Tune VSE - Detail of CICS SQL Activity

The following information is shown in the details:

- Name of the preprocessed package
- Statement number of preprocessed package
- Type of SQL request
- Total number of executions (how often)
- Total execution duration (if finished)
- Dynamic SQL statement or not
- Execution time so far (if not yet finished)

  It will be refreshed when you press ENTER.

Move the cursor to the line of the SQL request and press PF5, then you can inspect the related SQL statements shown below. This can only be done if the column, named Dyn, indicates **YES.** Otherwise from the Main Menu the Development function, *On-line Tuning*(PF2) should be used to display the statement.

```
XTS1
Date: 12/11/97                                                Time:  9:07:28
                              SQL  ACTIVITY
      User: ELO1        Taskno: 00148    Tran : CISQ     Term: A006
Prep: ARIISQL  Stmt:     4 Req: PREP  No times:    1 Duration:   119,476
********************************************************************************
**                              STATEMENT                                     **
** SELECT * FROM SYSTEM.SYSCATALOG, SYSTEM.SYSCOLUMNS, SYSTEM.SYSUSAGE **
** ORDER BY 14,2,9,5,4                                                        **
**                                                                           **
**                                                                           **
**                                                                           **
**                                                                           **
********************************************************************************
PF3:Return   PF5:On-line Tuning
```

Figure 54. SQL-Tune VSE - Detail of CICS SQL Statement

To tune the SQL statement, choose the On-line Tuning Option (PF5). For detailed information about how to tune an SQL statement, refer to 5.3.2.1, “Application Development Utility” on page 66.

Monitor Batch SQL Activity (PF13)

Using this function, you can monitor (see the snapshot of) the batch SQL activity.*:* The following figure shows a sample of ongoing batch SQL activity:

```
XTS1
Date: 12/17/97                                                Time:  9:05:45
                    BATCH  SQL  ACTIVITY

XTS1
********************************************************************************
** Partjobname act  prep    n¢    wait     user    duration  req    nb **
**  F5 DBSUPROD S ARIDSQL    3    18,414 CTABLINT           FETCH      **
**                                                                           **
**                                                                           **
**                                                                           **
**                                                                           **
**                                                                           **
********************************************************************************
PF3:Return   PF5: Select
```

Figure 55. SQL-Tune VSE - Monitoring the Batch SQL Activity

The following information is shown:

- Execution date and time

for each application:

- The name of the partition in which the job is running
- The job name
- What kind of resource the application is waiting for (for example, *'S'* means the application is waiting for DB2 processing)
- Name of the preprocessed package

- Statement number of preprocessed package
- The waiting time so far

  Each time you press ENTER, it will be refreshed.
- DB2 user ID used
- Total execution duration
- Type of SQL request
- Total number of executions (how often)

Move the cursor to the line of a batch job and press PF5 to see the whole batch job execution so far.

```
XTS1
Date: 12/17/97                                          Time:  9&conlon.06:45
                        Batch SQL Activity

************************************************************************
** Partjobname act  prep    n¢    wait     user    duration  req    nb **
**  F5 DBSUPROD   ARIDSQL    0                        0,000 CONN     1 **
**  F5 DBSUPROD   ARIDSQL   14                        0,001 PEXEC    1 **
**  F5 DBSUPROD   ARIDSQL    1                        0,000 COMIT    1 **
**  F5 DBSUPROD   ARIDSQL    0           CTABLINT     0,000 CONN     1 **
**  F5 DBSUPROD   ARIDSQL    3           CTABLINT     0,010 PREP     1 **
**  F5 DBSUPROD   ARIDSQL    3           CTABLINT     0,000 DESCR    1 **
**  F5 DBSUPROD   ARIDSQL    3           CTABLINT     0,001 OPEN     1 **
**  F5 DBSUPROD S ARIDSQL    3    48,425 CTABLINT     0,000 FETCH    1 **
**                                                                    **
**                                                                    **
**                                                                    **
**                                                                    **
**                                                                    **
************************************************************************
PF3: Return
```

Figure 56. SQL-Tune VSE - Detail of Batch SQL Activity

CICS Statistics Capture (PF11) and Batch Statistics Capture (PF14)

Using these two functions, you can switch statistics capture on and off.

5.3.2.4 Database Monitoring Utility on VM

On VM there will only be one panel showing all SQL activity on the database. The display has the same layout as Figure 52 on page 79. The only difference is that *Task number*, *Transaction name* and *Terminal id* are replaced by *VM Userid*.

Detailed information can also be requested and the shown panels have the same layout as in 5.3.2.3, “Database Monitoring Utility on VSE” on page 78.

To enable or disable the capture of statistics the option *Capture Statistics* (PF11) can be used. This will show you the next panel.

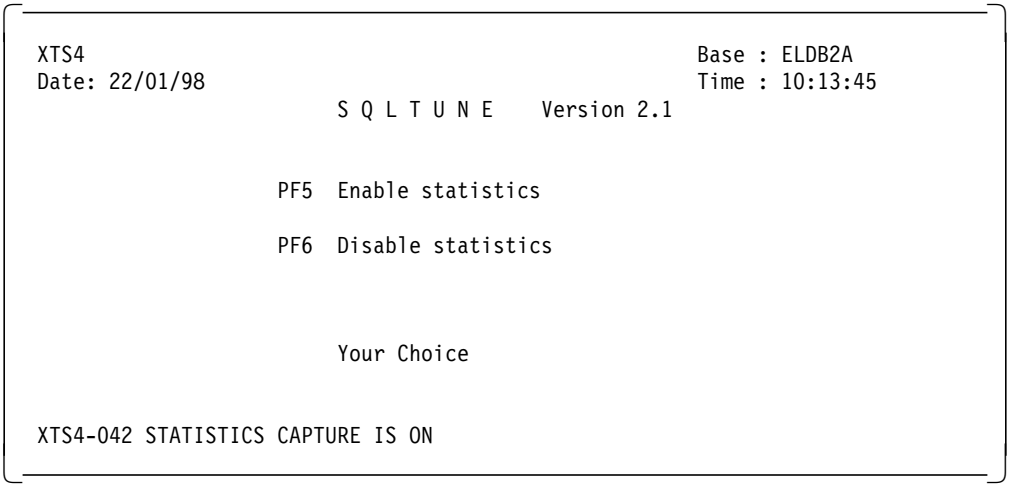


Figure 57. SQL-Tune VM - Statistics Capture

Exclusion/Inclusion (PF12)

You can use this function to exclude or include specific transactions from the statistics. This is rarely used because you might prefer to capture all statistics and select specific parts while you are tuning, as described in 5.3.2.1, “Application Development Utility” on page 66.

Chapter 6. VM:DB/Monitor

This chapter presents the installation and usage of VM:DB/Monitor.
VM:DB/Monitor is a comprehensive facility for monitoring and reporting on the performance of DB2 Server for VM. It collects data about how both applications and users access and use the DB2 databases as well as the resource consumption.

VM:DB/Monitor analyzes this information to produce more than 32 performance reports and 15 easy-to-read, real-time displays to enable a database administrator (DBA) to improve the database response times. VM:DB/Monitor does not react to crises, it anticipates them therefore enabling a DBA to prevent them. VM:DB/Monitor provides real time monitoring of database activity. All users' queries and application statements can be recorded. Statistics and on-line information can be displayed from predefined or customized panels. Offline reports can be created to have an overview in time or to keep history information.
The information from the reports can help you in tuning applications and adjust table or dbspace definitions.
The history reports can be very useful for capacity planning and trend analysis.

VM:DB/Monitor is a program offered by

STERLING Software Inc.
VM Software Division
1800 Alexander Bell Drive
Reston, Virgina 20191
USA
Tel: (+1 703) 264-8000
email vm@sterling.com

or visit:

http://www.vm.sterling.com

VM:DB/Monitor is also part of Sterling Software's VM:DBA package.

VM:DBA is an integrated package that automates and simplifies many database administrator functions such as:

- Archive and recovery
- Changing startup parameters
- Dbspace and table reorganization
- Database monitoring

VM:DBA also includes a Governor Facility for DB2 databases on VM. This facility allows the DBA to define a set of criteria, including resource thresholds and actions, to be automatically applied to users and applications running in the database. The Governor Facility and VM:DBA are not discussed within this document.

6.1 Features and Benefits

VM:DB/Monitor offers the DBA the following capabilities:

- Real-time monitoring (event-driven)
- Comprehensive historical reporting
- Customized displays and reports
- Session and LUW granularity
- Performing capacity planning and trend analysis
- Tuning of applications and tables
- Tuning of DB2 databases
- VMDSS & DRDA support
- Graphic IDBM overview display
- Access to the database CMS command line
- Pseudo-agent Force facility
- VSE guest-sharing support
- Does NOT require an agent
- Recording the usage of dynamic SQL statements
- Monitoring the thresholds of database activity
- Controlling the access to VM:DB/Monitor

6.2 Installation of VM:DB/Monitor

This section describes the installation of VM:DB/Monitor step by step. It is assumed that the reader is already familiar with VM systems.

The installation consists of the following steps:

- Prepare installation
- Install the VM:DB/Monitor from tape
- Customize and run VM:DB/Monitor

6.2.1 Prepare Installation

1. Acquire a password
2. Define VM user IDs

6.2.1.1 Acquire a Password

During the installation process you will be asked for the product password (CPUID). If you don't have one, please call your vendor representative.

6.2.1.2 Define VM User IDs

1. The second file on the tape contains directory samples for the three VM user IDs we will need. To unload them issue:

```
TAPE REW
TAPE FSF 1
TAPE LOAD * * A
```

2. Define the VM user ID which will receive the install procedure. The default name is RVMMAINT.
3. Define a VM user ID to run VM:DB/Monitor product. The default name is DBMONVM.
4. Optional: Define a VM user ID to run the VM:DB/Monitor Utility Processing. The default name is DBMONUP.

6.2.2 Install the VM:DB/Monitor from Tape

1. Attach a tape unit to the VM user ID which will receive the install procedure (Default is RVMMAINT).
2. Load the installation procedure from tape.

   ```
   TAPE REW
   TAPE LOAD * * A
   ```

3. Issue the RVMINST command from the CMS command line to begin the installation and follow the instructions on the screens. The installation procedure will ask you:
   - Are you upgrading from a previous release of VM:DBA?
   - Which components would you like from tape?
   - You can change the default minidisk addresses where the product will be loaded
   - Add the product password (CPUID)

6.2.3 Customize and Run VM:DB/Monitor

6.2.3.1 DBMON

For each DB2 database you want to monitor with VM:DB/Monitor, you must perform the following:

1. CP Directory changes to each DB2 database user ID
   - Give CP privilege class B (or the local equivalent) so that the VM:DB/Monitor DBMON component can issue ″message no-headers″ (CP MSGNOHs) instead of CP MSG in response to commands, making responses that require more than one line easier to read.
   - Optionally give CP privilege class E so that VM:DB/Monitor can collect virtual CPU usage data (via DIAG 4). If the database machine does not have Class E, all virtual CPU fields will be zero.
   - Add the ACCT VM directory option.
   - Increase MAXCONN by one, because DBMON communicates by IUCV to the DBMONVM virtual machine.
   - Optional: Add an MDISK for DETAIL recording.
2. Changes to PROFILE EXEC of each database
   - Access the VM:DB/Monitor Virtual Machine Code (RVMMAINT 250) minidisk with any file mode letter that comes before Q (DB2 Production Disk) in the search sequence.
   - Access the VM:DB/Monitor VM Common Material (RVMMAINT 200) minidisk with any file mode letter.

3. Create and customize a DBMON INITIAL file

```
*-------------------------
* DBMON Startup Definitions
*-------------------------
****************
* Issue Grants *
****************
GRANT CLASS ADM TO RVMMAINT
GRANT CLASS ADM TO DBMONUP
GRANT CLASS ADM TO ELRES1
GRANT CLASS ADM TO ELRES2
GRANT CLASS   M TO ELRES3

*****************
* Start Monitor *
*****************
START MONITOR TO DBMONVM INTERVAL 00:02:00

*************************
* Start Detail Recording *
*************************
*ACQUIRE DETAIL DISK * 100 MR
*START   DETAIL TO DETPDB TODAY CLOSE 1000 NOTIFY DBMONUP

*****************************
* Activate Detail Recording *
*****************************
```

Figure 58. DBMON INITIAL File

4. Change the database startup parameters
   - Modify or add the database initialization parameter ACCOUNT=D.

6.2.3.2 DBMONVM

1. Customize the PROFILE EXEC in the DBMONVM user ID
   - Add the DBMONVM command to the PROFILE EXEC to bring up DBMONVM automatically.
   - Access the VM:DB/Monitor VM Common Material minidisk (RVMMAINT 200) and the VM:DB/Monitor Virtual Machine Code (RVMMAINT 250).
2. Customize the DBMONVM INITIAL file

```
*---------------------------
* DBMONVM Startup Definitions
*---------------------------
ACQUIRE LOG       DISK DBMONVM 1DO MR * FM L
ACQUIRE RECORDING DISK DBMONVM 1D1 MR * FM R
LOG    SQLMACH EVERY 30 TO LOGSQL FM L CLOSE  10 NOTIFY DBMONUP
LOG    ELDB2A  EVERY 30 TO LOGTDB FM L CLOSE 500 NOTIFY DBMONUP
RECORD SQLMACH TO RECSQL FM R CLOSE    50 NOTIFY DBMONUP
RECORD ELDB2A  TO RECTDB FM R CLOSE   500 NOTIFY DBMONUP
GRANT CLASS AEMT TO RVMMAINT AT BOEVMCT1
GRANT CLASS AEMT TO DBMONUP  AT BOEVMCT1
GRANT CLASS AMT  TO ELRES1   AT BOEVMCT1
GRANT CLASS M    TO ELRES2   AT BOEVMCT1
RESTRICT ELRES1  AT BOEVMCT1 TO ELDB2A
THRESHOLD ELDB2A  R190 ADD EXIT(DBCUNLK R190 VM * 00:00:10 )
THRESHOLD ELDB2A  R190 ON
THRESHOLD ELDB2A  R191 ADD EXIT(DBCFORC R191 VM ELRES3  00:00:10 ROLLBACK )
THRESHOLD ELDB2A  R191 ON
```

Figure 59. DBMONVM INITIAL File

6.2.3.3 IDBM

To use the IDBM component in a user virtual machine you must:

1. Access the VM:DB/Monitor VM Common Material minidisk (RVMMAINT 200).
2. Enter the IDBM command in the CMS command line

Now you have access to the predefined screens or you can create your own customized screens.

Examples of the screens, together with some related description are shown in Appendix B, “Screens and Reports of VM:DB/Monitor” on page 147. It also contains a sample program for a user defined screen.

6.2.3.4 DBMAP

The following list will show you in a brief form how to create activity and monitor reports.

1. DBMONVM machine: link to the minidisk that will contain the activity or log files. Use the ACQUIRE command.
2. DBMONVM machine: start the recording. Use the RECORD or LOG command.
3. The file is only available after it is closed by DBMONVM, so if you need to process it before, you must close it using the CLOSE command.
4. Reaccess the DBMONVM minidisk.
5. Create the DBMAP specification file with the following records:
   - INPUT: to select which files you want to process (Mandatory).
   - RANGE: if you want to select only a specific period of time recorded in the files (Optional).
   - SELECT: selection criteria for the data to be included in the reports (Optional).
   - REPORT: identifies the report to be run and how it is to be printed (Mandatory).

The default name of the DBMAP specification file is DBMON DATA. The following figure is a sample of the DBMAP specification file.

```
*********************************************************************
* The DBMAP will proccess file RECDBA 980129AN T
INPUT RECDBA 980129AN T
*
* Date and Time range
RANGE FROM 01/28/98 16:41:00 TO 01/28/98 16:42:00
*
* Select only entries of user ID ELRES5
SELECT USERID ELRES5
*
* Create the PACKAGES activity report and sort the eleventh column
*   in descending order
REPORT PACKAGES SORT 11 D
*
* Create the ACTDETL activity report
REPORT ACTDETL
```

Figure 60. DBMAP Specification File

6. Access the VM:DB/Monitor VM Common Material minidisk (RVMMAINT 200).
7. Enter the DBMAP command in the CMS command line.

6.3 Components of VM:DB/Monitor

There are five components of VM:DB/Monitor

- DBMON
- DBMONVM
- IDBM
- DBMAP
- DBMONUP

The next figure shows the relationship between the VM:DB/Monitor components.

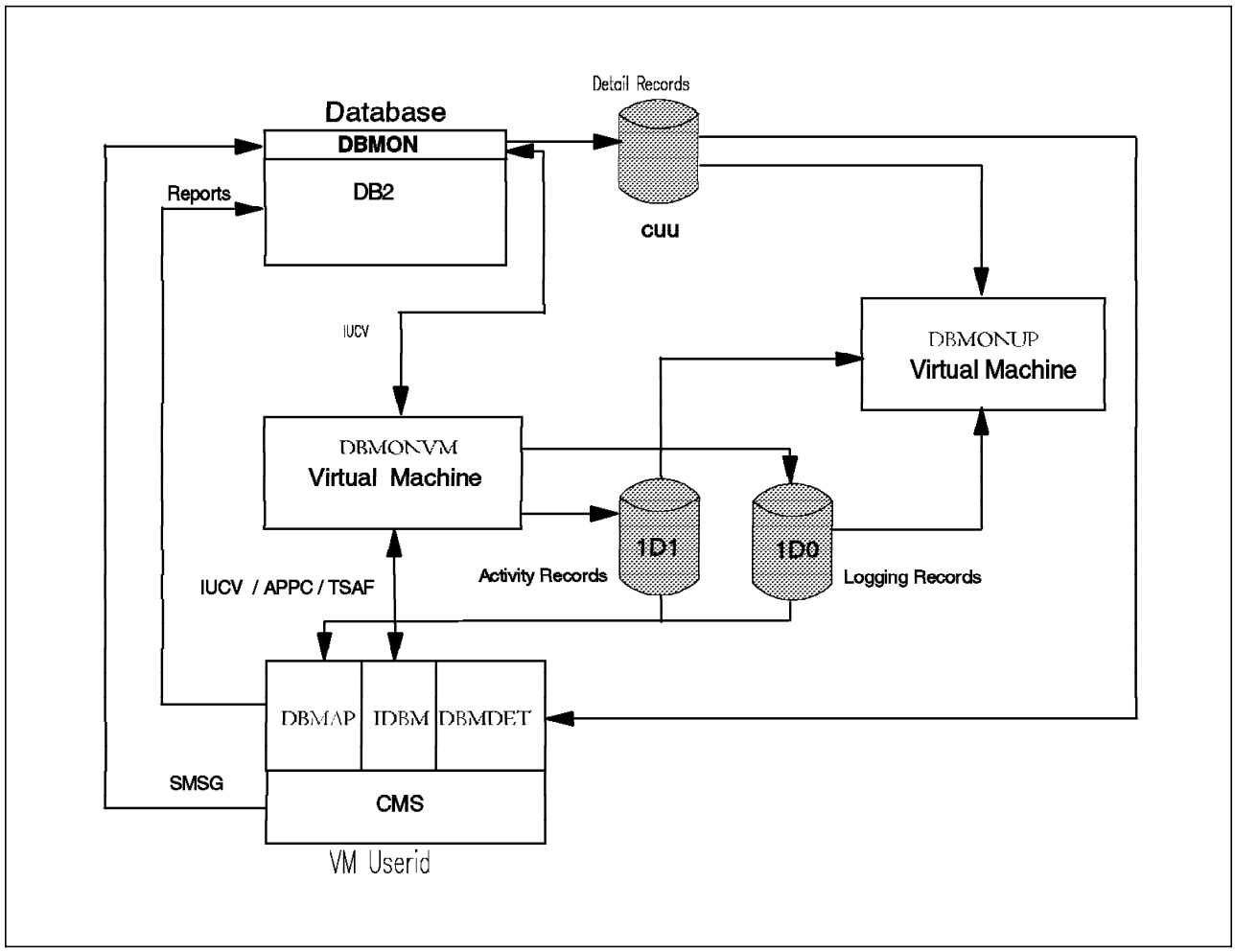


Figure 61. Components of VM:DB/Monitor

6.3.1 DBMON

DBMON is the primary data collection component of VM:DB/Monitor. It runs in the DB2 database machine as a CMS nucleus extension, co-existing with DB2 itself. It collects data from DB2, CMS and CP control blocks to produce monitor and activity records. DBMON creates monitor records either at timed intervals that the DBA specifies or upon request. DBMON creates activity records for each user's activity at the LUW level. These records are sent to the DBMONVM virtual machine via an IUCV connection.

6.3.2 DBMONVM

DBMONVM is the key VM:DB/Monitor component for real-time monitoring of DB2 database machines. DBMONVM is the name of the module that runs in the DBMONVM virtual machine.

A single DBMONVM virtual machine can handle any number of DB2 database machines on the same VM system. DBMONVM:

- Receives monitor records from DBMON, analyzes this data and prepares it for use by users of IDBM.
- Receives activity records from DBMON and manages the activity recording files.

- Summarizes monitor records and writes them to a log file.
- Notifies a specified list of users if predefined thresholds are met.

6.3.2.1 Thresholds

The monitor virtual machine can be set up to monitor if a predefined condition is met or a threshold has been exceeded.
This can be defined by two different methods:

1. Define the threshold in the DBMONVM INITIAL file. This trigger will only notify a predefined user that the trigger has been detected.
2. Define an exit routine that will be executed at each monitor interval.

Notify a Predefined User

This is an example showing how to set up a trigger which will notify a predefined user that a trigger has been detected. The definition has to be coded in the DBMONVM INITIAL file, which keeps all commands to be executed at startup of the monitor virtual machine. See Figure 62 for the additional statements coded in the DBMONVM INITIAL file to define the threshold. In this example we defined:

- Database ELDB2A will be monitored
- TR2 as threshold identification name. More can be defined, but the names must be unique.
- ADD is a reserved keyword, specifying the next expression should be added to the threshold list.
- HWMPAREAL > 2
  This expression means that the value of pseudo agents connected to a real agent should be monitored and if this value is greater than 2, this trigger is activated.
- NOTIFY ELRES5
  Notification of the trigger will be done to user ID ELRES5
- Activate the trigger with TR2 ON

```
  ......
THRESHOLD ELDB2A TR2 ADD HWMPAREAL > 2
THRESHOLD ELDB2A TR2 NOTIFY ELRES5
THRESHOLD ELDB2A TR2 ON
  .......
```

Figure 62. DBMONVM INITIAL File Setting Threshold and Notify

The next figure shows the message that was received by the user ID specified in the *NOTIFY Userid* statement.

```
 10:04:48  * MSG FROM ELDBMONV: PMVTHR021I The TR2 threshold for
             database ELDB2A AT BOEVMCT1 is true.
```

Figure 63. DBMONVM Notify Message

The notified user ID only gets the name of the defined trigger. This user ID needs to look up which criteria have been defined for this condition.

Monitor Exit Routine

This example shows how to define an exit routine that will be executed at each monitor interval. This REXX exec procedure can check predefined values and notify users or execute some statements when a condition has been detected. The next figure shows the additional statements to the DBMONVM INITIAL file to define an exit routine.

```
  ......
THRESHOLD ELDB2A TR1 ADD EXIT(TR1EX)
THRESHOLD ELDB2A TR1 ON
  .......
```

Figure 64. DBMONVM INITIAL File Setting Threshold and User Exit

In this example we defined:

- Database ELDB2A will be monitored
- ADD is a reserved keyword, specifying the next expression should be added to the threshold list.
- EXIT(TR1EX)
  This defines the exit routine **TR1EX**
- Activate the trigger with TR1 ON

```
/*-------------------------------------------------------------------*/
/* THRESHOLD MONITOR CALL ON DBMONITOR EXIT ------------------------*/
/*-------------------------------------------------------------------*/
"PMVEXTR DB.HWMPAREAL ACTUSER"       /* EXTRACT HWM PA-RA CONNECTED */
"PMVEXTR DB.NCUSERS   MAXUSER"       /* EXTRACT NCUSERS PARAMETER   */
CURUSER = FORMAT(100/(MAXUSER/ACTUSER),3,0)
Select
 When CURUSER >= 75 Then do
   "CP MESSAGE ELRES5 75% OF "MAXUSER" NCUSERS IS IN USE"
   Retc = 1
 End
 When CURUSER >= 50 Then Do
   "CP MESSAGE ELRES5 50% OF "MAXUSER" NCUSERS IS IN USE"
   Retc = 1
 End
 When CURUSER >= 5  Then Do
   "CP MESSAGE ELRES5 5% OF "MAXUSER" NCUSERS IS IN USE"
   Retc = 1
 Otherwise Retc = 0
End
EXIT(retc)
```

Figure 65. VM:DB/Monitor Sample Monitor Exit Routine

Explanation:

- PMVEXTR DB.HWMPAREAL ACTUSER
  From the database records, the field HWMPAREAL will be extracted, which keeps the value of pseudo agents connected to a real agent. ACTUSER is the name of a variable in this routine to keep this value.
- PMVEXTR DB.NCUSERS MAXUSER
  From the database records, the field NCUSERS will be extracted, which keeps the value of the NCUSERS initialization parameter. MAXUSER is the name of a variable in this routine to keep this value.
- The *CP MESSAGE* command will be executed for predefined conditions.

This is a sample of the message the user ID received when the previous condition is evaluated as true.

```
10:02:48  * MSG FROM ELDBMONV: 5% OF 20 NCUSERS IS IN USE
```

Figure 66. VM:DB/Monitor Output of Monitor Exit Routine

6.3.3 IDBM

IDBM is the interactive user interface for real-time monitoring. It runs in the user virtual machine and is the user interface to VM:DB/Monitor real-time monitoring function. IDBM users can monitor the activity of selected databases by refreshing the screen each time new data is received. They can also extract the monitor data into REXX variables for further analysis.
There are 15 predefined screens available.

6.3.3.1 Predefined Screens

Screen	Function
BUFFERS	Page Buffers by Dbspaceno
COUNTERS	DB2 Counter Values
DATABASE	Monitored Databases
DBSTATS	Database Statistics
DSCNTRS	VMDSS Counter Values
EXTENTS	DB Extent I/O Activity
OVERVIEW	Performance Overview
PAGENTS	Pseudo-Agent Information
PERFORM	Performance Summary
POOLS	Storage Pool I/O Activity
RAGENTS	Real-Agent Information
SQLLOG	DB2 Log Usage
SYSSTATS	System Statistics
TOPUSERS	Top Users of Virtual CPU
USER	Detailed User Data

See **B.1, "IDBM Sample Screens" on page 147** for an example for each.

6.3.4 DBMAP

DBMAP is the VM:DB/Monitor analysis program. It runs in a user virtual machine, and any user with access to the proper input files can execute it.

DBMAP reads VM:DB/Monitor data files and produces output in the form of reports or history files.

6.3.4.1 Reports

You can either use the predefined reports or build your own programs to access the activity and monitor files.

Type of Reports

DBMAP can read the following VM:DB/Monitor data files:

- Activity files created by the DBMON component
- Monitor Log files created by the DBMONVM component
- History files created by the DBMAP component

There are 32 predefined reports divided into three groups:

1. ACTIVITY (16)
2. MONITORING (13)
3. DETAIL (3)

Activity Reports

Activity reports are based on records created whenever there is activity in the database caused by a user performing some SQL function.*:* Activity records have detailed information about each session, LUW, and package within the database. You use VM:DB/Monitor to collect activity records that you can analyze at a later time. The data provided by the activity records is used for real-time monitoring and is helpful for debugging, benchmarking, and detailed tracing of DB2 applications. You use VM:DB/Monitor′s DBMAP component to read the activity records and create reports from them.

Predefined Activity Reports

Name	Contents
ACTDETL	Activity Detail
ACTVTIME	User Activity Across Time
DEADLOCK	Deadlock Occurrences
DYNSQL	Dynamic SQL Statement Usage
LUWSTATS	LUW Statistics
PACKAGED	Package Usage and Resource Consumption Detail
PACKAGES	Package Usage and Resource Consumption Summary
SECTIONS	Package Section Usage and Resource Consumption
SECTSTAT	Package Section Statistics

SESSIOND	Session Statistics Detail
SESSIONS	Session Statistics Summary
TABNDXS	Table and Index Usage Summary
TABNDXU	Table and Index Usage
USERRES	User Resource Consumption
USERTIME	User Activity Across Time
WAITLOCK	Waitlock Occurrences

See **B.2, "DBMAP Activity Reports" on page 166** for an example for each.

Monitor Log Reports

Monitor reports are based on records created on a timed basis or upon request. Monitor records are provided primarily for real-time monitoring of the database; however, you can also log the database statistics records and use the data they provide for historical analysis.

Predefined Monitor Log Reports

Name	Contents
BUFFERS	DB2 Page Buffer Utilization
COUNTERS	DB2 Counters
DBSTATS	Database Statistics
DSCNTRS	DB2 Dataspace Counters
EXTENTS	I/O Activity by DBEXTENT
HISTORY	Create History File
LOGDETL	Log File Detail
PERFORM	Performance Summary
POOLS	I/O Activity by Storage Pool
SQLLOG	DB2 Log Usage
SYSTATS	System Statistics
THRESHOLD	Threshold Report
TOPUSERS	Top Users of Virtual CPU

See **B.3, "DBMAP Monitor Reports" on page 183** for an example for each.

Detail Reports

Detail reports are based on records with detailed information about each transaction performed by the database on behalf of an end user, such as OPEN, FETCH or PREPARE. Detail records are only created for a specific list of user IDs and DB2 connect IDs. The DBMON component writes the detail records to an accessed disk on the database user ID. You use VM:DB/Monitor's DBMDET utility to read the detail records and create reports from the data they contain.

Detail reports can be useful for debugging and detailed tracing of DB2 applications. However, since collecting detail records adds a very high level of

overhead to a database, it should not be used for standard production level monitoring.

Predefined Detail Reports

Name	Contents
DETAIL	Transaction Detail
TABNDXD	Table and Index Usage Detail
TRANSTAT	Transaction Statistics

See **B.4, "DBMAP Detail Reports" on page 198** for an example for each.

Reports Usage

The following list helps to determine which reports should be used to find information related to:

- CPU
  - Activity Reports
    - ACTDETL: database CPU consumption of each LUW executed
    - DYNSQL: database CPU consumption of each dynamic SQL statement executed
    - PACKAGES: database CPU consumption of each package executed
    - SECTIONS: database CPU consumption of each package section executed
    - SESSIOND: database CPU consumption of each VM user ID that was connected to the database
    - LUWSTATS: list of the five LUWs consuming most of the database CPU time
    - USERRES: database CPU consumption of each VM user ID that was connected to the database
  - Monitor Log Reports
    - SYSSTATS: average, minimum, maximum and the evolution of the database CPU consumption during a period
  - Detail Reports
    - DETAIL: For each SQL statement traced:
      - CPU consumption
      - Number of DBSS calls
    - TRANSTAT: top five transaction DB2 opcodes (RDIIN) ordered by database CPU consumption
- I/O
  - Activity Reports
    - ACTDETL: database I/O consumption of each LUW executed
  - Detail Reports
    - TRANSTAT: top five transaction DB2 opcodes caused most I/O activity
  - Monitor Log Reports

        - COUNTERS: average, minimum, maximum, total and the evolution of the consumption of checkpoints, lock escalation, dbspace page reads and writes, directory page reads and writes, log reads and writes, total DASD reads and writes, dataspace reads and writes, total I/O during a period
        - EXTENTS: average, minimum, maximum, total and the evolution of the I/O activity in each database dbextent during a period
        - DBSTATS: average, minimum, maximum, total and the evolution of the I/O activity of the database during a period
        - SQLLOG: average, minimum, maximum, total and the evolution of the I/O activity on the database log file during a period
        - POOL: I/O activity on directory, internal dataspaces and on each storage pool.
- Buffers
  - Activity Reports
    - LUWSTATS: top five LUWs that make most looks in the page buffer pool
    - DYNSQL: number of looks in the page buffer pool for each dynamic SQL statement executed
    - PACKAGES: number of looks in the page buffer pool for each package executed
    - SECTIONS: number of looks in the page buffer pool for each package section executed
    - SESSIOND and USERRES: number of looks in the page buffer pool for each VM user ID
  - Monitor Log Reports
    - COUNTERS: average, minimum, maximum, total and evolution of page and directory buffer looks during a period
    - DBSTATS: average, minimum, maximum, total and evolution of page and directory buffers allocated during a period
    - SYSSTATS: database page buffer looks total and the evolution of consumption during a period
    - PERFORM: average, minimum, maximum, total and evolution of effective use of the page and directory buffers during a period
    - BUFFERS: which dbspaces used the page buffer pool most
  - Detail Reports
    - DETAIL: number of looks in the page and directory buffers executed by each SQL statement
    - TRANSTAT: top five transaction DB2 opcodes ordered by looks in the page and directory buffers
- Storage
  - Monitor Log Reports
    - DBSTATS: average, minimum, maximum and the evolution of the number of programs loaded in storage

        - SYSSTATS: average, minimum, maximum and the evolution of the database working set, working-storage target, resident pages in auxiliary and expanded storage and storage utilized during a period

- Database Accounting
  - Activity Reports
    - USERRES: Total consumption of database CPU time and page buffer activity for a VM user ID
    - USERTIME: CPU time consumption behavior of each VM user ID during a period
- Dynamic SQL
  - Activity Reports
    - DYNSQL: For each dynamic SQL statement executed:
      - Total database resource consumption
      - SQLCODE and SQLSTATE returned
      - Dynamic SQL statement executed
      - Name of the VM user ID that used the statement
- Application Debugging
  - Activity Reports
    - ACTDETL
      - Number of times an SQL statement was executed
      - Last SQLCODE and SQLSTATE returned to each SQL statement executed
      - Dbspaces, tables and indexes the application accessed
      - Database response time
- Locks
  - Activity Reports
    - PACKAGED: program name, VM user ID and time ordered by longest LUW time.
    - WAITLOCK: starting time, elapsed time, VM user ID and type of requester, the VM user ID, state and dbspaceno of the holding agent of each lock conflict occurred.
  - Monitor Log Reports
    - DEADLOCK: time, losing VM user ID, LUW id, LUW start time, mode of lock held on each resource, winning VM user ID, LUW id, LUW start time, mode of lock needed on each resource, dbspace number of each deadlock occurred.
    - COUNTERS: average, minimum, maximum, total and the evolution of lock escalates, waitlocks and deadlocks during a period.
    - DBSTATS: average, minimum, maximum, total and the evolution of lock request blocks (LRB) during a period.
    - PERFORM: average, minimum, maximum, total and the evolution of the number of waitlocks per RDSCALL and deadlocks per LUW during a period.
  - Detail Reports

        - DETAIL: number of database locks used to execute each SQL statement traced.

- Table and Dbspace
  - Activity Reports
    - TABNDXU: number of LUWs executed by each VM and SQL id referring to the table, index and dbspace.
    - TABNDXS: total number of LUWs that used a table, index and dbspace.

6.3.4.2 VM:DB/Monitor REXX Interface

VM:DB/Monitor also provides a REXX interface for reading the VM:DB/Monitor output files. This feature allows you to perform customized data analyses and to produce customized offline reports. With the interface you can, for example, write a REXX EXEC to read an activity file to create your own customized report. There are two routines you must use to read an activity file:

1. XDBMAP module
2. XDBMAP REXX function

XDBMAP Module

XDBMAP module loads the XDBMAP REXX function. This function must be loaded as a CMS nucleus extension prior to execution.

XDBMAP REXX Function

The function reads the activity and monitor log files. There are three steps to read a file:

1. OPEN the file
2. READ one or more records
3. CLOSE the file

The READ operation returns the records into REXX variables. The files ACTIVITY FORMATS and MON_LOG FORMATS on the VM:DB/Monitor public software minidisk have a list and a brief description of each variable name set.

Example

The following sample REXX program shows how to process an activity file. It reads an activity file and writes a CMS file with I/O statistics about each package executed.

```
/* -------------------Sample REXX program-------------------------- */
'PIPE CMS ERASE DBMAPPK RESULT A'  1
out_line='      RECTIME      NAME         I/O'
'PIPE LITERAL' out_line '| > dbmappk result a'  2
'NUCXLOAD XDBMAP'  3
'XDBMAP LOAD'  4
xrc=xdbmap('OPEN','RECDBA 980129AN T')  5
do while xrc = 0  6
  xrc = xdbmap('READ','REC.')  7
  if xrc<>0
  then leave
  select  8
     when REC.ID='APK' then call apk_record  9
     otherwise nop
  end
end
xrc = xdbmap('CLOSE')  10
'XDBMAP DROP'  11
exit 0
apk_record:  12
REC.RECTIME=left(REC.RECTIME,17)  13
do index=1 to REC.NPACKAGES  14
  REC.PACKAGES.index.IOTIME=format(REC.PACKAGES.index.IOTIME,10,0)  15
  out_line=REC.RECTIME REC.PACKAGES.index.NAME REC.PACKAGES.index.IOTIME
  'PIPE LITERAL' out_line '| >> dbmappk result a'  16
end
return
```

Figure 67. XDBMAP REXX Program Example

Notes:

1 Clean the output file

2 Write a header in the new output file

3 Load the XDBMAP module as a CMS nucleus extension

4 Load the XDBMAP REXX function

5 Open the activity file

6 Call then READ XDBMAP function to read each activity record

7 Set the variable REC. with fields of the record. You can use the command

```
PIPE REXXVARS | >> fn ft fm
```

to store all variables with their names and values created by the READ command into a CMS file.

8 Select a specific types of activity record

9 The REC.ID (record identification) variable is common for all type of activity records and contains the type of record, for example APK

10 Close the activity file

11 Drop the REXX function

12 Routine to process the package record

13 Save the record generation time

14 Access an array with the statistics about each package used by the LUW

15 Format the output fields

16 Write the output record

6.3.5 DBMONUP (Optional)

The DBMONUP facility is used to prevent the monitor disks from becoming full which would result in loss of data. Whenever the defined threshold is reached, DBMONVM sends a NOTE to this user. The PROP will receive this NOTE and copy the necessary file to another disk or sfs-area and then delete the original file on the monitor disk. Add the command `DBMONUP (DISC` to the PROFILE EXEC of the DBMONUP user ID to bring up the CMS Programmable Operator application automatically and add the DBMONUP user ID to your list of users to be autologged each time you IPL your system.

Part 3. DB2 Tuning

This part offers practical hints and tips about how to tune a database and applications using the database. Also, comprehensive recommendation lists are offered and checklists to analyze performance problem causes.

Chapter 7. Application Development

This chapter describes the steps typically done during development of a new SQL application. Additionally, it describes recommended procedures, test environments and responsibilities.

7.1 Test Databases

A test database should always be created in a test environment if possible. It would even be better to have two test databases.

1. A function test database, that would only contain a minimal quantity of data. The setup of the tables should reflect the reality, but it is not important where tables are defined in dbspaces or storage pools. There is even no necessity to define all indexes on these tables.
2. A simulation database would be used by application programmers to test the correctness of SQL statements by preprocessing their programs and check the application flow.

   This simulation test database should as much as possible reflect the real, production database. In this database you need to do the overall testing of application groups and see how an application behaves if another application has modified some data. In this database you can also verify the access path selected by the optimizer and have the applications or database objects modified according to the findings. If this database resides in a separated test environment you could also reproduce other files, such as VSAM from your production environment, to have a complete test system for on-line and batch applications with actual data. If testing is done in such an environment, try to have some concurrency of applications running together. This could help you in resolving locking problems or deadlock situations.

7.1.1 Defining a Simulation Test Environment

You should define a simulation test environment isolated as much as possible from the production environment.

Here are some recommendations, from the ″best″ to the ″worst″.

1. You have duplicate hardware: replicate your system.
2. You have two different processors and duplicate DASD: replicate your system.

   Note: When measuring performance of new applications, consider the difference between your systems.
3. You have two *different* processors and either *duplicated* DASD or at least enough space to replicate everything. In this case, proceed as step 2.
4. You have two *different* processors and *not* enough space to duplicate everything. In this case, try to extract a subset from the production system. For DB2 databases, update the catalog, setting the statistics to production values so that you can see the optimizer′s access path choice using *EXPLAIN* or SQL-Tune. Having DBA authority, you are allowed to update the statistical values stored in the catalog tables. See the paragraph *Keeping Database Statistics Current* in Chapter 5 of “*DB2 Server for VSE & VM Performance Tuning Handbook*” and 7.1.1.2, “Statistics” on page 106.

5. Whether you have a different system or not, for test purposes, you should have at least a test database that can be isolated from the production database.

Remember: The test environment is **never** identical to the production environment.

7.1.1.1 VSE only

You don't have two machines.
If running native VSE, LPAR should be used to dedicate the system resources to the different VSE systems.
If running VSE as a guest of VM, another virtual machine has to be defined to run this new VSE system.
Then generate another VSE system and proceed as4 on page 105 .
If you cannot have two different VSE systems in the same machine: define another CICS, VSAM catalog and another database.

In this case, you should:

1. Use different VSE and ICCF libraries.
2. Use different VSE sublibraries for your DB2 code, so that you can apply service for test purposes without impacting the production system.
3. Use different jobs to compile in test and compile in production.
4. Use different ways for running something in test and in production. For example: use different ICCF macros to submit jobs to execution, different jobs with significant different names in both systems and so on.
5. Use different CICS user IDs in each system.
6. Use static partitions for your production subsystems and dynamic partitions for your test subsystems, with less priority and less resources.
7. Instruct the operations staff about this difference.
8. Do **not** use the same "dbname directory" (ARISDIRD.A) for your databases. Do **not** specify in one directory the other database.
9. Proceed as step 4 on page 105.

7.1.1.2 Statistics

Because in some cases it is not possible to have all production data available in a simulation test system and you still want to analyze the application behavior as executed in your production database, you could copy some statistics from you production database to your test database.

Figure 68 on page 107 gives some SQL statements that were used to extract data from the catalogs. The output of these has been used as input to generate a DBSU job for updating the catalogs in the test database. The sample DBSU job as in Figure 69 on page 108 is based on the table *SQLDBA.DEPARTMENT* in dbspace *SAMPLE*.

A Warning about Updating Statistics

Refer to the paragraph *A Warning about Updating Statistics* in Chapter 5 of "*DB2 Server for VSE & VM Performance Tuning Handbook*." This chapter describes relations that exist between columns in the database catalogs which should be updated together. It also describes the contents of some specific columns.

```
Information about Dbspaces
 SELECT NPAGES,   NRHEADER,    PCTINDX,    FREEPCT,
        LOCKMODE, DBSPACETYPE, NACTIVE,    NTABS,
        OWNER,    DBSPACENAME, DBSPACENO ′
           FROM SYSTEM.SYSDBSPACES ′
              WHERE DBSPACENO = :H001 AND
                 OWNER ¬= ′′

Information about Tables
 SELECT CREATOR, TNAME,
        NCOLS,   AVGROWLEN, ROWCOUNT, NPAGES, PCTPAGES
           FROM SYSTEM.SYSCATALOG
              WHERE (DBSPACENO = :H001 AND
                     TABLETYPE = ′R′)

Information about Columns
 SELECT CNAME,    COLTYPE,  LENGTH,  NULLS,     COLNO,
        COLCOUNT, HIGH2KEY, LOW2KEY, AVGCOLLEN, COLINFO
           FROM SYSTEM.SYSCOLUMNS
              WHERE TNAME = :H001 AND CREATOR = :H002
              ′ORDER BY COLNO′

Information about Indexes
 SELECT I.INAME,     I.ICREATOR,      I.TNAME,        I.CREATOR,
        I.INDEXTYPE, I.IPCTFREE,      I.COLNUMBERS,
        I.KEYLEN,    I.FIRSTKEYCOUNT, I.FULLKEYCOUNT,
        I.NLEAF, I.NLEVELS, I.CLUSTER, I.CLUSTERRATIO
           FROM SYSTEM.SYSCATALOG C, SYSTEM.SYSINDEXES I
              WHERE (C.DBSPACENO = :H001 AND
                     C.CREATOR = I.CREATOR AND
                     C.TNAME = I.TNAME )

Information about Column Statistics
 SELECT VAL10,    VAL50,    VAL90,
        FREQ1VAL, FREQ1PCT, FREQ2VAL, FREQ2PCT
           FROM SYSTEM.SYSCOLSTATS
              WHERE (TNAME   = :H001 AND
                     CREATOR = :H002 AND
                     CNAME = :H003 )
```

Figure 68. SQL Statements for Statistics Capture

```
UPDATE SYSTEM.SYSDBSPACES SET NACTIVE=6 WHERE DBSPACENAME='SAMPLE' AND
 OWNER='PUBLIC';
UPDATE SYSTEM.SYSCOLUMNS SET COLCOUNT=9, HIGH2KEY='E11     ',
 LOW2KEY='B01     ', AVGCOLLEN=3 WHERE CREATOR='SQLDBA' AND
 TNAME='DEPARTMENT' AND CNAME='DEPTNO';
UPDATE SYSTEM.SYSCOLSTATS SET VAL1O='A00     ', VAL5O='D11     ',
 VAL9O='E21     ', FREQ1VAL='A00     ', FREQ1PCT=11, FREQ2VAL='E21
 ', FREQ2PCT=11 WHERE CREATOR='SQLDBA' AND TNAME='DEPARTMENT' AND
 CNAME='DEPTNO';
UPDATE SYSTEM.SYSCOLUMNS SET COLCOUNT=-1, HIGH2KEY='            ',
 LOW2KEY='            ', AVGCOLLEN=-1 WHERE CREATOR='SQLDBA' AND
 TNAME='DEPARTMENT' AND CNAME='DEPTNAME';
UPDATE SYSTEM.SYSCOLUMNS SET COLCOUNT=9, HIGH2KEY='000100 ',
 LOW2KEY='000020 ', AVGCOLLEN=7 WHERE CREATOR='SQLDBA' AND
 TNAME='DEPARTMENT' AND CNAME='MGRNO';
UPDATE SYSTEM.SYSCOLSTATS SET VAL1O='000010 ', VAL5O='000060 ',
 VAL9O=' ', FREQ1VAL='000010 ', FREQ1PCT=11, FREQ2VAL='
 ', FREQ2PCT=11 WHERE CREATOR='SQLDBA' AND TNAME='DEPARTMENT' AND
 CNAME='MGRNO';
UPDATE SYSTEM.SYSCOLUMNS SET COLCOUNT=-1, HIGH2KEY='            ',
 LOW2KEY='            ', AVGCOLLEN=-1 WHERE CREATOR='SQLDBA' AND
 TNAME='DEPARTMENT' AND CNAME='ADMRDEPT';
UPDATE SYSTEM.SYSCATALOG SET AVGROWLEN=39, ROWCOUNT=9, NPAGES=1,
 PCTPAGES=17 WHERE CREATOR='SQLDBA' AND TNAME='DEPARTMENT';
UPDATE SYSTEM.SYSINDEXES SET KEYLEN=3, FIRSTKEYCOUNT=9, FULLKEYCOUNT=9,
 NLEAF=1, NLEVELS=1, CLUSTER='F', CLUSTERRATIO=10000 WHERE
 CREATOR='SQLDBA' AND INAME='PKEYCHIWKILF8KE7';
UPDATE SYSTEM.SYSINDEXES SET KEYLEN=7, FIRSTKEYCOUNT=9, FULLKEYCOUNT=9,
 NLEAF=1, NLEVELS=1, CLUSTER='C', CLUSTERRATIO=10000 WHERE
 CREATOR='SQLDBA' AND INAME='MGRNOI';
```

Figure 69. SQL Statements for Statistics Update

7.2 Defining the Enterprise Rules

Each company should define rules to manage data and a database. Applications that are built according to these rules also provide an increased level of consistency and connectivity across platforms and enable an enterprise to make the best use of its resources.

An enterprise should have:

1. Naming conventions for each type of DB2 object. See Chapter 3 of the "*DB2 Server for VSE & VM SQL Reference*."
2. Naming conventions for the application programs, mapsets, maps and in VSE also for transactions, TS queues, RDO groups.
3. Coding conventions for the application programs, including flow standards, variable naming conventions, inter-application communications standards and so on.
4. Map standards, including PFkey usage, location of data within the map, highlight or underlining of objects within it, etc.
5. Preprocessing, compiling and link-editing parameters.
6. Security rules for accessing the data, basically when you must share information between different applications.

7. Cleanup rules for the DB2 tables and entities.

7.3 Application Development Steps

7.3.1 Overview

Firstly, the steps are given briefly in form of a list. Some of them are explained in the following sections in more detail.

Abbreviation:

AP Application Programmer

DA Data Administrator

DBA DataBase Administrator

SP System Programmer

Test Environment

- Definition
  1. AP defines the objects following the enterprise rules.
  2. AP shows the system structure and the relationship between the components of the new system to the DBA and SP.
  3. DBA and SP authorize the AP to begin the application coding.
- Function Test
  1. DBA creates objects in the function test database.
  2. AP tests the SQL statements and application flow.
- Simulation Test
  1. DBA creates objects in the simulation test database.
  2. DBA inspects the application source to see if the SQL statements are indexable and if possible sargable.
  3. AP and DBA use SQL-Tune or the explain tables to verify the access path.
  4. If access path not as expected, DBA defines new suitable indexes or requests AP to modify the SQL statements to get the best access path for data retrieval.
  5. Overall testing has to be performed by the end user to verify concurrency and application logic.

Production Environment

1. DA checks the object names for mismatches according to the enterprise rules.
2. DA or AP establishes the relationship between the new application and the existing ones.
3. AP estimates the space for each of the new tables to be defined.
4. DBA defines the new objects in the production database.
5. AP compiles the application.

6. DBA tests the new application behavior.
7. DBA checks the authorization structure looking for missing grants.
8. DBA asks the AP to run the application and uses VM:DB/Monitor to verify the application behavior in the production environment.
9. SP checks overall application performance.

Comment

VSE ONLY For CICS applications SP can restrict the number of concurrent executions, for example defining a special CICS class for the application transaction.

For more details, refer to 7.3.8, “Isolating CICS Applications in a VSE Environment” on page 113.

7.3.2 Application Coding

Having a clear definition of the data requirements of the business is the first step towards designing efficient databases that will perform well, support the business processes and be a valuable asset to the enterprise.

- Follow the enterprise rules as they make the maintenance easier.
- It is very important before starting coding an application program to know some basic things related to SQL usage. Please refer to 9.3.1.3, “Recommendations” on page 127.
- Use 31-bit addressing mode. Programs that reside above the 16MB line must be link-edited with the AMODE(31), RMODE(ANY) options on the MODE statement of the link-edit.
  COBOL programs should be compiled with option DATA(31) in the CBL statement. Assembler and High Level Assembler programs must not violate the 31-bit addressing rules.
- Read very carefully about “Implementing Your Design and Supporting Your Users” in chapters 2 to 4 of the appropriate DB2 *System Administration* manual.
- Reuse code if possible. Define copies, macros or members to be included into your programs.
- Code encapsulated functions providing general services such as retrieving, inserting or updating rows, which could be used by another application program. Example: code a program which receives a customer identification number as a parameter and looks for it in a table. If it exists, the program returns all the information related to it; if not, it calls the add customer function.

  This way, you have just one program retrieving information about one customer, which makes it easier to optimize and maintain.
- Document your work.
- Make the functions or subroutines you have created available to everyone, so that they can be used by another programmer.

7.3.3 Application Testing

This is one of the most important steps in the application's life. The better you test at this moment, the less problems you will have in the future.

1. Test for the obvious first: does the application run, or does it have
   - Cancellations
   - Loops
   - Missing referenced objects
2. Test the functionality: does it work as it is supposed to?
3. Become a user for a moment. Try to use your application "from the other side".
   - Try to misuse it.
   - Try it with wrong parameters.
   - Try it with no parameters.
   - Press all the keys you want (for on-line applications).
4. Try the integrity, cancelling the application program. Is everything left in a consistent state?
5. Look if the printed output is what you have planned.

Remember: **never** debug your applications in production systems.
VSE only: the use of the CEDF transaction must be avoided in production systems.

Important Consideration

If you are implementing a new application program group, then prepare a significant simulation environment with at least 15%-20% of your expected amount of transactions and data concurrency on your test system. Refer also to 7.1, "Test Databases" on page 105 for some remarks about a simulation environment. Remember also to change the statistics if needed. Ask the users for help, and simulate your workload for a significant period of time.

7.3.4 Using SQL-Tune to See the Application Behavior

SQL-Tune is a valuable tool to find performance problems related to access paths. For a description, refer to Chapter 5, "SQL-Tune" on page 53.

1. Analyze **all** the SQL statements within the application program even if you think some of them are trivial.
2. The most obvious reason for a `DBSPACE SCAN` as access path, is that no suitable index exists.
   If the index exists but is not used, then refer to9.3.1.3, "Recommendations" on page 127 .
3. If you still have problems with the selected access path or with performance, think about reformulating the query or splitting the table into two (or more) tables. One table could contain the frequently used columns and the other(s) could contain the infrequently used columns. You could then cluster the rows of frequently used columns on a fewer number of pages.

Notes:

- The disadvantage of splitting a table is the additional overhead for SQL statements that need all columns (which, by definition, is infrequent).
- Splitting a table also produces redundant data. That is, both (all) tables would have to contain the necessary column(s) to support the join.

4. If you need to reformulate your SQL statement, try it first with SQL-Tune on-line tuning until you get a suitable access path, before changing the application source.

7.3.5 Establishing Relationships Between Tables

- It is recommended to have a naming convention for column names.
- Columns in different tables, that keep the same data entity, should not only have the same name but also the same data type and additional attributes
- Implement referential integrity where possible

Referential integrity defines relations between tables in which the existence of values in one table depends on the existence of the same value in another table. By enforcing referential constraints (referential integrity rules) that are part of the table definitions, the database manager ensures the referential integrity of the data in the tables.

Note: Applications that currently enforce consistency and integrity of their data can be modified to let the database manager do the checking. Using the referential constraints and the integrity rules that apply to the tables containing the data, the system checks that the rules are adhered to, and thereby enforces integrity of the data. As this function can be performed by the database manager, some existing code can be removed from the application.

For more information, refer to Chapter 10 of the appropriate *Application Programming* manual.

7.3.6 Estimating Space for Objects

To estimate the space for objects, refer to Appendix B in the appropriate *System Administration*. This chapter describes how to determine the initial space requirement. Usage of the tables, dbspaces will show if the settings, such as pctfree, pctindex or other, have to be adjusted. This depends on the frequency of updates, usage of NULL attribute or defining columns with datatype VARCHAR. For inserts it is important to know if data is inserted at random or always at the end of the table.
Control Center can produce reports that give recommendations on adjusting settings for dbspaces. See Figure 16 on page 48 for a sample output of this report.

7.3.7 Defining Objects into Production

It's very important to know exactly what to do every time a new object must be defined into the production database.

1. In conjunction with the DA, the DBA must have set the standards for object definition.

2. Follow the general suggestions given in “Recommendations” on page 129.
3. Always use DBSU, rather then ISQL, for object definition and keep the jobs not only for documentation purpose but also to have the possibility to recreate these objects if needed.
4. Define the objects while no other user is accessing the database, and try to use small LUWs. DDL holds many locks on the database catalog so other applications can be placed in lock wait or even be rolled back by the system, due to a deadlock condition.
5. If new projects are implemented that use tables that can be isolated from other tables, define one or more new storage pools to keep these tables. If possible define a new DB2 user ID that will be the owner or creator of these new objects.
6. Grant the exact authorization needed to run. Don’t grant PUBLIC access on database objects ″just in case″.
7. Don’t use SQLDBA as the creator of user tables. If you need to change the SQLDBA password, then you must change every program using it.

7.3.8 Isolating CICS Applications in a VSE Environment

This is a very efficient way to isolate those applications that must be restricted for any reason.
Using RDO, change the TClass parameter from *NO* to the value you choose, say *01*. To do so, you must use the CEDA transaction as follows:

```
EXP G(CONTAB) TRAN(QCO1)
ENTER COMMANDS
 NAME     TYPE         GROUP                                  DATE   TIME
 QCO1     TRANSACTION  CONTAB   a                             97.342 16.42.17
```

Figure 70. RDO CEDA EXPAND to Alter a Transaction Definition

```
 OVERTYPE TO MODIFY
  CEDA  ALter
   TRansaction    : QCO1
   Group          : CONTAB
   PROGram      ==> QCO1
   TWasize      ==> 00000             0-32767
   PROFile      ==> DFHCICST
   PArtitionset ==>
   STatus       ==> Enabled           Enabled | Disabled
   PRIMedsize   ==> 00000             0-65520
  REMOTE ATTRIBUTES
   DYnamic      ==> No                No | Yes
   REMOTESystem ==>
   REMOTEName   ==>
   TRProf       ==>
   Localq       ==>                   No | Yes
  SCHEDULING
   PRIOrity     ==> 001               0-255
+  TClass       ==> 01                No | 1-10


PF 1 HELP       3 END            6 CRSR 7 SBH 8 SFH 9 MSG 10 SB 11 SF 1
```

Figure 71. RDO Screen Showing the TClass Parameter

If you use CICS tables, modify the DFHPCT table inserting a new line with the parameter *TCLASS=01*.

```
QC01     DFHPCT TYPE=ENTRY,                                          X
               TRANSID=QC01,                                         X
               PROGRAM=QC01,                                         X
               TWASIZE=00000,                                        X
               DTB=YES,                                              X
               TCLASS=01,                                            X
               SPURGE=YES,                                           X
               TPURGE=YES
```

Figure 72. CICS DFHPCT Table Showing the TClass Parameter

Change the value for the class in your DFHSIT definition, for example 03.

```
    BMS=FULL,               FULL BASIC MAPPING SUPPORT     *
    CMXT=(03,10,10,10,10,10,10,10,10), 10 TASKS/ TRANS CL*
    COBOL2=NO,              USAGE OF COBOL II APPL. PRGMS  *
```

Figure 73. CICS DFHSIT Showing the CMXT Parameter

If you cannot wait until the next CICS restart, you can issue

```
CEMT S TC(01) MAX(3)
```

to immediately activate these parameters. This command will only be valid until the next CICS restart.

Chapter 8. Application Tuning

This chapter describes some guidelines for performance tuning of existing applications, whether the source code still exists or not. It also describes some general techniques that can be helpful while tuning applications or the database.

8.1 Tuning in General

This part describes some ideas and techniques in general that can be used for tuning purposes. It lists some possibilities for monitoring, reporting and other ways to get performance information.

When tuning is performed on application programs, you should concentrate your efforts on:

- Those applications that are most used.
  You could even focus on a section instead of a complete application.
- Those applications that use a lot of system or database resources.
- Problems that can be corrected with the least effort.

Reports of VM:DB/Monitor can help you in providing information about the number of times an application or a section is being executed. Refer to Figure 119 on page 175 for an overview of package and resource usage and to Figure 120 on page 176 for an overview of section usage.

8.1.1 Table Growth

It is not always easy to estimate the real table size. Therefore, you could consider to frequently run a job that monitors and records the size of predefined tables. This would give you an idea of the real table growth and help you in defining the PCTFREE parameter on the next reorganization job, or to adjust the dbspace size before getting a dbspace full condition. To be sure that the captured data is the real data, it should be considered that this job also does execute an UPDATE STATISTICS before capturing the data. This job could be scheduled and controlled by Control Center. A sample job is shown in the next figure.

```
SELECT D.DBSPACENAME, T.TNAME,    T.ROWCOUNT,
       T.NPAGES,      T.PCTPAGES, D.NACTIVE, CURRENT DATE
  FROM SYSTEM.SYSCATALOG T, SYSTEM.SYSDBSPACES D
     WHERE T.DBSPACENO   = D.DBSPACENO   AND
           T.DBSPACENAME = D.DBSPACENAME AND
      D.DBSPACENAME IN ( :HOO1, :HOO2, :HOO3 ...)
```

Figure 74. Sample Program for Table Growth Analysis

This sample SQL statement would give the next output for dbspacename SAMPLE.

```
DBSPACENAME TNAME      ROWCOUNT NPAGES PCTPAGES NACTIVE CURRENT DATE
----------- ---------- -------- ------ -------- ------- ------------
SAMPLE      DEPARTMENT        9      1      100       6 1997-01-15
SAMPLE      EMPLOYEE         32      2      100       6 1997-01-15
SAMPLE      PROJECT          20      1      100       6 1997-01-15
SAMPLE      ACTIVITY         18      2      100       6 1997-01-15
SAMPLE      PROJ_ACT         77      2      100       6 1997-01-15
SAMPLE      EMP_ACT          74      2      100       6 1997-01-15
SAMPLE      CL_SCHED          3      1      100       6 1997-01-15
SAMPLE      IN_TRAY           1      1      100       6 1997-01-15
SAMPLE      DEPT             11      5      100       6 1997-01-15
```

Figure 75. Output Table Overview

After having captured and accumulated this data into a file, a program could be written to create some summary information to have an overview on the evolution in time.

```
DBSPACENAME TNAME      ROWCOUNT NPAGES PCTPAGES NACTIVE DATE
----------- ---------- -------- ------ -------- ------- ----------
SAMPLE      DEPT            134     23      100      28 1997-01-15
SAMPLE      DEPT            172     28      100      33 1997-01-16
SAMPLE      DEPT            210     34      100      39 1997-01-17
SAMPLE      DEPT            248     39      100      44 1997-01-18
SAMPLE      DEPT            286     45      100      49 1997-01-19
SAMPLE      DEPT            324     50      100      55 1997-01-20
SAMPLE      DEPT            362     56      100      61 1997-01-21
---------------------------------------------------------------------
SAMPLE      EMPLOYEE         32      2      100      28 1997-01-15
 output repeated for all tables
---------------------------------------------------------------------
Dbspace Summary :                                     61 1997-01-21
---------------------------------------------------------------------
```

Figure 76. History Table Growth

8.1.2 Dbspace Statistics and History

A similar job could be created to gather information about dbspaces, tables and indexes. In the next sample report you find a line for each dbspace available in the database. The same line also keeps data about tables and indexes residing in this dbspace.
The report does not only present figures from the database catalogs, but also gives data that was produced as output of the SHOW DBSPACE command.

A program could be created to produce a report that gives an overview of the evolution of dbspaces.

```
Service/RZ                                A D S H O W                                Date   25.01.1998   14:11:10  P
age:    1
                                          ==========                                 List : ADSHOW    Prog: ADSHOWX
   V01.31
Database : FVVPDB                 DBSPACE - Overview

DB                                AVG OVER          PO       <---------- D A T A ------------> <------- I N D E X -------------->
NO  DBSNAME    TNAME       ROWS   ROW FLOW I NPAGES  OL I% F%    DEF    USE  PCT WARN FREE EMPTY    DEF    USE  PCT WARN FREE EMPTY
--- ---------- -------- ------- ---- ---- - ------ --- -- -- ------ ------ ---- ---- ---- ----- ------ ------ ---- ---- ---- -----
267 COUNTERS   COUNTERS    6658  168    0 0   5632 -14 33 15   3766    395  10%         23     0   1858      0   0%          *     0
109 DBCMAINT   PARTS          6   36    0 1    128   9 33 15     78      1   1%         84     0     42      4   9%         97     0
202 DBINFO     CAT        81966  169    0 0  11648  10 15  0   9893   6330  63%          3     1   1747      1   0%         99     0
 75 FV_PBL_AKT AKTION     88803  328    0 1  12288   6  8  0  11301   8881  78% *       19     0    983    392  39%         10     0
 34 FV_PBL_ORT ORTKGS     41835  158    0 3   4096   5 10  0   3683   2092  56%         22     0    409    225  55%         10     0
 16 FV_PBL_000 ROUTINE        4   58    0 3   1024  -3 33 10    683      1   0%         94     0    337      1   0%         98     0
296 FV_PBL_001 ZADR      121447  420    0 5  37120  15 25 15  27836  24290  87% **      48     0   9280   3989  42%         10     5
 26 FV_PBL_002 BANK        7637   90    0 1    512   5 10  0    457    223  48%         24     0     51     19  37%         10     0
276 FV_PBL_003 KUNDE      89665  228    0 1  12288   5 15  0  10441   5602  53%          9   611   1843    223  12%         10   240
338 FV_PBL_004 ADRTYP    159136   42    0 2   5120   9 30  0   3580   1727  48%          4   306   1536    961  62%         10   487
 40 FV_PBL_005 TERMIN     30015  135    0 6   6400   5 25  0   4796   1113  23%         10   154   1600    514  32%         11    50
 42 FV_PBL_006 BANKVER    80997   89    0 1  12288   5 10  0  11056   1952  17%          8   187   1228    246  20%         10     1
294 FV_PBL_007 KDABL    1141021  111    0 3  55680   5 16  0  46768  34400  73% *       9     0   8908   7329  82% **      10    10
 47 FV_PBL_008 ZADRINT        0    0    0 1    512   5 25  0    380      0   0%          *     0    128      1   0%        100     0
146 FV_PBL_009 KB             1  105    0 1    256   5 33  0    164      1   0%         97     0     84      1   1%         99     0
147 FV_PBL_010 KI           109   61    0 1    256   5 33  0    164      3   1%         45     1     84      1   1%         70     0
 37 FV_PBL_011 KP           139   67    0 1   4480   5 33  0   2994      3   0%         24     0   1478      1   0%         55     0
151 FV_PBL_012 KD            12  110    0 2    512   5 33  0    336      1   0%         67    15    168      2   1%         96     4
148 FV_PBL_013 PS          2137   55    0 1    256   5 33  0    164     33  20%         12    30     84     12  14%         20    15
152 FV_PBL_014 P1          9922   65    0 3   8064   5 33  0   5395    198   3%         20     0   2661    135   5%         12     0
216 FV_PBL_015 P2         23738   37    0 1   2816   5 33  0   1879    229  12%          5    56    929     86   9%         11     1
 32 FV_PBL_016 P4         26640   84    0 1   2560   5 33  0   1708    714  41%         23     0    844     82   9%         11     1
153 FV_PBL_017 K1         13032  102    0 1   5504   5 33  0   3680    362   9%         10     0   1816    139   7%          3   121
199 FV_PBL_018 P5       1556585  112    0 3 241536  10 30  0 169072  47753  28%         10     0  72460  14884  20%          9  5349
```

Figure 77. Output Dbspace Overview

8.1.3 Index Statistics and History

The same kind of job has been created to gather information about indexes. This gives information about:

- Columns in indexes, to see if the index structure has been modified
- Pages needed for indexes. This can be helpful for adjusting the PCTINDEX parameter, when executing an ACQUIRE DBSPACE command.
- Clustering
  This can help you in monitoring when data becomes unclustered and gives an indication how to adjust the PCTFREE parameter at the next reorganization of the dbspace.
- Index usage
  Indexes that are never used occupy pages that could be used for other purposes. It should be considered to drop these indexes. Refer to 8.4, “Tuning Dynamic Statements” on page 123 where we talk about dynamic statements that might use these indexes, and this usage is not reflected in SYSTEM.SYSUSAGE.

```
Service/RZ                    VM:DB/Reporter                                        Date : 02.05.1995  07:35:5
3  Page: 1
                              INDEX History                                  List : DBIIDX01     Prog: DBIIDX
 V01.00
Database : FVVPDB

TNAME      INAME         ICREATOR I C     RATIO L   %  L     P Date       INDEX-Definition
---------- ------------  -------- - -   ------- - ---  - ----- ----------  ------------------------------------------------
ADRTBEW    ADRTBEWI01    FV       U F      9997 K  10  0     0 19.12.1994 +ADRTNR, +ADRTYP, +ZABLNR,
                         FV       U F      9995 K  10  0     0 02.01.1995 +ADRTNR, +ADRTYP, +ZABLNR,
                         FV       U F      9995 K  10  3  1506 09.01.1995 +ADRTNR, +ADRTYP, +ZABLNR,
                         FV       U W      9981 K  10  3  1531 06.03.1995 +ADRTNR, +ADRTYP, +ZABLNR,
                         FV       U W      9980 K  10  3  1533 10.03.1995 +ADRTNR, +ADRTYP, +ZABLNR,
                         FV       U W      9969 K  10  3  1549 18.04.1995 +ADRTNR, +ADRTYP, +ZABLNR,
                         FV       U W      9969 K  10  3  1549 18.04.1995 +ADRTNR, +ADRTYP, +ZABLNR,
                         FV       U W      9967 K  10  3  1551 24.04.1995 +ADRTNR, +ADRTYP, +ZABLNR,
                         FV       U W      9965 K  10  3  1554 02.05.1995 +ADRTNR, +ADRTYP, +ZABLNR,
ADRTYP     ADRTYPI01     FV       U W      9572 K  10  0     0 19.12.1994 +ADRTYP, +ADRTNR, +ADRKZ,
                         FV       U W      9530 K  10  0     0 02.01.1995 +ADRTYP, +ADRTNR, +ADRKZ,
                         FV       U F     10000 K  10  3   461 09.01.1995 +ADRTYP, +ADRTNR, +ADRKZ,
                         FV       U F     10000 K  10  3   462 13.01.1995 +ADRTYP, +ADRTNR, +ADRKZ,
                         FV       U F     10000 K  10  3   462 16.01.1995 +ADRTYP, +ADRTNR, +ADRKZ,
                         FV       U F     10000 K  10  3   462 16.01.1995 +ADRTYP, +ADRTNR, +ADRKZ,
                         FV       U W      9965 K  10  3   464 23.01.1995 +ADRTYP, +ADRTNR, +ADRKZ,
                         FV       U W      9949 K  10  3   465 27.01.1995 +ADRTYP, +ADRTNR, +ADRKZ,
                         FV       U W      9944 K  10  3   465 30.01.1995 +ADRTYP, +ADRTNR, +ADRKZ,
                         FV       U W      9927 K  10  3   466 06.02.1995 +ADRTYP, +ADRTNR, +ADRKZ,
                         FV       U W      9908 K  10  3   467 13.02.1995 +ADRTYP, +ADRTNR, +ADRKZ,
                         FV       U W      9902 K  10  3   468 15.02.1995 +ADRTYP, +ADRTNR, +ADRKZ,
```

Figure 78. Index History Information

8.2 Tuning Your Current Applications

It is not only important to define new applications so that they perform well, it is also important to have a look at existing applications that have not been modified for a long time.

To be more productive when you are planning to do a performance tuning job on your existing database applications you have to concentrate your efforts on applications using the database intensively. In other words, start with applications that use the most of both, system and database resources.

To discover these applications you will need a database monitor tool. VM:DB/Monitor shows you a list of resource consumption of all the packages used during a time interval, ordered by consumption. Such a report is shown in Figure 119 on page 175.

While tuning an application concentrate and prioritize your efforts on those SQL statements that use most of the database resources.

There are some tools that can help you in this job:

- VM:DB/Monitor produces a report that shows for each SQL statement executed the consumption of the database resources. Figure 120 on page 176 shows a sample of such a report.

  There are different reports produced by VM:DB/Monitor, that show you the tables accessed without the usage of an index, thus through a dbspace scan. A sample output of all reports can be found in Appendix B, “Screens and Reports of VM:DB/Monitor” on page 147. In report TABNDXU (Figure 125 on page 180) and TABNDXS (Figure 124 on page 180), the lines that have a zero for *Index-id* indicate, that for this table no index has been used for the access path. Further detailed analysis of the reports SESSIOND (Figure 122

on page 178) or PACKAGED (Figure 118 on page 174) point you to packages or applications that perform a dbspace scan.

In some cases, a dbspace scan can not be avoided.

- SQL-Tune displays SQL statements consuming more than a given threshold value of elapsed time. This helps you in locating a performance problem, and gives you the possibility to inspect the access path and to change a statement on-line to see the behavior of the statement after this modification.

  SQL-Tune will also warn you if some conditions occur that need special attention, such as recommended use of dynamic SQL to improve the access path. SQL-Tune also displays a list of critical dbspaces (see “List of Critical Dbspaces (PF8)” on page 78) or a list of critical tables (see “List of Critical Tables (PF9)” on page 78).

8.2.1.1 Example 1

The following figure shows an example of a statement that uses a dbspace scan to access the data.

```
SELECT W_ID, W_NAME, W_STREET_1, W_STREET_2, W_CITY, W_STATE, W_ZIP INTO
 :H001 , :H002 , :H003 , :H004 , :H005 , :H006 , :H007
FROM SQLDBA.WAREHOUSE
WHERE W_ID = :H008
```

Figure 79. Example 1: Statement from SQL-Tune

```
 Table : WAREHOUSE             Creator : SQLDBA
                               Method  : DBSPACE SCAN
                               Index   : NO POSSIBLE USE
```

Figure 80. Example 1: Access Path from SQL-Tune

There is an index available on the column W_ID that is being used in the predicate.

```
 ICREATOR INAME              CLUST RATIO LEAF COLNAMES
 -------- ------------------ ----- ----- ---- ---------------
 SQLDBA   WAREHOUSE_IX2        F   10000    1 +W_ID,+W_TAX,
```

Figure 81. Example 1: Index

Selecting information from SYSTEM.SYSCATALOG will show that this table only contains 10 rows and occupies 1 page.

Recommendation: Put this small table together with similar tables in a dbspace of 128 pages, as this is the smallest entity that can be defined. Another possibility would be to put this table alone in a dbspace and so there will be only one I/O to get this data page.

8.2.1.2 Example 2

Another case where some tuning or design change can be considered is the table FV.BK with 79,730 rows, from which 79,681 (almost 100%) have the same value *3120* in the column BKKREDID.
Indexes were built with the following specifications:

```
Index              Creator   seq  clust  unique  Lock   first   full
                                                  mode    key     key
BKI01              FV        YES   YES    YES    PAGE      24   79730
BKKREDID(ASC) UKID(ASC) BKAUSBL(ASC) BKLFDNR(ASC)
BKI04              FV        NO    NO     NO     PAGE      24   79654
BKKREDID(ASC) UKID(ASC) BKBUBTR(ASC) BKAUSBL(ASC) BKKZSH(ASC)
BKI05              FV        NO    NO     NO     PAGE    1130    1631
BKBLNR(ASC) BKBLART(ASC) BKBUJAHR(ASC)
BKI80              FV        NO    YES    NO     PAGE      24   41308
BKKREDID(ASC) BKAUSBL(ASC) BKAUSKZ(ASC)
```

Figure 82. Example 2: Index Definition

Some of the applications, accessing this table use the predicate

```
WHERE BKKREDID = 3120 ...
```

As this selection applies to almost 100% of table, in most cases a dbspace scan is performed.

Recommendation:

1. The indexes named BKI01, BKI04 and BKI80 start with the same columns. It would be better to put BKKREDID and UKID behind the column BKAUSBL in the column list of the indexes BKI01, BKI04 and BKI80. This fulfills also recommendation 2.
2. Put on top of the column list of any index a column with more different values, to increase the value of the *First Key Count*.
3. Put additional selection criteria in this query. Best would be to follow the column sequence of any index.

8.2.1.3 Example 3

This application is a combination of different statements that fetches data from an input table and depending on this data does insert and update on other related tables.

This application contains a statement for fetching the data which uses a dbspace scan as access path. There is a unique index available on the first ten columns of the table. All columns of the index are also available in the predicate.
SQL-Tune shows in the next figure that the access path is a dbspace scan and an additional sort. Obviously this will need much time.

```
SELECT CDESTABE,CDCLIENT,NRNFFAT,CDESPINS,CDORDINS, CDCORINS,CDTALINS,
       CDQUAINS,SUM(QTPRODUT)
   FROM PBQMRIP
     WHERE CDESTABE >= 0 AND
           CDCLIENT >= 0 AND
           NRNOFFAC >= 0 AND
           NRNFFAT  >= 0 AND
           CDESPINS >= 0 AND
           CDORDINS >= 0 AND
           CDCORINS >= 0 AND
           CDTALINS >= 0 AND
           CDQUAINS >= 0 AND
           CDORDPRO >= 0
       GROUP BY CDESTABE,CDCLIENT,NRNFFAT,
                CDESPINS,CDORDINS, CDCORINS,CDTALINS,CDQUAINS
```

Figure 83. Example 3: Statement with Dbspace Scan

```
  Table : PBQMRIP              Creator : ELRES5
                               Method  : DBSPACE SCAN + SORT
                               Index   : NO POSSIBLE USE
```

Figure 84. Example 3: Access Path from SQL-Tune

```
 Index Name          Creator                                    Clusterratio
 PBQMRIP_IO1         ELRES5        UNIQUE                             100,00
 CDESTABE(ASC) CDCLIENT(ASC) NRNOFFAC(ASC) NRNFFAT(ASC) CDESPINS(ASC)
 CDORDINS(ASC) CDCORINS(ASC) CDTALINS(ASC) CDQUAINS(ASC) CDORDPRO(ASC)
```

Figure 85. Example 3: Index Structure

Recommendation: The easiest way to resolve this problem would be to add the column QTPRODUT at the end of the index definition. This changes the access path from dbspace scan to direct index access.

Further analysis of the table and data, showed that 11,848 rows on a total of 43,026 were read. The figures related to the run of this statement in both cases, dbspace scan and direct index access, indicated on our system that there is no measurable difference in the elapsed time. The reason for this is the fact that the elapsed time was too short to give relevant results. But other differences could be noticed, summarized in the next figure. The sources for these are the reports created by the *DBMAP* utility of VM:DB/Monitor. Used reports were *SESSIOND*, *PACKAGED* and *PACKAGES*.

```
              Wait I/O    Page Reads   I/O     Dispatch Count
  Dbspace scan    14%          944       971         6172

  Index Use        3%          421       435         3315
```

Figure 86. Program Statistics

These differences could have been noticed easily on a system, where more than one agent was active, because this application is often dispatched.

Hint

All the above recommendations requested a modification of the index design or even table design. All tuning that involves structure modifications should not be analyzed as standalone cases, but in relation to other application programs that reference these tables. As mentioned before, modifying an index for one application could have the reverse effect on the performance of another application.

8.3 How to Tune without Source Code

If the application source code is no longer available, there is no possibility to modify the application program. The tables referenced by these applications can not be changed, neither the column specifications nor the table layout.

The only possibility you have to tune these applications is by changing the indexes:

1. Display the access path used for each SQL statement using SQL-Tune
2. Create or rearrange indexes, so that each SQL statement can benefit from data access through an index
3. Verify again the used access path

If new indexes are created, there will only be the impact of maintaining these indexes and the space for them.

The next sample statement can be used to verify which package uses a specific index (for examplw, ORDERITEM_IX1).

```
SELECT * FROM SYSTEM.SYSUSAGE WHERE BTYPE='I' AND BNAME='ORDERITEM_IX1'
 AND DTYPE = 'X'
```

Figure 87. Command to Display Index Usage

The following sample output indicates that three packages are using this index.

```
BNAME               BCREATOR  BTYPE  DNAME               DCREATOR  DTYPE
------------------  --------  -----  ------------------  --------  -----
ORDERITEM_IX1       SQLDBA    I      GPPLPNO             SQLDBA    X
ORDERITEM_IX1       SQLDBA    I      GPPLPDB             SQLDBA    X
ORDERITEM_IX1       SQLDBA    I      GPPLPOS             SQLDBA    X
* End of Result *** 3 Rows Displayed ***Cost Estimate is 1****************
```

Figure 88. Sample Index Usage Display

If there are no rows that apply to this query, this does not mean that the index is not being used. See 8.4, “Tuning Dynamic Statements” below where we talk about these cases.

8.3.1 Views on Read-Only Tables

If there is a need to split a table into more than one table, this normally can not be done if the source code of the application is not available.

A bypass for this problem could be to create a view:

1. Unload the original table with DATAUNLOAD command of DBSU
2. Create new tables with columns from the table to be split
3. Reload the data to the new tables with DATARELOAD of DBSU
4. Create a view that joins the new created tables, with the same name as the original table

Remark: This can only be done for read-only applications. There are restrictions on inserting or updating data on a view, if this view comprises more than one table.

8.4 Tuning Dynamic Statements

Although we recommend to avoid the use of dynamic SQL statements and of ISQL on a production database, there might be systems, for example a special query database, that allow the usage of ISQL, DBSU, REXX SQL, PIPE SQL or QMF. In most cases this type of application runs dynamic SQL statements. It should be considered to tune also these applications and avoid a dbspace scan for each query.

Although the DBA has no knowledge of these SQL statements executed by end users, SQL-Tune and VM:DB/Monitor can help the DBA to identify these statements. Another solution would be that the end users inform the DBA about the most often used statements.

SQL-Tune keeps track of these statements and they can be visualized from the statistics panel (PF4 from the main menu), refer to Figure 35 on page 67. There is no indication on this panel, that a statement is dynamic. The only information is the prepname (for example, ARIISQL for ISQL) that gives you an indication that this statement was invoked from within an ISQL session.

VM:DB/Monitor also creates a report about dynamic statements that have run on your system. In Figure 116 on page 171 you can identify statements that were executed from within ISQL (program ARIISQL), QMF (DSQxxxx), DBSU (ARIDSQL) or an application program (B752A). The output of this report can be

used as input for SQL-Tune to display the used access path, and thus to optimize this statement.

Indexes can be created to avoid a dbspace scan to get the data. To easily identify them there could be a naming convention for these indexes: for example, NEWORDER_ID1 where ID stands for index dynamic.

To resolve the missing reference in SYSTEM.SYSUSAGE of the dynamic statements, a dummy program could be created that keeps only SQL statements executed as dynamic statements. You could implement also a naming convention to identify such programs. See Figure 89 for a sample program. After preprocessing this program, there would be entries in the SYSTEM.SYSUSAGE for this program. This would give an indication of the need for these indexes and they would not be deleted just because there were no entries in SYSTEM.SYSUSAGE.

```
       IDENTIFICATION DIVISION.
       PROGRAM-ID.    CTPB112.
       ENVIRONMENT DIVISION.
       CONFIGURATION SECTION.
       SPECIAL-NAMES. DECIMAL-POINT IS COMMA
                      C01 IS CANAL-1.

       INPUT-OUTPUT SECTION.
       DATA DIVISION.
       WORKING-STORAGE SECTION.
       77  W77CDSQL                PIC 999-.
           EXEC SQL BEGIN DECLARE SECTION END-EXEC.
       01  VARS94                        PIC S9(4) COMP.
       01  VARS99                        PIC S9(9) COMP.
       01  VARS97C                       PIC S9(07)V9(03)  COMP-3.
           EXEC SQL END DECLARE SECTION END-EXEC.
      *
           EXEC SQL INCLUDE SQLCA       END-EXEC.
      *
       PROCEDURE DIVISION.
       0000-MAIN-CONTROL.
           EXEC SQL SELECT QTFATURA,DTAAAAMM
                       INTO :VARS97C,
                            :VARS99
                FROM ELRES5.PBQREFP
                   WHERE CDESTABE = :VARS94
                     AND CDCLIENT = :VARS99
                     AND CDQUAINS = :VARS94             END-EXEC.
           STOP RUN.
       0000-EXIT.
           EXIT.
```

Figure 89. Sample Program for Dynamic Statements

Chapter 9. Checklists and Recommendations

This chapter offers lists of items to be checked in case of a performance problem or during regular performance monitoring and tuning. Additionally, it contains lists of recommendations and considerations for the various subjects in order to help prevent performance problems from arising.

9.1 Introduction

When you or your end users think the performance behavior of an application or the whole system has become worse, you have to investigate what has happened. Generally, this is not easy due to many factors that can affect the system behavior, for example, memory constraints might appear when new applications have been put in production, or new users were added to the system.

Therefore, this chapter gives you checklists of what to review in order to determine what is the real cause of the performance degradation. Having found this you can use the product manuals and the redbook "*SQL/DS Version 3 Release 4 Performance Guide*" to find the optimal solution. In that redbook, the paragraph "Performance Processes" in Chapter 2, "Methodology of Performance Monitoring" describes the process of tuning.

9.1.1 Available Tools

Depending on the problem area and operating system, there are different techniques and tools to gather the required information. For example, on VM you could use facilities such as CP Monitor, VM/Performance Reporting Facility, the Real Time Monitor, the `CP INDICATE USER` and `QUERY TIME` commands.

On VSE, the Interactive Interface shows useful information such as CPU usage, system paging, active users, and system activity. As well, there are some facilities from CICS/VSE such as CICS Monitoring Facility, CICSPARS/VSE, CIRD transaction and the CICS statistics.

These are some of the facilities available, and this list is not complete. Besides these, this redbook presents some tools in Part 2, "Tools" on page 31. For more information about those tools that you don't have on your operating system, contact your IBM representative.

9.2 List of Questions to Users

The following questions might help if put to the end users complaining about performance:

1. What do you mean by "slow response"?
   Is it 10% slower than you expect it to be, or ten times slower?
2. When did you notice the problem? Is it recent or has it always been there?
   Does it correlate to a moment something has changed?
3. How many users are complaining?
   Is is just one or two individuals, or a whole group?

4. If a whole group of users is experiencing difficulties, are they connected to the same terminal controller?
5. Are the problems related to a specific transaction or application program?
6. Do the problems appear during regular periods, such as lunch hour, or are they continuous?

9.3 Performance Checklist for VSE and VM

This part covers the following areas:

1. Applications
2. Database
3. System

9.3.1 Applications

9.3.1.1 Most Frequent Errors

Please check Chapter 5 of the "*DB2 Server for VSE & VM Performance Tuning Handbook*" for more details.

1. **Do not** misunderstand the term *indexable*. Indexable means *"*index can be used if a suitable index exists and some rules are respected*"*, for example, see the next two list items.
2. **Do not** do wrong comparisons. The data types and CCSIDs of any columns and literals should match whenever possible. Numeric values should use the same representation, including the same precision and scale for DECIMAL values. Character and graphic values should have the same length. Columns and literals should use the same CCSID.
3. **Do not** apply a NULL predicate to columns with the NOT NULL attribute. Doing this converts the predicate from sargable to residual.
4. **Do not** SELECT columns that are not used in the application. Select only the columns you really need to read or for using the index. Refer to item 5 on page 127 in the recommendations list. There is overhead in processing each column.
5. **Do not** UPDATE columns with no special care. UPDATE cannot use an index containing this column. Therefore, never update a column being part of an index unless you have another suitable index without this column. If not, an alternative would be to delete the row and insert a new row with the updated value.
6. **Do not** preprocess every program with the SQL preprocessor. If your application program does not execute any SQL statement, you will probably get an abend (for example, ASRA when running in CICS).
7. **Do not** forget to `INCLUDE ARIPRDID` in your link-edit job. If you are using a main program without SQL statements but you are calling subroutines with SQL statements, you must specify `INCLUDE ARIPRDID` anyhow. For batch COBOL programs, `ARIPADR4` must also be included in the link-edit job.

   Compilers that use dynamic linking (such as *COBOL for VSE* and *COBOL II*) do not need these two INCLUDEs at link-edit time.

9.3.1.2 Checklist

1. Check that on-line applications close the LUW (commit) as soon as the application logic allows it.
2. Check that the application is accessing the tables by using the planned indexes. If not, correct the statement, or analyze the impact of creating a new index either permanently or temporarily. If you create a new index, remember to REBIND every package being affected.
3. If the application reads or inserts multiple rows, use the BLOCKING option during preprocessing. See “Using the Blocking Option to Process Rows in Groups” in Chapter 4 of the respective manual: *DB2 Application Programming*
4. Check for updates on the primary key. If there are no referential integrity constraints involved, you can also delete the row and insert a new one in order to make use of the index during the operation. If another index can be used this is not necessary.
5. In case of many LOCK WAITS, check the isolation level with which the application was preprocessed or is dynamically using (if preprocessed with ISOL USER). Consider using uncommitted read whenever possible, to avoid unnecessary locks.

9.3.1.3 Recommendations

The following are recommendations, to remember when you are coding applications or help you to identify performance problems:

1. Try to use only indexable or sargable predicates.
2. Sometimes you can make the optimizer find a better access path just by coding as many indexable or sargable predicates as you can. Do not omit them, even if you think they are obvious or transitive.
   For example, when using a join between tables a, b and c

```
    a.cola = b.cola
and b.cola = c.cola
and c.cola = a.cola      <--  this statement could be added
or
    a.cola = b.cola
and a.cola = xxx
and b.cola = xxx         <--  this statement could be added
```

3. Be always explicit.
   Code

```
SELECT col1,col2 FROM table
INSERT INTO table (col1, col2) VALUES (:h-col1, :h-col2)
```

   rather than

```
SELECT * FROM table
INSERT INTO table VALUES (:h-col1, :h-col2)
```

   as this may avoid problems if the layout or column sequence of the referenced table has changed.

4. Always check the access path used. You can use SQL-Tune, the EXPLAIN(YES) preprocessor parameter or the EXPLAIN command to see the access path being selected by the optimizer.
5. Avoid sorting when possible.

Only specify DISTINCT, UNION or ORDER BY if it is really necessary. Try to provide a suitable index for key-matching predicates and ORDER BY resolution. The optimizer can detect that the order asked for by a query can be obtained using the same index, so it does not sort the data.

Note: Sometimes you only have to include an additional column in the retrieval list in order to assist the optimizer when selecting the access path.

6. Avoid joining more than three tables.
7. If using both GROUP BY and ORDER BY, specify the columns in the same order.
8. Avoid the use of dynamic SQL. Only use it if the optimizer would select a better access path after host variable resolution. If you can't, avoid running it at peak hours.
9. Keep frequently updated columns close together in the same row to save log space. Only the portion of the row from the first column updated to the last column updated is recorded into the log.
10. Design your applications so that they can handle the SQLCODE -911 (LUW rolled back due to deadlock or lock timeout). Just re-issuing the SQL command could work. Don't loop too often: just retry three or four times, then cancel the program.
11. Sometimes you have to implement "second normal form" (2NF) instead of "third normal form" (3NF) if I/O is a constraint on your system. This could avoid some additional join statements in your applications. See 9.6.3, "Normalization" on page 142.
12. SQLCA should always be included in the application program, because SQLCODE alone not always gives the complete information about an error condition (for example, truncation error). For a specific need, you could only define SQLCODE as a host variable (integer), and add the preprocessor parameter NOSQLCA. This makes your program smaller and reduces its execution time. However, debugging could become more difficult.
13. Use routines, not stored queries. Routines are easier to copy to another user and easier to maintain (for example, formatting commands).
14. Avoid the use of NULLS where possible.
15. Add the IS NULL or IS NOT NULL predicate for column functions to nullable columns in order to be key matching or sargable, else the predicate will be residual.
16. Use a single UPDATE statement to change several rows of a table rather than using a cursor and a program loop to achieve the same result.
17. Create indexes that include columns frequently queried to benefit from index-only access.
18. Use the LOCK TABLE command to avoid lower level locking overhead if the application accesses (almost) all data anyway.

Resource Contention

Locking provides concurrency in data access while maintaining the logical and physical integrity of the DB2 database. Wait situations may occur: this is a normal condition and is not always a problem.
Locking becomes a problem when the wait times due to simultaneous accesses on the data are long enough to cause significant degradation in response or elapsed times.

19. Try to keep the LUWs as small as possible. Connect just before your first SQL verb, rather than at the beginning of program initialization.
20. Always finish the LUWs explicitly by COMMIT or ROLLBACK, whichever applies.
21. Use the isolation level UR whenever possible, considering the implications in 2.2.2, “Usage” on page 7.
22. For on-line applications, use pseudo-conversational technique rather than conversational.
23. In long-running batch jobs, force the application program to issue COMMIT WORK requests at pre-defined intervals (for example, after *n* rows processed). Postpone changing data as much as possible. Do all the UPDATEs, INSERTs and DELETEs last, followed immediately by a COMMIT WORK.
24. Use AUTOCOMMIT ON in ISQL and DBSU.

9.3.2 Database

9.3.2.1 Administration

Checklist

1. Check if users that are complaining are in agent wait, lock wait or communication wait. Use the commands

   ```
   SHOW ACTIVE
   SHOW LOCK DBSPACE ALL
   SHOW LOCK USER
   SHOW LOCK GRAPH
   SHOW LOCK WANTLOCK
   ```

   to have information about the database status and about resources in use for the different agents.

Recommendations

This is only a list of the most important items. Please see “*DB2 Server for VSE & VM Performance Tuning Handbook*” for more explanation and for additional topics.

1. Define your internal dbspaces alone in a **recoverable** (see remark below) storage pool. If storage is a constraint, define multiple dbextents for the storage pool containing the internal dbspaces and use a virtual disk for the first dbextent. For information regarding virtual disk support for VSE/ESA and VM/ESA for internal dbspaces, see Chapter 3 of *DB2 Server for VSE & VM Performance Tuning Handbook*.
2. An alternative for VM would be to set the SEPINTDB startup parameter to Y, so the internal dbspaces will no longer be physically mapped.
3. Overcommit the internal dbspaces′ logical definition. For example, if the storage pool (for example, pool 2), keeping the internal dbspaces, has 10,000 physical pages, define the number of dbspaces (for example, 80) with 10,000 pages each:

   ```
   INTERNAL 80 10000 2
   ```

The actual number of physical pages used by internal dbspaces depends on how much data is actually placed in them. A small SORT or JOIN will only use a small amount of DASD space for internal dbspaces.

Remark

Placing your internal dbspaces in a non-recoverable storage pool will gain you no performance advantage, but will cost you some performance overhead because of extra checkpoints. There will be an extra checkpoint when a COMMIT occurs, either implicit or explicit. Generally this will be once per LUW that uses internal dbspaces.

There is never logging for the internal dbspaces, nor will there be a checkpoint taken when the internal dbspace is dropped after it is used.

4. Place only one table in a dbspace whenever possible. Exceptions to this could be if all of the following conditions are fulfilled:
   - Tables are small
   - Tables are read-only
   - Tables are always accessed by use of an index
5. Have a checkpoint occurrence in a 10 to 15 minutes interval. To do so, monitor the number of checkpoints with COUNTER or DSPSTATS **2**1 and SET CHKINTVL accordingly. Remember that the time between checkpoints you have, could be the time it takes to restart the database manager in case of a system failure.
6. Define your log file big enough to hold all the modifications for at least your whole period between scheduled log or database archives to avoid having them occur automatically in critical moments.
7. Don't allow ISQL usage in production or restrict its use to the minimum. ISQL could hold locks for a long time, since it is a conversational application and, in addition, it uses many CICS resources.
8. Change the default ISQL's ISOLATION LEVEL from RR to CS or UR, whichever applies best.
9. Set the default for the preprocessor parameter ISOL to CS or UR, whichever applies best.
10. Whenever possible, UPDATE STATISTICS for the dbspaces that hold the most modified data and REBIND the packages that reference these.
11. Balance the utilization of your DASD volumes. Never put any of the *directory*, the *catalog*, the *log*, the *dual log*, the *internal dbspaces* and the *data* on the same volume.
12. When loading data, commit data on a regular basis. Don't let the LUW become too big. Use the COMMITCOUNT parameter on RELOAD or DATALOAD.
13. Use DROP DBSPACE instead of DROP TABLE, whenever possible. It's much faster.
14. Check your catalog indexes for fragmentation. To determine the number of index pages occupied in the catalog dbspace, enter

    ```
    SHOW DBSPACE 1
    ```

    If there is a high percentage of occupied pages, consider running the catalog index reorganization utility, SQLCIREO. Read the appropriate *Database Administration* manual for guidance.

Note: To avoid switching log mode, specify the same LOGMODE that you normally use.

15. Determine if the LUWs are either long-running or short-running to SET DISPBIAS accordingly.

16. Check your recovery procedures. For more details, see the redbook "*DB2 Recovery on VSE and VM Using the Data Restore Feature*." It describes how to improve the availability of your database.

17. Use dual log to protect your database against DASD failure unless you are confident about your hardware: RAID 5 or 3990 dual copy. With dual logging, updates are recorded in two log data sets. There is a small overhead for the additional I/O but it protects you from log failures, since it is unlikely that an unrecoverable DASD failure will occur on both log data sets at the same time if you obey item 11 on page 130 above. Read the appropriate *System Administration* manual for guidance.

9.3.2.2 Indexes

Checklist

1. Check that the first (clustering) index is clustered by monitoring the column CLUSTER of SYSTEM.SYSINDEXES. Just use the following command within a DBSU job

   ```
   SELECT * FROM SYSTEM.SYSINDEXES
   ```

   CLUSTER=F means the index is still clustered, where CLUSTER=W means, this is no longer the case.
   Avoid ISQL to do so. Just use the previous command within a DBSU job.

2. Check for invalid indexes. Use the SHOW INVALID operator command to display all the invalid indexes as well as the reason why each index is invalid.

3. Check for index fragmentation. When you have a significant amount of modification activity within an index, check if you have an excessive amount of free space in the index pages. If you have, this usually is spread unevenly and implies that index keys are also distributed unevenly.

```
SHOW DBSPACE 44

  TYPE      NUMBER      NUMBER OF       % FREE    NUMBER OF
 OF PAGES  OF PAGES   OCCUPIED PAGES    SPACE    EMPTY PAGES
  HEADER        8         1 ( 12 %)      98 %           0
  DATA       2737       334 ( 12 %)       1 %         111
  INDEX      1351        51 (  3 %)      34 %           0
ARIO065I Operator command processing is complete.
```

Figure 90. Sample Output of the SHOW DBSPACE Command

   Reorganize your indexes if they are fragmented, as in this case.

4. Force a REBIND of packages after additional indexes were created. This can be done, by either using the REBIND command of DBSU or by updating the column VALID in SYSTEM.SYSACCESS for this package to N. This last possibility implies a rebind of the package, the first time it will be executed.

Recommendations

One of the most important things when working with a relational database as DB2 Server for VSE & VM is the fact that user applications can access data without being dependent on how the data is stored or on the types of access paths available to locate the data (in VSAM, for example, you must change your application to include a new index you have defined). The optimizer tries to determine the most efficient access path to the data.

Note: If data is accessed through an index, the optimizer makes an estimation of the number of pages that have to be read. The filter factor, that depends on the predicate, and the clustering of the data will influence this estimation.

You can **help the optimizer** taking into account the following guidelines:

1. Create the indexes after the data has been loaded into the table.
2. Define at least one index for each table.
3. Create unique indexes, whenever possible. Create indexes on multiple columns to have a unique combination for this index.
4. Avoid defining indexes on LONG fields, or on columns with frequently changing values.
5. If an index is created on multiple columns, define the column with the most different values on top of the list of columns. The reason is that detailed statistics are gathered only for the first column of each index, as there are FIRSTKEYCOUNT and other. The optimizer makes use of this, but for the effect of additional columns there are only rough estimates called *Filter Factors* (FF). Refer to “Predicate Processing” in Chapter 5 of the “*DB2 Server for VSE & VM Performance Tuning Handbook*.”

   Note: Only the first eight bytes of a column are stored into HIGH2KEY and LOW2KEY of SYSTEM.SYSCOLUMNS, so it is important that columns be distinct for the first eight bytes. (Seven bytes if the column is nullable.)
6. Create the index in the same order (ASC or DESC) as that used most by the applications to retrieve the data.
7. If you have more than one index, cluster the table relative to that index (use as first index) which you are using most often to do index scans or to retrieve many rows.
8. Specify a suitable PCTFREE value when you create an index.
9. Specify PCTFREE=0 on the CREATE INDEX command when creating indexes on tables which will have no random inserts, as the index will only have new keys at the bottom of the index tree.
10. Avoid indexes with the same high order columns. They are unnecessary, occupy DASD space and consume resources in case of updates.
11. Define indexes on foreign keys to speed up the reference operations.
12. Do not create all indexes you think might be of use in the future. Start with the primary key and only create additional indexes to establish unique constraints (enforce data integrity without enforcing integrity between tables). Performance indexes should only be created when a need for them emerges.

13. Consider creating additional indexes temporarily for heavily changing data which is also heavily queried for limited periods (for example, lots of end-of-month reports), and dropping them again afterwards.
14. Vice versa, for heavy, periodic updating, consider dropping indexes, do mass updates, and re-create indexes.
15. Monitor the CLUSTERRATIO value of the most important indexes of your installation and reorganize the table if the value is less than 7500. For this monitoring and reorganizing, Control Center is recommended.

9.3.2.3 Tuning

1. Check that dbspace statistics are updated.

   Make sure that UPDATE STATISTICS (or even better, UPDATE ALL STATISTICS) is done after the indexes have been created. Also consider doing a REBIND of the packages affected by the update of the statistics.

   When loading tables by DBSU during peak hours, SET UPDATE STATISTICS OFF and run UPDATE STATISTICS during off-peak hours.
   Control Center produces a report of dbspaces on which no update statistics has been done for more than *nn* days. This number of days can be specified when requesting the report.

2. Check that there is only one table per dbspace wherever possible:

   ```
   SELECT * FROM SYSTEM.SYSDBSPACES
          WHERE NTABS > 1 AND
          DBSPACENAME NOT IN ('SYS0001','HELPTEXT','ISQL','SAMPLE')
   ```

3. Use SHOW POOL to check the allocation of dbextents to pools:

   a. Make sure that none of the storage pools is filled up to nearly the value of the initialization parameter SOSLEVEL. Such pools can cause additional checkpoints to free shadow pages, or even make applications fail.
   b. All dbextents in the same pool should have the same size but be on different physical I/O devices (logical I/O devices if RAMAC), so that the operating system can read several groups of pages for different agents (applications) at the same time. ESA/390 architecture does not allow more than one I/O at any point in time against a single (although logical) device. DB2 (without VMDSS) allows only one I/O per agent at one point in time.

4. Designing your dbspaces:

   - Start with PAGE locking to decrease CPU consumption.
   - If you have page lock conflicts, use ROW locking.
   - Internal dbspaces must have enough pages to hold the largest sort in one dbspace.
   - Specify adequate free space based on update activity, lock contention and planned re-organization frequency.

     Dbspace PCTFREE is the **minimum** free space reserved on each data page.

     The directory holds an approximation: the free class.

Rule:
A row is inserted on an existing page if ROWLENGTH + 40*PCTFREE <= MINFREE, otherwise a new page is acquired.
Example: if you have an empty data page, the dbspace has been defined with PCTFREE=50 and you want to insert rows that are 61 bytes long, then you will be able to insert just **one** row.
For further information, see Chapter 2 of the respective *DB2 Diagnosis Guide and Reference Manual*.

- Change the free space usage with the ALTER DBSPACE command and set PCTFREE=0, if free space is needed for future inserts.
  If no inserts will be done on this table (read only) or data will always be added at the end of the table, set PCTFREE=0 and reorganize this dbspace to reclaim the free space.
- Reorganization should also be considered if a high amount of empty pages is detected in a dbspace, as these pages can only be reused by the same dbspace in this storage pool.
- Use LOCK=DBSPACE, PCTFREE=0 for read-only tables and consider the usage of NONRECOVERABLE storage pools.

5. Place VARCHAR column definitions at the end of the list of a create table command.
6. Place nullable column definitions just before VARCHAR columns when creating tables.
7. Avoid VARCHAR columns, unless the space saved by the use of VARCHAR is significant. There is an additional processing overhead for retrieval of varying length rows. The database manager will not use index-only access to retrieve the data if the index is created on a VARCHAR column. Especially with a length greater than 254 these columns are treated as LONG VARCHAR columns, and these columns can not be indexed.
8. The same restrictions and remarks apply to the usage of VARGRAPHIC columns. The difference is that the length is specified in double-byte characters. A length of greater than 127 makes the column to be handled a LONG VARGRAPHIC.
9. Use referential integrity instead of coding referential constraints into the application program (primary keys, foreign keys, unique constraints).
10. Create (temporary) copies of tables or joins of tables for query-only, if the tables are performance ″hot-spots″.
11. Control Center also creates a report that provides information about dbspaces that should be considered to be reorganized. Some criteria can be defined and the report will present a list of dbspace that apply to these criteria. A sample output for VM can be seen in Figure 16 on page 48, for the equivalent function on VSE refer to 4.2.3, “Dbspace Analysis Tool” on page 37.

9.3.2.4 Unchangeable Initialization Parameters

Use SHOW INITPARM in either ISQL or on the operator console to see the current setting. Refer to “Startup Parameters” in Chapter 14, “Database Operations” of *SQL/DS Version 3 Release 4 Performance Guide* for a detailed description of the initialization parameters and setting recommendations.

1. Buffers:
   - NPAGBUF is the number of 4K pages reserved in virtual storage for data and index. Default is (NCUSERS * 4)+10. Check the paging activity in the system and monitor the total DASD I/O with the COUNTER command. NPAGBUF should be at least 256 to take advantage of blocking (Multi-Block *BLOCKIO in VM or Extended user buffering in VSE).
   - NDIRBUF is the number of 512-byte directory pages reserved in virtual storage. The default value is NPAGBUF. It should be tuned in conjunction with NPAGBUF as described in 2.8.2, “Measurement and Tuning” on page 19.

   Note: The defaults are almost always much too small.

2. Lock Request Blocks:
   - NLRBS is the total number of Lock Request Blocks for the Application Server. The default value is

     NCUSERS*2 + (NCUSERS*NLRBU)/2 + 10
   - NLRBU is the number of Lock Request Blocks per user. The default value for NLRBU is 1000.
   - Too few lock request blocks cause frequent lock escalations. These can increase lock contention but can improve performance for the applications that are running without contention.
   - Too many lock request blocks only waste virtual storage. It is almost always better to have a small excess of lock request blocks, than to have too few.

   Notes:

   a. Monitor the values of LOCKLMT and ESCALATE given by the COUNTER command to optimize the setting of both NLRBS and NLRBU.
   If one or both of these values is greater than 0, try to determine which application has triggered the lock escalation.

   b. Only change the default values if there is no other way no reduce the amount of locks being held by the application. Refer to “Resource Contention” above item 19 on page 129.

   c. Increase NLRBU value if only a few applications have problems.

   d. Increase NLRBU and NLRBS if all applications have problems.

3. LTIMEOUT defines the period of time in seconds that will be allowed before a LOCK WAIT will timeout. This parameter should be specified if distributed units of work are performed. Use the LTIMEOUT value from the COUNTER command to monitor settings.

4. NCUSERS specifies the number of users that are allowed to be active at the same time. Too few NCUSERS causes varying response time for the same application. Too many NCUSERS could cause lock contentions, waste virtual storage and even cause paging.
   Use SHOW ACTIVE, SHOW CONNECT and CIRD (in VSE) to monitor user activities. If you have sufficient real storage available you may increase the number of concurrent users if required. Check your database directory for the number of pseudo agents (MAXCONN entry under VM, or RMTUSERS parameter under VSE).

Note: Please note that the total virtual storage consumption is around NCUSERS * 300kB.

5. NPACKAGE defines how many packages are allowed to be used in an LUW. Default value is 10. The size of the package cache is equal to NPACKAGE * NCUSERS.
6. NPACKPCT defines the percentage of the package cache that is used in the calculation of the package cache threshold which is calculated as (NPACKAGE * NCUSERS * NPACKPCT) / 100. If the number of packages that is in the cache exceeds this threshold, the database manager releases the package that has been in the cache the longest. While NPACKAGE and NCUSERS appear in the calculation, do not tune the threshold with them, rather use NPACKPCT.

9.3.2.5 Changeable Initialization Parameters

1. DISPBIAS is the fair share interval size.

   The DISPBIAS parameter determines the behavior of the database manager's dispatcher. For information on fair-share auditing, refer to Chapter 4 in the respective *System Administration* manual.
   If DISPBIAS is set to 1 the dispatcher operates more as if using a round-robin algorithm, thus providing equal service to each agent regardless of how much it consumes. On the other hand, if it is set to 10 the dispatcher operates more as if using a pure priority algorithm favoring short LUWs. Remember that the dispatcher will always use a combination of both algorithms. The default value for DISPBIAS is 7.
2. CHKINTVL see 2.6.3, "Checkpoint Statistics" on page 13.

Increase Availability with SET CHKINTVL

When you issue the on-line command ARCHIVE, a pre-archive checkpoint is taken while LUWs are in progress. The database archive begins and other agents can proceed. Another checkpoint, however, cannot be taken during the archive. Thus if a checkpoint is supposed to be taken, all new users will be in checkpoint wait until the archive is completed.

- SET CHKINTVL big enough to avoid a checkpoint during the archive period; afterwards, set it back to its normal value.

Notes:

1. When you increase CHKINTVL during an archive, make sure that you have enough extra space in your storage pools to hold the increased number of shadow pages. Otherwise SOSLEVEL might be reached and a checkpoint is triggered.
2. During an on-line archive, avoid modifying data in a NONRECOVERABLE storage pool. A checkpoint is forced at the end of every LUW doing that.
3. If you are regularly taking only on-line archives, then use LOGMODE=L. Check Chapter 2 of the "*DB2 Recovery on VSE and VM Using the Data Restore Feature*" for more details.

9.3.2.6 User Example: Shorten Database Maintenance

If database maintenance (adding or deleting dbextents, adding dbspaces) is needed, you must shut down your database server because these procedures run in single user mode.
If you **add dbspaces** you can do this with the same logmode (LOGMODE=A or L) you are normally running, to avoid logmode switching. SQLADBSP supports the PARM statement, which has the same syntax as the parm statement for the SQLSTART command, for example PARM(LOGMODE=A). After this the database can be started with the regular logmode and you can take your on-line archive as discussed in "Increase Availability with SET CHKINTVL" on page 136.
But, when you **add or delete dbextents** you cannot specify this parameter. Instead, you will be prompted with the message:

```
ARI6114A Do you want to do a database archive (ARCHIVE), user
archive (UARCHIVE), or no archive (NOARCHIVE) at the end of the run?
```

You have to either reply:

1. ARCHIVE, and this implies that the database will not be available for the time of the maintenance **and** the archive.
2. UARCHIVE, with the same remark as for ARCHIVE.
3. NOARCHIVE, which is not recommended, because after a dbextent is deleted, the database cannot be restored from an archive taken prior to the deletion. If an archive, taken before an add dbextent, is restored, the add dbextent has to be redone and all data residing on this previously added dbextent will be lost.

If ARCHIVE or UARCHIVE has been chosen, the logmode will be set to LOGMODE=A. For NOARCHIVE, the logmode will be set to LOGMODE=Y.

If you need to have the database available as soon as possible, you could specify UARCHIVE, and not execute the backup. Then start the database and take the on-line archive as discussed in "Increase Availability with SET CHKINTVL" on page 136.
If more than one step (add-delete dbextent) has to be performed in the same maintenance operation, specify NOARCHIVE for all, except the last, where you could specify UARCHIVE.

9.3.3 System

1. Check that the amount of CPU time for an application is not too high.
2. Check that the number of I/Os to the DASDs is not too high according to the application or work being done on the system. Check separately the DASDs for:
   - directory extent (BDISK)
   - catalog extents (storage pool 1)
   - log extent(s)
   - extents containing internal dbspaces
   - data extents
3. Check if any device or channel utilization is high.
4. Check that there is not too much paging.
5. Check that often used dbspaces have their extents on different DASDs.

6. Check that the DASDs used for the database are high speed devices.

For more details and checklists, refer to the operating system documentation for VSE. You will also find comprehensive checklists in the performance documents on the internet address *http://www.s390.ibm.com/vse*.

9.4 Performance Checklist for VSE

1. Check that your partition priorities are correct: POWER, VTAM, CICS, DB2 and so on.
2. Check that partition balancing is **not** used for the database.
3. Check that the NOFASTTR option is used in the database startup JCL.
4. Use transaction security to prevent usage of the CEDF debugging tool in production CICS.

9.5 Performance Checklist for VM

1. Check that you are using **saved segments**.

   A saved segment is a range of storage, set up to hold data or re-entrant code (programs), which can be shared by multiple virtual machines. Loading the DB2 components in saved segments can improve the virtual machine performance for the following reasons:

   - Less real storage is used
   - The paging I/O rate decreases
   - Less DASD space is needed for paging
   - The virtual machine can use its defined storage for other purposes because the saved segments can reside above

   We recommend that the Resource Adapter **always** be placed in a saved segment. If you frequently have multiple ISQL users, put ISQL into a saved segment. If you only run a single database server on your system, you do not need to place DBSS and RDS into saved segments.

2. Check that you are using VM Data Spaces Support (VMDSS).

   A data space is a virtual space with real pages being either in main storage or in expanded storage, and on DASD. Instead of accessing data with conventional I/O methods, data spaces are mapped to virtual storage and accessed using the paging subsystem of VM/ESA. The storage pool specifications file (ARISPOOL, see glossary) controls the VMDSS usage for each individual storage pool level:

   - If data spaces are used
   - Residence priority
   - If striping is used

   For more details regarding VMDSS, refer to the manual "*DB2 Server Data Spaces Support for VM/ESA*."

   Hint

   Optimize your database without VMDSS. A well-tuned database profits most from introducing VMDSS.

Exploit Benefits of VMDSS

3. Check that the MAPPING parameter value is L (logical mapping, mostly recommended - keeps pages in data space according to the dbspace) rather than P (physical mapping, might be better for mostly updating applications - keeps pages in data space according to the physical storing on DASD).
4. Check that you use striping for all storage pools. With striping, a storage pool fills all extents equally and all related data is physically close together on DASD; without striping, a storage pool fills one extent after the other. To become effective, data must be loaded or reorganized **after** striping has been turned on.
5. Use SHOW POOL to check the allocation of dbextents to pools.

   The recommendation given above for both VSE and VM applies even more with VMDSS striping, that all dbextents in the same POOL should have the same size but be on different physical I/O devices (or logical I/O devices in case of, for example, RAMAC or Multiprise Internal Disk), so that the operating system can read several groups of pages at the same time. With VMDSS, this can be exploited even by a single agent. Each POOL should have at least four dbextents to effectively exploit striping.
6. Check that you use VMDSS for the directory: You do so if the BDISK extent is formatted with 4kB pages; you don′t if it is formatted to 512B pages.
7. Tune your buffer size using hit ratios only. Do **not** reduce your directory buffer size just because you have VMDSS.
8. Check that your pool for internal dbspaces uses data spaces. This can be achieved either through the initialization parameter SEPINTDB=Y (unmapped, using paging space, paging DASD - which is preferred) or by mapping the internal dbspaces to a storage pool that uses data spaces (through the ARISPOOL file).
9. Check that for all storage pools data spaces are **used** rather than the standard database DASD I/O system. This does **not** mean that all storage pools should be **kept** in data spaces:

   a. For larger or low priority storage pools set the residence priority to 1 (release pages always when possible) or 2 (keep the index pages until TARGETWS is reached), so that these storage pools still profit from shorter path length, blocking, prefetching and asynchronous writing.
   b. Then consider to move data into data spaces using residence priorities 5 (keep pages even if beyond TARGETWS for frequent dbspace scans) or 4 (keep data pages until TARGETWS, and keep index pages even beyond - for high-used pages, randomly accessed using indexes). Start with small storage pools containing often used read-only tables and stay within the limits of TARGETWS (see below).
   c. Then consider giving residence priority 3 (keep data and index pages until TARGETWS is reached) to tables of medium size or priority, staying within your TARGETWS limit.

   If you put too much data in data spaces, your working storage will constantly exceed TARGETWS. Then the storage pools with medium or even high priority might be released and you do not gain much performance. Additionally CP might increase paging.

   To estimate when reaching TARGETWS, sum up the storage consumption of DB2 itself, depending on the parameters including the buffers. For residence priority 2 add the space for the active index pages, and for 3 and above add all the space for all active pages.

COUNTER POOL * shows you all storage pools and whether they use data spaces or not. From this display you can calculate the buffer hit ratios and, if applicable, the data space hit ratios you need for the steps described above.

10. Issue SHOW TARGETWS and check that the value of the TARGETWS initialization parameter is reasonable. Default is 32. It can range from 1 through 999,999 and be changed on-line with the operator command SET TARGETWS n, where n in MB is the sum of main storage plus expanded storage. When this threshold value is reached, DB2 begins to release data space pages according to the storage residence priority (see above).

    Try to put data in memory, if you have enough real storage to back up a reasonable share of virtual storage. Otherwise you would just replace **disk I/O** (a single user is waiting, and the data is written to its final destination disk) by **page I/O** (the whole database is waiting, and the final storing to the correct disk must still be done no later than the next checkpoint).
    If you spend too much virtual storage just for your database you might force CP to page - uncontrolled by DB2.

    Check that neither the system nor DB2 are paging too much. These values must be watched carefully when setting TARGETWS and when putting storage pools into data spaces (with residence priority higher than 1).

11. SHOW TARGETWS also shows the current working storage actually used. If the latter exceeds constantly TARGETWS, do the following:

    - Check that the storage residence priorities are not set too high. This value defines which storage pool pages are first released when TARGETWS is reached. Low values even mean asynchronous writing **before** this value is reached, thus saving misuse of virtual (real) storage for example, for seldom re-used data in large storage pools.
    - Check that there are not too many unmapped internal dbspace pages. If this is a problem, you can use mapped internal dbspaces instead, see above.
    - Check that not too many modified pages are kept in main or expanded storage (see SAVEINTV below).
    - Look at Appendix F in the "*DB2 Server Data Spaces Support for VM/ESA*" manual, for an explanation of why you will frequently see the TARGETWS value exceeded.

12. Check that the SAVEINTV initialization parameter is not too high. It can be set by the database operator command SET SAVEINTV n, where n is the number of blocks of 32 pages of 4kB. When in a data space this number of blocks of modified pages is exceeded, the database manager starts to save all modified pages to DASD without interrupting service.

    SAVEINTV can reduce the time a checkpoint needs; the value of CHKINTVL can be increased if the longer recovery time for the UNDO and REDO process is acceptable, if it was set to a lower value only because of the checkpoint processing time. The occurrence of this save process can be monitored by the SAVEGNRL value from the COUNTER INTERNAL command.

13. COUNTER INTERNAL * provides the values of internal data space counters. For an experienced user these can be helpful in tuning. They are explained in Appendix E, "Internal Counters" of "*DB2 Server Data Spaces Support for VM/ESA*."

9.6 Design Checklist

For a more detailed description of the design process and the considerations for performance, refer to Chapter 9, "Data Analysis and Design" in "*SQL/DS Version 3 Release 4 Performance Guide*."

9.6.1 Entities

Business

1. Does the business need to manage it?
2. Is it an implementation? What is the business thing itself? (Avoid words such as form, documentation and note.)
3. Is it at the right level? For example, is it an actual car (that is, registered vehicle) or a car type (manufacturer model)?
4. Is it a role, and if so, where is the actor?

Generalization - Specialization

5. Is it a candidate for generalization?
6. Are there several entities which represent the same thing at different stages of its life (for example, caterpillar and butterfly)?
7. Has it been over generalized, is it a candidate for specialization?

Dependent Entities

8. Will every occurrence of the entity be related to one, and only one, occurrence of each of its parent entities?
9. Re-examine the list of rejected entities to ensure that earlier decisions are still valid.

9.6.2 Attributes

1. Does the attribute have a single meaning?
2. Does the attribute have a consistent and exclusive set of values?
3. May this attribute have multiple values for the same entity occurrence?
   a. At a particular moment?
   b. Changing over time?
4. If many occurrences of this entity may have the same value for this attribute, are the values of individual significance to the business?
5. Are there attributes which do not apply for some entity occurrences?
6. If this attribute is thought to be derivable (for example, Cost = Price * Quantity), can its value be derived for **all** applicable occurrences?
7. Is this attribute a piece of text that contains smaller pieces of text that are of themselves of business significance?
8. Is this property an attribute of *this* entity, or of some other entity within the business?

9.6.3 Normalization

1. First do a logical database design to 3NF. Then, denormalizing back towards 2NF should be considered during physical database design.
2. 3NF is recommended. 2NF could be considered for the case where there are no updates and data integrity is not a concern. 1NF should not be considered as a good solution because of the complexities. It is better to keep the number of tables as small as possible, from a performance point of view.
3. What processing takes place against these columns? If often updated, 3NF is better:
   - You can reduce locking problems.
   - You do not need to update redundant data.

   If mainly selected, 2NF is better.
4. How often are the involved columns referenced together?

   If these columns are referenced together many times in joins, you may need the 2NF.
5. If you use 2NF, how much additional DASD space will be required?

Note: The consequence of 3NF is DASD space saving, because there is no redundant data, except for keys. Normalization should be used to save space.

9.6.4 Data Types

Use efficient and correct data types for the column definitions whenever possible. Efficient data types normally reduce the row length, with the result that more rows fit on a page and thus reduce I/O. In addition, it enables the optimizer to calculate more accurate filter factors.

The following items must be considered when defining the data types for columns:

1. Avoid CHARACTER for purely numeric data.
2. Use DATE, TIME or TIMESTAMP where appropriate - do **not** use CHARACTER or INTEGER if one of those is applicable.
3. Use SMALLINT or INTEGER for numeric and integer data. If the values are between -32767 and 32768 use SMALLINT.
4. Specify NOT NULL whenever appropriate. Allowing NULLs adds another byte for each column.
5. Use CHARACTER instead of VARCHAR, which needs an extra two bytes for length. VARCHAR columns should be avoided unless significant savings in DASD storage are achieved.
6. If you have to use VARCHAR, define VARCHAR columns always at the end of the row.
7. Avoid LONGVARCHAR. VARCHAR(n) with n > 254 are treated as LONGVARCHAR.

9.6.5 Physical Design

Once the data modeling has been done (logical level), it must be translated to DB2 Data Definition (physical level).

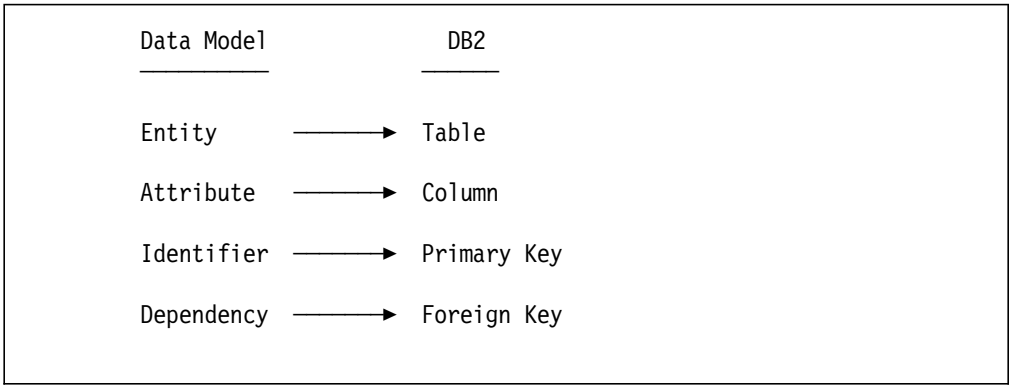


Figure 91. Basic Translation from Data Model to DB2

Figure 91 on page 143 depicts the basic translation from the Data Model to an RDBMS.

However, the following performance topics must be kept in mind when defining the operational (physical) databases:

1. The definition and use of indexes
2. Splitting a table

   Views may be used to achieve the same effect as splitting. Physical splitting of a table, especially moving columns into a different table, can improve response time where lock contention or the table size might affect performance.
3. Planned redundancy for performance reasons, see 9.6.3, “Normalization” on page 142
4. Considerations for referential integrity
5. Allocate only one table to a dbspace to increase flexibility of control

Appendix A. Installed Environment

The following gives an overview of the installed hardware and software at the ITSO. In addition, examples were taken from different customer installations.

Processor

- IBM ES/9000 9672 Model R72

Tape Unit

- IBM 3490

Operating System

- VM/ESA Version 2 Release 2
  with Minidisk Caching switched on.
- VSE/ESA Version 2 Release 2

VSE is running on VM.

Database Server

- Tests on DB2 Server for VSE & VM Version 5 Release 1

Database Layout

The database was defined with 14 storage pools. Each table has its own dbspace and its own storage pool. The internal dbspaces were defined on a virtual disk or in a dataspace.

VSE Settings

- VSE machine with SHARE RELATIVE 100 and QUICKDSP ON
- Total number of database pages 1,675,264
- Directory disk (BDISK) 80 cylinders on RAMAC (logically 3380)
- Tape unit is 3490

Appendix B. Screens and Reports of VM:DB/Monitor

The following appendix gives samples of all predefined panels and reports available with VM:DB/Monitor.

B.1 IDBM Sample Screens

To start the on-line part of VM:DB/Monitor, enter IDBM. Then you see the following screen where you need to enter the CONNECT command with the name of the VM:DB/Monitor Virtual Machine (DBMONVM), the GETDATA and the DISPLAY command.

```
MAIN             V M : D  B  /  M  O  N  I  T  O  R             VM:DB/Monitor
--------------------------------------------------------------------------------
                               Release 4.00


               Copyright (c) 1998, Sterling Software, Inc.
--------------------------------------------------------------------------------


            To access VM:DB/Monitor data, do the following steps:


  Step                Function                                  Command
  ---- ----------------------------------------     -------------------------
   1.  Connect to a VM:DB/Monitor Virtual Machine ==> CONNECT TO dbmonvm-id
   2.  Request data                               ==> GETDATA
   3.  Display the data                           ==> DISPLAY


--------------------------------------------------------------------------------
PF 1=Help       2=           3=Return     4=Top        5=Bottom     6=Retrieve
PF 7=Backward   8=Forward    9=          10=Left      11=Right     12=(Un)Fold

====>
```

Figure 92. IDBM - Startup Screen

As a fast path you can create a file named PROFILE IDBM with the three commands, and you will always see directly the main menu whenever you enter IDBM.

```
CONNECT TO DBMONVM
GETDATA
DISPLAY
```

Figure 93. PROFILE IDBM

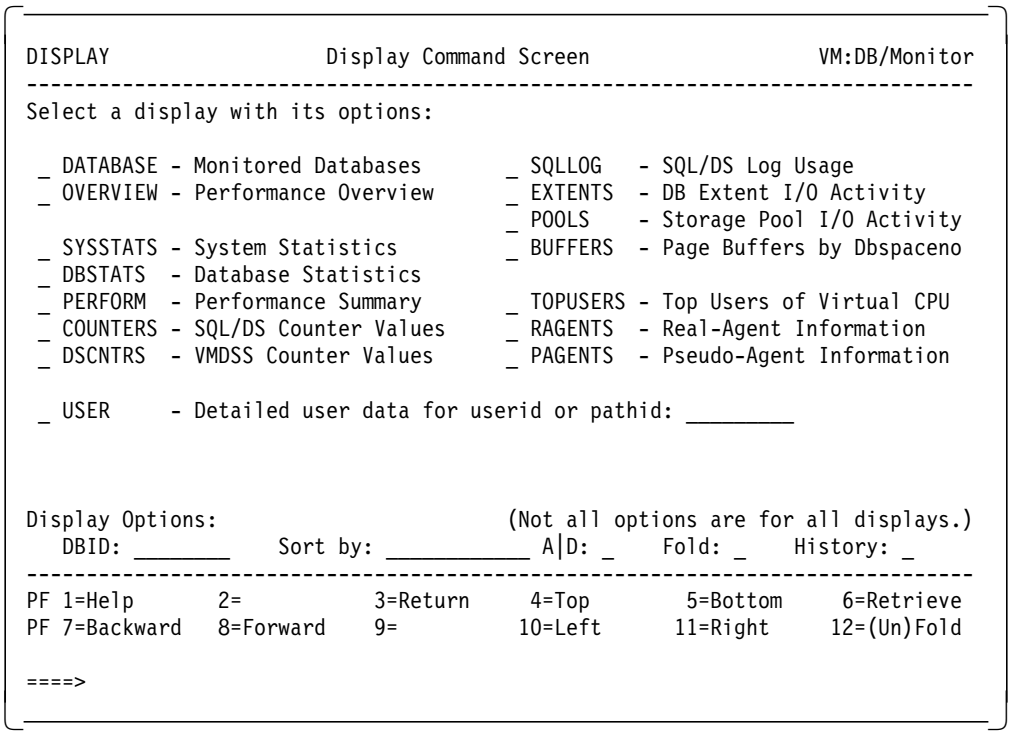


Figure 94. IDBM - Main Menu

The BUFFERS panel gives information about buffer usage by dbspace.

- NPAGBUF defined as startup parameter
- Total of page buffers in use by the database
- Number of page buffers in use by dbspace number

```
BUFFERS            Count of Page Buffers by DBspaceno            Panel  1 of  1
-------------------------------------------------------------------------------
Database  Time  NPAGBUF   In-Use   DBspno Count   DBspno Count   DBspno Count
FVVPDBA   14:46    4000     3244      385   455      296   317        1
                                      74   229      200   177      290
                                     142   129      287    94      276
                                     168    72      363    67      343
                                     264    62      275    61       40
                                     291    58       13    51      223
                                     338    40    32001    38      366
                                     260    32      203    31      236
                                       2    27      141    24      225
                                     154    23      376    22      259
                                     294    20       78    19      185
                                     136    14       43    13       60
                                      17    12      164    12      263
                                     421    12       89    11       42
                                      26    10       95    10      119
-------------------------------------------------------------------------------
PF 1=Help       2=           3=Return     4=Top        5=Bottom     6=Retrieve
PF 7=Backward   8=Forward    9=          10=Left      11=Right     12=(Un)Fold

====>
```

Figure 95. IDBM BUFFERS - Page Buffers by Dbspaceno

The COUNTERS panel displays the same data as is presented by the COUNTER operator command. Detailed information on each field can be found in the DB2 manuals.

```
COUNTERS                  SQL/DS Counter Values                  Panel  1 of  1
------------------------------------------------------------------------------
Database  Time    RDSCALL  DBSSCALL  BEGINLUW  ROLLBACK  CHKPOINT   LOCKLMT
FVVPDBA   14:54   1512624   9582592    101080       273        16         0

                 ESCALATE  WAITLOCK  DEADLOCK  LPAGBUFF  PAGEREAD  PAGWRITE
                       26        15         0  19056728   1354896    609354

                 LDIRBUFF   DIRREAD  DIRWRITE   LOGREAD  LOGWRITE  DASDREAD
                  3537461     47923     50041     11359     26506   1414170

                 DASDWRIT    DSREAD   DSWRITE    DASDIO  Reset Time
                   685899         0         0   2100077  01/28/98 18:16:39


------------------------------------------------------------------------------
PF 1=Help       2=           3=Return     4=Top        5=Bottom     6=Retrieve
PF 7=Backward   8=Forward    9=          10=Left      11=Right     12=(Un)Fold

====>
```

Figure 96. IDBM COUNTERS - DB2 Counter Values

The DATABASE panel shows information about the databases on which the monitor is running.

Some general information is displayed as:

- Database name
- Resource
- CP and CMS level
- Virtual machine mode (XC)
- If shared segments are being used
- Initialization time and last monitoring time

```
DATABASE                    List of Databases                    Panel  1 of  1
-------------------------------------------------------------------------------
Database  Nodeid    DBname              DB Resid  Initialization Time
ELDB2A    BOEVMCT1  ELDB2A              ELDB2A     01/15/98 10:17:23

          SQL/DS  DCSS  CP          CMS  Machine  Last Mon Time
           5.1DS    N   ESA 2.0      13    XC        10:51:33

Database  Nodeid    DBname              DB Resid  Initialization Time
ELDB2B    BOEVMCT1  ELDB2B              ELDB2B     01/13/98 13:56:02

          SQL/DS  DCSS  CP          CMS  Machine  Last Mon Time
           5.1DS    N   ESA 2.0      13    XC        10:52:15

-------------------------------------------------------------------------------
PF 1=Help       2=           3=Return     4=Top        5=Bottom     6=Retrieve
PF 7=Backward   8=Forward    9=          10=Left      11=Right     12=(Un)Fold

====>
```

Figure 97. IDBM DATABASE - Monitored Databases

The DBSTATS panel gives information about a database.

- NCUSERS defined as database startup parameter
- What was the highest value (HWM, high water mark) reached for real agents (RAGT) and when did it happen.
- What was the highest value and when was it reached for pseudo agents (PAGT); also the total number of pseudo agents available (Pavail).
- The maximum number of programs (MaxPrg) that can be loaded and the number of programs loaded (InUse) at the end of the interval
- NLRBS and NLRBU definition
- The number of lock request blocks in use and the HWM
- The number of lock waits and the HWM

This panel and the related report can be very useful to adjust some startup parameters of the database.

There is also a possibility to create history information from the monitor files. Only the DATABASE info, DATABASE stats and TOPUSERS log subrecords are used to create the history record. From these history files, reports as specified before can be created to give an overview over a longer period of time.

```
DBSTATS                    Database Statistics                   Panel  1 of  1
--------------------------------------------------------------------------------
Database  Time   NCUser  Ragt--HWM--HWM Time  Pagt--HWM--HWM Time  Pavail
FVVPDBA   10:55      12      3   12  09:00:33    99  126  10:39:53      97

                 Pwait--HWM--HWM Time  PagBuf-InUse  DirBuf-InUse  MaxPrg-InUse
                     0    3  16:00:45   4000   3597  15000  13798     720   312

                  NLRBS  NLRBU    LRBs---HWM--HWM Time  Lock Wait HWM--HWM Time
                  35000   7300       0  7300  21:27:28         0    1  14:36:49

                  RespTime
                 000:00.114

--------------------------------------------------------------------------------
PF 1=Help      2=          3=Return     4=Top        5=Bottom     6=Retrieve
PF 7=Backward  8=Forward   9=          10=Left      11=Right     12=(Un)Fold

====>
```

Figure 98. IDBM DBSTATS - Database Statistics

The DSCNTRS panel displays the same data as is presented by the COUNTER INTERNAL operator command. Detailed information on each field can be found in the DB2 manuals.

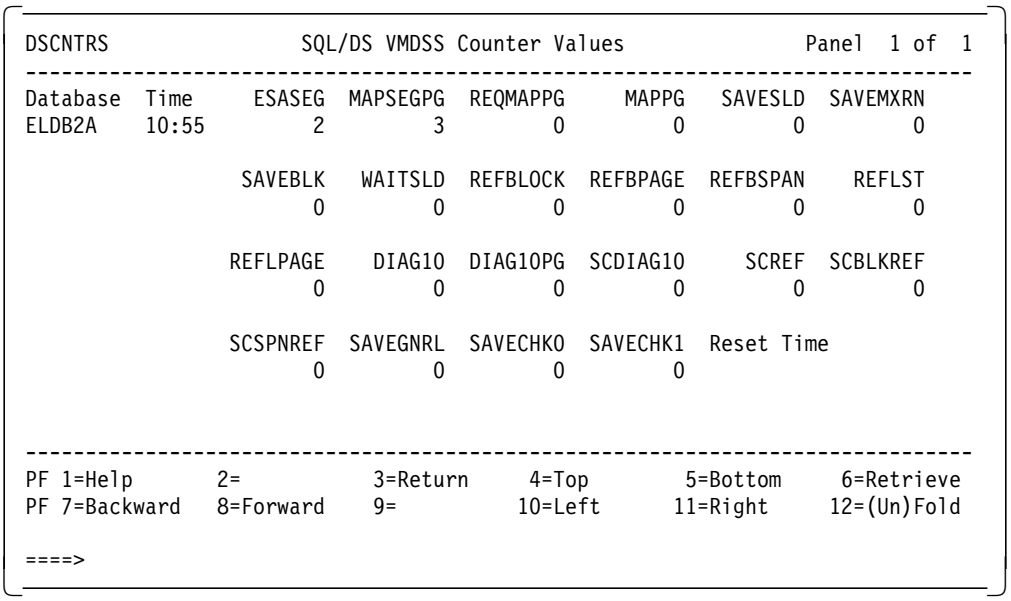


Figure 99. IDBM DSCNTRS - VMDSS Counter Values

The EXTENTS panel gives information about I/O by dbextent.

- Total number of read and write operations
- for each individual disk (dbextent)
  - the pool number
  - the number of read and write operations

```
EXTENTS                I/O Activity by DBextent                Panel  1 of  1
--------------------------------------------------------------------------------
Database  Time      Reads    Writes  Vaddr  DDname    Pool     Reads    Writes
FVVPDBA   14:46   6261940    347357   0200  BDISK      N/A    408579    2
                                      0201  LOGDSK1    N/A         9    1
                                      0210  DDSK1        1    102546
                                      0211  DDSK108      1      3588
                                      0220  DDSK2        2         0
                                      0221  DDSK92       2         0
                                      0222  DDSK93       2         0
                                      0223  DDSK94       2         0
                                      0224  DDSK95       2         0
                                      0230  DDSK3        3     14435
                                      0231  DDSK58       3         0
                                      0240  DDSK4        4     11409
                                      0241  DDSK96       4         0
                                      0242  DDSK97       4         0
--------------------------------------------------------------------------------
PF 1=Help       2=           3=Return     4=Top        5=Bottom     6=Retrieve
PF 7=Backward   8=Forward    9=          10=Left      11=Right     12=(Un)Fold

====>
```

Figure 100. IDBM EXTENTS - DB Extent I/O Activity

The OVERVIEW panel gives you warning indications about thresholds being exceeded.
The column named *Thresholds* keeps different values that can be user-defined or have default values. The colors of these threshold values and operands are from left to right: yellow, red and white.
If a threshold is exceeded, the column named *Field Name* will be displayed in the same color (yellow, red or white) as the exceeded threshold column.
An example would be: real agents value is 6. This means that the *Field Name* Real Agents will be displayed in red, because the second threshold (> 5) was attended.

```
10:55                          DB/MONITOR Overview
-------------------------------------------------------------------------------
10:53   FVVPDBA  | Field Name         Value   |           Thresholds
-----   --------  | ---------------  -------  |-------------------------------
        Response  | Total Response     0.114  | > 0.75     > 1.5      > 3.0
                  | User Response      1.27   | > 1.5      > 3.0      > 10.0
                  |                           |
        Users     | Real Agents        0      | > 3        > 5        > 10
                  | Pseudo Agents      99     | > 5        > 10       > 25
                  | P-Agents Waiting   0      | > 2        > 5        > 15
                  |                           |
        I/O       | Page Buffer Use    92.9%  | < 80.0     < 50.0     < 10.0
                  | Dir Buffer Use     98.7%  | < 50.0     < 25.0     < 10.0
                  | I/O Rate           110    | > 100      > 200      > 500
                  | Read Rate          76     | > 80       > 150      > 300
                  | Write Rate         34     | > 50       > 100      > 200
                  |                           |
        Locking   | Lock Wait          0      | > 2        > 3        > 5
                  | Lock Wait Time            | > 3        > 5        > 10
                  |                           |
        Log       | % Log Full         1.28%  | > 60       > 80       > 90
                  | CHKPOINT Rate      0      | > 2        > 3        > 5
-------------------------------------------------------------------------------
 PF:  1=Help   3=Return   6=IDBM   9=Activity         Other=Refresh
```

Figure 101. IDBM OVERVIEW - Performance Overview

The PAGENTS panel gives information about users currently connected to the database. A pseudo agent can also be a real agent. Among other, the following information is shown:

- Last active time
- Current status and length of time in this status
- Virtual CPU time since connecting
- Last package and section used

```
PAGENTS                 Pseudo-Agent Information                  Panel  1
------------------------------------------------------------------------------
Database: ELDB2A     Time: 13:42
Userid   SQLid    Path  ConnTime State StatTime  Virt-CPU  User-CPU  BufLooks
ELRES3   ELRES3     32  04:49:13 IDLE  00:31:25 00:00.252 00:00.075      5586

          RealTime  %-RUN %COMM %LOCK %-I/O %CHKP %OUTP %OUTB %DSPF %-NIW
         000:23.465     1    91     0     7     0     0     0     0     0

         Last Package              (Last Interval:) %Disp  %VCPU  %LPAGB
         SQLDBA  .ELORXSQL.42                          0      0       0

          RespTime   NLUWs    MaxLUW      AvgLUW
         000:00.033     17  000:13.158  000:01.380


------------------------------------------------------------------------------
PF 1=Help       2=           3=Return     4=Top        5=Bottom     6=Retrieve
PF 7=Backward   8=Forward    9=          10=Left      11=Right     12=(Un)Fold

====>
```

Figure 102. IDBM PAGENTS - Pseudo-Agent Information

The PERFORM summary panel displays information that is displayed in detail on other panels.

- Wait lock and deadlock information
- Page and directory buffer usage
- DASD I/O and data space I/O

```
PERFORM                    Performance Summary                    Panel  1 of  1
--------------------------------------------------------------------------------
Database  Time   LUWs/min CHKP/min DBSSCall Waitlock Deadlock  PagBuff  DirBuff
FVVPDBA   14:54    29.500    0.000   84.21%    0.33%    0.00%   92.89%   89.90%

                 DASDIO/min  DS-IO/min  DataSpace    VMDSS    Read Only
                     55.000        N/A      N/A%       N/A%     63.76%

--------------------------------------------------------------------------------
PF 1=Help       2=           3=Return     4=Top        5=Bottom     6=Retrieve
PF 7=Backward   8=Forward    9=          10=Left      11=Right     12=(Un)Fold

====>
```

Figure 103. IDBM PERFORM - Performance Summary

The POOLS panel is similar to the EXTENTS panel above, but here the data is given by storage pool instead of by dbextent.

- Whether this pool is enabled for dataspace usage or not
- Number of extents
- Number of buffer looks
- Number of read and write operations
- Total number of I/O

```
POOLS                 I/O Activity by Storage Pool               Panel  1 of  1
------------------------------------------------------------------------------
Database  Time   Pool  I/O Type  Extents  BufLooks    Reads   Writes  I/O Cnt
ELDB2A    10:55   Total:                    15429      722      303     1025
                  INT  DSO SEQ         0       45        0        0        0
                  DIR  BLK SEQ         1     1414      567      299      866
                    1  BLK STR         1    13932      152        4      156
                    2  DS3 STR         1       38        3        0        3
                    3  BLK STR         1        0        0        0        0
                    4  BLK STR         1        0        0        0        0
                    5  BLK STR         1        0        0        0        0
------------------------------------------------------------------------------
PF 1=Help       2=           3=Return     4=Top        5=Bottom     6=Retrieve
PF 7=Backward   8=Forward    9=          10=Left      11=Right     12=(Un)Fold

====>
```

Figure 104. IDBM POOLS - Storage Pool I/O Activity

The RAGENTS panel gives information about real agents.

- LUW starttime and status
- CPU usage
- Number of locks
- Number of rows retrieved
- Package name and section

```
RAGENTS                  Real Agent Information                   Panel  1
------------------------------------------------------------------------------
Database: ELDB2A     Time: 13:42
Userid   SQLid     Path  Agt  LUWStart  State Stattime  Actvtime   Virt-CPU
ELRES5   ELRES5      32    1  13:32:07  RUN   00:00:00  00:10:07  003:10.00

          User-CPU   BufLooks       Locks        Rows  Current Package
         000:03.312    556928          33         336  ELRES5  .PBPB998 .4


------------------------------------------------------------------------------
PF 1=Help       2=           3=Return     4=Top        5=Bottom     6=Retrieve
PF 7=Backward   8=Forward    9=          10=Left      11=Right     12=(Un)Fold

====>
```

Figure 105. IDBM RAGENTS - Real Agent Information

The SQLLOG panel gives output similar to that presented by the SHOW LOG command.

- Log size
- Log pages in use
- Checkpoint interval
- Number of log pages to be used before a checkpoint will occur
- Number of checkpoints

```
SQLLOG                      SQL/DS Log Usage                     Panel  1 of  1
------------------------------------------------------------------------------
Database  Time     Log-Size     Log-Used  %-Full  CHKINTVL  PgRemain  CHKPOINT
FVVPDBA   14:46   306774016      3851144   1.26%      2500      1716         1

------------------------------------------------------------------------------
PF 1=Help       2=           3=Return     4=Top        5=Bottom     6=Retrieve
PF 7=Backward   8=Forward    9=          10=Left      11=Right     12=(Un)Fold

====>
```

Figure 106. IDBM SQLLOG - SQL/DS Log Usage

If you select the HISTORY Display option you will see the following screen, which shows not only the values of the current interval, but also the values of the previous ones.

```
SQLLOG                      SQL/DS Log Usage                     Panel  1 of 1
------------------------------------------------------------------------------
Database  Time     Log-Size     Log-Used  %-Full  CHKINTVL  PgRemain  CHKPOINT
FVVPDBA   14:46   306774016      3851144   1.26%      2500      1716         1
FVVPDBA   14:44   306774016      3834493   1.25%      2500      1720         1
FVVPDBA   14:42   306774016      3703569   1.21%      2500      1752         1
FVVPDBA   14:40   306774016      3694408   1.20%      2500      1755         1
FVVPDBA   14:38   306774016      3689686   1.20%      2500      1756         1
FVVPDBA   14:36   306774016      3677062   1.20%      2500      1759         1
FVVPDBA   14:34   306774016      3643793   1.19%      2500      1767         1
FVVPDBA   14:32   306774016      3631389   1.18%      2500      1770         1
FVVPDBA   14:30   306774016      3621840   1.18%      2500      1772         1
FVVPDBA   14:28   306774016      3611424   1.18%      2500      1775         1
FVVPDBA   14:26   306774016      3601039   1.17%      2500      1777         1
FVVPDBA   14:24   306774016      3591690   1.17%      2500      1780         1
FVVPDBA   14:22   306774016      3571824   1.16%      2500      1784         1
FVVPDBA   14:20   306774016      3554899   1.16%      2500      1789         1
FVVPDBA   14:18   306774016      3531572   1.15%      2500      1794         1
FVVPDBA   14:16   306774016      3517791   1.15%      2500      1798         1
------------------------------------------------------------------------------
PF 1=Help       2=           3=Return     4=Top        5=Bottom     6=Retrieve
PF 7=Backward   8=Forward    9=          10=Left      11=Right     12=(Un)Fold

====>
```

Figure 107. IDBM SQLLOG - SQL/DS Log Usage HISTORY

The SYSSTATS panel has combined information about the virtual machine and the database.
The information about the virtual machine is similar to the output given by the CP IND USER command.

- CPU usage
- Working set
- Number of resident pages in main and expanded storage
- Number of page reads and writes
- MAXCONN setting in the CP directory

Database information

- Setting of TARGETWS and actual value
- Number of dbextents
- Number of pools
- Monitor Overhead, total value during the last interval

```
SYSSTATS                    System Statistics                    Panel  1 of  1
------------------------------------------------------------------------------
Database  Time    VCPU   TCPU Comb.I/O  PgReads PgWrites WorkSet ResMain ResExp
FVVPDBA   14:54 630:30 820:08  6611109    51640    38284    7636    7682      0

                 VirtStor %Stor  TargetWS  ActualWS  MAXCONN  #DBext  #Pools
                      80M   58%       N/A        31      333     137      15

                 OHDTotal  OHDIntvl
                    0.00%     0.00%

------------------------------------------------------------------------------
PF 1=Help       2=           3=Return     4=Top        5=Bottom     6=Retrieve
PF 7=Backward   8=Forward    9=          10=Left      11=Right     12=(Un)Fold

====>
```

Figure 108. IDBM SYSSTATS - System Statistics

The TOPUSERS panel gives information about users that have the highest value for Virtual CPU usage.

```
TOPUSERS                Top User of Virtual CPU                 Panel  1 of  1
------------------------------------------------------------------------------
Database  Time   Userid   SQLid      Dispatches     Virt._CPU      BufLooks
FVVPDBA   14:46  FVV422   FVV             13934      0:26.865         5381
                 FVV416   FVV             13452      0:29.100         7593
                 FVV505   FVV             14866      0:33.234         5535
                 FVV466   FVV             15438      0:34.097         8840
                 FVV730   FVV             11690      0:34.630         7454
                 FVV412   FVV             19023      0:39.104         7589
                 FVV415   FVV             27557      0:57.202        10782
                 FVV414   FVV             31348      1:01.650        10714
                 FVV424   FVV             35693      1:21.269        20613
                 FVV421   FVV             44492      1:27.792        20498
------------------------------------------------------------------------------
PF 1=Help       2=           3=Return     4=Top        5=Bottom     6=Retrieve
PF 7=Backward   8=Forward    9=          10=Left      11=Right     12=(Un)Fold

====>
```

Figure 109. IDBM TOPUSERS - Top Users of Virtual CPU

The USER panel gives all kinds of detailed information about a selected user ID and also for an active LUW, if any. Among others it shows:

- CPU usage
- Wait states
- Current package, section and opcode
- Dbspace number and table id currently being handled

```
USER                        Detailed User Data                      Panel  1
------------------------------------------------------------------------------
Database: ELDB2A     Time: 13:42
Userid   Path  Protocol Req-Type   Node                Conntime  REAL WAIT IDLE
ELRES4     33  SQLDS    SQLDS/VM   BOEVMCT1            00:02:24   95%   0%   5%
         State Stattime  RealTime   RUN COMM LOCK  I/O CHKP OUTP OUTB DSPF  NIW
         REAL  00:01:40 000:37.343   0% 100%   0%   0%   0%   0%   0%   0%   0%
         NLUWs  Max. LUW   Avg. LUW   DispCnt   Virt-CPU  User-CPU  BufLooks
             3 000:37.300 000:12.441       29  00:00.037 00:00.028       312
         Resp. Time   Last Package           (Last Interval) Disp  VCPU  LPGB
         000:00.005   SQLDBA  .ARIISQL .4                     91%   67%  100%

(Current LUW)
SQLid    Agent State Stattime  LUW Time  RUN COMM LOCK  I/O CHKP OUTP OUTB DSPF
ELRES4       2 RUN   00:00:00  00:00:00   0%   0%   0%   0%   0%   0%   0%   0%
         LUWID  DispCnt  Virt-CPU  User-CPU  BufLooks  RowCount   Locks L-Locks
          2D02        0 00:00.000 00:00.000         0         0       0       0
         Package:       SQLDBA  .ARIISQL .4
         RDIIN Opcode:    0 Unknown Call
         DBSS  Opcode:    0 Unknown Call
         DBspaceno:       0    Table-id:        0

 Last SQL Statement in this LUW:      (Estimated Cost: 16      )
 SELECT * FROM ELRES5.PBQMRIP


------------------------------------------------------------------------------
PF 1=Help       2=           3=Return     4=Top        5=Bottom     6=Retrieve
PF 7=Backward   8=Forward    9=          10=Left      11=Right     12=(Un)Fold

====>
```

Figure 110. IDBM USER - Detailed User Data

The INPUT panel is a user defined panel. It can display any information of any other panel, or even display a combination of data from different panels.

Figure 112 on page 165 shows a user written program that combines the SYSSTATS and DBSTATS displays for a specific database.

```
INPUT                    Statistics For ELDB2B                  Panel  1 of  1
------------------------------------------------------------------------------
Database  Time    VCPU   TCPU Comb.I/O  PgReads PgWrites WorkSet ResMain ResExp
ELDB2B    14:03 000:04 000:08      557      131       20    1451    1473      0

                 VirtStor %Stor  TargetWS  ActualWS  MAXCONN  #DBext  #Pools
                      16M   39%        32         6      159       4       2

                 OHDTotal  OHDIntvl
                    0.00%     0.00%


Database  Time   NCUser  Ragt--HWM--HWM Time  Pagt--HWM--HWM Time  Pavail
ELDB2B    14:03       5     4    4  13:42:38     4    4  13:42:38     151

                 Pwait--HWM--HWM Time  PagBuf-InUse  DirBuf-InUse  MaxPrg-InUse
                     0    0               30     30     30      3      50     0

                  NLRBS  NLRBU    LRBs---HWM--HWM Time  Lock Wait HWM--HWM Time
                   2520    1000      0     0                   0    0

                  RespTime
                 000:00.001

------------------------------------------------------------------------------
PF 1=Help       2=           3=Return     4=Top        5=Bottom     6=Retrieve
PF 7=Backward   8=Forward    9=          10=Left      11=Right     12=(Un)Fold

====>
```

Figure 111. IDBM STTS - User Defined Panel

```
/* ------------------------------------------------------------*/
/* STTS * V01.00 * 19.01.1998 * A.Wizemann * ITSO * * ...    */
/* IDBM                                                       */
/* ------------------------------------------------------------*/
 arg dbid .                          /* Check for a database.*/
 if dbid = '' then do
 'MESSAGE A database must be specified for the STTS macro.'
    exit 8
    end
 /*-----------------------------------------------------------*/
 'GETDATA'                           /* Request some data.   */
 if rc <> 0 then exit                /* If an error, exit.   */
 /*-----------------------------------------------------------*/
 /* Create SYSSTATS display.                                  */
 /*-----------------------------------------------------------*/
 'DISPLAY SYSSTATS FOLD DBID' dbid
 if rc <> 0 then exit                /* If an error, exit.   */
 'EXTRACT DISPLAY SYS'               /* Extract display stem.*/
 /*-----------------------------------------------------------*/
 /* Create DBSTATS  display.                                  */
 /*-----------------------------------------------------------*/
 'DISPLAY DBSTATS  FOLD DBID' dbid
 'EXTRACT DISPLAY DBS'               /* Extract display stem.*/
 /*-----------------------------------------------------------*/
 /* Begin to build the new display in the stem NEW.           */
 /*-----------------------------------------------------------*/
 new.title = 'Statistics For' dbid   /* Set the new title.   */
 new.panels = 1                      /* Set the nmbr panels. */
 n = 0
 do i = 1 to sys.0                   /* Copy SYSSTATS display*/
    n = n + 1                        /* to the NEW stem.     */
    new.n = sys.i
    end                              /* skip a line between  */
 n = n + 1                           /* displays.            */
 new.n = ' '
 do i = 1 to dbs.0                   /* Copy DBSTATS display */
    n = n + 1                        /* the NEW stem.        */
    new.n = dbs.i
    end                              /* set number of        */
 new.0 = n                           /*  of display lines.   */
 /*-----------------------------------------------------------*/
 /* Move the new display back into IDBM.                      */
 /* Exiting will cause the display to be written to the       */
 /*   screen.                                                 */
 /*-----------------------------------------------------------*/
 'INPUT DISPLAY NEW'

 RETURN
```

Figure 112. IDBM Sample Program for Screen Combination

B.2 DBMAP Activity Reports

The activity records can be printed in a summary form using one of the 13 summary reports, or each record entry can be printed using the ACTDETL activity report.

For almost all the reports you can use the SORT option to specify that the report will be sorted by a specified column number in either ascending or descending order. The only exceptions are the ACTDETL and LUWSTATS reports.

Normally the summary activity reports are divided into two parts: Part 2 is only a continuation of part 1.

Activity Reports Definitions

In the sequence occurring on ACTDETL

- Dispatch count: number of times an agent was dispatched for this session.
- Path: IUCV path id the LUW used.
- Trans: single RDIIN operation.
- UVCPU: sum of the virtual CPU consumption on the VM user ID in milliseconds.
- VCPU: sum of the virtual CPU consumption in milliseconds in the database machine.
- LPAGBUF: looks in page buffers.
- Max Cost: estimated cost for dynamic SQL statements prepared by this package.
- RUN COMM LOCK CHKP OUTP OUTB I/O DSPF: the states the real agent can be. You can find the meaning of each wait state in the manual : "*DB2 Server for VM Operation*"
- Response time: time from when the first request is received for the LUW to the first COMM wait. The user should have seen some database response by this time.
- Transaction: single RDIIN operation.
- Agent VCPU: sum of the virtual CPU consumption in milliseconds in the database machine.
- User VCPU: sum of the virtual CPU consumption on the VM user ID in milliseconds.
- Dispatch / Wait: how often and how much time a real agent was in RUN, COMM, LOCK, CHKP, OUTP, OUTB, I/O or DSPF states. You can find the meaning of each wait state in the manual *DB2 Server for VM Operation*.
- Retcode: SQLCODE returned by the last SQL statement executed by the LUW.
- SQLSTATE: SQLSTATE returned by the last SQL statement executed by the LUW.
- Table-id: column TABID of the SYSTEM.SYSCATALOG table.
- Index-id: column IID of the SYSTEM.SYSINDEXES table.

Not on ACTDETL

- Total time active: absolute time that a real agent was assigned to a pseudo agent.
- Elapsed: time that a pseudo agent was assigned to the user ID.
- Session states: relation between how much time a real agent was assigned to the pseudo agent.

ACTDETL Activity Report

This report lists each record of the activity file.

```
01/29/98 15:23:00  Session Started:            Database: ELDB2A   at BOEVMCT1
                   Userid: ELRES5    Path:    32
                   SQL/DS Dispatch Count:    144794  Virtual CPU:    212879  LPagBuf:   1090015
-------------------------------------------------------------------------------------------------------------------------------
01/29/98 15:23:00  LUW Started:               Database: ELDB2A   at BOEVMCT1
                   Userid: ELRES5    Path:    32  Agent:     2
01/29/98 15:24:45  Package Usage:             Database: ELDB2A   at BOEVMCT1
                   Userid: ELRES5    Path:    32  Agent:     2
                   SQL ID: ELRES5    LUW ID:    429B2
                   Package            Trans    UVCPU     VCPU  LPagBuf  Row-Cnt Max-Cost  RUN COMM LOCK CHKP OUTP OUTB  I/O DSPF
                   ELRES5.CTPB110     39602    30539    45426   211099        1    0.000  40%  51%   0%   0%   0%   0%   9%   0%
                   Section:    1  Stmtno:    0  Opcode:  30  Type:                                   RetCode
colon.      0  SQLSTATE: 00000
                                         1                2        7        0          100%        0%   0%   0%   0%   0%   0%
                   Section:    2  Stmtno:    0  Opcode:  30  Type:                                   RetCode
colon.      0  SQLSTATE: 00000
                                         1                0        6        0            0%        0%   0%   0%   0%   0%   0%
                   Section:    3  Stmtno:    0  Opcode:  50  Type: OPEN with Input host variables   RetCode
colon.      0  SQLSTATE: 00000
                                         1               16      215        0           77%        0%   0%   0%   0%  23%   0%
                   Section:    3  Stmtno:    0  Opcode:  30  Type: FETCH                            RetCode
colon.    100  SQLSTATE: 02000
                                         2                1        2        0          100%        0%   0%   0%   0%   0%   0%
                   Section:    4  Stmtno:    0  Opcode:  50  Type: OPEN with Input host variables   RetCode
colon.      0  SQLSTATE: 00000
                                         1                2        6        0           25%        0%   0%   0%   0%  75%   0%
                   Section:    4  Stmtno:    0  Opcode:  30  Type: FETCH                            RetCode
colon.      0  SQLSTATE: 00000
                                         1                0        0        0            0%        0%   0%   0%   0%   0%   0%
                   Section:    4  Stmtno:    0  Opcode:  45  Type: CLOSE call                       RetCode
colon.      0  SQLSTATE: 00000
                                         1                0        0        0            0%        0%   0%   0%   0%   0%   0%
                   Section:    5  Stmtno:    0  Opcode:  50  Type: OPEN with Input host variables   RetCode
colon.      0  SQLSTATE: 00000
                                         1                3        5        0            1%        0%   0%   0%   0%  99%   0%
                   Section:    5  Stmtno:    0  Opcode:  30  Type: FETCH                            RetCode
colon.    100  SQLSTATE: 02000
                                     28469            12622    57073        0           72%        0%   0%   0%   0%  28%   0%
                   Section:    6  Stmtno:    0  Opcode:  50  Type: OPEN with Input host variables   RetCode
colon.      0  SQLSTATE: 00000
                                         1                2        6        0            3%        0%   0%   0%   0%  97%   0%
                   Section:    6  Stmtno:    0  Opcode:  30  Type: FETCH                            RetCode
colon.      0  SQLSTATE: 00000
                                     11118            32772   153766        0           87%        0%   0%   0%   0%  13%   0%
01/29/98 15:24:45  LUW Ended:                Database: ELDB2A   at BOEVMCT1
                   Userid: ELRES5    Path:    32  Agent:     2    LUW Start: 01/29/98 15:23:00
                   SQL ID: ELRES5    LUW ID:    429B2
                   Number of Transactions: 05138  Longest Transaction: 000:00.273  Average Transaction: 000
colon.00.000
                   Response Time: 000:00.001
                   Agent VCPU:    45426  User VCPU:    30539  LPagBuf:   211099
                   <--- Dispatch / Wait --->
                   State    Count       Time
                   RUN:    108515 000:42.456          Page Reads:     3343  Writes:        0
                   COMM:   105137 000:53.412          DIR Reads:        27  Writes:        2
                   LOCK:        0 000:00.000          LOG Reads:         0  Writes:        1
                   CHKP:        0 000:00.000
                   OUTP:        0 000:00.000          Row Count:         1
                   OUTB:        0 000:00.000          Max Cost:      0.000
                   I/O:      3373 000:09.471          Max Locks:        29
                   DSPF:        0 000:00.000          Max LLocks:       27
                   Last Op:  30 Type: E (COMMIT WORK                                  )
                   RetCode:      0  SQLSTATE: 00000  Warning:
                   DBSS Opcode Count:    105220
                   RDIIN Opcode Summary:
                         45:         4  (CLOSE Call                                  )
                         50:         4  (OPEN Call                                   )
                   DBSS Table/Index Summary:   (10 entries)
                        Dbspaceno:      1  Table-id: -32740  Index-id:      0
                        Dbspaceno:     40  Table-id: -32767  Index-id:      0
                        Dbspaceno:      1  Table-id: -32760  Index-id: -32718
                        Dbspaceno:      1  Table-id: -32765  Index-id: -32761
                        Dbspaceno:     41  Table-id: -32767  Index-id:      0
                        Dbspaceno:      9  Table-id: -32767  Index-id: -32765
```

Figure 113. DBMAP ACTDETL Activity Report

ACTVTIME User Activity Across Time

This report lists the time users were active.

```
ACTVTIME      VM:DB/Monitor (c) 1998, Sterling Software, Inc.
                                                  User Activity Across Time
                                  First Time: 01/30/98 11:25:53     Last Time: 01/30/98 17:22:28
Database: ELDB2A   at BOEVMCT1   DBname: ELDB2A              DBresrce: ELDB2A    Initialized at: 01/30/98 11:45:13
Seq   Userid   SQLid    0   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19  20  21  22  23     Actv. Time
---- -------- -------- ||-+-|-+-|-+-|-+-|-+-|-+-|-+-|-+-|-+-|-+-|-+-|-+-|-+-|-+-|-+-|-+-|-+-|-+-|-+-|-+-|-+-|-+-|-+-|-+-| ----------
  1 ELRES3   ELRES3   |||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||>               *     |||||||||||||||||||||||||   0:00.001
  2 ELRES3   ELRES5   |||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||>                  **  |||||||||||||||||||||||||   0:00.614
  3 ELRES3   XTSOFT   |||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||>               *   *  |||||||||||||||||||||||||   0:00.193
  4 ELRES3   XTS4     |||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||>                  ***|||||||||||||||||||||||||   0:02.250
  5 ELRES5   ELRES5   |||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||*          **          |||||||||||||||||||||||||   0:54.658
                       ||-+-|-+-|-+-|-+-|-+-|-+-|-+-|-+-|-+-|-+-|-+-|-+-|-+-|-+-|-+-|-+-|-+-|-+-|-+-|-+-|-+-|-+-|-+-|-+-|-+-|-+-|
                        0   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19  20  21  22  23
Legend:  '*' -- User is active in database
         '>' -- Activity recording is started
         '<' -- Activity recording is stopped
```

Figure 114. DBMAP ACTVTIME Activity Report

DEADLOCK Deadlock Occurences

This report lists deadlocks.

```
DEADLOCK      VM:DB/Monitor (c) 1998, Sterling Software, Inc.                                                    Page     1
                                                  Deadlock Occurances
                                    First Time: 02/16/98 07:28:01     Last Time: 02/16/98 12:48:03
Database: FVVPDBA  at FVV        DBname: FVVPDB              DBresrce: FVVPDB    Initialized at: 02/16/98 07:59:30
Seq        Deadlock Time         Deadlock Information
---- --------------------------  ------------------------------------------------------------------------------------------
   1 02/16/98 10:04:46.243395    Losing  Userid: FVV209    SQLid: FVV       LUWid: 1CC8  LUW start time: 02/16/98 10:04:46.032205
                                         Package Author: FVV       Name: B212A     Section:    0
                                         Mode of lock held   on resource 1: S    Mode needed on resource 2: S
                                 Winning Userid: FVV251    SQLid: FVV       LUWid: 1CBE  LUW start time: 02/16/98 10:04:39.867607
                                         Package Author: FVV       Name: B912A     Section:    5
                                         Mode of lock needed on resource 1: X    Mode held   on resource 2: X
                                 Resource 1 Lock Type: IPAG  DBspace:    196  Qualifier:    0009E200
                                 Resource 2 Lock Type: IKEY  DBspace:    196  Qualifier:    000088OF
```

Figure 115. DBMAP DEADLOCK Activity Report

DYNSQL Summary Activity Report

This report lists resources used by each dynamic SQL statement prepared by a user.

```
DYNSQL       VM:DB/Monitor (c) 1998, Sterling Software, Inc.                                                          Page     1
                                                       Dynamic SQL Statements
                                   First Time: 01/30/98 08:59:59     Last Time: 01/30/98 17:00:0
Database: FVVPDBA  at FVV       DBname: FVVPDB             DBresrce: FVVPDB    Initialized at: 01/26/98 06
colon.14:47
                                                                                  Est.  # of                               SQL   SQL   LUW
Seq  Userid   SQLid    Author   Program  Sect Preparation Time      Elapsed  Cost Trans Agnt-CPU Buflooks # of Rows Code  State End
---- -------- -------- -------- -------- ---- --------------------- --------- ---- ----- -------- -------- --------- ----- ----- ---
   1 FVVWORK  DBINFO   SQLDBA   ARIDSQL     6 01/30/98 16:45:12.691     0.028    1     1      748     2050       291
0 00000 CMT
                       INSERT INTO DBINFO.CAT ( DATE           , TIME           , TNAME           , CREATOR        , ROWCOUN
                       T       , DBSPACENO      , DBSPACENAME    , REMARKS ) SELECT '30.01.1998'        , '16:45:03'
                        , TNAME          , CREATOR        , ROWCOUNT       , DBSPACENO      , DBSPACENAME    , REMARKS FROM
                        SYSTEM.SYSCATALOG WHERE CREATOR = 'FV' AND TABLETYPE = 'R' AND DBSPACENAME <> 'DSQTSDEF'

  85 FVV501   FVV501   Q        DSQ9FSQL    1 01/30/98 15:51:04.727     0.010    0     1       17       36         0  -5
1 42501 RLB
                       SELECT VERB,OBJECT,SYNONYM_DEFINITION FROM Q.COMMAND_SYNONYMS              ORDER BY 1, 2

  86 FVV501   FVV501   Q        DSQ9DYSQ    1 01/30/98 15:51:10.860     0.045    3     2       60      110         0
0 00000 CMT
                       SELECT OWNER, TNAME, TYPE, SUBTYPE, MODEL, RESTRICTED, CREATED, MODIFIED, LAST_USED, REMARKS, LABEL,
                        LOCATION, OWNER_AT_LOCATION, NAME_AT_LOCATION FROM Q.DSQEC_QMFOBJS

 101 FVV702   FVV702   SQLDBA   ARIISQL     4 01/30/98 10:45:44.323     0.215    1     4       60       92         2   1
0 02000 CMT
                       SELECT SEQNO,COMMAND FROM SQLDBA    .ROUTINE WHERE NAME='PROFILE '  ORDER BY SEQNO

 103 FVV702   FVV702   SQLDBA   ARIISQL     3 01/30/98 10:46:19.124     0.014    1     1      388      380         1
0 00000 CMT
                       UPDATE FV.SHUVERT SET SHUFLAGS = ' ' WHERE STAMMNR = 36507 AND SHUSPRT = 452 AND SHULFNR = 1

 106 FVV702   FVV702   Q        DSQ9ESQL    1 01/30/98 14:00:01.458     1.128    1     1      108      671       192
0 00000 CMT
                       UPDATE  FV.KTARF SET KTGAB = 19980101 WHERE KTARFA= 198 AND KDDECK = 'KH' AND KWAGKZ = 112

 193 FVV701   FVV      FVV      B752A       8 01/30/98 11:40:19.070     0.517    1     1       92      183         0
0 00000 CMT
                       SELECT   BKKREDID, BKKREDID, BKKREDID, UKID, BKAUSBL, BKLFDNR, BKBUART,  BKBLART, BKBLNR, BKBLDTM, B
                       KBUDTM, BKBUPER,  BKBUJAHR, BKKZKTO, BKHKID, BKUKID, BKANZKTO,  BKBUBTR, BKKZSH, BKBUTXT, BKAUSSP, B
                       KAUSKZ,  BKAUSDTM, BKAUSTM, BKERFSB, BKERFDTM, BKERFTM,  BKFAEDTM, BKDADTM, BKJDDTM, BKKDDTM, BKKVSI
                       D,  BKSPDTM FROM FV.BK T1 WHERE      BKKREDID BETWEEN ? and ?     AND ((BKBUPER  <= ? AND BKBUJAHR =
                        ?)    OR  BKBUJAHR <  ?)   AND (BKAUSDTM = 0 OR BKAUSDTM > ?) ORDER BY           BKKREDID, BKBUDTM
```

Figure 116. DBMAP DYNSQL Summary Activity Report

LUWSTATS Summary Activity Report

This report lists the count of each LUW opcode (RDIIN) executed by the database, and the top ten LUWs sorted by agent-CPU, DASD I/Os and elapsed time.

```
LUWSTATS      VM:DB/Monitor (c) 1998, Sterling Software, Inc.                                                    Page     1
                                              Logical Unit of Work Statistics
                                  First Time: 01/30/98 08:59:59     Last Time: 01/30/98 17:00:0
Database: FVVPDBA  at FVV        DBname: FVVPDB              DBresrce: FVVPDB    Initialized at: 01/26/98 06
colon.14:47
Opcode/Description          :  Count     Opcode/Description          :  Count     Opcode/Description          :  C
unt      Total
-----------------------------------     -----------------------------------     -----------------------------------     -------
( 30) AUX call              : 217525     (131) Create Program        :      0     (166) Modify Cancel         :
  0      456141
( 35) Setup (Prepare)       :    120     (132) Drop Statement        :      0     (170) Operator Cmd Continue :
  0
( 40) Describe              :    137     (135) Lookup                :  53086     (175) Open  to Un/Reload AUX:
  1
( 45) Close                 :  52654     (140) SQL call              :      0     (180) AUX call Un/Reload AUX:
  2
( 50) Open                  : 132565     (145) Prepfinish            :      0     (185) Close to Un/Reload AUX:
  1
(120) Connect/Schedule      :     50     (155) Operator Command      :      0     (190) Set/Change Language   :
  0
(125) Recovery List         :      0     (160) Prepare-to-Commit     :      0     (195) Start call            :
  0
(130) Prepinit              :      0     (165) Set/Reset Exit        :      0     (200) Reorganize call       :
  0
<-------------------- Top Ten LUWs By Agnt-CPU, Buflooks, and Elapsed --------------------->
Userid    SQLid       Starting Time of Day      Elapsed   Trans   Agnt-CPU  Buflooks  RowCount
--------  --------  ------------------------  ----------  -----  ---------  --------  --------
FVV990    FVV       01/30/98 14:33:39.322700    1:31.016   7245      17605*    19088      3289
FVV504    FVV       01/30/98 14:46:15.014280    1:04.540     47      10221*   115179        87
FVV990    FVV       01/30/98 14:32:45.634143    0:53.680   2063       3733*     7625         7
FVV491    FVV       01/30/98 15:57:17.028538    0:04.292      5       2461*    10553         1
FVV695    FVV695    01/30/98 10:34:54.027591    0:30.320     78       2290*     4140      2830

FVV504    FVV       01/30/98 14:46:15.014280    1:04.540     47      10221    115179*       87
FVV990    FVV       01/30/98 14:33:39.322700    1:31.016   7245      17605     19088*     3289
FVV491    FVV       01/30/98 15:57:17.028538    0:04.292      5       2461     10553*        1
FVV422    FVV       01/30/98 10:50:40.202585    0:04.718    207       1700      8509*      366
LA0500    FVV       01/30/98 10:54:02.810980    0:09.111      5       2057      8443*        1

FVV424    FVV       01/30/98 13:03:03.837415   38:34.122*     9         15        20         4
LA0100    FVV       01/30/98 09:32:40.824147   14:21.101*     9         19        34        11
FVV414    FVV       01/30/98 11:00:54.129648   14:01.774*     9         16        28         8
FVV501    FVV501    01/30/98 15:52:23.189537    9:23.922*    21        521       587       240
FVV422    FVV       01/30/98 15:03:37.732546    5:39.485*     9         17        24         6
```

Figure 117. DBMAP LUWSTATS Summary Activity Report

PACKAGED Summary Activity Report

This report lists the resources used by each package. The lines are grouped by program, author, VM user ID, SQLid.

```
PACKAGED      VM:DB/Monitor (c) 1998, Sterling Software, Inc.                                                    Page     1
                                      Package Usage and Resource Consumption - Detail - Part 1
                                   First Time: 01/29/98 09:51:03     Last Time: 01/29/98 10:04:
6
 Database: ELDB2A   at BOEVMCT1   DBname: ELDB2A              DBresrce: ELDB2A    Initialized at: 01/28/98 1
:41:03
                                             # of   # of   # of   Longest    Average                                  Row    # of    Ma
 Seq  Program  Author   Userid   SQLid       Runs  Trans   LUWs     LUW        LUW     Agnt-CPU User-CPU   LPAGB    Count  DynSQL  Cos
 ---- -------- -------- -------- --------  ------ ------ ------ ---------- ---------- -------- -------- -------- -------- ------ ----
    1 ARIDSQL  SQLDBA   ELRES5   ELRES5         2      4      2 000:00.625 000:00.326       65        2      362        2
     2     1
    2 ARIISQL  SQLDBA   ELRES5   ELRES5         1      8      2 000:00.068 000:00.035       10        2       53        0
     2     1
    3 CTPB110  ELRES5   ELRES3   XTS4           2     18      6 000:00.830 000:00.689       20        7      228       54
     0     0
    4 CTPB110  ELRES5   ELRES5   ELRES5         4  23902      4 000:13.579 000:11.244    12088     6812    37116        2
     0     0
    5 XTS4     XTS4     ELRES3   SQLDBA         1      8      8 000:00.001 000:00.001        6        0        0        0
     0     0
    6 XTS4     XTS4     ELRES3   XTSOFT         3     65     65 000:00.001 000:00.000       38        8        0        0
     0     0
    7 XTS4     XTS4     ELRES3   XTS4           2     14     14 000:00.830 000:00.296        8        6        0        0
     0     0
    8 XTS4     XTS4     ELRES5   XTSOFT         3     10     10 000:00.001 000:00.000        6        2        0        0
     0     0
    9 XTS4GPWD XTS4     ELRES3   ELRES3         3      3      3 000:00.001 000:00.001        6        0        9        0
     0     0
   10 XTS4GPWD XTS4     ELRES3   XTSOFT         3      6      3 000:00.002 000:00.002        9        2       21        0
     0     0
   11 XTS4GPWD XTS4     ELRES5   ELRES5         3      3      3 000:00.001 000:00.001        4        0        9        0
     0     0
   12 XTS4GPWD XTS4     ELRES5   XTSOFT         3      6      3 000:00.002 000:00.002        7        3       21        0
     0     0
   13 XTS4HOST SQLDBA   ELRES3   SQLDBA         1      2      2 000:00.020 000:00.017        2        0       34        0
     0     0
   14 XTS40005 XTS4     ELRES3   SQLDBA         1      4      4 000:00.001 000:00.002        2        1        6        0
     0     0
   15 XTS40005 XTS4     ELRES3   XTSOFT         2      4      4 000:00.001 000:00.007        3        1        6        0
     0     0
   16 XTS40005 XTS4     ELRES3   XTS4           2      9      9 000:00.001 000:00.001        8        0       12        0
     0     0
   17 XTS40025 XTS4     ELRES3   XTS4           1      2      2 000:00.003 000:00.003        4        0       14        0
     0     0
   18 XTS40129 XTS4     ELRES3   SQLDBA         1    105      9 000:00.138 000:00.057      220       39     2229       77
     29     1
   19 XTS40129 XTS4     ELRES3   XTS4           1      4      4 000:00.003 000:00.005        2        4        6        0
     0     0
PACKAGED      VM:DB/Monitor (c) 1998, Sterling Software, Inc.                                                    Page     2
                                      Package Usage and Resource Consumption - Detail - Part 2
                                   First Time: 01/29/98 09:51:03     Last Time: 01/29/98 10:04:
6
 Database: ELDB2A   at BOEVMCT1   DBname: ELDB2A              DBresrce: ELDB2A    Initialized at: 01/28/98 1
:41:03
                                             Total Time   <------- Dispatch / Wait State Times -------->
 Seq   Program  Author    Userid   SQLid        Active      Run  COMM  LOCK  CHKP  OUTP  OUTB   I/O  DSPF
 ----  -------- --------  -------- --------   -----------   ----  ----  ----  ----  ----  ----  ----  ----
    1  ARIDSQL  SQLDBA    ELRES5   ELRES5      000:00.649     9%    0%    0%    0%    0%    0%   89%    0%
    2  ARIISQL  SQLDBA    ELRES5   ELRES5      000:00.069    17%    2%    0%    0%    0%    0%   79%    0%
    3  CTPB110  ELRES5    ELRES3   XTS4        000:00.095    21%    7%    0%    0%    0%    0%   71%    0%
    4  CTPB110  ELRES5    ELRES5   ELRES5      000:44.947    26%   70%    0%    0%    0%    0%    2%    0%
    5  XTS4     XTS4      ELRES3   SQLDBA      000:00.004   100%    0%    0%    0%    0%    0%    0%    0%
    6  XTS4     XTS4      ELRES3   XTSOFT      000:00.013   100%    0%    0%    0%    0%    0%    0%    0%
    7  XTS4     XTS4      ELRES3   XTS4        000:04.032     0%   99%    0%    0%    0%    0%    0%    0%
    8  XTS4     XTS4      ELRES5   XTSOFT      000:00.002   100%    0%    0%    0%    0%    0%    0%    0%
    9  XTS4GPWD XTS4      ELRES3   ELRES3      000:00.002   100%    0%    0%    0%    0%    0%    0%    0%
   10  XTS4GPWD XTS4      ELRES3   XTSOFT      000:00.003   100%    0%    0%    0%    0%    0%    0%    0%
   11  XTS4GPWD XTS4      ELRES5   ELRES5      000:00.003   100%    0%    0%    0%    0%    0%    0%    0%
   12  XTS4GPWD XTS4      ELRES5   XTSOFT      000:00.003   100%    0%    0%    0%    0%    0%    0%    0%
   13  XTS4HOST SQLDBA    ELRES3   SQLDBA      000:00.002   100%    0%    0%    0%    0%    0%    0%    0%
   14  XTS40005 XTS4      ELRES3   SQLDBA      000:00.002   100%    0%    0%    0%    0%    0%    0%    0%
   15  XTS40005 XTS4      ELRES3   XTSOFT      000:00.002   100%    0%    0%    0%    0%    0%    0%    0%
   16  XTS40005 XTS4      ELRES3   XTS4        000:00.007   100%    0%    0%    0%    0%    0%    0%    0%
   17  XTS40025 XTS4      ELRES3   XTS4        000:00.002   100%    0%    0%    0%    0%    0%    0%    0%
   18  XTS40129 XTS4      ELRES3   SQLDBA      000:00.499    47%   10%    0%    0%    0%    0%   41%    0%
   19  XTS40129 XTS4      ELRES3   XTS4        000:00.004    50%   50%    0%    0%    0%    0%    0%    0%
```

Figure 118. DBMAP PACKAGED Summary Activity Report

PACKAGES Summary Activity Report

This report summarizes the resource used by each package. It's almost the same as the PACKAGED report but the report is grouped by package name and author.

```
PACKAGES      VM:DB/Monitor (c) 1998, Sterling Software, Inc.                                                    Page     1
                                  Package Usage and Resource Consumption - Summary - Part 1
                                First Time: 01/30/98 08:59:59     Last Time: 01/30/98 17:00:0
Database: FVVPDBA  at FVV        DBname: FVVPDB              DBresrce: FVVPDB    Initialized at: 01/26/98 06
colon.14:47
                           # of   # of    # of   # of   # of   Longest    Average                                  Row    # of    Max
Seq   Program  Author    Userids SQLids   Runs  Trans   LUWs     LUW        LUW    Agnt-CPU User-CPU   LPAGB    Count  DynSQL  Cost
----  -------- --------  ------- ------ ------ ------ ------ ---------- ---------- -------- -------- -------- -------- ------ -----
   1  YHNWZ    FVV            51      1     62  52985   3019 000:55.982 000:00.927    79380   126751   171601     1397
   0     0
   2  KV310    FVV            21      1     25  41636   1509 001:04.200 000:00.882   161343    82564   813086   342131
   0     0
   3  KV311    FVV            51      1     60  25677   2440 000:55.982 000:00.926    62892    95405   106967     7795
   0     0
   4  ZS11A    FVV            85      1    114  24943   6871 002:42.616 000:00.856    52739   115649    67765     7109
   0     0

PACKAGES      VM:DB/Monitor (c) 1998, Sterling Software, Inc.                                                    Page     2
                                  Package Usage and Resource Consumption - Summary - Part 2
                                First Time: 01/30/98 08:59:59     Last Time: 01/30/98 17:00:0
Database: FVVPDBA  at FVV        DBname: FVVPDB              DBresrce: FVVPDB    Initialized at: 01/26/98 06
colon.14:47
                          Total Time   <------- Dispatch / Wait State Times -------->
Seq   Program  Author       Active      Run  COMM  LOCK  CHKP  OUTP  OUTB   I/O  DSPF
----  -------- --------   -----------   ----  ----  ----  ----  ----  ----  ----  ----
   1  YHNWZ    FVV         010:33.105    44%   21%    0%    0%    0%    0%   34%    0%
   2  KV310    FVV         008:18.562    65%   21%    0%    0%    0%    0%   13%    0%
   3  KV311    FVV         010:58.173    30%   14%    0%    0%    0%    0%   54%    0%
   4  ZS11A    FVV         012:38.973    20%   59%    0%    0%    0%    0%   19%    0%
```

Figure 119. DBMAP PACKAGES Summary Activity Report

SECTIONS Summary Activity Report

This report describes the section usage and resource consumption of each package.

```
SECTIONS      VM:DB/Monitor (c) 1998, Sterling Software, Inc.                                                    Page     1
                                    Package Section Usage and Resource Consumption - Part 1
                                  First Time: 01/30/98 08:59:59     Last Time: 01/30/98 17:00:0
Database: FVVPDBA  at FVV        DBname: FVVPDB              DBresrce: FVVPDB    Initialized at: 01/26/98 06
colon.14:47
                                 # of   # of    # of   # of   # of   Longest    Average                        Row    # of    Max
Seq   Program  Author   Sect  Userids SQLids   Runs  Trans   LUWs     LUW        LUW    Agnt-CPU   LPAGB    Count  DynSQL  Cost
----  -------- -------- ----  ------- ------ ------ ------ ------ ---------- ---------- -------- -------- -------- ------ -----
   1  KV310    FVV         0       21      1     25  30399      0   1:04.200   0:00.000   137363   755347   342093
     0
   2  YTGOO    FVV         0       32      1     32   9495      0   1:01.594   0:00.000   118512   214686     2109
     0
   3  ZS11A    FVV         0       85      1    114  16730      0   0:55.982   0:00.000    38996    56943     7109
     0

SECTIONS      VM:DB/Monitor (c) 1998, Sterling Software, Inc.                                                    Page     2
                                    Package Section Usage and Resource Consumption - Part 2
                                  First Time: 01/30/98 08:59:59     Last Time: 01/30/98 17:00:0
Database: FVVPDBA  at FVV        DBname: FVVPDB              DBresrce: FVVPDB    Initialized at: 01/26/98 06
colon.14:47
                                Total Time   <---- Dispatch / Wait State Times ------>
Seq   Program  Author   Sect      Active      Run   LOCK  CHKP  OUTP  OUTB   I/O  DSPF
----  -------- -------- ----   -----------   ----   ----  ----  ----  ----  ----  ----
   1  KV310    FVV         0      4:39.442    84%     0%    0%    0%    0%   15%    0%
   2  YTGOO    FVV         0      4:02.226    72%     0%    0%    0%    0%   27%    0%
   3  ZS11A    FVV         0      3:47.667    45%     0%    0%    0%    0%   54%    0%
```

Figure 120. DBMAP SECTIONS Summary Activity Report

SECTSTAT Package Section Statistics

This report lists the package section statistics.

```
SECTSTAT      VM:DB/Monitor (c) 1998, Sterling Software, Inc.                                                    Page      1
                                                Package Section Statistics
                                    First Time: 01/30/98 11:25:53     Last Time: 01/30/98 17:22:28
Database: ELDB2A   at BOEVMCT1   DBname: ELDB2A              DBresrce: ELDB2A    Initialized at: 01/30/98 11:45:1
                                                          <----------------------- Average per Transaction ------------------------->
                                 Section         # of    Agent     Buffer     Row       Time    <---- Dispatch / Wait State Times ----->
Seq   Program  Author   Sect     Type           Trans CPU Time    Looks     Count     Active    Run  LOCK  CHKP  OUTP  OUTB   I/O  DSPF
----  -------- -------- ----     -------       ------ -------- -------- -------   -------- ----  ----  ----  ----  ----  ----  ----
  1   ARIDSQL  SQLDBA      1     Static            11     1.36     0.00     0.00      0.0010   9%    0%    0%    0%    0%   91%    0%
  2   ARIDSQL  SQLDBA      2     Static             3    11.67   310.00     0.00      0.0187  59%    0%    0%    0%    0%   41%    0%
  3   ARIDSQL  SQLDBA      3     Dynamic           15     1.73    13.00     0.20      0.0048  29%    0%    0%    0%    0%   71%    0%
  4   ARIDSQL  SQLDBA      6     Dynamic           92     6.91    75.98     0.00      0.0188  29%    0%    0%    0%    0%   71%    0%
  5   PBPB998  ELRES5      0     Static             3     1.67    13.67     3.67      0.0177   6%    0%    0%    0%    0%   94%    0%
  6   SQLRX    SQLREORG    2     Static            10     0.40     0.00     0.00      0.0001 100%    0%    0%    0%    0%    0%    0%
  7   SQLRX    SQLREORG    3     Dynamic           33     3.64    25.76     6.70      0.0049  74%    0%    0%    0%    0%   24%    2%
  8   XTS4     XTS4        1     Static            22     0.95     0.00     0.00      0.0001 100%    0%    0%    0%    0%    0%    0%
  9   XTS4GPWD XTS4        0     Static             1    30.00     3.00     0.00      0.0010 100%    0%    0%    0%    0%    0%    0%
 10   XTS4GPWD XTS4        1     Static             1    35.00    36.00     0.00      0.1920   3%    0%    0%    0%    0%   97%    0%
```

Figure 121. DBMAP SECTSTAT Activity Report

SESSIOND Summary Activity Report

This report lists the resources used by each user session. The information of this report comes from the Activity Session End and Activity Session Start records of the activity file. There is more information in these records, which you can get using the XDBMAP module to access the activity file records directly in a REXX program:

- Protocol used by the connection, DRDA or SQLDS.
- External SNA LUW ID for DRDA connections.
- External name of the job from where the connection is.
- Application requester type (SQLDS/VM, DB2, OS2).
- Requester's host node.
- Application requester id.
- Application requester version.

```
SESSIOND      VM:DB/Monitor (c) 1998, Sterling Software, Inc.                                                    Page     1
                                          Session Statistics - Detail - Part 1
                                   First Time: 01/30/98 08:59:59     Last Time: 01/30/98 17:00:0
Database: FVVPDBA  at FVV        DBname: FVVPDB              DBresrce: FVVPDB    Initialized at: 01/26/98 06
colon.14:47
                                     # of   Longest    Average    Average   Dispatch
Seq   Userid   Connect Time         LUWs     LUW        LUW    Resp. Time     Count----%  Agnt-CPU----%  User-CPU  BufLooks----%
----  -------- -----------------  ------ ---------- ---------- ----------  -------------  -------------  --------  -------------
   1  DBOO90   01/30/98 14:32:20      12 000:02.490 000:01.172 000:00.030       205   0%       296   0
      1153       392   0%
   2  FVVWORK  01/30/98 16:45:03       8 000:08.833 000:01.142 000:01.138       416 100%       881 100
       470      2561 100%
   3  FVV292   01/30/98 14:45:04       4 000:00.586 000:00.298 000:00.023        33   0%        52   0
      1778        61   0%
   4  FVV418   01/30/98 12:47:28     264 000:10.490 000:00.346 000:00.020      5412   3%      8528   2
     20154     13099   1%
   5  FVV425   01/30/98 09:49:56      23 000:02.920 000:00.703 000:00.042       636   3%      1279   2
      6485      3012   3%

SESSIOND      VM:DB/Monitor (c) 1998, Sterling Software, Inc.                                                    Page     2
                                          Session Statistics - Detail - Part 2
                                   First Time: 01/30/98 08:59:59     Last Time: 01/30/98 17:00:0
Database: FVVPDBA  at FVV        DBname: FVVPDB              DBresrce: FVVPDB    Initialized at: 01/26/98 06
colon.14:47
                                          Session States   Total Time <------ Dispatch / Wait State Times ------->
Seq   Userid   Connect Time         Elapsed  REAL WAIT IDLE     Active    Run COMM LOCK CHKP OUTP OUTB  I/O DSPF  NIW
----  -------- -----------------  ---------- ---- ---- ----  ----------- ---- ---- ---- ---- ---- ---- ---- ---- ----
   1  DBOO90   01/30/98 14:32:20  022:02.548   1%   0%  99%   000:14.062   8%  81%   0%   0%   0%   0%  10%
 0%   0%
   2  FVVWORK  01/30/98 16:45:03  000:09.896  92%   0%   8%   000:09.133  12%   0%   0%   0%   0%   0%  87%
 0%   0%
   3  FVV292   01/30/98 14:45:04  024:19.770   0%   0% 100%   000:01.191  12%  83%   0%   0%   0%   0%   5%
 0%   0%
   4  FVV418   01/30/98 12:47:28  144:50.379   1%   0%  99%   001:31.187  31%  36%   0%   0%   0%   0%  32%
 0%   0%
   5  FVV425   01/30/98 09:49:56  013:44.145   2%   0%  98%   000:15.915  42%  47%   0%   0%   0%   0%  11%
 0%   0%
```

Figure 122. DBMAP SESSIOND Summary Activity Report

- The percent in the Dispatch Count column represents the relation between the number of times the agent was dispatched for this session and the number of times the database was dispatched while this session was active.
- The percent in the Agnt-CPU column represents the relation between the agent virtual CPU consumption and the database virtual CPU consumption while this session was active.
- The percent in the Buflooks column represents the relation between the looks in the page buffers made by the agent and the total looks in the page buffers made by the database.

SESSIONS Summary Activity Report

This report lists resources used by each user for all complete sessions. It's almost the same as the SESSIOND report but the records are grouped by VM user ID.

```
SESSIONS      VM:DB/Monitor (c) 1998, Sterling Software, Inc.                                                    Page      1
                                          Session Statistics - Summary - Part 1
                                  First Time: 01/30/98 08:59:59     Last Time: 01/30/98 17:00:0
Database: FVVPDBA  at FVV        DBname: FVVPDB              DBresrce: FVVPDB    Initialized at: 01/26/98 06
colon.14:47
                  # of       # of   Longest    Average    Average   Dispatch
Seq   Userid    Sessions     LUWs     LUW        LUW    Resp. Time     Count----%  Agnt-CPU----%  User-CPU  BufLooks----%
----  --------  --------   ------ ---------- ---------- ----------  -------------  -------------  --------  -------------
   1  FVV990           4      206 001:31.016 000:02.365 000:00.032     14298  25%     29830  25%    120982     370
2  19%
   2  FVV418           1      264 000:10.490 000:00.346 000:00.020      5412   3%      8528   2%     20154     130
9   1%
   3  FVV699           1        7 000:00.424 000:00.224 000:00.042        74   0%       110   0%     -1813       1
3   0%
   4  FVV292           1        4 000:00.586 000:00.298 000:00.023        33   0%        52   0%     -1778
1   0%
   5  LAO430           1        7 000:09.000 000:02.229 000:00.020       361   0%      1528   1%     -1163      28
9   1%

SESSIONS      VM:DB/Monitor (c) 1998, Sterling Software, Inc.                                                    Page      2
                                          Session Statistics - Summary - Part 2
                                  First Time: 01/30/98 08:59:59     Last Time: 01/30/98 17:00:0
Database: FVVPDBA  at FVV        DBname: FVVPDB              DBresrce: FVVPDB    Initialized at: 01/26/98 06
colon.14:47
                    Total   Session States     Total Time <------ Dispatch / Wait State Times ------->
Seq   Userid        Elapsed  REAL WAIT IDLE       Active    Run COMM LOCK CHKP OUTP OUTB  I/O DSPF  NIW
----  --------   ----------- ---- ---- ----   ------------ ---- ---- ---- ---- ---- ---- ---- ---- ----
   1  FVV990      033:03.761  25%   0%  75%     008:07.225  17%  70%   0%   0%   0%   0%  13%   0%   0%
   2  FVV418      144:50.379   1%   0%  99%     001:31.187  31%  36%   0%   0%   0%   0%  32%   0%   0%
   3  FVV699      030:18.452   0%   0% 100%     000:01.567  38%  43%   0%   0%   0%   0%  18%   0%   0%
   4  FVV292      024:19.770   0%   0% 100%     000:01.191  12%  83%   0%   0%   0%   0%   5%   0%   0%
   5  LAO430      084:45.566   0%   0% 100%     000:15.554  27%  65%   0%   0%   0%   0%   9%   0%   0%
```

Figure 123. DBMAP SESSIONS Activity Report

TABNDXS Summary Activity Report

This report lists the number of LUWs that accessed each table and index pair.

```
TABNDXS       VM:DB/Monitor (c) 1998, Sterling Software, Inc.                                                    Page     1
                                                Table and Index Usage - Summary
                                      First Time: 01/30/98 08:59:59     Last Time: 01/30/98 17:00:0
Database: FVVPDBA  at FVV        DBname: FVVPDB              DBresrce: FVVPDB    Initialized at: 01/26/98 06
colon.14:47
                                          # of
Seq    Table-id  Dbspno  Index-id     LUWs
----   --------  ------  --------   -------
   1     -32767      26    -32765       455
   2     -32767      40    -32765       504
   3     -32767      40    -32764        20
   4     -32767      40    -32761      3587
   5     -32767      40    -32760       368
   6     -32767      40         0        73
```

Figure 124. DBMAP TABNDXS Summary Activity Report

TABNDXU Summary Activity Report

This report lists the number of LUWs that accessed each table and index pair by VM user ID and SQL connect-id.

```
TABNDXU       VM:DB/Monitor (c) 1998, Sterling Software, Inc.                                                    Page     1
                                                Table and Index Usage
                                      First Time: 01/29/98 09:51:03     Last Time: 01/29/98 10:04:
6
 Database: ELDB2A   at BOEVMCT1   DBname: ELDB2A              DBresrce: ELDB2A    Initialized at: 01/28/98 1
:41:03
                                                                    # of
 Seq   Table-id  Dbspno  Index-id    Userid    SQLid      LUWs
 ----  --------  ------  --------   --------  --------   ------
    1    -32767       9    -32765   ELRES5    ELRES5          4
    2    -32767      39    -32765   ELRES5    ELRES5          2
    3    -32767      40    -32764   ELRES5    ELRES5          2
    4    -32767      40         0   ELRES5    ELRES5          4
    5    -32767      41    -32765   ELRES5    ELRES5          2
    6    -32767      41         0   ELRES5    ELRES5          4
```

Figure 125. DBMAP TABNDXU Summary Activity Report

TABLES Auxiliary Activity Report

This is the same report as the TABNDXS summary activity report but the Table-id, Dbspno and Index-id columns are translated to their respective names. This report uses DBSU to search the catalog. To create the report you must call the TABLES command with the file name, type and mode of the activity file.

```
TABLES        VM:DB/Monitor (c) 1998, Sterling Software, Inc.                                    Page      1

                                        Table and Index Usage - Summary

                                First Time: 01/30/98 15:13:40     Last Time: 01/30/98 18:53:2

                                                                                                  # of
 Seq   Table Name          TCreator  Dbspno  Dbspacename         Index Name          ICreator      LUWs
 ----  ------------------  --------  ------  ------------------  ------------------  --------  --------
 1     ADRTBEW             FV           256  FV_PBL_021          ADRTBEWI01          FV               8
 2     ADRTBEW             FV           256  FV_PBL_021                                               8
 3     ADRTYP              FV           338  FV_PBL_004          ADRTYPI01           FV             129
 4     ADRTYP              FV           338  FV_PBL_004          ADRTYPI02           FV              36
 5     ADRTYP              FV           338  FV_PBL_004                                               3
 6     AIKDSD              FV           366  FV_PBL_051          AIKDSDI01           FV               7
 7     AIKDSD              FV           366  FV_PBL_051          AIKDSDI02           FV               2
 8     AIKDSD              FV           366  FV_PBL_051          AIKDSDI04           FV               6
 9     AIKDVT              FV           200  FV_PBL_050          AIKDVTI01           FV              30
 10    AIKDVT              FV           200  FV_PBL_050          AIKDVTI02           FV              47
 11    AIKDVT              FV           200  FV_PBL_050          AIKDVTI03           FV              45
 12    AIKDVT              FV           200  FV_PBL_050          AIKDVTI04           FV              20
 13    AIKDVT              FV           200  FV_PBL_050                                              28
```

Figure 126. DBMAP TABLES Summary Activity Report

USERRES Summary Activity Report

This report lists the resources used in the database by each VM user ID.

```
USERRES       VM:DB/Monitor (c) 1998, Sterling Software, Inc.                                    Page      1
                                          User Resource Consumption - Part 1
                                  First Time: 01/30/98 08:59:59     Last Time: 01/30/98 17:00:0
Database: FVVPDBA  at FVV        DBname: FVVPDB              DBresrce: FVVPDB    Initialized at: 01/26/98 06
colon.14:47
                           # of    # of   # of   Longest    Average                                  Row    # of    Max
Seq   Userid   SQLid       Pkgs   Trans   LUWs     LUW        LUW     Agnt-CPU  User-CPU BufLooks   Count  DynSQL  Cost
----  -------- --------   ------ ------ ------ ---------- ---------- --------- --------- -------- -------- ------ -----
   1  FVV424   FVV            71  31891   1348 038:34.122 000:02.483     82605    117719   205105    58253      0     0
   2  FVV415   FVV            61  28727   1491 001:19.167 000:00.581     79594    131250   197540    52857      1     1
   3  FVV422   FVV            77  27611   1522 005:39.485 000:01.353     69392    117739   169851    42691      0     0
   4  FVV441   FVV            64  23156   1087 002:33.169 000:00.807     62505     83802   133987    26235      0     0
   5  FVV434   FVV            59  23426   1235 005:28.611 000:01.090     55865     82686   113024    23788      0     0

USERRES       VM:DB/Monitor (c) 1998, Sterling Software, Inc.                                    Page      2
                                          User Resource Consumption - Part 2
                                  First Time: 01/30/98 08:59:59     Last Time: 01/30/98 17:00:0
Database: FVVPDBA  at FVV        DBname: FVVPDB              DBresrce: FVVPDB    Initialized at: 01/26/98 06
colon.14:47
                           Total Time   <------- Dispatch / Wait State Times -------->
Seq   Userid   SQLid         Active      Run  COMM  LOCK  CHKP  OUTP  OUTB   I/O  DSPF
----  -------- --------   -----------    ----  ----  ----  ----  ----  ----  ----  ----
   1  FVV424   FVV        0055:40.423     7%   88%    0%    0%    0%    0%    4%    0%
   2  FVV415   FVV        0013:55.092    27%   56%    0%    0%    0%    0%   15%    0%
   3  FVV422   FVV        0033:12.483    10%   81%    0%    0%    0%    0%    8%    0%
   4  FVV441   FVV        0014:33.242    19%   64%    0%    0%    0%    0%   15%    0%
   5  FVV434   FVV        0022:17.651    12%   78%    0%    0%    0%    0%    8%    0%
```

Figure 127. DBMAP USERRES Summary Activity Report

USERTIME Summary Activity Report

This report charts when each user was connected and active in the database.

```
USERTIME      VM:DB/Monitor (c) 1998, Sterling Software, Inc.                                                    Page     1
                                                  User Activity Across Time
                                    First Time: 01/30/98 08:59:59     Last Time: 01/30/98 17:00:0
Database: FVVPDBA  at FVV        DBname: FVVPDB              DBresrce: FVVPDB    Initialized at: 01/26/98 06
colon.14:47
                                                                                                                Connect    Time
Seq    Userid   0   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19  20  21  22  23     at Eof Connected
----  -------- ||-+-|-+-|-+-|-+-|-+-|-+-|-+-|-+-|-+-|-+-|-+-|-+-|-+-|-+-|-+-|-+-|-+-|-+-|-+-|-+-|-+-|-+-|-+-|-+-| ------- ---------
   1  FVV273   |>                                -***-********-*--*-*-*-**--|||||||||||||>|||||||||||||||||||||||            386&col
n.51
   2  FVV275   |>                                -**--***---*-----*---------|||||||||||||>|||||||||||||||||||||||            386&col
n.02
   3  FVV278   |>                                --**---****---------------|||||||||||||>|||||||||||||||||||||||            370&col
n.33
   4  FVV291   |>                                ----------------------------*--|||||||||>|||||||||||||||||||||||            447&col
n.38
   5  FVV427   |>                              ---************             |||||||||||||>|||||||||||||||||||||||            208&col
n.21
               ||-+-|-+-|-+-|-+-|-+-|-+-|-+-|-+-|-+-|-+-|-+-|-+-|-+-|-+-|-+-|-+-|-+-|-+-|-+-|-+-|-+-|-+-|-+-|-+-|-+-|-+-|
                0   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19  20  21  22  23
Legend:  '-' -- User is connected but not active
         '*' -- User is active
         '>' -- Activity recording is started
         '<' -- Activity recording is stopped
         '|' -- Time not in the file
```

Figure 128. DBMAP USERTIME Summary Activity Report

WAITLOCK Summary Activity Report

This report lists the time a user is put into LOCK wait.

```
WAITLOCK      VM:DB/Monitor (c) 1998, Sterling Software, Inc.                                                    Page     1
                                                  Waitlock Occurances
                                    First Time: 01/30/98 08:59:59     Last Time: 01/30/98 17:00:
0
 Database: FVVPDBA  at FVV        DBname: FVVPDB              DBresrce: FVVPDB    Initialized at: 01/26/98 0
:14:47
                                      <--- Waiting--->  <--- Request ---> <--- Holding ---> Hold  Lock             Lock        End
 Seq    Starting Time of Day   Elapsed  Userid   SQLid   State  Mode  Dur   Userid   SQLid   State Type Dbspno    Qualifier    State
 ---- ------------------------ ------- -------- -------- ------ ---- ------ -------- -------- ----- ---- ------ ---------------- -----
    1 01/30/98 11:01:50.307834    0:00 FVV415   FVV      G WAIT S    SHORT FVV428   FVV      I/O   IPAG    290
4C9CA00     GRANTD
    2 01/30/98 11:15:13.285119    0:01 FVV420   FVV      G WAIT X    LONG  FVV414   FVV      I/O   IKEY    290
FFB8002     GRANTD
    3 01/30/98 11:52:29.868894    0:00 FVV441   FVV      G WAIT X    LONG  FVV413   FVV      I/O   IKEY    290
FFB8002     GRANTD
    4 01/30/98 13:41:49.877886    0:00 FVV424   FVV      G WAIT X    LONG  FVV467   FVV      I/O   IKEY    290
FFB8002     GRANTD
    5 01/30/98 14:48:36.443200    0:00 FVV466   FVV      C WAIT X    SHORT FVV421   FVV      I/O   IPAG    343
01DD100     GRANTD
```

Figure 129. DBMAP WAITLOCK Summary Activity Report

B.3 DBMAP Monitor Reports

Monitor Log Reports Definitions

- # Obs: Number of monitor records in this time interval.
- Observations: Number of monitor records read.
- Intervals: time interval in minutes for each report line. Multiple log records are grouped into this interval, if necessary. It is specified when the report is requested in the REPORT record format of the monitor logging specification file.
- Response time: time from when the first request is received for the LUW.
- Combined I/O: DASD I/Os + DS Page Faults during the interval.

BUFFERS Summary Monitor Log Report

This report lists the count of DB2 page buffers by DBSPACE number at the end of a specified time interval. The most used DBSPACEs are the ones with the consistently larger counts.

```
BUFFERS       VM:DB/Monitor (c) 1998 Sterling Software, Inc.                                                    Page     1

                                                SQL/DS Page Buffer Utilization

 * * * Summary * * *              First Time: 01/30/98 08:28:00     Last Time: 01/30/98 16:26:4
          Intervals:        8
                                                                                                        Observations:   240

 Database: FVVPDBA  at FVV        DBname  : FVVPDB              DBresrce: FVVPDB

 Initialized at: 01/30/98 18:52:23
 Page Buffers defined:     4000                                          <== Value of NPAGBUF startup parameter

                 Intervals    Average    Minimum    Maximum   Std.Dev.
                 ---------   --------   --------   --------   --------
 # of Buffers                 3584.00       3299       3749     143.84    <== Number of page buffers in use

 DBspaceno                                                               <== DBspaceno associated with a buffer
        1                8      32.37          1         69      18.53    <== Number of buffers associated with the dbspaceno
        2                1       1.12          9          9       3.18    <== Number of buffers associated with the dbspaceno
        9                1       0.62          5          5       1.77    <== Number of buffers associated with the dbspaceno
       14                1       0.62          5          5       1.77    <== Number of buffers associated with the dbspaceno
       15                1       1.75         14         14       4.95    <== Number of buffers associated with the dbspaceno
       17                8      60.87         19        134      46.09    <== Number of buffers associated with the dbspaceno
       26                8      12.75          4         23       6.02    <== Number of buffers associated with the dbspaceno
    10091                1       0.25          2          2       0.71    <== Number of buffers associated with the dbspaceno
    10092                1       0.37          3          3       1.06    <== Number of buffers associated with the dbspaceno
    10093                1       0.25          2          2       0.71    <== Number of buffers associated with the dbspaceno
    10094                1       0.25          2          2       0.71    <== Number of buffers associated with the dbspaceno
    10095                1       0.25          2          2       0.71    <== Number of buffers associated with the dbspaceno
    32001                8      17.00          2         24       6.61    <== Number of buffers associated with the dbspaceno
```

Figure 130. DBMAP LOG BUFFERS Summary Monitor Report

COUNTERS Summary Monitor Log Report

This report identifies changes during the interval of each of the DB2 system counters.

```
COUNTERS      VM:DB/Monitor (c) 1998 Sterling Software, Inc.                                    Page     1

                                          SQL/DS Counters

                                First Time: 01/30/98 08:28:00     Last Time: 01/30/98 16:26:4

Database: FVVPDBA  at FVV        DBname  : FVVPDB              DBresrce: FVVPDB
Initialized at: 01/30/98 18:52:23

Time of last RESET *: 01/28/98 18:16:39

      RDSCALL   BEGINLUW            LOCKLMT   WAITLOCK  LPAGBUFF  PAGEREAD  DIRREAD   LOGREAD   DASDREAD  DSREAD              # OF
TIME  DBSSCALL  ROLLBACK  CHKPOINT  ESCALATE  DEADLOCK  LDIRBUFF  PAGWRITE  DIRWRITE  LOGWRITE  DASDWRIT  DSWRITE   TOTAL I/O  OBS.
----- --------  --------  --------  --------  --------  --------  --------  --------  --------  --------  --------  ---------  -----
09:00    68797      3901                 0         0    396039     12652      1273         0     13925         0
  30
       186059         2         1         0         0     32103      1801      2610       934      5345         0      19270

10:00    70970      4179                 0         0    413824     13832       213         0     14045         0
  30
       191351         0         0         0         0     26189      2108       907      1016      4031         0      18076

11:00    86689      4276                 0         2    555334     15207       193         0     15400         0
  30
       258073         4         0         0         0     31515      1910      1369      1062      4341         0      19741

12:00    64555      3435                 0         1    331985     10785       131         0     10916         0
  30
       153734         0         0         0         0     21657      1367       823       843      3033         0      13949

13:00    59970      3208                 0         0    397564      9395       159         0      9554         0
  30
       186436         1         0         0         0     21909      1709       932       780      3421         0      12975

14:00    54409      2916                 0         1    293378      8987       104         0      9091         0
  30
       135843         1         0         0         0     19042      1364       695       724      2783         0      11874

15:00    66238      3166                 0         1    453305     14834       230         0     15064         0
  30
       155450        11         0         0         0     29596      2341       882      1025      4248         0      19312

16:00    14114       661                 0         0    116910      4541      3116         0      7657         0
  30
        54188         1         1         0         0     29881      1008      3997       135      5140         0      12797

COUNTERS      VM:DB/Monitor (c) 1998 Sterling Software, Inc.                                    Page     2

                                          SQL/DS Counters

* * * Summary * * *              First Time: 01/30/98 08:28:00     Last Time: 01/30/98 16:26:4
          Intervals:        8
                                                                                  Observations:   24
```

Figure 131 (Part 1 of 2). DBMAP COUNTERS Summary Monitor Report

```
Database: FVVPDBA  at FVV        DBname  : FVVPDB              DBresrce: FVVPDB
Initialized at: 01/30/98 18:52:23

Time of last RESET *: 01/28/98 18:16:39

             Total Change   <-------------- Of the Interval Values ------------->
                 In Value       Average       Minimum       Maximum      Std.Dev.
             ------------   -----------   -----------   -----------   -----------
RDSCALL   -        485742     60717.75         14114         86689      21052.16   <== Calls to RDS
DBSSCALL  -       1321134    165141.75         54188        258073      58073.03   <== Calls to DBSS
BEGINLUW  -         25742      3217.75           661          4276       1145.72   <== LUW's started
ROLLBACK  -            20         2.50             0            11          3.66   <== LUW's rolled back
CHKPOINT  -             2         0.25             0             1          0.46   <== System checkpoints taken
LOCKLMT   -             0         0.00             0             0          0.00   <== Maximum locks exceeded
ESCALATE  -             0         0.00             0             0          0.00   <== Lock escalation
WAITLOCK  -             5         0.62             0             2          0.74   <== Waits for lock
DEADLOCK  -             0         0.00             0             0          0.00   <== Deadlocks detected
LPAGBUFF  -       2958339    369792.37        116910        555334     128739.51   <== Looks in page buffer
PAGEREAD  -         90233     11279.12          4541         15207       3603.92   <== DBspace page reads
PAGWRITE  -         13608      1701.00          1008          2341        436.61   <== DBspace page writes
LDIRBUFF  -        211892     26486.50         19042         32103       5039.97   <== Looks in directory buffer
DIRREAD   -          5419       677.37           104          3116       1058.84   <== Directory block reads
DIRWRITE  -         12215      1526.87           695          3997       1173.58   <== Directory block writes
LOGREAD   -             0         0.00             0             0          0.00   <== Log page reads
LOGWRITE  -          6519       814.87           135          1062        300.56   <== Log page writes
DASDREAD  -         95652     11956.50          7657         15400       3007.19   <== Total DASD reads
DASDWRIT  -         32342      4042.75          2783          5345        927.68   <== Total DASD writes
DSREAD    -             0         0.00             0             0          0.00   <== Total DataSpace reads
DSWRITE   -             0         0.00             0             0          0.00   <== Total DataSpace writes
TOTAL I/O -        127994     15999.25         11874         19741       3393.43   <== Total I/O
```

Figure 131 (Part 2 of 2). DBMAP COUNTERS Summary Monitor Report

DBSTATS Summary Monitor Log Report

This report includes statistics on the DB2 database.

```
DBSTATS       VM:DB/Monitor (c) 1998 Sterling Software, Inc.                                                    Page     1

                                                Database Statistics

                              First Time: 01/30/98 08:28:00     Last Time: 01/30/98 16:26:49

Database: FVVPDBA  at FVV        DBname: FVVPDB              DBresrce: FVVPDB    SQL/DS: 5.1    DCSS:
Initialized at: 01/30/98 18:52:23                                                CMS:     13   CP: ESA
2.0 XC

Page buffers defined :       4000
Dir  buffers defined :      15000
Max programs loaded  :        720
NCUSERS              :         12
NLRBS                :      35000
NLRBU                :       7300
Average Response Time: 000:00.131
HWM for pseudo-agents in use :      126 occurred at 01/26/98 11:38:35
HWM for pseudo-agents waiting:        3 occurred at 01/29/98 16:00:46
HWM for real-agents in use   :       12 occurred at 01/29/98 16:00:47
HWM for LRB's in use         :     7300 occurred at 01/29/98 22:41:22

HWM for users in lock wait   :        1 occurred at 01/30/98 15:08:50
                                <------------------------------- Average --------------------------------->
                                <----- Pseudo-Agents ---->   Real     LRB's   Page Buf   Dir Buf  Programs
Time   Dispatches  Connects  Severs  Available  In-Use  Waiting  Agents   in use    in use    in use    Loaded   # of Obs.
-----  ----------  --------  ------  ---------  ------  -------  ------  --------  --------  --------  --------  ---------
09:00       93315        44       8     114.23   81.77     0.00    0.73     11.27   3749.00  14887.00    431.00         30
10:00       93530         8       9      98.80   97.20     0.00    1.17     28.13   3669.00  14447.00    432.00         30
11:00      112247         5       4      98.63   97.37     0.00    1.30     34.00   3524.00  13468.00    432.00         30
12:00       82189         4       8     100.23   95.77     0.00    1.00     22.80   3684.00  14766.00    432.00         30
13:00       77199         3       7     104.03   91.97     0.00    1.20     19.23   3599.00  13538.00    432.00         30
14:00       69427         2       6     107.07   88.93     0.00    0.77     18.47   3661.00  13892.00    432.00         30
15:00       89029         8      55     124.23   71.77     0.00    0.83     82.47   3299.00  14523.00    432.00         30
16:00       27966         2      21     167.50   28.50     0.00    0.20     17.30   3487.00  10887.00    432.00         30
DBSTATS       VM:DB/Monitor (c) 1998 Sterling Software, Inc.                                                    Page     2

                                                Database Statistics

* * * Summary * * *          First Time: 01/30/98 08:28:00     Last Time: 01/30/98 16:26:49
              Intervals:        8
                                                                                                                Observations:
 240

Database: FVVPDBA  at FVV        DBname: FVVPDB              DBresrce: FVVPDB    SQL/DS: 5.1    DCSS:
Initialized at: 01/30/98 18:52:23                                                CMS:     13   CP: ESA
2.0 XC

Page buffers defined :       4000
Dir  buffers defined :      15000
Max programs loaded  :        720
NCUSERS              :         12
NLRBS                :      35000
NLRBU                :       7300
Average Response Time: 000:00.131
HWM for pseudo-agents in use :      126 occurred at 01/26/98 11:38:35    <== Number of users connected to the data
ase
HWM for pseudo-agents waiting:        3 occurred at 01/29/98 16:00:46    <== Waiting for a real agent
HWM for real-agents in use   :       12 occurred at 01/29/98 16:00:47    <== Maximum value is NCUSERS
HWM for LRB's in use         :     7300 occurred at 01/29/98 22:41:22    <== Maximum value is NLRBS
HWM for users in lock wait   :        1 occurred at 01/30/98 15:08:50    <== In lock wait at the same time

                              Total      Average    Minimum    Maximum   Std.Dev.
                            --------   ----------   --------   --------  --------
Dispatches              -    644902     80612.75      27966     112247  24815.61   <== Number of agent dispatches in the interval
Connects                -        76         9.50          2         44     14.14   <== Number of user connects during the interval
Severs                  -       118        14.75          4         55     17.05   <== Number of user severs during the interval
Pseudo-agents available -     (n/a)       114.34      98.63     167.50     23.21   <== Average number of available pseudo-agents
Pseudo-agents in-use    -     (n/a)        81.66      28.50      97.37     23.21   <== Average number of users connected
Pseudo-agents waiting   -     (n/a)         0.00       0.00       0.00      0.00   <== Average number waiting for a real agent
Real agents             -     (n/a)         0.90       0.20       1.30      0.35   <== Average number of real-agents in use
LRB's in use            -     (n/a)        29.21      11.27      82.47     22.62   <== Average number of lock request blocks
Page buf in use         -     (n/a)      3584.00    3299.00    3749.00    143.84   <== Average number of page buffers in use
Dir buf in use          -     (n/a)     13801.00   10887.00   14887.00   1294.44   <== Average number of directory buffers in use
Packages loaded         -     (n/a)       431.87     431.00     432.00      0.35   <== Average number of packages loaded
```

Figure 132. DBMAP DBSTATS Summary Monitor Report

DSCNTRS Summary Monitor Log Report

This report identifies changes during the interval of each of the DB2 Data Space counters.

```
DSCNTRS       VM:DB/Monitor (c) 1998, Sterling Software, Inc.                                                    Page     1
                                                  SQL/DS DataSpace Counters
                                    First Time: 02/26/98 08:25:05     Last Time: 02/26/98 16:23:24
Database: FVVPDBA  at FVV        DBname  : FVVPDB              DBresrce: FVVPDB
Initialized at: 02/26/98 06:22:34
Time of last RESET INTERNAL *:

      ESASEG    REQMAPPG  SAVESLD   SAVEBLK   REFBLOCK  REFBSPAN  REFLPAGE  DIAG10PG  SCREF     SCSPNREF  SAVECHK0   # OF
TIME  MAPSEGPG  MAPPG     SAVEMXRN  WAITSLD   REFBPAGE  REFLST    DIAG10    SCDIAG10  SCBLKREF  SAVEGNRL  SAVECHK1   OBS.
----- --------  --------  --------  --------  --------  --------  --------  --------  --------  --------  --------  -----
09:00       53       204        16       167       233     12336         0     23440        80       510         1     30
            71       204        28         0     24224         0      1436       130        80        15         0

10:00       12       172        27       270       473     35469         0     58193       160      1050         0     30
            21       168        49         0     40125         0      2476       220       160        27         0

11:00        8        96        16       160       539     55530         0     39151        70       540         0     30
            11        91        41         0     57530         0       990        70        70        16         0

12:00        6        99        13       130        78       742         0      5724        50       390         0     30
             7        92        27         0      2974         0       354        60        50        13         0

13:00        1        54        13       130       131       892         0      8840        90       660         0     30
             1        51        28         0      3136         0       463       100        90        13         0

14:00        0        55        15       150        40       260         0      5459        30       210         0     30
             0        48        27         0       960         0       352        60        30        15         0

15:00        0        66        13       130        60       460         0     13096        40       330         0     30
             0        61        25         0      1280         0       689        60        40        13         0

16:00        4       123        25       241       212      1463         0     48104       167      1135         1     30
             7       117        45         0      5666         0      2407       257       167        24         0

DSCNTRS       VM:DB/Monitor (c) 1998, Sterling Software, Inc.                                                    Page     2
                                                  SQL/DS DataSpace Counters
* * * Summary * * *              First Time: 02/26/98 08:25:05     Last Time: 02/26/98 16:23:24          Intervals:        8
                                                                                                        Observations:   240

Database: FVVPDBA  at FVV        DBname  : FVVPDB              DBresrce: FVVPDB
Initialized at: 02/26/98 06:22:34
Time of last RESET INTERNAL *:

              Total Change   <--------------- Of the Interval Values ------------->
                  In Value       Average       Minimum       Maximum      Std.Dev.
              ------------   -----------   -----------   -----------   -----------
ESASEG    -            84         10.50             0            53         17.68   <== Number of DataSpace segments mapped
MAPSEGPG  -           118         14.75             0            71         23.79   <== Calls to MAPMDISK DEFINE for ESASEG
REQMAPPG  -           869        108.62            54           204         55.09   <== Requests to remap a page
MAPPG     -           832        104.00            48           204         56.53   <== Calls to MAPMDISK DEFINE for REQMAPPG
SAVESLD   -           138         17.25            13            27          5.57   <== Calls to MAPMDISK SAVE to save a page
SAVEMXRN  -           270         33.75            25            49          9.60   <== Sum of Maximum Runs
SAVEBLK   -          1378        172.25           130           270         53.86   <== Count of blocks saved with MAPMDISK SAVE
WAITSLD   -             0          0.00             0             0          0.00   <== Count of waits for MAPMDISK SAVE completion
REFBLOCK  -          1766        220.75            40           539        189.77   <== Calls to block form of REFPAGE
REFBPAGE  -        135895      16986.87           960         57530      21552.03   <== Count of pages in block form REFPAGEs
REFBSPAN  -             0          0.00             0             0          0.00   <== Sum of spans used in block form REFPAGEs
REFLST    -             0          0.00             0             0          0.00   <== Calls to list form of REFPAGE
REFLPAGE  -        107152      13394.00           260         55530      20916.09   <== Count of pages in list form REFPAGEs
DIAG10    -          9167       1145.87           352          2476        878.35   <== Count of Diagnose x'10's executed
DIAG10PG  -        202007      25250.87          5459         58193      20681.31   <== Number of pages released with DIAG x'10's
SCDIAG10  -           957        119.62            60           257         77.98   <== Number of blocks released with DIAG x'10's
SCREF     -           687         85.87            30           167         51.95   <== # of blocks in REFPAGEs after MAPMDISK SAVE
SCBLKREF  -           687         85.87             0           167         51.95   <== Calls to block REFPAGEs after MAPMDISK SAVE
SCSPNREF  -          4825        603.12             0          1135        332.32   <== Sum of spans used in above REFPAGEs
SAVEGNRL  -           136         17.00            13            27          5.42   <== Save requests by general agents
SAVECHK0  -             2          0.25             0             1          0.46   <== Save Req by CHKP agent before checkpoint
SAVECHK1  -             0          0.00             0             0          0.00   <== Save Req by CHKP agent during checkpoint
```

Figure 133. DBMAP DSCNTRS Summary Monitor Report

EXTENTS Summary Monitor Log Report

This report lists the number of READs and WRITEs during the interval to each minidisk or dbextent.

```
DASDIO        VM:DB/Monitor (c) 1998 Sterling Software, Inc.                                                    Page     1
                                                    I/O Activity By DBEXTENT
* * * Summary * * *          First Time: 01/30/98 08:28:00     Last Time: 01/30/98 16:26:49
                Intervals:        8
                                                                                                          Observations:
  240
Database: FVVPDBA  at FVV        DBname: FVVPDB               DBresrce: FVVPDB
Initialized at: 01/30/98 18:52:23
# of DBextents:      137                                                <== Number of dbextents (minidisks)
                        Total       Average       Minimum       Maximum      Std.Dev.
                    -----------   -----------   -----------   -----------   -----------
# of Reads       -        95652      11956.50          7657         15400       3007.19   <== All DBextents for an interval
# of Writes      -        20137       2517.12          1419          4736       1285.55   <== All DBextents for an interval
                                <------------------------- Of the Interval Values ---------------------------->
                        <---- Total ---->   <--- Average --->   <--- Minimum --->   <--- Maximum --->   <-- Std. Dev. -->
Vaddr   DDname   Pool      Reads   Writes      Reads   Writes      Reads   Writes      Reads   Writes      Reads   Writes
-----  --------  ----    -------- --------   -------- --------   -------- --------   -------- --------   -------- --------
 0200  BDISK      N/A -      5419    12215     677.37  1526.87        104      695        3116     3997    1058.84  1173.58
 0201  LOGDSK1    N/A -         0     6519       0.00   814.87          0      135           0     1062       0.00   300.56
 0210  DDSK1        1 -      1672        0     209.00     0.00         86        0         480        0     128.67     0.00
 0211  DDSK108      1 -        28        0       3.50     0.00          1        0           9        0       2.93     0.00
 0220  DDSK2        2 -         0        0       0.00     0.00          0        0           0        0       0.00     0.00
 0221  DDSK92       2 -         0        0       0.00     0.00          0        0           0        0       0.00     0.00
 0222  DDSK93       2 -         0        0       0.00     0.00          0        0           0        0       0.00     0.00
 0223  DDSK94       2 -         0        0       0.00     0.00          0        0           0        0       0.00     0.00
 0224  DDSK95       2 -         0        0       0.00     0.00          0        0           0        0       0.00     0.00
 0230  DDSK3        3 -      1372        0     171.50     0.00        124        0         274        0      54.07     0.00
 0231  DDSK58       3 -         0        0       0.00     0.00          0        0           0        0       0.00     0.00
 0240  DDSK4        4 -       941        4     117.62     0.50         15        0         194        2      52.98     0.93
 0241  DDSK96       4 -         0        0       0.00     0.00          0        0           0        0       0.00     0.00
 0242  DDSK97       4 -         0        0       0.00     0.00          0        0           0        0       0.00     0.00
```

Figure 134. DBMAP EXTENTS Summary Monitor Report

HISTORY Monitor Log Report

This function compresses the log data and keeps it across long periods of time for additional historical reporting. This report only produces an output **file**; no printable output is produced.

LOGDETL Summary Monitor Log Report

This report lists each record of the log or history file.

```
01/30/98 00:25:15  Database: FVVPDBA  at FVV      DBname: FVVPDB             DBresid: FVVPDB
                   Initialized at: 01/26/98 06:14:47  Timezone=  1    DBmode=L                        DBMON= 4.00
                   SQL/DS=5.1    DCSS=Y  CMS= 13  CPversion=ESA      CPlevel=2.0 CPmach=XC   Virtual Storage=     80M
                   MAXCONN=  333  NCUSERS=   12    Max Pkgs=  720  NPACKAGE=   60  NPACKPCT=60%
                   DB Extents=  137  Storage Pools=   17    NPAGBUF= 4000  NDIRBUF=15000    NLRBS=35000  NLRBU= 7300
                   Log size=   306774016  Log mode=L  SLOGCUSH=95%  CHKINTVL=    2500
================================================================================================================================
01/30/98 09:26:09  Database: FVVPDBA  at FVV
                   First monitor record issued at=01/30/98 08:28:00   Last monitor record started at=01/30/98 09:2
:09
                   Number of monitor records in this log record=      30
                                CumValue  Interval      Virtual Storage Used= 60%
                   Total CPU    =66784.523   293.548     Overhead Total=    0.000   0.00%
                   Virtual CPU  =52494.570   198.046     Overhead Intvl=    0.000   0.00%
                   Page Reads   =    57440      2249
                   Page Writes  =    42455      1067     Working Set=    7774
                   Dispatches   = 15366761     93315     Resident   =    7827
                   User Connects=     1031        44     XStore     =       0
                   User Severs  =      933         8     TARGETWS   =      32
                   Run Agent    =    0         Average Response Time=000:00.138
                   Pseudo-agents available= 114.23
                   Pseudo-agents in use   =  81.77  High watermark=   126  occurred at 01/26/98 11:38:35
                   Pseudo-agents waiting  =   0.00  High watermark=     3  occurred at 01/29/98 16:00:46
                   Real agents in use     =   0.73  High watermark=    12  occurred at 01/29/98 16:00:47
                   LRB's in use           =  11.27  High watermark=  7300  occurred at 01/29/98 22:41:22
                   Users in lock wait high watermark              =     1  occurred at 01/30/98 08:59:38
                   Log used=      763266    Log pages remaining before next checkpoint=    2444
                   Number of packages loaded=   431
                   Page Buffers in use      =  3749    Dir Buffers in use= 14887
                   SQL/DS Counters:     (Time of last RESET *: 01/28/98 18:16:39)
                              CumValue    Interval     Occurred At                    CumValue    Interval
                   RDSCALL =   2227973       68797                          PAGWRITE=   12001113        1801
                   DBSSCALL=  28994206      186059                          LDIRBUFF=  23423304       32103
                   BEGINLUW=    139552        3901                          DIRREAD =    100935        1273
                   ROLLBACK=       603           2                          DIRWRITE=     99370        2610
                   CHKPOINT=        34           1 (01/30/98 09:00:03)      LOGREAD =     17667           0
                   LOCKLMT =         0           0 (                 )      LOGWRITE=     43882         934
                   ESCALATE=        39           0 (01/29/98 22:41:22)      DSREAD  =         0           0
                   WAITLOCK=        18           0 (01/29/98 14:58:56)      DSWRITE =         0           0
                   DEADLOCK=         1           0 (01/29/98 14:58:56)      DASDREAD=   3781728       13925
                   LPAGBUFF=  52771841      396039                          DASDWRIT=   1343365        5345
                   PAGEREAD=   3663126       12652                          TOTAL IO=   5125093       19270
                   VMDSS Internal Counters:   (Time of last RESET INTERNAL *:                 )
                              CumValue    Interval                                CumValue    Interval
                   ESASEG  =         0           0                          REFLST  =         0           0
                   MAPSEGPG=         0           0                          REFLPAGE=        0           0
                   REQMAPPG=         0           0                          DIAG10  =         0           0
                   MAPPG   =         0           0                          DIAG10PG=         0           0
                   SAVESLD =         0           0                          SCDIAG1O=         0           0
                   SAVEMXRN=         0           0                          SCREF   =         0           0
                   SAVEBLK =         0           0                          SCBLKREF=         0           0
                   WAITSLD =         0           0                          SCSPNREF=         0           0
                   REFBLOCK=         0           0                          SAVEGNRL=         0           0
                   REFBPAGE=         0           0                          SAVECHKO=         0           0
                   REFBSPAN=         0           0                          SAVECHK1=         0           0
                   Programs Loaded Into Storage:
                   Author   Program  Active   Author   Program  Active   Author   Program  Active   Author   Program  Active
                   FVV      KVOOO       N     FVV      ZS11A       N     FVV      KV311       N     FVV      YHNWZ       N
                   SQLDBA   ELORXSQL    N     SQLDBA   ARIDSQL     N     VMQMAINT VMVOO1O     N     FVV      FS62A       N
                   DMSPIPE  DMSPQI      N     FVV      C131A       N     FVV      SPO11       N     FVV      SPO15       N
                   Q        DSQ9NSQL    N     FVV      KV312       N     FVV      SFO24       N     FVV      SFO25       N
                   SQLDBA   ARIISQL     N     Q        DSQ9FSQL    N     SQLDBA   KSTSUBS     N
                   Count of Page Buffers by DBspaceno:
                   DBspaceno  Count     DBspaceno  Count     DBspaceno  Count     DBspaceno  Count     DBspaceno  Count
                        168     231           60      10           78      40          203      66           40     118
                        290     286          236      32            1      69          363     136          259      74
                        275     126          291     100           17      19          276     162          296     260
                      32001      24          154      29          217      11          160      12          161       4
                   I/O's by DBextent:                  <-- Interval ->                                    <-- Interval ->
                   Vaddr DDname   Pool   Reads  Writes   Reads  Writes    Vaddr DDname    Pool   Reads  Writes   Reads  Writes
                    02OO BDISK     N/A  461603  257472    1273    2610     0210 DDSK1         1  119812   21951     241       0
                    0220 DDSK2       2       0       0       0       0     0230 DDSK3         3   15937       0     179       0
                    0240 DDSK4       4   13096      10     131       2     0250 DDSK5         5   60545     617     148      57
                    0270 DDSK6       6   52935     965     348     346     02AO DDSK7         7   77189    1056     126      32
                    02BO DDSK8       8   70027     524     215      95     0251 DDSK9         5   66414     220     160       0
                    0271 DDSK10      6   41209     178      37       0     02CO DDSK11        9    7956     628      83      71
                    02DO DDSK12     10   80582     549      87     152     0300 DDSK13       11   11072     328      54       0
                    0252 DDSK14      5   64974     223     150       0     0310 DDSK15       12   16106     110     142      16
                    02O1 LOGDSK1   N/A      32  118947       0     934
```

Figure 135. DBMAP LOGDETL Summary Monitor Report

PERFORM Summary Monitor Log Report

This report includes agent counts and performance ratios based on formulas using the DB2 counters.

```
PERFORM       VM:DB/Monitor (c) 1998 Sterling Software, Inc.                                                    Page     1

                                                Performance Summary

                                    First Time: 01/30/98 08:28:00     Last Time: 01/30/98 16:26:49

Database: FVVPDBA  at FVV        DBname  : FVVPDB               DBresrce: FVVPDB
Initialized at: 01/30/98 18:52:23

HWM for LRB's in use         :     7300 occurred at 01/29/98 22:41:22
HWM for users in lock wait   :        1 occurred at 01/30/98 15:08:50

      <--- SQL/DS Activity ----> < Lock Activity >  <- Buffer Pools -->  <---------------- I/O Activity ---------------->
        LUW   ChkPoint DBSS Call  Waitlock DeadLock  Page Buff  Dir Buff   DASDIO  DataSpace DataSpace   VMDSS      Read    # of
Time    Load     Load    Eff. Use   Perform  Perform   Eff. Use  Eff. Use    Load      Load    Eff. Use  Eff. Use    Only    Obs.
----- ------- -------- ---------  -------- --------  --------- ---------  -------- --------- --------- ---------  -------  -----
09:00  67.085    0.017    63.02%     0.00%    0.00%     96.81%    96.03%   314.153       N/A      N/A%      N/A%   72.26%
0
10:00  71.944    0.000    62.91%     0.00%    0.00%     96.66%    99.19%   274.897       N/A      N/A%      N/A%   77.70%
0
11:00  73.608    0.000    66.41%     0.00%    0.00%     97.26%    99.39%   306.946       N/A      N/A%      N/A%   78.01%
0
12:00  59.111    0.000    58.01%     0.00%    0.00%     96.75%    99.40%   216.517       N/A      N/A%      N/A%   78.26%
0
13:00  55.221    0.000    67.83%     0.00%    0.00%     97.64%    99.27%   193.929       N/A      N/A%      N/A%   73.63%
0
14:00  50.197    0.000    59.95%     0.00%    0.00%     96.94%    99.45%   180.922       N/A      N/A%      N/A%   76.56%
0
15:00  54.507    0.000    57.39%     0.00%    0.00%     96.73%    99.22%   292.180       N/A      N/A%      N/A%   78.00%
0
16:00  11.381    0.017    73.95%     0.00%    0.00%     96.12%    89.57%   213.387       N/A      N/A%      N/A%   59.83%
0
PERFORM       VM:DB/Monitor (c) 1998 Sterling Software, Inc.                                                    Page     2

                                                Performance Summary

* * * Summary * * *              First Time: 01/30/98 08:28:00     Last Time: 01/30/98 16:26:49
               Intervals:        8
                                                                                                          Observations:
 240

Database: FVVPDBA  at FVV        DBname  : FVVPDB               DBresrce: FVVPDB
Initialized at: 01/30/98 18:52:23

HWM for LRB's in use         :     7300 occurred at 01/29/98 22:41:22    <== Maximum value is NLRBS
HWM for users in lock wait   :        1 occurred at 01/30/98 15:08:50    <== In lock wait at the same time

                            Average    Minimum    Maximum   Std.Dev.
                           --------   --------   --------   --------
<-- SQL/DS Activity -->
LUW Load                -      55.38      11.38      73.61      19.72    <== BEGINLUW/minutes
ChkPoint Load           -       0.00       0.00       0.02       0.01    <== CHKPOINT/minutes
DBSS Call Effective Use -     63.68%     57.39%     73.95%      5.57%    <== (1-(RDSCALL/DBSSCALL)) * 100

<--- Lock Activity --->
Waitlock Performance    -      0.00%      0.00%      0.00%      0.00%    <== (WAITLOCK/RDSCALL ) * 100
Deadlock Performance    -      0.00%      0.00%      0.00%      0.00%    <== (DEADLOCK/BEGINLUW) * 100

<--- Buffer Pools ---->
Page Buff Effective Use -     96.86%     96.12%     97.64%      0.45%    <== (1-(PAGEREAD/LPAGBUFF)) * 100
Dir  Buff Effective Use -     97.69%     89.57%     99.45%      3.48%    <== (1-(DIRREAD /LDIRBUFF)) * 100

<--- I/O Activity ---->
DASDIO Load             -     249.12     180.92     314.15      53.62    <== *BLOCKIO/minutes
DataSpace Load          -        N/A        N/A        N/A        N/A    <== (DSREAD+DSWRITE)/minutes
DataSpace Effective Use -       N/A%       N/A%       N/A%       N/A%    <== (1-(DSFAULT/DSREAD  )) * 100
VMDSS Effective Use     -       N/A%       N/A%       N/A%       N/A%    <== (1-(DSFAULT/LBUFLOOK)) * 100
Read Only               -     74.28%     59.83%     78.26%      6.25%    <== ((DASDREAD+DSREAD)/ Combined I/O) * 100
```

Figure 136. DBMAP PERFORM Summary Monitor Report

POOLS Summary Monitor Log Report

This report lists the number of READs and WRITEs during the interval to each storage pool.

```
POOLS          VM:DB/Monitor (c) 1998, Sterling Software, Inc.                                  Page      1
                                                I/O Activity By Storage Pool
                              First Time: 02/26/98 08:25:05     Last Time: 02/26/98 16:23:24
Database: FVVPDBA  at FVV        DBname: FVVPDB               DBresrce: FVVPDB
Initialized at:     02/26/98 06:22:34
# of Storage Pools:       17
      # of            # of     # of     # of
Time  Obs. BufLooks   Reads    Writes   I/Os     Pool  I/O type   Extents    BufLooks   Reads   Writes   I/Os
----- ---- -------- -------- -------- --------   ----  --------   -------    -------- -------- -------- --------
09:00   30   415698    39076     7623    32523    INT  DSO SEQ          0       15582        0        0        0
                                                DIR  BLK SEQ          1       27101    12316     6018    18334
                                                  1  DS5 STR          2        9519     1145        1      288
                                                  2  BLK SEQ          5           0        0        0        0
                                                  3  DS3 STR          2       69624      477        0      426
                                                  4  DS3 STR          3       28259      151        3      139

10:00   30   764658    98043     4753    67020    INT  DSO SEQ          0       97102        0        0        0
                                                DIR  BLK SEQ          1       47654     8941     2086    11027
                                                  1  DS5 STR          2       16556     1410        0      427
                                                  2  BLK SEQ          5           0        0        0        0
                                                  3  DS3 STR          2       85210      387        0      319
                                                  4  DS3 STR          3       36900      350        4      304

11:00   30   795701    58036     3538    26900    INT  DSO SEQ          0       88010        0        0        0
                                                DIR  BLK SEQ          1       29396     6263     1966     8229
                                                  1  DS5 STR          2       18021      897        0      331
                                                  2  BLK SEQ          5           0        0        0        0
                                                  3  DS3 STR          2      133636      190        0      123
                                                  4  DS3 STR          3       54283      202        0      114

12:00   30   428261    14734     3215    15844    INT  DSO SEQ          0        8224        0        0        0
                                                DIR  BLK SEQ          1       19710     3859     1336     5195
                                                  1  DS5 STR          2       12758      366        0      207
                                                  2  BLK SEQ          5           0        0        0        0
                                                  3  DS3 STR          2       48127       63        0       51
                                                  4  DS3 STR          3       20188       75        0       62

13:00   30   242444     9325     2081    10273    INT  DSO SEQ          0        7364        0        0        0
                                                DIR  BLK SEQ          1       12298     2262     1029     3291
                                                  1  DS5 STR          2        6701      166        0      139
                                                  2  BLK SEQ          5           0        0        0        0
                                                  3  DS3 STR          2       34563       47        0       39
                                                  4  DS3 STR          3       14890       43        3       45

14:00   30   373453    15915     2848    17071    INT  DSO SEQ          0       12843        0        0        0
                                                DIR  BLK SEQ          1       18386     3480     1327     4807
                                                  1  DS5 STR          2       10256      319        0      215
                                                  2  BLK SEQ          5           0        0        0        0
                                                  3  DS3 STR          2       70874      158        0      151
                                                  4  DS3 STR          3       28265      108        0       87

15:00   30   370937    22121     2992    23167    INT  DSO SEQ          0        9175        0        0        0
                                                DIR  BLK SEQ          1       22733     3307     1389     4696
                                                  1  DS5 STR          2       10041      297        0      114
                                                  2  BLK SEQ          5           0        0        0        0
                                                  3  DS3 STR          2       52035      147        0       92
                                                  4  DS3 STR          3       21287      142        0       94

16:00   30   543238    37932    17848    54131    INT  DSO SEQ          0       17364        0        0        0
                                                DIR  BLK SEQ          1       53575    19585    15935    35520
                                                  1  DS5 STR          2        7891      285        0      110
                                                  2  BLK SEQ          5           0        0        0        0
                                                  3  DS3 STR          2       59028      103        0       82
                                                  4  DS3 STR          3       24147       80        3       80
```

Figure 137 (Part 1 of 2). DBMAP POOLS Summary Monitor Report

```
POOLS         VM:DB/Monitor (c) 1998, Sterling Software, Inc.                                                    Page     2
                                                  I/O Activity By Storage Pool
* * * Summary * * *          First Time: 02/26/98 08:25:05     Last Time: 02/26/98 16:23:24                  Intervals:       8
                                                                                                             Observations:   240
Database: FVVPDBA  at FVV        DBname: FVVPDB               DBresrce: FVVPDB
Initialized at:     02/26/98 06:22:34
# of Storage Pools:       17
                          Total       Average       Minimum       Maximum      Std.Dev.
                       -----------   -----------   -----------   -----------   -----------
# of Buffer Looks -      3934390     491798.75        242444        795701     196423.08   <== All Storage Pools for an Interval
# of Reads        -       295182      36897.75          9325         98043      29523.59   <== All Storage Pools for an Interval
# of Writes       -        44898       5612.25          2081         17848       5230.69   <== All Storage Pools for an Interval
# of I/Os         -       246929      30866.12         10273         67020      19878.76   <== All Storage Pools for an Interval
                        <------------- Of the Interval Values -------------->
Pool  I/O Type               Total     Average    Minimum    Maximum   Std. Dev.
----  --------              ---------  ---------  ---------  ---------  ---------
 INT  DSO SEQ  -  BufLooks:    255664   31958.00       7364      97102   37643.71
                  Reads   :         0       0.00          0          0       0.00
                  Writes  :         0       0.00          0          0       0.00
                  I/Os    :         0       0.00          0          0       0.00
 DIR  BLK SEQ  -  BufLooks:    230853   28856.62      12298      53575   14504.65
                  Reads   :     60013    7501.62       2262      19585    5943.58
                  Writes  :     31086    3885.75       1029      15935    5128.33
                  I/Os    :     91099   11387.37       3291      35520   10906.86
   1  DS5 STR  -  BufLooks:     91743   11467.87       6701      18021    4023.44
                  Reads   :      4885     610.62        166       1410     471.10
                  Writes  :         1       0.12          0          1       0.35
                  I/Os    :      1831     228.87        110        427     112.92
   2  BLK SEQ  -  BufLooks:         0       0.00          0          0       0.00
                  Reads   :         0       0.00          0          0       0.00
                  Writes  :         0       0.00          0          0       0.00
                  I/Os    :         0       0.00          0          0       0.00
   3  DS3 STR  -  BufLooks:    553097   69137.12      34563     133636   30362.50
                  Reads   :      1572     196.50         47        477     154.85
                  Writes  :         0       0.00          0          0       0.00
                  I/Os    :      1283     160.37         39        426     138.73
   4  DS3 STR  -  BufLooks:    228219   28527.37      14890      54283   12309.86
                  Reads   :      1151     143.87         43        350      97.25
                  Writes  :        13       1.62          0          4       1.77
                  I/Os    :       925     115.62         45        304      81.46
```

Figure 137 (Part 2 of 2). DBMAP POOLS Summary Monitor Report

SQLLOG Summary Monitor Log Report

This report includes agent counts and performance ratios based on formulas using the DB2 counters.

```
SQLLOG        VM:DB/Monitor (c) 1998 Sterling Software, Inc.                                                    Page     1

                                SQL/DS Log Usage

        First Time: 01/30/98 08:28:00     Last Time: 01/30/98 16:26:49

Database: FVVPDBA  at FVV        DBname  : FVVPDB               DBresrce: FVVPDB
Initialized at: 01/30/98 18:52:23

Log Size           :    306774016
Checkpoint Interval:         2500

       <------- At End of Interval -------->  Checkpoints Taken
Time     Log Used    % Full  Pages Remaining    During Interval  # of Obs.
-----  ------------  ------  ---------------  -----------------  ---------
09:00        763266    0.2%             2444                  1         30
10:00       1307437    0.4%             2311                  0         30
11:00       1770413    0.6%             2198                  0         30
12:00       2157887    0.7%             2104                  0         30
13:00       2500051    0.8%             2020                  0         30
14:00       2836441    0.9%             1938                  0         30
15:00       3617025    1.2%             1747                  0         30
16:00       3669734    1.2%             2497                  1         30
SQLLOG        VM:DB/Monitor (c) 1998 Sterling Software, Inc.                                                    Page     2

                                                SQL/DS Log Usage

* * * Summary * * *               First Time: 01/30/98 08:28:00     Last Time: 01/30/98 16:26:49
             Intervals:        8
                                                                                                                Observations:
40

Database: FVVPDBA  at FVV        DBname  : FVVPDB               DBresrce: FVVPDB
Initialized at: 01/30/98 18:52:23

Log Size           :    306774016                                      <== Size of the log in bytes
Checkpoint Interval:         2500                                      <== Value of the CHKINTVL startup parameter

                         Average     Minimum     Maximum    Std.Dev.
                       ------------  ----------  ----------  ----------
Log Used           -   2327781.75      763266     3669734  1040585.83    <== Number of bytes in the log at end of a time interval
% Full             -         0.76           0           1        0.34    <== Percent of log full
Pages Remaining    -      2157.37        1747        2497      256.63    <== Number of log pages remaining before next checkpoint
Checkpoints Taken  -         0.25           0           1        0.46    <== Number of checkpoints taken during a time interval
```

Figure 138. DBMAP SQLLOG Summary Monitor Report

SYSSTATS Summary Monitor Log Report

The system statistics report includes statistics on the database virtual machine.

```
SYSSTATS      VM:DB/Monitor (c) 1998 Sterling Software, Inc.                                    Page     1

                                        System Statistics

                          First Time: 01/30/98 08:28:00     Last Time: 01/30/98 16:26:49

Database: FVVPDBA  at FVV        DBname: FVVPDB             DBresrce: FVVPDB    SQL/DS: 5.1    DCSS:
Initialized at: 01/30/98 18:52:23                                               CMS:     13   CP: ESA
2.0 XC

Virtual Storage:       80M   MAXCONN value     :    333
# of DBextents :       137   # of Storage Pools:     17
                                                  <-------------------------- Average -------------------------->

       <-- Milliseconds -->            <---- Paging ---->  Working  Working-Storage   Resident-Pages     Storage   Monitor  Number
Time   Virt. CPU  Total CPU  DASD I/Os     Reads    Writes    Set    Target  Actual    Main    Expanded  Utilized  Overhead  of Obs.
-----  ---------  ---------  ---------  --------  --------  -------  ------  ------  --------  --------  --------  --------  -------
09:00     198046     293548      19270      2249      1067     7774     N/A     31M      7827         0       60%     0.00%
   30
10:00     199262     295294      18076      1702      1136     7817     N/A     31M      7869         0       60%     0.00%
   30
11:00     255450     368647      19741      1058       691     7546     N/A     30M      7591         0       60%     0.00%
   30
12:00     176458     261371      13949      1373       948     7488     N/A     30M      7539         0       60%     0.00%
   30
13:00     171009     248963      12975       523       292     7577     N/A     30M      7623         0       60%     0.00%
   30
14:00     146044     216946      11874       408       222     7602     N/A     30M      7643         0       60%     0.00%
   30
15:00     194226     284528      19312       429         0     7915     N/A     32M      7959         0       60%     0.00%
   30
16:00      66589      91449      12797       191         0     8123     N/A     32M      8163         0       59%     0.00%
   30
SYSSTATS      VM:DB/Monitor (c) 1998 Sterling Software, Inc.                                    Page     2

                                        System Statistics

* * * Summary * * *              First Time: 01/30/98 08:28:00     Last Time: 01/30/98 16:26:49
    Intervals:        8
                                                                                          Observations:   240

Database: FVVPDBA  at FVV        DBname: FVVPDB             DBresrce: FVVPDB    SQL/DS: 5.1    DCSS:
Initialized at: 01/30/98 18:52:23                                               CMS:     13   CP: ESA
2.0 XC

Virtual Storage   :       80M                                          <== Virtual storage size of the database server u
erid
MAXCONN value     :       333                                          <== MAXCONN directory value (maximum IUCV connect
ons)
# of DBextents    :       137                                          <== Number of dbextents (minidisks) defined
# of Storage Pools:        17                                          <== Number Storage Pools defined

                      Total      Average    Minimum    Maximum   Std.Dev.
                    --------   ----------   --------   --------   --------
Virtual CPU      -   1407084    175885.50      66589     255450   54201.71   <==   Virtual CPU consumed during the interval
Total CPU        -   2060746    257593.25      91449     368647   80282.87   <==   Total CPU consumed during the interval
Combined I/Os    -    127994     15999.25      11874      19741    3393.43   <==   DASD I/O's + DS Page Faults during the interval
Page Reads       -      7933       991.62        191       2249     732.24   <==   VM page reads during the interval
Page Writes      -      4356       544.50          0       1136     473.27   <==   VM page writes during the interval
Working Set      -     (n/a)      7730.25       7488       8123     217.25   <==   Average working set size
Target Wkg Stor  -     (n/a)          N/A        N/A        N/A        N/A   <==   Average Target Working Storage size
Actual Wkg Stor  -     (n/a)       30.75M        30M        32M     0.89M   <==   Average Actual Working Storage size
Res Main         -     (n/a)      7776.75       7539       8163     215.80   <==   Average number of resident pages in main storage
Res Expanded     -     (n/a)         0.00          0          0       0.00   <==   Average number of resident pages in expanded stg
Storage Utilized -     (n/a)        59.87         59         60       0.35   <==   Average percent of virtual storage utilized
Monitor Overhead -     (n/a)         0.00       0.00       0.00       0.00   <==   Average percent of monitor overhead
```

Figure 139. DBMAP SYSSTATS Summary Monitor Report

THRESHOLD Summary Monitor Log Report

This report lists all THRESHOLDs which occured during the day.

```
THRESHOLD     VM:DB/Monitor (c) 1998, Sterling Software, Inc.                                                    Page     1
                                                Threshold Report
                             First Time: 01/29/98 09:51:03     Last Time: 01/29/98 10:04:06
Database: ELDB2A   at BOEVMCT1   DBname: ELDB2A               DBresrce: ELDB2A
Initialized at:     01/28/98 16:41:03
                                                                                                  Left-side  Right-side
  Time    Threshold   Expression                                                                    Value       Value
--------  ----------  ------------------------------------------------------------------------  ----------  ----------
09:51:18  TR1         EXIT(TR1EX)                                                                      0.0         0.0
09:53:18  TR1         EXIT(TR1EX)                                                                      0.0         0.0
09:55:18  TR1         EXIT(TR1EX)                                                                      0.0         0.0
09:57:18  TR1         EXIT(TR1EX)                                                                      0.0         0.0
THRESHOLD     VM:DB/Monitor (c) 1998, Sterling Software, Inc.                                                    Page     2
                                              Threshold Report
* * * Summary * * *          First Time: 01/29/98 09:51:03     Last Time: 01/29/98 10:04:06
Database: ELDB2A   at BOEVMCT1   DBname: ELDB2A               DBresrce: ELDB2A
Initialized at:     01/28/98 16:48:03
Seq   Threshold   Expression                                                                    Count
----  ----------  ------------------------------------------------------------------------  -----
   1  TR1         EXIT(TR1EX)                                                                     4
```

Figure 140. DBMAP THRESHOLD Summary Monitor Report

TOPUSERS Summary Monitor Log Report

This report lists the top users of virtual CPU time for the specified interval.

```
TOPUSERS      VM:DB/Monitor (c) 1998, Sterling Software, Inc.                                                    Page     1
                                                Top Users of Virtual CPU
                                First Time: 03/04/98 09:41:06     Last Time: 03/04/98 16:41:32
Database: FVVPDBA  at FVV        DBname: FVVPDB               DBresrce: FVVPDB
Initialized at: 03/02/98 05:37:25
Time    Userid     Node     SQLid     Observ.   Dispatches    Virt. CPU    BufLooks
-----  --------  --------  --------  -------   ----------   ----------   ---------
09:00  FVV429    FVV       FVV             2         8602        19470       53583
       FVV412    FVV       FVV             3         8363        16610       28122
       FVV434    FVV       FVV             5         8142        15883       31603
       FVV417    FVV       FVV             1         7121        13661       22638
10:00  FVV434    FVV       FVV             2        16757        32624       59262
       FVV412    FVV       FVV             2        14512        28528       50048
       FVV417    FVV       FVV             2        12525        25214       54592
       FVV429    FVV       FVV             2        10266        21953       57345
11:00  FVV434    FVV       FVV             1        19649        38907       75876
       FVV417    FVV       FVV             2        17605        35551       84007
       FVV412    FVV       FVV             1        17458        33807       57510
       FVV427    FVV       FVV             2        13453        30486       85954
12:00  FVV427    FVV       FVV             2        24099        54242      159715
       FVV421    FVV       FVV             2        22378        49629      127655
       FVV414    FVV       FVV             2        22018        47675      108301
       FVV416    FVV       FVV             1        19988        45161      121390
13:00  FVV427    FVV       FVV             1        31727        71548      211827
       FVV414    FVV       FVV             2        29093        64033      151894
       FVV421    FVV       FVV             2        28622        63394      167465
       FVV416    FVV       FVV             1        25665        55974      143219
14:00  FVV414    FVV       FVV             5        40246        85267      203871
       FVV421    FVV       FVV             4        30586        67112      171879
       FVV416    FVV       FVV             1        25796        56256      143489
       FVV412    FVV       FVV             4        28869        55247      105462
15:00  FVV421    FVV       FVV             2        37675        80292      196638
       FVV412    FVV       FVV             2        34258        64307      122057
       FVV434    FVV       FVV             1        30907        62396      127424
       FVV416    FVV       FVV             3        26842        59340      153950
16:00  FVV414    FVV       FVV             1        55979       116653      273362
       FVV511    FVV       FVV             1        34983        57857       81004
       FVV491    FVV       FVV             1        21917        55667      131803
       FVV464    FVV       FVV             1        21274        44848       97843
TOPUSERS      VM:DB/Monitor (c) 1998, Sterling Software, Inc.                                                    Page     2
                                                Top Users of Virtual CPU
* * * Summary * * *          First Time: 03/04/98 09:41:06     Last Time: 03/04/98 16:41:32                 Intervals:        8
                                                                                                            Observations:     8
Database: FVVPDBA  at FVV        DBname: FVVPDB               DBresrce: FVVPDB
Initialized at: 03/02/98 05:37:25
                                          # of Log    Total       Total        Total        Total
Seq    Userid     Node     SQLid     Intervals   Observ.   Dispatches    Virt. CPU    BufLooks
----  --------  --------  --------  ---------   -------   ----------   ----------   ---------
1     FVV414    FVV       FVV               5        12       159070       341639      803476
2     FVV421    FVV       FVV               7        16       142208       313685      781411
3     FVV412    FVV       FVV               7        15       147957       283367      514265
4     FVV434    FVV       FVV               7        14       144130       282482      554418
5     FVV417    FVV       FVV               7        14       124469       248634      575066
6     FVV416    FVV       FVV               4         6        98291       216731      562048
7     FVV429    FVV       FVV               6        13        99111       207391      512685
8     FVV427    FVV       FVV               4         7        76065       171576      498736
9     FVV423    FVV       FVV               6        12        82328       164778      333027
10    FVV420    FVV       FVV               6        14        73757       149863      360949
```

Figure 141. DBMAP TOPUSERS Summary Monitor Report

B.4 DBMAP Detail Reports

DETAIL Detail Report

This report is a record-by-record listing of the transaction detail records in the detail file.

```
Activity recording STARTED at 02/06/98 11:18:26.927397,  Database: ELDB2A   at BOEVMCT1,  DBNAME: ELDB2A
(Version: 4.00)
   1: Start=02/06/98 11:19:21.160937  VMid=ELRES5   Agent=  1     Opcode= 30  AUX Call (Execute Immediate)
      End=  02/06/98 11:19:21.219503 SQLid=ELRES5   Path=  32  Call-type=     Execute Immediate
      Start of LUW=  11:19:21.160863  Node=BOEVMCT1
      ACPU=      207  LUWID=   43311  Access Module= SQLDBA.ARIDSQL.6
      UCPU=        0  LPGB=      128  Isolation level=RR
      PAGR=       26  PAGW=        0  Row count=        1   RetCode=        0
      DIRR=        3  DIRW=        0  Locks=           67   SQLState=   00000
      LOGR=        0  LOGW=        0  Long Locks=      67   Warning=
      Estimated Cost=       4.33
      SQL Statement= UPDATE TCTPARAM SET FLAGEMIS='E' WHERE DTSOLICI = 970916 AND CODRELAT=110 AND FILIAL=1 AND EMPRESA=1
      DBSS Opcode Summary:   DBSS calls=      90   Last DBSS call at: 02/06/98 11:19:21.219298
      DBSS Table/Index Summary:   (14 entries)
         Dbspaceno=       1   Table-id=  -32742   Index-id=  -32700
         Dbspaceno=       1   Table-id=  -32740   Index-id=  -32699
         Dbspaceno=       1   Table-id=  -32746   Index-id=  -32705
         Dbspaceno=       2   Table-id=  -31933   Index-id=       0
         Dbspaceno=       1   Table-id=  -32740   Index-id=       0
         Dbspaceno=       1   Table-id=  -32758   Index-id=  -32717
         Dbspaceno=       1   Table-id=  -32760   Index-id=  -32718
         Dbspaceno=       1   Table-id=  -32765   Index-id=  -32761
         Dbspaceno=       1   Table-id=  -32754   Index-id=       0
         Dbspaceno=       1   Table-id=  -32756   Index-id=       0
         Dbspaceno=       1   Table-id=  -32732   Index-id=  -32695
         Dbspaceno=       1   Table-id=  -32748   Index-id=  -32710
         Dbspaceno=       1   Table-id=  -32730   Index-id=  -32694
         Dbspaceno=      38   Table-id=  -32759   Index-id=  -32757
   2: Start=02/06/98 11:19:21.222455  VMid=ELRES5   Agent=  1     Opcode= 30  AUX Call (Execute Immediate)
      End=  02/06/98 11:19:21.235777 SQLid=ELRES5   Path=  32  Call-type=     Execute Immediate
      Start of LUW=  11:19:21.160863  Node=BOEVMCT1
      ACPU=        8  LUWID=   43311  Access Module= SQLDBA.ARIDSQL.6
      UCPU=        2  LPGB=       78  Isolation level=RR
      PAGR=        4  PAGW=        0  Row count=        1   RetCode=        0
      DIRR=        0  DIRW=        0  Locks=           72   SQLState=   00000
      LOGR=        0  LOGW=        0  Long Locks=      72   Warning=
      Estimated Cost=       4.33
      SQL Statement= UPDATE TCTPARAM SET FLAGEMIS='M' WHERE DTSOLICI = 971119 AND CODRELAT=110 AND FILIAL=17 AND EMPRESA=
                     1
      DBSS Opcode Summary:   DBSS calls=     144   Last DBSS call at: 02/06/98 11:19:21.235589
      DBSS Table/Index Summary:   ( 9 entries)
         Dbspaceno=       1   Table-id=  -32758   Index-id=  -32717
         Dbspaceno=       1   Table-id=  -32760   Index-id=  -32718
         Dbspaceno=       1   Table-id=  -32765   Index-id=  -32761
         Dbspaceno=       1   Table-id=  -32754   Index-id=       0
         Dbspaceno=       1   Table-id=  -32756   Index-id=       0
         Dbspaceno=       1   Table-id=  -32732   Index-id=  -32695
         Dbspaceno=       1   Table-id=  -32748   Index-id=  -32710
         Dbspaceno=       1   Table-id=  -32730   Index-id=  -32694
         Dbspaceno=      38   Table-id=  -32759   Index-id=  -32757
```

Figure 142. DBMDET DETAIL Report

TABNDXD Detail Report

This report lists the dbspace, table-id and index-id for each table and index used by each package by user ID and SQL connect ID.

```
TABNDXD       VM:DB/Monitor (c) 1998, Sterling Software, Inc.            Page     1

                        Table and Index Usage - Detail

         First Time: 02/06/98 11:18:26     Last Time: 02/06/98 11:20:01
```

Seq	Table-id	Dbspno	Index-id	VMid	SQLid	Program	Author	# of Trans
1	-32767	9	-32765	ELRES5	ELRES5	CTPB110	ELRES5	3
2	-32767	39	-32765	ELRES5	ELRES5	CTPB110	ELRES5	2
3	-32767	40	-32764	ELRES5	ELRES5	CTPB110	ELRES5	1
4	-32767	40	0	ELRES5	ELRES5	CTPB110	ELRES5	1
5	-32767	41	-32765	ELRES5	ELRES5	CTPB110	ELRES5	1
6	-32767	41	0	ELRES5	ELRES5	CTPB110	ELRES5	1
7	-32765	1	-32761	ELRES5	ELRES5	ARIDSQL	SQLDBA	2
8	-32765	1	-32761	ELRES5	ELRES5	CTPB110	ELRES5	2
9	-32760	1	-32718	ELRES5	ELRES5	ARIDSQL	SQLDBA	2
10	-32760	1	-32718	ELRES5	ELRES5	CTPB110	ELRES5	2
11	-32759	38	-32757	ELRES5	ELRES5	ARIDSQL	SQLDBA	2
12	-32759	38	-32757	ELRES5	ELRES5	CTPB110	ELRES5	2
13	-32758	1	-32717	ELRES5	ELRES5	ARIDSQL	SQLDBA	2
14	-32756	1	0	ELRES5	ELRES5	ARIDSQL	SQLDBA	2
15	-32754	1	0	ELRES5	ELRES5	ARIDSQL	SQLDBA	2
16	-32748	1	-32710	ELRES5	ELRES5	ARIDSQL	SQLDBA	2
17	-32746	1	-32705	ELRES5	ELRES5	ARIDSQL	SQLDBA	1
18	-32746	1	-32705	ELRES5	ELRES5	CTPB110	ELRES5	1
19	-32742	1	-32700	ELRES5	ELRES5	ARIDSQL	SQLDBA	1
20	-32740	1	-32699	ELRES5	ELRES5	ARIDSQL	SQLDBA	1
21	-32740	1	-32699	ELRES5	ELRES5	CTPB110	ELRES5	1
22	-32740	1	0	ELRES5	ELRES5	ARIDSQL	SQLDBA	1
23	-32740	1	0	ELRES5	ELRES5	CTPB110	ELRES5	1
24	-32732	1	-32695	ELRES5	ELRES5	ARIDSQL	SQLDBA	2
25	-32730	1	-32694	ELRES5	ELRES5	ARIDSQL	SQLDBA	2
26	-32407	2	0	ELRES5	ELRES5	CTPB110	ELRES5	1
27	-31933	2	0	ELRES5	ELRES5	ARIDSQL	SQLDBA	1

Figure 143. DBMDET TABNDXD Report

TRANSTAT Detail Report

This report lists the count of each transaction opcode and the top ten transactions for agent CPU usage, I/O count and elapsed time.

```
TRANSTAT      VM:DB/Monitor (c) 1998, Sterling Software, Inc.            Page     1


                                          Transaction Statistics

                               First Time: 02/06/98 11:18:26     Last Time: 02/06/98 11:20:01

 Opcode/Description          :  Count     Opcode/Description          :  Count     Opcode/Description          :
ount      Total
 ------------------------------------     ------------------------------------     ------------------------------------     -------
 ( 30) AUX call              :  11077     (131) Create Program        :      0     (166) Modify Cancel         :
   0       11086
 ( 35) Setup (Prepare)       :      0     (132) Drop Statement        :      0     (170) Operator Cmd Continue :
   0
 ( 40) Describe              :      0     (135) Lookup                :      0     (175) Open  to Un/Reload AUX:
   0
 ( 45) Close                 :      4     (140) SQL call              :      0     (180) AUX call Un/Reload AUX:
   0
 ( 50) Open                  :      5     (145) Prepfinish            :      0     (185) Close to Un/Reload AUX:
   0
 (120) Connect/Schedule      :      0     (155) Operator Command      :      0     (190) Set/Change Language   :
   0
 (125) Recovery List         :      0     (160) Prepare-to-Commit     :      0     (195) Start call            :
   0
 (130) Prepinit              :      0     (165) Set/Reset Exit        :      0     (200) Reorganize call       :
   0


 <----------------------------- Top Ten Transactions By Agnt-CPU, DASDIO's, and Elapsed ------------------------------>

   VMid     SQLid      Starting Time of Day      Elapsed  Author.Program.Section  Opcode  Agnt-CPU    LPAGB   DASDIO's
 --------  --------  -------------------------  ---------  ----------------------  ------  --------  --------  --------
 ELRES5    ELRES5    02/06/98 11:19:27.453425   0:01.152  ELRES5.CTPB110.6            50       867*      437
209
 ELRES5    ELRES5    02/06/98 11:19:26.910733   0:00.541  ELRES5.CTPB110.5            50       308*      272
169
 ELRES5    ELRES5    02/06/98 11:19:21.160937   0:00.059  SQLDBA.ARIDSQL.6            30       207*      128
 29
 ELRES5    ELRES5    02/06/98 11:19:26.704729   0:00.071  ELRES5.CTPB110.3            50        20*      215
 38
 ELRES5    ELRES5    02/06/98 11:19:21.222455   0:00.013  SQLDBA.ARIDSQL.6            30         8*       78
  4
 ELRES5    ELRES5    02/06/98 11:19:26.685822   0:00.017  ELRES5.CTPB110.1            30         5*       28
  9
 ELRES5    ELRES5    02/06/98 11:20:01.618866   0:00.031  ELRES5.CTPB110.8            30         4*        0
  5
 ELRES5    ELRES5    02/06/98 11:19:26.902075   0:00.006  ELRES5.CTPB110.4            50         2*        6
  4
 ELRES5    ELRES5    02/06/98 11:19:30.291621   0:00.000  ELRES5.CTPB110.6            30         2*        2
  0
 ELRES5    ELRES5    02/06/98 11:19:30.995769   0:00.001  ELRES5.CTPB110.6            30         2*        1
  0

 ELRES5    ELRES5    02/06/98 11:19:27.453425   0:01.152  ELRES5.CTPB110.6            50       867       437
209*
 ELRES5    ELRES5    02/06/98 11:19:26.910733   0:00.541  ELRES5.CTPB110.5            50       308       272
169*
 ELRES5    ELRES5    02/06/98 11:19:26.704729   0:00.071  ELRES5.CTPB110.3            50        20       215
 38*
 ELRES5    ELRES5    02/06/98 11:19:21.160937   0:00.059  SQLDBA.ARIDSQL.6            30       207       128
 29*
 ELRES5    ELRES5    02/06/98 11:19:26.685822   0:00.017  ELRES5.CTPB110.1            30         5        28
  9*
 ELRES5    ELRES5    02/06/98 11:20:01.618866   0:00.031  ELRES5.CTPB110.8            30         4         0
  5*
 ELRES5    ELRES5    02/06/98 11:19:21.222455   0:00.013  SQLDBA.ARIDSQL.6            30         8        78
  4*
 ELRES5    ELRES5    02/06/98 11:19:26.902075   0:00.006  ELRES5.CTPB110.4            50         2         6
  4*
 ELRES5    ELRES5    02/06/98 11:19:21.237149   0:00.007  SQLDBA.ARIDSQL.1            30         1         0
  1*
 ELRES5    ELRES5    02/06/98 11:19:26.703735   0:00.000  ELRES5.CTPB110.2            30         0         7
  0*

 ELRES5    ELRES5    02/06/98 11:19:27.453425   0:01.152* ELRES5.CTPB110.6            50       867       437
209
 ELRES5    ELRES5    02/06/98 11:19:26.910733   0:00.541* ELRES5.CTPB110.5            50       308       272
169
 ELRES5    ELRES5    02/06/98 11:19:26.704729   0:00.071* ELRES5.CTPB110.3            50        20       215
 38
 ELRES5    ELRES5    02/06/98 11:19:21.160937   0:00.059* SQLDBA.ARIDSQL.6            30       207       128
 29
 ELRES5    ELRES5    02/06/98 11:20:01.618866   0:00.031* ELRES5.CTPB110.8            30         4         0
  5
 ELRES5    ELRES5    02/06/98 11:19:26.685822   0:00.017* ELRES5.CTPB110.1            30         5        28
  9
 ELRES5    ELRES5    02/06/98 11:19:21.222455   0:00.013* SQLDBA.ARIDSQL.6            30         8        78
  4
 ELRES5    ELRES5    02/06/98 11:19:21.237149   0:00.007* SQLDBA.ARIDSQL.1            30         1         0
  1
 ELRES5    ELRES5    02/06/98 11:19:26.902075   0:00.006* ELRES5.CTPB110.4            50         2         6
  4
 ELRES5    ELRES5    02/06/98 11:19:26.775829   0:00.004* ELRES5.CTPB110.3            30         1         0
  0
```

Figure 144. DBMDET TRANSTAT Report

Appendix C. Special Notices

This publication is intended to provide database administrators, application programmers, and system programmers guidance in the area of performance tuning of DB2 Server for VSE & VM Version 5 Release 1.

The information in this publication is not intended as a specification of any programming interfaces that are provided by DB2 Server for VSE & VM Version 5 Release 1 or any of its features. See the PUBLICATIONS section of the IBM Programming Announcement for DB2 Server for VSE & VM Version 5 Release 1 for more information on 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 pointers in this publication to external Web sites are provided for convenience only and do not in any manner serve as an endorsement of these Web sites.

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:

3090	3890
ACF/VTAM	Advanced Peer-to-Peer Networking
AIX	APPN
AS/400	C/370
C/VM	CICS
CICS/VSE	CUA
Current	DATABASE 2
DataGuide	DataHub
DataJoiner	DataPropagator
DB2	DFSMS
DFSMS/VM	Distributed Relational Database Architecture
DProp	DRDA
ECKD	ES/9000
ESA/370	ESA/390
ESCON	IBM
IIN	IMS
ISSC	Language Environment
Micro Channel	Multiprise
MVS	Nways
Operating System/2	OS/2
OS/3	OS/390
OS/400	Power Series
Presentation Manager	PROFS
PS/2	QMF
RAMAC	RETAIN
RISC System/6000	RS/6000
RT	S/370
S/390	SKI
SP	SQL Master
SQL/DS	System/370
System/390	Virtual Machine/Enterprise Systems Architecture
Visual Warehouse	VM/ESA
VSE/ESA	VTAM

The following terms are trademarks of other companies:

C-bus is a trademark of Corollary, Inc.

Java and HotJava are trademarks of Sun Microsystems, Incorporated.

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

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

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

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

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

Appendix D. Related Publications

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

D.1 International Technical Support Organization Publications

For information on ordering these ITSO publications see “How to Get ITSO Redbooks” on page 209.

- *DB2 Recovery on VSE and VM Using the Data Restore Feature*, SG24-2039
- *SQL/DS Version 3 Release 4 Performance Guide*, GG24-4047

D.2 Redbooks on CD-ROMs

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

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

D.3 Other Publications

These publications are also relevant as further information sources:

VSE and VM

- *DB2 Server for VSE & VM Performance Tuning Handbook, SC09-2402*
- *DB2 Server for VSE & VM Data Restore Guide Version 5 Release 1, SC09-2275*
- *DB2 Server for VSE & VM SQL Reference, SC09-2404*

VM only

- *DB2 Server for VM System Administration, GC09-2405*
- *DB2 Server for VM Database Administration, GC09-2388*
- *DB2 Server for VM Application Programming, SC09-2392*
- *DB2 Server for VM Database Service Utility, SH09-8088*
- *DB2 Server for VM Diagnosis Guide and Reference, SC09-2407*
- *DB2 Server for VSE & VM Control Center Installation and Operations Guide for VM, SC67-0205*
- *DB2 Server Data Spaces Support for VM/ESA, SC09-2411*

VSE only

- *DB2 Server for VSE System Administration, GC09-2406*
- *DB2 Server for VSE Database Administration, GC09-2389*
- *DB2 Server for VSE Application Programming, SC09-2393*
- *DB2 Server for VSE Database Service Utility, SC09-2395*
- *DB2 Server for VSE Diagnosis Guide and Reference, SC09-2408*
- *DB2 Server for VSE & VM Control Center Installation and Operations Guide for VSE, SC17-2000*
- *VSE System Control Statements, SC33-6513*
- *VSE/Virtual Storage Access Method: Commands and Macros, SC33-6532*

How to Get ITSO Redbooks

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

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

How IBM Employees Can Get ITSO Redbooks

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

- **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 BookManager BOOKs of redbooks, type the following command:

  ```
  TOOLCAT REDBOOKS
  ```

  To get lists of redbooks, type the following command:

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

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

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

  For a list of product area specialists in the ITSO: type the following command:

  ```
  TOOLS SENDTO WTSCPOK TOOLS ZDISK GET ORGCARD PACKAGE
  ```

- **Redbooks Web Site 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.

- **REDBOOKS 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.

Redpieces

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

How Customers Can Get ITSO Redbooks

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

- **Online Orders** — send orders to:

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

- **Telephone orders**

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

- **Mail Orders** — send orders to:

IBM Publications	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 Allerød
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)001-408-256-5422 (Outside USA)** — ask for:

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

- **Direct Services** - send note to softwareshop@vnet.ibm.com
- **On the World Wide Web**

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

- **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).

Redpieces

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

IBM Redbook Order Form

Please send me the following:

Title	Order Number	Quantity

First name Last name

Company

Address

City Postal code Country

Telephone number Telefax number VAT number

- Invoice to customer number
- Credit card number

Credit card expiration date Card issued to Signature

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

Glossary

2NF. Second normal form. Refer to 9.6.3, “Normalization” on page 142.

3NF. Third normal form. Refer to 9.6.3, “Normalization” on page 142.

address space. A virtual storage area holding program or data. A program residing there can be executed, see data space.

archive. Generic term for a backup of a complete database. Determine from context whether a Data Restore archive (created by the Data Restore Feature command BACKUP) or an DB2 archive is meant. See also “online” and “offline.”

ARCHIVE. The DB2 command ARCHIVE.

ARISPOOL. “FILEDEF ARISPOOL DISK fn ft fm” defines the file that controls the VMDSS settings, which can be individual for each storage pool.

backup. Generic term for backing up data. This can be a user archive, or any of the DB2 commands DATAUNLOAD, UNLOAD, ARCHIVE or of the Data Restore Feature commands UNLOAD, SELECT, BACKUP.

buffer. A portion of storage used to hold input or output data temporarily.

CEDA. A CICS transaction for resource definition online (RDO).

CEDF. A CICS transaction for debugging.

CEMT. A CICS control transaction.

checkpoint. A set of cleanup and recovery actions taken periodically by the database manager. Actions include writing a copy of the database to disk, recording information in the DB2 log, freeing storage pool space and updating the directory.

cluster. A VSAM file in VSE. In VSE, a dbextent is implemented by a VSAM cluster.

clustered index. An index whose sequence of key values closely corresponds to the sequence of rows stored in the table to which the index belongs.

clustering index. The first index created for a table. The DB2 database manager uses it to determine the placement of subsequent rows.

commit. (1) The operation that terminates a unit of work by releasing locks so that the database changes made by that unit of work can be perceived by other processes. (2) The process that allows data changes to be made permanent. When a commit occurs, other applications can reference the just-committed data.

concurrency. The shared use of resources by multiple interactive users or application processes at the same time.

COND. CONDITION for the UNLOAD command of Data Restore Feature. Identifies whether the list of dbspaces is to be included or excluded for unloading.

Control Center. 1. For VM & VSE - A set of database administration tools that automate database system administration and operations. Control Center is the successor of SQL Master, and is a charged feature of IBM DB2 Server for VSE & VM Version 5.

2. Part of DB2 Common Server, installed on either an OS/2 or AIX platform. Performs the administration for data replication.

CP. Control Program, the lowest layer of VM.

CP Monitor. A performance monitoring tool in VM.

CPU. Central Processing Unit (processor, the instruction executor of a computer)

CS. See Cursor Stability.

Cursor Stability . An isolation level that locks just the data you are using. When the cursor moves to a new row or page, the lock moves also (depending on the locking level defined) and the lock upon previously read data is released. CS does less locking than RR, but more than UR.

CUU. Channel and Unit address (virtual device address).

database management system (DBMS). A software system that controls the logical and physical resources and facilities of a database.

database manager. A program that processes SQL statements.

Data Restore archive. Archive, taken with the Data Restore Feature command BACKUP. Nearly identical is a translated DB2 archive, see “translated archive.”

DASD. Direct Access Storage Device, a hard disk.

Data Restore Feature. A feature of DB2 Server for VSE & VM that can be used for archiving and restoring DB2 Server for VSE & VM

DB2 archive. Archive, taken online with the DB2 command ARCHIVE or offline with the DB2 command SQLEND ARCHIVE.

data space. Also called ESA Data space (introduced with ESA/370 architecture). A virtual storage area used to keep data. A program residing there cannot be executed, as opposed to an address space. Exploited by DB2 in VM using VMDSS.

dbextent. The physical medium where database data is stored. In VM, an extent is implemented by a minidisk, in VSE by a VSAM cluster. Storage pools are composed of one or more dbextents.

dbspace. A logical allocation of space in a storage pool contained in a database. Contains one or more tables and their associated indexes.

DBMAP. VM:DB/Monitor analysis program. DBMAP reads activity and monitor log files to produce output in the form of reports and history files.

DBSS. Database Storage Subsystem, a part of DB2 Server for VSE & VM.

DBSU. Database Services Utility, a part of DB2 Server for VSE & VM.

DDR. DASD Dump Restore, a VM tool for disk backup.

deadlock. An impasse that occurs when a process is waiting for a resource that is being held by another process that is waiting for a resource currently being held by the first process.

DELETE. An SQL command to drop a row from a table.

dependent table. A DB2 table which contains the foreign key relating to a primary key in another table. See *parent table*.

DFHPCT. CICS Program control table.

DFHPPT. CICS Processing program table

DFHFCT. CICS File control table

DFHPLTPI. CICS Program list table (PLT) that contains a list of programs to be executed after system initialization processing.

DFHPLTSD. CICS Program list table (PLT) that contains a list of programs to be executed during system termination.

DFHSIT. CICS System initialization table

Dirty Read . See Uncommitted Read.

entity integrity. The state in which all values of a table′s primary key are unique and not null regardless of changes made to data in the table.

exec. A procedure language program executed under VM to process a sequence of commands.

EXEC. Filetype of an exec in VM.

extent. See dbextent.

foreign key. A DB2 column referring to a primary key column of another table. See *primary key*.

forward log recovery. see forward recovery

forward recovery. After a backup had been taken, e.g. from a table (or a database), changes can have been made. DB2 writes these changes into the log and possibly into log archive(s). When a failure occurs and the table (or the database) has been restored, those updates must be applied to get the identical status of the table (or database) at the moment of the failure.

guest sharing. A DB2 Server for VSE & VM facility that enables a user or application in a VSE, which runs as a guest operating system under VM, to access a VM DB2 database.

IDBM. Interactive VM:DB/Monitor program. The user interface to the real-time monitoring function of VM:DB/Monitor.

IDCAMS. A utility program, which is a part of VSAM in VSE.

INSERT. An SQL command to add a row to a table.

IPL. Initial Program Load, i.e. starting or booting an operating system.

JCL. Job Control Language, for batch processing in VSE.

key. See *foreign key* and *primary key*, *parent table* and *dependent table*.

lock. Mechanism used by the database manager to ensure the integrity of data. Locking prevents concurrent users from accessing inconsistent data.

log. A collection of records maintained by the DB2 database manager to describe events that occurred during the operation of the database. This information is used for recovery if a failure occurs while the database manager is executing.

log archive. A log archive is a tape or disk copy of the log, recording just the changes applied to the data since the last archive or log archive.

Logical Unit of Work (LUW). A recoverable sequence of operations within an application process. At any time, an application process is a single unit of work, but the life of an application process can involve many units of work as a result of “commit” or “rollback” operations.

LPAR. Hardware feature to emulate multiple VSE partitions in the same processor.

MDC. Minidisk Caching (VM)

MUM. Multiple User Mode. A mode of operating the DB2 database manager, in which one or more users or application programs can access the database at the same time. Contrast with single user mode (SUM).

NLRBS. Number of Lock Request Blocks for the System. This is an initialization parameter.

NLRBU. Number of Lock Request Blocks for each User. This is an initialization parameter.

NULL. NULL is the contents of an “empty” column in DB2. NULLs can expressly be allowed or forbidden for each column. A NULL is not a zero (“0”), but, for example, will be ignored when building an average value.

offline. While the database manager is not active. Archives that can be taken offline are DB2 archives, Data Restore Feature BACKUP and user archives.

online. While the database manager is active. Archives that can be taken online are only DB2 archives.

parent table. The related DB2 table with the primary key column referenced by the foreign key column in a dependent table.

PCT. Processing Control Table, one of many CICS control tables. Assembling these tables is the older method compared to RDO.

pool. See storage pool.

PPT. Processing Program Table, one of many CICS control tables. Assembling these tables is the older method compared to RDO.

primary key. The DB2 columns which together serve as the unique identifier of a relational row. See *foreign key*, *parent table* and *dependent table*.

PSP. Preventive Service Planning information offered for IBM programs.

QMF. Query Management Facility, an IBM application program running on VSE/ESA (and other platforms) to query DB2 Server for VSE & VM data.

R/O. Read only - data modification cannot take place.

R/W. Read and Write - data modification can take place.

recover. Either a **generic term** for repairing a system, a database or data from a failure.

specific term: to apply to a restored table those changes that had occurred after the backup copy had been taken. See forward recovery.

RDO. Resource Definition Online, a method to configure CICS.

referential constraint. The referential integrity rule that the non-null values of the foreign key are valid only if they also appear as values of a primary key.

referential integrity. The state of a database in which all values of all foreign keys are valid.

reload. Generic term for loading data or loading a part of a database, like a table. This can be performed with either of the DB2 Server for VSE & VM commands DATALOAD, RELOAD or the Data Restore Feature command RELOAD.

Repeatable Read . An isolation level that locks data for the length of an LUW. RR does more locking than CS and UR.

Request Parameter List. A VSAM control block used by VSAM to perform an I/O.

restore. Generic term for restoring data. This can be performed either through starting the database with the parameter STARTUP=R or F, or with a user restore (like DDR) and then starting the database with the parameter STARTUP=U, or with either of the DB2 commands DATALOAD, RELOAD or the Data Restore Feature commands RELOAD, RESTORE.

retention period . Specifies in VSE, how long a file is to be kept. 0 days means, it is not kept.

rollback. The process of restoring data changed by SQL statements to the state at its last commit point. All locks are freed. Contrast with commit.

RPL. Request Parameter List. This is a VSAM control block used by VSAM to perform an I/O.

RR. See Repeatable Read.

SELECT. Either

- An SQL command used to retrieve data from a database.
- A Data Restore Feature function to select data from DB2 tables directly out of the dbextents, bypassing the database manager.

SCIF. Single Console Image Facility, a tool incorporated in VM/ESA.

SQLDBSU. SQL Database Services Utility, a part of DB2.

SQL Master. The predecessor of Control Center. An IBM program that provides a set of database

administrator tools for SQL/DS databases within a VM or VSE environment.

SQL-Tune. A vendor program to tune database and applications.

storage pool. A storage pool is composed of one or more dbextents, and defines the physical space for one or more dbspaces.

SUM. Single User Mode. A mode of operation, in which the DB2 database manager and one application run in the same virtual machine. No other application programs or users can access the database at the same time. Contrast with multiple user mode (MUM).

SYSIN. An input file for an exec in VM, containing definitions.

SYSPRINT. An output file from a VM process.

Uncommitted Read. A new Isolation level that allows access to data that has changed but not yet committed. Also called dirty read. UR does less locking than CS and RR.

unit of work. See “logical unit of work (LUW).”

unload. Generic term to extract data from a database. This can be either of the DB2 commands DATAUNLOAD, UNLOAD, Data Restore Feature commands SELECT, UNLOAD or Control Center commands SQLTABLE, SQLREORG.

UNLOAD. Either the DB2 Database Services Utility command UNLOAD, or the Data Restore Feature command UNLOAD. See the context to find which is meant, either in the heading, or in the words “DB2 UNLOAD” versus “Data Restore UNLOAD.” Database Services Utility UNLOAD copies data from a dbspace or table. Data Restore UNLOAD copies data from one or more dbspaces.

update. Generic term. In DB2, this might include the commands UPDATE, INSERT and DELETE.

UPDATE. An SQL command to modify existing data.

UR. See Uncommitted Read.

user archive. An archive that is taken using non-DB2 facilities.

VM. Any version and release of VM/ESA, supported by DB2 Server for VSE & VM Version 5 Release 1.

VM/ESA. Virtual Machine/Enterprise Systems Architecture, a System/390 operating system.

VMDSS. DB2 Data Spaces Support for VM/ESA, previously a charged feature of SQL/DS. Integrated in DB2 Server for VM Version 5. Installation steps are required to activate it.

VMPRF. VM Performance Reporting Facility, an IBM program.

VSAM. Virtual Storage Access Method, a file access method in VSE, VM and MVS.

VSE. Any version and release of VSE/ESA, supported by DB2 Server for VSE & VM Version 5 Release 1.

VSE/ESA. Virtual Storage Extended/Enterprise Systems Architecture, a System/390 operating system.

List of Abbreviations

ACCT	ACCounT
ADMS	Application Display Management System
AIX	Advanced Interactive Executive
AMODE	Addressing Mode
AP	Application Programmer
APAR	Authorized Program Analysis Report
APPC	Advanced Program-to-Program Communication
APPLID	APPLication IDentifier
APPN	Advanced Peer-to-Peer Networking
AR	Attention Routine
AVG	AVeraGe
AVS	APPC/VM VTAM Support
BASIC	Beginners All-purpose Symbolic Instruction Code
BMS	Basic Mapping Support
CACHE	high speed buffer
CAD	Computer Aided Design
CADI	Computer Aided Design Integration
CCW	Channel Command Word
CD-ROM	Compact Disk - Read Only Memory
CDRM	Cross-Domain Resource Manager
CDRSC	Cross-Domain ReSourCes
CEDA	Resource Definition Online Transaction
CEMT	Master Terminal Transaction
CET	Central European Time
CGI	Common Gateway Interface
CICS	Customer Information Control System
CIRD	Change In Request Date
CMS	Conversational Monitor System
CNC	Communications Network Concepts
COBOL	COmmon Business Oriented Language
CP	Control Program
CPI-C	Common Programming Interface for Communications
CPU	Central Processing Unit
CRR	Coordinated Resource Recovery
CS	Cursor Stability
CSA	Common System Area
CTCA	Channel To Channel Adapter
CYL	CYLinder
DA	Data Administrator
DASD	Direct Access Storage Device
DB	Data base
DB2	DATABASE 2
DBA	Data Base Administrator
DBCS	Double Byte Character Set
DBM	Data Base Manager
DBS	Data Base Services
DBSS	Data Base SubSystem
DBSU	Data Base Services Utility
DCL	Data Control Language
DCS	Database Communication Service
DDCS	Distributed Database Connection Services
DDL	Data Definition Language
DFSMS	Data Facility Storage Management Subsystem
DISP	DISPosition
DL/I	Data Language 1
DLBL	Disk LaBeL
DML	Data Manipulation Language
DOS	Disk Operating System
DRDA	Distributed Relational Database Architecture
DS	Data Set
DUOW	Distributed Unit of Work
EOJ	End Of Job
ESA	Enterprise Systems Architecture
EXP	EXPlain
FCT	File Control Table

FM File Mode

FSF Forward Space File

GCS Group Control System

GIF Graphics Interchange Format

GUI Graphical User Interface

HP-UX Hewlett-Packard UNIX

HTML HyperText Markup Language

HTTP HyperText Transfer Protocol

HWM High Water Mark

I/O Input/Output

IBM International Business Machines

ICCF Interactive Computing and Control Facility

ICN InterChange Node

ID IDentification/IDentifier

IEEE Institute of Electrical and Electronics Engineers

II Interactive Interface

IML Initial Microprogram Load

IPL Initial Program Load

ISC InterSystem Communication

ISDN Integrated-Services Digital Network

ISQL Interactive Structured Query Language

ISSC Information Systems Support Center

ITSO International Technical Support Organization

IUCV Inter-User Communication Vehicle

JCL Job Control Language

JCT Journal Control Table

KB KiloByte

LAN Local Area Network

LEN Low End Network

LLC Logical Link Control

LRB Lock Request Block

LRU Least Recently Used

LU Logical Unit

LUA Logical Unit Address

LUW Logical Units of work

MB MegaByte

MDC MiniDisk Caching

MPTS Multiple Protocol Transport Services

MR Modified Read

MS Master Schedule

MUM Multiple User Mode

MVS Multiple Virtual Storage

NCP Network Control Program

NETBIOS NETwork Basic Input Output System

NLS National Language Support

NT Microsoft Windows NT

NTRI NCP Token Ring Interconnect

NVS Non Volatile Storage

OS/2 Operating System/2

OS/390 Operating System/390

PC Personal Computer

PCCU Primary Communication Control Unit

PCMCIA Personal Computer Memory Card International Association

PCT Program Control Table

PL/I Programming Language 1

PLT Program List Table

PPT Processing Program Table

PROP Programmable Operator

PTF Program Temporary Fix

PU Physical Unit

QMF Query Management Facility

R/O Read/Only

R/W Read/Write

RAMAC Brand name and trademark of IBM DASD family

RDBMS Relational Database Management System

RDO Resource Definition On-line

RDS Relational Data System

REW REWind

REXX REstructured eXtended eXecutor language

RISC Reduced Instruction-Set Computer

RMODE Residency MODE

RR Repeatable Read

RSL Remote Service Link

RU Request/response Unit

RUOW	Remote-Unit-Of-Work
SAM	Sequential Access Method
SCIF	Single Console Image Facility
SDK	Software Developer's Kit
SDLC	Synchronous Data Link Control
SIO	Start I/O
SIT	System Initialization Table
SNA	Systems Network Architecture
SP	System Programmer
SPLR	Storage Pool Level Recovery
SQL	Structured Query Language
SQL/DS	Structured Query Language/Data System
SRT	System Recovery Table
TCM	Target Communications Manager
TCP/IP	Transmission Control Protocol/Internet Protocol
TP	TeleProcessing
TPN	Transaction Program Name
TS	Temporary Storage
UR	Uncommitted Read
URL	Uniform Resource Locator
VM	Virtual Machine
VM/ESA	Virtual Machine/Enterprise Systems Architecture
VM/VTAM	VM/Virtual Teleprocessing Access Method
VMDSS	VM Data Spaces Support
VSAM	Virtual Storage Access Method
VSE	Virtual Storage Extended
VSE/ESA	Virtual Storage Extended/Enterprise Systems Architecture
VSE/POWER	VSE/Priority Output Writers, Execution processor, input Readers
VSE/VSAM	Virtual Storage Extended/Virtual Storage Access Method
VTAM	Virtual Telecommunications Access Method
WAN	Wide Area Network
WARP	Workstation Asset Reduction Program
WR	WRite
WWW	World Wide Web
XA	Extended Addressing
XEDIT	eXtended EDITor
XID	Exchange IDentifier
XRF	Extended Recovery Facility

Index

Special Characters

*BLOCKIO 23

Numerics

16MB limit 17
24-bit addressing 17
2GB 17
2NF 128, 213
31-bit addressing 17, 110
3990 23
3NF 128

A

abbreviations 217
access path 105
acronyms 217
ACTDETL report 168
ACTVTIME report 169
additional column in SELECT 128
address space 213
addressability extension 110
addressing mode 110
AMODE(31) 110
analyzing SQL statements 111
AP 109, 217
application
  check 126
  debugging 99
  define rules 108
  development steps 105
  life 111
  maintenance 110
  map conventions 108
  program naming conventions 108
  programmer 109
  restrict executions 110, 113
  tuning 115
archive 9, 44, 213
ARI2052I 12
ARISDIRD.A 106
ARISPOOL 138, 213
AUTOCOMMIT ON 129

B

backup 213
balance DASD 130
bibliography 207
buffer 135, 213
buffer maximums 17
buffer statistics 98
buffering extended in VSE 5
buffers consumption 166
BUFFERS report 184
BUFFERS screen 149
business data requirement 110

C

caching 23
cancelling application programs 111
catalog
  index fragmented 130
  locking 113
  update manually 105
CBL 110
CCSID 126
CEDA 113, 213
CEDF 111, 138
CEMT 213
check system 137
checkpoint 13, 213
  considerations 14, 18
  default setting 14
  information 12
  performance implications 14
  recommendation 15
CHKINTVL 10, 13, 136, 140
CICS
  groups 108
  restrict executions 110, 113
  TCLASS 110, 113
  user IDs 106
cluster 213
clustered index 213
COBOL for VSE 110
code reuse 110
column
  checking in table definition 112
  name 112
  unused in SELECT 126, 128
commit 213
COMMIT WORK 129
COMMMITCOUNT 130
comparisons 126
compiling 108
concurrency 213
concurrency defining objects 113
consistent state 111
contention in locking 8
Control Center 213
  administration tools 49
  advantages 33
  archive 44
  benefits 33
  CICS report controller 34

Control Center *(continued)*
code disk 40
code owner 39
control files 50
COUNTER * 35
data restore machine 39
database administration tools 45
database analysis 50
database commands 51
database monitoring 44
database operator commands 45
database startup and termination 45
database status 45
dbextent storage pool mapping 50
dbspace analysis 34, 37
dbspace maintenance 47
dbspace reorganization 38, 46
dbspace reorganization driver 46
DDL generation 33
environment 39
group authority 34
group authorization 51
index reorganization 48
job scheduling 49
message log 50
monitor thresholds 37
monitor utility 34, 36
object search 49
package utility 34
recovery 44
reload dbspace 33
reorganize dbspace 33
SCIF 43
service machine 39
SHOW ACTIVE 35
SHOW CONNECT 35
SHOW DBEXTENT 35
SHOW DBSPACE 35
SHOW LOCK 35
SHOW LOG 35
SQLMAINT 47
SQLREODR 46
SQLREORG 46
SQLRINDX 48
SQLTABLE 47
SQMUTIL 49
SUM 45, 46
support machine 39
system administration tools 43
table reorganization and redefinition 47
tape management 49
tools 42
unload dbspace 33
VSE 33
work disk 40
work file label definition 34
Controller DASD 23
conventions in naming objects 108
conversational 129
COUNTER 15, 135
ESCALATE 135
LOCKLMT 135
COUNTER * 35
COUNTER INTERNAL 140
COUNTER POOL 139
COUNTERS report 185
COUNTERS screen 150
CP Monitor 213
CPU 213
CPU statistics 97
cursor stability 6, 213
CUU 213

D

DA 109, 217
DASD 23, 213
DASD balancing 130
data
integrity 7
types 126, 142
uncommitted read 5
data administrator 109
Data Restore Feature 213
data space 213
checklist 138
DATA(31) 110
database
check 129
efficiency 110
growth 115
management system (DBMS) 213
manager 213
monitoring 44
objects definition 112
setup 133
status 45
test 105
database administrator 109
DATABASE screen 150
database service utility (see SQLDBSU)
DATALOAD COMMITCOUNT 130
DB/Monitor (see VM:DB/Monitor)
DBA 109
dbextent 214
DBMAP 95, 214
accounting information 99
activity reports 95, 166
activity session start/end 177
agent CPU 178
application debugging 99
buffers 98, 184
buflooks 178
CPU 97
creating reports 89
database performance 191

DBMAP *(continued)*
database statistics 186
dataspace counters 188
DB2 system counters 185
DBMONVM INITIAL file 88
dbspace references 180
dbspaces references 100
dbspno 181
detail reports 96
dispatch count 178
DRDA 177
dynamic SQL 99
dynamic statements 171
I/O 97
index-id 181
INPUT record 89
LOCK wait 182
locks 99
log statistics 194
minidisk statistics 189
monitor log reports 96, 183
monitor records 189
package statistics 173, 175
predefined activity reports 95
predefined detail reports 97
predefined monitor log reports 96
RANGE record 89
REPORT record 89
reports usage 97
section statistics 176
SELECT record 89
session statistics 177
specification file 89
storage 98
table references 180
table-id 181
tables references 100
user reports 177
VM user ID statistics 179, 181, 182, 195
XDBMAP module 177
DBMDET
overhead 97
DBMON 87, 91
DBMONUP 102
PROFILE EXEC 102
VM user ID 87
DBMONVM 88, 91
ACQUIRE command 89
CLOSE command 89
LOG command 89
PROFILE EXEC 88
RECORD command 89
VM user ID 87
DBMS 213
dbname directory 106
dbspace 214
analysis 37
maintenance 47
dbspace *(continued)*
reorganization 38, 46
scan 111
statistics 100, 116
DBSS 214
DBSTATS report 186
DBSTATS screen 152
DBSU 113, 123
DCL 217
DDL 113
DDR 214
DEADLOCK report 170
deadlocks 8, 214
default setting 14
DELETE 214
dependent table 214
destage 26
DETAIL report 198
DFHFCT 57, 214
DFHPCT 57, 114, 214
DFHPLTPI 57, 58, 214
DFHPLTSD 57, 58, 214
DFHPPT 57, 214
DFHSIT 57, 58, 114, 214
directory buffer 17
directory dbname 106
DIRREAD 12
dirty read 5, 214
DIRWRITE 12
DISPBIAS 10, 131, 136
DISTINCT 127
documention 110, 112
DROP DBSPACE 130
DROP TABLE 130
DSCNTRS report 188
DSCNTRS screen 153
DSPSTATS 10, 11, 15
DSREAD 12
DSWRITE 12
DUMPTYPE 10
duplicate system 105
dynamic partitions 106
dynamic statements 99, 123
DYNSQL report 171

E

efficient databases 110
email - see internet address
encapsulate function 110
entity integrity 214
environment parameters 10
ESCALATE 135
Exclusive mode 7
EXPLAIN 127
extended addressability 110
extended user buffering 5
extent 214

EXTENTS report 189
EXTENTS screen 154

F

Fast Write DASD 23
foreign key 214
forward recovery 214
function encapsulating 110
function test 109

G

glossary 213
GRANT 113
GROUP BY 128
guest sharing 214

H

HISTORY report 189
hit ratio 23

I

I/O statistics 97
IBM
ICCF 106
IDBM 89, 94, 214
  entry panel 147
  main menu 147
  predefined screens 94, 147
  user written screen 164
IDCAMS 214
IN lock mode 7
increased buffer maximums 17
index 131
  for UPDATE 126
  fragmentation 131
  reorganization 48
  statistics 117
  usage 122
index reorganization 48
index-only access 128
initialization parameters 10, 15, 134
  CHKINTVL 10, 13, 136, 140
  DISPBIAS 136
  DSPSTATS 10, 11, 15
  DUMPTYPE 10
  LTIMEOUT 10, 15, 135
  MAPPING 138
  NCUSERS 135
  NDIRBUF 135
  NLRBS 8, 135, 215
  NLRBU 8, 135, 215
  NPACKAGE 136
  NPACKPCT 136
  NPAGBUF 135
  SAVEINT 19
initialization parameters *(continued)*
  SAVEINTV 140
  SEPINTDB 139
  TARGETWS 139
INSERT 214
integrity 112
  of data 7
  reference 112
  testing 111
intent none mode 7
inter-application standards 108
internal dbspaces 129
internet address
  IBM redbooks - see last appendix
  Sterling Software 85
  The Paragon Collection 53
  XT-SOFT 53
IPL 214
IS NOT NULL 128
ISOL 130
isolation level 5, 129
ISQL 113, 123, 130

J

JCL 54, 56, 58, 78, 138, 214
jobs
  documentation 112
  run in test 106

K

key 214

L

level of isolation 5, 127, 129
life of application 111
limits of buffers 17
link-edit 108, 110
lock 214
  contention 8
  modes 7
  on catalog 113
  request block 135
  TABLE 128
  wait timeout 15
lock statistics 99
LOCKLMT 135
log 214
  archive 214
LOGDETL report 189
Logical Unit of Work 214
loops 111
LPAR 106, 215
LRB 8, 135, 218
LTIMEOUT 10, 15, 135
LUW 128, 214
  close 127

LUW *(continued)*
COMMIT WORK 129
keep small 129
statistics 172
LUWSTATS 172

M

managing data 108
MAPPING 138
maximum buffers 17
MDC 23, 215
minidisk caching 23
MINIOPT 23
monitoring 15
MUM 215

N

naming conventions 108
NCUSERS 135
NDIRBUF 135
NLRBS 8, 135, 215
NLRBU 8, 135, 215
NOFASTTR 138
normalization 142
NOSQLCA 128
NPACKAGE 136
NPACKPCT 136
NPAGBUF 135
NULL 126, 128, 215
NVS 23

O

object
definition into production 112
enterprise rules 109
naming conventions 108
standards 112
offline 215
online 215
operator commands 45
COUNTER * 35
COUNTER INTERNAL 140
COUNTER POOL 139
SET initparm 9
SET TARGETWS 140
SHOW ACTIVE 35
SHOW CONNECT 35
SHOW DBEXTENT 35
SHOW DBSPACE 35
SHOW LOCK 35
SHOW LOG 35
SHOW POOL 139
SHOW TARGETWS 140
optimize application maintenance 110
optimizer 132
optimizer choice 105
ORDER ASC or DSC 132
ORDER BY 127, 128
OVERVIEW screen 155

P

PACKAGED report 173
packages privileges 113
PACKAGES report 175
page buffer increase 17
PAGENTS screen 156
PAGEREAD 12
paging 23
PAGWRITE 12
parameters 10
parent table 215
partition priority 106
path to data in test 105
PCT 57, 114, 215
PCT changes to restrict transaction execution 114
PCTFREE 132, 133
PERFORM report 191
PERFORM screen 157
performance check 125
application 126, 134
database 129
database setup 133
index 131
initialization parameters 134
MAPPING 138
saved segments 138
SAVEINTV 140
system activity 137
table definition 112
user questions 125
VM database 138
VMDSS 138
VSE & VM 126
performance consideration 111
performance improvements
applications 126
buffers 18
caching 23
changing application source 112
checkpoint 13, 14
DASD balancing 130
data types 142
database administration 129
database setup 133
design 141
increased buffer maximums 17
indexes 131
lock wait timeout 15
LTIMEOUT 15
on VM 138
on VSE 138
physical design 143
reformulating queries 111

performance improvements *(continued)*
startup parameters 134, 136
system 137
termination statistics 15
uncommitted read 5
VMDSS 138
VSE Extended User Buffering 5
performance parameters 10
physical design 143
PIPE SQL 123
pool 215
pool limits 17
POOLS report 192
POOLS screen 158
PPT 57, 215
predicates
indexable 127
sargable 127
with NULL 128
preprocessing 6, 108
preprocessing ISOL 130
primary key 215
priority setting for VSE partitions 106
production environment
CEDF usage 111
comparing to test 106
compilations 106
defining objects 113
problems 111
running jobs 106
without ISQL 130
pseudo-conversational 129
PSP 215
PUBLIC 113

Q

QMF 123, 215
query reformulation 111

R

R/O 215
R/W 215
RAGENTS screen 159
RAMAC 23
RDIIN 166
RDO 113, 215
read uncommitted data 5
real agent states 166
recover 215
recovery 9, 44
recovery parameters 10
referential constraint 215
referential integrity 112, 215
reformulating SQL statements 111, 112
reload 215
RELOAD COMMITCOUNT 130
reload dbspace 33
reorganize dbspace 33
repeatable read 6, 215
replicating system 105
request parameter list 215
residence priority 139
response slow 125
restore 215
restrict transaction 114
retention period 215
REXX interface 100
REXX SQL 123
RMODE(ANY) 110
rollback 215
routines 128
RPL 215
RR 6, 215

S

saved segments 138
SAVEGNRL 140
SAVEINTV 140
SCIF 43, 215
SECTIONS report 176
SECTSTAT report 177
SELECT 215
SELECT unused columns 126, 128
SELECT WITH UR 5
SEPINTDB 139
service parameters 10
SESSIOND report 177
SESSIONS report 179
SET initparm 9
set isolation 6
SET TARGETWS 140
SHOW command (see also operator command)
ACTIVE 35
CONNECT 35
DBEXTENT 35
DBSPACE 35
INITPARM 11
INVALID 131
LOCK 35
LOG 35
POOL 139
TARGETWS 139
shutdown counter information 13
simulating workload 111
simulation test 109
SIT 57, 58, 114
slow response 125
snapshot 23
source change for performance 112
SP 109
splitting data 111
splitting tables 111
SPLR 9, 219

SQL
  Master 215
SQL-Tune 53, 59, 111, 112, 216
  installation on VM 59
  installatoin on VSE 53
  internet address 53
SQLCIREO 130
SQLDBA sqlid 113
SQLDBA userid 113
SQLDBSU 214, 215
SQLLOG HISTORY screen 160
SQLLOG report 194
SQLLOG screen 160
SQLPREP 6
SQLSTATE 166
stage 23
standards for objects 112
startup parameters (see initialization)
statements analyzing 111
static partitions 106
statistics 105, 106
storage pool 9, 216
storage statistics 98
stored queries 128
subroutines 110
SUM 46, 216
Super exclusive mode 7
SYSIN 216
SYSINDEXES 131
SYSPRINT 216
SYSSTATS report 195
SYSSTATS screen 161
system
  activity check 137
  duplication 105
  replication 105
system programmer 109
SYSTEM.SYSINDEXES 131
SYSTEM.SYSUSAGE 124

T

table definition 112
table reorganization 47
table splitting 111
table statistics 100
TABLES report 181
TABNDXD report 199
TABNDXS report 180, 181
TABNDXU report 180
TARGETWS 139
TCLASS 110, 113
TCLASS change 113
test environment 105, 109
  application testing 111
  comparing to production 106
  compilations 106
  different libraries 106
  extracting subset from production 105
test environment *(continued)*
  generating another VSE system 106
  operations 106
  running jobs 106
  statistics 105, 106
  testing integrity of data 111
THRESHOLD report 196
TOPUSERS report 197
TOPUSERS screen 162
transaction isolation 110, 113
TRANSTAT report 200
TS 108
tune application 105, 115, 126
tune database 105, 115, 129
tuning checklist 126

U

uncommitted read 5, 7, 129, 216
UNION 127
unit of work 216
unload 216
unload dbspace 33
unused columns in SELECT 126, 128
update 216
update statistics 106, 130
UPDATE using index 126
UR 5, 7, 129, 216
user
  archive 216
  buffering 5
  complaint 125
  questions 125
USER screen 163
USERRES report 181
USERTIME report 182

V

views 123
violating 31-bit addressing rules 110
virtual CPU 166
virtual disk 23
VM 216
VM:DB/Monitor
  components 90
  CP directory changes 87
  database changes 87
  DBMDET utility 96
  DBMON INITIAL file 88
  directory samples 86
  features 86
  installation 86
  internet address 85
  password 86
  RVMINST command 87
  RVMMAINT VM user ID 87
  see also DBMAP, DBMDET, IDBM

VM:DBA 85
VMDSS 138, 216
  checklist 138
  CHKINTVL 140
  COUNTER POOL 139
  MAPPING 138
  residence priority 139
  SAVEINT 19
  SAVEINTV 140
  SEPINTDB 139
  TARGETWS 139
VMPRF 216
VSAM 216
VSE 216
  partitions usage 106
VSE Extended User Buffering 5

W

WAITLOCK 182
workload simulation 111
WWW
  IBM redbooks - see last appendix
  Sterling 85

X

X lock mode 7
XDBMAP module 100
XDBMAP REXX function 100
XTS4CRE 60
XTS4GMOD 60
XTS4PREP 60
XTS4SEG 60

Z

Z lock mode 7

ITSO Redbook Evaluation

DB2 Performance Tuning on VSE and VM
SG24-5146-00

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

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

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

Overall Satisfaction _____

Please answer the following questions:

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

If no, please explain:

_____

_____

_____

_____

What other redbooks would you like to see published?

_____

_____

_____

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

_____

_____

_____

_____

_____

SG24-5146-00
Printed in the U.S.A.

Artwork Definitions

id	File	Page	References
WOLOGO	5146SU	i	
WOLOGOS	5146SU	i	
TILOGO	5146SU	i	
TILOGOS	5146SU	i	

Table Definitions

id	File	Page	References
INIPARM	5146VARS	i	14
CCSUM	5146VARS	i	42
R1	REDB$EVA	229	229, 229
R2	REDB$EVA	229	229

Figures

id	File	Page	References
FSELCOM	5146CH20	6	1
F3440	5146CH20	9	2
F3445	5146CH20	10	3
F3420	5146CH20	11	4
F3425	5146CH20	11	5
F3352	5146CH20	12	6
F3353	5146CH20	12	7
F261	5146CH20	16	8 16, 19
F221	5146CH20	21	9 19
CH4F1	5146CH30	25	10
REF001	5146CH40	36	11 35, 37
REF002	5146CH40	37	14 37
CCMAIN	5146CH40	42	15
CH3F1	5146CH40	48	16 47, 112, 134
QEXT	5146CH40	50	17
SCREEN0	5146CH50	54	18
SCREEN1	5146CH50	56	19
SCREEN2	5146CH50	56	20
SCREEN3	5146CH50	57	21
SCREEN4	5146CH50	57	22
SCREEN5	5146CH50	57	23
SCREEN6	5146CH50	58	24

SCREEN7	5146CH50	58	25	
SCREEN8	5146CH50	58	26	
SCRNVM0	5146CH50	60	27	
QNSSMAP	5146CH50	61	28	
VMFSTUP	5146CH50	61	29	
CH6F71	5146CH50	62	30	
CH6F72	5146CH50	62	31	
XTS1MAI	5146CH50	64	32	
XTS4MVM	5146CH50	65	33	
PSEG	5146CH50	66	34	66
ONLINES	5146CH50	67	35	123
VERIF	5146CH50	68	36	
SQLTUNE	5146CH50	68	37	
TUNEID	5146CH50	69	38	
SQLPATH	5146CH50	69	39	
TUNEHOS	5146CH50	70	40	
TUNNEXE	5146CH50	71	41	
TUNEEXE	5146CH50	71	42	70
TUNEDIS	5146CH50	72	43	70
TUNEZOO	5146CH50	72	44	70
DBSCAN	5146CH50	73	45	
DBSPACE	5146CH50	74	46	78
TABLES	5146CH50	75	47	78
TABSEL	5146CH50	76	48	
COLUMNS	5146CH50	76	49	69
INDEXES	5146CH50	77	50	69
XTSMONA	5146CH50	78	51	
MONCICS	5146CH50	79	52	82
DETCICS	5146CH50	80	53	
TUNCICS	5146CH50	81	54	
BATCHM	5146CH50	81	55	
DETBATC	5146CH50	82	56	
STATVM	5146CH50	83	57	
DBM71	5146CH60	88	58	
DBM72	5146CH60	89	59	
ACTDSP	5146CH60	90	60	
CH7001	5146CH60	91	61	
THRES1	5146CH60			

		92	62	92
THRES2	5146CH60	92	63	
THRES3	5146CH60	93	64	
THRES4	5146CH60	93	65	
THRES5	5146CH60	94	66	
REXX	5146CH60	101	67	
SELCAT	5146CH70	107	68	106
UPDCAT	5146CH70	108	69	106
SCREN0	5146CH70	113	70	
SCREN1	5146CH70	113	71	
SCREN2	5146CH70	114	72	
SCREN3	5146CH70	114	73	
CH9F3	5146CH80	115	74	
CH9F4	5146CH80	116	75	
CH9F5	5146CH80	116	76	
CH9F9	5146CH80	117	77	
CH9F10	5146CH80	118	78	
CH9FA	5146CH80	119	79	
CH9FB	5146CH80	119	80	
CH9FC	5146CH80	119	81	
CH9F31	5146CH80	120	82	
CH9F11	5146CH80	121	83	
CH9F13	5146CH80	121	84	
CH9F12	5146CH80	121	85	
CH9F32	5146CH80	122	86	
CH9C1	5146CH80	122	87	
CH9F1	5146CH80	123	88	
CH9F2	5146CH80	124	89	124
SCREN4	5146CH90	131	90	
C09F05E	5146CH90	143	91	143
IDBMSTA	5146AX20	147	92	
CH7801	5146AX20	147	93	
IDBMMAI	5146AX20	148	94	
IDBMBU	5146AX20	149	95	
IDBMCO	5146AX20	150	96	
IDBMDA	5146AX20	151	97	
IDBMDB	5146AX20	152	98	
IDBMDS	5146AX20	153	99	
IDBMEX	5146AX20	154	100	
IDBMOV	5146AX20	155	101	
IDBMPA	5146AX20			

		156	102	
IDBMPE	5146AX20			
		157	103	
IDBMPO	5146AX20			
		158	104	
IDBMRA	5146AX20			
		159	105	
IDBMSQ	5146AX20			
		160	106	
IDBMSH	5146AX20			
		160	107	
IDBMSY	5146AX20			
		161	108	
IDBMTO	5146AX20			
		162	109	
IDBMUS	5146AX20			
		163	110	
IDBMAW	5146AX20			
		164	111	
DBM781	5146AX20			
		165	112	
				164
RECDETL	5146AX20			
		168	113	
ACTVTI	5146AX20			
		169	114	
DEADLK	5146AX20			
		170	115	
DYNSQL	5146AX20			
		171	116	
				123
LUWSTAT	5146AX20			
		172	117	
PACAGD	5146AX20			
		174	118	
				119
PACAGS	5146AX20			
		175	119	
				115, 118
SECTION	5146AX20			
		176	120	
				115, 118
SECTST	5146AX20			
		177	121	
SESSID	5146AX20			
		178	122	
				119
SESSIS	5146AX20			
		179	123	
TABNDS	5146AX20			
		180	124	
				118
TABNDU	5146AX20			
		180	125	
				118
TABLES7	5146AX20			
		181	126	
USERRES	5146AX20			
		181	127	
USERTIM	5146AX20			
		182	128	
WAITLOC	5146AX20			
		182	129	
LOGBUF	5146AX20			
		184	130	
COUNTER	5146AX20			
		185	131	
DBSTAT	5146AX20			
		187	132	
DSCNTR	5146AX20			
		188	133	
EXTENT	5146AX20			
		189	134	
LOGDETL	5146AX20			
		190	135	
PERFORM	5146AX20			
		191	136	
POOLS	5146AX20			
		192	137	
SQLLOG	5146AX20			
		194	138	
SYSSTA	5146AX20			
		195	139	
THRESH	5146AX20			
		196	140	

TOPUSER	5146AX20	197	141
DETDETA	5146AX20	198	142
RECTABN	5146AX20	199	143
DETTRAN	5146AX20	201	144

Headings

id	File	Page	References
REDBCOM	REDB$COM	xiv	Comments Welcome
CH1000	5146CH10	1	Chapter 1, Introduction
PARTA	5146CH20	3	Part 1, Enhancements
CH2000	5146CH20	5	Chapter 2, Utilizing the New Functions 1
HD2100	5146CH20	5	2.1, VSE Extended User Buffering
HD2300	5146CH20	5	2.2, Isolation Level Uncommitted Read
HD2110	5146CH20	5	2.2.1, Invocation
HD2120	5146CH20	7	2.2.2, Usage 129
HD2130	5146CH20	7	2.2.3, Locking
HD2150	5146CH20	8	2.2.4, Storage Usage
HD2160	5146CH20	8	2.2.5, Performance Implications
ARCH	5146CH20	9	2.3, Faster Archiving
HD2400	5146CH20	9	2.4, SET Initialization Parameters 11
HD3440	5146CH20	9	2.4.1, Usage
CHPST	5146CH20	13	2.6.3, Checkpoint Statistics 136
COSTR	5146CH20	14	2.6.3.2, Considerations
ROSTR	5146CH20	15	2.6.3.3, Recommendation
DSPSTAT	5146CH20	15	2.6.4, Termination Statistics 19
HD2600	5146CH20	15	2.7, Lock Wait Timeout
HD2200	5146CH20	17	2.8, Increased Buffer Maximums
IBMMT	5146CH20	19	2.8.2, Measurement and Tuning 18, 135
CH4000	5146CH30	23	Chapter 3, Disks and Caching 1
CH4100	5146CH30	23	3.1, Minidisk Placement on RAMAC and Multiprise Internal Disk
CH4200	5146CH30	23	3.2, Caching 23
PARTB	5146CH40	31	Part 2, Tools 125
CH5000	5146CH40	33	Chapter 4, Control Center Feature 1
HD5200	5146CH40	33	4.2, Control Center for VSE
DBSANA	5146CH40	37	4.2.3, Dbspace Analysis Tool 134
CH513	5146CH40	44	Database Monitoring

CH6000	5146CH50
CH6100	5146CH50
CH61VM	5146CH50
APPLDEV	5146CH50
CRITDB	5146CH50
CRITTAB	5146CH50
DBMNVSE	5146CH50
CH7000	5146CH60
CH7100	5146CH60
CH7300	5146CH60
CH7200	5146CH60
CH7REP	5146CH60
PARTC	5146CH70
CH8000	5146CH70
TSTENV	5146CH70
CH80ST	5146CH70
HD8100	5146CH70
RESTRI	5146CH70
CH9000	5146CH80
CH9400	5146CH80
CH9100	5146CH80
CH9200	5146CH80
9201	5146CH80
CH9300	5146CH80
CH3000	5146CH90
MOFRERR	5146CH90
GENGUI	5146CH90
CHECDBA	5146CH90
INCAVAI	5146CH90
USEXAM1	5146CH90
NORM	5146CH90
AX10	5146AX10
AX23	5146AX10
AX20	5146AX20

53 Chapter 5, SQL-Tune
1, 111

53 5.1, Installation of SQL-Tune on VSE

59 5.2, Installation of SQL-Tune on VM

66 5.3.2.1, Application Development Utility
81, 83

78 List of Critical Dbspaces (PF8)
74, 119

78 List of Critical Tables (PF9)
76, 119

78 5.3.2.3, Database Monitoring Utility on VSE
82

85 Chapter 6, VM:DB/Monitor
2

86 6.1, Features and Benefits

86 6.2, Installation of VM:DB/Monitor

90 6.3, Components of VM:DB/Monitor

95 6.3.4.1, Reports

103 Part 3, DB2 Tuning

105 Chapter 7, Application Development
2

105 7.1, Test Databases
111

106 7.1.1.2, Statistics
105

109 7.3, Application Development Steps

113 7.3.8, Isolating CICS Applications in a VSE Environment
110

115 Chapter 8, Application Tuning
2

115 8.1, Tuning in General

118 8.2, Tuning Your Current Applications

122 8.3, How to Tune without Source Code

123 8.3.1, Views on Read-Only Tables

123 8.4, Tuning Dynamic Statements
117, 123

125 Chapter 9, Checklists and Recommendations
2

126 9.3.1.1, Most Frequent Errors

127 9.3.1.3, Recommendations
110, 111

129 Recommendations
113

136 Increase Availability with SET CHKINTVL
137, 137

137 9.3.2.6, User Example: Shorten Database Maintenance

142 9.6.3, Normalization
128, 143, 213, 213

145 Appendix A, Installed Environment
2

145 VSE Settings

		147	Appendix B, Screens and Reports of VM:DB/Monitor 2, 89, 118
AX2010	5146AX20		
		147	B.1, IDBM Sample Screens 95
AX2020	5146AX20		
		166	B.2, DBMAP Activity Reports 96
AX2030	5146AX20		
		183	B.3, DBMAP Monitor Reports 96
AX2040	5146AX20		
		198	B.4, DBMAP Detail Reports 97
NOTICES	SG245146 SCRIPT		
		203	Appendix C, Special Notices ii
BIBL	5146BIBL		
		207	Appendix D, Related Publications
REDBCDR	REDB$BIB		
		207	D.2, Redbooks on CD-ROMs
ORDER	REDB$ORD		
		209	How to Get ITSO Redbooks 207
REDBIBM	REDB$ORD		
		209	How IBM Employees Can Get ITSO Redbooks
REDBCUS	REDB$ORD		
		210	How Customers Can Get ITSO Redbooks
REDBFOR	REDB$ORD		
		211	IBM Redbook Order Form
REDBEVA	REDB$EVA		
		223	ITSO Redbook Evaluation xiv

Index Entries

id	File	Page	References
APPL	5146VARS		
		i	(1) application 99, 105, 108, 108, 108, 109, 110, 110, 110, 111, 113, 115, 126
ARC	5146VARS		
		i	(1) archive
CATIND	5146VARS		
		i	(1) catalog 105, 113, 130
CC	5146VARS		
		i	(1) Control Center 33, 33, 33, 33, 33, 33, 33, 34, 34, 34, 34, 34, 34, 35, 35, 35, 35, 35, 35, 35, 36, 37, 37, 38, 39, 39, 39, 39, 39, 40, 40, 42, 43, 43, 44, 44, 44, 45, 45, 45, 45, 45, 46, 46, 46, 46, 46, 47, 47, 47, 47, 48, 48, 49, 49, 49, 49, 49, 50, 50, 50, 50, 51, 51
CHKIN	5146VARS		
		i	(1) checkpoint 12, 14, 14, 14, 15, 18
CICSIND	5146VARS		
		i	(1) CICS 106, 108, 110, 110, 113, 113
COL	5146VARS		
		i	(1) column 112, 112, 126, 128
CNTR	5146VARS		
		i	(1) COUNTER 135, 135
DAT	5146VARS		
		i	(1) data 5, 7, 126, 142
DSP	5146VARS		
		i	(1) data space 138
DBIND	5146VARS		
		i	(1) database 44, 45, 105, 110, 112, 115, 129, 133, 213, 213
DBSU	5146VARS		
		i	(1) database service utility (see SQLDBSU)
DRFI	5146VARS		
		i	(1) Data Restore Feature
DBSP	5146VARS		
		i	(1) dbspace 37, 38, 46, 47, 100, 111, 116
DBM	5146VARS		

		i	(1) DB/Monitor (see VM:DB/Monitor)
DBMON	5146VARS		
		i	(1) VM:DB/Monitor 85, 86, 86, 86, 86, 87, 87, 87, 87, 88, 90, 96
EMAIL	5146VARS		
		i	(1) email - see internet address
IBMIND	5146VARS		
		i	(1) IBM
IDBM	5146VARS		
		i	(1) IDBM 94, 147, 147, 147, 164
INET	5146VARS		
		i	(1) internet address 53, 53, 85
DBMAP	5146VARS		
		i	(1) DBMAP 88, 89, 89, 89, 89, 89, 89, 95, 95, 96, 96, 96, 96, 97, 97, 97, 97, 98, 98, 99, 99, 99, 99, 100, 100, 166, 171, 173, 175, 176, 177, 177, 177, 177, 177, 178, 178, 178, 179, 180, 180, 180, 180, 181, 181, 181, 181, 182, 182, 183, 184, 185, 186, 188, 189, 189, 191, 194, 195
DBMDET	5146VARS		
		i	(1) DBMDET 97
DBMONVM	5146VARS		
		i	(1) DBMONVM 87, 88, 89, 89, 89, 89
DBMONUP	5146VARS		
		i	(1) DBMONUP 87, 102
INDIND	5146VARS		
		i	(1) index 48, 117, 122, 126, 131
STRTPAR	5146VARS		
		i	(1) initialization parameters 8, 8, 10, 10, 10, 10, 11, 13, 15, 15, 19, 135, 135, 135, 135, 135, 135, 136, 136, 136, 136, 138, 139, 139, 140, 140, 215, 215
INTEG	5146VARS		
		i	(1) integrity 7, 111, 112
JO	5146VARS		
		i	(1) jobs 106, 112
LOCKIND	5146VARS		
		i	(1) lock 7, 8, 15, 113, 128, 135
LOGIND	5146VARS		
		i	(1) log 214
LUW	5146VARS		
		i	(1) LUW 127, 129, 129, 172
OBJ	5146VARS		
		i	(1) object 108, 109, 112, 112
OPER	5146VARS		
		i	(1) operator commands 9, 35, 35, 35, 35, 35, 35, 35, 139, 139, 140, 140, 140
PCHK	5146VARS		
		i	(1) performance check 112, 125, 126, 126, 129, 131, 133, 134, 134, 137, 138, 138, 138, 138, 140
PERFIMP	5146VARS		
		i	(1) performance improvements 5, 5, 13, 14, 15, 15, 15, 17, 18, 23, 111, 112, 126, 129, 130, 131, 133, 134, 136, 137, 138, 138, 138, 141, 142, 143
PRED	5146VARS		
		i	(1) predicates 127, 127, 128
PRODENV	5146VARS		
		i	(1) production environment 106, 106, 106, 111, 111, 113, 130
RDOIND	5146VARS		
		i	(1) RDO
RELIND	5146VARS		
		i	(1) reload
SHOWIND	5146VARS		
		i	(1) SHOW command (see also operator command) 11, 35, 35, 35, 35, 35, 35, 131, 139, 139
SQLIND	5146VARS		
		i	(1) SQL 215
SQLT	5146VARS		
		i	(1) SQL-Tune

			53, 53, 59
INITPAR	5146VARS		
		i	(1) startup parameters (see initialization)
STORIND	5146VARS		
		i	(1) storage pool
SYS	5146VARS		
		i	(1) system 105, 105, 137
TESTENV	5146VARS		
		i	(1) test environment 105, 105, 106, 106, 106, 106, 106, 106, 106, 106, 111, 111
USERIND	5146VARS		
		i	(1) user 5, 125, 125, 216
VMIND	5146VARS		
		i	(1) VM
VMDSS	5146VARS		
		i	(1) VMDSS 19, 138, 138, 139, 139, 139, 139, 140, 140
VSEIND	5146VARS		
		i	(1) VSE 106
VSAMIND	5146VARS		
		i	(1) VSAM
WWW	5146VARS		
		i	(1) WWW 85
SETTI	5146CH20		
		14	(1) default setting

List Items

id	File	Page	References
MONITOR	5146CH20		
		20	4 20
STEP3	5146CH50		
		68	4 70
STEP8	5146CH50		
		70	9 70
TWODIF	5146CH70		
		105	2 105
TWO	5146CH70		
		105	4 106, 106
SPC	5146CH70		
		110	9
SORT	5146CH90		
		127	5 126
CONTENT	5146CH90		
		129	19 135
BALANCE	5146CH90		
		130	11 131

Processing Options

Runtime values:

Document fileid	SG245146 SCRIPT
Document type	USERDOC
Document style	REDBOOK
Profile	EDFPRF40
Service Level	0022
SCRIPT/VS Release	4.0.0
Date	98.05.06
Time	05:35:12
Device	3820A
Number of Passes	4
Index	YES
SYSVAR D	YES
SYSVAR G	INLINE
SYSVAR X	YES

Formatting values used:

Annotation	NO
Cross reference listing	YES
Cross reference head prefix only	NO
Dialog	LABEL
Duplex	YES
DVCF conditions file	(none)
DVCF value 1	(none)
DVCF value 2	(none)
DVCF value 3	(none)
DVCF value 4	(none)
DVCF value 5	(none)
DVCF value 6	(none)
DVCF value 7	(none)
DVCF value 8	(none)
DVCF value 9	(none)
Explode	NO
Figure list on new page	YES
Figure/table number separation	YES
Folio-by-chapter	NO
Head 0 body text	Part
Head 1 body text	Chapter
Head 1 appendix text	Appendix
Hyphenation	NO
Justification	NO
Language	ENGL
Keyboard	395
Layout	OFF
Leader dots	YES
Master index	(none)
Partial TOC (maximum level)	4
Partial TOC (new page after)	INLINE
Print example id's	NO
Print cross reference page numbers	YES
Process value	(none)
Punctuation move characters	.,
Read cross-reference file	(none)
Running heading/footing rule	NONE
Show index entries	NO
Table of Contents (maximum level)	3
Table list on new page	YES
Title page (draft) alignment	RIGHT
Write cross-reference file	(none)

Imbed Trace

Page 0	5146SU
Page 0	5146VARS
Page 0	REDB$BOE
Page i	REDB$ED1
Page i	5146EDNO
Page i	REDB$ED2
Page xiii	5146ABST
Page xiii	5146ACKS
Page xiii	REDB$COM
Page xiv	5146IMBD
Page xiv	5146CH10
Page 2	5146CH20
Page 21	5146CH30
Page 29	5146CH40
Page 51	5146CH50
Page 83	5146CH60
Page 102	5146CH70
Page 114	5146CH80
Page 124	5146CH90
Page 144	5146AX10
Page 145	5146AX20
Page 203	5146SPEC
Page 203	REDB$SPE
Page 204	5146TMKS
Page 205	5146BIBL
Page 207	REDB$BIB
Page 208	REDB$ORD
Page 211	5146GLOS
Page 216	5146ABRV
Page 228	REDB$EVA