CPExpert™

DB2 Component

Computer Management Sciences, Inc
6076-D Franconia Road
Alexandria, Virginia 22310-1756
voice: (703) 922-7027
fax: (703) 922-7305
www.cpexpert.com

CPExpert is a trademark of Computer Management Sciences, Inc.

This manual applies to the DB2 Component of **CPExpert**[tm], a proprietary software product of Computer Management Sciences, Inc., Alexandria, Virginia, United States of America.

The information in this document is subject to change. Comments or suggestions are welcome, and to the extent practicable, will be incorporated in revisions to this document. Please send all comments, suggested new rules, suggested changes to existing rules, suggestions for improvement to the software or documentation, or any other advice to:

Computer Management Sciences, Inc.
6076-D Franconia Road
Alexandria, Virginia 22310-1756
(703) 922-7027 FAX: (703) 922-7305
www.cpexpert.com

DISCLAIMER

The advice, recommendations, or otherwise contained in this document represent information generally available in the public domain, as contained in vendor manuals, published in articles or papers, presented at professional conferences, or otherwise commonly accepted in the professional community. Neither Computer Management Sciences, Inc. nor its representatives make representations or warranties with respect to the applicability or application of any advice, recommendations, or otherwise, contained in this document or in any results from applying the CPExpert software, to any particular computer system or computer installation.

TRADEMARKS

CPExpert is a trademark of Computer Management Sciences, Inc. IBM, MVS/370, MVS/SP, MVS/XA, MVS/ESA, Enterprise System/3090, Netview, PR/SM, Processor Resource/System Manager, Hiperspace, and ES/9000 are trademarks of the IBM Corporation. SAS, SAS/OR, and SAS/STAT are trademarks of the SAS Institute Inc. MXG is a trademark of Merrill Consultants. MICS is a trademark of Legent Corporation.

COPYRIGHT INFORMATION

©Copyright 1999, Computer Management Sciences, Inc.
All rights reserved. Printed in the United States of America.

This licensed work is confidential and propriety, and is the property of Computer Management Sciences, Inc. No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic, mechanical, photocopying, or otherwise, without the specific written authorization of Computer Management Sciences, Inc.

How to use this manual

This document describes how to use the DB2 Component of CPExpert to analyze system constraints to improved performance of your DB2 subsystems. The manual is organized into one volume, consisting of four sections and one appendix.

Section 1 provides an introduction to the DB2 Component. This section is organized into four chapters. Most of this section can be reviewed for general information.

Chapter 1 provides a brief overview of IBM's DB2.

Chapter 2 provides an overview of the DB2 Component of CPExpert.

Chapter 3 describes the sources of data used by the DB2 Component to analyze performance of your system.

Chapter 4 describes the performance data bases which CPExpert can use to analyze performance.

Section 2 describes installation considerations with the DB2 Component. This section will be important when you initially install the DB2 Component.

Section 3 describes how to provide guidance variables to the DB2 Component using the prefix.CPEXPERT.SOURCE(DB2GUIDE) PDS member. The instructions in this chapter will be important when you install the DB2 Component and at any time the guidance variables need to be changed. This section is organized into five chapters.

- Chapter 1 describes how to specify data selection and results presentation guidance variables. The instructions in this chapter will be important if you wish to select specific measurement periods for analysis.

- Chapter 2 describes how to specify analysis control variables to guide the DB2 Component in its overall analysis of your DB2 subsystem. The instructions in this chapter will be important if you wish to alter the defaults provided with the DB2 Component.

- Chapter 3 describes analysis guidance variables or thresholds that guide the DB2 Component in its analysis of DB2 subsystem. The instructions in this chapter will be important if you wish to alter the defaults provided with the DB2 Component. **Please note that many of the defaults should be altered for your installation!**

- Chapter 4 describes how to specify analysis control variables to guide the DB2 Component in its overall analysis of individual DB2 buffer pools. The instructions in this chapter will be important if you wish to alter the defaults for specific buffer pools.

- Chapter 5 describes how to specify analysis control variables to guide the DB2 Component in its overall analysis of individual DB2 group buffer pools. The instructions

in this chapter will be important if you wish to alter the defaults for specific group buffer pools in a data sharing environment.

Section 4 describes how to use the DB2 Component to analyze performance. The instructions in this section will be followed each time you execute the DB2 Component. This section is organized into two chapters.

- Chapter 1 provides detailed instructions on executing the DB2CPE Module to analyze the performance of your system. The instructions in this chapter will be important each time that the DB2 Component is executed.

- Chapter 2 contains a checklist for executing the DB2 Component.

Appendix A contains a detailed description of each finding based upon the DB2 Component analyzing performance of your DB2 subsystems. The description presents the findings as “rules” that are keyed to the “rule” summaries that would be produced in the DB2 Component reports. The “rule” description discusses the rationale for the finding, suggests action, and provides detailed references to IBM publications or other documents where additional information may be found.

You may wish to briefly review the possible findings in this appendix to appreciate the problems that are encountered in different installations. However, it is not necessary to read all of the possible findings. It is necessary only to read the “rules” that are identified by the reports produced by the DB2 Component based on an analysis of DB2 performance problems at **your** installation.

Acknowledgements

Computer Management Sciences would like to acknowledge the following individuals. They played a significant role in the concept, design, development, testing, or documentation of CPExpert[1]:

John Ebner, Litton Computer Services

Alan Greenberg, Social Security Administration

Stan Meacham, MCI Corporation

Barry Merrill, Merrill Consultants

Philip Mugglestone, European Software Product Services, NV

Bryant Osborn, Litton Computer Services

John Peterson, MCI Corporation

Bernie Pierce, IBM Corporation

Fred Voth, JOSTENS Corporation

CPExpert would not exist as a product if it had not been for the personal inspiration, professional advice, technical knowledge, and encouragement from these individuals!

Additionally, Computer Management Sciences would like to acknowledge the following individuals. They played a significant role in the concept, design, development, and testing of the DB2 Component of CPExpert:

Bill Hatcher, US Department of Veterans Affairs

Bryant Osborn, Bank of America

[1]The affiliation of the individuals is shown as of the time CPExpert was developed or when the DB2 Component was developed. Some of the individuals are no longer associated with the organizations shown.

Contents

Page

Changes ix

Section 1: Introduction

Chapter 1: Overview of DB2 1-1

Chapter 2: The DB2 Component of CPExpert 1-2

Chapter 3: Data Sources 1-4

Section 2: Installing the DB2 Component

Chapter 1: Installing CPExpert 2-1

Chapter 2: Ensuring that required DB2 information is available 2-1

Chapter 3. Starting DB2 Trace 2-3

Chapter 4. Modifying the guidance variables for the DB2 Component 2-4

Section 3: Specifying guidance variables

Chapter 1: Data Selection Variables 3-2

Chapter 1: Data Selection Variables 3-2
Chapter 1.1: DB2DATES and DB2TIMES variables 3-3
Chapter 1.2: DB2DATEE and DB2TIMEE variables 3-3
Chapter 1.3: SHIFT variable . . . . . 3-5 |
Chapter 1.4: SYSTEM variable . . . . . 3-5 |
Chapter 1.5: SYSTEMn variable(s) . . . . . 3-6 |
Chapter 1.6: DB2SID variable . . . . . 3-6 |
Chapter 1.7: ALL_DB2 variable . . . . . 3-7 |
Chapter 1.8: SAS Output Delivery System . . . . . 3-8 |

Chapter 2: Analysis Control Variables . . . . . 3-9 |
Chapter 2.1: Specifying Production or Test Subsystem . . . . . 3-10 |
Chapter 2.3: Specifying whether DB2 accounting data are available . . . . . 3-10 |
Chapter 2.4: Specifying availability of Trace IFCID 106 records . . . . . 3-10 |

Page

Chapter 2.6: Specifying availability of Trace IFCID 196 records 3-11
Chapter 2.7: Specifying availability of SMF Type 74 (structure) records 3-11
Chapter 2.8: Specifying individual guidance for buffer pools 3-12
Chapter 2.9: Specifying individual guidance for group buffer pools 3-12
Chapter 2.10: Specifying SAS library containing DB2 data 3-12
Chapter 2.11: Specifying data set name for DB2 datasets 3-12

Chapter 3: Analysis Guidance Variables 3-13
Chapter 3: Analysis Guidance Variables 3-13
Chapter 3.1: Turning OFF DB2 Component Rules 3-17
Chapter 3.2: ASYNCSRV variable 3-17
Chapter 3.3: BUFFPGRT - Acceptable page-in rate, all buffer pools 3-18
Chapter 3.4. LOCKCONT variable 3-18
Chapter 3.5: PAGERATE - Acceptable page-in rate, individual pool 3-20
Chapter 3.6: PCTCBA - Percent active buffers 3-20
Chapter 3.7: PCTCTL - Percent CT sections not found 3-21
Chapter 3.8: PCTDBDL - Percent DBD sections not found 3-22
Chapter 3.9: PCTDPF - Percent dynamic sequential prefetch 3-23
Chapter 3.10: PCTEDM- Percent free pages in EDM pool 3-23
Chapter 3.11: PCTESC - Percent lock escalation not effective 3-24
Chapter 3.12: PCTFLAM - Percent failed look-ahead tape mounts 3-24
Chapter 3.13: PCTESC - Percent lock escalation not effective 3-25
Chapter 3.14: PCTHITGB - Percent read hit cross-invalidated pages 3-26
Chapter 3.15: PCTRACT - Percent reads from active log 3-27
Chapter 3.16: PCTRAND - Percent GETPAGE from random requestors 3-27
Chapter 3.17: PCTRSAT - Percent reconnection attempts succeeded 3-28
Chapter 3.18: PCTSEQ - Percent GETPAGE, sequential requestors 3-29
Chapter 3.19: PCTSLOCK - Percent suspends for lock conflict 3-29
Chapter 3.20: PCTXIGBP - Acceptable percent reclaims, XI pages 3-30
Chapter 3.21: Q3STCTHW - Times threads were queued at create 3-31
Chapter 3.22: Q3STMEOM - End-of-memory (non-DB2 allied agent) 3-31
Chapter 3.23: Q3STMEOT - End-of-task (non-DB2 allied agent) 3-31
Chapter 3.24: QBGLAN - CF prefetch reads, no data, no directory 3-32
Chapter 3.25: QBGLAR - CF prefetch, no data, directory entry 3-32
Chapter 3.26: QBGLCN - Times CF castout engine not available 3-33
Chapter 3.27: QBGLRF - Times CF reads failed, lack of storage 3-33
Chapter 3.28: QBGLWF - Times CF writes failed, lack of storage 3-34
Chapter 3.29: QBGLSU - Times CF write engine was not available 3-34
Chapter 3.30: QBSTARF - Unsuccessful HP to VP read, async ADMF 3-34
Chapter 3.31: QBSTAWF - Unsuccessful HP to VP write, async ADMF 3-35
Chapter 3.32: QBSTDMC - Times Data Manager Threshold reached 3-35
Chapter 3.33: QBSTHRF - Unsuccessful HP to VP reads (MVPG) 3-36

Page

Chapter 3.34: QBSTHWF - Unsuccessful HP to VP writes (MVPG) 3-36
Chapter 3.35: QBSTIMW - Number of Immediate Writes 3-37
Chapter 3.36: QBSTJIS - Times prefetch I/O streams denied 3-37
Chapter 3.37: QBSTNGT - Number of unsuccessful GETPAGE 3-38
Chapter 3.38: QBSTPL1 - Times prefetch reduced to ½ of normal 3-38
Chapter 3.39: QBSTPL2 - Times prefetch reduced to 1/4 of normal 3-39
Chapter 3.40: QBSTPQF - Times I/O parallelism was downgraded 3-39
Chapter 3.41: QBSTREE - Times prefetch disabled, no read engine 3-40
Chapter 3.42: QBSTSPD - Times prefetch disabled, no buffers 3-40
Chapter 3.43: QBSTWEE - Times write engine was not available 3-40
Chapter 3.44: QBSTWFD - Times work files denied during sort 3-41
Chapter 3.45: QBSTWFF - Times sort was inefficient, buffer shortage 3-41
Chapter 3.46: QBSTWFR - Number of sort merge passes 3-42
Chapter 3.47: QBSTWIO - Pages written to DASD per write I/O 3-42
Chapter 3.48: QBSTWKPD - Prefetch canceled, prefetch quantity = 0 3-43
Chapter 3.49: QDSTQCRT - Conversations deallocated, ZPARMS limit . . . 3-43
Chapter 3.50: QDSTQDBT - DB2 access queued, max remote threads 3-44
Chapter 3.51: QISEFAIL - Times failures caused by full EDM pool 3-44
Chapter 3.52: QISTCOLS - Times invalid SELECT procedure 3-45
Chapter 3.53: QISTRLLM - Times RID terminated, over calculated limit 3-45
Chapter 3.54: QISTRMAX - Times RID terminated, maximum storage 3-46
Chapter 3.55: QISTRPLM - Times RID terminated, over physical limit 3-46
Chapter 3.56: QISTRSTG - Times RID terminated, insufficient storage 3-47
Chapter 3.57: QJSTALR - Number of archive log read allocations 3-47
Chapter 3.58: QJSTALW - Number of archive log write allocations 3-47
Chapter 3.59: QJSTRARH - Number of reads from archive log 3-48
Chapter 3.60: QJSTTVC - Number of reads delayed, tape volume 3-48
Chapter 3.61: QJSTWTB - Number of waits for output log buffer 3-49
Chapter 3.62: QJSTWUR - Number of read delays, resources 3-49
Chapter 3.63: QLSTCNVQ - Conversation requests queued by DDF 3-49
Chapter 3.64: QLSTINDT - Threads indoubt with remote coordinator 3-50
Chapter 3.65: QTXADEA - Number of deadlocks 3-50
Chapter 3.66: QTXALES - Number of locks escalated to shared mode 3-51
Chapter 3.67: QTXALEX - Number of locks escalated to exclusive 3-51
Chapter 3.68: QTXATIM - Number of locks that timed-out 3-51
Chapter 3.69: QWSBSBUF - Buffer overrun errors, SMF destination 3-52
Chapter 3.70: QXACALLAB - Number of stored procedure errors 3-52
Chapter 3.71: QXACALLRJ - Number of SQL calls rejected 3-53
Chapter 3.72: QXACALLTO - Number of times SQL calls timed out 3-53
Chapter 3.73: QXCOORNO - Parallel groups executed on a single DB2 3-53
Chapter 3.74: QXDEGBUF- Parallel groups fell back, buffers 3-54
Chapter 3.75: QXDEGENC - Parallel groups executed in sequential 3-54

Page

Chapter 3.76: QXISORR - Parallel groups executed on single DB2 3-55
Chapter 3.77: QXREDGRP - Times parallel groups reduced, buffers 3-55
Chapter 3.78: QXSTDEXP - Prepared statement exceeded MAXKEEPD . . . 3-56
Chapter 3.79: QZZCSKIP - Parallelism coordinator bypassed, buffers 3-56
Chapter 3.80: SYNCSRV variable and LOCKSRV 3-56
Chapter 3.81: SYNCCHG variable 3-58

Chapter 4: Specifying guidance for individual DB2 buffer pools 3-58
Chapter 4.1: Specify BPGUIDE guidance variable 3-59
Chapter 4.2: Identify DB2 buffer pools having unique guidance 3-60
Chapter 4.3: Place unique guidance in USOURCE(buffer pool) member 3-60
Chapter 4.4: Restrictions 3-60

Chapter 5: Specifying guidance for individual DB2 group buffer pools 3-61
Chapter 5.1: Specify GBPGUIDE guidance variable 3-61
Chapter 5.2: Identify DB2 group buffer pools having unique guidance 3-62
Chapter 5.3: Place unique guidance in USOURCE(buffer pool) member 3-62
Chapter 5.4: Restrictions 3-62

Chapter 6: Specifying guidance for individual DB2 subsystems 3-64
Chapter 6.1: Specify DB2GUIDE guidance variable 3-65
Chapter 6.2: Identify DB2 subsystems having unique guidance 3-65
Chapter 6.3: Create partitioned data set(s) 3-65
Chapter 6.4: Place unique guidance in partitioned data set(s) 3-66
Chapter 6.5: Create Job Control Language DD statement(s) 3-67
Chapter 6.6: Restrictions 3-67

Section 4: Using the DB2 Component

Chapter 1: Executing the DB2CPE Module 4-1
Step 1. Use TSO ISPF to change the "prefix" in the data set names 4-1
Step 2: Make any appropriate changes to the DB2GUIDE Module 4-2
Step 3: Review the output from the DB2CPE Module 4-2

Chapter 2: Checklist 4-4

Appendix A: Description of Rules

Page

Exhibits

3-1 Sample Display of prefix.CPEXPERT.USOURCE(DB2GUIDE) 3-2
3-2 Sample Display of prefix.CPEXPERT.USOURCE(DB2GUIDE) 3-10
3-3 Sample Display of prefix.CPEXPERT.USOURCE(DB2GUIDE) 3-14
4-1 Job Control Language to execute the DB2CPE Module 4-2

©Copyright 1994, Computer Management Sciences, Inc.

Changes

CPExpert Release 13.1:

The main changes to the DB2 Component for CPExpert Release 13.1 are to:

- Add the ability to select up to 10 systems individually for analysis. Until Release 13.1, a user had the options of analyzing data for all systems in the performance data based, analyzing data for a specific sysplex (in case the performance data base contained data for more than one sysplex), or analyzing data for a specific system in the performance data base. With Release 13.1, up to 10 systems can be individually selected for analysis.

- Enhance the options provided with the SAS Output Delivery System (ODS). With Release 13.1, users who exercise the SAS ODS feature for creating CPExpert output can optionally create the output as a PDF file, which can be emailed to other users. Additionally, users can optionally specify a STYLE feature for either HTML or PDF output, if they have a preferred STYLE for HTML or PDF output. The optional links that are available with the HTML have been revised; SAS at some user sites did not create the HTML output in the “standard” way, and the CPExpert code that inserted links into the HTML output did not work properly. I have revised the code to place the links into the output as the output is created, rather than attempting to place the links into the final HTML output created by SAS.

- CPExpert now specifies OPTIONS COMPRESS=N; to override any site specification for file compression. Experiments have shown that CPExpert code runs significantly faster (using much less CPU time) if compression has been turned off.

CPExpert Release 12.2:

The main changes to the DB2 Component for CPExpert Release 12.2 are to:

- Add an ALL_DB2 option that allows users with a large number of DB2 subsystems to have the DB2 Component analyze all DB2 subsystems, but produce a summary report from the result. Each unique finding (or Rule) will show all DB2 subsystems to which the finding applies.

- Correct software or documentation errors that have been reported by users.

CPExpert Release 12.1:

The main changes to the DB2 Component for CPExpert Release 12.1 are to correct errors reported by users.

CPExpert Release 11.2:

The main changes to the DB2 Component for CPExpert Release 11.2 are to:

- Add support for the SAS Output Delivery System (ODS) feature, to enable optional web access of CPExpert reports (this new option was suggested by **Harald Seifert** of HUK, Coburg, Germany).
- Add optional links in CPExpert reports (if the SAS ODS feature is invoked), that link rule output to CPExpert documentation for the rules produced.
- Added new group buffer pool variables that have been added to NeuMics. These group buffer pool variables significantly enhance CPExpert's ability to analyze DB2 data sharing problems for sites who use NeuMICS to create their performance data base.
- Added additional guidance to control the analysis of false lock contention.
- Correct software or documentation errors that have been reported by users.
- Update the DB2 Component as appropriate to support z/OS Version 1 Release 2.

CPExpert Release 11.1:

The main changes to the DB2 Component for CPExpert Release 11.1 are to:

- Enhance the discussion of alternatives that can solve DB2 Data Sharing performance problems.
- Provide the capability to analyze multiple DB2 subsystems in a single execution of CPExpert.
- Provide the capability to specify guidance for individual DB2 subsystems (and guidance for associated buffer pools and group buffer pools, if necessary) in a single execution of CPExpert.
- Update the DB2 Component as appropriate to support z/os Version 1 Release 1.
- Correct software or documentation errors that have been reported by users.

CPExpert Release 10.2:

The main changes to the DB2 Component for CPExpert Release 10.2 are to:

- Update the Component to support DB2 Release 6.
- Update the Component to support DB2 interval statistics in a CA-MICS performance data base.
- Add new rules to analyze DB2 Data Sharing performance problems (some of the analysis can be performed only if SMF Type 74 (structure) data is available in the performance data base):
    - Analysis to determine when coupling facility read requests for data in a DB2 group buffer pool could not complete because of coupling facility storage constraints.
    - Analysis to determine when coupling facility write requests to a DB2 group buffer pool failed because of coupling facility storage constraints.
    - Analysis to determine when a low read-hit percentage was experienced for cross invalidated pages in a DB2 group buffer pool.
    - Analysis to determine when a castout engine was not available for castout processing of pages from a DB2 group buffer pool.
    - Analysis to determine when a coupling facility write engine was not available for virtual buffer page writing to a DB2 group buffer pool.
    - Analysis to determine when lock contention was high for a DB2 group buffer pool.
    - Analysis to determine when false lock contention was high for a DB2 group buffer pool.
    - Analysis to determine when service time was high for synchronous requests to a DB2 group buffer pool.
    - Analysis to determine when service time was high for asynchronous requests to a DB2 group buffer pool.
    - Analysis to determine when too many synchronous requests were changed to asynchronous for a DB2 group buffer pool.

  - Analysis to determine when excessive cross-invalidations due to directory reclaims occurred for a DB2 group buffer pool.

- Update all documentation for the DB2 Component to support DB2 Version 6 references.

- Update the DB2 Component as appropriate to support OS/390 Version 2 Release 10.

- Correct software or documentation errors that have been reported by users.

Section 1: Introduction

This section provides an overview of the DB2 Component of CPExpert and describes the sources of data that the DB2 Component can use.

Chapter 1: Background

Since IBM's 1983 introduction of DATABASE 2 (DB2), it has become one of the world's most popular relational data base management systems. Industry sources report that IBM has more than one million licensed copies of DB2 on all supported platforms.

DB2 Version 3 (released in 1993) offered significant enhancements over DB2 Version 2 in the areas of availability, connectivity, performance, and usability.

DB2 Version 4 (released in 1995) supports all functions available in Version 3 and provided enhancements in the areas of client/server processing, availability, performance, and user productivity.

DB2 Version 5 (released in 1997) supports all functions available in DB2 for MVS/ESA Version 4 plus enhancements in the areas of performance, capacity, and availability, client/server and open systems, and user productivity. The increased power of Sysplex query parallelism in DB2 for OS/390 Version 5 allows DB2 to go far beyond DB2 for MVS/ESA Version 4 capabilities; from the ability to split and process a single query within a DB2 subsystem to processing that same query across many different DB2 subsystems in a data sharing group.

With the announcement of DB2 Universal Database Server for OS/390 Version 6 in 1998, IBM celebrated its 15th anniversary of the product and the tenth release of the product since its introduction on the MVS platform. DB2 UDB for OS/390 Version 6 delivers an enhanced relational database server solution, focuses on greater capacity, performance improvements for utilities and queries, easier database management, more powerful network computing, and DB2 family compatibility with new object-oriented capability, triggers, and more built-in functions.

With Version 7 of DB2 Universal Database Server for OS/390 and z/OS, the DB2 Family delivers more scalability and availability for your e-business and business intelligence applications. DB2 for OS/390 and z/OS Version 7 delivers an enhanced relational database server solution for OS/390. This release focuses on greater ease and flexibility in managing data, better reliability, scalability, and availability, and better integration with the DB2 family.

Although DB2 has achieved tremendous acceptance in the marketplace and provides extraordinary features for its users, many users have found that the performance of DB2 can be frustrating and that DB2 can consume significant system resources without proper tuning.

©Copyright 1999, Computer Management Sciences, Inc. Revised: October, 2001

Most sites have discovered that continual monitoring of performance constraints is required in order for DB2 to keep performing at a high level. Unfortunately, many sites do not have individuals with DB2 performance tuning expertise, and users typically find that a significant effort is required to monitor DB2's performance continually.

The DB2 Component of CPExpert was developed to help sites monitor DB2 performance, to identify problems automatically, and to suggest solutions to the problems.

©Copyright 1999, Computer Management Sciences, Inc.

Chapter 2: The DB2 Component of CPExpert

The DB2 Component evaluates the performance of DB2 for OS/390. The DB2 Component automatically applies most of the DB2 analysis techniques documented in IBM's *DB2 for OS/390 Administration Guides* or *DB2 Data Sharing Administration Guides*, supplemented by techniques and guidance contained in IBM Redbooks, by papers presented at various technical conferences, and by IBM's suggestions in the DB2 DSNWMGS macros.

The DB2 Component processes DB2 interval statistics data, DB2 accounting data, and selected DB2 trace data processed by MXG® , SAS/ITSV®, or NeuMICS® and placed into SAS® data sets.

The DB2 Component consists of numerous modules, working together to (1) shape DB2 statistics data contained in a MXG, SAS/ITSV, or MICS[1] performance data base, (2) evaluate the data to assess problems or potential problems with DB2 performance, and (3) report the results from the evaluation. These modules are loaded and controlled by the central DB2 Component of CPExpert (titled DB2CPE).

- **Shape Data**. The majority of "processing" in the DB2 Component is accomplished by numerous modules whose function is to extract, combine, sort, summarize, correlate, and prepare the data for analysis. These modules process relevant data elements from DB2 interval statistics data from SMF Type 100 records, DB2 accounting data from SMF Type 101 records, and selected DB2 trace data from SMF Type 102 records. The actual data from SMF must have been processed by MXG or MICS, and must be contained in a MXG, SAS/ITSV, or MICS SAS performance data base.

- **Evaluate Data**. The evaluation of the data is accomplished by rules whose purpose is to evaluate problem areas or potential problem areas as revealed by the DB2 statistics.

- **Report Results**. The DB2 Component reports the results from the evaluation in a narrative format, somewhat similar to a report that might be produced by a performance analyst. When a problem or potential problem is detected, the reports normally identify specific intervals when the problem occurred and usually provide supporting data for each interval.

- **Maintain Historical Data**. The DB2 Component optionally maintains a historical file of all rule results. This file is available for subsequent analysis by users who may wish to trend the frequency of different findings by the DB2 Component.

Each finding is described in Appendix A of this document. The description summarizes the finding, discusses the rationale for the finding, suggests alternatives that should be considered, and lists references related to the finding.

[1] NeuMICS is hereafter referred to as the more commonly used term: MICS.

- The summary presents a short description of the finding.

- The discussion of the finding describes as much as necessary of the operation of DB2 as it relates to the particular rule. The purpose of the discussion is to explain the reasoning behind the rule, and what causes the rule to be produced. If appropriate, the discussion might refer you to related findings documented in the DB2 Component User Manual.

- The suggestions list possible actions that should be considered based on the findings. In most cases, alternative actions are listed. You must determine which action should be taken (this determination is based upon the suitability of the action to your own environment, the financial implications of the action, and the "political" acceptability of the action.)

- References are listed with each rule description. The references are provided so you can verify the analysis and suggestions.

©Copyright 1999, Computer Management Sciences, Inc.

Chapter 3: Data Sources

CPExpert analyzes the performance of your DB2 system based upon data from two sources:

- **DB2-created information**. CPExpert does most of its analysis based upon the information created by DB2 and recorded into SMF Type 100 (DB2 interval statistics) records, SMF Type 101 (DB2 accounting) records, and selected SMF Type 102 (DB2 trace) records.

  The "raw" SMF data contained in the SMF Type 100, Type 101, and Type 102 records must be translated into SAS format and placed into a SAS-based performance data base before CPExpert can use the information. The performance data base can be created by Merrill's Expanded Guide (MXG) software (or by SAS/ITSV using a MXG View), or by MICS. MXG is provided by Merrill Consultants, Dallas, Texas. MXG provides a low-cost mechanism by which installations can create and maintain a performance data base. MICS is provided by Computer Associates.

  The DB2 Component uses the following SAS datasets created by MXG:

MXG DATASET	DESCRIPTION	Comment
DB2STATS	Type 100 subtype 0 and subtype 1 statistics	
DB2STAT2	Buffer pool attributes	
DB2STATB	Buffer pool detail statistics	
DB2STATR	Remote locations statistics (DDF)	Optional
DB2GBPAT	Group buffer pool attributes	Optional
DB2GBPST	Group buffer pool detail statistics	Optional
DB2ACCT	Type 101 subtype 0 (accounting)	Optional
T102S106	Trace record subtype 106 (ZPARMS values)	Optional
T102S172	Trace record subtype 172 (deadlock details)	Optional
T102S196	Trace record subtype 196 (lock timeout details)	Optional

©Copyright 1999, Computer Management Sciences, Inc. Revised: October, 2001

Alternatively, the DB2 Component uses the following SAS datasets created by MICS:

MICS DATASET	DESCRIPTION	Comment
DB2DSY	DB2 System Activity File	
DB2DSD	DB2 Data Base Activity File	
DBPDSB	DB2 System Buffer Pool Activity File	
DB2DDY	DB2 System DDF File	Optional

Please note that the information in a performance data base created by MICS is not as comprehensive as that created by MXG. Consequently, some of the rules in the CPExpert DB2 Component will not be executed if CPExpert is analyzing a MICS performance data base. The DB2 Component will list the rules that cannot execute because of missing data. Additionally, some rules will be executed, but output might contain “???” in case of missing data. For example, MICS does not process the buffer pool attributes information and group buffer pool attributes produced by DB2. Since the attributes are not available in a MICS performance data base, CPExpert either will not execute some analysis or will show “???” in descriptive output.

- **Guidance information**. Guidance information for CPExpert is contained in prefix.CPEXPERT.USOURCE(DB2GUIDE). Section 3 of this User Manual describes the guidance information that can be provided to the DB2 Component.

Section 2: Installing the DB2 Component

installing the DB2 Component normally will occur when you install CPExpert on your system. This section briefly recaps the installation process itself, and mainly focuses on the procedures that you would use to make sure that the required DB2 information is available for analysis.

Chapter 1: Installing CPExpert

CPExpert normally is distributed on a non-labeled tape containing two source format libraries: SOURCE and USOURCE. The tape was created using IBM's IEBUPDTE utility software.

- **Unload the distribution tape, using IEBUPDTE.** The first step is to unload the distribution tape containing the CPExpert code. The PDS requires five to ten cylinders of IBM-3380 space (depending upon how many components of CPExpert you have ordered), and consists of members containing standard 80-byte record.

- **Allocate space for CPExpert data sets.** The second step is to allocate space for SAS libraries maintained by CPExpert. CPExpert maintains SAS data sets to describe SYS1.PARMLIB members, to describe problems that are discovered, and to contain historical information. These data sets contain the results from each rule produced by each CPExpert component processing system measurement data.

- **Provide general guidance to CPExpert** The third step is to modify the General Environment Controls section of the GENGUIDE Module. Most users will make changes to the General Environment Controls section only when installing CPExpert, or if their environment changes) are required. **Most guidance parameters are optional**.

Please review the *CPExpert Installation Guide* for details of the Installation process. The above steps were summarized here so you can appreciate that the installation process is trivial.

Chapter 2: Ensuring that required DB2 information is available

The primary activities to be accomplished when installing the DB2 Component are to ensure that appropriate DB2 data is recorded to SMF.

The DB2 Component **requires** that SMF Type 100 records (DB2 interval statistics) be available, that the records be processed by MXG, and that the MXG data sets be available in either a MXG performance data base or available in a SAS/ITSV performance data base.

DB2 interval statistics are collected continuously while DB2 is running, and the statistics are recorded to SMF at regular intervals. The default recording interval is 30 minutes. Unless you have a particular requirement, you should use this default recording interval.

The DB2 Component can **optionally** process SMF Type 101 records (DB2 accounting statistics). The DB2 accounting statistics can be quite large, and many sites to not collect the information. Consequently, the DB2 Component views the accounting statistics as optional.

Some of the analysis that is performed by the DB2 Component cannot be done if the DB2 accounting information is not available (for example, Rule DB2-321*: High accumulated wait time caused by page latch contention* will be suppressed if the DB2 accounting information is not available). The DB2 Component will list the rules that will be suppressed because of a lack of information. There are only a few findings that the DB2 Component makes based on the DB2 accounting statistics. You may wish to exclude these statistics initially if you do not normally collect the accounting statistics.

The DB2 Component can **optionally** process certain SMF Type 102 records (DB2 trace information). At present, the following DB2 Instrumentation Facility Component Identification (IFCID) trace records can be processed by the DB2 Component:

- **Trace type IFCID 106**: These trace records are normally written at each SMF recording interval. The Type 106 records contain information describing values of different ZPARMS parameter specifications. The DB2 Component uses this information (1) to guide its analysis of DB2 performance constraints and (2) provide information about ZPARMS specifications in particular rule output.

  Trace type IFCID 106 records add *insignificant* overhead to DB2 statistics recording (a single record is written for each DB2 interval statics period). CPExpert strongly recommends that these trace records be collected.

  The Trace IFCID 106 information is available in MXG T102S106 data sets.

- **Trace type IFCID 172**: The trace type IFCID 172 records contain information about deadlock situations. Deadlocks should rarely occur in a DB2 system. Consequently, these records should rarely be produced. CPExpert uses the trace type IFCID 172 information to report on units of work involved in a deadlock situation. This information is essential in identifying the cause of deadlocks.

  Trace type IFCID 172 records add *insignificant* overhead to DB2 statistics recording (a single record is written only when a deadlock is detected by DB2). CPExpert strongly recommends that these trace records be collected.

  The Trace IFCID 172 information is available in MXG T102S172 data sets.

- **Trace type IFCID 196**: The trace type IFCID 196 records contain information about lock timeout situations. Lock timeouts should rarely occur in a DB2 system. Consequently, these records should rarely be produced. CPExpert uses the trace type IFCID 196 information to report on units of work involved in a time situation. This information is essential in identifying the cause of lock timeouts.

  Trace type IFCID 196 records add insignificant overhead to DB2 statistics recording (a single record is written only when a deadlock is detected by DB2 CPExpert strongly recommends that these trace records be collected.

  The Trace IFCID 196 information is available in MXG T102S196 data sets.

Chapter 3. Starting DB2 Trace

It is possible (and even likely) that DB2 statistics are routinely collected at your site. You should consult with your DB2 system administrator or MVS system programmers to determine whether the required DB2 SMF records are collected[1]

Collecting DB2 statistics can be implemented automatically by parameters contained on the DSNTIPN panel, or can be implemented dynamically by using the DB2 START TRACE command.

Detailed information about starting and stopping DB2 traces for performance, accounting, audit, and statistics data is presented in Section 5 (Volume 2) of the *DB2 Administration Guide*. The below information is presented as a summary of the process involved:

- To have DB2 collect *statistical* information, accept the default (YES Class 1) for the **SMF STATISTICS** option on installation panel DSNTIPN. To collect statistical information for deadlock or lock timeout, specify class 3. To collect information about DDF error conditions, specify class 4.

- To have DB2 collect *accounting* information, accept the default ("YES" which includes Class 1) or specify "*" (which includes all classes) for the **SMF ACCOUNTING** option on installation panel DSNTIPN. You must consider which classes should be turned on.

- To have DB2 collect auditing information, specify * (all classes) for the AUDIT TRACE option on installation panel DSNTIPN. You must consider which classes should be turned on.

Statistics trace classes 1, 3, 4, and 5 are the default classes for the statistics trace if you specified YES for SMF STATISTICS in panel DSNTIPN. If the statistics trace is started using the START TRACE command, then class 1 is the default class.

[1] Recall that these are the SMF Type 100 interval statistics, the optional SMF Type 101 accounting statistics, and the optional SMF Type 102 (IFCID 172 and IFCID 196).

- Class 1 provides information about system services and database statistics. It also includes the system parameters that were in effect when the trace was started.
- Class 3 provides information about deadlocks and lock timeouts.
- Class 4 provides information about exceptional conditions.
- Class 5 provides information about data sharing.

You must make sure that the **SMFPRMxx** member of SYS1.PARMLIB has been updated to reflect the proper TYPE subparameter of SYS and SUBSYS (if specified). The default SYS parameters provided by IBM are:

```
SYS(TYPE(0:255),EXITS(IEFU83,IEFU84,IEFU85,IEFACTRT,IEFUJV,IEFUSI,
  IEFUJP,IEFUSO,IEFUJI,IEFUTL,IEFU29,IEFUAV),NOINTERVAL,NODETAIL)
```

It is common for system programmers to modify the SYS parameters, particularly to exclude various types or subtypes of SMF records. You should make sure that the TYPE subparameter of the SYS statement and SUBSYS statement (if specified) include a specification of (100:102), or another specification that spans the range of 100 to 102 SMF record types. This is required to make sure that the DB2 SMF statistics, accounting, and trace records are written to SMF. Further, make sure that these SMF records have not been included in the NOTYPE statement (which would exclude their being written to SMF).

For more information on SMF, refer to *OS/390 MVS System Management Facilities* (GC28-1783) and *OS/390 MVS Initialization and Tuning Reference (*SC28-1752).

Additionally, you must ensure that an adequate supply of SMF buffers have been allocated. The default buffer settings are probably insufficient to contain the volume of DB2 records (particularly if you are recording accounting statistics).

The volume of data DB2 trace collects can be quite large. Consequently, the number of trace records you request will affect system performance. In particular, when you activate a performance trace, you should qualify the -START TRACE command with the particular classes, plans, authorization IDs, and IFCIDs you want to trace. Unless you require trace records for other purposes, you should collect only the IFCID 172 and IFCID 196 records that would be used by the DB2 Component.

Chapter 4. Modifying the guidance variables for the DB2 Component

The final step in the installation process of the DB2 Component is to modify the guidance variables. These variables are contained in prefix.USOURCE(DB2GUIDE). Section 3 of this document describes how to modify the guidance variables.

Section 3: Specifying Guidance Variables

The CPEXPERT.USOURCE(DB2GUIDE) PDS member contains variables to establish the overall guidance for the DB2 Component. You specify certain guidance variables when you initially use the DB2 Component, and you modify the variables in the DB2GUIDE member whenever you wish to change the guidance to CPExpert.

The variables in the DB2GUIDE module can be viewed as "data selection" variables, "analysis control" variables, and "analysis guidance" variables. These three types of variables are discussed separately.

- The data selection variables allow you to select particular time intervals to be analyzed, specify the DB2 subsystem to be analyzed, etc. Chapter 1 describes the data selection variables and how they are used.

- The analysis control variables allow you specify the sources of data the DB2 Component is to analyze and to control the analysis the DB2 Component will perform. Chapter 2 describes the analysis control variables associated with the DB2 Component.

- The analysis guidance variables allow you to provide guidance to the DB2 Component with regard to analysis thresholds that may be unique to your environment. The defaults for the various thresholds may be appropriate for the analysis performed by the DB2 Component. However, you may have unique situations (or you may simply disagree with the defaults selected). Chapter 3 describes the analysis guidance variables and their defaults.

  You should not hesitate to alter the guidance variables to meet your requirements. Some of the default values are deliberately set to cause rules to initially be produced. The purpose of this is to call your attention to the performance implications of certain decisions you may have made. Those decisions may be appropriate for your environment, even though CPExpert may "flag" them as potential problems.

Please do not allow CPExpert to perform analysis or produce reports which are meaningless in your environment. If the analysis and reports produced by CPExpert do not meet your needs, alter the guidance to CPExpert. If the guidance is insufficient, please call Computer Management Sciences at (703) 922-7027 (or e-mail **Don_Deese@cpexpert.com**) so we can make changes to improve CPExpert for you!

Chapter 1: Data Selection Variables

The data selection variables allow you to select particular data to be analyzed. This chapter describes these variables and how the variables are used.

Exhibit 3-1 illustrates the portion of prefix.CPEXPERT.USOURCE(DB2GUIDE) that contains the data selection variables.

```
*****************************************************************;
*           DATA SELECTION AND PRESENTATION VARIABLES            ;
*****************************************************************;
%LET DB2DATES  =01FEB1991;  * START DATE FOR DATA ANALYSIS         ;
%LET DB2TIMES  =08:00:00;   * START TIME FOR DATA ANALYSIS         ;
%LET DB2DATEE  =31DEC9999;  * END DATE FOR DATA ANALYSIS           ;
%LET DB2TIMEE  =16:00:00;   * END TIME FOR DATA ANALYSIS           ;
%LET SHIFT     =Y;          * START AND END TIMES REFER TO SHIFT   ;
%LET SYSTEM    =*ALL;       * PROCESS ALL SYSTEMS                  ;
%LET SYSTEMn   =system;     * PROCESS SYSTEMn (n = 1-9)            ;
%LET DB2SID    =*ALL;       * DB2 SUBSYSTEM TO ANALYZE             ;
%LET ALL_DB2   = N;         * OPTION TO SUMMARIZE DB2 SUBSYSTEMS   ;
%LET SASODS    = N;         * CONTROLS WHETHER SAS ODS IS USED     ;
%LET PATH      = ;          * PATH FOR ODS OUTPUT                  ;
%LET FRAME     = DB2FRAME;  * GENERIC ODS FRAME NAME               ;
%LET CONTENTS  = DB2CONT;   * GENERIC ODS CONTENTS NAME         ;
%LET BODY      = DB2BODY    * GENERIC ODS BODY NAME                ;
%LET STYLE     =;           * ODS HTML STYLE OPTION                ;
%LET PDFODS    =N;          * CONTROLS WHETHER SAS PDF IS USED     ;
%LET LINKPDF   = ;          * LINK TO CPEXPERT DOCUMENTATION       ;
%LET URL       = N;         * CONTROLS .HTM IN SAS ODS FRAME OUTPUT;
*****************************************************************;
```

SAMPLE DISPLAY OF CPEXPERT.USOURCE(DB2GUIDE) MODULE

EXHIBIT 3-1

For most users, the data selection variables rarely will be modified. This is because the DB2 Component normally will process a day's data, and the data will be processed after the data has been placed into a performance data base. The MXG daily files or MICS DETAIL files will contain the data you wish CPExpert to evaluate.

Other users might wish to restrict the analysis to only a specific shift (for example, you might not particularly care about DB2 performance at times other than prime shift). Consequently, you can select time intervals for analysis. The data selection variables may be used to select specific data intervals to analyze (for example, you may notice that DB2 performance is particularly poor during some interval and wish analysis focused only on that interval).

Chapter 1.1: DB2DATES and DB2TIMES variables

The DB2DATES and DB2TIMES variables specify the start date and start time, respectively, for the interval that the DB2 Component is to analyze. These variables (in conjunction with the DB2DATEE and DB2TIMEE variables) allow you to select specific periods of data to analyze. For example, to specify that data selection should start at 08:00:00 on March 4, 2002, specify:

```
%LET DB2DATES = 04MAR2002;  * START DATE FOR DATA ANALYSIS;
%LET DB2TIMES = 08:00:00;   * START TIME FOR DATA ANALYSIS;
```

CPExpert will use the DB2DATES and DB2TIMES variables to exclude data before the specified date and time.

The DB2DATES and DB2TIMES variables (and the DB2DATEE and DB2TIMEE variables described below) are not normally altered from their defaults. These variables are used **only** if you wish to analyze a subset of the data contained in your performance data base. Under most conditions, you simply use the defaults and CPExpert will analyze all available DB2 statistics.

Chapter 1.2: DB2DATEE and DB2TIMEE variables

The DB2DATEE and DB2TIMEE variables specify the end date and end time, respectively, for the interval of DB2 statistics or SMF data the DB2 Component is to analyze. For example, to specify that data selection should end at 17:00:00 on March 8, 2002, specify:

```
%LET DB2DATEE = 08MAR2002;  * END DATE FOR DATA ANALYSIS;
%LET DB2TIMEE = 17:00:00;   * END TIME FOR DATA ANALYSIS;
```

©Copyright 1999, Computer Management Sciences, Inc.

If you maintain DB2 statistics in a performance data base, CPExpert will use the DB2DATEE and DB2TIMEE variables to exclude data after the specified date and time.

Chapter 1.3: SHIFT variable

The SHIFT variable is used with the DB2DATES, DB2TIMES, DB2DATEE, and DB2TIMEE variables. CPExpert will use the DB2DATES and DB2TIMES variables to exclude data before the specified date and time. The SHIFT variable allows you indicate how the time-selection variables should be used.

- If the SHIFT variable is "N", the time-selection will be based upon the absolute start and end dates/times specified. For example, if you wish CPExpert to process **all** data during a week, the start date and start time would be specified as the beginning of the week, and the end date and end time would be specified as the end of the week. You would specify "%LET SHIFT = N;" to process each 24-hour day. In the example shown above, data would be processed from 08:00:00 on 4 March until 17:00:00 on 8 March.

- If the SHIFT variable is "Y", the time-selection will be based upon the start and end dates, and the start and end times within each selected date. In the example shown above, perhaps you wished to process only the daily shift beginning at 08:00:00 and ending at 17:00:00. You would specify "%LET SHIFT = Y;" to process only the identified shift data, during the selected dates. This option is valid only if you are processing data contained in a performance data base.

Chapter 1.4: SYSTEM variable

The SYSTEM variable is used to specify whether all systems in the performance data base should be evaluated, or to select a specific system identification to be evaluated.

Some users have data from multiple systems in their performance data base. For many of these users, or for users who have data for a single system represented in their performance data base, the default "*ALL" will be appropriate. No change of the SYSTEM variable would be required for these users.

However, some users who have data from multiple systems in their performance data base may wish to evaluate only a single system with the parameters specified in this member of DB2GUIDE. For example, they might be temporarily interested in evaluating the performance of only an "important" system (such as a major production system) and not be interested in evaluating the performance of other systems with data in the performance data base. This evaluation can be accomplished by changing the SYSTEM variable to specify the system identification to be evaluated. For example, to specify that only data from SYS1 should be evaluated, specify:

©Copyright 1999, Computer Management Sciences, Inc.

```
%LET SYSTEM = SYS1 ; * PROCESS ONLY DATA FROM SYS1;
```

In another situation, a CPExpert user might wish to evaluate different systems with different DB2GUIDE parameters. These different evaluations can be accomplished by different executions of the DB2 Component. For each execution of the DB2 Component, the USOURCE DD statement would be changed to reference different USOURCE libraries. Each USOURCE library would contain guidance members with appropriate guidance variables. The SYSTEM variable for each DB2GUIDE guidance member would specify the system identification to which the guidance applied.

Chapter 1.5: SYSTEMn variable(s)

The SYSTEMn variable(s) are used to select multiple systems to be evaluated.

As described in the SYSTEM guidance variable discussion above, some sites have data from multiple systems in their performance data base. These sites can process data from all systems by specifying %LET SYSTEM=ALL; in USOURCE(CICGUIDE), or can select a specific system to process by specifying %LET SYSTEM=system; in USOURCE(CICGUIDE), where “system” is the system identification of the system to be processed.

Some sites have data from multiple systems in their performance data base and do not want to process all systems, but do wish to process more than one system. For example, some systems might be production systems and some might be test systems. For these sites, the **SYSTEMn** guidance variable can be used to select more than one specific system to analyze.

The SYSTEM guidance variable can be used to select data from only one system to analyze, and the SYSTEMn guidance variable(s) can be used to select up to 9 additional systems to analyze. For example, if you wish to analyze data from four systems (named SYSA, SYSB, SYSC, AND SYSX) in a single execution of the DASD Component, specify:

```
%LET SYSTEM  = SYSA ; * PROCESS DATA FROM SYSA;
%LET SYSTEM1 = SYSB ; * PROCESS DATA FROM SYSB;
%LET SYSTEM2 = SYSC ; * PROCESS DATA FROM SYSC;
%LET SYSTEM3 = SYSX ; * PROCESS DATA FROM SYSX;
```

Chapter 1.6: DB2SID variable

The DB2SID variable is used to specify whether all DB2 subsystems in the performance data base should be evaluated, or to select a specific subsystem identification to be evaluated.

Some users have data from multiple DB2 subsystems in their performance data base. For many of these users, or for users who have data for a single DB2 subsystem represented in their performance data base, the default "*ALL" will be appropriate. No change of the DB2SID variable would be required for these users.

However, some users who have data from multiple DB2 subsystems in their performance data base may wish to evaluate only a single DB2 subsystem with the parameters specified in this member of DB2GUIDE. For example, they might be temporarily interested in evaluating the performance of only an "important" DB2 subsystem (such as a major production system) and not be interested in evaluating the performance of other DB2 subsystems with data in the performance data base. This evaluation can be accomplished by changing the DB2SID variable to specify the DB2 subsystem identification to be evaluated. For example, to specify that only data from the DB2 S003 subsystem should be evaluated, specify:

```
%LET DB2SID = S003 ; * PROCESS ONLY DATA FROM S003;
```

Please note that if you have a single DB2 subsystem per system, you can specify processing of data for the DB2 subsystem by using either the SYSTEM variable or the DB2SID variable. However, you may not specify selection of a system and a DB2 subsystem, unless the DB2 subsystem operates on the system selected. The selection logic is quite simple: select system based on SYSTEM variable, then select DB2 subsystem based on the DB2SID variable.

CPExpert allows a “wild card” specification with the DB2SID variable. If you wish to select all DB2 subsystems that begin with certain common characters, you can specify the leading characters of the DB2 subsystems, followed by the “%” symbol. For example, suppose that you have production DB2 subsystems and test DB2 subsystems. Further, suppose that the naming convention is that production DB2 subsystems begin with “DB2P” and test DB2 subsystems begin with “DB2T”. You can select only the production DB2 subsystems for evaluation by specifying:

```
%LET DB2SID = DB2P% ; * PROCESS ONLY PRODUCTION SUBSYSTEMS;
```

©Copyright 1999, Computer Management Sciences, Inc.

Chapter 1.7: ALL_DB2 variable

The ALL_DB2 variable is used to specify whether a single report should be produced to summarize the analysis of all DB2 subsystems analyzed.

Some organizations have a large number of DB2 subsystems. For these organizations, the output from the DB2 Component can become very large and difficult to review. When the ALL_DB2 variable is set as **%LET ALL_DB2=Y;** in USOURCE(DB2GUIDE), CPExpert will summarize the results from analyzing all DB2 subsystems, and produce a summary report from the result. Each unique finding (or Rule) will show all DB2 subsystems to which the finding applies.

The default of the ALL_DB2 variable is %LET ALL_DB2=N; to specify that CPExpert should produce a report for each DB2 subsystem analyzed. If you wish CPExpert to summarize the results from the analysis and produce a summary report for all DB2 subsystems, specify **%LET ALL_DB2=Y;** in USOURCE(DB2GUIDE).

Chapter 1.8: SAS Output Delivery System

Output from CPExpert is created using Basic SAS statements. This Basic SAS output is designed for a standard SAS printer (line) format. With SAS Release 8, SAS users can use the SAS Output Delivery System to create output that is formatted in Hypertext Markup Language (HTML). This output can be browsed with Internet Explorer, Netscape, or any other browser that fully supports the HTML 3.2 tag set.

The CPExpert WLM Component, DB2 Component, CICS Component, and DASD Component support the SAS ODS features.

Please reference the *CPExpert Installation Guide* for more detailed information about using the SAS ODS feature of CPExpert.

©Copyright 1999, Computer Management Sciences, Inc.

Chapter 2: Analysis Control Variables

The analysis control variables allow you to control how the DB2 Component analyzes your DB2 configuration. These variables are used to (1) specify whether the DB2 subsystem is production or test, (2) specify whether DB2 accounting data are available for analysis, (3) specify whether DB2 ZPARMS information is available for analysis, (4) specify whether analysis guidance is provided for individual virtual buffer pools, and (5) specify whether analysis guidance is provided for individual group buffer pools,.

This chapter describes the analysis control variables associated with the DB2 Component.

Exhibit 3-2 illustrates the portion of CPEXPERT.USOURCE(DB2GUIDE) that contains the analysis control variables.

```
*************************************************************;
*                    ANALYSIS CONTROL VARIABLES              ;
*************************************************************;

%LET PRODTEST  = PROD;      * PRODUCTION OR TEST SUBSYSTEM;
%LET DB2ACCTX  = N;         * DB2 ACCOUNTING DATA AVAILABILITY;
%LET T102S106  = Y;         * DB2 ZPARMS DATA AVAILABILITY;
%LET T102S172  = N;         * DB2 DEADLOCK TRACE RECORDS AVAILABILITY;
%LET T102S196  = N;         * DB2 TIMEOUT TRACE RECORDS AVAILABILITY;
%LET SMF74ST   = N;         * SMF TYPE 74 RECORDS AVAILABLILITY;
%LET BPGUIDE   = N;         * GUIDANCE PROVIDED FOR INDIVIDUAL BUFFER;
%LET GBPGUIDE  = N;         * GUIDANCE PROVIDED FOR GLOBAL BUFFERS;

%LET DB2LIB    = &PDBLIB;   * SAS LIBRARY CONTAINING DB2 DATA;
%LET DB2STATS  = &DB2LIB..DB2STATS; * SPECIFY DB2STATS SAS LIB.FILE;
%LET DB2STATB  = &DB2LIB..DB2STATB; * SPECIFY DB2STATB SAS LIB.FILE;
%LET DB2STAT2  = &DB2LIB..DB2STAT2; * SPECIFY DB2STAT2 SAS LIB.FILE;
%LET DB2GBPAT  = &DB2LIB..DB2GBPAT; * SPECIFY DB2GBPAT SAS LIB.FILE;
%LET DB2GBPST  = &DB2LIB..DB2GBPST; * SPECIFY DB2GBPST SAS LIB.FILE;
%LET DB2STATR  = &DB2LIB..DB2STATR; * SPECIFY DB2STATR SAS LIB.FILE;
%LET DB2ACCT   = &DB2LIB..DB2ACCT ; * SPECIFY DB2ACCT SAS LIB.FILE;

*************************************************************;
```

SAMPLE DISPLAY OF CPEXPERT.USOURCE(DB2GUIDE) MODULE

EXHIBIT 3-2

Chapter 2.1: Specifying Production or Test Subsystem

The **PRODTEST** variable tells the DB2 Component whether the DB2 subsystem is a production test subsystem. The analysis performed by the DB2 Component varies, depending upon whether the DB2 subsystem is production or test. Some situations may be acceptable in a DB2 test subsystem, but would be unacceptable in a DB2 production subsystem.

Specify **%LET PRODTEST=TEST;** to change the guidance from the default DB2 production region classification to a DB2 test subsystem classification.

Chapter 2.3: Specifying whether DB2 accounting data are available

The **DB2ACCTX** variable tells the DB2 Component whether DB2 accounting data are available and are to be processed.

The DB2 accounting data contain a wealth of information, but unfortunately the data set is very large. At present, the DB2 Component does very little analysis of the DB2 accounting data, so the default for this parameter is set to “N” to suppress processing of DB2 accounting data. Future releases of CPExpert may process the accounting data. If this enhancement should be implemented, the default will be changed from “N” to “Y” to allow processing of the DB2 accounting data.

CPExpert will suppress any rule which depends upon DB2 accounting data if the accounting data are not available. CPExpert will produce a report listing all rules suppressed because variables were not available.

Chapter 2.4: Specifying availability of Trace IFCID 106 records

DB2 stores ZPARMS information into the SMF Type 102 (Subtype 106) records.
The **T102S106** variable tells the DB2 Component whether the DB2 Trace IFCID 106 records are available.

CPExpert will suppress any rule which depends upon DB2 ZPARMS data if the data are not available. CPExpert will produce a report listing all rules suppressed because relevant variables were not available.

Chapter 2.5: Specifying availability of Trace IFCID 172 records

DB2 stores detaled information about deadlock situations into the SMF Type 102 (Subtype 172) Trace records. The **T102S172** variable tells the DB2 Component whether the DB2 Trace IFCID 172 records are available.

CPExpert will suppress any rule which depends upon DB2 deadlock data contained in Trace IFCID 172 if the data are not available. CPExpert will produce a report listing all rules suppressed because relevant variables were not available.

Chapter 2.6: Specifying availability of Trace IFCID 196 records

DB2 stores detailed information about timeout situations into the SMF Type 102 (Subtype 196) Trace records. The **T102S196** variable tells the DB2 Component whether the DB2 Trace IFCID 196 records are available.

CPExpert will suppress any rule which depends upon DB2 timeout data contained in Trace IFCID 196 if the data are not available. CPExpert will produce a report listing all rules suppressed because relevant variables were not available.

Chapter 2.7: Specifying availability of SMF Type 74 (structure) records

This analysis control variable applies only to DB2 Data Sharing environments.

With DB2 data sharing environments, CPExpert analyzes some potential problems associated directly with DB2 and uses standard DB2 interval statistics for this analysis. For example, CPExpert analyzes data relating to group buffer pools using QBGLxxx variables contained in the MXG DB2GBPST file or the MICS DB2DSDxx file. However, some important analysis related to data sharing can be done only if the SMF Type 74 (structure) information is available for analysis. For example, analysis of synchronous service time delays can be performed only using data from SMF Type 74 (structure) records.

The **SMF74ST** variable tells the DB2 Component whether the SMF Type 74 (structure) records are available. CPExpert will suppress any rule which depends upon coupling facility structure data contained in SMF Type 74 (structure) records if the data are not available. CPExpert will produce a report listing all rules suppressed because relevant variables were not available.

Please note that if you are processing data in a MICS performance data base, the SMF Type 74 (structure) information is the HARCFH and HARCFR. You can use the HARLIB guidance variable to specify the location of the HAR Information Area if different from the DB2 Information Area. Please refer to the *CPExpert Installation Guide* for additional information.

Chapter 2.8: Specifying individual guidance for buffer pools

The optional **BPGUIDE** variable tells the DB2 Component whether guidance is provided for individual buffer pools. Please refer to Chapter 4 of this section.

Chapter 2.9: Specifying individual guidance for group buffer pools

The optional **GBPGUIDE** variable tells the DB2 Component whether guidance is provided for individual group buffer pools. Please refer to Chapter 5 of this section.

Chapter 2.10: Specifying SAS library containing DB2 data

The optional **DB2LIB** guidance variable allows you to specify a SAS library containing the MXG or MICS DB2 performance data base files. This SAS library can be different from the SAS library containing your normal performance data acquired from SMF, RMF, etc. The default value for the DB2LIB guidance variable points to the SAS library identified by the PDBLIB guidance variable in USOURCE(GENGUIDE).

You should alter the default DB2LIB guidance variable only if your DB2 performance data base is contained in a SAS library different from the normal MXG or MICS performance data base. The value specified for DB2LIB is used by CPExpert as the DDNAME used to access DB2 performance data.

Chapter 2.11: Specifying data set name for DB2 datasets

The optional **DB2STATS**, **DB2STATB**, **DB2STAT2**, **DB2GBPAT**, **DB2GBPST, DB2STATR, and DB2ACCT** guidance variables apply only if your DB2 datasets are not contained in your standard MXG performance data base. These variables allow you to specify a SAS library.file for any MXG DB2 data set used by CPExpert. These SAS libraries.files can be different from the SAS library.file containing any other MXG data.

The default values for these **optional** MXG guidance variables point to the SAS library identified by the DB2LIB guidance variable in USOURCE(DB2GUIDE), which in turn normally points to the SAS library identified by the PDBLIB guidance variable in USOURCE(GENGUIDE). You should alter the default guidance variables only if you have any of these DB2 files contained in a SAS library different from the normal DB2 performance data base. Additionally, you can specify a file name different from the standard MXG file.

©Copyright 1999, Computer Management Sciences, Inc.

Chapter 3: Analysis Guidance Variables

The analysis guidance variables allow you to provide guidance to the DB2 Component as CPExpert applies the DB2 analysis rules. The CPEXPERT.USOURCE(DB2GUIDE) module contains defaults for each guidance variable. These defaults may be appropriate for the analysis performed by the DB2 Component. However, you may have unique situations (or you may simply disagree with the defaults selected).

This chapter describes the analysis guidance variables and their defaults. **Do not hesitate to make changes if the defaults for the analysis guidance variables do not meet your needs.**

Please contact Computer Management Sciences if the guidance variables are inadequate for your needs.

Exhibit 3-3 illustrates the portion of CPEXPERT.USOURCE(DB2GUIDE) that contains the analysis guidance variables.

©Copyright 1999, Computer Management Sciences, Inc.

```
*********************************************************************;
*                  ANALYSIS GUIDANCE VARIABLES                      ;
*********************************************************************;

%LET DB2Rnnn  =OFF;  * EXAMPLE: TURN OFF DB2-nnn RULE;
%LET ASYNCSRV =5000; * ACCEPTABLE SERVICE TIME, ASYNCHRONOUS REQUESTS;
%LET BUFFPGRT =100;  * ACCEPTABLE PAGE FAULT RATE FOR ALL BUFFER POOLS;
%LET LOCKCONT =1;    * MAXIMUM PERCENT LOCK CONTENTION.
%LET LOCKSRV  =250;  * ACCEPTABLE SERVICE TIME, LOCK REQUESTS;
%LET PAGERATE =10;   * ACCEPTABLE PAGE-IN RATE, INDIVIDUAL BUFFER POOL;
%LET PCTCBA   =50%;  * PERCENT BUFFERS AVAILABLE;
%LET PCTCTL   =95%;  * PERCENT CURSOR TABLE SECTIONS FOUND;
%LET PCTDBDL  =99%;  * PERCENT DATA BASE DESCRIPTOR SECTIONS FOUND;
%LET PCTDPF   = 0%;  * PERCENT DYNAMIC SEQUENTIAL PREFETCH;
%LET PCTEDM   =25%;  * PERCENT FREE PAGES IN EDM POOL;
%LET PCTESC   = 0%;  * PERCENT LOCK ESCALATION WAS NOT EFFECTIVE;
%LET PCTFLAM  = 0%;  * PERCENT FAILED LOOK-AHEAD TAPE MOUNTS;
%LET PCTPTL   =95%;  * PERCENT PACKAGE TABLE SECTIONS FOUND;
%LET PCTRACT  =25;   * PERCENT READS SATISFIED FROM ACTIVE LOG DATA SET;
%LET PCTRAND  =95%;  * PERCENT GETPAGE REQUESTS FROM RANDOM REQUESTORS;
%LET PCTRSAT  =95%;  * PERCENT RECONNECTION ATTEMPTS SUCCEEDED;
%LET PCTSEQ   =95%;  * PERCENT GETPAGE REQUESTS FROM SEQ. REQUESTORS;
%LET PCTSLOCK =.5%;  * PERCENT LOCK REQUESTS THAT WERE SUSPENDED;
%LET PCTHITGB =95%;  * PERCENT READ HIT FOR CROSS-INVALIDATED PAGES;
%LET PCTXIGBP = 0%;  * ACCEPTABLE PCT RECLAIMS, CROSS-INVALIDATED PAGES;
%LET Q3STCTHW = 0;   * THREAD REQUESTS WERE QUEUED AT CREATE;
%LET Q3STMEOM = 0;   * END-OF-MEMORY (NON-DB2 TASK MVS DELETED);
%LET Q3STMEOT = 0;   * END-OF-TASK (NON-DB2 TASK ABENDS);
*********************************************************************;
```

SAMPLE DISPLAY OF CPEXPERT.USOURCE(DB2GUIDE) MODULE

EXHIBIT 3-3

```
*************************************************************;
*                  ANALYSIS GUIDANCE VARIABLES        ;
*************************************************************;
%LET QBGLAD    = 0;    * CF READS PREFETCH DATA RETURNED;
%LET QBGLAN    = 0;    * CF PREFETCH READS, NO DATA, NO DIRECTORY;
%LET QBGLAR    = 0;    * CF PREFETCH READS, NO DATA, DIRECTORY CREATED;
%LET QBGLCN    = 0;    * TIMES WHEN CASTOUT ENGINE NOT AVAILABLE;
%LET QBGLRF    = 0;    * CF READS FAILED DUE LACK OF CF STORAGE;
%LET QBGLWF    = 0;    * CF WRITES FAILED DUE LACK OF CF STORAGE;
%LET QBGLSU    = 0;    * TIMES WHEN CF WRITE ENGINE WAS AVAILABLE;
%LET QBSTARF   = 0;    * UNSUCCESSFUL HP TO VP READS, USING ASYNC ADMF;
%LET QBSTAWF   = 0;    * UNSUCCESSFUL HP TO VP WRITES, USING ASYNC ADMF;
%LET QBSTDMC   = 0;    * DATA MANAGER THRESHOLD REACHED;
%LET QBSTHRF   = 0;    * UNSUCCESSFUL HP TO VP MVPG READS;
%LET QBSTHWF   = 0;    * UNSUCCESSFUL VP TO HP PAGE WRITES;
%LET QBSTIMW   = 0;    * NUMBER OF IMMEDIATE WRITES;
%LET QBSTJIS   = 0;    * PREFETCH I/O STREAMS DENIED;
%LET QBSTMIG   = 0;    * TIMES MIGRATED DATA SETS WERE ENCOUNTERED;
%LET QBSTNGT   = 0;    * UNSUCCESSFUL GETPAGE REQUESTS;
%LET QBSTPL1   = 0;    * PREFETCH REDUCED TO ½ OF NORMAL;
%LET QBSTPL2   = 0;    * PREFETCH REDUCED TO 1/4 OF NORMAL;
%LET QBSTPQF   = 0;    * TIMES I/O PARALLELISM WAS DOWNGRADED;
%LET QBSTREE   = 0;    * PREFETCH DISABLED NO READ ENGINE;
%LET QBSTRTO   = 0;    * TIMES RECALL TIMEOUTS OCCURRED;
%LET QBSTSPD   = 0;    * SEQ PREFEETCH REQ DISABLED UNAVAIL BUFFER;
%LET QBSTWEE   = 0;    * WRITE ENGINE NOT AVAILABLE FOR ASYNC I/O;
%LET QBSTWFD   = 0;    * WORKFILES DENIED DURING SORT;
%LET QBSTWFF   = 0;    * SORT INEFFICIENT (BUFFER SHORTAGE);
%LET QBSTWFR   = 0;    * NUMBER OF MERGE PASSES (CONTROLS DB2-226);
%LET QBSTWIO   =20;    * ASYNC WRITE I/O (CONTROLS DB2-225);
%LET QBSTWKPD  = 0;    * PREFETCH ABORT - ZERO*PREFETCH QTY;
%LET QBSTXFL   = 0;    * TIMES BUFFER FULL CONDITION OCCURRED;
```

SAMPLE DISPLAY OF CPEXPERT.USOURCE(DB2GUIDE) MODULE

EXHIBIT 3-3 (Continued)

```
%LET QDSTQCRT = 0;   * CONVERSATIONS DEALLOC DUE TO ZPARM LIMIT;
%LET QDSTQDBT = 0;   * DB ACCESS QUEUED DUE TO MAX REMOTE THREADS;
%LET QISEFAIL = 0;   * FAILURES DUE TO POOL FULL;
%LET QISTCOLS = 0;   * COLS BYPASSED INVALIDATED DB2 SERVICE;
%LET QISTRLLM = 0;   * RID TERMINATED, ENTRIES GT CALCULATED LIMIT;
%LET QISTRMAX = 0;   * RID TERMINATED, MAXIMUM STORAGE;
%LET QISTRPLM = 0;   * RID TERMINATED, ENTRIES GT PHYSICAL LIMIT;
%LET QISTRSTG = 0;   * RID TERMINATED, INSUFFICIENT STORAGE;
%LET QJSTALR  = 0;   * ARCHIVE LOG READ ALLOCATIONS;
%LET QJSTALW  = 0;   * ARCHIVE LOG WRITE ALLOCATIONS;
%LET QJSTRARH = 0;   * READS SATISFIED FROM ARCHIVE LOG DATA SET;
%LET QJSTTVC  = 0;   * READ ACCESSES DELAYED, TAPE VOLUME CONTENTION;
%LET QJSTWTB  = 0;   * WAITS DUE TO UNAVAILABLE WRITE BUFFER;
%LET QJSTWUR  = 0;   * READ ACCESSES DELAYED, UNAVAILABLE RESOURCES;
%LET QLSTCNVQ = 0;   * CONV REQS QUEUED BY DDF, WAITING FOR ALLOC
%LET QLSTINDT = 0;   * THREADS INDOUBT WITH REMOTE AS COORDINATOR
%LET QTXADEA  = 0;   * LOCK DEADLOCK COUNT
%LET QTXALES  = 0;   * LOCKS ESCALATED TO SHARED MODE
%LET QTXALEX  = 0;   * LOCKS ESCALATED TO EXCLUSIVE MODE
%LET QTXASLOC = 0;   * SUSPEND COUNT,  LOCK CONFLICT
%LET QTXATIM  = 0;   * LOCK TIMEOUT COUNT
%LET QWACAWTP = 0;   * WAIT TIME DUE TO PAGE LATCH CONTENTION
%LET QWSBSBUF = 0;   * BUFFER ERRORS, SMF DESTINATION
%LET QXCALLAB = 0;   * STORED PROCEDURE ABENDS
%LET QXCALLRJ = 0;   * SQL CALLS REJECTED
%LET QXCALLTO = 0;   * SQL CALLS TIMED OUT WAITING SCHEDULE
%LET QXCOORNO = 0;   * PARALLEL GROUPS EXECUTED ON A SINGLE DB2
%LET QXDEGBUF = 0;   * PARALLEL GROUPS FELL BACK BUFFERS
%LET QXDEGENC = 0;   * PARALLEL GROUPS EXECUTED IN SEQUENTIAL
%LET QXISORR  = 0;   * GROUPS EXECUTED ON SINGLE DB2, REPEAT READ
%LET QXREDGRP = 0;   * PARALLEL GROUPS REDUCED DUE TO BUFFERS
%LET QXSTDEXP = 0;   * DB2 DISCARDED PREPARED STATEMENT
%LET QZZCSKIP = 0;   * BYPASSED DB2, NOT ENOUGH BUFFERS
%LET SYNCSRV  =350;  * ACCEPTABLE SERVICE TIME (MICROSEC) SYNCH REQ;
%LET SYNCCHG  = 10;  * ACCEPTABLE PCT CHANGED (SYNCH TO ASYNCH);
*******************************************************************************
****;
```

SAMPLE DISPLAY OF CPEXPERT.USOURCE(DB2GUIDE) MODULE

EXHIBIT 3-3 (Continued)

Chapter 3.1: Turning OFF DB2 Component Rules

The default guidance values for the DB2 Component are specified based on IBM's guidance contained in IBM's *DB2 for OS/390 Administration Guides* supplemented by techniques and guidance contained in IBM Redbooks, by papers presented at various technical conferences, and by IBM's suggestions in the DSNWMGS macros. For most findings, users can use analysis guidance variables to control the analysis; rule results will be produced only when situations exceed the guidance provided.

However, some users of the DB2 Component wish to suppress the analysis and findings of particular rules. This desire typically is caused by (1) an overall disagreement with the finding, (2) an inability to make a suggested change, or (3) a decision that a particular finding is inapplicable to a particular DB2 subsystem.

Regardless of the reason for wishing to suppress particular findings by the DB2 Component, users wish the ability to "turn off" certain rules.

All rules are ON by default, although the DB2 Component may turn rules OFF if insufficient data exists to perform analysis or if the rule does not apply to the version of DB2 being analyzed.

Rules can be turned OFF by specifying **%LET DB2Rnnn = OFF;** , where "nnn" is the rule number that you wish to turn OFF. For example, you can turn OFF Rule DB2-101 by specifying **%LET DB2R101=OFF;** in USOURCE(DB2GUIDE).

If you are specifying specific guidance for particular DB2 buffer pools or global buffer pools, you can **NOT** turn rules OFF (or turn rules ON) for individual buffer pools. This is because the guidance for buffer pools or global buffer pools is read into SAS data sets, and the specific guidance values are placed into arrays, overlaying the defaults, as each buffer pool or global buffer pool is analyzed. Turning rules "off" employs the SAS %LET statements, which cannot be dynamically modified. However, the guidance values for particular rules can be specified for an individual buffer pool so that a rule would effectively be "on" or "off" for that rule.

Please note that the DB2 Component verifies that all required data is present in your performance data base before invoking each rule. A rule will be suppressed if any required data is missing, regardless of your specification to suppress or enable the rule.

Chapter 3.2: ASYNCSRV variable

This variable applies only if you have indicated that SMF Type 74 (structure) information is available (that is, you have specified **%LET SMF74ST=Y;** in USOURCE(DB2GUIDE). Signaling requests to a coupling facility can occur only if a subchannel to the coupling facility is available. If no subchannel is available, the cross-system extended services

(XES) will either enter a CPU "spin loop" waiting for a subchannel to become available or queue the request until a subchannel is available.

CPExpert compares the asynchronous service time (R744ASTM) against the **ASYNCSRV** variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-661 when the asynchronous service time is greater than the ASYNCSRV guidance variable.

The default value for the ASYNCSRV variable is 5000, indicating that CPExpert should produce Rule DB2-661 when asynchronous service time is more than 5000 microseconds. You can alter this analysis using the ASYNCSRV guidance variable. For example, if you wish to be notified only when the synchronous service time is greater than 10000 microseconds, specify:

```
%LET ASYNCSRV = 10000 ; * ACCEPTABLE SERVICE TIME, ASYNCH REQUESTS;
```

Chapter 3.3: BUFFPGRT - Acceptable page-in rate, all buffer pools

CPExpert sums the QB1TRPI, QB2TRPI, QB3TRPI, QB4TRPI, QB1TWPI, QB2TWPI, QB3TWPI, and QB4TWPI variables (the number of page-ins required for read I/O and the number of page-ins required for write I/O) in DB2STATS, to yield the total page-ins for all buffer pools. The sum is divided by the DB2 statistics interval to yield a page-in rate per second for all buffer pools. CPExpert then compares this total page-in rate with the **BUFFPGRT** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-215 when the number of page-ins for read I/O and write I/O exceeds the value specified by the **BUFFPGRT** guidance variable.

The default value for the **BUFFPGRT** guidance variable is 100, indicating that CPExpert should produce Rule DB2-215 whenever total page-ins for read I/O and write I/O exceed 100 page-ins per second. You can alter the analysis by specifying a different value for the acceeptable page-in rate. For example, if you wish to be notified whenever more than 50 page-ins per second occur, specify:

```
%LET BUFFPGRT  = 50 ; * ACCEPTABLE PAGE-IN RATE;
```

Chapter 3.4. LOCKCONT variable

This variable applies only if you have indicated that SMF Type 74 (structure) information is available (that is, you have specified **%LET SMF74ST=Y;** in USOURCE(DB2GUIDE).

Locking is the mechanism used to reserve all or part of a database so that other programs will not be able to update the data until you have finished processing the data. By locking the data, users can be sure that the information they are processing is current. Without locking, users might lose updates or access invalid or incomplete data. Locking is necessary, of course, only if one or more of the users of the data will be performing updates. If no updating of the data is performed, locking is unnecessary; the data may be concurrently accessed by any number of user without worry that the data is incomplete or invalid.

Lock contention occurs when one user wishes to access data and some other user has placed a lock on the data. The user wishing to access the data usually is suspended until the data is available (that is, until the lock is released). Techniques such as separating data, choosing locking parameters, and monitoring for contention can be used to provide a balance between concurrency of access, isolation and integrity of data, and efficient use of system resources.

The coupling facility lock structure contains information used to determine cross-system contention on a particular resource. IRLM assigns (or "hashes") locked resources to an entry value in the lock structure in the coupling facility. IRLM uses the lock table to determine whether a resource is locked. If the lock structure defined on the coupling facility is too small, the hashing algorithm can select the same lock table entry for two different locks. This situation is termed *false lock contention*. The user wishing to access the locked data is suspended until it is determined that there is no real lock contention on the resource.

CPExpert selects DB2 lock structure information from the SMF Type 74 (structure) information.

- CPExpert divides SMF Type 74 (Subtype 4) field R744SSCN (the number of times any request encountered lock contention) by field R744STRC (the total number of lock-related requests), to yield the percent of requests that experienced lock contention. CPExpert compares this percentage with the **LOCKCONT** guidance variable in USOURCE(DB2GUIDE). CPExpert producess Rule DB2-651 when the percent of lock contention exceeds the value specified by the LOCKCONT variable.

- CPExpert divides field R744SFCN (the number of times any request encountered false lock contention) by field R744STRC (the total number of lock-related requests), to yield the percent of requests that experienced false lock contention. CPExpert produces Rule DB2-652 when more than 0.1% of the total requests encountered false lock contention. This threshold (0.1% of total requests) is based on the advice from several IBM documents listed in the Reference section of this discussion. Additionally, CPExpert verifies that overall lock contention was at least 25% of the LOCKCONT value, to make sure that a relatively significant amount of lock contention occurred.

©Copyright 1999, Computer Management Sciences, Inc.

The default specification for the LOCKCONT guidance variable is **1%** indicating that Rule DB2-651 will be produced when more than 1% of the requests experienced lock contention. Additionally, Rule DB2-652 will not be produced unless at least 0.5% of the requests experienced lock contention. You can alter this analysis using the LOCKCONT guidance variable. For example, if you wish to be notified when 1 percent of the requests experienced lock contention, specify:

```
%LET LOCKCONT = 1 ; * ACCEPTABLE PERCENT LOCK CONTENTION;
```

Chapter 3.5: PAGERATE - Acceptable page-in rate, individual pool

CPExpert sums the QBSTRPI (the number of page-ins required for read I/O) and QBSTWPI (the number of page-ins required for write I/O) in DB2STATB. The sum is divided by the DB2 statistics interval to yield a page-in rate per second. CPExpert then compares this page-in rate with the **PAGERATE** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-216 when the number of page-ins for read I/O and write I/O exceeds the value specified by the **PAGERATE** guidance variable.

The default value for the **PAGERATE** guidance variable is 10, indicating that CPExpert should produce Rule DB2-216 whenever page faults for read I/O and write I/O exceed 10 page-ins per second. You can alter the analysis by specifying a different value for the acceeptable page-in rate. For example, if you wish to be notified whenever more than 50 page-ins per second occur, specify:

```
%LET PAGERATE  = 50 ; * ACCEPTABLE PAGE-IN RATE;
```

Chapter 3.6: PCTCBA - Percent active buffers

CPExpert computes the percent of buffers that are active at the end of a DB2 statistics interval. The calculation uses data in DB2STATB, and is performed as shown below:

$$\textit{Percent buffers active} = \frac{\textit{Buffers active at end of DB2 statistics interval}}{\textit{Bufferss allocated to buffer pool}}$$

CPExpert compares the computed percent with the **PCTCBA** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-209 when the percent buffers active is **less than** the value specified by the **PCTCBA** guidance variable.

The default value for the **PCTCBA** guidance variable is 50%, indicating that Rule DB2-209 should be produced when less than 50% of the buffers in the buffer pool are active. You can alter the analysis by specifying a different value (and you can override the analysis completely by specifying **%LET PCTCBA = 100;** for the guidance). For example, if you wish to be notified when less than 25% of the buffers in the buffer pool are active, specify:

```
%LET PCTCBA = 25%; * PERCENT ACTIVE BUFFERS IN BUFFER POOL;
```

WARNING: The information reported in the DB2 statistics is only a "snap-shot" of the status of the buffer pool when the statistics were written. Consequently, the number of active buffers can be considered only an indication of the use of the buffer pool. The active buffers value should be considered a "sample" of the status of the buffer pool. The values should be tracked over time to see whether they consistently are a large percent of the total buffers assigned to the buffer pool.

Chapter 3.7: PCTCTL - Percent CT sections not found

CPExpert computes the "hit ratio" of Cursor Table requests that found the Cursor Table in the EDM pool. The calculation is uses data in DB2STATS, and is performed as shown below:

$$Hit\ ratio = \frac{Requests\ for\ CTs - CTs\ loaded\ DASD}{Requests\ for\ CTs}$$

CPExpert compares the computed hit ratio with the **PCTCTL** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-263 when the hit ratio is **less than** the value specified by the **PCTCTL** guidance variable.

The default value for the **PCTCTL** guidance variable is 95%, indicating that Rule DB2-263 should be produced when the hit ratio for cursor table sections in the EDM pool is less than 95%.

IBM suggests that a hit ratio of 80% would be acceptable in most situations in small to medium sites. However, a higher hit ratio would be required to achieve good performance in an important system with a high transaction volume. **Consequently, CPExpert has set the default to be higher than IBM's recommendation, with the expectation that you will lower the guidance if your DB2 system is not processing important applications with a high volume of transactions.** Please note that this high hit ratio would not be achieved when DB2 first starts, but should be achieved after DB2 reaches a stable state.

©Copyright 1999, Computer Management Sciences, Inc.

The default value for the **PCTCTL** guidance variable is 95%, indicating that Rule DB2-264 would be produced when the hit ratio for CTs in the EDM pool is less than 95%. You can alter the analysis by specifying a different value (and you can override the analysis completely by specifying **%LET PCTCTL = 100;** for the guidance). For example, if you wish to be notified when less than 80% of the cursor table sections were found in the EDM pool, specify:

```
%LET PCTCTL = 80%; * PERCENT CURSOR TABLE SECTIONS FOUND;
```

Chapter 3.8: PCTDBDL - Percent DBD sections not found

CPExpert computes the “hit ratio” of Data Base Descriptor requests that found the Data Base Descriptors in the EDM pool. The calculation is uses data in DB2STATS, and is performed as shown below:

$$Hit\ ratio = \frac{Requests\ for\ DBDs - DBDs\ loaded\ DASD}{Requests\ for\ DBDs}$$

CPExpert compares the computed hit ratio with the **PCTDBDL** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-262 when the hit ratio is **less than** the value specified by the **PCTDBDL** guidance variable.

The default value for the **PCTDBDL** guidance variable is 99%, indicating that CPExpert should produce Rule DB2-262 when the hit ratio for DBDs in the EDM pool is less than 99%.

IBM suggests that a hit ratio of 80%-90% would be acceptable in most situations in small to medium sites. However, a higher hit ratio would be required to achieve good performance in an important system with a high transaction volume. **Consequently, CPExpert has set the default to be very high, with the expectation that you will lower the guidance if your DB2 system is not processing important applications with a high volume of transactions.** Please note that this high hit ratio would not be achieved when DB2 first starts, but should be achieved after DB2 reaches a stable state.

The default value for the **PCTDBDL** guidance variable is 95%, indicating that Rule DB2-264 would be produced when the hit ratio for CTs in the EDM pool is less than 95%. You can alter the analysis by specifying a different value (and you can override the analysis completely by specifying **%LET PCTDBDL = 100;** for the guidance). For example, if you wish to be notified when less than 85% of the data base descriptor sections were found in the EDM pool, specify:

```
%LET PCTDBDL = 85%; * PERCENT DATA BASE DESCRIPTOR  SECTIONS FOUND;
```

Chapter 3.9: PCTDPF - Percent dynamic sequential prefetch

CPExpert computes the percent of total sequential prefetch operations that were as a result of sequential detection. The calculation is uses data in DB2STATB, and is performed as shown below:

$$\textit{Percent dynamic sequential prefetch} = \frac{\textit{Dynamic sequential prefetch}}{\textit{Dynamic sequential prefetch} + \textit{Sequential prefetch}}$$

CPExpert produces Rule DB2-235 when the percent of requests that were dynamic sequential prefetch requests exceeds the value specified by the **PCTDPF** guidance variable.

The default value for the **PCTDPF** guidance variable is 0, indicating that CPExpert should produce Rule DB2-235 whenever dynamic sequential prefetch was implemented. **Please note that this low threshold was selected only to alert you to the performance issue. You may wish to adjust the threshold for a particular buffer pool after you considered the alternatives listed below.**

You can alter the analysis by specifying a different value (and you can override the analysis completely by specifying **%LET PCTDPF = 100;** for the guidance). For example, if you wish to be notified when more than 25% of the sequential prefetch operations were as a result of dynamic sequential detection, specify:

```
%LET PCTDPF = 25 ; * PERCENT DYNAMIC SEQUENTIAL PREFETCH ;
```

Chapter 3.10: PCTEDM- Percent free pages in EDM pool

CPExpert computes the percent of unused (or “free”) pages in the EDM pool. The calculation is:

$$\textit{Percent free pages in EDM pool} = \frac{\textit{Number of free EDM pool pages}}{\textit{Total pages in EDM pool}}$$

CPExpert compares the computed percent free pages in EDM pool with the **PCTEDM** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-265 when

the computed percent free pages in EDM pool exceeds the value specified by the **PCTEDM** guidance variable.

The default value for the **PCTEDM** guidance variable is 25%, indicating that CPExpert should produce Rule DB2-265 when the percent free pages in EDM pool is greater than 25%. You can alter the analysis by specifying a different value (and you can override the analysis completely by specifying **%LET PCTEDM = 100;** for the guidance). For example, if you wish to be notified when more than 50% of the EDM pool is free, specify:

```
%LET PCTEDM = 50 % * PERCENT FREE PAGES IN EDM POOL;
```

Chapter 3.11: PCTESC - Percent lock escalation not effective

CPExpert computes the percent of lock escalations that were not effective, by computing an escalation effectiveness metric. The calculation is uses data in DB2STATS, and is performed as shown below:

$$\textit{Escalation effectiveness} = \frac{\textit{Lock timeouts} + \textit{Deadlocks}}{\textit{Escalations to shared mode} + \textit{Escalations to exclusive mode}}$$

CPExpert compares the computed effectiveness metric with the **PCTESC** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-305 when the escalation effectiveness is **less than** the value specified by the **PCTESC** guidance variable.

The default value for the **PCTESC** guidance variable is 25%, indicating that CPExpert should produce Rule DB2-305 when the escalation effectiveness is less than 25%. You can alter the analysis by specifying a different value (and you can override the analysis completely by specifying **%LET PCTFLAM = 100;** for the guidance). For example, if you wish to be notified when less than 10% of the escalations were effective, specify:

```
%LET PCTESC = 10 % * PERCENT LOCK ESCALATION WAS NOT EFFECTIVE;
```

Chapter 3.12: PCTFLAM - Percent failed look-ahead tape mounts

CPExpert computes the percent of failed look-ahead tape mounts. The calculation is uses data in DB2STATS, and is performed as shown below:

$$\textit{Percent failed look-ahead mounts} = \frac{(\textit{Look-ahead tape mounts attempted}) - (\textit{Successful look-ahead tape mounts})}{\textit{Look-ahead tape mounts attempted}}$$

The default value for the **PCTFLAM** guidance variable is 0%, indicating that Rule DB2-515 would be produced when any look-ahead tape mount attempts failed. You can alter the analysis by specifying a different value (and you can override the analysis completely by specifying **%LET PCTFLAM = 100;** for the guidance). For example, if you wish to be notified when more than 25% of the look-ahead tape mounts failed, specify:

```
%LET PCTFLAM = 25 ; * PERCENT FAILED LOOK-AHEAD TAPE MOUNTS;
```

Chapter 3.13: PCTESC - Percent lock escalation not effective

CPExpert computes the percent of lock escalations that were not effective, by computing an escalation effectiveness metric. The calculation is uses data in DB2STATS, and is performed as shown below:

$$\textit{Escalation effectiveness} = \frac{\textit{Lock timeouts} + \textit{Deadlocks}}{\textit{Escalations to shared mode} + \textit{Escalations to exclusive mode}}$$

CPExpert compares the computed effectiveness metric with the **PCTESC** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-305 when the escalation effectiveness is **less than** the value specified by the **PCTESC** guidance variable.

The default value for the **PCTESC** guidance variable is 25%, indicating that CPExpert should produce Rule DB2-305 when the escalation effectiveness is less than 25%. You can alter the analysis by specifying a different value (and you can override the analysis completely by specifying **%LET PCTFLAM = 100;** for the guidance). For example, if you wish to be notified when less than 10% of the escalations were effective, specify:

```
%LET PCTESC = 10 % * PERCENT LOCK ESCALATION WAS NOT EFFECTIVE;
```

Chapter 3.14: PCTHITGB - Percent read hit cross-invalidated pages

CPExpert computes a read hit percentage for **cross-invalidated** pages, based on the following algorithm:

$$read\ hit\ percentage = \frac{QBGLXD}{QBGLXD + QBGLXR + QBGLXN}$$

where:

QBGLXD = Reads due to cross-invalidation, data returned. This means that the page was in the group buffer pool and DB2 did not have refresh the page from DASD. This is the ideal situation.

QBGLXR = Reads due to cross-invalidation, data not returned, directory entry existed or created. This means that the page was not in the group buffer pool, but another DB2 still had read/write interest in the page set or partition. The page had to be refreshed from DASD. This is an undesirable read-miss condition.

QBGLXN = Reads due to cross-invalidation, data not returned, directory entry not created. This means that the page was not in the group buffer pool, but no other DB2 currently had a read/write interest in the page set or partition. Normally, there would be few of these conditions, since reverting between read/write interest and no read/write interest should happen infrequently.

CPExpert compares the computed read hit percentage for cross-invalidated pages with the **PCTHITGB** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-603 when the computed read hit percentage for cross-invalidated pages is less than the value specified by the **PCTHITGB** guidance variable.

The default value for the **PCTHITGB** guidance variable is 95, indicating that CPExpert should produce Rule DB2-603 when less than ninety-five percent of the reads for cross-invalidated buffers did not find the page in the group buffer pool.

IBM suggests that a hit ratio of 80% would be acceptable in most situations in small to medium sites. However, a higher hit ratio would be required to achieve good performance in an important system with a high transaction volume. **Consequently, CPExpert has set the default to be higher than IBM's recommendation, with the expectation that you will lower the guidance if your DB2 system is not processing important applications with a high volume of transactions.** Please note that this high hit ratio would not be achieved when DB2 first starts, but should be achieved after DB2 reaches a stable state.

You can alter the analysis by specifying a different value. For example, if you wish to be notified when less than eighty percent of the reads for cross-invalidated buffers did not find the page in the group buffer pool, specify:

```
%LET PCTHITGB = 80 ; * PERCENT READ HIT FOR CROSS-INVALIDATED PAGES
```

Chapter 3.15: PCTRACT - Percent reads from active log

CPExpert computes the percent of reads satisfied from the active log by the following algorithm:

$$Percent\ reads\ from\ active\ \log = \frac{Reads\ from\ active\ \log\ (QBSTRACT)}{Total\ reads\ from\ \log}$$

CPExpert compares the resultant percent) with the **PCTRACT** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-512 when the percent of reads satisfied from the active log exceeds the value specified by the **PCTRACT** guidance variable.

The default value for the **PCTRACT** guidance variable is 25 indicating that CPExpert should produce Rule DB2-512 when more than 25% of the reads were satisfied from the active log. You can alter the analysis by specifying a different value (and you can override the analysis completely by specifying **%LET PCTRACT = 100;** for the guidance). For example, if you wish to be notified when more than 50% of the reads were satisfied from the active log per DB2 statistics interval, specify:

```
%LET PCTRACT= 50 ; * PERCENT READ SATISFIED FROM ACTIVE LOG;
```

Chapter 3.16: PCTRAND - Percent GETPAGE from random requestors

CPExpert computes the percent of GETPAGE requests that were from random requestors. The calculation is uses data in DB2STATB, and is performed as shown below:

$$Percent\ random\ GETPAGE\ requests = \frac{GETPAGE\ requests - Sequential\ GETPAGE\ requests}{GETPAGE\ requests}$$

CPExpert compares the computed percent of GETPAGE requests that were issued by random requestors with the **PCTRAND** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-224 when the percent of GETPAGE requests that were issued by random requestors is **greater than** the value specified by the **PCTRAND**

guidance variable, and the value specified for VPSEQT threshold for the buffer pool (obtained from the QDBPVPSH variable in DB2STAT2) is **greater than 10%**.

The default value for the **PCTRAND** guidance variable is 95%, indicating that Rule DB2-224 should be produced when the percent of GETPAGE requests that were issued by random requestors was **greater than 95%**, and the value for the VPSEQT threshold is **greater than 10%**. You can alter the analysis by specifying a different value (and you can override the analysis completely by specifying **%LET PCTRAND = 100;** for the guidance). For example, if you wish to be notified only when more than 75% of the GETPAGE requests were from random requestors and the VPSEQT was greater than 10%, specify:

```
%LET PCTRAND = 50 ; * PERCENT GETPAGE FROM RANDOM REQUESTORS;
```

Chapter 3.17: PCTRSAT - Percent reconnection attempts succeeded

CPExpert computes the percent of reconnection attempts that succeeded. The calculation uses data in DB2STATB, and is performed as shown below:

$$\textit{Percent successful reconnection attempts} = \frac{\textit{Successful reconnection attempts}}{\textit{Reconnection attempts}}$$

CPExpert compares the computed percent successful reconnection attempts with the **PCTRSAT** guidance variable in USOURCE(DB2GUIDE). Rule DB2-402 is produced when the percent of reconnection attempts that succeeded is **less than** the value specified by the **PCTRSAT** guidance variable.

The default value for the **PCTRSAT** guidance variable is 95%, indicating that CPExpert should produce Rule DB2-402 when the percent reconnection attempts that succeeded was **less than 95%** of the reconnection attempts. You can alter the analysis by specifying a different value (and you can override the analysis completely by specifying **%LET PCTRSAT = 0;** for the guidance). For example, if you wish to be notified when less than 75% of the reconnection attempts succeeded, specify:

```
%LET PCTRSAT = 75; * PERCENT SUCCESSFUL RECONNECTION ATTEMPTS;
```

Chapter 3.18: PCTSEQ - Percent GETPAGE, sequential requestors

CPExpert computes the percent of GETPAGE requests that were from sequential requestors. The calculation is uses data in DB2STATB, and is performed as shown below:

$$\textit{Percent sequential GETPAGE requests} = \frac{\textit{Sequential GETPAGE requests}}{\textit{GETPAGE requests}}$$

CPExpert compares the computed percent of GETPAGE requests that were issued by sequential requestors with the **PCTSAT** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-223 when the percent of GETPAGE requests that were issued by sequential requestors is **greater than** the value specified by the **PCTSEQ** guidance variable, and the value specified for VPSEQT threshold for the buffer pool (obtained from the QDBPVPSH variable in DB2STAT2) is **less than 99%**.

The default value for the **PCTSEQ** guidance variable is 95%, indicating that Rule DB2-223 should be produced the percent of GETPAGE requests that were issued by sequential requestors was **greater than 95%**, and the value for the VPSEQT threshold is **less than 99%**. You can alter the analysis by specifying a different value (and you can override the analysis completely by specifying **%LET PCTSEQ = 100;** for the guidance). For example, if you wish to be notified only when more than 85% of the GETPAGE requests were from sequential requestors and the VPSEQT was less than 90%, specify:

```
%LET PCTSEQ = 85 ; * PERCENT GETPAGE FROM SEQUENTIAL REQUESORS;
```

Chapter 3.19: PCTSLOCK - Percent suspends for lock conflict

CPExpert divides the QTXASLOC variable in DB2STATS (the number of suspends for lock conflict) by the QTXALOCK variable in DB2STATS (the lock request count). CPExpert compares the resulting percent of lock requests that were suspended for lock conflict with the **PCTSLO0CK** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-301 when the pecent of lock requests suspended because of lock conflict exceeds the value specified by the **PCTSLOCK** guidance variable.

The default value for the **PCTSLOCK** guidance variable is 5%, indicating that CPExpert should produce Rule DB2-301 when more than 5% of lock requests were suspended for lock conflict occurred.

It is unlikely that this value is correct for your installation! While it is "ideal" that zero suspensions occur for lock conflict, the number will be highly dependent upon the applications involved.

A low value was selected simply to alert you to this consideration and to allow you to specify a value that is more relevant to your installation. You normally should alter the analysis by specifying a different value. For example, if you wish to be notified when more than 10% of the lock requests were suspended because of lock conflict occurred, per DB2 statistics interval, specify:

```
%LET PCTSLOCK = 10 ; * PERCENT SUSPENDS FOR LOCK CONFLICT;
```

Chapter 3.20: PCTXIGBP - Acceptable percent reclaims, XI pages

CPExpert computes the percentage of **cross-invalidated** pages caused by directory reclaims relative to the total directory entry reclaims, based on the following algorithm:

$$percent\ cross\ invalidated\ pages = \frac{R744CXDR}{R744CDER}$$

where:

R744CXDR = Cross-invalidate directory reclaim counter. This is the number of pages that were cross-invalidated because the corresponding directory entry had been reclaimed.

R744CDER = Directory entry reclaim counter. This is the total number of directory entries that were reclaimed.

CPExpert compares the percentage for cross-invalidated pages because of directory reclaims with the **PCTXIGBP** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-671 when the computed percentage of cross-invalidated pages because of directory reclaims is less than the value specified by the **PCTXIGBP** guidance variable.

IBM states that for **best** DB2 performance, there should no cross-invalidate actions due to directory reclaims. Consequently, the default value for the **PCTXIGBP** guidance variable is 0, indicating that CPExpert should produce Rule DB2-671 when any cross-invalidate actions were taken because of directory reclaims

You can alter the analysis by specifying a different value (and you can override the analysis completely by specifying **%LET PCTXIGBP= 100;** for the guidance). For example, if you wish to be notified only when the computed percentage of cross-invalidated pages because of directory reclaims is less than 5%, specify:

```
%LET PCTXIGBP = 5 ; *ACCEPTABLE PERCENT RECLAIMS, XI PAGES;
```

Chapter 3.21: Q3STCTHW - Times threads were queued at create

CPExpert compares the Q3STCTHW variable in DB2STATS (the number of create thread requests queued) with the **Q3STCTHW** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-101 when the number of create thread requests queued exceeds the value specified by the **Q3STCTHW** guidance variable.

The default value for the **Q3STCTHW** guidance variable is 0, indicating that CPExpert should produce Rule DB2-101 when any requests were queue at CREATE. You can alter the analysis by specifying a different value. For example, if you wish to be notified whenever more than 100 thread create requests were queued during a DB2 statistics interval, specify:

```
%LET Q3STCTHW =100 ; * TIMES THREADS WERE QUEUED AT CREATE;
```

Chapter 3.22: Q3STMEOM - End-of-memory (non-DB2 allied agent)

CPExpert compares the Q3STMEOM variable in DB2STATS (the number of times an allied task was deleted by MVS) with the **Q3STMEOM** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-122 when the number of times an allied task was deleted by MVS exceeds the value specified by the **Q3STMEOM** guidance variable.

The default value for the **Q3STMEOM** guidance variable is 0, indicating that CPExpert should produce Rule DB2-122 when any allied task was deleted by MVS. You can alter the analysis by specifying a different value. For example, if you wish to be notified whenever more than 10 allied tasks were deleted by MVS during a DB2 statistics interval, specify:

```
%LET Q3STMEOM =10 ; * END OF MEMORY (NON-DB2 ALLIED AGENT);
```

Chapter 3.23: Q3STMEOT - End-of-task (non-DB2 allied agent)

CPExpert compares the Q3STMEOT variable in DB2STATS (the number of times an allied task ABENDED) with the **Q3STMEOT** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-121 when the number of times an allied task ABENDED exceeds the value specified by the **Q3STMEOT** guidance variable.

The default value for the **Q3STMEOT** guidance variable is 0, indicating that CPExpert should produces Rule DB2-121 when any allied task ABEND occurs. You can alter the analysis by specifying a different value. For example, if you wish to be notified whenever more than 10 allied tasks ABEND during a DB2 statistics interval, specify:

```
%LET Q3STMEOT =10 ; * TASK ABEND (NON-DB2 ALLIED AGENT);
```

Chapter 3.24: QBGLAN - CF prefetch reads, no data, no directory

CPExpert compares the QBGLAN variable in DB2GBPST (the number times a coupling facility prefetch read was returned with no data, and no directory entry was created) with the **QBGLAN** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-610 when the value of the QBGLAN variable in DB2GBPST exceeds the value specified by the **QBGLAN** guidance variable.

The default value for the **QBGLAN** guidance variable is 0, indicating that CPExpert should produce Rule DB2-610 when the QBGLAN variable was non-zero. You can alter the analysis by specifying a different value. For example, if you wish to be notified when more than coupling facility prefetch read was returned with no data, and no directory entry was created more than 50 times per DB2 statistics interval, specify:

```
%LET QBGLAN = 50 ; * CF PREFETCH READS RETURNED NO DATA, NO DIRECTORY;
```

Chapter 3.25: QBGLAR - CF prefetch, no data, directory entry

CPExpert compares the QBGLAR variable in DB2GBPST (number of coupling facility reads for prefetch in which data was not returned from the coupling facility and a coupling facility directory entry was created) with the **QBGLAR** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-611 when the value of the QBGLAR variable in DB2GBPST exceeds the value specified by the **QBGLAR** guidance variable.

The default value for the **QBGLAR** guidance variable is 0, indicating that CPExpert should produce Rule DB2-221 when the QBGLAR variable was non-zero. You can alter the analysis by specifying a different value. For example, if you wish to be notified when more than coupling facility prefetch read was returned with no data, and a directory entry was created more than 50 times per DB2 statistics interval, specify:

```
%LET QBGLAR = 50 ; * CF PREFETCH READS RETURNED NO DATA, DIRECTORY CREATED;
```

Chapter 3.26: QBGLCN - Times CF castout engine not available

CPExpert compares the QBGLCN variable in DB2GBPST (the number of number of times a coupling facility castout engine was not available) with the **QBGLCN** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-605 when the value of the QBGLCN variable in DB2GBPST exceeds the value specified by the **QBGLCN** guidance variable.

The default value for the **QBGLCN** guidance variable is 0, indicating that CPExpert should produce Rule DB2-605 whenever a castout engine was not available in a DB2 statistics interval. You can alter the analysis by specifying a different value. For example, if you wish to be notified when a coupling facility castout engine was not available more than 100 times per DB2 statistics interval, specify:

```
%LET QBGLCN   = 100;   * TIMES WHEN CASTOUT ENGINE NOT AVAILABLE;
```

Chapter 3.27: QBGLRF - Times CF reads failed, lack of storage

CPExpert compares the QBGLRF variable in DB2GBPST (the number of number of times a coupling facility read failed) with the **QBGLRF** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-601 when the value of the QBGLRF variable in DB2GBPST exceeds the value specified by the **QBGLRF** guidance variable.

The default value for the **QBGLRF** guidance variable is 0, indicating that CPExpert should produce Rule DB2-601 whenever a coupling facility read failed in a DB2 statistics interval. You can alter the analysis by specifying a different value. For example, if you wish to be notified when a coupling facility read failed more than 50 times per DB2 statistics interval, specify:

```
%LET QBGLRF = 50 *CF READS FAILED DUE TO LACK OF CF STORAGE;
```

Chapter 3.28: QBGLWF - Times CF writes failed, lack of storage

CPExpert compares the QBGLWF variable in DB2GBPST (the number of number of times a coupling facility write failed) with the **QBGLWF** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-602 when the value of the QBGLWF variable in DB2GBPST exceeds the value specified by the **QBGLWF** guidance variable.

The default value for the **QBGLWF** guidance variable is 0, indicating that CPExpert should produce Rule DB2-602 whenever a coupling facility write failed in a DB2 statistics interval. You can alter the analysis by specifying a different value. For example, if you wish to be notified when a coupling facility write failed more than 50 times per DB2 statistics interval, specify:

```
%LET QBGLWF = 50 *CF WRITES FAILED DUE TO LACK OF CF STORAGE;
```

Chapter 3.29: QBGLSU - Times CF write engine was not available

CPExpert compares the QBGLSU variable in DB2GBPST (the number of number of times a coupling facility write engine was not available) with the **QBGLSU** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-606 when the value of the QBGLSU variable in DB2GBPST exceeds the value specified by the **QBGLSU** guidance variable.

The default value for the **QBGLSU** guidance variable is 0, indicating that CPExpert should produce Rule DB2-606 whenever a coupling facility write engine was not available in a DB2 statistics interval. You can alter the analysis by specifying a different value. For example, if you wish to be notified when a coupling facility write engine was not available more than 100 times per DB2 statistics interval, specify:

```
%LET QBGLSU   = 100;   * TIMES WHEN CF WRITE ENGINE WAS NOT AVAILABLE;
```

Chapter 3.30: QBSTARF - Unsuccessful HP to VP read, async ADMF

CPExpert compares the QBSTARF variable in DB2STATB (the number of unsuccessful hiperpool to virtual pool reads, using asynchronous ADMF) with the **QBSTARF** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-210 when the value

of the QBSTARF variable in DB2STATB exceeds the value specified by the **QBSTARF** guidance variable.

The default value for the **QBSTARF** guidance variable is 0, indicating that CPExpert should produce Rule DB2-210 whenever there were any unsuccessful hiperpool to virtual pool reads in a DB2 statistics interval. You can alter the analysis by specifying a different value. For example, if you wish to be notified when there were more than 100 unsuccessful hiperpool to virtual pool reads per DB2 statistics interval, specify:

```
%LET QBSTARF = 100 * UNSUCCESSFUL HP TO VP READS, USING ASYNC ADMF;
```

Chapter 3.31: QBSTAWF - Unsuccessful HP to VP write, async ADMF

CPExpert compares the QBSTAWF variable in DB2STATB (the number of unsuccessful hiperpool to virtual pool writes, using asynchronous ADMF) with the **QBSTAWF** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-211 when the value of the QBSTAWF variable in DB2STATB exceeds the value specified by the **QBSTAWF** guidance variable.
The default value for the **QBSTAWF** guidance variable is 0, indicating that CPExpert should produce Rule DB2-211 whenever there were any unsuccessful hiperpool to virtual pool writes in a DB2 statistics interval. You can alter the analysis by specifying a different value. For example, if you wish to be notified when there were more than 100 unsuccessful hiperpool to virtual pool writes per DB2 statistics interval, specify:

```
%LET QBSTAWF = 100 * UNSUCCESSFUL HP TO VP WRITES, USING ASYNC ADMF;
```

Chapter 3.32: QBSTDMC - Times Data Manager Threshold reached

CPExpert compares the QBSTDMC variable in DB2STATS (the number of times the Data Management Threshold was reached) with the **QBSTDMC** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-220 when the number of times the DMTH was reached exceeds the value specified by the **QBSTDMC** guidance variable.

The default value for the **QBSTDMC** guidance variable is 0, indicating that CPExpert should produce Rule DB2-220 whenever the Data Management Threshold was reached. You can alter the analysis by specifying a different value. For example, if you wish to be notified when the Data Management Threshold was reached more than 10 times per DB2 statistics interval, specify:

```
%LET QBSTDMC = 10 ; *DATA MANAGEMENT THRESHOLD REACHED;
```

Chapter 3.33: QBSTHRF - Unsuccessful HP to VP reads (MVPG)

CPExpert compares the QBSTHRF variable in DB2STATB (the number of unsuccessful hiperpool to virtual pool reads, using MVPG) with the **QBSTHRF** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-206 when the value of the QBSTHRF variable in DB2STATB exceeds the value specified by the **QBSTHRF** guidance variable.

The default value for the **QBSTHRF** guidance variable is 0, indicating that CPExpert should produce Rule DB2-206 whenever there were any unsuccessful hiperpool to virtual pool reads (using MVPG) in a DB2 statistics interval. You can alter the analysis by specifying a different value. For example, if you wish to be notified when there were more than 100 unsuccessful hiperpool to virtual pool reads (using MVPG) per DB2 statistics interval, specify:

```
%LET QBSTHRF = 100 * UNSUCCESSFUL HP TO VP READS, USING MVPG;
```

Chapter 3.34: QBSTHWF - Unsuccessful HP to VP writes (MVPG)

CPExpert compares the QBSTHWF variable in DB2STATB (the number of unsuccessful hiperpool to virtual pool writes, using MVPG) with the **QBSTHWF** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-207 when the value of the QBSTHWF variable in DB2STATB exceeds the value specified by the **QBSTHWF** guidance variable.

The default value for the **QBSTHWF** guidance variable is 0, indicating that CPExpert should produce Rule DB2-206 whenever there were any unsuccessful hiperpool to virtual pool writes (using MVPG) in a DB2 statistics interval. You can alter the analysis by specifying a different value. For example, if you wish to be notified when there were more than 100 unsuccessful hiperpool to virtual pool writes (using MVPG) per DB2 statistics interval, specify:

```
%LET QBSTHWF = 100 * UNSUCCESSFUL HP TO VP WRITES, USING MVPG;
```

Chapter 3.35: QBSTIMW - Number of Immediate Writes

The Immediate Write Threshold (IWTH) is one of the **fixed** thresholds. The IWTH has a fixed value of 97.5%, and is checked whenever a page is to be updated. If the number of unavailable pages in a virtual buffer pool exceeds 97.5% of the size of the virtual buffer pool, the updated page is written to DASD as soon as the update completes. The write is synchronous with the SQL request; that is, the request waits until the write has been completed and the two operations are not carried out concurrently.

Reaching the IWTH threshold has a significant effect on processor usage and I/O resource consumption. IBM documentation provides an example: updating three rows per page in 10 sequential pages ordinarily requires one or two write operations. When IWTH is exceeded, however, the updates require 30 synchronous writes.

Unfortunately, it is not possible to determine whether the IWTH threshold was reached from the DB2 statistics. This is because DB2 uses synchronous (immediate) writes even when the IWTH is not exceeded.

- An immediate write might be issued when more than two checkpoints pass without a page being written. Two checkpoints passing without a page being written does not indicate a problem.

- An immediate write might be issued when DB2 detected that no deferred write engines were available. Rule DB2-234 analyzes the case where no write engine was available.

CPExpert compares the QBSTIMW variable in DB2STATB (the number of immediate writes) with the **QBSTIMW** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-221 when the number of immediate writes exceeds the value specified by the **QBSTIMW** guidance variable **AND** Rule DB2-220 has been produced.

The default value for the **QBSTIMW** guidance variable is 0, indicating that CPExpert should produce Rule DB2-221 when any immediate writes occurred. You can alter the analysis by specifying a different value. For example, if you wish to be notified when more than 50 immediate writes occurred per DB2 statistics interval, specify:

```
%LET QBSTIMW = 50 ; * NUMBER OF IMMEDIATE WRITES;
```

Chapter 3.36: QBSTJIS - Times prefetch I/O streams denied

CPExpert compares the QBSTJIS variable in DB2STATB (the number of times prefetch I/O streams were denied because of buffer availability) with the **QBSTJIS** guidance

variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-253 when the value of the QBSTJIS variable in DB2STATB exceeds the value specified by the **QBSTJIS** guidance variable.

The default value for the **QBSTJIS** guidance variable is 0, indicating that CPExpert should produce Rule DB2-253 when any prefetch I/O streams were denied because buffers were not available. You can alter the analysis by specifying a different value. For example, if you wish to be notified when more than 100 prefetch I/O streams were denied because of buffers, per DB2 statistics interval, specify:

```
%LET QBSTJIS = 100 ; * PREFETCH I/O STREAMS DENIED, NO BUFFERS;
```

Chapter 3.37: QBSTNGT - Number of unsuccessful GETPAGE

CPExpert compares the QBSTNGT variable in DB2STATB (the number unsuccessful GETPAGE requests for a parallel query) with the **QBSTNGT** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-254 when the value of the QBSTNGT variable in DB2STATB exceeds the value specified by the **QBSTNGT** guidance variable.

The default value for the **QBSTNGT** guidance variable is 0, indicating that CPExpert should produce Rule DB2-254 when any GETPAGE requests were unsuccessful. You can alter the analysis by specifying a different value. For example, if you wish to be notified when more than 100 GETPAGE requests for a parallel query were unsuccessful per DB2 statistics interval, specify:

```
%LET QBSTNGT = 100 ; * UNSUCCESSFUL GETPAGE REQUESTS;
```

Chapter 3.38: QBSTPL1 - Times prefetch reduced to ½ of normal

CPExpert compares the QBSTPL1 variable in DB2STATB (the number of times the prefetch quantity was reduced from normal to ½ of normal) with the **QBSTPL1** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-237 when the value of the QBSTPL1 variable in DB2STATB exceeds the value specified by the **QBSTPL1** guidance variable.

The default value for the **QBSTPL1** guidance variable is 0, indicating that CPExpert should produce Rule DB2-237 whenever the prefetch quantity was reduced to ½ of normal. You can alter the analysis by specifying a different value. For example, if you wish to be

notified when the prefetch quantity was reduced to ½ of normal more than 5 times per DB2 statistics interval, specify:

```
%LET QBSTPL1 = 5 ; * PREFETCH REDUCED TO ½ OF NORMAL;
```

Chapter 3.39: QBSTPL2 - Times prefetch reduced to 1/4 of normal

CPExpert compares the QBSTPL2 variable in DB2STATB (the number of times the prefetch quantity was reduced from normal to 1/4 of normal) with the **QBSTPL2** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-238 when the value of the QBSTPL2 variable in DB2STATB exceeds the value specified by the **QBSTPL2** guidance variable.

The default value for the **QBSTPL2** guidance variable is 0, indicating that CPExpert should produce Rule DB2-238 whenever the prefetch quantity was reduced to 1/4 of normal. You can alter the analysis by specifying a different value. For example, if you wish to be notified when the prefetch quantity was reduced to 1/4 of normal more than 5 times per DB2 statistics interval, specify:

```
%LET QBSTPL2 = 5 ; * PREFETCH REDUCED TO ½ OF NORMAL;
```

Chapter 3.40: QBSTPQF - Times I/O parallelism was downgraded

CPExpert compares the QBSTPQF variable in DB2STATB (the number of times that I/O parallelism was downgraded because of a lack of buffers) with the **QBSTPQF** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-251 when the value of the QBSTPQ variable in DB2STATB exceeds the value specified by the **QBSTPQF** guidance variable.

The default value for the **QBSTPQF** guidance variable is 0, indicating that CPExpert should produce Rule DB2-251 whenever I/O parallelism was downgraded because of lack of buffers. You can alter the analysis by specifying a different value. For example, if you wish to be notified when I/O parallelism was downgraded more than 5 times per DB2 statistics interval, specify:

```
%LET QBSTPQF = 5 ; * TIMES I/O PARALLELISM WAS DOWNGRADED;
```

Chapter 3.41: QBSTREE - Times prefetch disabled, no read engine

CPExpert compares the QBSTREE variable in DB2STATB (the number of times that prefetch was disabled because no read engine was available) with the **QBSTREE** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-231 when the value of the QBSTPQ variable in DB2STATB exceeds the value specified by the **QBSTREE** guidance variable.

The default value for the **QBSTREE** guidance variable is 0, indicating that CPExpert should produce Rule DB2-231 whenever prefetch was disabled because no read engine was available. You can alter the analysis by specifying a different value. For example, if you wish to be notified when prefetch was disabled because no read engine was available, more than 5 times per DB2 statistics interval, specify:

```
%LET QBSTREE = 5 ; * TIMES PREFETCH DISABLED, NO READ ENGINE;
```

Chapter 3.42: QBSTSPD - Times prefetch disabled, no buffers

CPExpert compares the QBSTSPD variable in DB2STATB (the number of times that prefetch was disabled because no buffers were available) with the **QBSTSPD** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-230 when the value of the QBSTPQ variable in DB2STATB exceeds the value specified by the **QBSTSPD** guidance variable.

The default value for the **QBSTSPD** guidance variable is 0, indicating that CPExpert should produce Rule DB2-230 whenever prefetch was disabled because no buffers were available. You can alter the analysis by specifying a different value. For example, if you wish to be notified when prefetch was disabled because no buffers were available, more than 5 times per DB2 statistics interval, specify:

```
%LET QBSTSPD = 5 ; * TIMES PREFETCH DISABLED, NO BUFFERS;
```

Chapter 3.43: QBSTWEE - Times write engine was not available

CPExpert compares the QBSTWEE variable in DB2STATB (the number of times that prefetch was disabled because no write engine was available) with the **QBSTWEE** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-231 when the value of the QBSTPQ variable in DB2STATB exceeds the value specified by the **QBSTWEE** guidance variable.

The default value for the **QBSTWEE** guidance variable is 0, indicating that CPExpert should produce Rule DB2-231 whenever prefetch was disabled because no write engine was available. You can alter the analysis by specifying a different value. For example, if you wish to be notified when prefetch was disabled because no write engine was available, more than 5 times per DB2 statistics interval, specify:

```
%LET QBSTWEE = 5 ; * TIMES PREFETCH DISABLED, NO WRITE ENGINE;
```

Chapter 3.44: QBSTWFD - Times work files denied during sort

CPExpert compares the QBSTWFD variable in DB2STATB (the number of times that work files were denied during sort/merge operation) with the **QBSTWFD** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-205 when the value of the QBSTPQ variable in DB2STATB exceeds the value specified by the **QBSTWFD** guidance variable.

The default value for the **QBSTWFD** guidance variable is 0, indicating that CPExpert should produce Rule DB2-205 whenever work files were denied during sort/merge. You can alter the analysis by specifying a different value. For example, if you wish to be notified when work files were denied during sort/merge, more than 5 times per DB2 statistics interval, specify:

```
%LET QBSTWPD = 5 ; * TIMES WORK FILES DENIED DURING SORT/MERGE;
```

Chapter 3.45: QBSTWFF - Times sort was inefficient, buffer shortage

The QBSTWFF variable in the DB2 buffer statistics contains a count of the number of times that the merge pass was not efficiently performed because of a shortage of buffer space. This counter is incremented for each merge pass when the maximum number of work files allowed is less than the number of work files requested.

CPExpert compares the QBSTWFF variable in DB2STATB with the **QBSTWFF** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-203 when the times that the merge pass was not efficiently performed because of a shortage of buffer space exceeds the value specified by the **QBSTWFF** guidance variable.

The default value for the **QBSTWFF** guidance variable is 0, indicating that CPExpert should produce Rule DB2-203 whenever a merge pass was not efficiently performed

because of a shortage of buffer space. You can alter the analysis by specifying a different value. For example, if you wish to be notified when a merge pass was not efficiently performed because of a shortage of buffer space, more than 5 times per DB2 statistics interval, specify:

```
%LET QBSTWFF = 5 ; * NUMBER OF INEFFICIENT MERGE PASSES;
```

Chapter 3.46: QBSTWFR - Number of sort merge passes

CPExpert examines the QBSTWFR variable (the number of sort merge passes) in the DB2STATB statistics. When the value of this variable exceeds the **QBSTWFR** guidance variable in USOURCE(DB2GUIDE), CPExpert concludes that the buffer pool is heavily used for sort activity. CPExpert produces Rule DB2-226 when (1) buffer pools are heavily used by sort activity, and (2) the DWQT threshold is less than 90 and VDWQT threshold is less than 90. This finding means that the DWQT, VDWQT, or VPSEQT thresholds might be too small.

The default value for the **QBSTWFR** guidance variable is 1, indicating that CPExpert should conclude that the buffer pool is used exclusively (or heavily) for sort processing when merge passes occurred. **It is unlikely that this value is correct for every buffer pool in your installation!** A low value was selected simply to alert you to this consideration and to allow you to specify a value that is more relevant to the buffer pools in your installation. You normally should alter the analysis by specifying a different value. For example, if you wish CPExpert to consider a buffer pool heavily used for sort operations when there are 50 sort merge passes per DB2 statistics interval, specify:

```
%LET QBSTWFR = 50 ; * NUMBER OF SORT/MERGE PASSES;
```

Chapter 3.47: QBSTWIO - Pages written to DASD per write I/O

CPExpert computes the average pages written per asynchronous write by dividing QBSTWIO (asynchronous writes) into QBSTPWS (number of pages written). The resulting average pages written per asynchronous write activity is compared against the **QBSTWIO** guidance variable in USOURCE(DB2GUIDE).

Additionally, CPExpert examines the QBSTWFR variable (the number of merge passes) in the DB2STATB statistics. When the value of this variable exceeds the **QBSTWFR** guidance variable in USOURCE(DB2GUIDE), CPExpert concludes that the buffer pool is

heavily used for sort activity. CPExpert suppresses Rule DB2-225 for buffer pools that are heavily used for sort activity. Please refer to Rule DB2-226 for further information.

Additionally, CPExpert examines theQDBPVDQT variable (the value of the VDWQT parameter) in DB2STAT2. CPExpert suppresses Rule DB2-225 if this value is zero, as DB2 will initiate write I/O whenever 32 buffers are used for a single data set for updated pages.

CPExpert produces Rule DB2-225 when the average pages written per asynchronous write activity exceeds the value specified by the **QBSTWIO** guidance variable. The default value for the **QBSTWIO** guidance variable is 20, indicating that CPExpert should produce Rule DB2-225 when more than 20 pages were written per asynchronous write activity. You can alter the analysis by specifying a different value. For example, if you wish to be notified when more than 30 pages were written per asynchronous I/O operation, per DB2 statistics interval, specify:

```
%LET QBSTWIO = 30 ; * NUMBER OF PAGES PER ASYNC I/O;
```

Chapter 3.48: QBSTWKPD - Prefetch canceled, prefetch quantity = 0

CPExpert compares the QBSTWKPD variable in DB2STATB (the number of times that prefetch was canceled because the prefetch quantity was zero) with the **QBSTWKPD** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-230 when the value of the QBSTPQ variable in DB2STATB exceeds the value specified by the **QBSTWKPD** guidance variable.

The default value for the **QBSTWKPD** guidance variable is 0, indicating that CPExpert should produce Rule DB2-230 whenever prefetch was canceled because the prefetch quantity was zero. You can alter the analysis by specifying a different value. For example, if you wish to be notified when prefetch was canceled because the prefetch quantity was zero, more than 20 times per DB2 statistics interval, specify:

```
%LET QBSTWKPD = 20 ; * TIMES PREFETCH CANCELED, QUANTITY=0;
```

Chapter 3.49: QDSTQCRT - Conversations deallocated, ZPARMS limit

CPExpert compares the QDSTQCRT variable in DB2STATS (the number of times that conversations were deallocated due to ZPARMS limit) with the **QDSTQCRT** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-422 when the value

of the QDSTQCRT variable in DB2STATS exceeds the value specified by the **QDSTQCRT** guidance variable.

The default value for the **QDSTQCRT** guidance variable is 0, indicating that CPExpert should produce Rule DB2-422 whenever conversations were deallocated due to ZPARMS limit. You can alter the analysis by specifying a different value. For example, if you wish to be notified when conversations were deallocated due to ZPARMS limit, more than 5 times per DB2 statistics interval, specify:

```
%LET QDSTQCRT = 5 ; * CONVERSATIONS DEALLOCATED, ZPARMS LIMIT;
```

Chapter 3.50: QDSTQDBT - DB2 access queued, max remote threads

CPExpert compares the QDSTQDBT variable in DB2STATS (the number of times that database access threads were queued due to maximum remote concurrent threads specified in ZPARMS) with the **QDSTQDBT** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-431 when the value of the QDSTQDBT variable in DB2STATS exceeds the value specified by the **QDSTQDBT** guidance variable.

The default value for the **QDSTQDBT** guidance variable is 0, indicating that CPExpert should produce Rule DB2-431 whenever database access threads were queued due to limit of maximum remote concurrent threads. You can alter the analysis by specifying a different value. For example, if you wish to be notified when database access threads were queued due to limit of maximum remote concurrent threads, more than 5 times per DB2 statistics interval, specify:

```
%LET QDSTQDBT = 5 ; * NUMBER OF IMMEDIATE WRITES;
```

Chapter 3.51: QISEFAIL - Times failures caused by full EDM pool

CPExpert compares the QISEFAIL variable in DB2STATS (the number of failures caused by full EDM pool) with the **QISEFAIL** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-261 when the value of the QISEFAIL variable in DB2STATS exceeds the value specified by the **QISEFAIL** guidance variable.

The default value for the **QISEFAIL** guidance variable is 0, indicating that CPExpert should produce Rule DB2-261 whenever failures were caused by full EDM pool. You can alter the analysis by specifying a different value. For example, if you wish to be notified when more than 10 failures were caused by full EDM pool, per DB2 statistics interval, specify:

```
%LET QISEFAIL = 10 ; * NUMBER OF FAILURES, FULL EDM POOL ;
```

Chapter 3.52: QISTCOLS - Times invalid SELECT procedure

CPExpert compares the QISTCOLS variable in DB2STATS (the number of invalid select procedures) with the **QISTCOLS** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-120 when the number of invalid select procedures exceeds the value specified by the **QISTCOLS** guidance variable.

The default value for the **QISTCOLS** guidance variable is 0, indicating that CPExpert should produce Rule DB2-120 when any invalid select procedures were encountered. You can alter the analysis by specifying a different value. For example, if you wish to be notified when more than 50 invalid select procedures were encountered, per DB2 statistics interval, specify:

```
%LET QISECOLS = 50 ; * NUMBER OF INVALID SELECT PROCEDURES ;
```

Chapter 3.53: QISTRLLM - Times RID terminated, over calculated limit

CPExpert compares the QISTRLLM variable in DB2STATS with the **QISTRLLM** guidance variable in USOURCE(DB2GUIDE). The QISTRLLM value contains a count of the number of times RID list was terminated because the number of RID entries was greater than the limit of MAX(25% of table size, number of RIDs that can fit into the guaranteed number of RID blocks). CPExpert produces Rule DB2-181 when the value of the QISTRLLM variable in DB2STATS exceeds the value specified by the **QISTRLLM** guidance variable.

The default value for the **QISTRLLM** guidance variable is 0, indicating that CPExpert should produce Rule DB2-181 whenever processing of the RID list is terminated because the number of RID entries was greater than the limit of MAX(25% of table size, number of RIDs that can fit into the guaranteed number of RID blocks). You can alter the analysis by specifying a different value. For example, if you wish to be notified when RID list was terminated more than 50 times per DB2 statistics interval, specify:

```
%LET QISTRLLM = 50; RID TERMINATED, CALCULATED LIMIT;
```

Chapter 3.54: QISTRMAX - Times RID terminated, maximum storage

CPExpert compares the QISTRMAX variable in DB2STATS with the **QISTRMAX** guidance variable in USOURCE(DB2GUIDE). The QISTRMAX value contains a count of the number of times RID list was terminated because the RID had exceeded maximum allowed storage. CPExpert produces Rule DB2-184 when the value of the QISTRMAX variable in DB2STATS exceeds the value specified by the **QISTRMAX** guidance variable.

The default value for the **QISTRMAX** guidance variable is 0, indicating that CPExpert should produce Rule DB2-184 whenever processing of the RID list is terminated because the RID had exceeded maximum allowed storage. You can alter the analysis by specifying a different value. For example, if you wish to be notified when more than RID list was terminated more than 50 times, because the RID had exceeded maximum allowed storage, per DB2 statistics interval, specify:

```
%LET QISTRMAX = 50; RID TERMINATED, MAXIMUM STORAGE;
```

Chapter 3.55: QISTRPLM - Times RID terminated, over physical limit

CPExpert compares the QISTRPLM variable in DB2STATS with the **QISTRPLM** guidance variable in USOURCE(DB2GUIDE). The QISTRPLM value contains a count of the number of times RID list was terminated because the number of RID entries was greater than the physical limit of the RID pool. CPExpert produces Rule DB2-182 when the value of the QISTRPLM variable in DB2STATS exceeds the value specified by the **QISTRPLM** guidance variable.

The default value for the **QISTRPLM** guidance variable is 0, indicating that CPExpert should produce Rule DB2-182 whenever processing of the RID list is terminated because the number of RID entries was greater than the physical limit of the RID pool. You can alter the analysis by specifying a different value. For example, if you wish to be notified when more than RID list was terminated more than 50 times, because the number of RID entries was greater than the physical limit of the RID pool, per DB2 statistics interval, specify:

```
%LET QISTRPLM = 50; RID TERMINATED, PHYSICAL LIMIT;
```

©Copyright 1999, Computer Management Sciences, Inc.

Chapter 3.56: QISTRSTG - Times RID terminated, insufficient storage

CPExpert compares the QISTRSTG variable in DB2STATS (the number of times RID list processing exhausted virtual storage) with the **QISTRSTG** guidance variable in USOURCE(DB2GUIDE).

CPExpert produces Rule DB2-183 when the number of times RID list processing exhausted virtual storage exceeds the value specified by the **QISTRSTG** guidance variable.

The default value for the **QISTRSTG** guidance variable is 0, indicating that CPExpert should produce Rule DB2-183 whenever RID list processing exhausted virtual storage. You can alter the analysis by specifying a different value. For example, if you wish to be notified when RID list exhausted storage more than 10 times per DB2 statistics interval, specify:

```
%LET QISTRSTG =10 ; * RID TERMINATED, INSUFFICIENT STORAGE;
```

Chapter 3.57: QJSTALR - Number of archive log read allocations

CPExpert compares the QJSTALR variable in DB2STATS (the number of archive log read allocations) with the **QJSTALR** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-501 when the number of archive log read allocations exceeds the value specified by the **QJSTALR** guidance variable.

The default value for the **QJSTALR** guidance variable is 0, indicating that CPExpert should produce Rule DB2-501 whenever archive log read allocations occurred. You can alter the analysis by specifying a different value. For example, if you wish to be notified when more than 10 archive log read allocations occurred per DB2 statistics interval, specify:

```
%LET QJSTALR = 10 ; * NUMBER OF ARCHIVE LOG READ ALLOCATIONS;
```

Chapter 3.58: QJSTALW - Number of archive log write allocations

CPExpert compares the QJSTALW variable in DB2STATS (the number of archive log write allocations) with the **QJSTALW** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-502 when the number of archive log write allocations exceeds the value specified by the **QJSTALW** guidance variable.

©Copyright 1999, Computer Management Sciences, Inc.

The default value for the **QJSTALW** guidance variable is 0, indicating that CPExpert should produce Rule DB2-502 whenever archive log write allocations occurred. You can alter the analysis by specifying a different value. For example, if you wish to be notified when more than 10 archive log write allocations occurred per DB2 statistics interval, specify:

```
%LET QJSTALW= 10 ; * NUMBER OF ARCHIVE LOG WRITE ALLOCATIONS;
```

Chapter 3.59: QJSTRARH - Number of reads from archive log

CPExpert compares the QJSTRARH variable in DB2STATS (the number of reads satisfied from the archive log) with the **QJSTRARH** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-513 when the number of reads satisfied from the archive log exceeds the value specified by the **QJSTRARH** guidance variable.

The default value for the **QJSTRARH** guidance variable is 0, indicating that CPExpert should produce Rule DB2-513 whenever reads were satisfied from the archive log. You can alter the analysis by specifying a different value. For example, if you wish to be notified when more than 10 reads were satisfied from the archive log per DB2 statistics interval, specify:

```
%LET QJSTRARH= 10 ; * NUMBER OF READ SATISFIED FROM ARCHIVE LOG;
```

Chapter 3.60: QJSTTVC - Number of reads delayed, tape volume

CPExpert compares the QJSTTVC variable in DB2STATS (the number of reads delayed because of tape volume contention) with the **QJSTTVC** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-522 when the number of reads satisfied from the active log exceeds the value specified by the **QJSTTVC** guidance variable.

The default value for the **QJSTTVC** guidance variable is 0, indicating that CPExpert should produce Rule DB2-522 whenever reads were delayed because of tape volume contention. You can alter the analysis by specifying a different value. For example, if you wish to be notified when more than 100 reads were delayed because of tape volume contention, per DB2 statistics interval, specify:

```
%LET QJSTTVC= 100 ; * NUMBER OF READ DELAYED, TAPE CONTENTION;
```

Chapter 3.61: QJSTWTB - Number of waits for output log buffer

CPExpert compares the QJSTWTB variable in DB2STATS (the number of waits for output log buffer) with the **QJSTWTB** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-511 when the number of waits for output log buffer exceeds the value specified by the **QJSTWTB** guidance variable.

The default value for the **QJSTWTB** guidance variable is 0, indicating that CPExpert should produce Rule DB2-511 when any waits for output log buffer occurred. You can alter the analysis by specifying a different value. For example, if you wish to be notified when more than 50 waits for output log buffer occurred, per DB2 statistics interval, specify:

```
%LET QBSTWTB = 50 ; * WAITS FOR OUTPUT LOG BUFFER;
```

Chapter 3.62: QJSTWUR - Number of read delays, resources

CPExpert compares the QJSTWUR variable in DB2STATS (the number of read accesses delayed because of unavailable resources) with the **QJSTWUR** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-521 when the number of waits for output log buffer exceeds the value specified by the **QJSTWUR** guidance variable.

The default value for the **QJSTWUR** guidance variable is 0, indicating that CPExpert should produce Rule DB2-521 when any read accesses were delayed because of unavailable resources. You can alter the analysis by specifying a different value. For example, if you wish to be notified when more than 50 read accesses were delayed because of unavailable resources, per DB2 statistics interval, specify:

```
%LET QBSTIMW = 50 ; * NUMBER OF IMMEDIATE WRITES;
```

Chapter 3.63: QLSTCNVQ - Conversation requests queued by DDF

CPExpert compares the QLSTCNVQ variable in DB2STATR (the number of conversation requests queued by DDF waiting allocation) with the **QLSTCNVQ** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-421 when the number of

conversation requests queued by DDF waiting allocation exceeds the value specified by the **QLSTCNVQ** guidance variable.

The default value for the **QLSTCNVQ** guidance variable is 0, indicating that CPExpert should produce Rule DB2-421 when any conversation requests were queued by DDF waiting allocation. You can alter the analysis by specifying a different value. For example, if you wish to be notified when more than 50 conversation requests were queued by DDF waiting allocation, per DB2 statistics interval, specify:

```
%LET QLSTCNVQ = 50 ; * CONV REQUESTS QUEUED BY DDF, ALLOC;
```

Chapter 3.64: QLSTINDT - Threads indoubt with remote coordinator

CPExpert compares the QLSTINDT variable in DB2STATS (the number of threads indoubt with the remote coordinator) with the **QLSTINDT** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-403 when the number of threads indoubt with the remote coordinator exceeds the value specified by the **QLSTINDT** guidance variable.

The default value for the **QLSTINDT** guidance variable is 0, indicating that CPExpert should produce Rule DB2-403 when any threads indoubt with the remote coordinator. You can alter the analysis by specifying a different value. For example, if you wish to be notified when more than 50 threads became indoubt with the remote coordinator, per DB2 statistics interval, specify:

```
%LET QLSTINDT = 50 ; * THREADS INDOUBT WITH REMOTE COORD;
```

Chapter 3.65: QTXADEA - Number of deadlocks

CPExpert compares the QTXADEA variable in DB2STATS (the number deadlocks) with the **QTXADEA** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-311 when the number deadlocks exceeds the value specified by the **QTXADEA** guidance variable.

The default value for the **QTXADEA** guidance variable is 0, indicating that CPExpert should produce Rule DB2-311 when any deadlocks occurred. You can alter the analysis by specifying a different value. For example, if you wish to be notified when more than 10 deadlocks occurred, per DB2 statistics interval, specify:

```
%LET QTXADEA = 10 ; * NUMBER OF DEADLOCKS;
```

Chapter 3.66: QTXALES - Number of locks escalated to shared mode

CPExpert compares the QTXALES variable in DB2STATS (the number of locks escalated to shared mode) with the **QTXALES** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-311 when the number of locks escalated to shared mode exceeds the value specified by the **QTXALES** guidance variable.

The default value for the **QTXALES** guidance variable is 0, indicating that CPExpert should produce Rule DB2-311 when any locks were escalated to shared mode. You can alter the analysis by specifying a different value. For example, if you wish to be notified when more than 50 locks escalated to shared mode, per DB2 statistics interval, specify:

```
%LET QTXALES = 50 ; * NUMBER OF ESCALATIONS TO SHARED MODE ;
```

Chapter 3.67: QTXALEX - Number of locks escalated to exclusive

CPExpert compares the QTXALEX variable in DB2STATS (the number of locks escalated to exclusive mode) with the **QTXALEX** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-311 when the number of locks escalated to exclusive mode exceeds the value specified by the **QTXALEX** guidance variable.

The default value for the **QTXALEX** guidance variable is 0, indicating that CPExpert should produce Rule DB2-311 when any locks were escalated to exclusive mode. You can alter the analysis by specifying a different value. For example, if you wish to be notified when more than 50 locks escalated to exclusive mode, per DB2 statistics interval, specify:

```
%LET QTXALES = 10 ; * NUMBER OF ESCALATIONS TO EXCLUSIVE  MODE ;
```

Chapter 3.68: QTXATIM - Number of locks that timed-out

CPExpert compares the QTXATIM variable in DB2STATS (the number of times work was suspended for longer than the time-out value) with the **QTXATIM** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-310 when the number of times work timed-out exceeds the value specified by the **QTXATIM** guidance variable.

The default value for the **QTXATIM** guidance variable is 0, indicating that CPExpert should produce Rule DB2-310 when any time-outs occurred. You can alter the analysis by specifying a different value. For example, if you wish to be notified when more than 10 locks timed out, per DB2 statistics interval, specify:

```
%LET QTXATIM = 10 ; * NUMBER OF LOCKS THAT TIMED-OUT ;
```

Chapter 3.69: QWSBSBUF - Buffer overrun errors, SMF destination

CPExpert selects the SMF destination (QWSBNM = “SMF”) from DB2STATS. For this destination, CPExpert compares the QWSBSBUF variable in DB2STATS (the number of buffer overrun errors) with the **QWSBSBUF** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-110 when the number of buffer overrun errors exceeds the value specified by the **QWSBSBUF** guidance variable.

The default value for the **QWSBSBUF** guidance variable is 0, indicating that CPExpert should produce Rule DB2-110 when there were any buffer overrun errors for the SMF destination. You can alter the analysis by specifying a different value. For example, if you wish to be notified when more than 5 buffer overrun errors occurred for the SMF destination per DB2 statistics interval, specify:

```
%LET QWSBSBUF = 5 ; * NUMBER OF SMF BUFFER OVERRUN ERRORS;
```

Chapter 3.70: QXACALLAB - Number of stored procedure errors

CPExpert compares the QXACALLAB variable in DB2STATS (the number stored procedure errors) with the **QXACALLAB** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-131 when the number of stored procedure errors exceeds the value specified by the **QXACALLAB** guidance variable.

The default value for the **QXACALLAB** guidance variable is 0, indicating that CPExpert should produce Rule DB2-131 when there were any stored procedure errors. You can alter the analysis by specifying a different value. For example, if you wish to be notified when there were more than 10 stored procedure errors, per DB2 statistics interval, specify:

```
%LET QXACALLAB = 10 ; * NUMBER OF STORED PROCEDURE ERRORS;
```

©Copyright 1999, Computer Management Sciences, Inc.

Chapter 3.71: QXACALLRJ - Number of SQL calls rejected

CPExpert compares the QXACALLRJ variable in DB2STATS (the number SQL calls rejected) with the **QXACALLRJ** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-131 when the number of SQL calls rejected exceeds the value specified by the **QXACALLRJ** guidance variable.

The default value for the **QXACALLRJ** guidance variable is 0, indicating that CPExpert should produce Rule DB2-133 when there were any SQL calls rejected. You can alter the analysis by specifying a different value. For example, if you wish to be notified when more than 10 SQL calls were rejected, per DB2 statistics interval, specify:

```
%LET QXACALLRJ = 10 ; * NUMBER OF SQL CALLS REJECTED;
```

Chapter 3.72: QXACALLTO - Number of times SQL calls timed out

CPExpert compares the QXACALLTO variable in DB2STATS (the number of times SQL calls timed out) with the **QXACALLTO** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-132 when the number of times SQL calls timed out exceeds the value specified by the **QXACALLTO** guidance variable.

The default value for the **QXACALLTO** guidance variable is 0, indicating that CPExpert should produce Rule DB2-132 when any SQL calls timed out. You can alter the analysis by specifying a different value. For example, if you wish to be notified when more than 10 SQL calls timed out, per DB2 statistics interval, specify:

```
%LET QXACALLTO = 10 ; * NUMBER OF SQL CALLS THAT TIMED OUT;
```

Chapter 3.73: QXCOORNO - Parallel groups executed on a single DB2

CPExpert compares the QXCOORNO variable in DB2STATS (the number of times parallel groups executed on a single DB2) with the **QXCOORNO** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-643 when the number of times parallel groups executed on a single DB2 exceeds the value specified by the **QXCOORNO** guidance variable.

The default value for the **QXCOORNO** guidance variable is 0, indicating that CPExpert should produce Rule DB2-643 when any parallel groups executed on a single DB2. You can alter the analysis by specifying a different value. For example, if you wish to be

notified when more than 50 parallel groups executed on a single DB2, per DB2 statistics interval, specify:

```
%LET QXCOORNO = 50 ; * PARALLEL GROUPS EXECUTED ON SINGLE DB2;
```

Chapter 3.74: QXDEGBUF- Parallel groups fell back, buffers

CPExpert compares the QXDEGBUF variable in DB2STATS (the number of parallel groups that fell back to sequential mode because of buffer shortage) with the **QXDEGBUF** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-522 when the number of times parallel groups fell back to sequential exceeds the value specified by the **QXDEGBUF** guidance variable.

The default value for the **QXDEGBUF** guidance variable is 0, indicating that CPExpert should produce Rule DB2-252 when any parallel groups fell back to sequential mode. You can alter the analysis by specifying a different value. For example, if you wish to be notified when more than 10 parallel groups fell back to sequential mode because of buffer shortage, per DB2 statistics interval, specify:

```
%LET QXDEGBUF = 10 ; * PARALLEL GROUPS FELL BACK, BUFFERS;
```

Chapter 3.75: QXDEGENC - Parallel groups executed in sequential

CPExpert compares the QXDEGENC variable in DB2STATS (the number of parallel groups that executed in sequential mode because MVS enclave services were not available) with the **QXDEGENC** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-663 when the number of times parallel groups executed in sequential mode because MVS enclave services were not available exceeds the value specified by the **QXDEGENC** guidance variable.

The default value for the **QXDEGENC** guidance variable is 0, indicating that CPExpert should produce Rule DB2-663 when any parallel groups executed in sequential mode because MVS enclave services were not available. You normally should not alter this value, since the finding indicates that you should BIND the plan or package. However, yoiu may have a unique situation that we do not forsee. You could either turn off Rule DB2-633 (using the “turn off rule” techinique) or alter the guidance. For example, if you wish to be notified when more than 10 parallel groups executed in sequential mode because MVS enclave services were not available per DB2 statistics interval, specify:

```
%LET QXDEGENC = 10 ; * PARALLEL GROUPS EXECUTED IN SEQUENTIAL;
```

Chapter 3.76: QXISORR - Parallel groups executed on single DB2

CPExpert compares the QXISORR variable in DB2STATS (the number of parallel groups that executed on a single DB2 because repeatable read was specified) with the **QXISORR** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-642 when the number of times parallel groups executed on a single DB2 because repeatable read was specified exceeds the value specified by the **QXISORR** guidance variable.

The default value for the **QXISORR** guidance variable is 0, indicating that CPExpert should produce Rule DB2-642 when any parallel groups executed on a single DB2 because repeatable read was specified. You can alter the analysis by specifying a different value. For example, if you wish to be notified when more than 50 parallel groups executed on a single DB2 because repeatable read was specified, per DB2 statistics interval, specify:

```
%LET QXISORR = 50 ; * PARALLEL EXECUTED ON SINGLE DB2, REPEAT READ;
```

Chapter 3.77: QXREDGRP - Times parallel groups reduced, buffers

CPExpert compares the QXREDGRP variable in DB2STATS (the number of parallel groups were reduced because of buffer shortage) with the **QXREDGRP** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-642 when the number of parallel groups that were reduced because of buffer shortage exceeds the value specified by the **QXREDGRP** guidance variable.

The default value for the **QXREDGRP** guidance variable is 0, indicating that CPExpert should produce Rule DB2-642 when any parallel groups were reduced because of buffer shortage. You can alter the analysis by specifying a different value. For example, if you wish to be notified when more than 10 parallel groups were reduced because of buffer shortage, per DB2 statistics interval, specify:

```
%LET QXREDGRP = 10 ; * PARALLEL GROUPS REDUCED, BUFFERS;
```

Chapter 3.78: QXSTDEXP - Prepared statement exceeded MAXKEEPD

CPExpert compares the QXSTDEXP variable in DB2STATS (the number of times DB2 discarded a prepared statement from the prepared statement cache because the number of prepared statements exceeded the MAXKEEPD value) with the **QXSTDEXP** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-642 when QXSTDEXP value in DB2STATS exceeds the value specified by the **QXSTDEXP** guidance variable.

The default value for the **QXSTDEXP** guidance variable is 0, indicating that CPExpert should produce Rule DB2-642 whenever DB2 discarded a prepared statement from the prepared statement cache because the number of prepared statements exceeded the MAXKEEPD value. You can alter the analysis by specifying a different value. For example, if you wish to be notified when more than 50 prepared statements were rejected because the MAXKEEPD value was exceeded, per DB2 statistics interval, specify:

```
%LET QXSTDEXP = 50 ; * PREPARED STATEMENTS EXCEEDED MAXKEEPD;
```

Chapter 3.79: QZZCSKIP - Parallelism coordinator bypassed, buffers

CPExpert compares the QZZCSKIP variable in DB2STATS (the number of times the parallism coordinator bypassed a DB2 due to lack of buffers) with the **QZZCSKIP** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-641 when the number of times the parallism coordinator bypassed a DB2 due to lack of buffers exceeds the value specified by the **QZZCSKIP** guidance variable.

The default value for the **QZZCSKIP** guidance variable is 0, indicating that CPExpert should produce Rule DB2-641 whenever the parallism coordinator bypassed a DB2 due to lack of buffers. You can alter the analysis by specifying a different value. For example, if you wish to be notified when the parallism coordinator bypassed a DB2 due to lack of buffers more than 10 times, per DB2 statistics interval, specify:

```
%LET QZZCSKIP = 10 ; * BYPASSED DB2, NOT ENOUGH BUFFERS;
```

Chapter 3.80: SYNCSRV variable and LOCKSRV

Signaling requests to a coupling facility can occur only if a subchannel to the coupling facility is available. If no subchannel is available, the cross-system extended services

(XES) will either enter a CPU "spin loop" waiting for a subchannel to become available or queue the request until a subchannel is available.

For non-lock structures, CPExpert compares the synchronous service time (R744SSTM) against the **SYNCSRV** variable in USOURCE(DB2GUIDE). For lock structures, CPExpert compares the synchronous service time (R744SSTM) against the **LOCKSRV** variable in USOURCE(db2GUIDE). Requests to lock structures are very short, and the service time should be significantly less than the requests to non-lock structures.

CPExpert produces Rule DB2-660 when the synchronous service time is greater than the SYNCSRV guidance variable for non-lock structures and produces Rule DB2-660 when the synchronous service time is greater than the LOCKSRV guidance variable for lock structures.

The default value for the SYNCSRV variable is 350, indicating that CPExpert should produce Rule DB2-660 when synchronous service time is more than 350 microseconds for non-lock structures. The default value for the LOCKSRV variable is 250, indicating that CPExpert should produce Rule DB2-660 when synchronous service time is more than 250 microseconds for lock structures.

You can alter this analysis using the SYNCSRV and LOCKSRV guidance variables. For example, if you wish to be notified only when the synchronous service time is greater than 500 microseconds for non-lock structures or greater than 100 for lock structures, specify:

```
%LET SYNCSRV = 500 ; * ACCEPTABLE SERVICE TIME, SYNCH REQUESTS;
%LET LOCKSRV = 500 ; * ACCEPTABLE SERVICE TIME, LOCK REQUESTS;
```

IBM provides the following example service times based on measurements of coupling facility lock requests for various combinations of sender CPCs and CFs. These measurements were reported in *S/390 MVS Parallel Sysplex Configuration, Volume 2: Cookbook*, document Number SG24-2076-00.

Central Processor	**Coupling facility**	**Service Time, Lock Requests** (microseconds)
9672R1 based	9674C01	250
9672R2/R3 based	9674C02/3	180
9672G3	9674C04	140

9672G4	9674C04	100
9672G4	9674C05	70
9021 711 based	9674C01	160
9021 711 based	9674C02/3	130
9021 711 based	9674C04	100
9021 711 based	9021 711 based	80

Chapter 3.81: SYNCCHG variable

Signaling requests to a coupling facility can occur only if a subchannel to the coupling facility is available. If no subchannel is available, the cross-system extended services (XES) will either enter a CPU "spin loop" waiting for a subchannel to become available or queue the request until a subchannel is available.

For synchronous requests, XES will either (1) satisfy the request if a subchannel is available, (2) enter CPU "spin-looping" until a subchannel is available and the request is satisfied, or (3) convert the synchronous request to an asynchronous request if the type of request permits the conversion.

CPExpert computes the percent of synchronous requests changed to asynchronous requests by the following algorithm:

$$\textit{Percent synchronous changed} = \frac{\textit{Synchronous requests changed}}{\textit{Synchronous requests}} \quad \textit{or} \quad \frac{\textit{R742SSTA}}{\textit{R742SSRC}}$$

CPExpert produces Rule DB2-665 when this percent is greater than the SYNCCHG guidance variable.

The default value for the SYNCCHG guidance variable is 10, indicating that CPExpert should produce Rule DB2-665 when more than 10% of the synchronous requests are changed to asynchronous requests.

You can alter this analysis using the SYNCCHG guidance variable. For example, if you wish to be notified only when more than 15% of the synchronous requests were changed to asynchronous requests, specify:

```
%LET SYNCCHG = 15 ; * ACCEPTABLE PERCENT CHANGED REQUESTS;
```

Chapter 4: Specifying guidance for individual DB2 buffer pools

This chapter applies only if you wish to provide guidance to individual DB2 buffer pools.

Most users of the DB2 Component use CPExpert to analyze DB2 subsystems containing multiple buffer pools. One set of analysis guidance variables would not normally apply to all buffer pools being analyzed (for example, analysis of buffer pools devoted to non-sort pages would be quite different from analysis of buffer pools devoted to sort pages). Consequently, CPExpert provides the capability to override the basic guidance on a buffer pool-by-buffer pool basis. This chapter describes how to implement this override guidance.

Unique guidance for particular DB2 buffer pools can be accomplished three steps;

Step 1: Specify **%LET BPGUIDE = YES;** in USOURCE(DB2GUIDE). This specification alerts CPExpert that specific guidance will be provided for individual buffer pools. See Chapter 4.1 below.

Step 2: Specify **%LET BPn = GUIDE;** in USOURCE(DB2GUIDE). This specification alerts CPExpert that specific guidance will be provided for buffer pool “n”. See Chapter 4.2 below.

Step 3: Place the guidance for buffer pool “n” into USOURCE as member **BPn** where “n” is the buffer pool number to which the guidance applies. See Chapter 4.3 below.

Chapter 4.1: Specify BPGUIDE guidance variable

The **BPGUIDE** guidance variable specifies whether override guidance exists for one or more buffer pools (not that this variable does **not** relate to group buffer pools).

CPExpert normally uses the guidance in USOURCE(DB2GUIDE) to control its analysis of DB2 buffer pools. You can override this process by specifying **%LET BPGUIDE = Y;** in USOURCE(DB2GUIDE). When you make this specification, CPExpert generates code to execute %INCLUDE SAS statements to include additional guidance for particular DB2 buffer pools, as controlled by the **BPn = GUIDE** guidance variable(s).

The purpose of having a “general” controlling parameter is to provide a “switch” that allows you to normally use override guidance, but you can easily turn off the overrides and analyze all buffer pools using the basic guidance.

©Copyright 1999, Computer Management Sciences, Inc.

Chapter 4.2: Identify DB2 buffer pools having unique guidance

Each DB2 buffer pool for which you wish to provide specific guidance must be identified to CPExpert. This identification is done by specifying **%LET BPn = GUIDE;** in USOURCE(DB2GUIDE), where "n" is the buffer pool number. For example, if you wish to specify unique guidance for Buffer Pool 7, you would specify **%LET BP7=GUIDE;** in USOURCE(DB2GUIDE). This specification tells CPExpert that unique guidance for Buffer Pool 7 is contained in USOURCE(BP7).

If you wished to specify guidance for Buffer Pool 32K, you would specify %LET BP32K=GUIDE; and if you wished to specify guidance for Buffer Pool 32K1, you would specify %LET BP32K1=GUIDE;.

Chapter 4.3: Place unique guidance in USOURCE(buffer pool) member

Create a unique analysis guidance member in the USOURCE partitioned data set. For example, if you wish to specify unique guidance for Buffer Pool 7, you would create a USOURCE(BP7) member.

You can specify any unique guidance for the particular buffer pool by entering the appropriate SAS %LET statements into the USOURCE(buffer pool number) member. For example, if you wish to specify unique guidance for Bufer Pool 7, you would specify appropriate SAS %LET statements in USOURCE(BP7).

Important: If you should simply copy the USOURCE(DB2GUIDE) member to USOURCE(buffer pool number) member, you must delete from the new module all information unrelated to the unique guidance. **Do not retain data selection and reporting variables**! Unpredictable analysis could result if you retain the data selection and reporting variables.

Chapter 4.4: Restrictions

The following restrictions apply to specifying unique guidance for DB2 buffer pools:

- The override guidance specific to a particular DB2 buffer pool applies only to that buffer pool. CPExpert reverts to the original guidance contained in USOURCE(DB2GUIDE) for subsequent DB2 buffer pools (unless those DB2 buffer pools have unique guidance).

- You will get a SAS error if you specify %LET BPn=GUIDE; and do not have the corresponding member as USOURCE(BPn).

- You can "comment out" any %LET BPn=GUIDE; statement simply by replacing the "%" with "*" (that is, replace the percent with an asterisk).

©Copyright 1999, Computer Management Sciences, Inc. Revised: April, 2003

Chapter 5: Specifying guidance for individual DB2 group buffer pools

This chapter applies only if you wish to provide unique guidance to individual DB2 group buffer pools on a data sharing environment.

Most users of the DB2 Component use CPExpert to analyze DB2 subsystems containing multiple group buffer pools. One set of analysis guidance variables would not normally apply to all group buffer pools being analyzed. Consequently, CPExpert provides the capability to override the basic guidance on a group buffer pool-by-pool basis. This chapter describes how to implement this override guidance.

Unique guidance for particular DB2 buffer pools can be accomplished three steps;

Step 1: Specify **%LET GBPGUIDE = YES;** in USOURCE(DB2GUIDE). This specification alerts CPExpert that specific guidance will be provided for individual group buffer pools. See Chapter 5.1 below.

Step 2: Specify **%LET GBPn = GUIDE;** in USOURCE(DB2GUIDE). This specification alerts CPExpert that specific guidance will be provided for group buffer pool “n”. See Chapter 5.2 below.

Step 3: Place the guidance for group buffer pool “n” into USOURCE as member **GBPn** where “n” is the group buffer pool number to which the guidance applies. See Chapter 5.3 below.

Chapter 5.1: Specify GBPGUIDE guidance variable

The **GBPGUIDE** guidance variable specifies whether override guidance exists for one or more group buffer pools (not that this variable does **not** relate to virtual buffer pools).

CPExpert normally uses the guidance in USOURCE(DB2GUIDE) to control its analysis of DB2 group buffer pools. You can override this process by specifying **%LET GBPGUIDE = Y;** in USOURCE(DB2GUIDE). When you make this specification, CPExpert generates code to execute %INCLUDE SAS statements to include additional guidance for particular DB2 group buffer pools, as controlled by the **GBPn = GUIDE** guidance variable(s).

The purpose of having a “general” controlling parameter is to provide a “switch” that allows you to normally use override guidance, but you can easily turn off the overrides and analyze all group buffer pools using the basic guidance.

Chapter 5.2: Identify DB2 group buffer pools having unique guidance

Each DB2 group buffer pool for which you wish to provide specific guidance must be identified to CPExpert. This identification is done by specifying **%LET GBPn = GUIDE;** in USOURCE(DB2GUIDE), where "n" is the group buffer pool number. For example, if you wish to specify unique guidance for Group Buffer Pool 3, you would specify **%LET GBP3=GUIDE;** in USOURCE(DB2GUIDE). This specification tells CPExpert that unique guidance for Group Buffer Pool 7 is contained in USOURCE(GBP3).

If you wished to specify guidance for Group Buffer Pool 32K, you would specify %LET GBP32K=GUIDE; and if you wished to specify guidance for Group Buffer Pool 32K1, you would specify %LET GBP32K1=GUIDE;.

Chapter 5.3: Place unique guidance in USOURCE(buffer pool) member

Create a unique analysis guidance member in the USOURCE partitioned data set. For example, if you wish to specify unique guidance for Group Buffer Pool 3, you would create a USOURCE(GBP3) member.

You can specify any unique guidance for the particular buffer pool by entering the appropriate SAS %LET statements into the USOURCE(buffer pool number) member. For example, if you wish to specify unique guidance for Group Buffer Pool 3, you would specify appropriate SAS %LET statements in USOURCE(GBP3).

Important: If you should simply copy the USOURCE(DB2GUIDE) member to USOURCE(buffer pool number) member, you must delete from the new module all information unrelated to the unique guidance. **Do not retain data selection and reporting variables**! Unpredictable analysis could result if you retain the data selection and reporting variables.

Chapter 5.4: Restrictions

The following restrictions apply to specifying unique guidance for DB2 buffer pools:

- The override guidance specific to a particular DB2 group buffer pool applies only to that buffer pool. CPExpert reverts to the original guidance contained in USOURCE(DB2GUIDE) for subsequent DB2 group buffer pools (unless those DB2 buffer pools have unique guidance).

- You will get a SAS error if you specify %LET GBPn=GUIDE; and do not have the corresponding member as USOURCE(GBPn).

©Copyright 1999, Computer Management Sciences, Inc.

- You can “comment out” any %LET BPn=GUIDE; statement simply by replacing the “%” with “*” (that is, replace the percent with an asterisk).

©Copyright 1999, Computer Management Sciences, Inc.

Chapter 6: Specifying guidance for individual DB2 subsystems

This chapter applies only if you wish to provide unique guidance for analysis of performance constraints experienced by **individua**l DB2 subsystems.

Some users of the DB2 Component wish to analyze DB2 performance constraints in environments with multiple DB2 subsystems. For these users, one set of analysis guidance variables might not apply to all DB2 subsystems being analyzed. These users could iteratively process their performance data base, specifying different parameters to the DB2 Component. This approach is time consuming and wastes system resources since the performance data base must be processed multiple times. Consequently, CPExpert provides the capability to override the basic guidance on a DB2 subsystem-by-subsystem basis. This chapter describes how to implement this override guidance.

Unique guidance for particular DB2 subsystems can be accomplished in five steps;

Step 1: Specify **%LET DB2GUIDE = YES;** in USOURCE(DB2GUIDE). This specification alerts CPExpert that specific guidance will be provided for individual DB2 subsystems. See Chapter 6.1 below.

Step 2: Specify **%LET *db2subsystemname* = GUIDE;** in USOURCE(DB2GUIDE). This specification alerts CPExpert that specific guidance will be provided for DB2 subsystem *db2subsystemname*. See Chapter 6.2 below for additional information.

WARNING: If you have used the special characters #, **$**, or @ in the name of your DB2 subsystem, please contact Computer Management Sciences. MVS allows these characters to be used in DB2 subsystem names. However, SAS will not allow these characters to be used in data set names or in variable names. If you have used these special characters, a different approach must be used to specify guidance for individual DB2 subsystems.

Step 3: Create a partitioned data set using standard MVS data set creation techniques, for each DB2 subsystem for which you wish to specify unique guidance. This partitioned data set can be a single track, and its name can be anything you chose. However, you might find it easier for reference purposes to name the partitioned data set ***prefix.CPEXPERT.USOURCE.db2subsystemname***. For example, if you wanted CPExpert to apply your individual guidance to DB2 subsystem **DB2B**, you would find it convenient to name the partitioned data set ***prefix.CPEXPERT.USOURCE.DB2B***. See Chapter 6.3 below for additional information.

Step 4: Place the guidance for the individual DB2 subsystem into a DB2GUIDE member placed in the partitioned data set describing the DB2 subsystem for which the guidance applies. In the above example, you would place the guidance for DB2B

into **prefix.CPEXPERT.USOURCE.DB2B(DB2GUIDE)**. See Chapter 6.4 below for additional information.

Step 5: Create a Job Control Language (JCL) DD statement with the name of **db2subsystemname** and have this DD statement describe the partitioned data set containing individual guidance for the DB2 subsystem. See Chapter 6.5 below for additional information.

Chapter 6.1: Specify DB2GUIDE guidance variable

The **DB2GUIDE** guidance variable specifies whether override guidance exists for one or more DB2 subsystems.

CPExpert normally uses the guidance in USOURCE(DB2GUIDE) to control its analysis of DB2 subsystems. You can override this process by specifying **%LET DB2GUIDE = Y;** in USOURCE(DB2GUIDE). When you make this specification, CPExpert generates code to execute %INCLUDE SAS statements to include additional guidance for the specific DB2 subsystem(s), as controlled by the **db2subsystemname = GUIDE** guidance variable(s).

The purpose of having a “general” controlling parameter is to provide a “switch” that allows you to normally use override guidance, but you can easily turn off the overrides and analyze all DB2 subsystems using the basic guidance.

Chapter 6.2: Identify DB2 subsystems having unique guidance

Each DB2 subsystem for which you wish to provide specific guidance must be identified to CPExpert. This identification is done by specifying **%LET db2subsystemname = GUIDE;** in USOURCE(DB2GUIDE), where “db2subsystemname” is the name of the DB2 subsystem for which you wish to specify individual guidance. For example, if you wish to specify unique guidance for DB2B, you would specify **%LET DB2B=GUIDE;** in USOURCE(DB2GUIDE). This specification tells CPExpert that unique guidance for the DB2B subsystem is contained in the partitioned data set described by the Job Control Language DD statement named DB2B.

Please refer to Step 2 on the previous page for a WARNING if you have used the special characters **#**, **$**, or @ in the name of your DB2 subsystem.

Chapter 6.3: Create partitioned data set(s)

Create a partitioned data set using standard pds creation techniques, for each DB2 subsystem for which you wish to specify unique guidance variables. This partitioned data set can be a single track, and its name can be anything you chose. However, you might

find it easier for control purposes to name the partitioned data set ***prefix.CPEXPERT.USOURCE.db2subsystemname***. For example, if you wanted CPExpert to apply your individual guidance to DB2 subsystem **DB2B**, you would find it convenient to name the partitioned data set ***prefix.CPEXPERT.USOURCE.DB2B***. If you also wished to specify unique guidance for **DB2F**, for example, you would also create a partitioned data set named ***prefix.CPEXPERT.USOURCE.DB2F.***

Chapter 6.4: Place unique guidance in partitioned data set(s)

Insert a member named DB2GUIDE into the partitioned data set(s) that you have defined. ***db2subsystemname*(DB2GUIDE)** would contain the guidance that would be unique to the DB2 subsystem *db2subsystemname*. For example, if you wanted CPExpert to apply your individual guidance to DB2 subsystem **DB2B**, you would place the unique guidance for DB2B into **prefix.CPEXPERT.USOURCE.DB2B(DB2GUIDE)**.

CPExpert initially %INCLUDES guidance from prefix.CPEXPERT.USOURCE(DB2GUIDE). After selecting and shaping DB2 interval statistics and before entering the evaluation phase of each subsystem, CPExpert will read *db2subsystemname*(DB2GUIDE) for each DB2 subsystem to be analyzed with unique guidance. The guidance variables will be controlling while CPExpert is analyzing data for each DB2 subsystem with unique guidance.

After analyzing data for a particular DB2 subsystem, CPExpert will again %INCLUDE guidance from USOURCE(DB2GUIDE). Consequently, you can (1) place the original DB2GUIDE member into the *db2subsystemname*(DB2GUIDE) member and modify guidance variables as you wish, or (2) you can place **only** those SAS %LET statements containing guidance **unique** to a particular subsystem into the *db2subsystemname*(DB2GUIDE) member.

You can have any number of unique guidance modules applying to specific DB2 subsystems. For each DB2 subsystem, you can alter **any** of the analysis guidance variables described in Chapter 3 (Analysis Guidance Variables) of this section, including buffer pool override guidance or group buffer pool override guidance.

For example, you might wish to specify unique guidance for Buffer Pool 3 and have that guidance apply only to DB2B. In this example, you would follow the procedures outlined in Chapter 4 (Specifying guidance for individual DB2 buffer pools) of this Section, except the procedures would apply to the guidance variables contained in **prefix.CPEXPERT.USOURCE.DB2B(DB2GUIDE)**. You would specify the appropriate SAS %LET statements(**%LET BPGUIDE=YES;** and **%LET BP3 = GUIDE;**) in **prefix.CPEXPERT.USOURCE.DB2B(DB2GUIDE)** to tell CPExpert that unique guidance was specified for buffer pools used by DB2B and that DB2B's Buffer Pool 3 had unique guidance. You would place the specific guidance that was to be used when analyzing DB2B's Buffer Pool 3 into **prefix.CPEXPERT.USOURCE.DB2B(BP3)**.

©Copyright 1999, Computer Management Sciences, Inc.

You may **not** specify unique data selection and reporting variables as described in Chapter 1 (Data Selection Variables) of this section, and you may **not** specify unique analysis control variables as described in Chapter 2 (Analysis Control Variables) except for the Test versus Production variable. The reason for these restrictions is that the analysis guidance variables are %INCLUDED after all data shaping has been performed, and CPExpert is preparing to analyze the resulting data.

Chapter 6.5: Create Job Control Language DD statement(s)

Create a JCL DD statement named **db2subsystemname** and have this DD statement describe the partitioned data set containing individual guidance for the DB2 subsystem. If you wanted CPExpert to apply your individual guidance to DB2 subsystem **DB2B**, using the above example, you would create a DD statement as follows:

```
//DB2B     DD   DSN=prefix.CPEXPERT.USOURCE.DB2B,DISP=SHR
```

You must create a JCL DD statement for each DB2 subsystem for which you wish to specify individual guidance, and this DD statement must describe the partitioned data set that contains the guidance.

Chapter 6.6: Restrictions

The following restrictions apply to specifying unique guidance for DB2 buffer pools:

- The override guidance specific to a particular DB2 subsystem applies only to that specific DB2 subsystem. CPExpert reverts to the original guidance contained in USOURCE(DB2GUIDE) for subsequent DB2 subsystems (unless those DB2 subsystems also have unique guidance).

- You will get a SAS error if you specify %LET *db2subsystemname* =GUIDE; and do not have the corresponding DB2GUIDE member contained in a partitioned data set identified by a DD statement. CPExpert uses *db2subsystemname* as the name of the DD statement to reads the DB2GUIDE member.

- You will get a JES error if you have the appropriate DD statement but do not have the partitioned data set defined.

- You can “comment out” any %LET *db2subsystemname*=GUIDE; statement simply by replacing the “%” with “*” (that is, replace the percent with an asterisk). In this case, the

DB2 subsystem would be analyzed using the guidance contained in USOURCE(DB2GUIDE).

- As mentioned above (and repeated here for emphasis), you may not specify unique data selection or reporting variables as described in Chapter 1 (Data Selection Variables) of this section, nor may you specify unique analysis control variables as described in Chapter 2 (Analysis Control Variables) except for the Test versus Production variable.

  The reason for these restrictions is that the analysis guidance variables are %INCLUDED after all data shaping has been performed, and CPExpert is preparing to analyze the resulting data.

Chapter 1: Data Selection Variables 3-2
Chapter 1.1: DB2DATES and DB2TIMES variables 3-3
Chapter 1.2: DB2DATEE and DB2TIMEE variables 3-3
Chapter 1.3: SHIFT variable 3-5
Chapter 1.4: SYSTEM variable 3-5
Chapter 1.5: SYSTEMn variable(s) 3-6
Chapter 1.6: DB2SID variable 3-6
Chapter 1.7: ALL_DB2 variable 3-7

Chapter 1.8: SAS Output Delivery System 3-8
Chapter 2: Analysis Control Variables 3-9
Chapter 2.1: Specifying Production or Test Subsystem 3-10
Chapter 2.3: Specifying whether DB2 accounting data are available 3-10
Chapter 2.4: Specifying availability of Trace IFCID 106 records 3-10
Chapter 2.6: Specifying availability of Trace IFCID 196 records 3-11
Chapter 2.7: Specifying availability of SMF Type 74 (structure) records 3-11
Chapter 2.8: Specifying individual guidance for buffer pools 3-12
Chapter 2.9: Specifying individual guidance for group buffer pools 3-12
Chapter 2.10: Specifying SAS library containing DB2 data 3-12
Chapter 2.11: Specifying data set name for DB2 datasets 3-12
Chapter 3: Analysis Guidance Variables 3-13
Chapter 3.1: Turning OFF DB2 Component Rules 3-17
Chapter 3.2: ASYNCSRV variable 3-17
Chapter 3.3: BUFFPGRT - Acceptable page-in rate, all buffer pools 3-18
Chapter 3.4. LOCKCONT variable 3-18
Chapter 3.5: PAGERATE - Acceptable page-in rate, individual pool 3-20
Chapter 3.6: PCTCBA - Percent active buffers 3-20
Chapter 3.7: PCTCTL - Percent CT sections not found 3-21
Chapter 3.8: PCTDBDL - Percent DBD sections not found 3-22
Chapter 3.9: PCTDPF - Percent dynamic sequential prefetch 3-23
Chapter 3.10: PCTEDM- Percent free pages in EDM pool 3-23
Chapter 3.11: PCTESC - Percent lock escalation not effective 3-24
Chapter 3.12: PCTFLAM - Percent failed look-ahead tape mounts 3-24
Chapter 3.13: PCTESC - Percent lock escalation not effective 3-25
Chapter 3.14: PCTHITGB - Percent read hit cross-invalidated pages 3-26
Chapter 3.15: PCTRACT - Percent reads from active log 3-27
Chapter 3.16: PCTRAND - Percent GETPAGE from random requestors . . 3-27

Chapter 3.17: PCTRSAT - Percent reconnection attempts succeeded . . . 3-28
Chapter 3.18: PCTSEQ - Percent GETPAGE, sequential requestors 3-29
Chapter 3.19: PCTSLOCK - Percent suspends for lock conflict 3-29
Chapter 3.20: PCTXIGBP - Acceptable percent reclaims, XI pages 3-30
Chapter 3.21: Q3STCTHW - Times threads were queued at create 3-31
Chapter 3.22: Q3STMEOM - End-of-memory (non-DB2 allied agent) 3-31
Chapter 3.23: Q3STMEOT - End-of-task (non-DB2 allied agent) 3-31
Chapter 3.24: QBGLAN - CF prefetch reads, no data, no directory 3-32
Chapter 3.25: QBGLAR - CF prefetch, no data, directory entry 3-32
Chapter 3.26: QBGLCN - Times CF castout engine not available 3-33
Chapter 3.27: QBGLRF - Times CF reads failed, lack of storage 3-33
Chapter 3.28: QBGLWF - Times CF writes failed, lack of storage 3-34
Chapter 3.29: QBGLSU - Times CF write engine was not available 3-34
Chapter 3.30: QBSTARF - Unsuccessful HP to VP read, async ADMF . . . 3-34
Chapter 3.31: QBSTAWF - Unsuccessful HP to VP write, async ADMF . . 3-35
Chapter 3.32: QBSTDMC - Times Data Manager Threshold reached 3-35
Chapter 3.33: QBSTHRF - Unsuccessful HP to VP reads (MVPG) 3-36
Chapter 3.34: QBSTHWF - Unsuccessful HP to VP writes (MVPG) 3-36
Chapter 3.35: QBSTIMW - Number of Immediate Writes 3-37
Chapter 3.36: QBSTJIS - Times prefetch I/O streams denied 3-37
Chapter 3.37: QBSTNGT - Number of unsuccessful GETPAGE 3-38
Chapter 3.38: QBSTPL1 - Times prefetch reduced to ½ of normal 3-38
Chapter 3.39: QBSTPL2 - Times prefetch reduced to 1/4 of normal 3-39
Chapter 3.40: QBSTPQF - Times I/O parallelism was downgraded 3-39
Chapter 3.41: QBSTREE - Times prefetch disabled, no read engine 3-40
Chapter 3.42: QBSTSPD - Times prefetch disabled, no buffers 3-40
Chapter 3.43: QBSTWEE - Times write engine was not available 3-40
Chapter 3.44: QBSTWFD - Times work files denied during sort 3-41
Chapter 3.45: QBSTWFF - Times sort was inefficient, buffer shortage . . . 3-41
Chapter 3.46: QBSTWFR - Number of sort merge passes 3-42
Chapter 3.47: QBSTWIO - Pages written to DASD per write I/O 3-42
Chapter 3.48: QBSTWKPD - Prefetch canceled, prefetch quantity = 0 . . . 3-43
Chapter 3.49: QDSTQCRT - Conversations deallocated, ZPARMS limit . 3-43
Chapter 3.50: QDSTQDBT - DB2 access queued, max remote threads . . . 3-44
Chapter 3.51: QISEFAIL - Times failures caused by full EDM pool 3-44
Chapter 3.52: QISTCOLS - Times invalid SELECT procedure 3-45
Chapter 3.53: QISTRLLM - Times RID terminated, over calculated limit . . 3-45
Chapter 3.54: QISTRMAX - Times RID terminated, maximum storage . . . 3-46
Chapter 3.55: QISTRPLM - Times RID terminated, over physical limit 3-46
Chapter 3.56: QISTRSTG - Times RID terminated, insufficient storage . . . 3-47
Chapter 3.57: QJSTALR - Number of archive log read allocations 3-47
Chapter 3.58: QJSTALW - Number of archive log write allocations 3-47
Chapter 3.59: QJSTRARH - Number of reads from archive log 3-48
Chapter 3.60: QJSTTVC - Number of reads delayed, tape volume 3-48

©Copyright 1999, Computer Management Sciences, Inc. Revised: April, 2003

Chapter 3.61: QJSTWTB - Number of waits for output log buffer 3-49
Chapter 3.62: QJSTWUR - Number of read delays, resources 3-49
Chapter 3.63: QLSTCNVQ - Conversation requests queued by DDF 3-49
Chapter 3.64: QLSTINDT - Threads indoubt with remote coordinator 3-50
Chapter 3.65: QTXADEA - Number of deadlocks 3-50
Chapter 3.66: QTXALES - Number of locks escalated to shared mode . . . 3-51
Chapter 3.67: QTXALEX - Number of locks escalated to exclusive 3-51
Chapter 3.68: QTXATIM - Number of locks that timed-out 3-51
Chapter 3.69: QWSBSBUF - Buffer overrun errors, SMF destination 3-52
Chapter 3.70: QXACALLAB - Number of stored procedure errors 3-52
Chapter 3.71: QXACALLRJ - Number of SQL calls rejected 3-53
Chapter 3.72: QXACALLTO - Number of times SQL calls timed out 3-53
Chapter 3.73: QXCOORNO - Parallel groups executed on a single DB2 . . 3-53
Chapter 3.74: QXDEGBUF- Parallel groups fell back, buffers 3-54
Chapter 3.75: QXDEGENC - Parallel groups executed in sequential 3-54
Chapter 3.76: QXISORR - Parallel groups executed on single DB2 3-55
Chapter 3.77: QXREDGRP - Times parallel groups reduced, buffers 3-55
Chapter 3.78: QXSTDEXP - Prepared statement exceeded MAXKEEPD . 3-56
Chapter 3.79: QZZCSKIP - Parallelism coordinator bypassed, buffers . . . 3-56
Chapter 3.80: SYNCSRV variable and LOCKSRV 3-56
Chapter 3.81: SYNCCHG variable 3-58
Chapter 4: Specifying guidance for individual DB2 buffer pools 3-59
Chapter 4.1: Specify BPGUIDE guidance variable 3-59
Chapter 4.2: Identify DB2 buffer pools having unique guidance 3-60
Chapter 4.3: Place unique guidance in USOURCE(buffer pool) member . . . 3-60
Chapter 4.4: Restrictions 3-60
Chapter 5: Specifying guidance for individual DB2 group buffer pools 3-61
Chapter 5.1: Specify GBPGUIDE guidance variable 3-61
Chapter 5.2: Identify DB2 group buffer pools having unique guidance 3-62
Chapter 5.3: Place unique guidance in USOURCE(buffer pool) member . . . 3-62
Chapter 5.4: Restrictions 3-62
Chapter 6: Specifying guidance for individual DB2 subsystems 3-64
Chapter 6.1: Specify DB2GUIDE guidance variable 3-65
Chapter 6.2: Identify DB2 subsystems having unique guidance 3-65
Chapter 6.3: Create partitioned data set(s) 3-65
Chapter 6.4: Place unique guidance in partitioned data set(s) 3-66
Chapter 6.5: Create Job Control Language DD statement(s) 3-67
Chapter 6.6: Restrictions 3-67

©Copyright 1999, Computer Management Sciences, Inc. Revised: April, 2003

Section 4: Using the DB2 Component

This section describes how to execute the DB2 Component of CPExpert.

The instructions in this section assume that you have installed the CPExpert software. The DB2 Component is installed as normal part of installing CPExpert, and the installation steps unique to the DB2 Component are described in Section 2 of this User Manual. If you have not installed CPExpert, please install the software and implement Section 2 of this manual before continuing.

Before executing the DB2 Component, you should review the guidance contained in the prefix.CPEXPERT.USOURCE(DB2GUIDE) PDS member, to make sure that the guidance variables are appropriate for your environment. Exhibit 3-1 illustrates the guidance variables in the DB2GUIDE PDS member.

Chapter 1: Executing the DB2CPE Module

This chapter describes how to execute the DB2CPE Module of the DB2 Component. We suggest that you execute the DB2CPE Module on a daily basis, after the normal update of your performance data base.

As stated in the Introduction to this document, the DB2 Component consists of numerous modules, working together to (1) shape DB2 statistics data contained in a MXG or SAS/ITSV performance data base, (2) evaluate the data to assess problems or potential problems with DB2 performance, and (3) report the results from the evaluation. These modules are loaded and controlled by the central DB2 Component of CPExpert (titled DB2CPE).

Executing the DB2 Component consists of three steps, described in the below text.

Step 1: Use TSO ISPF to change the "prefix" in the data set names

Use TSO ISPF to change the "prefix" in the data set names (DSN) in the **USOURCE** DD statement, the **SOURCE** DD statement, the **CPEDATA** DD statement, the **HISTORY** DD statement, the **PDBLIB** DD statement, and the **SYSIN** DD statement of the JCL in accordance with your installation standards. The JCL is illustrated in Exhibit 4-1. (A "shell" of this JCL is contained on the distribution tape as "DB2JCL1")

- The CPEDATA DD statement in Exhibit 4-1 refers to the SAS data library maintained by CPExpert. The space for this library was created during the installation of CPExpert.

```
//jobname     JOB    job card information
//STEP01      EXEC   SAS,OPTIONS='MACRO DQUOTE PAGESIZE=65 ERRORABEND'
//USOURCE     DD     DSN=prefix.CPEXPERT.USOURCE,DISP=SHR
//SOURCE      DD     DSN=prefix.CPEXPERT.SOURCE,DISP=SHR
//CPEDATA     DD     DSN=prefix.CPEXPERT.CPEDATA,DISP=OLD
//HISTORY     DD     DSN=prefix.CPEXPERT.HISTORY,DISP=OLD
//PDBLIB      DD     DSN=prefix.MXG.PDBLIB.SASDB,DISP=SHR
//LIBRARY     DD     DSN=saslib containing MXG FORMATs
//SYSIN       DD     DSN=prefix.CPEXPERT.SOURCE(DB2CPE),DISP=SHR
```

JOB CONTROL LANGUAGE TO EXECUTE THE DB2CPE MODULE

EXHIBIT 4-1

- The **optional** HISTORY DD statement in Exhibit 4-1 refers to the SAS history data library maintained by CPExpert. The space for this library was created during the installation of CPExpert.

- The PDBLIB DD statement in Exhibit 4-1 refers to the SAS library containing the performance data base to be analyzed. The example shows a sample DSN for a typical MXG performance data base. The DSN would be changed to "DSN=prefix.DB2.MICS.DETAIL" to use a MICS performance data base.

- The SASLIB DD statement in Exhibit 4-1 refers to the SAS library containing the MXG FORMATs. If you execute CPExpert against a MICS performance data base, the SASLIB DD statement would be changed to refer to the MICS FORMAT library.

Please refer to the CPExpert Installation Guide for other options that can be used with the JCL statements. For example, you can optionally include a JCL statement to specify a different location for the MICS HAR Information Area if you are using CPExpert to analyze DB2 data sharing and wish to provide the MICS HARCFH and HARCFR files to CPExpert.

Step 2: Make any appropriate changes to the DB2GUIDE Module

Before submitting the above JCL shown in Exhibit 4-1 and executing the DB2CPE Module, you should make appropriate changes to the

prefix.CPEXPERT.USOURCE(DB2GUIDE) module. These changes are described in Section 3 of this manual.

Step 3: Review the output from the DB2CPE Module

If any rules are produced, refer to the specific rule description in Appendix A of this document for a description of the rule, a discussion of why the rule fired, and a suggestion for actions that should be taken.

Depending upon the output, you may wish to make changes or wait to see if the problems are identified in an analysis of a subsequent day's performance.

The following points should be considered in deciding whether to make a change:

- The DB2 Component may identify problems which clearly should be solved because their effect is so serious. In many cases, once the problem is identified, users immediately realize that the problem and suggested solutions make sense.

- The DB2 Component may identify problems which you do not feel will commonly occur. For example, you may realize that the results are based upon abnormal workload and changes may be unnecessary since the conditions will not occur often.

- **You generally should make only one change at a time if you decide to make changes!** This sound tuning advice if founded on the principles that:

  - Tuning is an art. No one (and certainly not CPExpert) can guarantee that any particular change will have a beneficial effect in all environments.

  - Changes may have unexpected effects. Most systems are complex, parameters may improve performance of one area at the expense of performance in another area, and management may wish resources focused on the second area.

  - If you make multiple changes and performance deteriorates, you will be unable to identify easily the change causing the problem. You are then faced with the problem of backing out all of the changes and starting over, one at a time.

  - Some changes are not "precise" in that, for example, keyword values might need to be adjusted a little at a time until a suitable value is reached. If multiple changes are made, you will be unable to detect the effect of the fine- tuning of the changes.

- Above all, **remember that the recommendations from CPExpert are simply options** to be considered in the context of overall objectives. You must decide

whether the recommendations are reasonable. Rarely should a recommendation be implemented without first evaluating how the recommendation will effect other workloads.

Please remember that CPExpert is not intended to replace a performance analyst. Rather, CPExpert was developed to help analyze the performance of MVS systems. CPExpert automates much of the routine of computer performance evaluation. Performance analysts can then focus on the areas which are not routine and which "require thinking".

With this philosophy, please let us know when you discover areas in which CPExpert could have helped you analyze a problem. We will improve our product and you will have more help!

Chapter 2: Checklist

The following checklist should be followed to execute the DB2 Component of CPExpert.

Checklist for Executing the DB2 Component, Mainframe

- ☐ Change the job card information in the JCL.
- ☐ Change the "prefix" in the data set names in the DD statements.
  - ☐ Change the "prefix" in USOURCE DD statement.
  - ☐ Change the "prefix" in SOURCE DD statement.
  - ☐ Change the "prefix" in CPEDATA DD statement.
  - ☐ Change the "prefix" in the **optional** HISTORY DD statement.
  - ☐ Change the "prefix" in PDBLIB DD statement.
  - ☐ Change the "prefix" in SASLIB DD statement.
- ☐ Make any necessary changes to the DB2GUIDE Module in USOURCE.
- ☐ Submit the JCL to execute the DB2CPE Module.

©Copyright 1994, Computer Management Sciences, Inc. Revised: October, 2001

Description of Rules

Appendix A

©Copyright 1999, Computer Management Sciences, Inc.

This appendix contains a description of each rule that results in a finding by the DB2 Component of CPExpert. The description summarizes the rule, discusses the rationale for the finding, suggests action, and provides detailed references to IBM publications or other documents.

The summary of the rule presents a short description of the finding.

The discussion describes as much as necessary of the operation of the computer system (the hardware, DB2, MVS, etc.) as it relates to the particular rule. The purpose of the discussion is to explain the reasoning behind the rule, and what causes the rule to be produced.

The suggestions list possible actions that should be considered based on the findings. In many cases, multiple possible actions are listed. You must determine which actions should be taken (this determination is based upon the suitability of the actions to your own environment, the financial implications of the action, and the "political" acceptability of the action.)

It is important to realize that these findings normally identify a POTENTIAL problem. Your systems programming staff must decide whether the findings (and their associated recommendations) make sense in your environment. For example, your systems programming staff might have deliberately selected certain parameter values. The values might be appropriate for your installation and your management objectives, even though CPExpert might produce a rule indicating that there is a potential problem with the parameter.

The rules are organized in numerical order. However, not all numbers are represented (for example, RULE DB2-110 follows RULE DB2-101). The LIST OF RULES in this appendix lists all rules that are included in the initial release of the DB2 Component. Within the rule framework, the following general categories apply:

- **General Findings**. The General Findings are rules in the DB-100(series) range. These findings help identify problems or potential problems with overall DB2 operation.

- **Local Buffer Findings**. Local Buffer Findings are rules in the DB-200(series) range. These findings help identify problems or potential problems with local buffer composition.

- **Lock Manager Findings**. Lock Manager Findings are rules in the DB-300(series) range. These findings help identify problems or potential problems with the Lock Manager.

©Copyright 1999, Computer Management Sciences, Inc. Revised: October, 2000

- **DDF Findings**. DDF Findings are rules in the DB-400(series) range. These findings help identify problems or potential problems with IBM's Distributed Data Facility.

- **Log Manager Findings**. Log Manager Findings are rules in the DB-500(series) range. These findings help identify problems or potential problems with the Log Manager.

- **Data Sharing Findings**. Data Sharing Findings are rules in the DB-600(series) range. These findings help identify problems or potential problems with data sharing and use of the Coupling Facility.

You may wish to read all of the rules in this appendix, just to see the type of problems that are encountered in different installations. However, it is not necessary to read all of the rules. It is necessary only to read the rules that apply to your installation. The rules that apply to your installation are identified by the report produced from the DB2CPE Module.

©Copyright 1999, Computer Management Sciences, Inc.

List of Rules

RULE	DESCRIPTION
DB2-101	Threads were queued at CREATE
DB2-110	Buffer shortage occurred for SMF destination
DB2-111	Recording interval for DB2 statistics might be too long
DB2-120	Invalid select procedure was encountered
DB2-121	Allied task (non-DB2) ABENDED
DB2-122	Allied task (non-DB2) deleted by MVS
DB2-131	Stored procedure terminated abnormally
DB2-132	SQL CALL statement timed out
DB2-133	SQL CALL statement was rejected
DB2-161	Migrated data sets were encountered
DB2-162	Applications experienced recall timeouts
DB2-181	RID list processing terminated, exceeded calculated limit
DB2-182	RID list processing terminated, exceeded physical limit
DB2-183	RID list processing exhausted virtual storage
DB2-184	Maximum RID pool storage was exceeded

©Copyright 1999, Computer Management Sciences, Inc.

RULE	DESCRIPTION
DB2-201	There might be insufficient buffers for work files
DB2-202	There were insufficient buffers for work files in merge passes
DB2-203	Merge passes were not efficiently performed
DB2-206	Hiperpool read requests failed (pages stolen by system)
DB2-207	Hiperpool write requests failed (pages stolen by system)
DB2-208	Virtual buffer pool was full
DB2-209	Active buffers account for large percent of buffer pool
DB2-210	Read requests (using ADMF) failed
DB2-211	Write requests (using ADMF) failed, expanded storage not available
DB2-215	Total DB2 buffer pool page fault rate was high
DB2-216	Page fault rate was high for individual buffer pool
DB2-220	Data management buffer critical threshold (DMTH) was reached
DB2-221	Immediate writes occurred
DB2-222	Deferred write threshold (DWQT) was reached
DB2-223	VPSEQT value might be too small
DB2-224	VPSEQT value might be too large
DB2-225	DWQT and VDWQT might be too large
DB2-226	DWQT, VDWQT, or VPSEQT might be too small

©Copyright 1999, Computer Management Sciences, Inc. Revised: October, 2000

RULE	DESCRIPTION
DB2-230	Sequential prefetch was disabled - buffer shortage
DB2-231	Sequential prefetch was disabled - unavailable read engine
DB2-232	Sequential prefetch not scheduled, prefetch quantity = 0
DB2-233	Synchronous read I/O and sequential prefetch was high
DB2-234	Write engine was not available for asynchronous I/O
DB2-235	Dynamic sequential prefetch was invoked frequently
DB2-236	Synchronous read I/O was high
DB2-241	Parallel groups reduced due to buffer shortage
DB2-242	Prefetch quantity reduced to one-half of normal
DB2-243	Prefetch quantity reduced to one-quarter of normal
DB2-244	Prefetch I/O streams were denied, shortage of buffers
DB2-245	Page requested for a parallel query was unavailable
DB2-246	Parallel groups fell back to sequential mode
DB2-261	Failures were caused by full EDM pool
DB2-262	Low percent of Data Base Descriptors found in EDM pool
DB2-263	Low percent of cursor table sections found in EDM pool
DB2-264	Low percent of package table sections found in EDM pool
DB2-265	Size of EDM pool could be reduced

©Copyright 1999, Computer Management Sciences, Inc.

RULE	DESCRIPTION
DB2-301	Work was suspended because of lock conflict
DB2-302	Locks were escalated to shared mode
DB2-303	Locks were escalated to exclusive mode
DB2-305	Lock escalation was not effective
DB2-310	Work was suspended for longer than time-out value
DB2-311	Deadlocks were detected

©Copyright 1999, Computer Management Sciences, Inc. Revised: October, 2000

RULE	DESCRIPTION
DB2-401	Resynchronization connections attempted
DB2-402	Network problems might exist
DB2-403	Threads became indoubt with the remote location
DB2-421	Remote create thread requests rejected, ZPARM limit was reached
DB2-422	Remote threads were queued, ZPARM limit was reached
DB2-423	Database access thread was queued, ZPARM limit reached

©Copyright 1999, Computer Management Sciences, Inc. Revised: October, 2000

RULE	DESCRIPTION
DB2-501	Archive log read allocations exceeded guidance
DB2-502	Archive log write allocations exceeded guidance
DB2-511	Waits were caused by unavailable output log buffer
DB2-512	Log reads satisfied from active log data set
DB2-513	Log reads were satisfied from archive log data set
DB2-515	Failed look-ahead tape mounts
DB2-521	Read accesses delayed because of unavailable resources
DB2-522	Read accesses were delayed, tape volume contention

©Copyright 1999, Computer Management Sciences, Inc. Revised: October, 2000

RULE	DESCRIPTION
DB2-601	Coupling facility read requests could not complete
DB2-602	Coupling facility write requests failed (storage)
DB2-603	Low read-hit percentage for cross invalidated pages
DB2-605	Castout engine was not available
DB2-606	Coupling facility write engine was not available
DB2-641	Parallelism coordinator had to bypass a DB2 (buffers)
DB2-642	Parallel groups executed on a single DB2 (RR/RS)
DB2-643	Parallel groups executed on a single DB2 (COORDINATOR=NO)
DB2-644	Assisting DB2 had VPSEQT, VPPSEQT, or VPXPSEQT = zero
DB2-651	Lock contention was high
DB2-652	False lock contention was high
DB2-660	Service time was high for synchronous requests
DB2-661	Service time was high for asynchronous requests
DB2-662	Subchannel contention was high for synchronous requests
DB2-665	Too many synchronous requests were changed to asynchronous
DB2-671	Excessive cross-invalidations due to directory reclaims

©Copyright 1999, Computer Management Sciences, Inc.

Rule DB2-101: Threads were queued at CREATE

Finding: Threads were queued because the number of current threads had reached the value of the MAX USERS parameter.

Impact: This finding can have a MEDIUM IMPACT, or HIGH IMPACT on the performance of the DB2 subsystem.

Discussion: A thread is the DB2 structure that describes an application's connection, traces its progress, processes resource functions, and delimits its accessibility to DB2 resources and services. Most DB2 functions execute under a thread structure. An allied thread is a thread originating at the local DB2 subsystem that may access data at a remote DB2 subsystem. A database access thread is a thread accessing data at the local subsystem on behalf of a remote subsystem.

The MAX USERS parameter on the DSNTIPE panel specifies the maximum number of users or allied threads that can be allocated concurrently[1]. New threads are queued when the number of current threads reaches the value specified. Prior to DB2 Version 4, the default for the MAX USERS parameter is 30, meaning that new threads are queued when the number of current threads reaches 30. With DB2 Version 4, t he default for the MAX USERS parameter is 70, meaning that new threads are queued when the number of current threads reaches 70.

DB2 counts the following as separate users:

- Each TSO user (whether running a DSN command or a DB2 request from QMF)
- Each batch job (whether running a DSN command or a DB2 utility)
- Each IMS region that can access DB2
- Each active CICS transaction that can access DB2
- Each task connected to DB2 through the Call Attachment Facility. Queuing for threads causes elongation of transaction response time, and can result in significantly poor performance.

[1] *The MAX REMOTE ACTIVE on the DSNTIPE* panel specifies the maximum number of database access threads that can be allocated concurrently. The combined maximum allowed for MAX USERS and MAX REMOTE ACTIVE) is 2000. New thread allocations are queued when the combined maximum reaches 2000. The MAX USERS and MAX REMOTE ACTIVE parameters are reflected in ZPARMS as CTHREAD and MAXDBAT, respectively.

©Copyright 1999, Computer Management Sciences, Inc.

CPExpert compares the Q3STCTHW variable in DB2STATS (the number of create thread requests queued) with the **Q3STCTHW** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-101 when the number of create thread requests queued exceeds the value specified by the **Q3STCTHW** guidance variable.

The default value for the **Q3STCTHW** guidance variable is 0, indicating that CPExpert should produce Rule DB2-101 when any requests were queued because of the MAX USERS value.

The following example illustrates the output from Rule DB2-101:

Suggestion: The MAX USERS value and MAX REMOTE ACTIVE value should be set these values to provide good response time without wasting resources, such as virtual and real storage. The value you specify depends upon your machine size, your work load, and other factors. When specifying values for these fields, consider the following:

- Specifying fewer threads than needed under-utilize the processor and cause queuing for threads.

- Specifying more threads than needed do not improve the response time. They require more real storage for the additional threads and might cause more paging and, hence, performance degradation.

If real storage is the limiting factor, set MAX USERS and MAX REMOTE ACTIVE according to the available storage. For more information on storage, refer to Section 2 of the *DB2 Installation Guide*.

For the TSO and call attachment facilities, you can limit the number of threads indirectly by choosing values for the MAX TSO CONNECT and MAX BATCH CONNECT fields of installation panel DSNTIPE. These values limit the number of connections to DB2. The number of threads and connections allowed affects the amount of work that DB2 can process.

Thread creation is a significant cost for small and medium transactions. When execution of a transaction is terminated, the thread can sometimes be reused by another transaction using the same plan. For more information on thread reuse, see "Providing for Thread Reuse" in the *DB2 Administration Guide*.

Under some circumstances, you might not be concerned that threads queue at CREATE. Indeed, such queuing might be a desired result to

control the amount of storage used by DB2. If this is the case, you should alter CPExpert's analysis by modifying the **Q3STCTHW** guidance variable in USOURCE(DB2GUIDE) to reflect a value at which queuing would be unacceptable.

Reference: DB2 for OS/390 Version 3: Administration Guide
Section 2.4.1.13 (Storage Sizes Panel: DSNTIPE)

DB2 for OS/390 Version 4: Installation Guide
Section 2.2.4 (Virtual Storage for Storage Pools and Working Storage)
Section 2.5.1.19 (Thread Management Panel: DSNTIPE)

DB2 for OS/390 Version 5: Installation Guide
Section 2.2.4 (Virtual Storage for Storage Pools and Working Storage)
Section 2.5.13 (Thread Management Panel: DSNTIPE)

DB2 for OS/390 Version 6: Installation Guide
Section 2.2.3 (Virtual Storage for Storage Pools and Working Storage)
Section 2.5.12 (Thread Management Panel: DSNTIPE)

©Copyright 1999, Computer Management Sciences, Inc.

Rule DB2-110: Buffer shortage occurred for SMF destination

Finding: SMF rejected DB2 trace records because of an SMF buffer shortage.

Impact: This finding has no known impact on the performance of the DB2 subsystem. However, the finding can have a HIGH IMPACT on applications that use the lost DB2 trace records (for example, performance information may be lost).

Discussion: When you install DB2, you can specify whether DB2 statistical, accounting, and audit trace data are to be collected. Additionally, you can request that DB2 collect the data by using the START TRACE command.

You must make some additional updates if you request that DB2 pass accounting and statistics data to SMF, including modifying the SYS1.PARMLIB member**SMFPRMxx** to reflect the proper TYPE subparameter of SYS and SUBSYS (if specified).

Additionally, you must allocate an adequate supply of SMF buffers since the default SMF buffer settings are probably insufficient. SMF rejects trace records sent to it if an SMF buffer shortage occurs. DB2 sends a message (DSNW133I TRACE DATA LOST) to the MVS operator when SMF rejects trace records.

It is not desirable that trace records be lost. While losing DB2 trace records has no impact on performance, the lost information can have a significant impact on other applications, such as performance analysis or performance reporting.

CPExpert selects records from DB2STATS relating to the SMF destination (QWSBNM = “SMF”). CPExpert compares the QWSBSBUF variable in DB2STATS (the number of buffer overrun errors) with the **QWSBSBUF** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-110 when the number of buffer overrun errors exceeds the value specified by the **QWSBSBUF** guidance variable.

The default value for the **QWSBSBUF** guidance variable is 0, indicating that CPExpert should produce Rule DB2-110 when there were any buffer overrun errors for the SMF destination.

The following example illustrates the output from Rule DB2-110:

©Copyright 1999, Computer Management Sciences, Inc.

```
RULE DB2-110: BUFFER SHORTAGE OCCURRED FOR SMF DESTINATION

   SMF rejects trace records if an SMF buffer shortage occurs, and DB2
   sends a message (DSNW133I TRACE DATA LOST) to the MVS operator. DB2
   trace records were rejected during the intervals shown below.  You
   should consider adding SMF buffers.

                                          TIMES TRACE RECORDS
   MEASUREMENT INTERVAL                   WERE REJECTED BY SMF
   21:32-21:47, 08SEP1998                          112
```

Suggestion: If your location uses SMF for performance data or global trace data, be sure that:

- Your SMF data sets are large enough to hold the data. SMF is set up to accept record type 102. (Specify member SMFPRMxx, for which 'xx' are two user-defined alphanumeric characters.)

- Your SMF buffers are large enough. Specify SMF buffering on the VSAM BUFSP parameter of the access method services DEFINE CLUSTER statement. **Do not use the default settings if DB2 performance or global trace data is sent to SMF**. Specify CISZ(4096) and BUFSP(81920) on the DEFINE CLUSTER statement for each SMF VSAM data set. These values are the minimum required for DB2; you might have to increase them, depending on your MVS environment. DB2 runs above the 16MB line of virtual storage in a cross-memory environment.

You can alter CPExpert's analysis by modifying the **QWSBSBUF** guidance variable in USOURCE(DB2GUIDE).

Reference: DB2 for OS/390 Version 3: Administration Guide
Section 2.5.7 (Installation Step 7: Record DB2 Data to SMF)

DB2 for OS/390 Version 4: Installation Guide
Section 2.6.7 (Installation Step 7: Record DB2 Data to SMF)

DB2 for OS/390 Version 5: Installation Guide
Section 2.6.7 (Installation Step 7: Record DB2 Data to SMF)

DB2 for OS/390 Version 6: Installation Guide
Section 2.6.7 (Installation Step 7: Record DB2 Data to SMF)

Rule DB2-120: Invalid select procedure was encountered

Finding: An invalid select procedure was encountered.

Impact: This finding can have a LOW IMPACT, or MEDIUM IMPACT on the performance of the DB2 subsystem.

Discussion: A select procedure is made invalid by applying DB2 maintenance that directly affects the select procedure. Plans and packages with invalid select procedures continue to function correctly, but with a small performance degradation.

DB2 statistics record the total number of columns (Rows times columns) for which an invalid select procedure was encountered. Invalid select procedures are bypassed by DB2, and this can cause some degradation in performance.

CPExpert compares the QISTCOLS variable in DB2STATS (the number of invalid select procedures) with the **QISTCOLS** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-120 when the number of invalid select procedures exceeds the value specified by the **QISTCOLS** guidance variable.

The default value for the **QISTCOLS** guidance variable is 0, indicating that CPExpert should produce Rule DB2-120 when any invalid select procedures were encountered.

The following example illustrates the output from Rule DB2-120:

```
RULE DB2-120: INVALID SELECT PROCEDURE WAS ENCOUNTERED

   A select procedure is made invalid by applying DB2 maintenance that
   directly affects the select procedure.  Invalid select procedures are
   bypassed by DB2, and this can cause some degradation in performance.
   Plans and packages with invalid select procedures continue to function
   correctly but incur some performance degradation. You should consider
   running REBIND for any plan or package that has encountered invalid
   select procedures to eliminate this degradation.  An invalid select
   procedure was encountered during the intervals shown below:

                                          TIMES INVALID SELECT
   MEASUREMENT INTERVAL                 PROCEDURES WERE ENCOUNTERED
   21:44-21:59, 08SEP1998                        270,864
```

©Copyright 1999, Computer Management Sciences, Inc.

Suggestion: To eliminate the performance degradation caused by invalid select procedures, rebind any plan or package with the invalid select procedure. Rebinding a plan or package reenables its select procedures. Use IFCID 0224 to determine which plans and packages should be rebound, and to prioritize which rebinds should be done first.

The IBM DSNWSMGS macro gives some excellent guidance in this regard:

> "_____IFCID 0224_____
>
> Rebinding a plan or package reenables its select procedures. Use IFCID 0224 to determine which plans and packages should be rebound, and to prioritize which rebinds should be done first. We recommend that the plans with the highest column count be rebound first.
>
> The IFCID 0224 column count is collected by plan, not by package. Therefore, if a plan's column count is nonzero, any packages executed by the plan should also be rebound."

This is the approach that one user of CPExpert implemented:

_START TRACE (PERFM) CLASS(30) IFCID(224)

After collecting the IFCID224 data, the site used a SAS procedure to summarize the information after having run MXG against the SMF records:

```
PROC FREQ DATA=PDB.T102S224;
   TABLES QWHCPLAN;
   WEIGHT QW0224CL;
```

The resulting output will give the plans with the most invalid selects, ordered by number of invalid selects. You can go down the ordered list, rebinding until the opportunities for significant performance improvement are exhausted.

You can alter CPExpert's analysis by modifying the **QISTCOLS** guidance variable in USOURCE(DB2GUIDE).

Reference: DB2 for OS/390 Version 3: DSNWSMGS macro (QISTCOLS variable)

DB2 for OS/390 Version 4: DSNWSMGS macro (QISTCOLS variable)

DB2 for OS/390 Version 5: DSNWSMGS macro (QISTCOLS variable)

DB2 for OS/390 Version 6: DSNWSMGS macro (QISTCOLS variable)

©Copyright 1999, Computer Management Sciences, Inc.

Rule DB2-111: Recording interval for DB2 statistics might be too long

Finding: CPExpert believes that the interval at which DB2 statistics are recorded to SMF should be specified as 15 minutes.

Impact: This finding can have NO IMPACT on the performance of the DB2 subsystem, but the finding can significantly improve performance analysis.

Discussion: DB2 writes statistics at intervals specified by the STATISTICS TIME parameter on the DSNTIPN panel. The default interval for DB2 to write the statistics is 30 minutes.

CPExpert noticed that the recording interval for the DB2 statistics was longer than 15 minutes. The default DB2 statistics recording interval is 30 minutes. However, 15 minute statistics recording interval provides significantly better view of DB2 performance and performance constraints. Recording DB2 statistics every 15 minutes provides a significantly better view of DB2 performance and performance constraints at 15 minute intervals increases overhead an insignificant amount.

Additionally, most MVS installations write RMF statistics to the SMF file every 15 minutes. Selecting 15 minutes for the DB2 statistics recording interval would allow the DB2 statistics to more easily be correlated with the overall systems performance information recorded by RMF.

CPExpert examines the DURATM variable in DB2STATS (the statistics interval duration). CPExpert produces Rule DB2-111 when the value for the DURATM variable is greater than 15, indicating that the interval statistics are written at more than 15 minute intervals.

The following example illustrates the output from Rule DB2-111:

```
RULE DB2-111: RECORDING INTERVAL FOR DB2 STATISTICS MIGHT BE TOO LONG

  CPExpert noticed that the recording interval for the DB2 statistics was
  longer than 15 minutes.  The default DB2 statistics recording interval
  is 30 minutes.  However, recording DB2 statistics every 15 minutes
  provides a significantly better view of DB2 performance and performance
  constraints.  Recording at 15 minute intervals increases overhead an
  insignificant amount.  CPExpert suggests that you consider recording
  the DB2 statistics at 15 minute intervals.  Please "turn off" this rule
  by inserting %LET DB2R111=OFF; in USOURCE(DB2GUIDE) if you are unable
  to make the change (so you are not annoyed by continual production of
  this finding).
```

Suggestion: CPExpert suggests that you consider recording the DB2 statistics at 15 minute intervals.

You can alter CPExpert's analysis by "turning off" this rule as described in Section 3 of this document, if you do not agree that DB2 statistics should be recorded at 15 minute intervals.

Reference: DB2 for OS/390 Version 3: Administration Guide
Section 2.4.1.19 (Tracing Panel: DSNTIPN)

DB2 for OS/390 Version 4: Installation Guide
Section 2.5.1.22 (Tracing Panel: DSNTIPN)

DB2 for OS/390 Version 5: Installation Guide
Section 2.5.16 (Tracing Panel: DSNTIPN)

DB2 for OS/390 Version 6: Installation Guide
Section 2.5.16 (Tracing Panel: DSNTIPN)

Rule DB2-121: Allied task (non-DB2) ABENDED

Finding: An allied task (such as CICS) ABENDED during the intervals being analyzed.

Impact: This finding can have a LOW IMPACT, MEDIUM IMPACT, or HIGH IMPACT on the performance of the DB2 subsystem. The level of impact depends on how much backout processing was required.

Discussion: When an application program (allied task) fails, DB2 will back out any updates, returning the data to its state at the start of the unit of recovery; that is, DB2 undoes the work. These actions bring the tables back to the point of the last commit (or to the beginning of the tables if no commits were issued). The back out process also starts when an allied agent requests a rollback[1], and when deadlocks or timeouts occur.

For example, when a CICS application ABENDS, a message is sent to the system operator, and the failing unit of recovery is backed out in both CICS and DB2.

During the backout process, DB2 must read all log data, first from the log buffers, then from the active log, and finally from the archive log (if necessary).

Backouts can have a significant impact on system performance, depending on how much work DB2 must accomplish to perform the backout. This is because the backout is performed at the dispatching priority of the DB2 System Services Address Space, and operates in SRB mode.

Additionally, the backout uses synchronous I/O to retrieve the necessary pages from the table and restore the original data values to the columns that were updated. While the backout is in process, DB2 logs new "undo -redo" records. This causes the log to fill even faster than it did during the original update, and may force additional new archiving.

CPExpert compares the Q3STMEOT variable in DB2STATS (the number of times an allied task ABENDED) with the **Q3STMEOT** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-121 when the number of times an allied task ABENDED exceeds the value specified by the **Q3STMEOT** guidance variable.

---

[1] An allied agent might request a rollback because of a catastrophic error in the data, but not one that causes the program to ABEND. In such a case, the program might request a rollback, and terminate.

The default value for the **Q3STMEOT** guidance variable is 0, indicating that CPExpert should produce Rule DB2-121 when any requests were queued because of the CTHREAD value.

The following example illustrates the output from Rule DB2-121:

```
RULE DB2-121: NON-DB2 TASKS ABENDED WHILE CONNECTED TO  DB2

   Non-DB2 allied agent task was ABENDED while connected to DB2 during the
   intervals shown below:

                                          TIMES NON-DB2
   MEASUREMENT INTERVAL                   TASK ABENDED
    3:26- 3:55, 15SEP1999                        2
    7:55- 8:25, 15SEP1999                        3
    9:54-10:24, 15SEP1999                        7
   11:24-11:56, 15SEP1999                       13
   11:54-12:24, 15SEP1999                        3
   12:24-12:54, 15SEP1999                       17
```

Suggestion: If Rule DB2-121 is produced regularly, you should consider the following alternatives:

- Analyze the non-DB2 allied agents that ABENDED. CPExpert will produce a report that provides information about the jobs that experienced ABENDs, the connection name of the jobs, and the resources that were used before the jobs abended.
  - For DSN processes running in TSO foreground, the connection name "TSO" is used.
  - For DSN processes running in TSO batch, the connection name “BATCH” is used.
  - For the Call Attachment Facility (CAF), the connection name “DB2CALL”is used.
  - For the Recoverable Resource Manager Services attachment facility (RRSAF), the connection name “RRSAF” is used.
  - For IMS and CICS processes, the connection name is the system identification name.
- You can alter CPExpert’s analysis by modifying the **Q3STMEOT** guidance variable in USOURCE(DB2GUIDE).

Reference: “Performance Impacts of DB2 Backout Processing”, Joel Goldstein, CMG’90 conference proceedings, pages 98-111.

Rule DB2-122: Allied task (non-DB2) deleted by MVS (end of memory)

Finding: An allied task (such as CICS) was deleted by MVS because of an end of memory condtion, during the intervals being analyzed.

Impact: This finding can have a LOW IMPACT, MEDIUM IMPACT, or HIGH IMPACT on the performance of the DB2 subsystem. The level of impact depends on how much backout processing was required.

Discussion: When an application program (allied task) fails, DB2 will back out any updates, returning the data to its state at the start of the unit of recovery; that is, DB2 undoes the work. These actions bring the tables back to the point of the last commit (or to the beginning of the tables if no commits were issued). The back out process also starts when an allied agent requests a rollback[1], and when deadlocks or timeouts occur.

For example, when a CICS application ABENDS due to an end of memory condition, a message is sent to the system operator, and the failing unit of recovery is backed out in both CICS and DB2.

During the backout process, DB2 must read all log data, first from the log buffers, then from the active log, and finally from the archive log (if necessary).

Backouts can have a significant impact on system performance, depending on how much work DB2 must accomplish to perform the backout. This is because the backout is performed at the dispatching priority of the DB2 System Services Address Space, and operates in SRB mode.

Additionally, the backout uses synchronous I/O to retrieve the necessary pages from the table and restore the original data values to the columns that were updated. While the backout is in process, DB2 logs new “undo -redo” records. This causes the log to fill even faster than it did during the original update, and may force additional new archiving.

CPExpert compares the Q3STMEOM variable in DB2STATS (the number of times an allied task was deleted by MVS) with the **Q3STMEOM** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-122 when the number of times an allied task was deleted by MVS exceeds the value specified by the **Q3STMEOM** guidance variable.

---

[1] An allied agent might request a rollback because of a catastrophic error in the data, but not one that causes the program to ABEND. In such a case, the program might request a rollback, and terminate.

©Copyright 1999, Computer Management Sciences, Inc.

The default value for the **Q3STMEOM** guidance variable is 0, indicating that CPExpert should produce Rule DB2-122 when any allied task was deleted by MVS.

The following example illustrates the output from Rule DB2-122:

```
RULE DB2-122: NON-DB2 TASKS DELETED BY MVS (END OF MEMORY)

   Non-DB2 allied agent task was deleted by MVS while connected to DB2
   during the intervals shown below:

                                          TIMES NON-DB2
   MEASUREMENT INTERVAL                 DELETED BY MVS
   21:31-21:46, 08SEP1998                       16
```

Suggestion: If Rule DB2-122 is produced regularly, you should consider the following alternatives:

- Analyze the non-DB2 allied agents that ABENDED. CPExpert will produce a report that provides information about the jobs that experienced ABENDs, the connection name of the jobs, and the resources that were used before the jobs abended.
  - For DSN processes running in TSO foreground, the connection name "TSO" is used.
  - For DSN processes running in TSO batch, the connection name “BATCH” is used.
  - For the Call Attachment Facility (CAF), the connection name “DB2CALL”is used.
  - For the Recoverable Resource Manager Services attachment facility (RRSAF), the connection name “RRSAF” is used.
  - For IMS and CICS processes, the connection name is the system identification name.
- You can alter CPExpert’s analysis by modifying the **Q3STMEOT** guidance variable in USOURCE(DB2GUIDE).

Reference: “Performance Impacts of DB2 Backout Processing”, Joel Goldstein, CMG’90 conference proceedings, pages 98-111.

Rule DB2-131: Stored procedure terminated abnormally

Finding: Stored procedures terminated abnormally.

Impact: This finding can have a MEDIUM IMPACT or HIGH IMPACT on the performance of the DB2 subsystem. The level of impact depends on how many times stored procedures abnormally terminate.

Discussion: The DB2 stored procedures function is available with DB2 Version 4. DB2's stored procedures function offers significant time and overhead savings for applications using DRDA[1]. Requests from remote applications can call a DB2 server to execute a compiled program that contains SQL statements. DB2 Version 5 extended this support and provides stored procedure query result sets.

A stored procedure is simply a DB2 application compiled program, containing SQL statements, that runs in a DB2-managed or WLM-managed stored procedures address space. With a single operation, a series of SQL statements are executed in the stored procedure, thus significantly decreasing the costs of distributed SQL statement processing. Consequently, stored procedures can greatly reduce the message traffic between requester and server. They also allow servers to execute sensitive procedures in a more secure environment than workstations typically provide. You must have Language Environment to run stored procedures.

Applications that run in a stored procedures address space can access any resources available to MVS address spaces, such as VSAM files, flat files, MVS/APPC conversations, and IMS or CICS transactions.

If a stored procedure abnormally terminates:

- The calling program receives an SQL error as notification that the stored procedure failed.
- DB2 places the calling program's unit of work in a "must rollback" state.
- If the stored procedure does not handle the ABEND condition, DB2 refreshes the Language Environment to recover the storage that the

[1] Stored procedures can be also be accessed locally.

©Copyright 1999, Computer Management Sciences, Inc.

application uses. In most cases, the Language Environment does not need to restart.

CPExpert compares the QXCALLAB variable in DB2STATS (the number of times stored procedure calls terminated abnormally) with the **QXCALLAB** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-131 when the number of stored procedure calls terminated abnormally exceeds the value specified by the **QXCALLAB** guidance variable.

The default value for the **QXCALLAB** guidance variable is 0, indicating that CPExpert should produce Rule DB2-131 when any stored procedure calls terminated abnormally.

The following example illustrates the output from Rule DB2-131:

```
RULE DB2-131: STORED PROCEDURE TERMINATED ABNORMALLY

   Stored procedures terminated abnormally during the intervals shown
below:
                                          TIMES STORED PROCEDURES
   MEASUREMENT INTERVAL                    TERMINATED ABNORMALLY
   21:44-21:59, 08SEP1998                             1
```

Suggestion: If Rule DB2-131 is produced, you should consider the following alternatives:

- From a DB2 system view, you can set a limit for the number of times a procedure can terminate abnormally. This is done by specifying a value in field MAX ABEND COUNT on installation panel DSNTIPX. Implementing this alternative prevents a problem procedure from overwhelming the system with ABEND dump processing.

  The default value for the MAX ABEND COUNT is 0, which mean that the first ABEND of a stored procedure or user defined function causes SQL CALLs to that procedure or function to be rejected. For production systems, IBM suggests that this default value be used.

- You should request that the application programming staff review the problem procedure to determine why abnormal termination occurs, and correct the problem.

- You can alter CPExpert's analysis by modifying the **QXCALLAB** guidance variable in USOURCE(DB2GUIDE). You normally should **not**

alter this guidance variable, however. Even in a test system, you should be aware of the conditions under which stored procedures terminated abnormally!

Reference: DB2 for OS/390 Version 4: Installation Guide
Section 2.5.1.33 (Stored Procedures Parameters Panel: DSNTIPX)

DB2 for OS/390 Version 4: Administration Guide
Section 5.5.1.5 (Limit Resources for a Stored Procedure)

DB2 for OS/390 Version 5: Installation Guide
Section 2.5.29 (Stored Procedures Parameters Panel: DSNTIPX)

DB2 for OS/390 Version 5: Administration Guide
Section 5.5.1.5 (Limit Resources for a Stored Procedure)

DB2 for OS/390 Version 6: Installation Guide
Section 2.5.29 (Stored Procedures Parameters Panel: DSNTIPX)

DB2 for OS/390 Version 6: Administration Guide
Section 5.5.1.5 (Limit Resources for a Stored Procedure)

©Copyright 1999, Computer Management Sciences, Inc.

Rule DB2-132: SQL CALL statement timed out

Finding: An SQL CALL statement to a stored procedure timed out while waiting to be scheduled.

Impact: This finding can have a MEDIUM IMPACT or HIGH IMPACT on the performance of the DB2 subsystem. The level of impact depends on how many times stored procedure SQL CALL statements timed out.

Discussion: The DB2 stored procedures function is available with DB2 Version 4. DB2's stored procedures function offers significant time and overhead savings for applications using DRDA[1]. Requests from remote applications can call a DB2 server to execute a compiled program that contains SQL statements.

A stored procedure is simply a DB2 application compiled program, containing SQL statements, that runs in a DB2-managed or WLM-managed stored procedures address space. With a single operation, a series of SQL statements are executed in the stored procedure, thus significantly decreasing the costs of distributed SQL statement processing. Consequently, stored procedures can greatly reduce the message traffic between requester and server. They also allow servers to execute sensitive procedures in a more secure environment than workstations typically provide.

Applications that run in a stored procedures address space can access any resources available to MVS address spaces, such as VSAM files, flat files, MVS/APPC conversations, and IMS or CICS transactions.

The TIME OUT value on the DSNTIPX panel specifies the number of seconds before DB2 ceases to wait for an SQL CALL to be assigned to one of the task control blocks (TCBs) in a DB2 stored procedures address space. The SQL CALL statement fails if the time interval expires.

CPExpert compares the QXCALLTO variable in DB2STATS (the number of times an SQL CALL statement timed out) with the **QXCALLTO** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-132 when the number times an SQL CALL statement timed out of exceeds the value specified by the **QXCALLTO** guidance variable.

---

[1] Stored procedures can be also be accessed locally.

©Copyright 1999, Computer Management Sciences, Inc.

The default value for the **QXCALLTO** guidance variable is 0, indicating that CPExpert should produce Rule DB2-132 when any SQL CALL statements to stored procedures timed out.

Suggestion: If Rule DB2-132 is produced, you should consider the following alternatives:

- Determine whether the STOP PROCEDURE ACTION(QUEUE) command had been issued. This command can cause SQL CALL requests to be queued for a long time, and can result in a time out.

- Review the value that you have specified for the TIME OUT parameter on panel DSNTIPX. You might want to choose a higher value if your system has long queues and SQL CALL requests are timing out. A higher value might reduce the number of SQL CALL requests that time out, but could increase the waiting time for end-user requests. This is because the higher value would allow users to queue for a longer interval, and effectively mask the problems that cause the queue to be long.

- Determine whether the stored procedures are hanging onto the ssnmSPAS TCBs for too long. In this case, you need to review the application code to determine why this is happening.

- Determine whether Rule DB2-301 was produced by CPExpert, indicating that too many DB2 lock suspensions occurred[2]. If this is the case, you might have too many ssnmSPAS TCBs, causing them to encounter too many lock conflicts with one another.

- Determine whether the stored procedures calls quickly completing. In this case, individual SQL CALLs to stored procedures might execute satisfactorily, but perhaps you do not have enough ssnmSPAS TCBs to handle the work load. In this case, increase the value of field NUMBER OF TCBS on installation panel DSNTIPX.

- You can alter CPExpert's analysis by modifying the **QXCALLTO** guidance variable in USOURCE(DB2GUIDE). This would not normally be an appropriate action, however. Having SQL CALL requests time out should be cause for action and should not normally be ignored.

[2] You should verify whether you have changed the PCTSLOCK guidance variable in USOURCE(DB2GUIDE). The default value for the PCTSLOCK guidance variable is 0.5%, meaning that Rule DB2-301 will be produced when more than 0.5% of the locks are suspended. It is possible that you could have increased this value so that DB2-301 would not be produced along with DB2-132. Additionally, it is possible that the default value is too large (even at 0.5%) to show the lock suspensions that would cause SQL CALL requests to time out. In either case, you can examine the QTXASLOC variable in DB2STATS to determine whether lock suspensions occurred often.

Reference: DB2 for OS/390 Version 4: Installation Guide
Section 2.5.1.33 (Stored Procedures Parameters Panel: DSNTIPX)

DB2 for OS/390 Version 4: Administration Guide
Section 5.11.6.3 (Monitoring Stored Procedures)

DB2 for OS/390 Version 5: Installation Guide
Section 2.5.29 (Stored Procedures Parameters Panel: DSNTIPX)

DB2 for OS/390 Version 5: Administration Guide
Section 5.11.7.3 (Accounting Trace)

DB2 for OS/390 Version 6: Installation Guide
Section 2.5.29 (Stored Procedures Parameters Panel: DSNTIPX)

DB2 for OS/390 Version 6: Administration Guide
Section 5.11.6.3.1 (Accounting Trace)

©Copyright 1999, Computer Management Sciences, Inc.

Rule DB2-133: SQL CALL statement was rejected

Finding: An SQL CALL statement request to a stored procedure was rejected because the procedure was in the STOP ACTION(REJECT) state..

Impact: This finding can have a MEDIUM IMPACT or HIGH IMPACT on the performance of the DB2 subsystem.

Discussion: The DB2 stored procedures function is available with DB2 Version 4. DB2's stored procedures function offers significant time and overhead savings for applications using DRDA[1]. Requests from remote applications can call a DB2 server to execute a compiled program that contains SQL statements.

A stored procedure is simply a DB2 application compiled program, containing SQL statements, that runs in a DB2-managed or WLM-managed stored procedures address space. With a single operation, a series of SQL statements are executed in the stored procedure, thus significantly decreasing the costs of distributed SQL statement processing. Consequently, stored procedures can greatly reduce the message traffic between requester and server. They also allow servers to execute sensitive procedures in a more secure environment than workstations typically provide.

Applications that run in a stored procedures address space can access any resources available to MVS address spaces, such as VSAM files, flat files, MVS/APPC conversations, and IMS or CICS transactions.

The DB2 command STOP PROCEDURE prevents DB2 from accepting SQL CALL statements for one or more stored procedures. The STOP PROCEDURE command has two forms: ACTION(QUEUE) and ACTION(REJECT).

- ACTION(QUEUE) queues the request until either the queue wait time exceeds the installation timeout value, or the stored procedure is started by the command START PROCEDURE.

- ACTION(REJECT) stops access to all stored procedures, and terminate the DB2 stored procedures address space. While the STOP

[1] Stored procedures can be also be accessed locally.

©Copyright 1999, Computer Management Sciences, Inc.

PROCEDURE command is in effect, attempts to execute stored procedures are rejected.

DB2 implicitly issues the command STOP PROCEDURE ACTION(REJECT) for any stored procedure that exceeds the maximum ABEND count. That count is set by the MAX ABEND COUNT field of installation panel DSNTIPX.

CPExpert compares the QXCALLRJ variable in DB2STATS (the number of times an SQL CALL statement was rejected because the procedure was in the STOP ACTION(REJECT) state) with the **QXCALLRJ** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-133 when the number times an SQL CALL statement was rejected because the procedure was in the STOP ACTION(REJECT) state of exceeds the value specified by the **QXCALLRJ** guidance variable.

The default value for the **QXCALLRJ** guidance variable is 0, indicating that CPExpert should produce Rule DB2-133 when any SQL CALL statements were rejected because the procedure was in the STOP ACTION(REJECT) state.

Suggestion: As indicated in the discussion above, the STOP ACTION(REJECT) state can occur because of an operator action in stopping the stored procedures address space, or can occur because of DB2 taking an implicit action because the number of stored procedure ABENDs exceeded the value specified on the DSNTIPX panel.

SQL CALL rejects because of an operation action in stopping the stored procedures address space normally should be acceptable. However, SQL CALL rejects because number of stored procedure ABENDs exceeded the value specified on the DSNTIPX panel should be cause for action.

If Rule DB2-133 is produced, you should consider the following alternatives:

- Determine whether the STOP ACTION(REJECT) state was due to normal operator action in stopping the stored procedures address space. If this was the situation, you still might wish to investigate why the action was taken.

- Rule DB2-131 should have been produced if the STOP ACTION(REJECT) state was due to DB2 implicitly issuing the command because the stored procedures had exceeded the MAX ABEND value on panel DSNTIPX. This is because the default guidance for Rule DB2-131 is zero (indicating that Rule DB2-131 should be produced when any

stored procedures ABEND). Rule DB2-131 might not have been produced, however, if you have altered the QXCALLAB guidance variable in USOURCE(DB2GUIDE).

Regardless of whether Rule DB2-131 was produced, you should review the suggestions associated with Rule DB2-131 if Rule DB2-133 is produced.

- You can alter CPExpert's analysis by modifying the **QXCALLRJ** guidance variable in USOURCE(DB2GUIDE). You normally should **not** alter this guidance variable, however. You should be aware of any SQL CALL rejects, and determine whether they are cause for alarm!

Reference: DB2 for OS/390 Version 4: Installation Guide
Section 2.5.1.33 (Stored Procedures Parameters Panel: DSNTIPX)

DB2 for OS/390 Version 4: Administration Guide
Section 5.11.6.3 (Monitoring Stored Procedures)

DB2 for OS/390 Version 5: Installation Guide
Section 2.5.29 (Stored Procedures Parameters Panel: DSNTIPX)

DB2 for OS/390 Version 5: Administration Guide
Section 5.11.7.3 (Accounting Trace)

DB2 for OS/390 Version 6: Installation Guide
Section 2.5.29 (Stored Procedures Parameters Panel: DSNTIPX)

DB2 for OS/390 Version 6: Administration Guide
Section 5.11.6.3.1 (Accounting Trace)

Rule DB2-161: Migrated data sets were encountered

Finding: DB2 applications attempted to reference data sets that had been migrated by DFSMShsm.

Impact: This finding can have a LOW IMPACT, MEDIUM IMPACT, or HIGH IMPACT on the performance of the DB2 subsystem. The level of impact depends on how long applications were required to wait on the migrated data.

Discussion: The Hierarchical Storage Management functional component (DFSMShsm) of DFSMS/MVS manages space and data availability among the storage devices in your system. DFSMShsm can be used to move data sets that have not been recently used to slower, less expensive storage devices; this helps to ensure that DASD space is managed efficiently.

DFSMShsm can automatically migrate and recall archive log and image copy data sets. If DB2 needs an archive log data set or an image copy data set that DFSMShsm has migrated, a recall begins automatically and DB2 waits for the recall to complete before continuing.

DB2 table spaces and index spaces can also be automatically migrated and recalled.

If you accepted the default value YES for the RECALL DATABASE parameter on the Operator Functions panel (DSNTIPO), DB2 recalls migrated table spaces and index spaces. At data set open time, DB2 waits for DFSMShsm to perform the recall. The amount of time DB2 waits while the recall is being performed is specified on the RECALL DELAY parameter, which is also on panel DSNTIPO. If RECALL DELAY is set to zero, DB2 does not wait, and the recall is performed asynchronously.

CPExpert compares the QBSTMIG variable in DB2STATB (the number of times migrated data sets were encountered) with the **QBSTMIG** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-161 when the number of times migrated data sets were encountered exceeds the value specified by the **QBSTMIG** guidance variable.

The default value for the **QBSTMIG** guidance variable is 0, indicating that CPExpert should produce Rule DB2-161 when any migrated data sets were encountered.

©Copyright 1999, Computer Management Sciences, Inc.

The following example illustrates the output from Rule DB2-161:

```
RULE DB2-161: MIGRATED DATA SETS WERE ENCOUNTERED

   Buffer Pool 10: DB2 applications attempted to reference data sets
   that had been migrated by DFSMShsm. You should consider recalling the
   migrated data sets before starting DB2 or avoid migrating the data sets
   by assigning a management class that prevents migration of the data
   sets. This situation occurred during the intervals shown below:

                                     TIMES MIGRATED DATA SETS
   MEASUREMENT INTERVAL                  WERE ENCOUNTERED
    2:53- 3:22, 15SEP1999                        3
```

Suggestion: If Rule DB2-161 is produced regularly, you should consider the following alternatives:

- DB2 subsystem data sets, including the DB2 catalog, DB2 directory, active logs, and temporary databases, can reside on system managed storage (SMS). However, you should consider recalling these migrated data sets via DFSMShsm before starting DB2.

- Alternatively, avoid migrating by assigning a management class that prevents migration to these data sets.

- You can alter CPExpert's analysis by modifying the **QBSTMIG** guidance variable in USOURCE(DB2GUIDE).

Reference: DB2 for OS/390 Version 3: Administration Guide
Section 7.6.6.3 (Using DFSMShsm to Manage Data Sets)
Section 2.4.1.20 (Operator Functions Panel: DSNTIPO)

DB2 for OS/390 Version 4: Installation Guide
Section 2.5.17 (Operator Functions Panel: DSNTIPO)

DB2 for OS/390 Version 4: Administration Guide
Section 5.5.6.1.3 (Using DFSMShsm to Manage Data Sets)

DB2 for OS/390 Version 5: Installation Guide
Section 2.5.17 (Operator Functions Panel: DSNTIPO)

DB2 for OS/390 Version 5: Administration Guide
Section 5.5.6.1.3 (Using DFSMShsm to Manage Data Sets)

DB2 for OS/390 Version 6: Installation Guide
Section 2.5.17 (Operator Functions Panel: DSNTIPO)

DB2 for OS/390 Version 6: Administration Guide
Section 2.7.3 (Managing your DB2 data sets with DFSMShsm)

©Copyright 1999, Computer Management Sciences, Inc.

Rule DB2-162: Applications experienced RECALL timeouts

Finding: DB2 applications attempted to reference data sets that had been migrated by DFSMShsm.

Impact: This finding can have a LOW IMPACT, MEDIUM IMPACT, or HIGH IMPACT on the performance of the DB2 subsystem. The level of impact depends on how long applications were required to wait on the migrated data.

Discussion: The Hierarchical Storage Management functional component (DFSMShsm) of DFSMS/MVS manages space and data availability among the storage devices in your system. DFSMShsm can be used to move data sets that have not been recently used to slower, less expensive storage devices; this helps to ensure that DASD space is managed efficiently.

DFSMShsm can automatically migrate and recall archive log and image copy data sets. If DB2 needs an archive log data set or an image copy data set that DFSMShsm has migrated, a recall begins automatically and DB2 waits for the recall to complete before continuing.

DB2 table spaces and index spaces can also be automatically migrated and recalled.

If you accepted the default value YES for the RECALL DATABASE parameter on the Operator Functions panel (DSNTIPO), DB2 recalls migrated table spaces and index spaces. At data set open time, DB2 waits for DFSMShsm to perform the recall. The amount of time DB2 waits while the recall is being performed is specified on the RECALL DELAY parameter, which is also on panel DSNTIPO. If RECALL DELAY is set to zero, DB2 does not wait, and the recall is performed asynchronously.

CPExpert compares the QBSTRTO variable in DB2STATB (the number of times migrated data sets were encountered) with the **QBSTRTO** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-162 when the number of times migrated data sets were encountered exceeds the value specified by the **QBSTRTO** guidance variable.

The default value for the **QBSTRTO** guidance variable is 0, indicating that CPExpert should produce Rule DB2-162 when any migrated data sets were encountered.

©Copyright 1999, Computer Management Sciences, Inc.

The following example illustrates the output from Rule DB2-162:

```
RULE DB2-162: APPLICATIONS EXPERIENCED RECALL TIMEOUTS

   Buffer Pool 10: DB2 applications attempted to reference data sets that
   had been migrated by DFSMShsm, and timed out before the RECALL was
   successful. You should consider (1) recalling the migrated data sets
   before starting DB2, (2) avoid migrating the data sets by assigning
   a management class that prevents migration of the data sets, or (3)
   increasing the value specified for the RECALL DELAY parameter on panel
   DSNTIPO.  This situation occurred during the intervals shown below:

                                   TIMES MIGRATED DATA SETS
   MEASUREMENT INTERVAL                WERE ENCOUNTERED
    2:53- 3:22, 15SEP1999                      1
```

Suggestion: If Rule DB2-162 is produced regularly, you should consider the following alternatives:

- DB2 subsystem data sets, including the DB2 catalog, DB2 directory, active logs, and temporary databases, can reside on system managed storage (SMS). However, you should consider recalling these migrated data sets via DFSMShsm before starting DB2.
- Alternatively, avoid migrating by assigning a management class that prevents migration to these data sets.
- Alternatively, increase the value specified for the RECALL DELAY parameter, which is on panel DSNTIPO.
- You can alter CPExpert's analysis by modifying the **QBSTRTO** guidance variable in USOURCE(DB2GUIDE).

Reference: DB2 for OS/390 Version 3: Administration Guide
Section 7.6.6.3 (Using DFSMShsm to Manage Data Sets)
Section 2.4.1.20 (Operator Functions Panel: DSNTIPO)

DB2 for OS/390 Version 4: Installation Guide
Section 2.5.17 (Operator Functions Panel: DSNTIPO)

DB2 for OS/390 Version 4: Administration Guide
Section 5.5.6.1.3 (Using DFSMShsm to Manage Data Sets)

DB2 for OS/390 Version 5: Installation Guide
Section 2.5.17 (Operator Functions Panel: DSNTIPO)

DB2 for OS/390 Version 5: Administration Guide
Section 5.5.6.1.3 (Using DFSMShsm to Manage Data Sets)

DB2 for OS/390 Version 6: Installation Guide
Section 2.5.17 (Operator Functions Panel: DSNTIPO)

DB2 for OS/390 Version 6: Administration Guide
Section 2.7.3 (Managing your DB2 data sets with DFSMShsm)

©Copyright 1999, Computer Management Sciences, Inc.

Rule DB2-181: RID list processing terminated, exceeded calculated limit

Finding: Processing of the Record Identifier (RID) list was terminated because the number of RID entries was greater than the limit of MAX(25% of table size, number of RIDs that can fit into the guaranteed number of RID blocks).

Impact: This finding can have a LOW IMPACT or MEDIUM IMPACT on the performance of the DB2 subsystem.

Discussion: A RID list pool is an area of main storage above the 16MB line that is reserved for sorting record identifiers during list prefetch processing. The RID pool is created at start up time, but no space is allocated until RID storage is needed. When RID storage is needed, it is allocated above the 16MB line in 16KB blocks known as RID blocks.

The RID pool is used for all record identifier (RID) processing. It is used for sorting RIDs during the following operations:

- List prefetch, including single index list prefetch,
- Access via multiple indexes
- Hybrid joins

RID pool storage is also used when DB2 enforces unique keys while updating multiple rows.

SQL statements that use those methods of access can benefit from using the RID pool. RID pool processing can help reduce I/O resource consumption and elapsed time.

Whether SQL statements that use RID processing complete efficiently or not depends on other concurrent work using the RID pool. Processing of the RID list is terminated if the number of RID entries was greater than the limit of MAX(25% of table size, number of RIDs that can fit into the guaranteed number of RID blocks). List prefetch terminates after the 25% threshold is passed because a table space scan is generally faster than list prefetch if more than 25% of the rows need to be retrieved.

When list prefetch is terminated, it affects access paths in different ways:

- If the access path is multiple index ORing or single index access using list prefetch, DB2 falls back to a table space scan.

©Copyright 1999, Computer Management Sciences, Inc.

- If the access path step is the index access portion of multiple index ANDing, that step terminates and processing continues with the next multiple index access step. If there is no remaining step, and no RID list has been accumulated, DB2 falls back to a table space scan.

CPExpert compares the QISTRLLM variable in DB2STATS with the **QISTRLLM** guidance variable in USOURCE(DB2GUIDE). The QISTRLLM value contains a count of the number of times RID list was terminated because the number of RID entries was greater than the limit of MAX(25% of table size, number of RIDs that can fit into the guaranteed number of RID blocks). CPExpert produces Rule DB2-181 when the value of the QISTRLLM variable in DB2STATS exceeds the value specified by the **QISTRLLM** guidance variable.

The default value for the **QISTRLLM** guidance variable is 0, indicating that CPExpert should produce Rule DB2-181 whenever processing of the RID list is terminated because the number of RID entries was greater than the limit of MAX(25% of table size, number of RIDs that can fit into the guaranteed number of RID blocks).

The following example illustrates the output from Rule DB2-181:

```
RULE DB2-181: RID LIST PROCESSING TERMINATED

   RID list processing was terminated because the number of RID entries
   was greater than the limit of MAX(25% of table size, number of RIDs
   that can fit into the guaranteed number of RID blocks).  This finding
   occurred during the intervals shown below:

                                   TIMES RID LIST PROCESSING
   MEASUREMENT INTERVAL                 WAS TERMINATED
    7:51- 8:21, 15SEP1999                        2
    8:21- 8:51, 15SEP1999                       47
    8:51- 9:21, 15SEP1999                       66
    9:21- 9:51, 15SEP1999                       52
    9:51-10:21, 15SEP1999                      242
   10:21-10:51, 15SEP1999                      177
   10:51-11:21, 15SEP1999                      103
```

Suggestion: When the DSNTINST CLIST calculates the value for RID POOL SIZE on panel DSNTIPC, the default is calculated as 50% of the sum of virtual buffer pools BP0, BP1, BP2, and BP32K.

You can modify the maximum RID pool size that was specified on installation panel DSNTIPC by using the installation panels in UPDATE mode, as follows:

- To favor the selection and efficient completion of list prefetch, multiple index access, or hybrid join, you can increase the maximum RID pool size.

- To disable list prefetch, multiple index access, and hybrid join, specify a RID pool size of 0.

  If you do this, plans or packages that were previously bound with a non-zero RID pool size might experience significant performance degradation. Rebind any plans or packages that include SQL statements that use RID processing.

Alternatively, you can curtail the number of activities concurrently using RID pools.

Alternatively, you can alter CPExpert's analysis by modifying the **QISTRLLM** guidance variable in USOURCE(DB2GUIDE).

Reference: DB2 for OS/390 Version 3: Administration Guide
Section 2.4.1.32 (Install DB2--CLIST Calculations Panel 1: DSNTIPC)
Section 7.5.3 (Increasing RID Pool Size)
Section 7.9.7.2 (Bind Time and Execution Time Thresholds)

DB2 for OS/390 Version 4: Installation Guide
Section 2.2.4 (Virtual Storage for Storage Pools and Working Storage)
Section 2.5.1.36 (Install DB2--CLIST Calculations Panel 1: DSNTIPC)

DB2 for OS/390 Version 4: Administration Guide
Section 5.10.5.2 (List Sequential Prefetch)

DB2 for OS/390 Version 5: Installation Guide
Section 2.2.4 (Virtual Storage for Storage Pools and Working Storage)
Section 2.5.32 (Install DB2--CLIST Calculations Panel 1: DSNTIPC)

DB2 for OS/390 Version 5: Administration Guide
Section 5.10.5.2.3 (Bind Time and Execution Time Thresholds)

DB2 for OS/390 Version 6: Installation Guide
Section 2.2.3 (Virtual Storage for Storage Pools and Working Storage)
Section 2.5.32 (Install DB2--CLIST Calculations Panel 1: DSNTIPC)

DB2 for OS/390 Version 6: Administration Guide
Section 5.10.5.2.3 (Bind Time and Execution Time Thresholds)

Rule DB2-182: RID list processing terminated, exceeded physical limit

Finding: Processing of the Record Identifier (RID) list was terminated because the number of RID entries was greater than the physical limit of approximately 16 million RIDs.

Impact: This finding can have a LOW IMPACT or MEDIUM IMPACT on the performance of the DB2 subsystem.

Discussion: A RID list pool is an area of main storage above the 16MB line that is reserved for sorting record identifiers during list prefetch processing. The RID pool is created at start up time, but no space is allocated until RID storage is needed. When RID storage is needed, it is allocated above the 16MB line in 16KB blocks known as RID blocks.

The RID pool is used for all record identifier (RID) processing. It is used for sorting RIDs during the following operations:

- List prefetch, including single index list prefetch,
- Access via multiple indexes
- Hybrid joins

RID pool storage is also used when DB2 enforces unique keys while updating multiple rows.

SQL statements that use those methods of access can benefit from using the RID pool. RID pool processing can help reduce I/O resource consumption and elapsed time. However, if there is not enough RID pool storage, it is possible that the statement might revert to a table space scan.

At bind time, DB2 does not consider list prefetch if the estimated number of RIDs to be processed would take more than 50% of the RID pool. RID storage is virtual storage acquired from the RID storage pool at the time the query is executed. You can modify the size of the RID pool by updating the RID POOL SIZE field on installation panel DSNTIPC. The maximum size of a RID pool is 1000MB. The size of a single RID list is limited to approximately 16 million RIDs. Processing of the RID list is terminated if the number of RID entries is greater than the physical limit (approximately 16 million RIDs).

When list prefetch is terminated, it affects access paths in different ways:

©Copyright 1999, Computer Management Sciences, Inc.

- If the access path is multiple index ORing or single index access using list prefetch, DB2 falls back to a table space scan.

- If the access path step is the index access portion of multiple index ANDing, that step terminates and processing continues with the next multiple index access step. If there is no remaining step, and no RID list has been accumulated, DB2 falls back to a table space scan.

CPExpert compares the QISTRPLM variable in DB2STATS with the **QISTRPLM** guidance variable in USOURCE(DB2GUIDE). The QISTRPLM value contains a count of the number of times RID list was terminated because the number of RID entries was greater than the physical limit of the RID pool. CPExpert produces Rule DB2-182 when the value of the QISTRPLM variable in DB2STATS exceeds the value specified by the **QISTRPLM** guidance variable.

The default value for the **QISTRPLM** guidance variable is 0, indicating that CPExpert should produce Rule DB2-182 whenever processing of the RID list is terminated because the number of RID entries was greater than the physical limit of the RID pool.

The following example illustrates the output from Rule DB2-182:

```
RULE DB2-182: RID LIST PROCESSING TERMINATED

   RID list processing was terminated because the number of RID entries
   was greater than the physical limit of approximately 16 million RIDs.
   This finding occurred during the intervals shown below:

                                   TIMES RID LIST PROCESSING
   MEASUREMENT INTERVAL                 WAS TERMINATED
   21:32-21:47, 08SEP1998                        1
```

Suggestion: You should consider curtailing the number of activities concurrently using RID pools if an RID processing was terminated because the number of RID entries was greater than the physical limit of the RID pool.

Alternatively, you can alter CPExpert's analysis by modifying the **QISTRPLM** guidance variable in USOURCE(DB2GUIDE).

Reference: DB2 for OS/390 Version 3: Administration Guide
Section 2.4.1.32 (Install DB2--CLIST Calculations Panel 1: DSNTIPC)
Section 7.5.3 (Increasing RID Pool Size)
Section 7.9.7.2 (Bind Time and Execution Time Thresholds)

DB2 for OS/390 Version 4: Installation Guide
Section 2.2.4 (Virtual Storage for Storage Pools and Working Storage)
Section 2.5.1.36 (Install DB2--CLIST Calculations Panel 1: DSNTIPC)

DB2 for OS/390 Version 4: Administration Guide
Section 5.10.5.2 (List Sequential Prefetch)

DB2 for OS/390 Version 5: Installation Guide
Section 2.2.4 (Virtual Storage for Storage Pools and Working Storage)
Section 2.5.32 (Install DB2--CLIST Calculations Panel 1: DSNTIPC)

DB2 for OS/390 Version 5: Administration Guide
Section 5.10.5.2.3 (Bind Time and Execution Time Thresholds)

DB2 for OS/390 Version 6: Installation Guide
Section 2.2.3 (Virtual Storage for Storage Pools and Working Storage)
Section 2.5.32 (Install DB2--CLIST Calculations Panel 1: DSNTIPC)

DB2 for OS/390 Version 6: Administration Guide
Section 5.10.5.2.3 (Bind Time and Execution Time Thresholds)

©Copyright 1999, Computer Management Sciences, Inc.

Rule DB2-183: RID list processing exhausted virtual storage

Finding: RID list processing was terminated because RID list processing exhausted virtual storage.

Impact: This finding can have a LOW IMPACT or MEDIUM IMPACT on the performance of the DB2 subsystem.

Discussion: A RID list pool is an area of main storage above the 16MB line that is reserved for sorting record identifiers during list prefetch processing. The RID pool is created at start up time, but no space is allocated until RID storage is needed. When RID storage is needed, it is allocated above the 16MB line in 16KB blocks known as RID blocks.

The RID pool is used for all record identifier (RID) processing. It is used for sorting RIDs during the following operations:

- List prefetch, including single index list prefetch,
- Access via multiple indexes
- Hybrid joins

RID pool storage is also used when DB2 enforces unique keys while updating multiple rows.

SQL statements that use those methods of access can benefit from using the RID pool. RID pool processing can help reduce I/O resource consumption and elapsed time. An SQL statement using RID processing might successfully complete or revert to a table space scan, depending on other concurrent work using the RID pool.

When list prefetch is terminated, it affects access paths in different ways:

- If the access path is multiple index ORing or single index access using list prefetch, DB2 falls back to a table space scan.
- If the access path step is the index access portion of multiple index ANDing, that step terminates and processing continues with the next multiple index access step. If there is no remaining step, and no RID list has been accumulated, DB2 falls back to a table space scan.

©Copyright 1999, Computer Management Sciences, Inc.

CPExpert compares the QISTRSTG variable in DB2STATS (the number of times RID list processing exhausted virtual storage) with the **QISTRSTG** guidance variable in USOURCE(DB2GUIDE).

CPExpert produces Rule DB2-183 when the number of times RID list processing exhausted virtual storage exceeds the value specified by the **QISTRSTG** guidance variable.

The default value for the **QISTRSTG** guidance variable is 0, indicating that CPExpert should produce Rule DB2-183 whenever RID list processing exhausted virtual storage.

The following example illustrates the output from Rule DB2-183:

```
RULE DB2-183: RID LIST PROCESSING EXHAUSTED VIRTUAL STORAGE

   RID list processing was terminated because RID list processing
   exhausted virtual storage.  This finding occurred during the intervals
   shown below:

                                   TIMES RID LIST PROCESSING
   MEASUREMENT INTERVAL                 WAS TERMINATED
   21:32-21:47, 08SEP1998                        1
```

Suggestion: You should consider the following alternatives:

- Increase the size of the region to allow more virtual storage for DB2.
- Curtail the work processed by the DB2 subsystem during the intervals shown.

- Alter CPExpert's analysis by modifying the **QISTRSTG** guidance variable in USOURCE(DB2GUIDE).

Reference: DB2 for OS/390 Version 3: Administration Guide
Section 2.4.1.32 (Install DB2--CLIST Calculations Panel 1: DSNTIPC)
Section 7.5.3 (Increasing RID Pool Size)
Section 7.9.7.2 (Bind Time and Execution Time Thresholds)

DB2 for OS/390 Version 4: Installation Guide
Section 2.2.4 (Virtual Storage for Storage Pools and Working Storage)
Section 2.5.1.36 (Install DB2--CLIST Calculations Panel 1: DSNTIPC)

DB2 for OS/390 Version 4: Administration Guide
Section 5.10.5.2 (List Sequential Prefetch)

DB2 for OS/390 Version 5: Installation Guide
Section 2.2.4 (Virtual Storage for Storage Pools and Working Storage)
Section 2.5.32 (Install DB2--CLIST Calculations Panel 1: DSNTIPC)

DB2 for OS/390 Version 5: Administration Guide
Section 5.10.5.2.3 (Bind Time and Execution Time Thresholds)

DB2 for OS/390 Version 6: Installation Guide
Section 2.2.3 (Virtual Storage for Storage Pools and Working Storage)
Section 2.5.32 (Install DB2--CLIST Calculations Panel 1: DSNTIPC)

DB2 for OS/390 Version 6: Administration Guide
Section 5.10.5.2.3 (Bind Time and Execution Time Thresholds)

©Copyright 1999, Computer Management Sciences, Inc.

Rule DB2-184: Maximum RID pool storage was exceeded

Finding: The maximum amount of Record Identifier (RID) pool storage was exceeded.

Impact: This finding can have a LOW IMPACT or MEDIUM IMPACT on the performance of the DB2 subsystem.

Discussion: A RID list pool is an area of main storage above the 16MB line that is reserved for sorting record identifiers during list prefetch processing. The RID pool is created at start up time, but no space is allocated until RID storage is needed. When RID storage is needed, it is allocated above the 16MB line in 16KB blocks known as RID blocks.

The RID pool is used for all record identifier (RID) processing. It is used for sorting RIDs during the following operations:

- List prefetch, including single index list prefetch,
- Access via multiple indexes
- Hybrid joins

RID pool storage is also used when DB2 enforces unique keys while updating multiple rows.

SQL statements that use those methods of access can benefit from using the RID pool. RID pool processing can help reduce I/O resource consumption and elapsed time.

When the DSNTINST CLIST calculates the value for RID POOL SIZE on panel DSNTIPC, the default is calculated as 50% of the sum of virtual buffer pools BP0, BP1, BP2, and BP32K. Depending upon your specification for these buffer pools, the RID pool might not be sufficiently large for your workload. Alternatively, you might have overridden the default calculated value to a smaller value. In either case, the RID pool storage can become exceeded.

CPExpert compares the QISTRMAX variable in DB2STATS with the **QISTRMAX** guidance variable in USOURCE(DB2GUIDE). The QISTRMAX value contains a count of the number of times maximum amount RID pool storage was exceeded.

©Copyright 1999, Computer Management Sciences, Inc.

CPExpert produces Rule DB2-184 when the value of the QISTRMAX variable in DB2STATS exceeds the value specified by the **QISTRMAX** guidance variable.

The default value for the **QISTRMAX** guidance variable is 0, indicating that CPExpert should produce Rule DB2-184 whenever the maximum amount of RID pool storage was exceeded.

The following example illustrates the output from Rule DB2-184:

```
RULE DB2-184: THE MAXIMUM RID POOL STORAGE WAS EXCEEDED

   The maximum RID pool storage was exceeded during the intervals shown
   below.  You should consider increasing the amount of storage allocated
   to the RID Pool.  Alternatively, you might consider curtailing the work
   being processed concurrently.

                                    TIMES MAXIMUM RID POOL
   MEASUREMENT INTERVAL             STORAGE WAS EXHAUSTED
   10:21-10:51, 15SEP1999                        5
   10:51-11:21, 15SEP1999                        2
   12:45-13:15, 16SEP1999                        4
   14:44-15:14, 16SEP1999                        2
   15:14-15:44, 16SEP1999                        3
```

Suggestion: You should consider the following alternatives:

- Increase the amount of storage allocated to the RID Pool by updating the RID POOL SIZE field on installation panel DSNTIPC.
- Curtail the work processed by the DB2 subsystem during the intervals shown.
- Alter CPExpert's analysis by modifying the **QISTRMAX** guidance variable in USOURCE(DB2GUIDE).

Reference: DB2 for OS/390 Version 3: Administration Guide
Section 2.4.1.32 (Install DB2--CLIST Calculations Panel 1: DSNTIPC)
Section 7.5.3 (Increasing RID Pool Size)
Section 7.9.7.2 (Bind Time and Execution Time Thresholds)

DB2 for OS/390 Version 4: Installation Guide
Section 2.2.4 (Virtual Storage for Storage Pools and Working Storage)
Section 2.5.1.36 (Install DB2--CLIST Calculations Panel 1: DSNTIPC)

DB2 for OS/390 Version 4: Administration Guide
Section 5.10.5.2 (List Sequential Prefetch)

DB2 for OS/390 Version 5: Installation Guide
Section 2.2.4 (Virtual Storage for Storage Pools and Working Storage)
Section 2.5.32 (Install DB2--CLIST Calculations Panel 1: DSNTIPC)

DB2 for OS/390 Version 5: Administration Guide
Section 5.10.5.2.3 (Bind Time and Execution Time Thresholds)

DB2 for OS/390 Version 6: Installation Guide
Section 2.2.3 (Virtual Storage for Storage Pools and Working Storage)
Section 2.5.32 (Install DB2--CLIST Calculations Panel 1: DSNTIPC)

DB2 for OS/390 Version 6: Administration Guide
Section 5.10.5.2.3 (Bind Time and Execution Time Thresholds)

©Copyright 1999, Computer Management Sciences, Inc.

Rule DB2-201: There might be insufficient buffers for work files

Finding: The total number of buffers required by concurrent work files was more than had been allocated. Consequently, DB2 might not have been able to perform efficient sequential prefetch during sort merge passes.

Impact: This finding can have a MEDIUM IMPACT, or HIGH IMPACT on the performance of the DB2 subsystem.

Discussion: DB2's sort process begins when ordered sets of rows (called initial runs) are written to work files. At the completion of the runs, when all the rows have been sorted and inserted into the work files, the work files are merged together into one work file containing the sorted data.

The work files used in sort are logical work files, which reside in temporary work file table spaces in database DSNDB07. DB2 uses the buffer pool when writing to the logical work file. The number of work files that can be used for sorting is limited only by the buffer pool size when you have the sort assist hardware.

Temporary work file table spaces can be associated with different buffer pools. However, for best performance during sorts, all 4KB work file table spaces should be associated with a single 4KB buffer pool.

IBM suggests that, for efficient sort performance, each work file should have 16 buffers available. This is so DB2 can perform sequential prefetch operations for the work files during merge processing. The total number of buffers required is 16 times the number of concurrent work files.

CPExpert computes the average number of buffers available for merge processing by multiplying QBSTWFM variable in DB2STATB (the maximum number of work files concurrently used during sort merge processing) by 16 (QBSTWFM * 16). This result is compared with the QBSTVPL variable in DB2STATB (the number of buffers allocated for the virtual buffer pool). CPExpert produces Rule DB2-201 when there exists less than 16 buffers available for the sort merge processing.

The following example illustrates the output from Rule DB2-201:

©Copyright 1999, Computer Management Sciences, Inc.

```
RULE DB2-201: THERE MIGHT BE INSUFFICIENT BUFFERS FOR WORK FILES

   Buffer Pool 0: Ideally, each work file should have 16 buffers available
   so DB2 can perform sequential prefetch operations for the work files
   during merge processing.  The total number of buffers required is 16
   times the number of concurrent work files.  For the intervals shown
   below, the total number of buffers required by concurrent work files
   was more than were allocated.  Consequently, DB2 might not have been
   able to perform sequential prefetch. This situation occurred for Buffer
   Pool 0 during the intervals shown below:

                                  BUFFERS       CONCURRENT
   MEASUREMENT INTERVAL          ALLOCATED      WORK FILES
   21:32-21:32, 08SEP1998          2,000            132
```

Suggestion: The referenced sections of the DB2 Administrative Guide offers numerous suggestions on how to improve sorting. You should review the suggestions to select those actions that are applicable to your environment.

You can alter CPExpert's analysis only by "turning off" this rule, either turning it off globally for all buffers, or turning the rule off just for the particular buffer pool. Please refer to Section 3 for a description of how to turn off rules.

Reference: DB2 for OS/390 Version 3: Administration Guide
Section 7.5.4 (Controlling Sort Pool Size and Sort Processing)
DB2 for OS/390 Version 3: DSNWSMGS macro (QBSTWFM variable)

DB2 for OS/390 Version 4: Administration Guide
Section 5.4.4 (Controlling Sort Pool Size and Sort Processing)
DB2 for OS/390 Version 4: DSNWSMGS macro (QBSTWFM variable)

DB2 for OS/390 Version 5: Administration Guide
Section 5.4.4 (Controlling Sort Pool Size and Sort Processing)
DB2 for OS/390 Version 5: DSNWSMGS macro (QBSTWFM variable)

DB2 for OS/390 Version 6: Administration Guide
Section 5.4.4 (Controlling Sort Pool Size and Sort Processing)
DB2 for OS/390 Version 6: DSNWSMGS macro (QBSTWFM variable)

Rule DB2-202: There were insufficient buffers for work files in merge passes

Finding: The total number of buffers required by concurrent work files was more than had been allocated. Consequently, DB2 was not able to perform efficient sequential prefetch during sort merge passes.

Impact: This finding can have a MEDIUM IMPACT, or HIGH IMPACT on the performance of the DB2 subsystem.

Discussion: DB2's sort process begins when ordered sets of rows (called initial runs) are written to work files. At the completion of the runs, when all the rows have been sorted and inserted into the work files, the work files are merged together into one work file containing the sorted data.

The work files used in sort are logical work files, which reside in temporary work file table spaces in database DSNDB07. DB2 uses the buffer pool when writing to the logical work file. The number of work files that can be used for sorting is limited only by the buffer pool size when you have the sort assist hardware.

Temporary work file table spaces can be associated with different buffer pools. However, for best performance during sorts, all 4KB work file table spaces should be associated with a single 4KB buffer pool.

IBM suggests that, for efficient sort performance, each work file should have 16 buffers available. This is so DB2 can perform sequential prefetch operations for the work files during merge processing. The total number of buffers required is 16 times the number of concurrent work files.

CPExpert computes the average number of buffers available for merge processing by (1) computing the average number of work files per merge pass (QBSTWFT/QBSTWFR), and (2) multiplying the average by 16. This result is compared with the QBSTVPL variable in DB2STATB (the number of buffers allocated for the virtual buffer pool).

CPExpert produces Rule DB2-202 when the average number of buffers available for merge processing is less than 16.

The following example illustrates the output from Rule DB2-202:

©Copyright 1999, Computer Management Sciences, Inc.

```
RULE DB2-202: INSUFFICIENT BUFFERS FOR WORK FILES IN MERGE PASSES

   Buffer Pool 2: CPExpert used the total number of work files requested
   for all merge passes and the number of merge passes, to compute the
   average number of work files that were required for the average merge
   pass.  Ideally, each work file should have 16 buffers available so DB2
   can perform sequential prefetch operations for the work files during
   merge processing.  The average number of buffers required is 16 times
   the average number of concurrent work files.  For the intervals shown
   below, the average number of buffers required by concurrent work files
   was more than were allocated.  Consequently, DB2 was not always able
   to perform sequential prefetch.  This situation occurred for Buffer
   Pool 2 during the intervals shown below:

                              BUFFERS       TOTAL     MERGE     AVERAGE
   MEASUREMENT INTERVAL      ALLOCATED   WORK FILES   PASSES   WORK FILES
   14:27-14:57, 15SEP1999      2,000         260         2         130
   17:22-17:52, 16SEP1999      2,000         130         1         130
```

Suggestion: The referenced sections of the DB2 Administrative Guide offers numerous suggestions on how to improve sorting. You should review the suggestions to select those actions that are applicable to your environment.

You can alter CPExpert's analysis only by "turning off" this rule, either turning it off globally for all buffers, or turning the rule off just for the particular buffer pool. Please refer to Section 3 for a description of how to turn off rules.

Reference: DB2 for OS/390 Version 3: Administration Guide
Section 7.5.4 (Controlling Sort Pool Size and Sort Processing)
DB2 for OS/390 Version 3: DSNWSMGS macro (QBSTWFM variable)

DB2 for OS/390 Version 4: Administration Guide
Section 5.4.4 (Controlling Sort Pool Size and Sort Processing)
DB2 for OS/390 Version 4: DSNWSMGS macro (QBSTWFM variable)

DB2 for OS/390 Version 5: Administration Guide
Section 5.4.4 (Controlling Sort Pool Size and Sort Processing)
DB2 for OS/390 Version 5: DSNWSMGS macro (QBSTWFM variable)

DB2 for OS/390 Version 6: Administration Guide
Section 5.4.4 (Controlling Sort Pool Size and Sort Processing)
DB2 for OS/390 Version 6: DSNWSMGS macro (QBSTWFM variable)

Rule DB2-203: Merge passes were not efficiently performed

Finding: Merge passes were not efficiently performed; work files were rejected because of insufficient buffers.

Impact: This finding can have a MEDIUM IMPACT, or HIGH IMPACT on the performance of the DB2 subsystem.

Discussion: DB2's sort process begins when ordered sets of rows (called initial runs) are written to work files. At the completion of the runs, when all the rows have been sorted and inserted into the work files, the work files are merged together into one work file containing the sorted data.

The work files used in sort are logical work files, which reside in temporary work file table spaces in database DSNDB07. DB2 uses the buffer pool when writing to the logical work file. The number of work files that can be used for sorting is limited only by the buffer pool size when you have the sort assist hardware.

Temporary work file table spaces can be associated with different buffer pools. However, for best performance during sorts, all 4KB work file table spaces should be associated with a single 4KB buffer pool.

IBM suggests that, for efficient sort performance, each work file should have 16 buffers available. This is so DB2 can perform sequential prefetch operations for the work files during merge processing. The total number of buffers required is 16 times the number of concurrent work files.

The QBSTWFF variable in the DB2 buffer statistics contains a count of the number of times that the merge pass was not efficiently performed because of a shortage of buffer space. This counter is incremented for each merge pass when the maximum number of work files allowed is less than the number of work files requested. The maximum number of work files allowed is computed as:

*Buffers consumed = 2 * (work files already allocated)*

*Buffers available = 0.8 * (bufferpool size - buffers consumed)*

$$\textit{Maximum work files allowed} = \frac{\textit{buffers available}}{16}$$

CPExpert compares the QBSTWFF variable in DB2STATB with the **QBSTWFF** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-203 when the times that the merge pass was not efficiently performed because of a shortage of buffer space exceeds the value specified by the **QBSTWFF** guidance variable.

Suggestion: The referenced sections of the DB2 Administrative Guide offers numerous suggestions on how to improve sorting. You should review the suggestions to select those actions that are applicable to your environment.

In the DSNWSMGS macro comments, the primary suggestion is to increase the number of buffers assigned to the buffer pool.

You can alter CPExpert's analysis only by "turning off" this rule, either turning it off globally for all buffers, or turning the rule off just for the particular buffer pool. Please refer to Section 3 for a description of how to turn off rules.

Reference: DB2 for OS/390 Version 3: Administration Guide
Section 7.5.4 (Controlling Sort Pool Size and Sort Processing)
DB2 for OS/390 Version 3: DSNWSMGS macro (QBSTWFM variable)

DB2 for OS/390 Version 4: Administration Guide
Section 5.4.4 (Controlling Sort Pool Size and Sort Processing)
DB2 for OS/390 Version 4: DSNWSMGS macro (QBSTWFM variable)

DB2 for OS/390 Version 5: Administration Guide
Section 5.4.4 (Controlling Sort Pool Size and Sort Processing)
DB2 for OS/390 Version 5: DSNWSMGS macro (QBSTWFM variable)

DB2 for OS/390 Version 6: Administration Guide
Section 5.4.4 (Controlling Sort Pool Size and Sort Processing)
DB2 for OS/390 Version 6: DSNWSMGS macro (QBSTWFM variable)

Rule DB2-206: Hiperpool read requests failed (pages stolen by system)

Finding: Read requests from a hiperpool failed because the backing expanded storage page was stolen by MVS.

Impact: This finding can have a MEDIUM IMPACT, or HIGH IMPACT on the performance of the DB2 subsystem.

Discussion: A *hiperpool* can be used to extend DB2's virtual buffer pools. A hiperpool resides in an MVS *hiperspace*, which is a storage space of up to 2GB that a program can use as a data buffer.

On systems that have the prerequisite hardware and software, DB2 maintains two levels of storage for each buffer pool:

- The first level of storage is the virtual buffer pool. The virtual buffer pool is allocated from DB2's ssnmDBM1 address space. A virtual buffer pool is backed by central storage, expanded storage, or auxiliary storage. The sum of all DB2 virtual buffer pools cannot exceed 1.6GB.

- The second level of storage is the hiperpool. Hiperpools are optional. The hiperpool uses the MVS/ESA hiperspace facility to utilize expanded storage only (ESO) hiperspace. More than one hiperpool can be defined (each hiperpool is associated with a virtual buffer pool). However, the sum of all hiperpools cannot exceed 8GB.

  Hiperpools can be created only if the system meets the following requirements:

  - A processor that supports the MVPG hardware instruction and has the Asynchronous Data Mover Facility (ADMF) licensed internal code installed.

  - Expanded storage available in ES/9000 hardware.

  - MVS/ESA Version 4 Release 3 (or later) with the Special Programming Enhancement containing support for Asynchronous Data Mover Facility.

Virtual buffer pools hold the most frequently accessed data, while hiperpools serve as a cache for data that is accessed less frequently. When a row of data is needed from a page in a hiperpool, the entire page is read into the corresponding virtual buffer pool. Because DASD read

©Copyright 1999, Computer Management Sciences, Inc.

operations are not required for accessing data that resides in hiperspace, response time is shorter than for DASD retrieval. Retrieving pages cached in hiperpools takes only microseconds, rather than the milliseconds needed for retrieving a page from DASD.

If the row is changed, the page is not written back to the hiperpool until it has been written to DASD. All read and write operations to data in the page, and all DASD I/O operations, take place in the virtual buffer pool. The hiperpool holds only pages that have been read into the virtual buffer pool and might have been discarded; they are kept in case they are needed again.

An unsuccessful read occurs when a requested page was found in the hiperpool, but its content was discarded by MVS. Data in hiperspace can be discarded by MVS utilizing one of two processes:

- For hiperpools defined as CASTOUT=YES, the written data can be discarded by MVS if the hiperpool usage was low or if the expanded memory was not large enough to back the hiperpool. This would be the more common process by which data would be discarded.

- For hiperpools defined as CASTOUT=NO, an unsuccessful read can only happen when the backing expanded storage page was explicitly reconfigured out of the system.

DB2 can request a page from a hiperpool using either the Move Page instruction or the Asynchronous Data Mover Facility (ADMF).

- The MVPG instruction moves a single page at a time from the hiperpool to the virtual buffer pool. The MVPG instruction requires less initialization overhead than ADMF and is therefore more cost affective for a small number of pages.

- ADMF is a hardware facility that uses the I/O processor to move large numbers of pages between the virtual buffer pool and hiperpool. ADMF is efficient in moving a large number of pages, whereas MVPG is efficient in moving a small number of pages.

The number of pages that determines whether to use ADMF or MVPG is processor dependent. If the number of pages is greater than a processor dependent cutoff number, ADMF is used. Otherwise, MVPG is used.

As mentioned earlier, DASD read operations are not required for accessing data that resides in hiperspace, and the time to access the data is significantly shorter. On the other hand, unnecessary overhead and delay occurs if DB2 attempts to reference a page in a hiperpool and MVS has

©Copyright 1999, Computer Management Sciences, Inc. Revised: October, 2000

discarded the page. If the page has been discarded, DB2 must incur the overhead of a missed page read, and then fetch the page from DASD.

The QBSTWRF variable in DB2STATB contains a count of the number of times a hiperpool read request failed because the backing expanded storage page was stolen by MVS. This count does not include pages moved by the ADMF[1].

CPExpert compares the QBSTHRF with the **QBSTHRF** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-206 when the number of failed hiperpool read requests exceeds the value specified by the **QBSTHRF** guidance variable, and the hiperpool had the CASTOUT=YES specification.

The default value for the **QBSTHRF** guidance variable is 0, indicating that CPExpert should produce Rule DB2-206 when any failed hiperpool read requests occur.

The following example illustrates the output from Rule DB2-206:

```
RULE DB2-206: HIPERPOOL READ REQUESTS FAILED (PAGES STOLEN BY SYSTEM)

   Buffer Pool 4: Synchronous or asynchronous read requests failed
   because the backing expanded storage page was stolen by the system.
   An unsuccessful read occurs when a requested page was found in the
   hiperpool, but its content was discarded by MVS.  For hiperpools
   defined as CASTOUT=YES, the written data can be discarded by MVS if
   the hiperpool usage was low or if the expanded memory was not large
   enough to back the hiperpool.  For hiperpools defined as CASTOUT=NO,
   an unsuccessful read can only happen when the backing expanded storage
   page was explicitly reconfigured out of the system. If this number is
   high for a hiperpool with CASTOUT=YES specified, you should consider
   reducing the size of the hiperpool.  This situation occurred for Buffer
   Pool 4 during the intervals shown below:

                                  BUFFERS      HIPERPOOL     FAILED READ
   MEASUREMENT INTERVAL          ALLOCATED      BUFFERS       REQUESTS
   10:03-10:33, 29AUG2000          5,000         10,000        122,328
```

Suggestion: If CPExpert regularly produces Rule DB2-206, you should consider the following alternatives:

- You should consider reducing the size of the hiperpool. Unnecessary overhead is generated if the hiperpool is too large for the amount of available expanded storage. When DB2 believes that a page is in a hiperpool, but the pages have been discarded by MVS, DB2 must fetch

[1] Failed read requests (using the ADMF) are reported in Rule DB2-210.

the page from DASD. The overhead required to determine that the page was unavailable could be eliminated if the hiperpool were smaller (DB2 would not believe that the page was in the hiperpool and would directly fetch the page from DASD).

- You should consider rescheduling the work on the system to reduce the demand on expanded storage during the intervals when DB2 has a high demand for hiperpool pages.

- You should consider altering the Expanded Storage Control Table (ESCT) variables in the IEAOPTxx member of SYS1.PARMLIB. These variables could cause MVS to send pages for other workload directly to DASD, depending upon the activity of expanded storage[2].

- You should consider acquiring additional expanded storage. Additional expanded storage would provide additional resources for other work in the system, and could allow the hiperpool to function adequately without MVS discarding pages.

- You can alter CPExpert's analysis by modifying the **QBSTHRF** guidance variable in USOURCE(DB2GUIDE).

Reference: OS/390 Initialization and Tuning Guide
Adjusting Constants Options (Expanded Storage Control)

OS/390 Initialization and Tuning Reference
IEAOPTxx (OPT Parameters)

"Expanded Storage Management with MVS/ESA" by Deese, Donald R., available at **www.cpexpert.com**.

DB2 for OS/390 Version 3: Installation Guide
Section 7.5.1.1 (Buffer Pools and Hiperpools)

DB2 for OS/390 Version 4: Administration Guide
Section 5.4.1.1 (Buffer Pools and Hiperpools)

DB2 for OS/390 Version 5: Administration Guide
Section 5.4.1.1 (Buffer Pools and Hiperpools)

DB2 for OS/390 Version 6: Administration Guide
Section 5.4.1.1.1 (Buffer Pools and Hiperpools)

[2] Please carefully read the referenced paper "Expanded Storage Controls with MVS/ESA" before taking action with the ESCTxxx variables. Not all ESCTxxx variables have the effects that are described in IBM's *Initialization and Tuning Guide* and *Initialization and Tuning References*.

Rule DB2-207: Hiperpool write requests failed (expanded storage not available)

Finding: Write requests to a hiperpool failed because the backing expanded storage page was unavailable.

Impact: This finding can have a MEDIUM IMPACT, or HIGH IMPACT on the performance of the DB2 subsystem.

Discussion: A *hiperpool* can be used to extend DB2's virtual buffer pools. A hiperpool resides in an MVS *hiperspace*, which is a storage space of up to 2GB that a program can use as a data buffer.

On systems that have the prerequisite hardware and software, DB2 maintains two levels of storage for each buffer pool:

- The first level of storage is the virtual buffer pool. The virtual buffer pool is allocated from DB2's ssnmDBM1 address space. A virtual buffer pool is backed by central storage, expanded storage, or auxiliary storage. The sum of all DB2 virtual buffer pools cannot exceed 1.6GB.

- The second level of storage is the hiperpool. Hiperpools are optional. The hiperpool uses the MVS/ESA hiperspace facility to utilize expanded storage only (ESO) hiperspace. More than one hiperpool can be defined (each hiperpool is associated with a virtual buffer pool). However, the sum of all hiperpools cannot exceed 8GB.

  Hiperpools can be created only if the system meets the following requirements:

  - A processor that supports the MVPG hardware instruction and has the Asynchronous Data Mover Facility (ADMF) licensed internal code installed.

  - Expanded storage available in ES/9000 hardware.

  - MVS/ESA Version 4 Release 3 (or later) with the Special Programming Enhancement containing support for Asynchronous Data Mover Facility.

Virtual buffer pools hold the most frequently accessed data, while hiperpools serve as a cache for data that is accessed less frequently. When a row of data is needed from a page in a hiperpool, the entire page

©Copyright 1999, Computer Management Sciences, Inc.

is read into the corresponding virtual buffer pool. Because DASD read operations are not required for accessing data that resides in hiperspace, response time is shorter than for DASD retrieval. Retrieving pages cached in hiperpools takes only microseconds, rather than the milliseconds needed for retrieving a page from DASD.

If the row is changed, the page is not written back to the hiperpool until it has been written to DASD. All read and write operations to data in the page, and all DASD I/O operations, take place in the virtual buffer pool. The hiperpool holds only pages that have been read into the virtual buffer pool and might have been discarded; they are kept in case they are needed again.

DB2 can write a page to a hiperpool either using the Move Page Facility or using the Asynchronous Data Mover Facility (ADMF).

- The MVPG instruction moves a single page at a time from the hiperpool to the virtual buffer pool. The MVPG instruction requires less initialization overhead than ADMF and is therefore more cost affective for a small number of pages.

- ADMF is a hardware facility that uses the I/O processor to move large numbers of pages between the virtual buffer pool and hiperpool. ADMF is efficient in moving a large number of pages, whereas MVPG is efficient in moving a small number of pages.

The number of pages that determines whether to use ADMF or MVPG is processor dependent. If the number of pages is greater than a processor dependent cutoff number, ADMF is used. Otherwise, MVPG is used.

When DB2 writes the page to hiperspace in expanded storage, sufficient expanded storage must be available. If expanded storage is not available, the write fails.

The QBSTHWF variable in DB2STATB contains a count of the number of times a hiperpool write request failed because there was insufficient expanded storage page to satisfy the expanded storage operation. This count does not include pages moved by the ADMF[1].

CPExpert compares the QBSTHWF variable with the **QBSTHWF** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-207 when the number of failed hiperpool write requests exceeds the value specified by the **QBSTHWF** guidance variable.

[1]Failed write requests (using the ADMF) are reported in Rule DB2-211.

The default value for the **QBSTHWF** guidance variable is 0, indicating that CPExpert should produce Rule DB2-207 when any failed hiperpool write requests occur because expanded storage was unavailable.

The following example illustrates the output from Rule DB2-207:

```
RULE DB2-207: HIPERPOOL SHOULD BE REDUCED (INSUFFICIENT EXP STORAGE)

   Buffer Pool 4: A synchronous or asynchronous write request failed
   because expanded storage page could be allocated by MVS. If this occurs
   often, you should consider reducing the size of the hiperpool.  This
   situation occurred for Buffer Pool 4 during the intervals shown below:

                                 BUFFERS      HIPERPOOL     FAILED WRITE
   MEASUREMENT INTERVAL         ALLOCATED      BUFFERS        REQUESTS
   10:03-10:33, 29AUG2000         5,000         10,000         38,380
```

Suggestion: If CPExpert regularly produces Rule DB2-207, you should consider the following alternatives:

- You should consider reducing the size of the hiperpool. Unnecessary overhead is generated if the hiperpool is too large for the amount of available expanded storage. The overhead required MVB to determine that the expanded storage was unavailable could be eliminated if the hiperpool were smaller (DB2 would detect that the hiperpool was full and would not attempt to write to expanded storage).

- You should consider rescheduling the work on the system to reduce the demand on expanded storage during the intervals when DB2 has a high demand for hiperpool pages.

- If your environment is operating under MVS **Compatibility Mode**, you should consider altering the Expanded Storage Control Table (ESCT) variables in the IEAOPTxx member of SYS1.PARMLIB. These variables could cause MVS to send pages for other workload directly to DASD, depending upon the activity of expanded storage[2].

- If your environment is operating under MVS **Goal Mode** with OS/390 Version 2 Release 10 (and with APAR OW43810 installed), you should consider specifying long-term storage protection. **Warning**: long-term storage protection could have serious negative effects on non-DB2

[2] Please carefully read the referenced paper "Expanded Storage Managements with MVS/ESA" before taking action with the ESCTxxx variables. Not all ESCTxxx variables have the effects that are described in IBM's *Initialization and Tuning Guide* and *Initialization and Tuning References*.

workload operating on the system. The implications of this option should be considered carefully.

- You should consider acquiring additional expanded storage. Additional expanded storage would provide additional resources for other work in the system, and could allow the hiperpool to function adequately.

- You can alter CPExpert's analysis by modifying the **QBSTHWF** guidance variable in USOURCE(DB2GUIDE).

Reference: OS/390 Initialization and Tuning Guide
Adjusting Constants Options (Expanded Storage Control)

OS/390 Initialization and Tuning Reference
IEAOPTxx (OPT Parameters)

"Expanded Storage Management with MVS/ESA" by Deese, Donald R., available at **www.cpexpert.com**.

DB2 for OS/390 Version 3: Installation Guide
Section 7.5.1.1 (Buffer Pools and Hiperpools)

DB2 for OS/390 Version 4: Administration Guide
Section 5.4.1.1 (Buffer Pools and Hiperpools)

DB2 for OS/390 Version 5: Administration Guide
Section 5.4.1.1 (Buffer Pools and Hiperpools)

MVS Planning: Workload Management
OS/390 V2R10 Section 12 (Defining Special Protection Options for Critical Work)

Rule DB2-208: Virtual buffer pool was full

Finding: A usable buffer could not be located in a virtual buffer Pool because the virtual buffer pool was full.

Impact: This finding can have a HIGH IMPACT on the performance of the DB2 subsystem.

Discussion: Buffer pools are areas of virtual storage used temporarily to store pages of table spaces or indexes. When an application program needs to access a row of a table, DB2 retrieves the page containing that row and places the page in a buffer. If the row is changed, the buffer must be written back to the table space. If the needed data is already in a buffer, the application program will not have to wait for it to be retrieved from DASD.

On systems that have the prerequisite hardware and software, DB2 maintains two levels of storage for each buffer pool[1]:

- The first level of storage is the virtual buffer pool. The virtual buffer pool is allocated from DB2's ssnmDBM1 address space. A virtual buffer pool is backed by central storage, expanded storage, or auxiliary storage. The sum of all DB2 virtual buffer pools cannot exceed 1.6GB.

- The second level of storage is the hiperpool. Hiperpools are optional. The hiperpool uses the MVS/ESA hiperspace facility to utilize expanded storage only (ESO) hiperspace. More than one hiperpool can be defined (each hiperpool is associated with a virtual buffer pool). However, the sum of all hiperpools cannot exceed 8GB.

Virtual buffer pools hold the most frequently accessed data, while hiperpools serve as a cache for data that is accessed less frequently. When a row of data is needed from a page in a hiperpool, the entire page is read into the corresponding virtual buffer pool.

Initially, the sizes (in number of pages) of virtual buffer pools and hiperpools on installation panels are established using the DSNTIP1 and DSNTIP2 panels. The sizes of virtual buffer pools and hiperpools can be modified using the ALTER BUFFERPOOL command.

At any moment, a virtual buffer pool can have three types of pages:

[1] Another level of buffer pools is the group buffer pool, associated with the DB2 data sharing environment.

©Copyright 1999, Computer Management Sciences, Inc.

- In-use Pages: These are pages that are currently being read or updated. The data they contain is available for use by other applications.

- Updated Pages: These are pages whose data has been changed but have not yet been written to DASD. After the updated page has been written to DASD, it remains in the virtual buffer pool available for migration to the corresponding hiperpool. In this case, the page is not considered to be "updated" until it is changed again.

- Available pages: These pages can be considered for new use, to be overwritten by an incoming page of new data. Both in-use pages and updated pages are unavailable in this sense; they are not considered for new use.

DB2 will attempt to keep some pages “available” for use by GETPAGE operations. There are three fixed thresholds that are used to control the availability of pages in an individual buffer pool: (1) the Sequential Prefetch Threshold (SPTH), (2) the Data Management Threshold (DMTH), and (3) the Immediate Write threshold. Each threshold is a level of use which, when exceeded, causes DB2 to take some action.

- The SPTH threshold is a fixed value of 90%. DB2 checks the SPTH before scheduling a prefetch operation. If the SPTH threshold has been exceeded, the prefetch is not scheduled. DB2 also checks the SPTH during buffer allocation for an already-scheduled prefetch operation. DB2 cancels the prefetch if the threshold has been exceeded.

- The Data Management Threshold (DMTH) has a fixed value of 95%, This threshold is checked before a page is read or updated. If the threshold has been exceeded, DB2 accesses the page in the virtual buffer pool once for each row that is retrieved or updated in that page. In other words, retrieving or updating several rows in one page causes several page access operations.

- The Immediate Write threshold (IMTH) has a fixed value of 97.5%. This threshold is checked whenever a page is to be updated. If the Immediate Write threshold has been exceeded, DB2 considers the buffer pool to be in serious danger of being “full” and the updated page is written to DASD as soon as the update completes. The write is synchronous with the SQL request; that is, the request waits until the write has been completed and the two operations are not carried out concurrently.

Under normal circumstances, the thresholds will be sufficient to maintain an available supply of buffers in the virtual buffer pool. However, if the buffer pool is too small (or there are too many active threads updating

pages), the thresholds might not allow DB2 to maintain an available supply of buffers. When this occurs, the virtual buffer pool is “full” and DB2 cannot read additional pages into the pool. Performance can be seriously degraded when a buffer pool is full.

CPExpert compares the QBSTXFL variable in DB2STATB (the number of times a virtual buffer pool was full) with the **QBSTXFL** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-208 when the number of times the virtual buffer pool was full exceeds the value specified by the **QBSTXFL** guidance variable.

The default value for the **QBSTXFL** guidance variable is 0, indicating that CPExpert should produce Rule DB2-208 whenever a virtual buffer pool was full.

The following example illustrates the output from Rule DB2-208:

```
RULE DB2-208: VIRTUAL BUFFER POOL WAS FULL

   Buffer Pool 2: A usable buffer could not be located in virtual Buffer
   Pool 2, because the virtual buffer pool was full.  This condition
   should not normally occur, as there should be ample buffers.  You
   should consider using the -ALTER BUFERPOOL command to increase the
   virtual buffer pool size (VPSIZE) for the virtual buffer pool.  This
   situation occurred during the intervals shown below:

                                 BUFFERS       NUMBER OF TIMES
   MEASUREMENT INTERVAL         ALLOCATED       POOL WAS FULL
   10:24-10:54, 15SEP1999           100                4
   10:54-11:24, 15SEP1999           100               12
   11:24-11:54, 15SEP1999           100               13
   11:54-12:24, 15SEP1999           100               16
   12:24-12:54, 15SEP1999           100                2
   12:54-13:24, 15SEP1999           100                3
```

Suggestion: Under normal circumstances, you should not encounter a virtual buffer pool condition, since the DB2 thresholds will allow DB2 to keep a supply of available pages. You should consider the following alternatives if Rule DB2-208 is produced:

- Use the ALTER BUFFERPOOL command to increase the size of the buffer pool. This is the easiest alternative to implement. Of course, you might not have sufficient real or virtual storage to implement this alternative.

- Expand the size of the data sets assigned to the buffer pool, and examine the DASD performance (that is, make sure that there are not contention problems at the DASD volume, path, or controller level). The basic problem is a lack of available buffers. Since the DB2 fixed

thresholds are unable to free buffers fast enough to keep up with thread updating pages, I/O contention problems might have prevented the buffers being written to DASD sufficiently fast. The DASD Component of CPExpert can aid in this analysis.

- Consider reassigning selected datasets from this buffer pool to another buffer pool. Reducing the number of active datasets would spread the update activity among buffer pools.

- Consider curtailing the number of threads that reference the buffer pool. This would entail reducing the MAX USERS or MAX REMOTE USERS values. This alternative should be considered only in unusual situations, since the result would be queuing of thread create (and would cause Rule DB2-101 to be produced).

- Alternatively, you can alter CPExpert's analysis by modifying the **QBSTXFL** guidance variable in USOURCE(DB2GUIDE). You normally should not alter the default value of 0 for this guidance variable, since a buffer full condition is quite serious.

Reference: DB2 for OS/390 Version 3: Administration Guide
Section 7.5.1 (Tuning Database Buffer Pools)

DB2 for OS/390 Version 4: Administration Guide
Section 5.4.1 (Tuning Database Buffer Pools)

DB2 for OS/390 Version 5: Administration Guide
Section 5.4.1 (Tuning Database Buffer Pools)

DB2 for OS/390 Version 6: Administration Guide
Section 5.4.1 (Tuning Database Buffer Pools)

Rule DB2-209: Active buffers account for large percent of buffer pool

Finding: The DB2 statistics information showed that the active buffers accounted for a large percent of the buffer pool.

Impact: This finding can have a LOW IMPACT on the performance of the DB2 subsystem. The finding is presented to report trend information.

Discussion: Buffer pools are areas of virtual storage used temporarily to store pages of table spaces or indexes. When an application program needs to access a row of a table, DB2 retrieves the page containing that row and places the page in a buffer. If the row is changed, the buffer must be written back to the table space. If the needed data is already in a buffer, the application program will not have to wait for it to be retrieved from DASD.

At any moment, a database virtual buffer pool can have three types of pages:

- In-use Pages: These are pages that are currently being read or updated. The data they contain is available for use by other applications. In-use pages are considered *active buffers*, and are in a “non-stealable” status (that is, the buffers will not be used by DB2 for new pages).

- Updated Pages: These are pages whose data has been changed but have not yet been written to DASD. After the updated page has been written to DASD, it remains in the virtual buffer pool available for migration to the corresponding hiperpool. In this case, the page is not considered to be "updated" until it is changed again.

- Available pages: These pages can be considered for new use, to be overwritten by an incoming page of new data. Both in-use pages and updated pages are unavailable in this sense; they are not considered for new use.

DB2 will attempt to keep some pages “available” for use by GETPAGE operations. There are three fixed thresholds that are used to control the availability of pages in an individual buffer pool: (1) the Sequential Prefetch Threshold (SPTH), (2) the Data Management Threshold (DMTH), and (3) the Immediate Write threshold. Each threshold is a “level of use” which, when exceeded, causes DB2 to take some action[1]. The “level of use” is

[1] Rule DB2-208 provides some discussion of these thresholds, and DB2's actions when a threshold is reached.

simply the percent of total buffers that are represented by in-use or updated pages. These are considered “active” buffers.

The DB2 statistics report the number of active buffers in a buffer pool, as a “snapshot” that is taken when the statistics are written.

CPExpert computes the percent of buffers that are active by dividing the QBSTVPL variable (the number of buffers) into the QBSTCBA (the number of active buffers). The resulting percent is compared with the **PCTCBA** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-209 when the percent of active buffers exceeds the value specified by the **PCTCBA** guidance variable.

The default value for the **PCTCBA** guidance variable is 50, indicating that CPExpert should produce Rule DB2-209 when more than 50% of the buffers in the buffer pool are active.

The following example illustrates the output from Rule DB2-209:

```
RULE DB2-209: ACTIVE BUFFERS ACCOUNT FOR LARGE PERCENT OF BUFFER POOL

   Buffer Pool 0: The DB2 statistics contain a snapshot of the status of
   buffers when the statistics are written to SMF.  If the number of
   active buffers is too large a percent of the total buffers, it is
   possible that the number of buffers is too small.  During the intervals
   shown below, the percent of active buffers was greater than 50% of the
   buffers allocated.  This situation occurred for Buffer Pool 0 during
   the intervals shown below:

                                  BUFFERS    ACTIVE BUFFERS    PERCENT OF
   MEASUREMENT INTERVAL          ALLOCATED    AT SMF WRITE     ALLOCATED
   10:24-10:54, 15SEP1999            400            350           87.5
   10:54-11:24, 15SEP1999            400            350           87.5
   11:24-11:54, 15SEP1999            400            350           87.5
```

Suggestion: The buffer pool might be too small if most of the buffer pool contains active pages. If most of the buffer pool is used by active buffers, it might be an indication that a “buffer full” condition could be pending.

However, the information reported in the DB2 statistics is only a “snapshot” of the status of the buffer pool when the statistics were written. Consequently, the number of active buffers can be considered only an indication of the use of the buffer pool. The active buffers value should be considered a “sample” of the status of the buffer pool. The values should be tracked over time to see whether they consistently are a large percent of the total buffers assigned to the buffer pool.

Rule DB2-208 would be produced if a buffer full condition existed. If Rule DB2-208 is not produced (and Rule DB2-209 is produced) you should look for any increasing trend in the values reported by Rule DB2-209. If the trend is increasing, then you should review the alternatives listed with Rule DB2-208, since a buffer full condition might be encountered if no action is taken.

You should consider producing a SAS plot or graph of the value of the QBSTCBA variable to more easily discover whether a trend exists.

You can alter CPExpert's analysis by modifying the **PCTCBA** guidance variable in USOURCE(DB2GUIDE).

Reference: DB2 for OS/390 Version 3: Administration Guide
Section 7.5.1 (Tuning Database Buffer Pools)

DB2 for OS/390 Version 4: Administration Guide
Section 5.4.1 (Tuning Database Buffer Pools)

DB2 for OS/390 Version 5: Administration Guide
Section 5.4.1 (Tuning Database Buffer Pools)

DB2 for OS/390 Version 6: Administration Guide
Section 5.4.1 (Tuning Database Buffer Pools)

Rule DB2-210: Hiperpool read requests (using ADMF) failed

Finding: Read requests from a hiperpool (using the Asynchronous Data Mover Facility) failed because the backing expanded storage page was stolen by MVS.

Impact: This finding can have a MEDIUM IMPACT, or HIGH IMPACT on the performance of the DB2 subsystem.

Discussion: A *hiperpool* can be used to extend DB2's virtual buffer pools. A hiperpool resides in an MVS *hiperspace*, which is a storage space of up to 2GB that a program can use as a data buffer.

On systems that have the prerequisite hardware and software, DB2 maintains two levels of storage for each buffer pool:

- The first level of storage is the virtual buffer pool. The virtual buffer pool is allocated from DB2's ssnmDBM1 address space. A virtual buffer pool is backed by central storage, expanded storage, or auxiliary storage. The sum of all DB2 virtual buffer pools cannot exceed 1.6GB.

- The second level of storage is the hiperpool. Hiperpools are optional. The hiperpool uses the MVS/ESA hiperspace facility to utilize expanded storage only (ESO) hiperspace. More than one hiperpool can be defined (each hiperpool is associated with a virtual buffer pool). However, the sum of all hiperpools cannot exceed 8GB.

Hiperpools can be created only if the system meets the following requirements:

- A processor that supports the MVPG hardware instruction and has the Asynchronous Data Mover Facility (ADMF) licensed internal code installed.

- Expanded storage available in ES/9000 hardware.

- MVS/ESA Version 4 Release 3 (or later) with the Special Programming Enhancement containing support for Asynchronous Data Mover Facility.

Virtual buffer pools hold the most frequently accessed data, while hiperpools serve as a cache for data that is accessed less frequently.

©Copyright 1999, Computer Management Sciences, Inc.

When a row of data is needed from a page in a hiperpool, the entire page is read into the corresponding virtual buffer pool. Because DASD read operations are not required for accessing data that resides in hiperspace, response time is shorter than for DASD retrieval. Retrieving pages cached in hiperpools takes only microseconds, rather than the milliseconds needed for retrieving a page from DASD.

If the row is changed, the page is not written back to the hiperpool until it has been written to DASD. All read and write operations to data in the page, and all DASD I/O operations, take place in the virtual buffer pool. The hiperpool holds only pages that have been read into the virtual buffer pool and might have been discarded; they are kept in case they are needed again.

An unsuccessful read occurs when DB2 believed that a requested page was in the hiperpool, but its content was discarded by MVS. Data in hiperspace can be discarded by MVS utilizing one of two processes:

- For hiperpools defined as CASTOUT=YES, the written data can be discarded by MVS if the hiperpool usage was low or if the expanded memory was not large enough to back the hiperpool. This would be the more common process by which data would be discarded.

- For hiperpools defined as CASTOUT=NO, an unsuccessful read can only happen when the backing expanded storage page was explicitly reconfigured out of the system.

DB2 can request a page from a hiperpool using either the Move Page instruction or the Asynchronous Data Mover Facility (ADMF).

- The MVPG instruction moves a single page at a time from the hiperpool to the virtual buffer pool. The MVPG instruction requires less initialization overhead than ADMF and is therefore more cost affective for a small number of pages.

- ADMF is a hardware facility that uses the I/O processor to move large numbers of pages between the virtual buffer pool and hiperpool. ADMF is efficient in moving a large number of pages, whereas MVPG is efficient in moving a small number of pages.

The number of pages that determines whether to use ADMF or MVPG is processor dependent. If the number of pages is greater than a processor dependent cutoff number, ADMF is used. Otherwise, MVPG is used.

©Copyright 1999, Computer Management Sciences, Inc.

As mentioned earlier, DASD read operations are not required for accessing data that resides in hiperspace, and the time to access the data is significantly shorter. On the other hand, unnecessary overhead and delay occurs if DB2 attempts to reference a page in a hiperpool and MVS has discarded the page. If the page has been discarded, DB2 must incur the overhead of a missed page read, and then fetch the page from DASD.

The QBSTARF variable in DB2STATB contains a count of the number of times a hiperpool read request failed because the backing expanded storage page was stolen by MVS. This count includes only pages moved by the ADMF[1].

CPExpert compares the QBSTARF variable with the **QBSTARF** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-210 when the number of failed hiperpool read requests exceeds the value specified by the **QBSTARF** guidance variable, and the hiperpool had the CASTOUT=YES specification.

The default value for the **QBSTARF** guidance variable is 0, indicating that CPExpert should produce Rule DB2-210 when any failed hiperpool read requests occur.

The following example illustrates the output from Rule DB2-210:

```
RULE DB2-210: READ REQUEST (USING ADMF) FAILED

   Buffer Pool 4: Asynchronous read requests using the Asynchronous Data
   Mover Facility (ADMF) failed because the backing expanded storage page
   was stolen by MVS.  For hiperpools defined as CASTOUT=YES, the written
   data can be discarded by MVS if the hiperpool usage was low or if the
   expanded memory was not large enough to back the hiperpool.  For
   hiperpools defined as CASTOUT=NO, an unsuccessful read can only happen
   when the backing expanded storage page was explicitly reconfigured out
   of the system. If this number is high for a hiperpool with CASTOUT=YES
   specified, you should consider reducing the size of the hiperpool.
   This situation occurred for Buffer Pool 4 during the intervals shown

                                 BUFFERS      HIPERPOOL     FAILED READ
   MEASUREMENT INTERVAL         ALLOCATED      BUFFERS        REQUESTS
   10:03-10:33, 29AUG2000         5,000         10,000          78,695
```

Suggestion: If CPExpert regularly produces Rule DB2-210, you should consider the following alternatives:

- You should consider reducing the size of the hiperpool. Unnecessary overhead is generated if the hiperpool is too large for the amount of

[1]Failed read requests (using the Move Page Facility) are reported in Rule DB2-206.

available expanded storage. When DB2 believes that a page is in a hiperpool, but the pages have been discarded by MVS, DB2 must fetch the page from DASD. The overhead required to determine that the page was unavailable could be eliminated if the hiperpool was smaller (DB2 would not believe that the page was in the hiperpool and would directly fetch the page from DASD).

- You should consider rescheduling the work on the system to reduce the demand on expanded storage during the intervals when DB2 has a high demand for hiperpool pages.

- If you are in MVS Compatibility Mode, you should consider specifying storage isolation for the DB2 address space. While storage isolation applies mainly to central (real) storage, MVS will honor storage isolation for all processor storage (central and expanded storage). Thus, MVS will not normally migrate pages from expanded storage if the resulting processor storage would be less than the "low" value of the PWSS parameter. If you are in MVS Goal Mode, the MVS Workload Manager might implement both central and expanded storage isolation, based on its algorithms and assessment of whether goals were missed because of page stealing from either central or expanded storage.

- You should consider altering the Expanded Storage Control Table (ESCT) variables in the IEAOPTxx member of SYS1.PARMLIB. These variables could cause MVS to send pages for other workload directly to DASD, depending upon the activity of expanded storage[2].

- You should consider acquiring additional expanded storage. Additional expanded storage would provide additional resources for other work in the system, and could allow the hiperpool to function adequately without MVS discarding pages.

- You can alter CPExpert's analysis by modifying the **QBSTARF** guidance variable in USOURCE(DB2GUIDE).

Reference: OS/390 Initialization and Tuning Guide
Adjusting Constants Options (Expanded Storage Control)

OS/390 Initialization and Tuning Reference
IEAOPTxx (OPT Parameters)

[2] Please carefully read the referenced paper "Expanded Storage Management with MVS/ESA" before changing the ESCTxxx variables. Not all ESCTxxx variables have the effects that are described in IBM's *Initialization and Tuning Guide* and *Initialization and Tuning References*.

"Expanded Storage Management with MVS/ESA" by Deese, Donald R., available at **www.cpexpert.com**.

DB2 for OS/390 Version 3: Administration Guide
Section 7.5.1.1 (Buffer Pools and Hiperpools)

DB2 for OS/390 Version 4: Administration Guide
Section 5.4.1.1 (Buffer Pools and Hiperpools)

DB2 for OS/390 Version 5: Administration Guide
Section 5.4.1.1 (Buffer Pools and Hiperpools)

DB2 for OS/390 Version 6: Administration Guide
Section 5.4.1.1.1 (Buffer Pools and Hiperpools)

Rule DB2-211: Hiperpool write requests failed (expanded storage not available)

Finding: Write requests to a hiperpool failed because the backing expanded storage page was unavailable.

Impact: This finding can have a MEDIUM IMPACT, or HIGH IMPACT on the performance of the DB2 subsystem.

Discussion: A *hiperpool* can be used to extend DB2's virtual buffer pools. A hiperpool resides in an MVS *hiperspace*, which is a storage space of up to 2GB that a program can use as a data buffer.

On systems that have the prerequisite hardware and software, DB2 maintains two levels of storage for each buffer pool:

- The first level of storage is the virtual buffer pool. The virtual buffer pool is allocated from DB2's ssnmDBM1 address space. A virtual buffer pool is backed by central storage, expanded storage, or auxiliary storage. The sum of all DB2 virtual buffer pools cannot exceed 1.6GB.

- The second level of storage is the hiperpool. Hiperpools are optional. The hiperpool uses the MVS/ESA hiperspace facility to utilize expanded storage only (ESO) hiperspace. More than one hiperpool can be defined (each hiperpool is associated with a virtual buffer pool). However, the sum of all hiperpools cannot exceed 8GB.

Hiperpools can be created only if the system meets the following requirements:

- A processor that supports the MVPG hardware instruction and has the Asynchronous Data Mover Facility (ADMF) licensed internal code installed.

- Expanded storage available in ES/9000 hardware.

- MVS/ESA Version 4 Release 3 (or later) with the Special Programming Enhancement containing support for Asynchronous Data Mover Facility.

Virtual buffer pools hold the most frequently accessed data, while hiperpools serve as a cache for data that is accessed less frequently. When a row of data is needed from a page in a hiperpool, the entire page is read into the corresponding virtual buffer pool. Because DASD read

©Copyright 1999, Computer Management Sciences, Inc.

operations are not required for accessing data that resides in hiperspace, response time is shorter than for DASD retrieval. Retrieving pages cached in hiperpools takes only microseconds, rather than the milliseconds needed for retrieving a page from DASD.

If the row is changed, the page is not written back to the hiperpool until it has been written to DASD. All read and write operations to data in the page, and all DASD I/O operations, take place in the virtual buffer pool. The hiperpool holds only pages that have been read into the virtual buffer pool and might have been discarded; they are kept in the hiperpool in case they are needed again.

DB2 can write a page to a hiperpool either using the Move Page Facility or using the Asynchronous Data Mover Facility (ADMF).

- The MVPG instruction moves a single page at a time from the hiperpool to the virtual buffer pool. The MVPG instruction requires less initialization overhead than ADMF and is therefore more cost affective for a small number of pages.

- ADMF is a hardware facility that uses the I/O processor to move large numbers of pages between the virtual buffer pool and hiperpool. ADMF is efficient in moving a large number of pages, whereas MVPG is efficient in moving a small number of pages.

The number of pages that determines whether to use ADMF or MVPG is processor dependent. If the number of pages is greater than a processor dependent cutoff number, ADMF is used. Otherwise, MVPG is used.

When DB2 writes the page to hiperspace in expanded storage, sufficient expanded storage must be available. If expanded storage is not available, the write fails.

The QBSTAWF variable in DB2STATB contains a count of the number of times a hiperpool write request failed because there was insufficient expanded storage page to satisfy the expanded storage operation. This count includes only pages moved by the ADMF[1].

CPExpert compares the QBSTAWF variable with the **QBSTAWF** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-211 when the number of failed hiperpool write requests exceeds the value specified by the **QBSTAWF** guidance variable.

[1]Failed write requests (using the Move Page Facility) are reported in Rule DB2-207.

The default value for the **QBSTAWF** guidance variable is 0, indicating that CPExpert should produce Rule DB2-211 when any failed hiperpool write requests occur because expanded storage was unavailable.

The following example illustrates the output from Rule DB2-211:

```
RULE DB2-211: WRITE REQUESTS (USING ADMF) FAILED

   Buffer Pool 4: Asynchronous write requests using the Asynchronous Data
   Mover Facility (ADMF) failed because no backing expanded storage page
   could be allocated. If this occurs often, you should consider reducing
   the size of the hiperpool.  This situation occurred for Buffer Pool 4
   during the intervals shown below:

                                  BUFFERS      HIPERPOOL     FAILED WRITE
   MEASUREMENT INTERVAL          ALLOCATED      BUFFERS        REQUESTS
   10:03-10:33, 29AUG2000          5,000         10,000           6,72
```

Suggestion: If CPExpert regularly produces Rule DB2-211, you should consider the following alternatives:

- You should consider reducing the size of the hiperpool. Unnecessary overhead is generated if the hiperpool is too large for the amount of available expanded storage. The overhead required for MVS to determine that the expanded storage was unavailable could be eliminated if the hiperpool was smaller (DB2 would detect that the hiperpool was full and would not attempt to write to expanded storage).

- You should consider rescheduling the work on the system to reduce the demand on expanded storage during the intervals when DB2 has a high demand for hiperpool pages.

- You should consider altering the Expanded Storage Control Table (ESCT) variables in the IEAOPTxx member of SYS1.PARMLIB. These variables could cause MVS to send pages for other workload directly to DASD, depending upon the activity of expanded storage[2].

- You should consider acquiring additional expanded storage. Additional expanded storage would provide additional resources for other work in the system, and could allow the hiperpool to function adequately.

- You can alter CPExpert's analysis by modifying the **QBSTAWF** guidance variable in USOURCE(DB2GUIDE).

[2] Please carefully read the referenced paper "Expanded Storage Management with MVS/ESA" before taking action with the ESCTxxx variables. Not all ESCTxxx variables have the effects that are described in IBM's *Initialization and Tuning Guide* and *Initialization and Tuning References*.

Reference: OS/390 Initialization and Tuning Guide
Adjusting Constants Options (Expanded Storage Control)

OS/390 Initialization and Tuning Reference
IEAOPTxx (OPT Parameters)

"Expanded Storage Management with MVS/ESA" by Deese, Donald R., available at **www.cpexpert.com**.

DB2 for OS/390 Version 3: Administration Guide
Section 7.5.1.1 (Buffer Pools and Hiperpools)

DB2 for OS/390 Version 4: Administration Guide
Section 5.4.1.1 (Buffer Pools and Hiperpools)

DB2 for OS/390 Version 5: Administration Guide
Section 5.4.1.1 (Buffer Pools and Hiperpools)

DB2 for OS/390 Version 6: Administration Guide
Section 5.4.1.1.1 (Buffer Pools and Hiperpools)

©Copyright 1999, Computer Management Sciences, Inc.

Rule DB2-214: Pages per write I/O operation was low

Finding: The average number of pages written per asynchronous write was relatively high, indicating that pages will be written in “spikes” of activity. This finding means that the DWQT and VDWQT thresholds might be too large.

Impact: This finding can have a LOW IMPACT or MEDIUM IMPACT on the performance of the DB2 subsystem.

Discussion: Buffer pool write operations are usually performed concurrently with user requests. Updated pages are queued by data set until they are written when:

- A DB2 *checkpoint* is taken
- The percentage of unavailable pages in a virtual buffer pool exceeds a preset limit called the Deferred Write Threshold (DWQT).
- The percentage of updated pages in a virtual buffer pool for a single data set exceeds a preset limit called the Vertical Deferred Write Threshold (VDWQT).

Up to 32 4KB or 4 32KB pages can be written in a single I/O operation.

Two variable thresholds control the asynchronous write of pages from buffer pools: (1) the Deferred Write Threshold and (2) the Vertical Deferred Write Threshold.

- The Deferred Write Threshold is a percentage of the virtual buffer pool that might be occupied by unavailable pages, including both updated pages and pages in use. The default value for the Deferred Write Threshold is 50%. DB2 checks this threshold when an update to a page is completed. If the percentage of unavailable pages in the virtual buffer pool exceeds the threshold, write operations are scheduled for enough data sets (at up to 128 pages per data set) to decrease the number of unavailable buffers to 10% below the threshold. For example, if the threshold is 50%, the number of unavailable buffers is reduced to 40%.
- The Vertical Deferred Write Threshold is expressed as a percentage of the virtual buffer pool that might be occupied by updated pages from a **single data set**. The default value for the Deferred Write Threshold is 10%. DB2 checks this threshold whenever an update to a page is

completed. If the percentage of updated pages for the data set exceeds the threshold, writes are scheduled for that data set.

If a virtual buffer pool is large enough, it is unlikely that the default values of either DWQT or VDWQT will ever be reached. In this case, surges of write I/Os tend to occur as deferred writes are triggered by DB2 *checkpoints*. Lowering the VDWQT and the DWQT could improve performance by distributing the write I/Os more evenly over time.

Additionally, specifying the default or specifying a larger DWQT or VDWQT value for a small buffer pool could

CPExpert computes the average pages written per asynchronous write by dividing QBSTWIO (asynchronous writes) into QBSTPWS (number of pages written)[1]. The resulting average pages written per asynchronous write activity is compared against the **QBSTWIO** guidance variable in USOURCE(DB2GUIDE).

Additionally, CPExpert examines the QBSTWFR variable (the number of merge passes) in the DB2STATB statistics. When the value of this variable exceeds the **QBSTWFR** guidance variable in USOURCE(DB2GUIDE), CPExpert concludes that the buffer pool is heavily used for sort activity. CPExpert suppresses Rule DB2-214 for buffer pools that are heavily used for sort activity. Please refer to Rule DB2-226 for further information.

Additionally, CPExpert examines the QDBPVDQT variable (the value of the VDWQT parameter) in DB2STAT2. CPExpert suppresses Rule DB2-214 if this value is zero, as DB2 will initiate write I/O whenever 32 buffers are used for a single data set for updated pages.

CPExpert produces Rule DB2-214 when the average pages written per asynchronous write activity exceeds the value specified by the **QBSTWIO** guidance variable, and the other conditions of this rule are met.

The default value for the **QBSTWIO** guidance variable is 20, indicating that CPExpert should produce Rule DB2-214 when more than 20 pages were written per asynchronous write activity .

The following example illustrates the output from Rule DB2-214:

[1] CPExpert excludes intervals with less than 500 asynchronous writes, to eliminate spurious findings.

```
RULE DB2-214: DWQT AND VDWQT MIGHT BE TOO LARGE

   Buffer Pool 0: The average number of pages written per asynchronous
   write was relatively high.  This means that buffer write operations
   will tend to occur with "spikes" of pages written, since a relatively
   large number of pages will be written at once.  You should consider
   reducing the DWQT and VDWQT values to reduce the "spikes" of pages
   written. The DWQT threshold was specified as 50% and the VDWQT threshold
   was specified as 10%.  This situation occurred for Buffer Pool 0 during
   the intervals shown below:

                                 BUFFERS     PAGES      ASYNC   AVG PAGES
   MEASUREMENT INTERVAL         ALLOCATED   WRITTEN    WRITES   PER WRITE
   21:32-21:32, 08SEP1998         2,000      79,829    21,339       24
```

Suggestion: If Rule DB2-214 is produced regularly, you should consider the following alternatives:

- You can lower the DQWT and VDWQT thresholds, by using the ALTER BUFFERPOOL command. Changing a threshold in one virtual buffer pool or hiperpool has no effect on any other virtual buffer pool or hiperpool.

  Of the two thresholds, it is much more advantageous to lower the VDWQT threshold. You should consider setting the VDWQT threshold to zero. When VDWQT is set to zero, DB2 will initiate write I/O whenever 32 buffers are used for a single data set for updated pages.

- You can alter CPExpert's analysis by modifying the **QBSTWIO** guidance variable in USOURCE(DB2GUIDE).

Reference: DB2 for OS/390 Version 3: Administration Guide
Section 7.5.1 (Tuning Database Buffer Pools)

DB2 for OS/390 Version 4: Administration Guide
Section 5.4.1 (Tuning Database Buffer Pools)

DB2 for OS/390 Version 5: Administration Guide
Section 5.4.1 (Tuning Database Buffer Pools)

DB2 for OS/390 Version 6: Administration Guide
Section 5.4.1.1.1 (Buffer Pools and Hiperpools)

Rule DB2-215: Total DB2 buffer pool page fault rate was high

Finding: The total page fault rate for read and write I/O was high for DB2 buffer pools.

Impact: This finding can have a MEDIUM IMPACT, or HIGH IMPACT on the performance of the DB2 subsystem and of the MVS system.

Discussion: DB2 buffer pools are allocated and managed by DB2. Pages in the buffer pools normally reside in real (central) storage. However, MVS might need pages of central storage and steal the buffer pool pages.

If buffer pool pages are stolen from central storage, MVS will send the pages to expanded storage, or send them to auxiliary storage on DASD[1]. Subsequent access to these pages results in a page fault and MVS must initiate I/O to bring the data into central storage. There is little delay if the page fault is resolved from expanded storage, as these page faults are resolved in microseconds. The delay can be more serious if the page faults are resolved from auxiliary storage, as these delays are in milliseconds.

Paging of buffer pool storage can cause poor DB2 performance. Additionally, a high page fault rate can cause performance problems for the entire MVS system. In general, a high page fault rate indicates that there is a problem with DB2's use of buffer pools, a problem with the amount of central storage available, or a problem with the amount of expanded storage available.

CPExpert sums the following variables in DB2STATS: the number of page-ins required for read I/O (QB1TRPI, QB2TRPI, QB3TRPI, QB4TRPI) and the number of page-ins required for write I/O (QB1TWPI, QB2TWPI, QB3TWPI, QB4TWPI). This yields the total page-is required for all DB2 buffer pools. The sum is divided by the DB2 statistics interval to yield a page-in rate per second. CPExpert then compares this page-in rate with the **BUFFPGRT** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-215 when the total page fault rate for all DB2 buffer pools exceeds the value specified by the **BUFFPGRT** guidance variable.

The default value for the **BUFFPGRT** guidance variable is 50, indicating that CPExpert should produce Rule DB2-215 whenever the total page

[1] Please see the referenced papers at www.cpexpert.com

©Copyright 1999, Computer Management Sciences, Inc.

faults for read I/O and write I/O for all DB2 buffer pools exceeds 50 page-ins per second.

The following example illustrates the output from Rule DB2-215:

```
RULE DB2-215: BUFFER POOL PAGE FAULT RATE WAS HIGH

   The page fault rate for DB2 read and write I/O was larger than 50 pages
   per second. This situation occurred during the intervals shown below:

                              TOTAL         TOTAL         TOTAL
                             BUFFERS     PAGE-IN FOR   PAGE-IN FOR  PAGE
   MEASUREMENT INTERVAL     ALLOCATED      READ I/O     WRITE I/O   RATE
   10:45-11:15, 16SEP1999    96,000        115,492       5,016      67.2
   11:15-11:45, 16SEP1999    96,000        103,871       4,314      60.3
   11:45-12:15, 16SEP1999    96,000        121,040       6,421      71.1
   12:15-12:45, 16SEP1999    96,000        118,052       6,877      69.7
   12:45-13:15, 16SEP1999    96,000        101,197       4,045      58.7
```

Suggestion: The basic problem is that the amount of central (real) storage is insufficient for the demands of the workload (demands from DB2 and demands from other work executing concurrently). As a result, MVS is stealing pages from DB2 buffer pools.

If Rule DB2-215 is produced more than occasionally, you should consider the following alternatives:

- Please note that an acceptable page fault rate generally is dependent on the overall workload executing on the system and on the importance of the DB2 work. **You should consider specifying an acceptable BUFFPGRT guidance value for specific buffer pools**.

- Reduce the size of buffer pools so that they will not require as much central (real) storage. This will cause DB2 to manage the pages in the buffer pool, rather than allowing the MVS page fault process to manage the pages. DB2 normally will more efficiently manage pages in the buffer pool than the MVS page fault process.

- If you are operating in MVS Compatibility Mode, verify whether storage isolation is specified for any performance group periods **other** than DB2. This specification will be reflected in the IEAIPSxx member of SYS1.PARMLIB (see the PWSS keyword for other performance group periods).

  If storage isolation is specified for any performance periods **other** than DB2, you should either remove the storage isolation specification of the other performance group periods, or specify storage isolation for the DB2

address spaces and the IRLM. You can use the PWSS keyword to specify storage isolation for the DB2 performance group and the IRLM.

- If you are operating in MVS Goal Mode, verify that the Goal Importance for DB2 (and that of allied agents) is sufficiently high (compared with other service class periods) that the Workload Manager will adequately manage processor (central and expanded) storage for DB2.

- If you are operating in MVS Goal Mode with OS/390 Version 2 Release 10, verify that Protective Processor Protection has not been specified for other service classes operating on the system.

- If you have the necessary hardware, consider using hiperpools so that movement of the pages in the buffer pool between central storage and expanded storage will be managed by DB2 rather than by MVS. DB2 normally manages pages in virtual pools and hiperpools better than the normal page fault resolution process in MVS. This normal page fault resolution consumes CPU resource. By using hiperpools, page movement between expanded storage and central storage is performed by the ADMF or the MVPG instruction . Using hiperpools provides two advantages.

  - The CPU resource used for MVS paging is reduced, and the CPU savings are available to service other useful requests, benefiting overall system performance and increasing system throughput.

  - ADMF is much more efficient than MVS paging in moving pages between expanded storage and central storage. The larger the buffer pool, the more opportunity for reducing MVS paging by allocating a majority of the buffer pool requirements in expanded storage using hiperpool.

- If you are already using a hiperpool for this buffer pool, consider increasing the size of the hiperpool[2].

- Review the Expanded Storage Control Table (ESCT) parameters in the IEAOPTxx member of SYS1.PARMLIB. These parameters guide MVS in sending stolen pages to expanded storage or auxiliary storage. With proper settings, you might be able to guide the system in sending stolen pages to expanded storage rather than to auxiliary storage. Page fault resolution from expanded storage is significantly faster than page fault resolution from auxiliary storage.

[2] However, depending on the workload executing on the system and the amount of expanded storage on the system, CPExpert could begin producing Rule DB2-206 or Rule DB2-207 if the hiperpool is increased. These rules would indicated that DB2 is experiencing problems with the availability of expanded storage.

- Consider using the CASTOUT=NO option for the buffer pool if you have established hiperpools. This alternative should be considered **only** if the DB2 system is extremely important from a management objectives view. Specifying CASTOUT=NO should not **normally** be specified.

- You can alter CPExpert's analysis by modifying the **BUFFPGRT** guidance variable in USOURCE(DB2GUIDE).

Reference: DB2 for OS/390 Version 3: Administration Guide
Section 7.5.1.1 (Buffer Pools and Hiperpools)

DB2 for OS/390 Version 4: Administration Guide
Section 5.4.1.1 (Buffer Pools and Hiperpools)

DB2 for OS/390 Version 5: Administration Guide
Section 5.4.1.1 (Buffer Pools and Hiperpools)

DB2 for OS/390 Version 6: Administration Guide
Section 5.4.1.1.1 (Buffer Pools and Hiperpools)

"Central Storage Management with MVS/ESA" by Deese, Donald R., available at **www.cpexpert.com**.

"Expanded Storage Management with MVS/ESA" by Deese, Donald R., available at **www.cpexpert.com**.

MVS Planning: Workload Management
OS/390 V2R10 Section 12 (Defining Special Protection Options for Critical Work)

Rule DB2-216: Page fault rate was high for individual buffer pool

Finding: The page fault rate for read and write I/O was high for an individual buffer pool.

Impact: This finding can have a MEDIUM IMPACT, or HIGH IMPACT on the performance of the DB2 subsystem and of the MVS system.

Discussion: DB2 buffer pools are allocated and managed by DB2. Pages in the buffer pools normally reside in real (central) storage. However, MVS might need pages of central storage and steal the buffer pool pages.

If buffer pool pages are stolen from central storage, MVS will send the pages to expanded storage, or send them to auxiliary storage on DASD[1]. Subsequent access to these pages results in a page fault and MVS must initiate I/O to bring the data into central storage. There is little delay if the page fault is resolved from expanded storage, as these page faults are resolved in microseconds. The delay can be more serious if the page faults are resolved from auxiliary storage, as these delays are in milliseconds.

Paging of buffer pool storage can cause poor DB2 performance. Additionally, a high page fault rate can cause performance problems for the entire MVS system. In general, a high page fault rate indicates that there is a problem with DB2's use of buffer pools, a problem with the amount of central storage available, or a problem with the amount of expanded storage available.

CPExpert sums the QBSTRPI (the number of page-ins required for read I/O) and QBSTWPI (the number of page-ins required for write I/O) in DB2STATB. The sum is divided by the DB2 statistics interval to yield a page-in rate per second. CPExpert then compares this page-in rate with the **PAGERATE** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-216 when the number of page-ins for read I/O and write I/O exceeds the value specified by the **PAGERATE** guidance variable.

The default value for the **PAGERATE** guidance variable is 25, indicating that CPExpert should produce Rule DB2-216 whenever page faults for read I/O and write I/O for a particular buffer pool exceed 25 page-ins per second.

[1] Please see the referenced papers at www.cpexpert.com

©Copyright 1999, Computer Management Sciences, Inc.

The following example illustrates the output from Rule DB2-216:

```
RULE DB2-216: PAGE FAULT RATE WAS HIGH FOR INDIVIDUAL BUFFER POOL

   Buffer Pool 1: The page fault rate for read and write I/O was larger
   than 25 pages per second. This situation occurred for Buffer Pool
   during the intervals shown below:

                                BUFFERS     PAGE-IN FOR   PAGE-IN FOR  PAGE
   MEASUREMENT INTERVAL        ALLOCATED     READ I/O     WRITE I/O   RATE
    8:46- 9:16, 16SEP1999      25,000         39,315          69      43.8
   10:45-11:15, 16SEP1999      25,000         25,707         226      28.8
   11:15-11:45, 16SEP1999      25,000         36,904         195      41.2
   11:45-12:15, 16SEP1999      25,000         30,892         563      35.0
   12:45-13:15, 16SEP1999      25,000         23,890         170      26.7
```

Suggestion: The basic problem is that the amount of central (real) storage is insufficient for the demands of the workload (demands from DB2 and demands from other work executing concurrently). As a result, MVS is stealing pages from DB2 buffer pools.

If Rule DB2-216 is produced more than occasionally, you should consider the following alternatives:

- Please note that an acceptable page fault rate generally is dependent on the overall workload executing on the system and on the importance of the DB2 work. **You should consider specifying an acceptable PAGERATE guidance value for specific buffer pools**.

- Reduce the size of buffer pools so that they will not require as much central (real) storage. This will cause DB2 to manage the pages in the buffer pool, rather than allowing the MVS page fault process to manage the pages. DB2 normally will more efficiently manage pages in the buffer pool than the MVS page fault process.

- If you are operating in MVS Compatibility Mode, verify whether storage isolation is specified for any performance group periods **other** than DB2. This specification will be reflected in the IEAIPSxx member of SYS1.PARMLIB (see the PWSS keyword for other performance group periods)..

  If storage isolation is specified for any performance periods **other** than DB2, you should either remove the storage isolation specification of the other performance group periods, or specify storage isolation for the DB2 address spaces and the IRLM. You can use the PWSS keyword to specify storage isolation for the DB2 performance group and the IRLM.

- If you are operating in MVS Goal Mode, verify that the Goal Importance for DB2 (and that of allied agents) is sufficiently high (compared with other service class periods) that the Workload Manager will adequately manage processor (central and expanded) storage for DB2.

- If you are operating in MVS Goal Mode with OS/390 Version 2 Release 10, verify that Protective Processor Protection has not been specified for other service classes operating on the system.

- If you have the necessary hardware, consider using hiperpools so that movement of the pages in the buffer pool between central storage and expanded storage will be managed by DB2 rather than by MVS. DB2 normally manages pages in virtual pools and hiperpools better than the normal page fault resolution process in MVS.

  If you are already using a hiperpool for this buffer pool, consider increasing the size of the hiperpool[2].

- Review the Expanded Storage Control Table (ESCT) parameters in the IEAOPTxx member of SYS1.PARMLIB. These parameters guide MVS in sending stolen pages to expanded storage or auxiliary storage. With proper settings, you might be able to guide the system in sending stolen pages to expanded storage rather than to auxiliary storage. Page fault resolution from expanded storage is significantly faster than page fault resolution from auxiliary storage.

- Consider using the CASTOUT=NO option for the buffer pool if you have established hiperpools. This alternative should be considered **only** if the DB2 system is extremely important from a management objectives view. Specifying CASTOUT=NO should not **normally** be specified.

- You can alter CPExpert's analysis by modifying the **PAGERATE** guidance variable in USOURCE(DB2GUIDE).

Reference: DB2 for OS/390 Version 3: Administration Guide
Section 7.5.1.1 (Buffer Pools and Hiperpools)

DB2 for OS/390 Version 4: Administration Guide
Section 5.4.1.1 (Buffer Pools and Hiperpools)

DB2 for OS/390 Version 5: Administration Guide
Section 5.4.1.1 (Buffer Pools and Hiperpools)

[2] However, depending on the workload executing on the system and the amount of expanded storage on the system, CPExpert could begin producing Rule DB2-206 or Rule DB2-207 if the hiperpool is increased. These rules would indicated that DB2 is experiencing problems with the availability of expanded storage.

©Copyright 1999, Computer Management Sciences, Inc.

DB2 for OS/390 Version 6: Administration Guide
Section 5.4.1.1.1 (Buffer Pools and Hiperpools)

"Central Storage Management with MVS/ESA" by Deese, Donald R., available at **www.cpexpert.com**.

"Expanded Storage Management with MVS/ESA" by Deese, Donald R., available at **www.cpexpert.com**.

"DB2 Buffer Pool Tuning - Top Down or Bottom Up", Joel Goldstein, *CMG98 Conference Proceedings*, pages 51-62.

MVS Planning: Workload Management
OS/390 V2R10 Section 12 (Defining Special Protection Options for Critical Work)

Rule DB2-220: Data management threshold (DMTH) was reached

Finding: The Data management threshold (DMTH) was reached.

Impact: This finding can have a MEDIUM IMPACT, or HIGH IMPACT on the performance of the DB2 subsystem.

Discussion: Buffer pools can contain three types of pages: (1) pages in use, (2) pages that have been updated but have not been written to DASD, and (3) pages that are available.

- Pages that are in-use are those pages that are currently being read, are being processed in some fashioned, or are being handled by DB2 (in the process of changing from in-use to updated, for example). The number of pages that are in-use can vary based on the number of threads concurrently accessing pages, the number of pages that are being accessed by each thread, and the number of datasets being accessed via the buffer pool

- Pages that have been updated are not immediately written to DASD. Rather, these pages are retained in the buffer pool awaiting the possibility that they will be referenced again. These updated pages are queued by data set until they are written when:

  - A DB2 *checkpoint* is taken

  - The percentage of unavailable pages in a virtual buffer pool exceeds a preset limit called the Deferred Write Threshold (DWQT).

  - The percentage of updated pages in a virtual buffer pool for a single data set exceeds a preset limit called the Vertical Deferred Write Threshold (VDWQT).

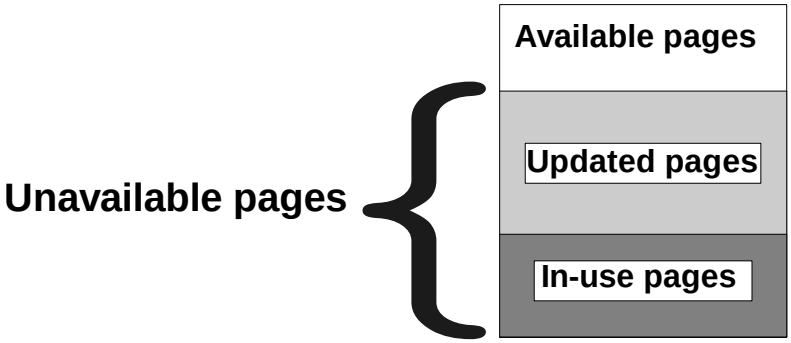
- Available pages in the buffer pool are the total number of pages in the buffer pool, minus the above two categories of pages. From one perspective, once DB2 has been operational for awhile, all pages will contain data since DB2 will retain data in buffer pools as long as possible (or read, via prefetch I/O, pages in the buffer pool). This is so unnecessary I/O can be avoided if a GETPAGE request could be satisfied from a previously-updated page or from a pre-fetch I/O page.

  Thus, the “available” pages could be viewed as “used” in the sense that they can hold pages that were read via pre-fetch I/O operations or they

could hold updated pages that have been written to DASD but have not yet been moved to a hiperpool (in case a hiperpool was defined for the virtual buffer pool).

Nonetheless, these categories of pages are considered to be available when DB2 needs to issue a GETPAGE. If the GETPAGE cannot be satisfied from a previously updated page or from a page that has been pre-fetched, DB2 will use one of the "available" pages to satisfy the GETPAGE via DASD (or via a read from a hiperpool).

The below figure shows a hypothetical distribution of pages in a buffer pool.



The relative number of pages in each category shown in the above figure can vary considerably, but DB2 will attempt to keep some number of pages "available" for use by GETPAGE operations.

There are three fixed thresholds that are used to control the availability of pages in an individual buffer pool: (1) the Sequential Prefetch Threshold (SPTH), (2) the Data Management Threshold (DMTH), and (3) the Immediate Write threshold.

- The SPTH threshold is a fixed value of 90%, and it is checked at two different times:

  - DB2 checks the SPTH before scheduling a prefetch operation. If the SPTH threshold has been exceeded, the prefetch is not scheduled. The SPTH value is 90%, indicating that DB2 will not schedule a prefetch operation if less than 10% of the buffer pool is available.

  - DB2 checks the SPTH during buffer allocation for an already-scheduled prefetch operation. DB2 cancels the prefetch if the threshold has been exceeded.

  Reaching the SPTH threshold has an adverse effect on DB2 performance since sequential prefetch of pages is inhibited until more buffers become available. While this can adversely effect performance,

it is not as serious as reaching the next two thresholds. CPExpert analyzes whether DB2 reached the SPTH threshold in Rule DB2-230.

- The Data Management Threshold (DMTH) has a fixed value of 95%, This threshold is checked before a page is read or updated. If the threshold has not been exceeded, DB2 accesses the page in the virtual buffer pool once for each page, no matter how many rows are retrieved or updated in that page.

  If the threshold has been exceeded, DB2 accesses the page in the virtual buffer pool once for each row that is retrieved or updated in that page. In other words, retrieving or updating several rows in one page causes several page access operations.

  Reaching this threshold has a significant effect on processor usage. This rule (Rule DB2-220) is used to check whether DB2 reaches the DMTH threshold.

- The Immediate Write threshold (IWTH) has a fixed value of 97.5%. This threshold is checked whenever a page is to be updated. If the Immediate Write threshold has been exceeded, the updated page is written to DASD as soon as the update completes. The write is synchronous with the SQL request; that is, the request waits until the write has been completed and the two operations are not carried out concurrently.

  Reaching this threshold has a significant effect on processor usage and I/O resource consumption. For example, updating three rows per page in 10 sequential pages ordinarily requires one or two write operations. When IWTH is exceeded, the updates require 30 (3 * 10 = 30) synchronous writes. CPExpert analyzes whether DB2 reached the IWTH threshold in Rule DB2-221.

Since reaching the fixed thresholds can have adverse effects on performance, DB2 provides several variable thresholds that can be used to control the distribution of page types in individual buffer pools. CPExpert analyzes whether DB2 reached these variable threshold, or whether the thresholds have been properly specified relative to each other, as described in other rules. This analysis is done in other rules; **this** rule analyzes whether the DMTH threshold was reached.

In analyzing whether DB2 reached the DMTH threshold, CPExpert compares the QBSTDMC variable in DB2STATB (the number of times the Data Management Threshold was reached) with the **QBSTDMC** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-220 when the number of times the DMTH was reached exceeds the value specified by the **QBSTDMC** guidance variable.

©Copyright 1999, Computer Management Sciences, Inc. Revised: October, 2000

The default value for the **QBSTDMC** guidance variable is 0, indicating that CPExpert should produce Rule DB2-220 whenever the Data Management Threshold was reached.

Suggestion: As mentioned above, reaching the DMTH threshold has a significant effect on processor usage. Consequently, you should not allow the DMTH threshold to be reached. If the Data Management Threshold was reached, you should consider the following alternatives:

- Review the value that was specified for the Deferred Write Queue (DWQT) threshold and the Vertical Deferred Write Queue (VDWQT) threshold. These variable thresholds control when DB2 decides to write to DASD the update pages that are retained in the buffer pool.

  If the DWQT threshold (or VDWQT threshold) have been set too high, DB2 might not begin writing the updated pages until the Data Management Threshold has been nearly reached.

  For example, if the DWQT threshold were set at 90% for a relatively small buffer pool, only a few GETPAGE operations would be necessary for the number of unavailable pages in the buffer pool to reach the 95% level, above which the DMTH threshold would be exceeded.

- Use the ALTER BUFFERPOOL command to increase the size of the virtual buffer pool of hiperpool.

- Modify application programs so that they COMMIT more frequently. This will free pages in the buffer pool.

- You may evaluate the workload being processed when the Data Management Threshold was reached, to determine whether non-DB2 workload or low-priority DB2 workload (such as batch updates) can be scheduled at other times.

- You can alter CPExpert's analysis by modifying the **QBSTDMC** guidance variable in USOURCE(DB2GUIDE).

Reference: DB2 for OS/390 Version 3: Administration Guide
Section 7.5.1.7 (Buffer Pool Thresholds)

DB2 for OS/390 Version 4: Administration Guide
Section 5.4.1.7 (Buffer Pool Thresholds)

DB2 for OS/390 Version 5: Administration Guide
Section 5.4.1.7 (Buffer Pool Thresholds)

DB2 for OS/390 Version 6: Administration Guide
Section 5.4.1.6 (Buffer Pool Thresholds)

Rule DB2-221: Immediate writes occurred

Finding: Immediate writes occurred, possibly because the Immediate Write Threshold (IWTH) was reached.

Impact: This finding can have a range of impact on performance, from NO IMPACT in the case that immediate writes were naturally occurring, to HIGH IMPACT in the case that immediate writes were because the Immediate Write Threshold (IWTH) was reached.

Discussion: DB2's use of a virtual buffer pool or hiperpool is governed by several preset values called thresholds. Each threshold is a level of use which, when exceeded, causes DB2 to take some action. The level of use is usually expressed as a percentage of the total size of the virtual buffer pool or hiperpool.

Some thresholds are fixed (i.e., they cannot be changed by users) and other thresholds are variable (they can be changed by users). **Please see Rule DB2-220 for a discussion of the fixed thresholds.**

The Immediate Write Threshold (IWTH) is one of the **fixed** thresholds. The IWTH has a fixed value of 97.5%, and is checked whenever a page is to be updated. If the number of unavailable pages in a virtual buffer pool exceeds 97.5% of the size of the virtual buffer pool, the updated page is written to DASD as soon as the update completes. The write is synchronous with the SQL request; that is, the request waits until the write has been completed and the two operations are not carried out concurrently.

Reaching the IWTH threshold has a significant effect on processor usage and I/O resource consumption. IBM documentation provides an example: updating three rows per page in 10 sequential pages ordinarily requires one or two write operations. When IWTH is exceeded, however, the updates require 30 synchronous writes.

CPExpert compares the QBSTIMW variable in DB2STATB (the number of immediate writes) with the **QBSTIMW** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-221 when the number of immediate writes exceeds the value specified by the **QBSTIMW** guidance variable **AND** Rule DB2-220 has been produced.

©Copyright 1999, Computer Management Sciences, Inc.

The default value for the **QBSTIMW** guidance variable is 0, indicating that CPExpert should produce Rule DB2-221 when any immediate writes occurred.

The following example illustrates the output from Rule DB2-221:

```
RULE DB2-221: IMMEDIATE WRITES OCCURRED

   Buffer Pool 0: Immediate writes occur when (1) any synchronous write
   is triggered, (2) an immediate write threshold is reached, or (3) no
   deferred write engines are available.  The immediate write is
   synchronous with the SQL request (the request waits until the write
   has been completed).  If the Immediate Write Threshold (IWTH) is
   reached, there is a significant effect on processor usage and I/O
   resource consumption.  There are situations when DB2 uses immediate
   write without there being a shortage of buffers.  Consequently, this
   finding does not automatically indicate a serious performance problem.
   Please note that Rule DB2-222 will be produced if no write engines were
   available.  This situation occurred for Buffer Pool 0 during the
   intervals shown below:

                                 BUFFERS     BUFFERS ACTIVE    IMMEDIATE
   MEASUREMENT INTERVAL         ALLOCATED   AT SMF WRITE TIME    WRITES
   21:32-21:32, 08SEP1998         2,000              0             121
```

Suggestion: As mentioned above, reaching the IWTH threshold has a significant effect on processor usage and I/O resource consumption. Consequently, you should not allow the IWTH threshold to be reached.

Unfortunately, it is not possible to determine whether the IWTH threshold was reached from the DB2 statistics. This is because DB2 uses synchronous (immediate) writes even when the IWTH is not exceeded.

- An immediate write might be issued when more than two checkpoints pass without a page being written. Two checkpoints passing without a page being written does not indicate a problem.

- An immediate write might be issued when DB2 detected that no deferred write engines were available. Rule DB2-234 analyzes the case where no write engine was available.

The Immediate Write Threshold is higher than another fixed threshold: the Data Management Threshold (DMTH). The DMTH is set at 95%, and the DMTH is checked before a page is read or updated. Rule DB2-220 checks whether the DMTH has been reached. As mentioned earlier, Rule DB2-221 is suppressed if Rule DB2-220 was **not** produced, as it would not be possible for the Immediate Write Threshold to be reached unless the Data Management Threshold was also reached.

CPExpert provides information regarding the number of buffers active at SMF write time, when Rule DB2-221 is produced. Sadly, this count is simply a “snapshot” of the buffer status when DB2 statistics information was placed in the SMF buffer. It is not an indication of the average or maximum values that occurred during the previous statistics interval.

However, If a low number of buffers regularly were active when CPExpert produced Rule DB2-221, you might conclude that the Immediate Write Threshold probably was **not** reached. Conversely, if a high number of buffers regularly were active when CPExpert produced Rule DB2-221, you might conclude that the Immediate Write Threshold **was** reached. This conclusion would be reasonably valid since Rule DB2-221 is produced only if Rule DB2-220 was produced (indicating that the Data Management Threshold had been reached).

If you conclude that the Immediate Write Threshold was reached, you should consider the following alternatives:

- Review the value that was specified for the Deferred Write Queue (DWQT) threshold and the Vertical Deferred Write Queue (VDWQT) threshold. These variable thresholds control when DB2 decides to write to DASD the update pages that are retained in the buffer pool.

  If the DWQT threshold (or VDWQT threshold) have been set too high, DB2 might not begin writing the updated pages until the Data Management Threshold has been nearly reached. With a small buffer pool, once the DMTH was reached, it is likely that the Immediate Write Threshold would be reached..

  For example, if the DWQT threshold were set at 90% for a relatively small buffer pool, only a few GETPAGE operations would be necessary for the number of unavailable pages in the buffer pool to reach the 95% level, above which the DMTH threshold would be exceeded. With a small buffer pool, only a few more GETPAGE requests would result in the IWTH threshold being reached.

- Use the ALTER BUFFERPOOL command to increase the size of the virtual buffer pool of hiperpool.

- Modify application programs so that they COMMIT more frequently. This will free pages in the buffer pool.

- You may evaluate the workload being processed when the Immediate Write Threshold was reached, to determine whether non-DB2 workload or low-priority DB2 workload (such as batch updates) can be scheduled at other times.

- You can alter CPExpert's analysis by modifying the **QBSTIMW** guidance variable in USOURCE(DB2GUIDE).

Reference: DB2 for OS/390 Version 3: Administration Guide
Section 7.5.1.7 (Buffer Pool Thresholds)

DB2 for OS/390 Version 4: Administration Guide
Section 5.4.1.7 (Buffer Pool Thresholds)

DB2 for OS/390 Version 5: Administration Guide
Section 5.4.1.7 (Buffer Pool Thresholds)

DB2 for OS/390 Version 6: Administration Guide
Section 5.4.1.6 (Buffer Pool Thresholds)

©Copyright 1999, Computer Management Sciences, Inc.

Rule DB2-222: Deferred write threshold (DWQT) was reached

Finding: The Deferred Write Threshold (DWQT) was reached.

Impact: This finding can have a LOW IMPACT or MEDIUM IMPACT on the performance of the DB2 subsystem.

Discussion: DB2's management of buffer pools in controlled by two types of thresholds: fixed thresholds and variable thresholds.

- There are three fixed thresholds that are used to control the availability of pages in an individual buffer pool: (1) the Sequential Prefetch Threshold (SPTH), (2) the Data Management Threshold (DMTH), and (3) the Immediate Write threshold. These fixed thresholds are discussed in Rule DB2-220.

- Since reaching the fixed thresholds can have adverse effects on performance, DB2 provides several variable thresholds that can be used to control the distribution of page types, or how the pages are used, in individual buffer pools.

  These variable thresholds are (1) the Sequential Steal Threshold (VPSEQT), (2) the Hiperpool Sequential Steal Threshold (HPSEQT), (3) the Virtual Buffer Pool Parallel Sequential Threshold (VPPSEQT), (4) Virtual Buffer Pool Assisting Parallel Sequential Threshold (VPXPSEQT), (5) the Deferred Write Threshold (DWQT), and (6) the Vertical Deferred Write Threshold (VDWQT).

  For the purposes of this rule, the DWQT and VDWQT thresholds are important to understand.

Rule DB2-220 illustrates that two categories of pages in the buffer pool are “unavailable” in the sense that they cannot be used by DB2 to acquire pages to satisfy GETPAGE requests or for sequential prefetch I/O. These categories are the “in-use” pages and the “updated but not written to DASD” pages. Collectively, these pages are in an “unavailable” category, as contrasted to the remaining pages, which are in an “available” category.

- The “in-use” page category cannot be used to satisfy GETPAGE requests or for sequential prefetch I/O, as these pages are being used by applications.

©Copyright 1999, Computer Management Sciences, Inc.

- DB2 retains pages of the “updated but not written to DASD” category expecting that there is some probability that the updated pages will be referenced again.

  This probability might be relatively high if the pages are referenced in a random pattern, and would be particularly high if the buffer pool size was a reasonable percent of the size of the data set being referenced (in the best case, the entire data set would be in the buffer pool and, after every page in the data set had been referenced once, there would be a 100% probability that a requested page would be found in the buffer pool). The probability would decrease as the size of the buffer pool became smaller relative to the size of the data set being accessed.

  By retaining the pages in the buffer pool, DB2 can avoid an I/O operation to DASD if a GETPAGE can be satisfied from the buffer pool.

  On the other hand, if a large percent of the GETPAGE requests are sequential, DB2 can implement sequential prefetch of pages. Sequential prefetch is a mechanism that triggers consecutive asynchronous I/O operations. Sequential prefetch brings pages into the virtual buffer pool before they are required and reads several pages with a single I/O operation. Consequently, sequential prefetch allows CPU processing and I/O operations to be overlapped.

  DB2 can implement sequential prefetch only if there are *available* pages in the buffer pool. If all (or most) pages in the buffer pool are occupied by in-use pages or updated but not written to DASD pages, DB2 would not be able to implement sequential prefetch. In this case, the space occupied by pages that were “updated but not written to DASD” would be wasted in the sense that these buffer pool pages otherwise could be used for prefetch of sequential I/O.

  Consequently, performance of applications sequentially processing data would suffer, and overall system performance would not be as efficient since CPU processing and I/O operations would not be overlapped.

Clearly, the best distribution of unavailable and available pages in a buffer pool depends on the data access characteristics of the applications using DB2. To accommodate a range of requirements depending on application differences and organizational priorities, DB2 provides two variable thresholds for each buffer pool: the **Deferred Write Queue Threshold (DWQT)** and the **Vertical Deferred Write Queue Threshold (VDWQT)**.

- The Deferred Write Threshold (DWQT) controls how much of the total buffer pool can be in the “unavailable” category.

DB2 checks the DWQT threshold when an update to a page is completed. If the percent of unavailable pages in the virtual buffer pool exceeds the DWQT threshold, write operations are scheduled for enough data sets (at up to 128 pages per data set) to decrease the number of unavailable buffers to 10% below the threshold.

For example, suppose 2000 buffers had been allocated to the buffer pool and the DWQT had been specified as 50% (the default specification). DB2 would detect that the DWQT threshold had been exceeded when more than 1000 (2000 * 50% = 1000) buffers were unavailable. DB2 would schedule write operations to decrease the number of unavailable buffers to 40% of the buffer pool, or decreased to 800 buffers (2000 * 40% = 800). This means that DB2 would schedule write operations for 200 pages[1].

When the DB2 is selecting updated pages to write to DASD, the data sets with the oldest updated pages are written asynchronously, using a Least Recently Used (LRU) algorithm to select the pages to be written. DB2 continues writing pages until the percent of unavailable pages goes below the DWQT threshold.

On the surface, it would seem that selecting a proper value for the DWQT threshold would only be a balance between (1) keeping updated pages in the buffer pool (in the expectation that they will be referenced again), versus (2) allowing space in the buffer pool for sequentially prefetched pages. With this initial view, the DWQT threshold should be high if the buffer pool supports mostly randomly accessed data sets. On the other hand, the DWQT threshold should be low if the buffer pool supports mostly sequentially accessed data sets, so that most of the buffer pool would be “available” for sequential prefetch I/O.

Unfortunately, this initial view must be tempered with an appreciation of what happens when an application issues a COMMIT. During COMMIT, all updated pages are written to DASD. With a large buffer pool and a large DWQT value, many hundreds (or thousands) of pages might be written after a COMMIT. This large I/O volume could cause severe CPU and I/O processing in a “spurt” of activity, and likely causing serious performance problems for applications using DB2.

Additionally, the view must be tempered with the appreciation that DB2 does not “discard” pages in the buffer pool after they have been written to DASD. Available pages in the buffer pool are used only when require for a GETPAGE request or for sequential prefetch.

[1] Note that DB2 is restricted to scheduling write operations for those pages that have been updated but have not been written to DASD. It is possible that the more than 800 pages were “in-use” and DB2 would not be able to write 200 pages. This would be an unusual situation, and likely would indicate that the buffer pool was seriously undersized.

If the buffer pool has mostly random I/O requests, there is little difference between having a low DWQT value and a high DWQT value with respect to whether DB2 might find a page already in the buffer pool. In either case, DB2 will examine the buffer pool (examining both "updated but not written to DASD" pages and "available" pages) to see whether a GETPAGE can be satisfied from the buffer pool.

The main difference would be if there were sequential GETPAGE requests and DB2 implemented sequential prefetch. With a low value for DWQT, DB2 could prefetch a large number of sequential pages, using the "available" buffer pool pages for the sequential I/O. These prefetched pages would still be considered "available" if needed to satisfy a random GETPAGE request. However, these prefetch pages could have usurped buffer pool pages that **had** been in the "updated but not written to DASD" category if these pages had become "available" because they had been written to DASD. Thus, a low value for the DWQT threshold could allow sequential pages to use much of the buffer pool[2].

- The Vertical Deferred Write Threshold (VDWQT) controls how much of the buffer pool in the "unavailable" category can be used to support an individual data set. This threshold is expressed as a percentage of the virtual buffer pool that might be occupied by updated pages from a single data set.

  The VDWQT threshold is checked whenever an update to a page is completed. If the percent of updated pages <u>for the data set</u> exceeds the threshold, writes are scheduled for that data set. The default value for the VDWQT threshold is 10%, indicating that writes are scheduled for any data set when its "updated but not written to DASD" pages exceed 10% of the buffer pool.

Reaching the DWQT threshold can result in spurts of I/O activity, with possible adverse effects on both other DB2 buffer pool I/O and other system I/O. Consequently, it is not desirable to reach the DWQT threshold, particularly for large buffer pools.

[2]The Sequential Steal Threshold (VPSEQT) can be used to control how much of the buffer pool can be used to support sequential I/O. This threshold is a percentage of the virtual buffer pool that might be occupied by sequentially accessed pages. DB2 checks the VPSEQT threshold before acquiring an "available" buffer for a sequentially accessed page. If the VPSEQT threshold has been exceeded, DB2 tries to acquire an "available" buffer holding a sequentially accessed page rather than one holding a randomly accessed page. Additionally, other variable thresholds can be used to control how much of the sequential I/O portion of the buffer pool can support parallel I/O operations and to assist with parallel operations initiated from another DB2 in the data sharing group.

©Copyright 1999, Computer Management Sciences, Inc.

CPExpert compares the QBSTDWT variable in DB2STATB (the number of times the DWQT threshold was reached) with the **QBSTDWT** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-222 when the number of times the DWQT threshold was reached exceeds the value specified by the **QBSTDWT** guidance variable.

The default value for the **QBSTDWT** guidance variable is 0, indicating that CPExpert should produce Rule DB2-222 whenever the DWQT threshold was reached. **This finding is suppressed if the buffer pool supports sort activity.**

The following example illustrates the output from Rule DB2-222:

```
RULE DB2-222: DEFERRED WRITE THRESHOLD WAS REACHED

   Buffer Pool 1: If the percentage of unavailable pages in the virtual
   buffer pool exceeds the Deferred Write Threshold, write operations are
   scheduled for enough data sets (at up to 128 pages per data set) to
   decrease the number of unavailable buffers to 10% below the threshold.
   The default value for this threshold is 50%, and the value specified
   in your system is 50%.  The Deferred Write Threshold normally should
   not be reached, unless the buffer pool is too small or the Vertical
   Deferred Write Threshold is too large (the Vertical Deferred Write
   Threshold was set to 10% for this buffer pool.  This situation occurred
   for Buffer Pool 1 during the intervals shown below:

                                 BUFFERS    TIMES DEFERRED WRITE
   MEASUREMENT INTERVAL         ALLOCATED     THRESHOLD REACHED
   21:32-21:47, 08SEP1998         2,000             32
```

Suggestion: Since the VDWQT threshold is normally 10% and the DWQT threshold is 50%, you normally would not reach the DWQT threshold. The most likely causes of reaching the DWQT threshold are (1) there were a relatively large number of active data sets with update activity assigned to the buffer pool, (2) the I/O subsystem was experiencing poor performance, (3) the VDWQT threshold was nearly as large as the DWQT threshold, or (4) the MVS I/O priority for DB2 was low relative to the MVS dispatching priority of DB2. If Rule DB2-222 is produced regularly, you should consider the following alternatives:

- Review the number of active data sets with update activity assigned to the buffer pool. Even if the VDWQT threshold was set at 10% (the default), if more than five active data sets were assigned to the buffer pool, you might encounter the DWQT threshold (with a default of 50%) before triggering any of the VDWQT threshold.

  You should consider reassigning some active data sets to other buffer pools.

- You should examine the overall performance of your I/O subsystem. If the I/O subsystem used by DB2 is performing poorly, DB2 write operations might not be completed in a timely fashion. Improving overall I/O performance might be necessary for DB2 to write the updated buffers.

- You should consider lowering the VDWQT threshold. Lowering the VDWQT threshold will cause updated pages to be written when the percent of updated pages for a single data set exceeds the threshold. This alternative is particularly relevant if (1) the VDWQT threshold is nearly as high as the DWQT threshold or (2) a relatively large number of active data sets with updates are assigned to the buffer pool.

- Review the MVS I/O dispatching priority for DB2. If you are running DB2 with MVS/SP 4.3.0 or later and DFSMS/MVS 1.1 or later, DB2 can schedule synchronous read/write I/Os and prefetch read I/Os under the application address space's I/O scheduling priority. To do this, you must do both of the following:

  - Enable MVS I/O priority scheduling by specifying IOQ=PRTY in the IEAIPSxx member of SYS1.PARMLIB. Use the IOP parameter to set the I/O priority for the address space of a performance group. When you do this, the following scheduling priorities are in effect:

    - The application's address space determines the I/O scheduling priority for single-page synchronous read and write I/Os, sequential prefetch, list sequential prefetch, and sequential detection.

    - DB2's database services address space (ssnmDBM1) determines the I/O scheduling priority for asynchronous write I/Os.

  - If you specify IOQ=PRTY, it is critical that you specify the proper IOP value for each address space. If IOQ=PRTY is specified and the IOP parameter is not set for an address space, the I/O scheduling priority for that address space defaults to the address space's processor scheduling priority; in other words, the IOP value defaults to the dispatching priority (DP) value.

    If you do not specify values for the IOP parameter, CICS and IMS regions might have lower I/O scheduling priority than DB2's ssnmDBM1 address space. An I/O scheduling priority lower than ssnmDBM1's I/O scheduling priority could result in inconsistent I/O response time for transaction applications.

    To improve response time for transaction processing, set the CICS- and IMS-dependent IOP values higher than DB2's ssnmDBM1 address space. To favor transaction processing over query users, set

the IOP values for CICS and IMS MPP regions higher than those for TSO and batch users.

Also be sure that ssnmDBM1 has a higher priority than TSO and batch. This helps ensure that deferred write I/Os are scheduled before prefetch read I/Os, thereby preventing a shortage of available buffers.

- You can alter CPExpert's analysis by modifying the **QBSTDWT** guidance variable in USOURCE(DB2GUIDE).

Reference: DB2 for OS/390 Version 3: Installation Guide
Section 7.5.1.7 (Buffer Pool Thresholds)
Section 7.6.9 (MVS Performance Options for DB2)

DB2 for OS/390 Version 4: Administration Guide
Section 5.4.1.7 (Buffer Pool Thresholds)
Section 5.5.9 (MVS Performance Options for DB2)

DB2 for OS/390 Version 5: Administration Guide
Section 5.4.1.7 (Buffer Pool Thresholds)
Section 5.5.9 (MVS Performance Options for DB2)

DB2 for OS/390 Version 6: Administration Guide
Section 5.4.1.6 (Buffer Pool Thresholds)
Section 5.5.9 (MVS Performance Options for DB2)

©Copyright 1999, Computer Management Sciences, Inc.

Rule DB2-223: VPSEQT value might be too small

Finding: CPExpert believes that the Sequential Steal Threshold (VPSEQT) value might be too small.

Impact: This finding can have a MEDIUM IMPACT, or HIGH IMPACT on the performance of the DB2 subsystem.

Discussion: DB2 keeps track of the access characteristics of pages in the buffer pool: whether the pages were accessed as a result of a sequential GETPAGE request, or whether the pages were accessed as a result of a random GETPAGE request.

The Sequential Steal Threshold (VPSEQT) specifies the percentage of the virtual buffer pool that can be occupied by sequentially accessed pages (as distinguished from those pages that are randomly accessed).

When DB2 needs a page in the buffer pool, it first checks the percent of the buffer pool that is occupied by sequentially accessed pages.

- DB2 will steal sequentially accessed pages if more than the VPSEQT percent of the buffer pool is occupied by sequentially accessed pages.

- DB2 will steal a randomly accessed page if less than the VPSEQT percent of the buffer pool is occupied by sequentially accessed pages.

The default value for the VPSEQT threshold is 80%, indicating that DB2 would begin stealing sequentially accessed pages when more than 80% of the buffer pool were sequentially accessed.

For example, suppose that the VPSIZE (the number of buffers in the buffer pool) had been specified as 2000 pages. If the default VPSEQT of 80% were used, DB2 would begin stealing sequentially accessed pages when more than 1600 pages (2000 * .8 = 1600) were sequentially accessed. If less than 1600 pages were sequentially accessed, DB2 would steal randomly accessed pages (the remaining 400 pages in this example).

For most environments, the value specified for the VPSEQT threshold determines the available space in the buffer pool that can be used for pages read in by DB2's sequential prefetch mechanism[1].

[1] Please refer to Rule DB2-230 for a discussion of sequential prefetch.

©Copyright 1999, Computer Management Sciences, Inc.

A low value for VPSEQT indicates that less space in the buffer pool can be used by sequential prefetch, while a high value for VPSEQT indicates that more of the buffer pool can be used for sequential prefetch.

Sequential prefetch can significantly improve DB2 performance if most of the GETPAGE requests are sequential. This is because sequential prefetch brings pages into the virtual buffer pool before they are required and reads several pages with a single I/O operation. Consequently, sequential prefetch allows CPU processing and I/O operations to be overlapped.

On the other hand, it would not be a good use of the buffer pool to allow a significant percent of the buffer pool to be used for sequentially accessed pages if most of the GETPAGE requests are random. This is because a low buffer hit ratio would result from monopolizing the buffer pool with low-activity sequentially accessed pages.

From the above discussion, if should be clear that the ratio of randomly accessed pages in the buffer pool to sequentially accessed pages in the buffer pool normally would be the determining factor in selecting values for the VPSEQT threshold. The VPSEQT threshold would be lowered if there are a relatively large percentage of randomly accessed pages. The VPSEQT threshold should be high (set to 99, for example) if there are few randomly accessed pages.

CPExpert computes the percent of GETPAGE requests that were sequential. The calculation is uses data in DB2STATB, and is performed as shown below:

$$\textit{Percent sequential GETPAGE requests} = \frac{\textit{GETPAGE requests issued by sequential requestors}}{\textit{GETPAGE requests}}$$

CPExpert compares the computed percent of GETPAGE requests that were issued by sequential requestors with the **PCTSEQ** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-223 when the percent of GETPAGE requests that were issued by sequential requestors is **greater than** the value specified by the **PCTSEQ** guidance variable, and the value specified for VPSEQT threshold for the buffer pool (obtained from the QDBPVPSH variable in DB2STAT2) is **less than 99%**.

The default value for the **PCTSEQ** guidance variable is 95%, indicating that Rule DB2-223 should be produced the percent of GETPAGE requests that were issued by sequential requestors was **greater than 95%**, and the value for the VPSEQT threshold is **less than 99%**.

The following example illustrates the output from Rule DB2-223:

```
RULE DB2-223: VPSEQT VALUE MIGHT BE TOO SMALL

   Buffer Pool 5: CPExpert believes that the Sequential Steal Threshold
   (VPSEQT) value might be too small.  Most of the GETPAGE requests for
   this buffer pool were issued by sequential requestors.  However, the
   VPSEQT threshold was not specified to allow sequential prefetch to be
   most efficient. This situation occurred for Buffer Pool 5 during the
   intervals shown below:
```

MEASUREMENT INTERVAL	BUFFERS ALLOCATED	GETPAGE REQUESTS	PERCENT SEQUENTIAL	VPSEQT VALUE
8:51- 9:21, 15SEP1999	20,000	1,914,240	99	75
9:21- 9:51, 15SEP1999	20,000	2,032,640	99	75
9:51-10:21, 15SEP1999	20,000	3,207,168	99	75
10:21-10:51, 15SEP1999	20,000	3,240,832	99	75
10:51-11:21, 15SEP1999	20,000	4,895,872	99	75
11:21-11:50, 15SEP1999	20,000	4,333,184	99	75
11:50-12:20, 15SEP1999	20,000	4,240,384	99	75
12:20-12:50, 15SEP1999	20,000	4,082,176	99	75
12:50-13:20, 15SEP1999	20,000	2,999,552	99	75

Suggestion: If Rule DB2-223 is **consistently** produced, you should consider the following alternatives:

- First, make sure that Rule DB2-223 is consistently produced, particularly for intervals when important DB2 work is executing. This finding might be produced for a few intervals when unimportant work is being processed (for example, during off-shift work). Consequently, make sure that the finding is produced for all (or most) intervals when important work is executing.

  Alternatively, you might have a situation in which one class of work (e.g., work with mostly sequential requests) executes during a particular shift, while another class of work (e.g., work with mostly random requests) executes during a different shift. In this situation, you might consider specifying two different settings for the VPSEQT threshold.

- Unless you have an unusual situation, you should consider increasing the value of the VPSEQT threshold to 99%.

  The result from CPExpert's analysis is that almost all of the GETPAGE requests are issued by sequential requestors and very few GETPAGE requests are issued by random requestors. In this situation, it is advantageous to allow almost all of the buffer pool to be dedicated to sequentially accessed pages and allow the sequential prefetch mechanism to operate efficiently.

- You can alter CPExpert's analysis by modifying the **PCTSEQ** guidance variable in USOURCE(DB2GUIDE).

Reference: DB2 for OS/390 Version 3: Administration Guide
Section 7.5.1.7.1 (Fixed Thresholds)
Section 7.9.5 (Sequential Prefetch and Query Performance)

DB2 for OS/390 Version 4: Administration Guide
Section 5.4.1.7.1 (Fixed Thresholds)
Section 5.10.5.1 (Sequential Prefetch)

DB2 for OS/390 Version 5: Administration Guide
Section 5.4.1.7.1 (Fixed Thresholds)
Section 5.10.5.1 (Sequential Prefetch)

DB2 for OS/390 Version 6: Administration Guide |
Section 5.4.1.6.1 (Fixed Thresholds) |
Section 5.10.5.1 (Sequential Prefetch) |

©Copyright 1999, Computer Management Sciences, Inc.

Rule DB2-224: VPSEQT value might be too large

Finding: CPExpert believes that the Sequential Steal Threshold (VPSEQT) value might be too large.

Impact: This finding can have a MEDIUM IMPACT, or HIGH IMPACT on the performance of the DB2 subsystem.

Discussion: DB2 keeps track of the access characteristics of pages in the buffer pool: whether the pages were accessed as a result of a sequential GETPAGE request, or whether the pages were accessed as a result of a random GETPAGE request.

The Sequential Steal Threshold (VPSEQT) specifies the percentage of the virtual buffer pool that can be occupied by sequentially accessed pages (as distinguished from those pages that are randomly accessed).

When DB2 needs a page in the buffer pool, it first checks the percent of the buffer pool that is occupied by sequentially accessed pages.

- DB2 will steal sequentially accessed pages if more than the VPSEQT percent of the buffer pool is occupied by sequentially accessed pages.
- DB2 will steal a randomly accessed page if less than the VPSEQT percent of the buffer pool is occupied by sequentially accessed pages.

The default value for the VPSEQT threshold is 80%, indicating that DB2 would begin stealing sequentially accessed pages when more than 80% of the buffer pool were sequentially accessed.

For example, suppose that the VPSIZE (the number of buffers in the buffer pool) had been specified as 2000 pages. If the default VPSEQT of 80% were used, DB2 would begin stealing sequentially accessed pages when more than 1600 pages (2000 * .8 = 1600) were sequentially accessed. If less than 1600 pages were sequentially accessed, DB2 would steal randomly accessed pages (the remaining 400 pages in this example).

For most environments, the value specified for the VPSEQT threshold determines the available space in the buffer pool that can be used for pages read in by DB2's sequential prefetch mechanism[1].

[1] Please refer to Rule DB2-230 for a discussion of sequential prefetch.

©Copyright 1999, Computer Management Sciences, Inc.

A low value for VPSEQT indicates that less space in the buffer pool can be used by sequential prefetch, while a high value for VPSEQT indicates that more of the buffer pool can be used for sequential prefetch.

Sequential prefetch can significantly improve DB2 performance if most of the GETPAGE requests are sequential. This is because sequential prefetch brings pages into the virtual buffer pool before they are required and reads several pages with a single I/O operation. Consequently, sequential prefetch allows CPU processing and I/O operations to be overlapped.

On the other hand, it would not be a good use of the buffer pool to allow a significant percent of the buffer pool to be used for sequentially accessed pages if most of the GETPAGE requests are random.

From the above discussion, if should be clear that the ratio of randomly accessed pages in the buffer pool to sequentially accessed pages in the buffer pool normally would be the determining factor in selecting values for the VPSEQT threshold. The VPSEQT threshold would be lowered if there are a relatively large percentage of randomly accessed pages. The VPSEQT threshold should be high (set to 99, for example) if there are few randomly accessed pages.

CPExpert computes the percent of GETPAGE requests that were random. The calculation is uses data in DB2STATB, and is performed as shown below:

$$\textit{Percent random GETPAGE requests} = \frac{\textit{GETPAGE requests} - \textit{Sequential GETPAGE requests}}{\textit{GETPAGE requests}}$$

CPExpert compares the computed percent of GETPAGE requests that were issued by random requestors with the **PCTRAND** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-224 when the percent of GETPAGE requests that were issued by random requestors is **greater than** the value specified by the **PCTRAND** guidance variable, and the value specified for VPSEQT threshold for the buffer pool (obtained from the QDBPVPSH variable in DB2STAT2) is **greater than 10%**.

The default value for the **PCTRAND** guidance variable is 50%, indicating that Rule DB2-402 should be produced when the percent of GETPAGE requests that were issued by random requestors was **greater than 95%**, and the value for the VPSEQT threshold is **greater than 10%**.

Suggestion: If Rule DB2-224 is **consistently** produced, you should consider the following alternatives:

- First, make sure that Rule DB2-224 is consistently produced, particularly for intervals when important DB2 work is executing. This finding might be produced for a few intervals when unimportant work is being processed (for example, during off-shift work). Consequently, make sure that the finding is produced for all (or most) intervals when important work is executing.

  Alternatively, you might have a situation in which one class of work (e.g., work with mostly sequential requests) executes during a particular shift, while another class of work (e.g., work with mostly random requests) executes during a different shift. In this situation, you might consider specifying two different settings for the VPSEQT threshold.

- Unless you have an unusual situation, you should consider reducing the value of the VPSEQT threshold to a low value (for example, 5%).

  The result from CPExpert's analysis is that almost all of the GETPAGE requests are issued by random requestors and very few GETPAGE requests are issued by sequential requestors. In this situation, it is advantageous to allow almost all of the buffer pool to be dedicated to randomly accessed pages.

- You can alter CPExpert's analysis by modifying the **PCTRAND** guidance variable in USOURCE(DB2GUIDE).

Reference: DB2 for OS/390 Version 3: Administration Guide
Section 7.5.1.7.1 (Fixed Thresholds)
Section 7.9.5 (Sequential Prefetch and Query Performance)

DB2 for OS/390 Version 4: Administration Guide
Section 5.4.1.7.1 (Fixed Thresholds)
Section 5.10.5.1 (Sequential Prefetch)

DB2 for OS/390 Version 5: Administration Guide
Section 5.4.1.7.1 (Fixed Thresholds)
Section 5.10.5.1 (Sequential Prefetch)

DB2 for OS/390 Version 6: Administration Guide
Section 5.4.1.6.1 (Fixed Thresholds)
Section 5.10.5.1 (Sequential Prefetch)

©Copyright 1999, Computer Management Sciences, Inc.

Rule DB2-225: DWQT and VDWQT might be too large

Finding: The average number of pages written per asynchronous write was relatively high, indicating that pages will be written in “spikes” of activity. This finding means that the DWQT and VDWQT thresholds might be too large.

Impact: This finding can have a LOW IMPACT or MEDIUM IMPACT on the performance of the DB2 subsystem.

Discussion: Buffer pool write operations are usually performed concurrently with user requests. Updated pages are queued by data set until they are written when:

- A DB2 *checkpoint* is taken
- The percentage of unavailable pages in a virtual buffer pool exceeds a preset limit called the Deferred Write Threshold (DWQT).
- The percentage of updated pages in a virtual buffer pool for a single data set exceeds a preset limit called the Vertical Deferred Write Threshold (VDWQT).

Up to 32 4KB or 4 32KB pages can be written in a single I/O operation.

Two variable thresholds control the asynchronous write of pages from buffer pools: (1) the Deferred Write Threshold and (2) the Vertical Deferred Write Threshold.

- The Deferred Write Threshold is a percentage of the virtual buffer pool that might be occupied by unavailable pages, including both updated pages and pages in use. The default value for the Deferred Write Threshold is 50%. DB2 checks this threshold when an update to a page is completed. If the percentage of unavailable pages in the virtual buffer pool exceeds the threshold, write operations are scheduled for enough data sets (at up to 128 pages per data set) to decrease the number of unavailable buffers to 10% below the threshold. For example, if the threshold is 50%, the number of unavailable buffers is reduced to 40%.
- The Vertical Deferred Write Threshold is expressed as a percentage of the virtual buffer pool that might be occupied by updated pages from a **single data set**. The default value for the Vertical Deferred Write Threshold is 10%. DB2 checks this threshold whenever an update to a

page is completed. If the percentage of updated pages for the data set exceeds the threshold, writes are scheduled for that data set.

If a virtual buffer pool is large enough, it is unlikely that the default values of either DWQT or VDWQT will ever be reached. In this case, surges of write I/Os tend to occur as deferred writes are triggered by DB2 *checkpoints*. Lowering the VDWQT and the DWQT could improve performance by distributing the write I/Os more evenly over time.

While it is normally suggested that the DWQT and VDWQT thresholds values be relatively low to avoid "spikes" in page write activity, this advice does not apply to buffer pools heavily used by sort processing. Instead, the advice is reversed for these buffer pools, to allow sort processing to process sorted data in central storage. Rule DB2-226 analyzes whether buffer pools are heavily used by sort processing, and have relatively low DWQT, VDWQT, or VPSEQT thresholds.

Additionally, specifying a large DWQT or VDWQT value for a small buffer pool could result in the Data Management Threshold (DMTH) or Immediate Write Threshold (IWTH) being reached. CPExpert analyzes whether these thresholds were reached in Rule DB2-220 and Rule DB2-221, respectively.

CPExpert computes the average pages written per asynchronous write by dividing QBSTWIO (asynchronous writes) into QBSTPWS (number of pages written)[1]. The resulting average pages written per asynchronous write activity is compared against the **QBSTWIO** guidance variable in USOURCE(DB2GUIDE).

Additionally, CPExpert examines the QBSTWFR variable (the number of merge passes) in the DB2STATB statistics. When the value of this variable exceeds the **QBSTWFR** guidance variable in USOURCE(DB2GUIDE), CPExpert concludes that the buffer pool is heavily used for sort activity. CPExpert suppresses Rule DB2-225 for buffer pools that are heavily used for sort activity. Please refer to Rule DB2-226 for further information.

Additionally, CPExpert examines the QDBPVDQT variable (the value of the VDWQT parameter) in DB2STAT2. CPExpert suppresses Rule DB2-225 if this value is zero, as DB2 will initiate write I/O whenever 32 buffers are used for a single data set for updated pages.

CPExpert produces Rule DB2-225 when the average pages written per asynchronous write activity exceeds the value specified by the **QBSTWIO** guidance variable, and the other conditions of this rule are met.

[1] CPExpert excludes intervals with less than 500 asynchronous writes, to eliminate spurious findings.

The default value for the **QBSTWIO** guidance variable is 20, indicating that CPExpert should produce Rule DB2-225 when more than 20 pages were written per asynchronous write activity .

The following example illustrates the output from Rule DB2-225:

```
RULE DB2-225: DWQT AND VDWQT MIGHT BE TOO LARGE

   Buffer Pool 0: The average number of pages written per asynchronous
   write was relatively high.  This means that buffer write operations
   will tend to occur with "spikes" of pages written, since a relatively
   large number of pages will be written at once.  You should consider
   reducing the DWQT and VDWQT values to reduce the "spikes" of pages
   written. The DWQT threshold was specified as 50% and the VDWQT threshold
   was specified as 10%.  This situation occurred for Buffer Pool 0 during
   the intervals shown below:

                                 BUFFERS     PAGES      ASYNC   AVG PAGES
   MEASUREMENT INTERVAL         ALLOCATED   WRITTEN    WRITES   PER WRITE
   21:32-21:32, 08SEP1998         2,000      79,829    21,339       24
```

Suggestion: If Rule DB2-225 is produced regularly, you should consider the following alternatives:

- You can lower the DQWT and VDWQT thresholds, by using the ALTER BUFFERPOOL command. Changing a threshold in one virtual buffer pool or hiperpool has no effect on any other virtual buffer pool or hiperpool.

  Of the two thresholds, it is much more advantageous to lower the VDWQT threshold. You should consider setting the VDWQT threshold to zero. When VDWQT is set to zero, DB2 will initiate write I/O whenever 32 buffers are used for a single data set for updated pages.

  Please note that the VDWQT value may not be greater than the DWQT value.

- You can alter CPExpert's analysis by modifying the **QBSTWIO** guidance variable in USOURCE(DB2GUIDE).

Reference: DB2 for OS/390 Version 3: Administration Guide
Section 7.5.1 (Tuning Database Buffer Pools)

DB2 for OS/390 Version 4: Administration Guide
Section 5.4.1 (Tuning Database Buffer Pools)

DB2 for OS/390 Version 5: Administration Guide
Section 5.4.1 (Tuning Database Buffer Pools)

DB2 for OS/390 Version 6: Administration Guide |
Section 5.4.1 (Tuning Database Buffer Pools) |

Rule DB2-226: DWQT, VDWQT, or VPSEQT might be too small

Finding: CPExpert noticed that a buffer pool was used for sort activity, and the DWQT, VDWQT, or VPSEQT thresholds were relatively low. This finding means that the DWQT, VDWQT, or VPSEQT thresholds might be too small.

Impact: This finding can have a LOW IMPACT or MEDIUM on the performance of the DB2 subsystem.

Discussion: Buffer pool write operations are usually performed concurrently with user requests. Updated pages are queued by data set until they are written when:

- A DB2 *checkpoint* is taken
- The percentage of unavailable pages in a virtual buffer pool exceeds a preset limit called the Deferred Write Threshold (DWQT).
- The percentage of updated pages in a virtual buffer pool for a single data set exceeds a preset limit called the Vertical Deferred Write Threshold (VDWQT).

Up to 32 4KB or 4 32KB pages can be written in a single I/O operation.

Please refer to Rule DB2-225 for a more complete discussion of the DWQT and VDWQT thresholds.

The VPSEQT threshold is a percentage of the virtual buffer pool that might be occupied by sequentially accessed pages. These pages can be in any state: updated, in-use, or available. Hence, any page might or might not count toward exceeding any other buffer pool threshold. The default value for the VPSEQT is 80%.

While it is normally suggested that the DWQT and VDWQT thresholds values be relatively low to avoid "spikes" in page write activity, this advice does not apply to buffer pools heavily used by sort processing. Instead, the advice is reversed for these buffer pools, to allow sort processing to process sorted data in central storage.

DNSDB07 is the work area for DB2 sorting (and other related activity). Not only will the pages in DSNDB07 normally be referenced quickly, but the references typically are sequential. If the DWQT and VDWQT were set low, the pages would be written to DASD when these thresholds were

©Copyright 1999, Computer Management Sciences, Inc.

reached. However, the pages would be quickly reloaded. Consequently, the buffer pool that contains the DSNDB07 work files should have the DWQT, VDWQT, and VPSEQT thresholds set very high[1] to improve sort performance.

CPExpert examines the QBSTWFR variable (the number of sort merge passes) in the DB2STATB statistics. When the value of this variable exceeds the **QBSTWFR** guidance variable in USOURCE(DB2GUIDE), CPExpert concludes that the buffer pool is heavily used for sort activity. CPExpert produces Rule DB2-226 when (1) buffer pools are heavily used by sort activity, (2) the DWQT threshold is less than 90, (3) the VDWQT threshold is less than 90, and (4) the VPSEQT threshold is less than 90.

The default value for the **QBSTWFR** guidance variable is 1, indicating that CPExpert should conclude that the buffer pool is used heavily for sort processing when merge passes occurred. **It is unlikely that this value is correct for every buffer pool in your installation!** A low value was selected simply to alert you to this consideration and to allow you to specify a value that is more relevant to the buffer pools in your installation.

The following example illustrates the output from Rule DB2-226:

```
RULE DB2-226: DWQT, VDWQT, OR VPSEQT MIGHT BE TOO SMALL

   Buffer Pool 7:  CPExpert noticed that this buffer pool is used for
   sort activity.  The DWQT threshold was specified as 50, the VDWQT
   threshold was specified as 10, and the VPSEQT threshold was specified
   as 100.  While it is normally suggested that the DWQT and VDWQT values
   be relatively low to avoid "spikes" in page write activity, this advice
   does not apply to buffer pools heavily used by sort processing.
   Instead, the advice is reversed for these buffer pools, to allow sort
   processing to process sorted data in central storage. This situation
   occurred for Buffer Pool 7 during the intervals shown below:

                              BUFFERS   TOTAL MERGE   THRESHOLDS REACHED
   MEASUREMENT INTERVAL      ALLOCATED     PASSES     DWQT         VDWQT
   15:20-15:20, 15SEP1999      5,000        1,647        7           270
   15:50-15:50, 15SEP1999      5,000        1,782        0            17
   16:19-16:19, 15SEP1999      5,000        1,660        0           277
   16:49-16:49, 15SEP1999      5,000        1,486        0           102
   17:19-17:19, 15SEP1999      5,000        1,104        0           151
```

Suggestion: If Rule DB2-226 is produced regularly, you should consider the following alternatives:

- Verify whether the buffer pool is used heavily (or exclusively) for sorting. If this is the case, you should set the DWQT, VDWQT, and VPSEQT

[1] This comment is valid only if the buffer pool is used exclusively (or extensively) for sorting.

thresholds to very high values. If the buffer pool is used exclusively for sorting, you should set these values to 90.

- You should alter CPExpert's analysis by modifying the **QBSTWFR** guidance variable in USOURCE(DB2GUIDE), for any buffer pools that are not exclusively (or heavily) used for sort activity.

  Please note that CPExpert allows guidance variables to be specified globally (that is, they apply to all buffer pools) or for a specific buffer pool. A reasonable approach would be to set the global value for the QBSTWFR guidance variable to some high value (e.g., 999999999), but set QBSTWFR to the default value of "1" for any buffer pool that is used primarily for sort activity.

  Alternatively, you can simply "turn off" Rule DB2-226 after you have decided on appropriate values for the DWQT, VDWQT, and VPSEQT parameters for the buffer pool used primarily for sort activity.

- You can increase the DWQT, VDWQT, and VPSEQT thresholds for buffer pools that are exclusively (or heavily) used for sort activity, by using the ALTER BUFFERPOOL command. Changing a threshold in one virtual buffer pool or hiperpool has no effect on any other virtual buffer pool or hiperpool. You might wish to set the DWQT threshold to 90, the VDWQT threshold to 90, and the VPSEQT threshold to 95. This would indicate that 95% of the use of DSNDB07 is sequential and that updated pages should be written to DASD only when the pool is almost totally full.

  This alternative is valid only if the buffer pool is sufficiently large that these thresholds are not reached frequently. If the thresholds are reached frequently, system performance could suffer. In this case, you should consider increasing the size of the buffer pool.

  Additionally, note that the Sequential Prefetch (SPTH) is set at 90, which means that sequential prefetch would be disabled when the percent of unavailable buffers reached 90% of the buffer pool. This situation would be highly undesirable for most buffer pools, but is not a concern for buffer pools that are dedicated to sort processing.

Reference: DB2 for OS/390 Version 3: Installation Guide
Section 7.5.1 (Tuning Database Buffer Pools)
Section 7.5.4 (Controlling Sort Pool Size and Sort Processing)

DB2 for OS/390 Version 4: Installation Guide
Section 5.4.1 (Tuning Database Buffer Pools)
Section 5.4.4 (Controlling Sort Pool Size and Sort Processing)

DB2 for OS/390 Version 5: Installation Guide
Section 5.4.1 (Tuning Database Buffer Pools)
Section 5.4.4 (Controlling Sort Pool Size and Sort Processing)

DB2 for OS/390 Version 6: Installation Guide
Section 5.4.1 (Tuning Database Buffer Pools)
Section 5.4.4 (Controlling Sort Pool Size and Sort Processing)

©Copyright 1999, Computer Management Sciences, Inc.

Rule DB2-230: Sequential prefetch was disabled - buffer shortage

Finding: Sequential prefetch was disabled because the Sequential Prefetch Threshold (SPTH) was reached

Impact: This finding can have a MEDIUM IMPACT, or HIGH IMPACT on the performance of the DB2 subsystem.

Discussion: DB2 uses three read mechanisms: normal read, sequential prefetch, and list sequential prefetch.

- **Normal Read**: Normal read is used when just one or a few consecutive pages are retrieved. The unit of transfer for a normal read is one page.

  Additionally, normal read is performed when the number of concurrent sequential prefetch I/O operations exceeds the number of DB2 sequential prefetch *read engines*. The maximum number of sequential prefetch read engines is a constant within DB2 and is set to 300. When the maximum read engines has been exhausted, DB2 disables sequential prefetch and performs asynchronous I/O operations. Please refer to Rule DB2-231 for additional information.

- **Sequential Prefetch**: Sequential prefetch is a mechanism that triggers consecutive asynchronous I/O operations. Sequential prefetch brings pages into the virtual buffer pool before they are required and reads several pages with a single I/O operation. Consequently, sequential prefetch allows CPU processing and I/O operations to be overlapped.

  Sequential prefetch can be used to read data pages, used by table space scans, or used by index scans with clustered data reference. Sequential prefetch can also be used to read index pages in an index scan.

- **List Sequential Prefetch**: List Sequential Prefetch is used to prefetch data pages that are not contiguous (such as through non-clustered indexes). List prefetch can also be used by incremental image copy.

The maximum number of pages read by sequential prefetch is determined by the size of the buffer pool used. Prior to DB2 Version 6, buffer pools can be composed of either 4K buffers or 32K buffers[1]. The following shows the pages read by a sequential prefetch for an application[2]:

[1] With DB2 Version 6, buffer pools can also be composed of 8K and 16K buffers.

[2] For certain utilities (LOAD, REORG, RECOVER), the prefetch quantity can be twice as much as shown.

©Copyright 1999, Computer Management Sciences, Inc.

- **4K buffer pools**

Buffer Pool Size	Pages Read by Prefetch
< 224 buffers	8 pages for each asynchronous I/O
224-999 buffers	16 pages for each asynchronous I/O
1000+ buffers	32 pages for each asynchronous I/O

- **32K buffer pools**

Buffer Pool Size	Pages Read by Prefetch
< 17 buffers	0 pages for each asynchronous I/O
17-99 buffers	2 pages for each asynchronous I/O
100+ buffers	4 pages for each asynchronous I/O

The Sequential Prefetch Threshold (SPTH) is a fixed threshold that controls whether sequential prefetch is enabled at execution time. The Sequential Prefetch Threshold (SPTH) is fixed at 90% of the buffer pool that might be occupied by unavailable pages.

This threshold is checked at two different times:

- Before scheduling a prefetch operation. If the threshold has been exceeded, the prefetch is not scheduled.

- During buffer allocation for an already-scheduled prefetch operation. If the threshold has been exceeded, the prefetch is canceled.

Since sequential prefetch allows CPU processing and I/O operations to be overlapped, sequential prefetch significantly improves performance.

CPExpert compares the QBSTSPD variable in DB2STATB (the number of times sequential prefetch was disabled because the SPTH threshold was reached) with the **QBSTSPD** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-230 when the number of number of times sequential prefetch was disabled exceeds the value specified by the **QBSTSPD** guidance variable.

©Copyright 1999, Computer Management Sciences, Inc.

The default value for the **QBSTSPD** guidance variable is 0, indicating that CPExpert should produce Rule DB2-230 whenever sequential prefetch was disabled during execution.

The following example illustrates the output from Rule DB2-230:

```
RULE DB2-230: SEQUENTIAL PREFETCH WAS DISABLED - BUFFER SHORTAGE

   Buffer Pool 9: Sequential prefetch is disabled when there is a buffer
   shortage, as controlled by the Sequential Prefetch Threshold (SPTH).
   Ideally, sequential prefetch should not be disabled, since performance
   is adversely affected.  If sequential prefetch is disabled a large
   number of times, the buffer pool size might be too small.  The
   sequential prefetch threshold was reached for Buffer Pool 9 during
   the intervals shown below.

                                 BUFFERS      TIMES SEQUENTIAL PREFETCH
   MEASUREMENT INTERVAL         ALLOCATED     DISABLED (BUFFER SHORTAGE)
   21:32-21:32, 08SEP1998           16                    4
```

Suggestion: Disabling sequential prefetch because the SPTH threshold was reached is highly undesirable. If Rule DB2-230 is produced more than occasionally, you should consider the following alternatives:

- Increase the size of the buffer pool. Increasing the buffer pool will effectively raise the level at which the SPTH threshold is reached.

- Review the settings of the Deferred Write Threshold (DWQT) and Vertical Deferred Write Threshold (VDWQT). Large values for the DWQT and VDWQT thresholds allows updated pages to use a larger portion of the virtual buffer pool. Consequently, specifying large values for DWQT and VDWQT can have a significant effect on the other thresholds.

  For example, for a workload in which pages are frequently updated, and the set of pages updated exceeds the size of the virtual buffer pool, setting both DWQT and VDWQT to 90% would probably cause the sequential prefetch threshold (and possibly the data management threshold and the immediate write threshold) to be reached frequently.

- Modify application programs so that they do more COMMITs. This will free pages in the buffer pool.

- You can alter CPExpert's analysis by modifying the **QBSTSPD** guidance variable in USOURCE(DB2GUIDE).

Reference: DB2 for OS/390 Version 3: Administration Guide
Section 7.5.1.7.1 (Fixed Thresholds)
Section 7.9.5 (Sequential Prefetch and Query Performance)

DB2 for OS/390 Version 4: Administration Guide
Section 5.4.1.7.1 (Fixed Thresholds)
Section 5.10.5.1 (Sequential Prefetch)

DB2 for OS/390 Version 5: Administration Guide
Section 5.4.1.7.1 (Fixed Thresholds)
Section 5.10.5.1 (Sequential Prefetch)

DB2 for OS/390 Version 6: Administration Guide
Section 5.4.1.6.1 (Fixed Thresholds)
Section 5.10.5.1 (Sequential Prefetch)

©Copyright 1999, Computer Management Sciences, Inc.

Rule DB2-231: Sequential prefetch was disabled - unavailable read engine

Finding: Sequential prefetch was disabled because no read engine was available.

Impact: This finding can have a MEDIUM IMPACT, or HIGH IMPACT on the performance of the DB2 subsystem.

Discussion: DB2 uses three read mechanisms: normal read, sequential prefetch, and list sequential prefetch.

- **Normal Read**: Normal read is used when just one or a few consecutive pages are retrieved. The unit of transfer for a normal read is one page.

- **Sequential Prefetch**: Sequential prefetch is a mechanism that triggers consecutive asynchronous I/O operations. Sequential prefetch brings pages into the virtual buffer pool before they are required and reads several pages with a single I/O operation. Consequently, sequential prefetch allows CPU processing and I/O operations to be overlapped.

  Sequential prefetch can be used to read data pages, by table space scans, or index scans with clustered data reference. It can also be used to read index pages in an index scan.

- **List Sequential Prefetch**: List Sequential Prefetch is used to prefetch data pages that are not contiguous (such as through non-clustered indexes). List prefetch can also be used by incremental image copy.

The sequential prefetches are implemented by DB2 subtasks, called *read engines*. The maximum number of concurrent prefetch I/O operations is a constant designed into DB2 (the maximum *read engines*). No more than 300 concurrent prefetch I/O operations can occur. Sequential prefetch is disabled when 300 read engines are active, and a normal read is performed.

CPExpert compares the QBSTREE variable in DB2STATB (the number of times sequential prefetch was disabled because of unavailable read engines) with the **QBSTREE** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-231 when the number of times sequential prefetch was disabled because of unavailable read engines exceeds the value specified by the **QBSTREE** guidance variable.

©Copyright 1999, Computer Management Sciences, Inc.

The default value for the **QBSTREE** guidance variable is 0, indicating that CPExpert should produce Rule DB2-231 whenever sequential prefetch was disabled because of unavailable read engines.

Suggestion: In practice, DB2 should not be constrained by the number of read engines, so Rule DB2-231 should never be produced. However, if Rule DB2-231 **is** produced, you should consider the following alternatives:

- **Reduce the time to perform I/O Operations**. You should review the "Reducing I/O Operations to Improve Response Time" sections listed in the references. These sections give detailed guidance, depending on the version of DB2.

- **Evaluate the overall performance of your system**. One basic cause of a large number of outstanding I/O operations can be the overall performance of your system. You should evaluate overall system performance to eliminate or reduce bottlenecks.

- **Reschedule workload**. You may evaluate the workload being processed when DB2 reaches the maximum number of read engines, to determine whether non-DB2 workload or low-priority DB2 workload (such as batch updates) can be scheduled at other times. By rescheduling work, you may improve the I/O performance, resulting in faster I/O response to individual sequential prefetch operations.

- **Evaluate the individual plans or packages**. You can evaluate selected plans or packages to improve their performance (particularly in the areas affecting I/O operations). The DB2 Application Guides can be consulted with regard to improving query and SQL performance.

- You can alter CPExpert's analysis by modifying the **QBSTREE** guidance variable in USOURCE(DB2GUIDE). Only in unusual circumstances would you wish to alter the default of QBSTREE=0 (in fact, if you should wish to alter this specification, please contact Computer Management Sciences so we can better understand your environment).

Reference: DB2 for OS/390 Version 3: Administration Guide
Section 7.4.1 (Reducing I/O Operations to Improve Response Time)
Section 7.6.4 (Placement of DB2 Data Sets)
Section 7.9.5 (Sequential Prefetch and Query Performance)
QBSTREE variable description in DSNWMSGS macro

DB2 for OS/390 Version 3: Application Guide
Section 1.1.7.3 (Performance)

©Copyright 1999, Computer Management Sciences, Inc.

DB2 for OS/390 Version 4: Administration Guide
Section 5.3.2 (Reducing the Time Needed to Perform I/O Operations)
Section 5.5.4 (Planning the Placement of DB2 Data Sets)
Section 5.10.5.1 (Sequential Prefetch)
QBSTREE variable description in DSNWMSGS macro

DB2 for OS/390 Version 4: Application Programming and SQL Guide
Section 6.3 (Tuning queries)

DB2 for OS/390 Version 5: Administration Guide
Section 5.3.2 (Reducing the Time Needed to Perform I/O Operations)
Section 5.5.4 (Planning the Placement of DB2 Data Sets)
Section 5.10.5.1 (Sequential Prefetch)
QBSTREE variable description in DSNWMSGS macro

DB2 for OS/390 Version 5: Application Programming and SQL Guide
Section 6.3 (Tuning your queries)

DB2 for OS/390 Version 6: Administration Guide
Section 5.3.2 (Reducing the Time Needed to Perform I/O Operations)
Section 5.5.4 (Planning the Placement of DB2 Data Sets)
Section 5.10.5.1 (Sequential Prefetch)
QBSTREE variable description in DSNWMSGS macro

DB2 for OS/390 Version 6: Application Programming and SQL Guide
Section 7.3 (Tuning your queries)

©Copyright 1999, Computer Management Sciences, Inc.

Rule DB2-232: Sequential prefetch not scheduled, Prefetch quantity = 0

Finding: Sequential prefetch was not scheduled for sort work files because the prefetch quantity was zero.

Impact: This finding can have a MEDIUM IMPACT, or HIGH IMPACT on the performance of the DB2 subsystem.

Discussion: DB2's sort process begins when ordered sets of rows (called initial runs) are written to work files. At the completion of the runs, when all the rows have been sorted and inserted into the work files, the work files are merged together into one work file containing the sorted data.

The work files used in sort are logical work files, which reside in temporary work file table spaces in database DSNDB07. DB2 uses the buffer pool when writing to the logical work file. The number of work files that can be used for sorting is limited only by the buffer pool size when you have the sort assist hardware.

Sequential prefetch is a mechanism that triggers consecutive asynchronous I/O operations. Sequential prefetch brings pages into the virtual buffer pool before they are required and reads several pages with a single I/O operation. Consequently, sequential prefetch allows CPU processing and I/O operations to be overlapped

Sequential prefetch can be used for the sort work files, but DB2 will use sequential prefetch only if sufficient buffers are available in the buffer pool. Normally the prefetch quantity for sort work files is up to a maximum of eight pages. However, if insufficient buffer pool space exists, DB2 will reduce the prefetch quantity for work files, and may set the prefetch quantity to zero. Prefetch will not be scheduled if the prefetch quantity is zero.

.

CPExpert compares the QBSTWKPD variable in DB2STATB (the number of times sequential prefetch was not scheduled for sort work files because the prefetch quantity was zero) with the **QBSTWKPD** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-232 when the number of times prefetch was not scheduled for sort work files exceeds the value specified by the **QBSTWKPD** guidance variable.

The default value for the **QBSTWKPD** guidance variable is 0, indicating that CPExpert should produce Rule DB2-232 whenever sequential prefetch

was not scheduled for sort work files because the prefetch quantity was zero.

Suggestion: If CPExpert regularly produces Rule DB2-232, you should consider the following alternatives:

- Review the settings of the Deferred Write Threshold (DWQT) and Vertical Deferred Write Threshold (VDWQT). Large values for the DWQT and VDWQT thresholds allow updated pages to use a larger portion of the virtual buffer pool. Conversely, small values for the DWQT and VDWQT thresholds cause DB2 to write updated pages to DASD, freeing more of the buffer pool for use by sequential prefetch.

- Increase the size of the buffer pool so that more pages are available.

- Allocate a buffer pool specifically for DSNDB07 usage. This alternative can be especially effective with high-use query systems whose reports make extensive use of sort activity.

- You can alter CPExpert's analysis by modifying the **QBSTWKPD** guidance variable in USOURCE(DB2GUIDE).

Reference: DB2 for OS/390 Version 3: Administration Guide
Section 7.5.1.7.1 (Fixed Thresholds)
Section 7.9.5 (Sequential Prefetch and Query Performance)

DB2 for OS/390 Version 4: Administration Guide
Section 5.4.1.7.1 (Fixed Thresholds)
Section 5.10.5.1 (Sequential Prefetch)

DB2 for OS/390 Version 5: Administration Guide
Section 5.4.1.7.1 (Fixed Thresholds)
Section 5.10.5.1 (Sequential Prefetch)

DB2 for OS/390 Version 6: Administration Guide
Section 5.4.1.6.1 (Fixed Thresholds)
Section 5.10.5.1 (Sequential Prefetch)

Rule DB2-234: Write engine was not available for asynchronous I/O

Finding: Asynchronous I/O was disabled because no write engine was available.

Impact: This finding can have a LOW IMPACT, MEDIUM IMPACT, or HIGH IMPACT on the performance of the DB2 subsystem. The level of impact depends on how often write engines were not available, and the effect of the significant write activity on other I/O operations in the system (such as prefetch I/O operations).

Discussion: Buffer pool write operations are usually performed concurrently (asynchronously) with user requests. Updated pages are queued by data set until they are written when:

- A DB2 *checkpoint* is taken
- The percentage of unavailable pages in a virtual buffer pool exceeds a preset limit called the Deferred Write Threshold (DWQT).
- The percentage of updated pages in a virtual buffer pool for a single data set exceeds a preset limit called the Vertical Deferred Write Threshold (VDWQT).

Up to 32 4KB or 4 32KB pages can be written in a single I/O operation.

DB2 writes data using either (1) asynchronous (deferred) writes as described above, or (2) synchronous writes.

The asynchronous write operations are implemented by DB2 subtasks, called *write engines*. The maximum number of concurrent asynchronous write I/O operations is a constant designed into DB2 (the maximum *write engines*). No more than 100 concurrent asynchronous write I/O operations can occur prior to DB2 Version 4. With DB2 Version 4, a maximum of 300 concurrent asynchronous write I/O operations can occur.

Asynchronous write operations are disabled when the maximum number of write engines are active. Synchronous writes are performed until an asynchronous write engine is available. The application is suspended during synchronous writes, and performance is adversely affected.

Additionally, overall system performance could be significantly degraded if more than 300 write operations are in process.

©Copyright 1999, Computer Management Sciences, Inc.

CPExpert compares the QBSTWEE variable in DB2STATB (the number of times a write engine was not available for asynchronous write I/O) with the **QBSTWEE** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-234 when the number of times a write engine was not available for asynchronous write I/O exceeds the value specified by the **QBSTWEE** guidance variable.

The default value for the **QBSTWEE** guidance variable is 0, indicating that CPExpert should produce Rule DB2-234 anytime a write engine (or subtask) was not available for asynchronous write I/O.

Suggestion: DB2 normally should not be constrained by the number of write engines, although IBM states (in comments related to macro DSNWMSGS) that DB2 might occasionally be constrained by the number of write engines. If Rule DB2-234 **is** produced, you should consider the following alternatives:

- **Reduce the time to perform I/O operations**. You should review the "Reducing I/O Operations to Improve Response Time" sections listed in the references. These sections give detailed guidance, depending on the version of DB2.

- Review the values of the DWQT and VDWQT parameters. These parameters control when DB2 initiates writes of updated pages in the buffer pool that are in deferred write status. The values represent a percent of the buffer pool occupied by unavailable pages (DWQT is a percent of the overall buffer pool, while VDWQT is a percent of the buffer pool for any data set). See Rule DB2-225 for more information.

  For very large buffer pools, it is unlikely that the default values of either DWQT or VDWQT will ever be reached. In this case, surges of write I/Os tend to occur as deferred writes are triggered by DB2 *checkpoints*. Since a maximum number of pages can be written with each asynchronous I/O operation (or write engine), it is possible that the number of available write engines could be exhausted.

- **Evaluate the overall performance of your system**. One basic cause of a large number of outstanding I/O operations can be the overall performance of your system. You should evaluate overall system performance to eliminate or reduce bottlenecks.

- **Reschedule workload**. You may evaluate the workload being processed when DB2 the maximum number of read engines, to determine whether non-DB2 workload or low-priority DB2 workload (such as batch updates) can be scheduled at other times. By rescheduling

work, you may improve the I/O performance, resulting in faster I/O response to individual sequential prefetch operations.

- **Evaluate the individual plans or packages**. You can evaluate selected plans or packages to improve their performance (particularly in the areas affecting I/O operations). The DB2 Application Guides can be consulted with regard to improving query and SQL performance.

- You can alter CPExpert's analysis by modifying the **QBSTWEE** guidance variable in USOURCE(DB2GUIDE). Only in unusual circumstances would you wish to alter the default of QBSTWEE=0 (in fact, if you should wish to alter this specification, please contact Computer Management Sciences so we can better understand your environment).

Reference: DB2 for OS/390 Version 3: Administration Guide
Section 7.4.1 (Reducing I/O Operations to Improve Response Time)
Section 7.6.4 (Placement of DB2 Data Sets)
QBSTWEE variable description in DSNWMSGS macro

DB2 for OS/390 Version 3: Application Guide
Section 1.1.7.3 (Performance)

DB2 for OS/390 Version 4: Administration Guide
Section 5.3.2 (Reducing the Time Needed to Perform I/O Operations)
Section 5.5.4 (Planning the Placement of DB2 Data Sets)
QBSTWEE variable description in DSNWMSGS macro

DB2 for OS/390 Version 4: Application Programming and SQL Guide
Section 6.3 (Tuning queries)

DB2 for OS/390 Version 5: Administration Guide
Section 5.3.2 (Reducing the Time Needed to Perform I/O Operations)
Section 5.5.4 (Planning the Placement of DB2 Data Sets)
Section 5.10.5.1 (Sequential Prefetch)
QBSTREE variable description in DSNWMSGS macro

DB2 for OS/390 Version 5: Application Programming and SQL Guide
Section 6.3 (Tuning your queries)

DB2 for OS/390 Version 6: Administration Guide
Section 5.3.2 (Reducing the Time Needed to Perform I/O Operations)
Section 5.5.4 (Planning the Placement of DB2 Data Sets)
Section 5.10.5.1 (Sequential Prefetch)
QBSTREE variable description in DSNWMSGS macro

Rule DB2-235: Dynamic sequential prefetch was invoked frequently

Finding: DB2 invoked dynamic sequential prefetch a large percent of total sequential prefetch, as a result of sequential detection.

Impact: This finding can have a LOW IMPACT, MEDIUM IMPACT, or HIGH IMPACT on the performance of the DB2 subsystem.

Discussion: DB2 uses three read mechanisms: normal read, sequential prefetch, and list sequential prefetch.

- **Normal Read**: Normal read is used when just one or a few consecutive pages are retrieved. The unit of transfer for a normal read is one page.

  Additionally, normal read is performed when the number of concurrent sequential prefetch I/O operations exceeds the number of DB2 sequential prefetch *read engines*. The maximum number of sequential prefetch read engines is a constant within DB2 and is set to 300. When the maximum read engines has been exhausted, DB2 disables sequential prefetch and performs asynchronous I/O operations. Please refer to Rule DB2-231 for additional information.

- **Sequential Prefetch**: Sequential prefetch is a mechanism that triggers consecutive asynchronous I/O operations. Sequential prefetch brings pages into the virtual buffer pool before they are required and reads several pages with a single I/O operation. Consequently, sequential prefetch allows CPU processing and I/O operations to be overlapped.

  Sequential prefetch can be used to read data pages, by table space scans, or index scans with clustered data reference. It can also be used to read index pages in an index scan.

- **List Sequential Prefetch**: List Sequential Prefetch is used to prefetch data pages that are not contiguous (such as through non-clustered indexes). List prefetch can also be used by incremental image copy.

DB2 normally selects sequential prefetch at bind time. If DB2 does not choose prefetch at bind time, it can sometimes use prefetch at execution time by a method called sequential detection, or dynamic sequential prefetch.

©Copyright 1999, Computer Management Sciences, Inc.

DB2 can use sequential detection for both index leaf pages and data pages. It is most commonly used on the inner table of a nested loop join, if the data is accessed sequentially.

If a table is accessed repeatedly using the same statement (for example, DELETE in a do-while loop), the data or index leaf pages of the table can be accessed sequentially. This is common in a batch processing environment. Sequential detection can then be used if access is through:

- SELECT or FETCH statements
- UPDATE and DELETE statements
- INSERT statements when existing data pages are accessed sequentially

The pattern of accessing a page is tracked when the application scans DB2 data through an index. Tracking is done to detect situations where the access pattern that develops is sequential or nearly sequential.

The most recent 8 pages are tracked. A page is considered page-sequential if it is within P/2 advancing pages of the current page, where P is the prefetch quantity. P is usually 32.

If a page is page-sequential, DB2 determines further if data access is sequential or nearly sequential. Data access is declared sequential if more than 4 out of the last 8 pages are page-sequential; this is also true for index-only access. The tracking is continuous, allowing access to slip into and out of data access sequential.

DB2 can use sequential detection if it did not choose sequential prefetch at bind time because of an inaccurate estimate of the number of pages to be accessed. This is because at bind time, the number of pages to be accessed can only be estimated. Consequently, sequential detection acts as a safety net and is employed when the data is being accessed sequentially.

Implementing dynamic sequential prefetch using sequential detection can significantly improve performance over asynchronous reads. However, sequential detection results in increased DB2 overhead. This overhead should be eliminated if possible.

Additionally, dynamic sequential prefetch is not as efficient as normal sequential prefetch selected at bind time, with respect to I/O buffer handling.

CPExpert uses the QBSTDPF variable in DB2STATB (the number of dynamic sequential prefetch requests) and the QBSTSEQ variable (the number of sequential prefetch requests, excluding dynamic sequential

prefetch requests) to compute the percent of total sequential prefetch requests that were dynamic sequential prefetch requests. To avoid spurious reporting, CPExpert ignores any interval in which dynamic sequential prefetch occurred less than 1,000 times in the interval.

CPExpert computes the percent of total sequential prefetch operations that were as a result of sequential detection. The calculation is uses data in DB2STATB, and is performed as shown below:

$$\textit{Percent dynamic sequential prefetch} = \frac{\textit{Dynamic sequential prefetch}}{\textit{Dynamic sequential prefetch} + \textit{Sequential prefetch}}$$

CPExpert produces Rule DB2-235 when the percent of requests that were dynamic sequential prefetch requests exceeds the value specified by the **PCTDPF** guidance variable.

The default value for the **PCTDPF** guidance variable is 0, indicating that CPExpert should produce Rule DB2-235 whenever dynamic sequential prefetch was implemented. **Please note that this low threshold was selected only to alert you to the performance issue. You may wish to adjust the threshold for a particular buffer pool after you considered the alternatives listed below.** Also, please recall that CPExpert will ignore any interval in which less than 1,000 dynamic sequential prefetch operations occurred.

The following example illustrates the output from Rule DB2-235:

```
RULE DB2-235: DYNAMIC SEQUENTIAL PREFETCH WAS HIGH

   Buffer Pool 3: DB2 invoked dynamic sequential prefetch a large percent
   of total sequential prefetch, as a result of DB2's sequential detection.
   Please review the discussion in the DB2 Component User Manual related
   to this finding.  You should either use the RUNSTATS utility to reduce
   the percent of dynamic detection, or use the PCTDPF guidance variable
   to alter CPExpert's analysis of dynamic sequential prefetch for this
   buffer pool. This situation occurred for Buffer Pool 3 during the
   intervals shown below:

                              --SEQUENTIAL PREFETCH OPERATIONS--   PERCENT
   MEASUREMENT INTERVAL       NORMAL        DYNAMIC        TOTAL   DYNAMIC
    0:53- 1:23, 15SEP1999      1,966          6,868        8,834     77.7
    1:23- 1:53, 15SEP1999      4,701          7,288       11,989     60.8
    4:52- 5:22, 15SEP1999      3,087          1,140        4,227     27.0
    7:22- 7:51, 15SEP1999     10,151          1,008       11,159      9.0
    8:21- 8:51, 15SEP1999      6,428          6,185       12,613     49.0
    8:51- 9:21, 15SEP1999      2,375          1,927        4,302     44.8
    9:21- 9:51, 15SEP1999      2,660         10,720       13,380     80.1
    9:51-10:21, 15SEP1999      2,578          2,339        4,917     47.6
   10:21-10:51, 15SEP1999     19,075          3,151       22,226     14.2
```

Suggestion: If Rule DB2-235 is produced more than occasionally, you should consider the following alternatives:

- Use the RUNSTATS utility so that the DB2 catalog statistics take into account the newly loaded data, and SQL paths can be selected with accurate information. Following this, rebind any application plans that depend on the loaded tables. This is necessary to update the path selection of any embedded SQL statements.

- Review the characteristics of the DB2 activity, since it is possible that DB2 will not select sequential prefetch regardless of the RUNSTATS values. For example, a single SQL SELECT statement is not usually optimized with sequential prefetch I/O at bind time. Additionally, adhoc queries would not be bound, and the DB2 optimizer would not select sequential prefetch. Dynamic sequential detection would be the most optimum approach in this environment.

- You can alter CPExpert's analysis by modifying the **QBSTDPF** guidance variable in USOURCE(DB2GUIDE).

Reference: DB2 for OS/390 Version 3: Administration Guide
Section 7.9.5.2 (Sequential Detection at Execution Time)

DB2 for OS/390 Version 4: Administration Guide
Section 5.10.5.3 (Sequential Detection at Execution Time)

DB2 for OS/390 Version 5: Administration Guide
Section 5.10.5.3 (Sequential Detection at Execution Time)

DB2 for OS/390 Version 6: Administration Guide
Section 5.10.5.3 (Sequential Detection at Execution Time)

Rule DB2-236: Synchronous read I/O was high

Finding: DB2 issued a large number of synchronous read I/O operations and there was a relatively low buffer hit ratio.

Impact: This finding can have a MEDIUM IMPACT, or HIGH IMPACT on the performance of the DB2 subsystem.

Discussion: DB2 reads pages to satisfy a sequential read request using either an asynchronous read operation (prefetching pages) or a synchronous sequential read operation (reading a single page).

- **Asynchronous Read**: DB2 can read pages using an asynchronous I/O method, using sequential prefetch of pages. Sequential prefetch brings pages into the virtual buffer pool before they are required and reads several pages with a single I/O operation. Consequently, sequential prefetch allows CPU processing and I/O operations to be overlapped.

- **Synchronous sequential Read**: Synchronous sequential reads occur when prefetch is disabled or when the requested pages are not consecutive[1]. Additionally, a synchronous sequential read is issued as the first I/O operation when prefetch starts. During a synchronous sequential read operation, the DB2 thread is waiting for the I/O to complete.

Since synchronous sequential read operations cause the DB2 thread to wait (rather than overlapping the CPU and I/O requirements of the thread), a large amount of synchronous sequential read I/O indicates performance problems. When there is a large number of synchronous sequential read operations combined with low or "negative" buffer pool hit ratios, a serious performance problem is indicated.

Goldstein (see reference) lists three likely causes of a large number of synchronous sequential read operations: (1) the processor is too busy (greater than 95% busy with DB2 and higher priority work), (2) the application using DB2 has a MVS dispatching priority that is too low, and (3) regardless of the processor activity. several large scan jobs execute concurrently.

[1] Synchronous read I/O is also used in other circumstances (for example, when reading random pages.

©Copyright 1999, Computer Management Sciences, Inc.

Since the most serious performance implications arise when there is a low or “negative” buffer hit ratio, CPExpert uses the following algorithm to calculate the buffer hit ratio:

$$\textit{Buffer hit ratio} = \frac{QBSTGET - (QBSTRIO + QBSTSPP + QBSTLPP + QBSTDPP)}{QBSTGET}$$

where:

QBSTGET = GETPAGE requests
QBSTRIO = Synchronous reads (including both random and sequential)
QBSTSPP = Pages read by normal sequential prefetch
QBSTLPP = Pages read by list prefetch
QBSTDPP = Pages read by dynamic sequential prefetch

The resulting buffer hit ratio shows the percent of GETPAGE requests that were satisfied from pages already in the buffer pool.

Note that there are situations when the resulting buffer hit ratio can be negative. A page in the buffer pool is in either of three states: in-use, updated, or available. Pages could be read into the pool via sequential prefetch operations, but until a page is “in use” or updated, the page is available for reuse by DB2. Consequently, pages that were read via sequential prefetch could be reused by GETPAGE requests or by other sequential prefetch operations before the application triggering the original prefetch could access the page. When this happens, the page would have to be read again via synchronous sequential I/O. As mentioned earlier, Goldstein lists three likely causes of a large number of synchronous sequential read operations.

CPExpert computes the percent of sequential pages read that were read by synchronous I/O operations. The calculation is uses data in DB2STATB, and is performed as shown below:

$$\textit{Percent synchronous sequential read I/Os} = \frac{\textit{Synchronous sequential read I/O count}}{\textit{Sequential page requests}}$$

CPExpert compares the computed percent synchronous sequential read I/Os with the **PCTSIO** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-236 when the percent synchronous read I/Os is **greater than** the value specified by the **PCTSIO** guidance variable , **and** the buffer hit ratio was less than 25%.

The default value for the **PCTSIO** guidance variable is 5%, indicating that Rule DB2-236 should be produced when the percent synchronous

sequential read I/Os is greater than 5%, **and** the buffer hit ratio was less than 25%.

The following example illustrates the output from Rule DB2-236:

```
RULE DB2-236: SYNCHRONOUS READ I/O WAS HIGH

   Buffer Pool 9: DB2 issued synchronous sequential read I/O for more than
   5.0% of the sequential prefetch I/O operations, and there was a relatively
   low buffer hit ratio.  This often indicates that there is a performance
   problem with the buffer pool. The problem should be considered serious
   if the buffer hit ratio is low or negative.  Please refer to the DB2
   Component User Manual for a discussion of this finding.  This situation
   occurred for Buffer Pool 9 during the intervals shown below:

                             BUFFERS   BUFFER POOL  ----SEQUENTIAL I/O----
   MEASUREMENT INTERVAL     ALLOCATED   HIT RATIO   PREFETCH   SYNCHRONOUS
   10:51-11:21, 15SEP1999     15,000       12.9      280,016     15,491
   11:50-12:20, 15SEP1999     15,000        0.2      723,968     45,550
   12:20-12:50, 15SEP1999     15,000       -1.4    2,760,496    174,418
   13:20-13:50, 15SEP1999     15,000       15.0      285,184     18,104
   14:20-14:50, 15SEP1999     15,000       22.8      284,720     17,253
```

Suggestion: If Rule DB2-236 is produced more than occasionally, you should consider the following alternatives:

- **Increase the buffer pool size**. You should consider increasing the buffer pool size until the number of read I/Os decreases, while avoiding paging between real storage and expanded storage.

  This alternative is likely to produce satisfactory results only if the buffer pool is relatively small and you can significantly increase its size. For example, satisfactory results likely would be produced if the buffer pool were 1000 buffers and you could add another 500 or 1000 buffers.

  If buffer pool is already large, a significant number of buffers would be required in order to increase the buffer pool a meaningful percent.

  Depending on the work executing on the system, significantly increasing a large buffer pool might result in an increase of overall paging of the system or of the DB2 buffer pool. This is not a desirable result. DB2 provides much better management of pages in its buffer pools than does the MVS Paging Supervisor. Thus, it is better to have smaller buffer pools (with DB2 managing the pages in the buffer pools) than it is to have buffer pools that are too large for available central storage (with the result that MVS must manage page faults).

Consequently, even though increasing the buffer pool might be the first reaction to solving the serious problem of a large percent of synchronous sequential reads, it is not often the correct solution.

- **Reduce DWQT or VDWQT values**. The DWQT and VWDQT variable thresholds control the amount of the buffer pool that is occupied by "unavailable" pages (see Rule DB2-220 for a discussion of unavailable and available pages and see Rule DB2-222 for a discussion of the DWQT and VDWQT thresholds).

  Reducing the amount of the buffer pool that is occupied by "unavailable" pages would cause more of the buffer pool to be available for pages read via sequential prefetch. If more of the buffer pool were available for sequential prefetch, it is less likely that these pages would be reused by DB2 before the application that originally triggered the prefetch would use the pages.

  Recall that "updated but not written to DASD" pages are retained in the buffer pool in anticipation that they might be referenced again in the immediate future. Consequently, reducing the DWQT or VDWQT thresholds can be considered a tradeoff of the probability that an updated page might be referenced again in the immediate future, against the certain probability that a page prefetched will be referenced in the immediate future. For most applications, the latter choice would be best.

  Of the two thresholds, the VDWQT threshold is more likely to achieve satisfactory results, particularly if the threshold is set to 0.

- **Increase the VPSEQT value**. The Sequential Steal Threshold (VPSEQT) can be used to control how much of the buffer pool can be used to support sequential I/O. This threshold is a percentage of the virtual buffer pool that might be occupied by sequentially accessed pages.

  DB2 checks the VPSEQT threshold before acquiring an "available" buffer for a sequentially accessed page. If the VPSEQT threshold **has** been exceeded, DB2 tries to acquire an "available" buffer holding a sequentially accessed page rather than one holding a randomly accessed page. On the other hand, if the VPSEQT threshold has **not** been exceeded, DB2 will try to acquire an "available" buffer holding a randomly accessed page.

  Increasing the VPSEQT threshold would allow more of the buffer pool to be used by sequential prefetch pages.

- **Reduce VPPSEQT value if parallel I/O operations are in effect**. The Virtual Buffer Pool Parallel Sequential Threshold (VPPSEQT) threshold specifies the portion of the buffer pool that might be used to support parallel operations. It is measured as a percentage of the sequential steal threshold (VPSEQT).

  If parallel I/O operations are in effect, multiple sequential prefetch I/O streams operate concurrently. These multiple I/O operations can use a significant part of the buffer pool for sequential prefetch unless they are constrained.

  Reducing the VPPSEQT value would allow parallel I/O operations to use less of the buffer pool, and thus more of the buffer pool would be available for normal sequential prefetch. Constraining the parallel I/O operations would not normally be desirable, but this action would be preferable to experiencing synchronous sequential read I/O operations!

- **Reduce VPXPSEQT if sysplex parallel I/O operations are in effect**. The Virtual Buffer Pool Assisting Parallel Sequential Threshold (VPXPSEQT) specifies how much of the be used to assist with parallel operations initiated from another DB2 in the data sharing group. It is measured as a percentage of VPPSEQT.

  If assisting parallel I/O operations are in effect, multiple sequential prefetch I/O streams, both local and assisting parallel operations, can operate concurrently. These multiple I/O operations can use a significant part of the buffer pool for assisting parallel prefetch unless they are constrained. Reducing the VPXPSEQT value would reduce the amount of the buffer pool that could be used in support of sysplex parallel assistant I/O.

  Alternatively, remove the DB2 from acting as a parallel assistant (specify ASSISTANT=NO on the DSNTIPK panel) if this DB2 is acting as an assistant. Removing this DB2 from assisting as a parallel assistant would not normally be desirable, but this action would be preferable to experiencing synchronous sequential read I/O operations!

- **Reassign data sets**. If feasible (and if there is a relatively large amount of page updates), you can Split accesses (assign random and sequential to different buffer pools)

- **Examine MVS dispatching priority for applications using DB2**. You should examine the application using DB2 to verify that the MVS dispatching priority is adequate. The basic problem is that the DB2 applications generating the sequential requests are not referencing the pages that were read via prefetch, before the pages were selected for

If you are executing in MVS Compatibility Mode, you can directly change the dispatching of the work to reflect an appropriately high dispatching priority. If you are executing in MVS Goal Mode, you might need to (1) increase the importance of the work or (2) specify a more stringent performance goal (that is, lower the response time goal or increase the execution velocity goal).

- **Examine processor utilization**. You should examine the processor utilization, for DB2 and higher priority work. If the processor utilization for this work is greater than 95%, you may be overloading the computer system. Either a rescheduling of work or acquiring a faster processor might be indicated.

- You can alter CPExpert's analysis by modifying the **PCTSIO** guidance variable in USOURCE(DB2GUIDE). You normally should not increase the PCTSIO guidance variable, however. Most DB2 performance analysts agree that this finding is serious and should not be ignored by increasing the PCTSIO guidance variable.

Reference: DB2 for OS/390 Version 3: Administration Guide
Section 7.5.1 (Tuning Database Buffer Pools)

DB2 for OS/390 Version 4: Administration Guide
Section 5.4.1 (Tuning Database Buffer Pools)

DB2 for OS/390 Version 5: Administration Guide
Section 5.4.1 (Tuning Database Buffer Pools)

DB2 for OS/390 Version 6: Administration Guide
Section 5.4.1 (Tuning Database Buffer Pools)

"DB2 Buffer Pool Tuning - Top Down or Bottom Up", Joel Goldstein, *CMG98 Conference Proceedings*, pages 51-62.

Rule DB2-237: Dynamic sequential prefetch was invoked frequently

Finding: DB2 invoked dynamic sequential prefetch a large number of times, as a result of sequential detection.

Impact: This finding can have a MEDIUM IMPACT, or HIGH IMPACT on the performance of the DB2 subsystem.

Discussion: DB2 uses three read mechanisms: normal read, sequential prefetch, and list sequential prefetch.

- **Normal Read**: Normal read is used when just one or a few consecutive pages are retrieved. The unit of transfer for a normal read is one page.

  Additionally, normal read is performed when the number of concurrent sequential prefetch I/O operations exceeds the number of DB2 sequential prefetch *read engines*. The maximum number of sequential prefetch read engines is a constant within DB2 and is set to 300. When the maximum read engines has been exhausted, DB2 disables sequential prefetch and performs asynchronous I/O operations. Please refer to Rule DB2-231 for additional information.

- **Sequential Prefetch**: Sequential prefetch is a mechanism that triggers consecutive asynchronous I/O operations. Sequential prefetch brings pages into the virtual buffer pool before they are required and reads several pages with a single I/O operation. Consequently, sequential prefetch allows CPU processing and I/O operations to be overlapped.

  Sequential prefetch can be used to read data pages, by table space scans, or index scans with clustered data reference. It can also be used to read index pages in an index scan.

- **List Sequential Prefetch**: List Sequential Prefetch is used to prefetch data pages that are not contiguous (such as through non-clustered indexes). List prefetch can also be used by incremental image copy.

DB2 normally selects sequential prefetch at bind time. If DB2 does not choose prefetch at bind time, it can sometimes use prefetch at execution time by a method called sequential detection, or dynamic sequential prefetch.

©Copyright 1999, Computer Management Sciences, Inc.

DB2 can use sequential detection for both index leaf pages and data pages. It is most commonly used on the inner table of a nested loop join, if the data is accessed sequentially.

If a table is accessed repeatedly using the same statement (for example, DELETE in a do-while loop), the data or index leaf pages of the table can be accessed sequentially. This is common in a batch processing environment. Sequential detection can then be used if access is through:

- SELECT or FETCH statements
- UPDATE and DELETE statements
- INSERT statements when existing data pages are accessed sequentially

The pattern of accessing a page is tracked when the application scans DB2 data through an index. Tracking is done to detect situations where the access pattern that develops is sequential or nearly sequential.

The most recent 8 pages are tracked. A page is considered page-sequential if it is within P/2 advancing pages of the current page, where P is the prefetch quantity. P is usually 32.

If a page is page-sequential, DB2 determines further if data access is sequential or nearly sequential. Data access is declared sequential if more than 4 out of the last 8 pages are page-sequential; this is also true for index-only access. The tracking is continuous, allowing access to slip into and out of data access sequential.

DB2 can use sequential detection if it did not choose sequential prefetch at bind time because of an inaccurate estimate of the number of pages to be accessed. This is because at bind time, the number of pages to be accessed can only be estimated. Consequently, sequential detection acts as a safety net and is employed when the data is being accessed sequentially.

Implementing dynamic sequential prefetch using sequential detection can significantly improve performance over asynchronous reads. However, sequential detection results in increased DB2 overhead. This overhead should be eliminated if possible.

CPExpert uses the QBSTDPF variable in DB2STATB (the number of dynamic sequential prefetch requests) to compute the percent of requests that were dynamic sequential prefetch requests. CPExpert compares this percent with the **PCTDPF** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-237 when the percent of requests that were

dynamic sequential prefetch requests exceeds the value specified by the **QBSTDPF** guidance variable.

The default value for the **QBSTDPF** guidance variable is 0, indicating that CPExpert should produce Rule DB2-237 whenever dynamic sequential prefetch was implemented. **Please note that this low threshold was selected only to alert you to the performance issue. You may wish to adjust the threshold for a particular buffer pool after you considered the alternatives listed below.**

The following example illustrates the output from Rule DB2-237:

Suggestion: If Rule DB2-237 is produced more than occasionally, you should consider the following alternatives:

- Use the RUNSTATS utility so that the DB2 catalog statistics take into account the newly loaded data, and SQL paths can be selected with accurate information. Following this, rebind any application plans that depend on the loaded tables to update the path selection of any embedded SQL statements.

- You can alter CPExpert's analysis by modifying the **QBSTDPF** guidance variable in USOURCE(DB2GUIDE).

Reference: DB2 for OS/390 Version 3: Administration Guide
Section 7.9.5.2 (Sequential Detection at Execution Time)

DB2 for OS/390 Version 4: Administration Guide
Section 5.10.5.3 (Sequential Detection at Execution Time)

DB2 for OS/390 Version 5: Administration Guide
Section 5.10.5.3 (Sequential Detection at Execution Time)

©Copyright 1999, Computer Management Sciences, Inc.

Rule DB2-241: Parallel groups reduced due to buffer shortage

Finding: Buffer shortage caused the number of parallel groups to be reduced.

Impact: This finding can have a MEDIUM IMPACT, or HIGH IMPACT on the performance of the DB2 subsystem.

Discussion: DB2 can initiate multiple parallel operations to speed the response time of a query. The type of parallel operations differs depending on the version of DB2 that is installed.

- With DB2 Version 3, when DB2 plans to use sequential prefetch to access data from a table in a partitioned table space, it can initiate parallel I/O operations. *Query I/O parallelism* manages concurrent I/O requests for a single query, fetching pages into the buffer pool in parallel. For queries whose response time is dominated by I/O operations, the response time can be significantly reduced.

- With DB2 Version 4, parallel operation was expanded to enable true multi-tasking within a query, by including *Query CP parallelism*. With Query CP parallelism, a large query can be broken into multiple smaller queries. These smaller queries run simultaneously on multiple processors accessing data in parallel. The response time for I/O-intensive or processor-intensive queries can be significantly reduced.

- With DB2 Version 5, parallel operation was expanded by adding *Sysplex query parallelism* to take advantage of the sysplex processing capacity. With Sysplex query parallelism, DB2 can split a large query across different DB2 members in a data sharing group. Sysplex query parallelism is similar to Query CP Parallelism, with the difference being that DB2 can spread the queries across multiple systems in the sysplex.

A *parallel task* is an execution unit that is dynamically created to process a query in parallel. It is implemented by an MVS service request block.

A *parallel group* is a set of consecutive operations, executed in parallel, that have the same number of parallel tasks.

The *degree of parallelism* is the number of parallel tasks or I/O operations that DB2 determines can be used for the operations on the parallel group.

©Copyright 1999, Computer Management Sciences, Inc.

Queries can only take advantage of parallelism if you enable parallel processing[1]. To enable parallel processing:

- For static SQL, specify DEGREE(ANY) on BIND or REBIND. This bind option affects static SQL only and does not enable parallelism for dynamic statements.

- For dynamic SQL, set the CURRENT DEGREE special register to 'ANY'. Setting the special register affects dynamic statements only. It will have no effect on static SQL statements

- The virtual buffer pool parallel sequential threshold (VPPSEQT) value must be large enough to provide adequate buffer pool space for parallel processing.

- If you bind with isolation CS, choose also the option CURRENTDATA(NO), if possible. This option can improve performance in general, but it also ensures that DB2 will consider parallelism for ambiguous cursors. If you bind with CURRENDATA(YES) and DB2 cannot tell if the cursor is read-only, DB2 does not consider parallelism.

The number of parallel operations or tasks used to access data is initially determined at bind time, and the number is adjusted when the query is executed.

For each parallel group, parallelism (either CP or I/O) can execute at a reduced degree or degrade to sequential operations. One cause[2] of reduced degree of parallelism (or degradation to sequential operations) is the amount of virtual buffer pool space available.

CPExpert compares the QBSTPQF variable in DB2STATB (the number of parallel groups processed at a degree less than planned) with the **QBSTPQF** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-241 when the number of parallel groups processed at a degree less than planned exceeds the value specified by the **QBSTPQF** guidance variable.

The default value for the **QBSTPQF** guidance variable is 0, indicating that CPExpert should produce Rule DB2-241 when any parallel groups were processed at a degree less than planned.

The following example illustrates the output from Rule DB2-241:

---

[1] Parallel processing can not be used with all queries. The DB2 Administration Guide (beginning with DB2 Version 4) lists situations when parallel processing is not used.

[2] Other causes include host variable values, availability of the MVS/ESA sort, and ambiguous cursors.

```
RULE DB2-241: PARALLEL GROUPS REDUCED DUE TO BUFFER SHORTAGE

   Buffer Pool 7: DB2 could not allocate the requested number of buffer
   pages to allow a parallel group to run to the planned degree. This
   is caused by a storage shortage in the buffer pool.  This finding
   is applicable only for non-work file page sets and when queries are
   processing in parallel in DB2. This situation occurred for Buffer
   Pool 7 during the intervals shown below:

                                BUFFERS    TIMES BUFFERS   GETPAGE REQUESTS
   MEASUREMENT INTERVAL       ALLOCATED    UNAVAILABLE    TOTAL    PCT SEQ
   11:59-12:29, 14OCT1999       2,000            39     795,728       92.6
```

Suggestion: If Rule DB2-241 is regularly produced, you should consider the following alternatives:

- Use the ALTER BUFFERPOOL command to increase the amount of buffer space available for parallel operations, by modifying the following parameters:

  - VPSIZE (the size of the virtual buffer pool). This parameter is used to increase the size of the virtual pool. Before increasing the size of the buffer pool, you should first make sure that the following two parameters are properly set for the DB2 work being processed.

  - VPSEQT (the sequential steal threshold) specifies the percentage of the virtual buffer pool that can be occupied by sequentially accessed pages (as distinguished from those pages that are randomly accessed). When DB2 needs a page in the buffer pool, it first checks the percent of the buffer pool that is occupied by sequentially accessed pages. DB2 will steal sequentially accessed pages if more than the VPSEQT percent of the buffer pool is occupied by sequentially accessed pages. Otherwise, DB2 will steal a randomly accessed page.

    For example, suppose that the VPSIZE had been specified as 2000 pages. If the default VPSEQT of 80% were used, DB2 would begin stealing sequentially accessed pages when more than 1600 pages (2000 * .8 = 1600) were sequentially accessed. Otherwise, DB2 would steal randomly accessed pages (the remaining 400 pages in this example).

  - VPPSEQT (the parallel sequential threshold) specifies the percentage of the sequential steal threshold (VPSEQT) that can be used to support parallel I/O operations.

Continuing with the above example, suppose that the VPPSEQT default of 50% were used. DB2 would not initiate parallel I/O operations once more than 800 pages (1600 * .5 = 800) were used to support parallel I/O operations.

- VPXPSEQT (the assisting parallel sequential threshold). This parameter is used only for sysplex query parallelism. The VPXPSEQT parameter is included in this discussion only because it was in the relevant section of the DB2 Administration Guides. Actually, DB2 skips an assistant DB2 if the VPXPSEQT threshold has been reached (see Rule DB2-641).

From the above discussion, if should be clear that the ratio of randomly accessed pages in the buffer pool to sequentially accessed pages in the buffer pool would be the determining factor in selecting values for these parameters. The VPSEQT threshold would be lowered if there are a relatively large number of randomly accessed pages. The VPSEQT threshold should be high[3] (set to 99, for example) if there are few randomly accessed pages.

The information provided with Rule DB2-241 includes the total GETPAGE requests and the percent of this total that were sequentially accessed pages[4]. This information can be used to provide a general assessment of the random versus sequential nature of the buffer pool.

WARNING: The percent of GETPAGE requests that were sequential can be used only as a general view of the random versus sequential nature of "unavailable" pages in the buffer pool. This distribution of randomly accessed pages versus sequentially accessed pages directly relates to the "in-use" pages in the buffer pool, as these "in-use" pages are a result of GETPAGE requests. However, "unavailable" pages consist of both "in-use" pages and "updated but not written to DASD" pages[5].

You must keep in mind that the "updated but not written to DASD" pages in the buffer pool could consist entirely of randomly accessed pages, could consist entirely of sequentially accessed pages, or could consist of any combination of the two types of pages. Unfortunately, there is no measure available in the DB2 interval statistics that shows the distribution of pages in the buffer pool, by page access type.

---

[3] Rule DB2-223 will be produced if CPExpert believes that the VPSEQT value is too low.

[4] For optimal buffer pool performance, you should separate randomly-accessed objects and sequentially-accessed objects into different buffer pools. This optimal separation may not be possible, of course, depending on your environment.

[5] Please refer to Rule DB2-220 for a discussion of "in-use" pages, "updated but not written to DASD" pages, and "available" pages.

Consequently, a “best guess” must be made based on the distribution of GETPAGE requests.

One way to limit the effect of the uncertain random versus sequential nature of the “updated but not written to DASD” pages is to specify a relatively low value for both the DWQT and VDWQT thresholds. These thresholds control when DB2 will write the updated pages to DASD[6]. By specifying low values for these thresholds, DB2 will write updated pages to DASD before these pages account for a significant percent of the buffer pool. With only a few “updated but not written to DASD” pages in the buffer pool, the distribution of these pages into randomly accessed or sequentially accessed would have little effect on your decisions regarding the value for the VPSEQT parameter. You then can use the random versus sequential nature of the GETPAGE requests to guide your decision about the VPSEQT threshold value.

Once you have specified an appropriate value for the VPSEQT threshold, you can decide what percentage of the pages reserved for sequentially accessed pages you wish to be used in support of parallel I/O operations. This decision will guide your value for the VPPSEQT threshold. When making this decision, you should keep in mind that parallel I/O operations can **significantly improve** DB2 performance.

- It is quite likely that you should override the default value of 50% for the VPPSEQT parameter and specify 100% as the value for VPPSEQT. This action should be taken, of course, only after you have verified that the work being processed with parallel I/O is important relative to other DB2 work.

- Alternatively, you can use an iterative approach to setting the VPPSEQT value, by incrementally raising the value until CPExpert stops producing Rule DB2-241[7].

- Use the ALTER TABLESPACE statement to assign table spaces accessed by this query to a different buffer pool.

- Use the utility RUNSTATS to keep DB2 statistics current. This will help DB2 determine the optimal degree of parallelism

- You can alter CPExpert’s analysis by modifying the **QBSTPQF** guidance variable in USOURCE(DB2GUIDE).

[6] Please refer to Rule DB2-222 for a discussion of the DWQT and VDWQT thresholds.

[7] Of course, you might never reach the point at which Rule DB2-241 is not produced, as the buffer pool might be too small and the VPSIZE might need to be increased.

Reference: DB2 for OS/390 Version 3: Administration Guide
Section 7.9.6 (Parallel I/O Operations and Query Performance)

DB2 for OS/390 Version 4: Administration Guide
Section 5.10.8 (Parallel Operations and Query Performance)

DB2 for OS/390 Version 5: Administration guide
Section 5.10.8 (Parallel Operations and Query Performance)

DB2 for OS/390 Version 6: Administration guide
Section 5.11 (Parallel Operations and Query Performance)

Rule DB2-242: Prefetch quantity was reduced to one-half of normal

Finding: DB2 reduced the prefetch quantity to one-half of normal because of a buffer shortage.

Impact: This finding can have a MEDIUM IMPACT, or HIGH IMPACT on the performance of the DB2 subsystem.

Discussion: DB2 uses three mechanisms for reading pages: normal read, sequential prefetch, and list sequential prefetch.

- **Normal Read**: DB2 uses normal read when just one or a few consecutive pages are retrieved. The unit of transfer for a normal read is one page.

- **Sequential Prefetch**: DB2 performs sequential prefetch concurrently with other operations of the originating application program. With sequential prefetch, DB2 brings pages into the virtual buffer pool before they are required and reads several pages with a single I/O operation.

  Sequential prefetch brings pages into the virtual buffer pool before they are required and reads several pages with a single I/O operation. Consequently, sequential prefetch allows CPU processing and I/O operations to be overlapped.

- **List Sequential Prefetch**: DB2 uses list sequential prefetch to prefetch data pages that are not contiguous (such as through non-clustered indexes). List prefetch can also be used by incremental image copy.

DB2 normally selects sequential prefetch at bind time. If DB2 does not choose prefetch at bind time, it can sometimes use prefetch at execution time by a method called sequential detection, or dynamic sequential prefetch.

Since prefetch reads a set of pages into the buffer pool with only one asynchronous I/O operation, prefetch can allow substantial savings in both processor cycles and I/O costs.

The maximum number of pages read by sequential prefetch is determined by the size of the buffer pool used. Prior to DB2 Version 6, buffer pools can be composed of either 4K buffers or 32K buffers[1]. The following shows the pages read by a sequential prefetch for an application:

[1] With DB2 Version 6, buffer pools can also be composed of 8K and 16K buffers.

©Copyright 1999, Computer Management Sciences, Inc.

- **4K buffer pools**

Buffer Pool Size	Pages Read by Prefetch
< 224 buffers	8 pages for each asynchronous I/O
224-999 buffers	16 pages for each asynchronous I/O
1000+ buffers	32 pages for each asynchronous I/O

- **32K buffer pools**

Buffer Pool Size	Pages Read by Prefetch
< 17 buffers	0 pages for each asynchronous I/O
17-99 buffers	2 pages for each asynchronous I/O
100+ buffers	4 pages for each asynchronous I/O

As a separate operations issue, DB2 can initiate multiple parallel operations to speed the response time of a query. The type of parallel operations differs depending on the version of DB2 that is installed. Please refer to Rule DB2-241 for a discussion of parallel operations.

Depending on buffer pool availability at run time, DB2 could reduce the degree of parallelism or revert to a sequential plan for a given parallel group, or *DB2 can reduce the prefetch quantity*.

There are three levels that DB2 can reduce the prefetch quantity: (1) the prefetch quantity can be reduced to one-half of normal, (2) the prefetch quantity can be reduced to one-quarter of normal, or (3) the prefetch quantity can be reduced to zero (that is, prefetch I/O streams can be denied). Each level has an increasingly significant impact on performance.

This rule (Rule DB2-242) analyzes the situation in which the prefetch quantity was reduced to one-half of normal. Other rules (Rule DB2-243 and Rule DB2-244) analyze the other two situations.

CPExpert compares the QBSTPL1 variable in DB2STATB (the number of times the prefetch quantity was reduced to one-half of normal) with the **QBSTPL1** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-242 when the number of times the prefetch quantity

was reduced to one-half of normal exceeds the value specified by the **QBSTPL1** guidance variable.

The default value for the **QBSTPL1** guidance variable is 0, indicating that CPExpert should produce Rule DB2-242 when the prefetch quantity was reduced to one-half of normal.

The following example illustrates the output from Rule DB2-242:

```
RULE DB2-242: PREFETCH QUANTITY REDUCED TO ONE-HALF OF NORMAL

   Buffer Pool 7: DB2 had to reduce sequential prefetch quantity to 50%
   of normal to continue to execute concurrently with parallel query
   processing in the system.  This reduction degrades performance, but may
   be acceptable if it occurs infrequently.  This finding is applicable
   only when queries are processing in parallel in DB2.  This situation
   occurred for Buffer Pool 7 during the intervals shown below:

                                 BUFFERS        TIMES PREFETCH QUANTITY
   MEASUREMENT INTERVAL         ALLOCATED      REDUCED TO ONE-HALF NORMAL
   11:59-12:29, 14OCT1999         2,000                 33,781
```

Suggestion: If Rule DB2-242 is regularly produced, you should consider the following alternatives:

- Increase the buffer pool space available for parallel I/O operations by modifying the following parameters:

  - VPSIZE (the size of the virtual buffer pool).

  - VPSEQT (the sequential steal threshold).

  - VPPSEQT (the parallel sequential threshold).

  Please refer to Rule DB2-241 for a discussion of the above parameters and guidance about selecting proper values.

- Review the settings of the Deferred Write Threshold (DWQT) and Vertical Deferred Write Threshold (VDWQT). Large values for the DWQT and VDWQT thresholds allows updated pages to use a larger portion of the virtual buffer pool. Consequently, setting DWQT and VDWQT to large values can have a significant effect on the other thresholds.

  For example, for a workload in which pages are frequently updated, and the set of pages updated exceeds the size of the virtual buffer pool,

setting both DWQT and VDWQT to 90% would probably cause the sequential prefetch threshold (and possibly the data management threshold and the immediate write threshold) to be reached frequently.

- Modify application programs so that they do more COMMITs. This will free pages in the buffer pool.

- You can alter CPExpert's analysis by modifying the **QBSTPL1** guidance variable in USOURCE(DB2GUIDE).

Reference: DB2 for OS/390 Version 3: Administration Guide
Section 7.9.6 (Parallel I/O Operations and Query Performance)

DB2 for OS/390 Version 4: Administration Guide
Section 5.10.8 (Parallel Operations and Query Performance)

DB2 for OS/390 Version 5: Administration guide
Section 5.10.8 (Parallel Operations and Query Performance)

DB2 for OS/390 Version 6: Administration guide
Section 5.11 (Parallel Operations and Query Performance)

Rule DB2-243: Prefetch quantity was reduced to one-quarter of normal

Finding: DB2 reduced the prefetch quantity to one-quarter of normal because of a buffer shortage.

Impact: This finding can have a MEDIUM IMPACT, or HIGH IMPACT on the performance of the DB2 subsystem.

Discussion: DB2 uses three mechanisms for reading pages: normal read, sequential prefetch, and list sequential prefetch.

- **Normal Read**: DB2 uses normal read when just one or a few consecutive pages are retrieved. The unit of transfer for a normal read is one page.

- **Sequential Prefetch**: DB2 performs sequential prefetch concurrently with other operations of the originating application program. With sequential prefetch, DB2 brings pages into the virtual buffer pool before they are required and reads several pages with a single I/O operation.

  Sequential prefetch brings pages into the virtual buffer pool before they are required and reads several pages with a single I/O operation. Consequently, sequential prefetch allows CPU processing and I/O operations to be overlapped.

- **List Sequential Prefetch**: DB2 uses list sequential prefetch to prefetch data pages that are not contiguous (such as through non-clustered indexes). List prefetch can also be used by incremental image copy.

DB2 normally selects sequential prefetch at bind time. If DB2 does not choose prefetch at bind time, it can sometimes use prefetch at execution time by a method called sequential detection, or dynamic sequential prefetch.

Since prefetch reads a set of pages into the buffer pool with only one asynchronous I/O operation, prefetch can allow substantial savings in both processor cycles and I/O costs.

The maximum number of pages read by sequential prefetch is determined by the size of the buffer pool used. Prior to DB2 Version 6, buffer pools can be composed of either 4K buffers or 32K buffers[1]. The following shows the pages read by a sequential prefetch for an application:

[1] With DB2 Version 6, buffer pools can also be composed of 8K and 16K buffers.

©Copyright 1999, Computer Management Sciences, Inc.

- **4K buffer pools**

Buffer Pool Size	Pages Read by Prefetch
< 224 buffers	8 pages for each asynchronous I/O
224-999 buffers	16 pages for each asynchronous I/O
1000+ buffers	32 pages for each asynchronous I/O

- **32K buffer pools**

Buffer Pool Size	Pages Read by Prefetch
< 17 buffers	0 pages for each asynchronous I/O
17-99 buffers	2 pages for each asynchronous I/O
100+ buffers	4 pages for each asynchronous I/O

As a separate operations issue, DB2 can initiate multiple parallel operations to speed the response time of a query. The type of parallel operations differs depending on the version of DB2 that is installed. Please refer to Rule DB2-241 for a discussion of parallel operations.

Depending on buffer pool availability at run time, DB2 could reduce the degree of parallelism or revert to a sequential plan for a given parallel group, or *DB2 can reduce the prefetch quantity*.

There are three levels that DB2 can reduce the prefetch quantity: (1) the prefetch quantity can be reduced to one-half of normal, (2) the prefetch quantity can be reduced to one-quarter of normal, or (3) the prefetch quantity can be reduced to zero (that is, prefetch I/O streams can be denied). Each level has an increasingly significant impact on performance.

This rule (Rule DB2-243) analyzes the situation in which the prefetch quantity was reduced to one-quarter of normal. Other rules (Rule DB2-242 and Rule DB2-244) analyze the other two situations.

CPExpert compares the QBSTPL2 variable in DB2STATB (the number of times the prefetch quantity was reduced to one-quarter of normal) with the **QBSTPL2** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-243 when the number of times the prefetch quantity

was reduced to one-quarter of normal exceeds the value specified by the **QBSTPL2** guidance variable.

The default value for the **QBSTPL2** guidance variable is 0, indicating that CPExpert should produce Rule DB2-243 when the prefetch quantity was reduced to one-quarter of normal.

The following example illustrates the output from Rule DB2-243:

```
RULE DB2-243: PREFETCH QUANTITY REDUCED TO ONE-QUARTER OF NORMAL

   Buffer Pool 7: DB2 had to reduce sequential prefetch quantity to 25%
   of normal to continue to execute concurrently with parallel query
   processing in the system.  The query response for queries processed in
   parallel can be significantly worse when this action is taken.  This
   finding is applicable only when queries are processing in parallel in
   DB2.  This situation occurred for Buffer Pool 7 during the intervals
   shown below:

                                BUFFERS       TIMES PREFETCH QUANTITY
   MEASUREMENT INTERVAL        ALLOCATED   REDUCED TO ONE-QUARTER NORMAL
   11:59-12:29, 14OCT1999        2,000               20,644
```

Suggestion: If Rule DB2-243 is regularly produced, you should consider the following alternatives:

- Increase the buffer pool space available for parallel I/O operations by modifying the following parameters:

  - VPSIZE (the size of the virtual buffer pool).

  - VPSEQT (the sequential steal threshold).

  - VPPSEQT (the parallel sequential threshold).

  Please refer to Rule DB2-241 for a discussion of the above parameters and guidance about selecting proper values.

- Review the settings of the Deferred Write Threshold (DWQT) and Vertical Deferred Write Threshold (VDWQT). Large values for the DWQT and VDWQT thresholds allow updated pages to use a larger portion of the virtual buffer pool. Consequently, setting DWQT and VDWQT to large values can have a significant effect on the other thresholds.

  For example, for a workload in which pages are frequently updated, and the set of pages updated exceeds the size of the virtual buffer pool,

setting both DWQT and VDWQT to 90% would probably cause the sequential prefetch threshold (and possibly the data management threshold and the immediate write threshold) to be reached frequently.

- Modify application programs so that they do more COMMITs. This will free pages in the buffer pool.

- You can alter CPExpert's analysis by modifying the **QBSTPL2** guidance variable in USOURCE(DB2GUIDE).

Reference: DB2 for OS/390 Version 3: Administration Guide
Section 7.9.6 (Parallel I/O Operations and Query Performance)

DB2 for OS/390 Version 4: Administration Guide
Section 5.10.8 (Parallel Operations and Query Performance)

DB2 for OS/390 Version 5: Administration guide
Section 5.10.8 (Parallel Operations and Query Performance)

DB2 for OS/390 Version 6: Administration guide
Section 5.11 (Parallel Operations and Query Performance)

Rule DB2-244: Prefetch I/O streams were denied, shortage of buffers

Finding: Prefetch I/O streams were denied because of a buffer shortage.

Impact: This finding can have a MEDIUM IMPACT, or HIGH IMPACT on the performance of the DB2 subsystem.

Discussion: DB2 uses three mechanisms for reading pages: normal read, sequential prefetch, and list sequential prefetch.

- **Normal Read**: DB2 uses normal read when just one or a few consecutive pages are retrieved. The unit of transfer for a normal read is one page.

- **Sequential Prefetch**: DB2 performs sequential prefetch concurrently with other operations of the originating application program. With sequential prefetch, DB2 brings pages into the virtual buffer pool before they are required and reads several pages with a single I/O operation.

  Sequential prefetch brings pages into the virtual buffer pool before they are required and reads several pages with a single I/O operation. Consequently, sequential prefetch allows CPU processing and I/O operations to be overlapped.

- **List Sequential Prefetch**: DB2 uses list sequential prefetch to prefetch data pages that are not contiguous (such as through non-clustered indexes). List prefetch can also be used by incremental image copy.

DB2 normally selects sequential prefetch at bind time. If DB2 does not choose prefetch at bind time, it can sometimes use prefetch at execution time by a method called sequential detection, or dynamic sequential prefetch.

Since prefetch reads a set of pages into the buffer pool with only one asynchronous I/O operation, prefetch can allow substantial savings in both processor cycles and I/O costs.

The maximum number of pages read by sequential prefetch is determined by the size of the buffer pool used. Prior to DB2 Version 6, buffer pools can be composed of either 4K buffers or 32K buffers[1]. The following shows the pages read by a sequential prefetch for an application:

[1] With DB2 Version 6, buffer pools can also be composed of 8K and 16K buffers.

©Copyright 1999, Computer Management Sciences, Inc.

- **4K buffer pools**

Buffer Pool Size	Pages Read by Prefetch
< 224 buffers	8 pages for each asynchronous I/O
224-999 buffers	16 pages for each asynchronous I/O
1000+ buffers	32 pages for each asynchronous I/O

- **32K buffer pools**

Buffer Pool Size	Pages Read by Prefetch
< 17 buffers	0 pages for each asynchronous I/O
17-99 buffers	2 pages for each asynchronous I/O
100+ buffers	4 pages for each asynchronous I/O

As a separate operations issue, DB2 can initiate multiple parallel operations to speed the response time of a query. The type of parallel operations differs depending on the version of DB2 that is installed. Please refer to Rule DB2-241 for a discussion of parallel operations.

Depending on buffer pool availability at run time, DB2 could reduce the degree of parallelism or revert to a sequential plan for a given parallel group, or *DB2 can reduce the prefetch quantity*.

There are three levels that DB2 can reduce the prefetch quantity: (1) the prefetch quantity can be reduced to one-half of normal, (2) the prefetch quantity can be reduced to one-quarter of normal, or (3) the prefetch quantity can be reduced to zero (that is, prefetch I/O streams can be denied). Each level has an increasingly significant impact on performance.

This rule (Rule DB2-244) analyzes the situation in which the prefetch quantity was reduced to zero. Other rules (Rule DB2-242 and Rule DB2-243) analyze the other two situations.

CPExpert compares the QBSTJIS variable in DB2STATB (the number of times prefetch I/O streams were denied) with the **QBSTJIS** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-244 when the number of times prefetch I/O streams were denied exceeds the value specified by the **QBSTJIS** guidance variable.

The default value for the **QBSTJIS** guidance variable is 0, indicating that CPExpert should produce Rule DB2-244 when prefetch I/O streams were denied.

The following example illustrates the output from Rule DB2-244:

```
RULE DB2-244: PREFETCH I/O STREAMS WERE DENIED, SHORTAGE OF BUFFERS

   Buffer Pool 7: Requested prefetch I/O streams were denied because of a
   storage shortage in the buffer pool.  When this finding is made, you
   should consider increasing the size of the buffer pool. This finding
   is applicable only for non-work file page sets and when queries are
   processing in parallel in DB2.  This situation occurred for Buffer
   Pool 7 during the intervals shown below:

                                 BUFFERS          NUMBER OF PREFETCH I/O
   MEASUREMENT INTERVAL         ALLOCATED        STREAMS THAT WERE DENIED
   11:59-12:29, 14OCT1999         2,000                     272
```

Suggestion: If Rule DB2-244 is regularly produced, you should consider the following alternatives:

- Increase the buffer pool space available for parallel I/O operations by modifying the following parameters:

  - VPSIZE (the size of the virtual buffer pool).

  - VPSEQT (the sequential steal threshold).

  - VPPSEQT (the parallel sequential threshold).

  Please refer to Rule DB2-241 for a discussion of the above parameters and guidance about selecting proper values.

- Review the settings of the Deferred Write Threshold (DWQT) and Vertical Deferred Write Threshold (VDWQT). Large values for the DWQT and VDWQT thresholds allows updated pages to use a larger portion of the virtual buffer pool. Consequently, setting DWQT and VDWQT to large values can have a significant effect on the other thresholds.

  For example, for a workload in which pages are frequently updated, and the set of pages updated exceeds the size of the virtual buffer pool, setting both DWQT and VDWQT to 90% would probably cause the sequential prefetch threshold (and possibly the data management threshold and the immediate write threshold) to be reached frequently.

- Modify application programs so that they do more COMMITs. This will free pages in the buffer pool.

- You can alter CPExpert's analysis by modifying the **QBSTJIS** guidance variable in USOURCE(DB2GUIDE).

Reference: DB2 for OS/390 Version 3: Administration Guide
Section 7.9.6 (Parallel I/O Operations and Query Performance)

DB2 for OS/390 Version 4: Administration Guide
Section 5.10.8 (Parallel Operations and Query Performance)

DB2 for OS/390 Version 5: Administration guide
Section 5.10.8 (Parallel Operations and Query Performance)

DB2 for OS/390 Version 6: Administration guide
Section 5.11 (Parallel Operations and Query Performance)

©Copyright 1999, Computer Management Sciences, Inc.

Rule DB2-245: Page requested for a parallel query was unavailable

Finding: A page requested for a parallel query was unavailable and asynchronous prefetch I/O was triggered.

Impact: This finding can have a MEDIUM IMPACT, or HIGH IMPACT on the performance of the DB2 subsystem.

Discussion: When DB2 plans to use sequential prefetch to access data from a table in a partitioned table space, it can initiate parallel I/O operations. *Query I/O parallelism* manages concurrent I/O requests for a single query, fetching pages into the buffer pool in parallel. For queries whose response time is dominated by I/O operations, the response time can be significantly reduced.

A page can be requested for a query processed in parallel, but the page might not be available because an I/O was in progress or the page was not found in the buffer pool. In this situation, the agent does not wait. Control returns to the agent and an asynchronous prefetch I/O is triggered.

If this situation rarely occurs, most pages are already prefetched into the buffer pool and there is little performance degradation waiting for I/O operations (wait time for synchronous I/O is small). This situation can occur often, for example, if there is a cluster index scan and the data is not truly clustered by the index key. In this case, the datapages are not accessed in their true order, and the cluster ratio is not valid.

CPExpert compares the QBSTNGT variable in DB2STATS (the number of times a page requested for a parallel query was unavailable) with the **QBSTNGT** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-245 when the number of times a page requested for a parallel query was unavailable exceeds the value specified by the **QBSTNGT** guidance variable.

The default value for the **QBSTNGT** guidance variable is 0, indicating that CPExpert should produce Rule DB2-245 whenever a page requested for a parallel query was unavailable

The following example illustrates the output from Rule DB2-245:

©Copyright 1999, Computer Management Sciences, Inc.

```
RULE DB2-245: PAGE REQUESTED FOR A PARALLEL QUERY WAS UNAVAILABLE

   Buffer Pool 4: The Buffer Statistics reported that a page was requested
   for a query processed in parallel, but the page was unavailable because
   an I/O was in progress or the page was not found in the buffer pool.
   The agent does not wait for the page, but control is returned to the
   agent and an asynchronous prefetch I/O is triggered.  This situation
   can occur often if, for example, there is a cluster index scan and the
   data is not truly clustered by the index key, and the cluster ratio is
   not valid.  You should consider using the RUNSTATS utility to update
   the cluster ration. This situation occurred for Buffer Pool 4 during
   the intervals shown below:

                                     NUMBER OF              PAGES
   MEASUREMENT INTERVAL          GETPAGE REQUESTS        UNAVAILABLE
   11:59-12:29, 14OCT1999              2,000                  554
```

Suggestion: If Rule DB2-245 is regularly produced, you should consider the following alternatives:

- Use the utility RUNSTATS to keep DB2 statistics current.
- You can alter CPExpert's analysis by modifying the **QBSTNGT** guidance variable in USOURCE(DB2GUIDE).

Reference: DB2 for OS/390 Version 3: Administration Guide
Section 7.9.6 (Parallel I/O Operations and Query Performance)

DB2 for OS/390 Version 4: Administration Guide
Section 5.10.8 (Parallel Operations and Query Performance)

DB2 for OS/390 Version 5: Administration guide
Section 5.10.8 (Parallel Operations and Query Performance)

DB2 for OS/390 Version 6: Administration guide
Section 5.11 (Parallel Operations and Query Performance)

Rule DB2-246: Parallel groups fell back to sequential mode

Finding: Parallel groups fell back to sequential mode for the threads identified from the DB2 accounting statistics.

Impact: This finding can have a MEDIUM IMPACT, or HIGH IMPACT on the performance of the DB2 threads identified.

Discussion: DB2 can initiate multiple parallel operations to speed the response time of a query. The type of parallel operations differs depending on the version of DB2 that is installed.

- With DB2 Version 3, when DB2 plans to use sequential prefetch to access data from a table in a partitioned table space, it can initiate parallel I/O operations. *Query I/O parallelism* manages concurrent I/O requests for a single query, fetching pages into the buffer pool in parallel. For queries whose response time is dominated by I/O operations, the response time can be significantly reduced.

- With DB2 Version 4, parallel operation was expanded to enable true multi-tasking within a query, by including *Query CP parallelism*. With Query CP parallelism, a large query can be broken into multiple smaller queries. These smaller queries run simultaneously on multiple processors accessing data in parallel. The response time for I/O-intensive or processor-intensive queries can be significantly reduced.

- With DB2 Version 5, parallel operation was expanded by adding *Sysplex query parallelism* to take advantage of the sysplex processing capacity. With Sysplex query parallelism, DB2 can split a large query across different DB2 members in a data sharing group. Sysplex query parallelism is similar to Query CP Parallelism, with the difference being that DB2 can spread the queries across multiple systems in the sysplex.

With Sysplex query parallelism, one DB2 subsystem acts as a *Coordinator* while other DB2 subsystems on other systems in the sysplex act as *Assistants*. The DB2 acting as a Coordinator is designated on the DSNTIP When

CPExpert compares the QZZCSKIP variable in DB2STATS (the number of times DB2 skipped an assistant DB2 because of buffer shortage) with the **QZZCSKIP** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-246 when the number of times DB2 skipped an

assistant DB2 because of buffer shortage exceeds the value specified by the **QZZCSKIP** guidance variable.

The default value for the **QZZCSKIP** guidance variable is 0, indicating that CPExpert should produce Rule DB2-246 whenever DB2 skipped an assistant DB2 because of buffer shortage.

The following example illustrates the output from Rule DB2-246:

```
RULE DB2-246: PARALLEL GROUPS REDUCED DUE TO BUFFER SHORTAGE

   Buffer Pool 7: DB2 could not allocate the requested number of buffer
   pages to allow a parallel group to run to the planned degree. This is
   caused by a storage shortage in the buffer pool. If this value is
   significant, increase the size of the buffer pool.  This finding is
   applicable only for non-work file page sets and when queries are
   processing in parallel in DB2. This situation occurred for Buffer
   Pool 7 during the intervals shown below:

                                BUFFERS ALLOCATED      TIMES REQUESTED
   MEASUREMENT INTERVAL          TO BUFFER POOL      BUFFERS UNAVAILABLE
   11:59-12:29, 14OCT1999            2,000                    39
```

Suggestion: If Rule DB2-246 is regularly produced, you should consider the following alternatives:

- Increase the buffer pool space available for parallel I/O operations by modifying the following parameters:

  - VPSIZE (the size of the virtual buffer pool).

  - VPSEQT (the sequential steal threshold).

  - VPPSEQT (the parallel sequential threshold).

  - VPXPSEQT (the assisting parallel sequential threshold) This parameter is used only for Sysplex query parallelism.

  Please refer to Rule DB2-241 for a discussion of the above parameters and guidance about selecting proper values.

- Increase the size of the buffer pool. .

- Review the settings of the Deferred Write Threshold (DWQT) and Vertical Deferred Write Threshold (VDWQT). Large values for the DWQT and VDWQT thresholds allows updated pages to use a larger portion of the virtual buffer pool. Consequently, setting DWQT and

VDWQT to large values can have a significant effect on the other thresholds.

For example, for a workload in which pages are frequently updated, and the set of pages updated exceeds the size of the virtual buffer pool, setting both DWQT and VDWQT to 90% would probably cause the sequential prefetch threshold (and possibly the data management threshold and the immediate write threshold) to be reached frequently.

- Modify application programs so that they do more COMMITs. This will free pages in the buffer pool.

- Use the ALTER TABLESPACE statement to assign table spaces accessed by this query to a different buffer pool.

- Use the utility RUNSTATS to keep DB2 statistics current. This will help DB2 determine the optimal degree of parallelism

- You can alter CPExpert's analysis by modifying the **QBSTPQF** guidance variable in USOURCE(DB2GUIDE).

Reference: DB2 for OS/390 Version 3: Administration Guide
Section 7.9.6 (Parallel I/O Operations and Query Performance)

DB2 for OS/390 Version 4: Administration Guide
Section 5.10.8 (Parallel Operations and Query Performance)

DB2 for OS/390 Version 5: Administration guide
Section 5.10.8 (Parallel Operations and Query Performance)

DB2 for OS/390 Version 6: Administration guide
Section 5.11 (Parallel Operations and Query Performance)

©Copyright 1999, Computer Management Sciences, Inc.

Rule DB2-261: Failures were caused by full EDM pool

Finding: Application failures were caused by the EDM pool being full.

Impact: This finding can have a MEDIUM IMPACT, or HIGH IMPACT on the performance of the DB2 subsystem. It will have a HIGH IMPACT on applications.

Discussion: The environmental descriptor manager (EDM) pool is a pool of main storage used for database descriptors and application plans. The EDM pool contains:

- Database descriptors (DBDs)
- Skeleton cursor tables (SKCTs)
- Cursor tables (CTs), or copies of the SKCTs
- Skeleton package tables (SKPTs)
- Package tables (PTs), or copies of the SKPTs
- An authorization cache block for each plan, excluding plans that were created specifying CACHESIZE(0)
- Skeletons of dynamic SQL if YES has been specified for the CACHE DYNAMIC SQL field of installation panel DSNTIP4

The EDM pool helps to minimize the response time when DB2 would otherwise repeatedly read objects from DASD. The EDM pool also is used as a working storage area for threads as the threads are executing.

During the installation process, DSNTINST CLIST calculates the size of the EDM pool, based on parameters specified on the DSNTIPD and DSNTIPE panels. The calculated size of the EDM pool is only an estimate of the space required. If the EDM pool is too small or the available storage is fragmented, applications and ad hoc queries may fail. Additionally, all users of the system may receive poor performance because DB2 would spend unnecessary time managing the storage area by stealing pages and repeatedly re-loading DB2 objects. Consequently, users should monitor the EDM pool and adjust the size as necessary.

When pages are needed for the EDM pool, DB2 acquires pages as follows:

©Copyright 1999, Computer Management Sciences, Inc.

- Any pages in an “available” status are allocated first. The available pages are those with an in-use count=0.

- If the available pages do not provide enough space to satisfy the request, pages are "stolen" from an inactive SKCT, SKPT, DBD, or dynamic SQL skeleton. The pages are stolen on a lease-recently-used (LRU) basis.

- If there is still not enough space, allocation fails and an SQL error code (-904: resource unavailable return code) is sent to the application program.

Even if applications do not fail, DB2 spends more time stealing pages and repeatedly reloading DB2 objects if the EDM pool is too small. The increased number of loads can increase application response time. Consequently, a failure because of a full EDM pool should be viewed as the result of a continuing problem that finally manifests itself with a full EDM pool.

CPExpert compares the QISEFAIL variable in DB2STATS (the number of failures caused by a full EDM pool) with the **QISEFAIL** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-261 when the number of failures caused by a full EDM pool exceeds the value specified by the **QISEFAIL** guidance variable.

The default value for the **QISEFAIL** guidance variable is 0, indicating that CPExpert should produce Rule DB2-261 when any failures were caused by a full EDM pool. In general, a single EDM pool failure should cause you to investigate the situation. Consequently, you normally should not change the default value of this guidance variable.

Suggestion: When applications receive a -904 SQL return code indicating an EDM pool load failure, a DSNT500I (type=600) message is also generated. The error message may contain one of the following DB2 reason codes:

- **00C9008F**: The database descriptor's size (DBD) has reached the limit of 25% of the EDM Pool size. When this reason code is produced, you should create the database object in a different database or drop unused database objects from the database. If you drop an object that is an index or a table space, the space becomes available in the DBD as soon as the drop is committed.

- **00E40609**. A DB2 utility encountered a DBD which was noncontiguous in the EDM pool due to prior operations and the DBD was currently accessed by users. The utility requires the DBD to be contiguous in the EDM pool. When this reason code is produced, you should allow all SQL

access to this database to cease; this will allow the DBD to be flushed from the EDM pool. After all users have finished accessing the database, invoke the utility again. A contiguous DBD will be brought into the EDM pool. This reason code likely would be detected by the DB2 Administrator and corrective action taken immediately.

- **00C90089**: The environmental descriptor manager (EDM) function of the data manager sub-component did not have enough storage in its pool to support the retrieval of the identified object, or did not have enough contiguous storage to support the retrieval (because storage was fragmented). The object can be either a database descriptor (DBD), a skeleton cursor table (SKCT) or a skeleton package table (SKPT). This reason code is the one most likely to generate Rule DB2-261 on a recurring basis.

  If Rule DB2-261 is produced, the DB2 reason code is 00C90089, and EDM storage shortage is a recurring problem, CPExpert suggests that you consider the following alternatives:

  - Reduce the number of concurrent users using DB2.

  - Try reducing the number of SQL statements in the plan or package and rebind.

  - Try to simplify the database design--for example, reduce the number of partitions in the database.

  - Unless the DB2 system is heavily stressed and there is an overall memory constraint, you should increase the size of the EDM pool. Change the installation parameters that affect the size of the EDM pool--that is, increase the number of concurrent users and the number of currently active databases on the UPDATE install panels, then restart DB2. IBM recommends that the EDM pool be at least 10 times the size of the largest database descriptor or plan, whichever is greater.

  - If you cannot increase the size of the EDM pool, you should consider (1) rescheduling workload to relieve system constraints, (2) perform overall tuning of the system to achieve better overall performance, or (3) acquire additional memory.

  - If your EDM pool storage grows continually, consider having DB2 periodically free unused thread storage. To do this, specify YES for the CONTSTOR subsystem parameter and then reassemble DSNTIJUZ. This option can affect performance and is best used when

your system has many long-running threads and your EDM storage is constrained.

- You can alter CPExpert's analysis by modifying the **QISEFAIL** guidance varialble in USOURCE(DB2GUIDE).

Reference: DB2 for OS/390 Version 3: Administration Guide
Section 7.5.2 (Tuning the EDM Pool)

DB2 for OS/390 Version 3: Messages and Codes

DB2 for OS/390 Version 4: Administration Guide
Section 5.4.2 (Tuning the EDM Pool)

DB2 for OS/390 Version 4: Messages and Codes

DB2 for OS/390 Version 5: Administration Guide
Section 5.4.2 (Tuning the EDM Pool)

DB2 for OS/390 Version 5: Messages and Codes

DB2 for OS/390 Version 6: Administration Guide
Section 5.4.2 (Tuning the EDM Pool)

DB2 for OS/390 Version 6: Messages and Codes

Rule DB2-262: Low percent of Data Base Descriptors found in EDM pool

Finding: A low percent of Data Base Descriptors (DBDs) were found in the EDM pool.

Impact: This finding can have a MEDIUM IMPACT, or HIGH IMPACT on the performance of the DB2 subsystem.

Discussion: A database descriptor (DBD) is an internal representation of DB2 database definition which reflects the data definition found in the DB2 catalog. The objects defined in a database descriptor are table spaces, tables, indexes, index spaces, and relationships. A DBD is created when a user issues a "CREATE DATABASE" SQL statement.

DBDs must be loaded into the EDM pool as a DB2 thread executes, unless the required DBDs are already in the EDM pool. Even if a plan accesses only a single tablespace of a DBD, the entire DBD must reside in the EDM pool (in contiguous storage) before access is allowed.

A particular DBD can be used by more than one plan, and a "use count" keeps track of whether the DBD is being used. If space is needed in the EDM pool, a DBD can be freed when its use count reaches zero.

Whether a process locks a target DBD depends largely on whether the DBD is already in the EDM pool.

- If the DBD is not in the EDM pool, most processes acquire locks on the database descriptor table space (DBD01[1]). That has the effect of locking the DBD and can cause conflict with other processes.

- If the DBD is in the EDM pool, the lock on the DBD depends on the type of process; there is no lock for static DML statements (SELECT, DELETE, INSERT, UPDATE).

The DBDs are intended to provide reduced I/O to obtain file definitions, provide fast symbolic resolution, and provide quick identification of related objects. However, the DBDs can be very large and they usually span many 4K pages. The amount of time to load a DBD into the EDM pool can elongate response time. Consequently, it is desirable that the DBD already reside in the EDM pool when it is required.

---

[1] DBD01 is the database descriptor (DBD) table space.

©Copyright 1999, Computer Management Sciences, Inc.

CPExpert computes the “hit ratio” of DBD requests that found the DBD in the EDM pool. The calculation is:

$$\textit{Hit ratio} = \frac{\textit{Requests for DBDs} - \textit{DBDs loaded DASD}}{\textit{Requests for DBDs}}$$

CPExpert compares the computed hit ratio with the **PCTDBDL** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-262 when the hit ratio is less than the value specified by the **PCTDBDL** guidance variable.

The default value for the **PCTDBDL** guidance variable is 99%, indicating that CPExpert should produce Rule DB2-262 when the hit ratio for DBDs in the EDM pool is less than 99%.

IBM suggests that a hit ratio of 80%-90% would be acceptable in most situations in small to medium sites. However, a higher hit ratio would be required to achieve good performance in an important system with a high transaction volume. **Consequently, CPExpert has set the default to be very high, with the expectation that you will lower the guidance if your DB2 system is not processing important applications with a high volume of transactions.** Please note that this high hit ratio would not be achieved when DB2 first starts, but should be achieved after DB2 reaches a stable state.

Within the context of the recommendations about hit ratios, IBM considers the following DB2 site characteristics:

- The small site supports a small number of DB2 users. The small site has about 100 plans, 50 application databases, and 500 tables.

- The medium site supports more extensive use of DB2 databases. The medium-sized site has about 200 plans, 200 application databases, and 2000 tables.

- The large site supports heavy use of DB2. The large site has about 400 plans, 400 application databases, and 4000 tables.

- The extra large site supports very heavy use of DB2. The extra large site has about 600 plans, 600 application databases, and 6000 tables.

The following example illustrates the output from Rule DB2-262:

```
RULE DB2-262: LOW PERCENT OF DATA BASE DESCRIPTORS FOUND IN EDM POOL

   Data base descriptors (DBD) were found in the EDM pool for less than
   99% of the DBD requests.  Once a DB2 system has been operating for
   awhile (and reached a stable condition), at least 99% of the DBD
   requests should find the DBD in the EDM pool. This situation occurred
   during the intervals shown below:

                             PAGES USED  REQUESTS    TIMES DBD     "HIT"
   MEASUREMENT INTERVAL       FOR DBD    FOR  DBD   IN EDM POOL    RATIO
   11:20-11:49, 16SEP1999         46         338       234           69
```

Suggestion: CPExpert suggests that you consider the following alternatives if Rule DB2-262 is regularly produced and your DB2 application is important:

- Run a job to pre-load all DBDs ("prime" the EDM pool with DBDs) so that subsequent applications will find the requested DBD in the EDM pool. A DISPLAY DATABASE command with cause the load of all named DBDs if not already in the EDM pool.

- Increase the size of your EDM pool so that DBDs pages will not be stolen when DB2 requires pages in the EDM pool.

- Review the application to ensure that all database objects are required to be in the database. Perhaps you can reduce the size of DBDs by removing unused objects. Please note that if you drop objects, storage is not automatically reclaimed in the DBD, which can mean that DB2 must take more locks for the DBD. To reclaim storage in the DBD, use the MODIFY utility, as described in Section 2 of the *DB2 Utility Guide and References*.

- Review the application to determine whether you can create the database object in a different database. DBDs must be loaded into contiguous storage in the EDM pool. Perhaps by reducing the size of DBDs, more DBDs can the loaded into contiguous storage in the EDM pool.

- Alter CPExpert's analysis by modifying the **PCTDBDL** guidance variable in USOURCE(DB2GUIDE).

Reference: DB2 for OS/390 Version 3: Administration Guide
Section 7.5.2 (Tuning the EDM Pool)

DB2 for OS/390 Version 3: Installation Guide
Section 2.2 (Estimating DB2 Storage Needs)

DB2 for OS/390 Version 4: Administration Guide
Section 5.4.2 (Tuning the EDM Pool)

DB2 for OS/390 Version 4: Installation Guide
Section 2.2 (Estimating DB2 Storage Needs)

DB2 for OS/390 Version 5: Administration Guide
Section 5.4.2 (Tuning the EDM Pool)

DB2 for OS/390 Version 5: Installation Guide
Section 2.2 (Estimating DB2 Storage Needs)

DB2 for OS/390 Version 6: Administration Guide
Section 5.4.2 (Tuning the EDM Pool)

DB2 for OS/390 Version 6: Installation Guide
Section 2.2 (Estimating DB2 Storage Needs)

©Copyright 1999, Computer Management Sciences, Inc.

Rule DB2-263: Low percent of cursor table sections found in EDM pool

Finding: A low percent of cursor table sections (CTs) were found in the EDM pool.

Impact: This finding can have a LOW IMPACT or MEDIUM IMPACT on the performance of the DB2 subsystem.

Discussion: The cursor table (CT) is the copy of the skeleton cursor table (SKCT) used by an executing application process. The CTs are created in the EDM pool during plan execution, and contain the sections (access paths) corresponding to the SQL statements that each user executes. Each thread requires its own CT.

When a SQL section is needed and it exists in the SKCT in the EDM pool, it is copied to create a new CT section for the user. However, if the SQL section does not exist in the EDM pool, GETPAGE requests are issued to the buffer manager. For any SQL sections not found in the DB2 buffer pool (BP0), the section is loaded from DASD[1]. Loading CTs from DASD can elongate response time.

CPExpert computes the "hit ratio" of CT requests that found the CT in the EDM pool. The calculation is:

$$\textit{Hit ratio} = \frac{\textit{Requests for CTs} - \textit{CTs loaded DASD}}{\textit{Requests for CTs}}$$

CPExpert compares the computed hit ratio with the **PCTCTL** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-263 when the hit ratio is less than the value specified by the **PCTCTL** guidance variable.

The default value for the **PCTCTL** guidance variable is 95%, indicating that CPExpert should produce Rule DB2-263 when the hit ratio for CTs in the EDM pool is less than 95%.

IBM suggests that a hit ratio of 80% would be acceptable in most situations in small to medium sites. However, a higher hit ratio would be required to achieve good performance in an important system with a high transaction volume. **Consequently, CPExpert has set the default to be higher than IBM's recommendation, with the expectation that you will lower the guidance if your DB2 system is not processing important applications**

[1] Note that the section is copied to the appropriate SKCT to ensure that it is available for other users.

©Copyright 1999, Computer Management Sciences, Inc.

with a high volume of transactions. Please note that this high hit ratio would not be achieved when DB2 first starts, but should be achieved after DB2 reaches a stable state.

Within the context of the recommendations about hit ratios, IBM considers the following DB2 site characteristics:

- The small site supports a small number of DB2 users. The small site has about 100 plans, 50 application databases, and 500 tables.
- The medium site supports more extensive use of DB2 databases. The medium-sized site has about 200 plans, 200 application databases, and 2000 tables.
- The large site supports heavy use of DB2. The large site has about 400 plans, 400 application databases, and 4000 tables.
- The extra large site supports very heavy use of DB2. The extra large site has about 600 plans, 600 application databases, and 6000 tables.

The following example illustrates the output from Rule DB2-263:

```
RULE DB2-263: LOW PERCENT OF CURSOR TABLE SECTIONS FOUND IN EDM POOL

   Cursor table sections (CTS) were found in the EDM pool for less than
   90% of the CTS requests.  Once a DB2 system has been operating for
   awhile (and reached a stable condition), at least 95% of the CTS
   requests should find the CTS in the EDM pool. This situation occurred
   during the intervals shown below:

                             PAGES USED  REQUESTS    TIMES CTS     "HIT"
   MEASUREMENT INTERVAL       FOR CTS    FOR  CTS   IN EDM POOL    RATIO
   10:28-10:58, 15SEP1999          1            7           5         71
   13:22-13:52, 16SEP1999          1            4           2         50
```

Suggestion: CPExpert suggests that you consider the following alternatives if Rule DB2-263 is regularly produced and your DB2 application is important:

- Increase the size of your EDM pool to increase the cache size for CT pages.
- Alter CPExpert's analysis by modifying the **PCTCTL** guidance variable in USOURCE(DB2GUIDE).

Reference: DB2 for OS/390 Version 3: Administration Guide
Section 7.5.2 (Tuning the EDM Pool)

DB2 for OS/390 Version 4: Administration Guide
Section 5.4.2 (Tuning the EDM Pool)

DB2 for OS/390 Version 5: Administration Guide
Section 5.4.2 (Tuning the EDM Pool)

DB2 for OS/390 Version 6: Administration Guide
Section 5.4.2 (Tuning the EDM Pool)

Rule DB2-264: Low percent of package table sections found in EDM pool

Finding: A low percent of package table sections (PTs) were found in the EDM pool.

Impact: This finding can have a MEDIUM IMPACT, or HIGH IMPACT on the performance of the DB2 subsystem.

Discussion: The package table (PT) is the copy of the skeleton package table (SKPT) used by an executing application process. The skeleton package table space contains a table that describes the internal form of SQL statements in application programs. When a package is bound, DB2 creates a skeleton package table in SPT01.

The PTs are created in the EDM pool during plan execution, and contain the sections (access paths) corresponding to the SQL statements that each user executes. Each thread requires its own PT.

When a SQL section is needed and it exists in the SKPT in the EDM pool, it is copied to create a new PT section for the user. However, if the SQL section does not exist in the EDM pool, GETPAGE requests are issued to the buffer manager. For any SQL sections not found in the DB2 buffer pool (BP0), the section is loaded from DASD[1]. Loading PTs from DASD can elongate response time.

CPExpert computes the "hit ratio" of PT requests that found the PT in the EDM pool. The calculation is:

$$\textit{Hit ratio} = \frac{\textit{Requests for PTs} - \textit{PTs loaded DASD}}{\textit{Requests for PTs}}$$

CPExpert compares the computed hit ratio with the **PCTPTL** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-264 when the hit ratio is less than the value specified by the **PCTPTL** guidance variable.

The default value for the **PCTPTL** guidance variable is 95%, indicating that CPExpert should produce Rule DB2-264 when the hit ratio for PTs in the EDM pool is less than 95%.

IBM suggests that a hit ratio of 80% would be acceptable in most situations in small to medium sites. However, a higher hit ratio would be required to

[1] Note that the section is copied to the appropriate SKPT to ensure that it is available for other users.

©Copyright 1999, Computer Management Sciences, Inc.

achieve good performance in an important system with a high transaction volume. **Consequently, CPExpert has set the default to be higher than IBM's recommendation, with the expectation that you will lower the guidance if your DB2 system is not processing important applications with a high volume of transactions.** Please note that this high hit ratio would not be achieved when DB2 first starts, but should be achieved after DB2 reaches a stable state.

Within the context of the recommendations about hit ratios, IBM considers the following DB2 site characteristics:

- The small site supports a small number of DB2 users. The small site has about 100 plans, 50 application databases, and 500 tables.
- The medium site supports more extensive use of DB2 databases. The medium-sized site has about 200 plans, 200 application databases, and 2000 tables.
- The large site supports heavy use of DB2. The large site has about 400 plans, 400 application databases, and 4000 tables.
- The extra large site supports very heavy use of DB2. The extra large site has about 600 plans, 600 application databases, and 6000 tables.

The following example illustrates the output from Rule DB2-264:

```
RULE DB2-264: LOW PERCENT OF PACKAGE TABLE SECTIONS FOUND IN EDM POOL

   Package table sections (PTS) were found in the EDM pool for less than
   90% of the PTS requests.  Once a DB2 system has been operating for
   awhile (and reached a stable condition), at least 95% of the PTS
   requests should find the PTS in the EDM pool. This situation occurred
   during the intervals shown below:

                            PAGES USED   REQUESTS    TIMES PTS     "HIT"
   MEASUREMENT INTERVAL      FOR PTS     FOR  PTS   IN EDM POOL    RATIO
   10:28-10:58, 15SEP1999         0           12          6          50
   14:27-14:57, 15SEP1999         0           40         33          83
    9:23- 9:53, 16SEP1999         0           14         11          79
   13:22-13:52, 16SEP1999         0           24         12          50
```

Suggestion: CPExpert suggests that you consider the following alternatives if Rule DB2-264 is regularly produced and your DB2 application is important:

- Increase the size of your EDM pool to increase the cache size for PT pages.

- Alter CPExpert's analysis by modifying the **PCTPTL** guidance variable in USOURCE(DB2GUIDE).

Reference: DB2 for OS/390 Version 3: Administration Guide
Section 7.5.2 (Tuning the EDM Pool)

DB2 for OS/390 Version 4: Administration Guide
Section 5.4.2 (Tuning the EDM Pool)

DB2 for OS/390 Version 5: Administration Guide
Section 5.4.2 (Tuning the EDM Pool)

DB2 for OS/390 Version 6: Administration Guide
Section 5.4.2 (Tuning the EDM Pool)

Rule DB2-265: Size of EDM pool could be reduced

Finding: A relatively low number of pages was “in use” in the EDM pool, indicating that the size of the EDM pool could be reduced.

Impact: This finding can have a LOW IMPACT or MEDIUM IMPACT on the performance of the DB2 subsystem.

Discussion: The environmental descriptor manager (EDM) pool is a pool of main storage used for database descriptors and application plans. The EDM pool contains:

- Database descriptors (DBDs)
- Skeleton cursor tables (SKCTs)
- Cursor tables (CTs), or copies of the SKCTs
- Skeleton package tables (SKPTs)
- Package tables (PTs), or copies of the SKPTs
- An authorization cache block for each plan, excluding plans that were created specifying CACHESIZE(0)
- Skeletons of dynamic SQL if YES has been specified for the CACHE DYNAMIC SQL field of installation panel DSNTIP4

The EDM pool helps to minimize the response time when DB2 would otherwise repeatedly read objects from DASD. The EDM pool also is used as a working storage area for threads as the threads are executing.

During the installation process, DSNTINST CLIST calculates the size of the EDM pool, based on parameters specified on the DSNTIPD and DSNTIPE panels. The calculated size of the EDM pool is only an estimate of the space required. If the EDM pool is too small or the available storage is fragmented, applications and ad hoc queries may fail. Additionally, all users of the system may receive poor performance because DB2 would spend unnecessary time managing the storage area by stealing pages and repeatedly re-loading DB2 objects. Consequently, users should monitor the EDM pool and adjust the size as necessary.

©Copyright 1999, Computer Management Sciences, Inc.

IBM recommends that the EDM pool be at least 10 times the size of the largest database descriptor or plan, whichever is greater.

However, depending on the application environment, the EDM pool could be too large. If the EDM pool is too large, real and virtual storage would be wasted. This storage could be reclaimed for other uses (for example, used to increase buffer pool allocation).

CPExpert computes the percent of unused (or “free”) pages in the EDM pool. The calculation is:

$$\textit{Percent free pages in EDM pool} = \frac{\textit{Number of free EDM pool pages}}{\textit{Total pages in EDM pool}}$$

CPExpert compares the computed percent free pages in EDM pool with the **PCTEDM** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-265 when the computed percent free pages in EDM pool exceeds the value specified by the **PCTEDM** guidance variable.

The default value for the **PCTEDM** guidance variable is 25%, indicating that CPExpert should produce Rule DB2-265 when the percent free pages in EDM pool is greater than 25%.

The following example illustrates the output from Rule DB2-265:

```
RULE DB2-265: SIZE OF EDM POOL COULD BE REDUCED

   More than 0.25% of the EDM pool was unused during the intervals shown
   below. This finding could indicate that you could reduce the number
   of buffers allocated to the EDM pool.  Please note that the finding is
   based on data that is only a "snapshot" at the time the DB2 statistics
   were written.  You should NOT reduce the size of the EDM Pool unless
   the finding is consistently produced!  Please “turn off” this rule if
   you do not wish to alter the EDM pool size.

                              PAGES IN      FREE PAGES         PCT EDM
   MEASUREMENT INTERVAL       EDM POOL      IN EDM POOL       POOL USED
    8:57- 9:27, 15SEP1999       4,000          3,955             1.1
    9:27- 9:57, 15SEP1999       4,000          3,955             1.1
    9:57-10:27, 15SEP1999       4,000          3,955             1.1
   10:27-10:57, 15SEP1999       4,000          3,955             1.1
   10:57-11:27, 15SEP1999       4,000          3,955             1.1
```

Suggestion: This finding is useful primarily for sites which experience a shortage of storage for DB2. If shortage of storage is not a problem, you should ignore (and “turn off”) this rule so you will not be annoyed by CPExpert producing a finding for which you will take no action.

If CPExpert produces Rule DB2-265 on a **regular basis during peak periods**, you should consider the following alternatives:

- The DSNTIPC panel can be used to override (and reduce) the EDM pool size.

  Caution: The maximum size of a database descriptor (DBD) is 25% of the size of the EDM pool. If a DBD exceeds 25% of the EDM pool size, applications receive a -904 SQL return code indicating an EDM pool load failure. You must ensure that the EDM pool size is at least four times the estimated size of your largest database descriptor.

- You can alter CPExpert's analysis by modifying the **PCTEDM** guidance variable in USOURCE(DB2GUIDE).

Reference:

DB2 for OS/390 Version 3: Administration Guide
Section 2.4.1.32 (Install DB2--CLIST Calculations Panel 1: DSNTIPC)
Section 7.6.7 (Improving Main Storage Utilization)
Section 7.7.2 (Tuning the EDM Pool)

DB2 for OS/390 Version 4: Administration Guide
Section 5.4.2 (Tuning the EDM Pool)
Section 5.5.7 (Improving Main Storage Utilization)

DB2 for OS/390 Version 4: Installation Guide
Section 2.2.4 (Virtual Storage for Storage Pools and Working Storage)
Section 2.5.1.36 (Install DB2--CLIST Calculations Panel 1: DSNTIPC)

DB2 for OS/390 Version 5: Administration Guide
Section 5.4.2: Tuning the EDM Pool
Section 5.5.7 (Improving Main Storage Utilization)

DB2 for OS/390 Version 5: Installation Guide
Section 2.2.4 (Virtual Storage for Storage Pools and Working Storage)
Section 2.5.32 (Install DB2--CLIST Calculations Panel 1: DSNTIPC)

DB2 for OS/390 Version 6: Administration Guide
Section 5.4.2: Tuning the EDM Pool
Section 5.5.7 (Improving Main Storage Utilization)

DB2 for OS/390 Version 6: Installation Guide
Section 2.2.3 (Virtual Storage for Storage Pools and Working Storage)
Section 2.5.32 (Install DB2--CLIST Calculations Panel 1: DSNTIPC)

©Copyright 1999, Computer Management Sciences, Inc.

Rule DB2-301: Work was suspended because of lock conflict

Finding: A lock could not be obtained because of a lock conflict, and the unit of work was suspended.

Impact: This finding can have a MEDIUM IMPACT, or HIGH IMPACT on the performance of the DB2 subsystem.

Discussion: Locking is a means of controlling concurrent events or access to data. Locking prevents concurrent users from accessing inconsistent data.

DB2 uses locks to ensure that no process accesses data that has been changed (but not yet committed) by another process. With DB2, locking is performed by the Internal Resource Lock Manager (IRLM)[1].

A lock associates a DB2 resource with an application process in a way that affects how other processes can access the same resource. The resource can be a table space, a table (if table spaces are segmented), pages, and rows. The process associated with the resource is said to "hold" or "own" the lock.

There are three basic locks that can be held or requested for page or row locks: (1) share locks, (2) update locks, and (3) exclusive locks.

- **Share Lock:** The lock owner and any concurrent process can read, but not change, the locked DB2 object. Other concurrent processes may acquire Share or Update locks on the DB2 object. A share lock is called an S-lock.

- **Update Lock:** The lock owner can read the DB2 object and intends to change it. Concurrent processes may acquire Share locks and read the DB2 object but no other process can acquire an Update lock. Update locks must be promoted to Exclusive locks before DB2 actually changes the DB2 object. Promotion to Exclusive lock may cause a suspension if other processes are holding Share locks. An update lock is called a U-lock.

- **Exclusive Lock:** Only the lock owner can read or change the locked data, with the following exceptions.

[1] Formally known as the IMS Resource Lock Manager or IMS/VS Resource Lock Manager, depending on the IBM document your referenced.

©Copyright 1999, Computer Management Sciences, Inc.

- Concurrent applications using uncommitted read isolation can read the exclusively locked data.

- If lock avoidance techniques indicate the exclusively locked data is committed, the data can be returned to concurrent cursor-stability applications that do not require currency of data.

  An exclusive lock is also called an U-lock.

When one application process holds a lock on a resource, whether another application process can access the resource depends on the type of lock held and on the type of lock requested.

The following table shows the compatibility between the type of lock held and the type of lock requested for page or row locks:

Held lock	Requested Lock		
	Share	**Update**	**Exclusive**
Share	Yes	Yes	No
Update	Yes	No	No
Exclusive	No	No	No

The locking modes are more complex with table, partition, and table space locks. There are six basic locks that can be held or requested for table, partition, and table space locks: (1) intent share locks, (2) intent exclusive locks, (3) share locks, (4) update locks, (5) share with intent exclusive locks, and (6) exclusive locks.

- **Intent Share Lock:** The lock owner can read data in the table, partition, or table space, but not change it. Concurrent processes can both read and change the data. The lock owner might acquire a page or row lock on any data it reads. An intent share lock is also called an IS-lock.

- **Intent Exclusive Lock:** The lock owner and concurrent processes can read and change data in the table, partition, or table space. The lock owner acquire a page or row lock on any data it reads; it must acquire one on any data it changes. An intent exclusive lock is called an IX-lock.

- **Share Lock:** The lock owner and any concurrent processes can read, but not change, data in the table, partition, or table space. The lock owner does not need page or row locks on data it reads.

- **Update Lock:** The lock owner can read, but not change, the locked data; however, the owner can promote the lock to an X lock and then can change the data. Processes concurrent with the U lock can acquire S locks and read the data, but no concurrent process can acquire a U lock. The lock owner does not need page or row locks. U locks reduce the chance of deadlocks when the lock owner is reading data to determine whether to change it.

- **Share with Intent Exclusive Lock:** The lock owner can read and change data in the table, partition, or table space. Concurrent processes can read data in the table partition, or table space, but not change it. Only when the lock owner changes data does it acquire page or row locks.

- **Exclusive Lock:** The lock owner can read or change data in the table, partition, or table space. A concurrent process can access the data if the process runs with UR isolation. The lock owner does not need page or row locks.

The following table shows the compatibility between the type of lock held and the type of lock requested for with table, partition, and table space locks:

Held Lock	Requested Lock					
	IS	**IX**	**S**	**U**	**SIX**	**X**
Intent Share (IS)	Yes	Yes	Yes	Yes	Yes	No
Intent Exclusive (IX)	Yes	Yes	No	No	No	No
Share (S)	Yes	No	Yes	Yes	No	No
Update (U)	Yes	No	Yes	No	No	No
Share with Intent Exclusive (SIX)	Yes	No	No	No	No	No
Exclusive (X)	No	No	No	No	No	No

A process is **suspended** when it requests a lock that is already held by another application process and cannot be shared. The suspended process temporarily stops running.

- All processes that hold the conflicting lock release it.

- The requesting process times out or deadlocks, and the process resumes to deal with an error condition.

If programs are involved in lock contention problems, the main symptom is lock suspensions. From a performance view, lock suspensions increase the response time of the programs.

Unfortunately, wait time for locks is not *normally* externalized unless there is a timeout or deadlock. Wait time for lock suspensions is available only if tracing for Performance Class 7 is turned on, and this option significantly increases the CPU overhead of tracing. Consequently, users do not normally turn on tracing of lock suspensions unless they are analyzing a particular contention problem.

CPExpert computes the percent of lock requests that were suspended by the following algorithm:

$$\textit{Percent locks suspended} = \frac{\textit{Locks suspended}}{\textit{Lock requests}} \quad \textit{or} \quad \frac{\textit{QTXASLOC}}{\textit{QTXALOCK}}$$

CPExpert compares the computed percent locks suspended with the **PCTSLOCK** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-301 when the calculated percent locks suspended exceeds the value specified by the **PCTSLOCK** guidance variable.

The default value for the **PCTSLOCK** guidance variable is 0.5%, indicating that CPExpert should produce Rule DB2-301 when more than one-half percent of the lock requests were suspended. This relatively low percent was selected to alert you to the potential problem of lock suspensions. You should consider altering this guidance value after reviewing the suggestions below.

The following example illustrates the output from Rule DB2-301:

```
RULE DB2-301: WORK WAS SUSPENDED BECAUSE OF LOCK CONFLICT

   DB2 uses locks to ensure that no process accesses data that has been
   changed (but not yet committed) by another process.  A process is
   suspended when it requests a lock that is already held by another
   application process and cannot be shared. The suspended process
   temporarily stops running.  The number of lock suspensions should be
   low (ideally, lock suspensions should not exist).  However, lock
   suspensions are highly dependent on the application and table space
   locking protocols.  Lock suspensions occurred for over 0.5% of the locks
   during the intervals shown below:

                                 NUMBER OF       NUMBER OF LOCK
   MEASUREMENT INTERVAL        LOCK REQUESTS       SUSPENSIONS     PERCENT
   20:43-21:13, 16SEP1999          101,664             614            0.6
   21:13-21:43, 16SEP1999          157,696             882            0.6
```

Suggestion: When a lock request cannot be satisfied because object that is already locked and has an incompatible lock state, the process requesting the lock is suspended. The suspension of a process can be resolved in one of the following general ways:

- By removing the cause for the suspension, for example, by releasing the locked resource.

- By a reaching a timeout situation, when the process waits for the resource for longer than a preset interval. When a timeout occurs, IRLM passes control back to DB2 with an error condition. Timeout situations are analyzed by CPExpert and reported in Rule DB2-310.

- By detecting a deadlock as the cause for the suspension and having DB2 roll back one of the deadlocked application processes. Deadlock situations are analyzed by CPExpert and reported in Rule DB2-311.

In removing the cause of the lock suspension, you should consider the following alternatives:

- If a task is waiting or is swapped out and the unit of work has not been committed, then it still holds locks. When a system is heavily loaded, contention for processing, I/O, and storage can cause waiting. Consider reducing the number of initiators, increasing the priority for the DB2 tasks, and providing more processing, I/O, or storage resources.

- Make sure that the IRLM has a high MVS-dispatching priority. IBM recommends that IRLM come next after VTAM and before DB2. If you are running in Goal Mode, you should assign IRLM to SYSSTC, so it will have high dispatching priority.

- If you can define more ECSA, then start the IRLM with PC=NO rather than PC=YES. You can make this change without changing your application process. This change can also reduce processing time.

- Alternatively, you can revise the application so that it issues COMMIT more frequently.

You should review the sections titled *"Tuning Your Use of Locks"* in IBM's DB2 *Administration Guides* for more suggestions from IBM on how to minimize or eliminate the negative effects of locks.

You can alter CPExpert's analysis by modifying the **PCTSLOCK** guidance variable in USOURCE(DB2GUIDE).

Reference: DB2 for OS/390 Version 3: Administration Guide
Section 7.8.1 (Aspects of Transaction Locking)
Section 7.8.4 (Tuning Your Use of Locks)

DB2 for OS/390 Version 4: Administration Guide
Section 5.7.4 (Aspects of Transaction Locks)
Section 5.7.5 (Tuning Your Use of Locks)

DB2 for OS/390 Version 5: Administration Guide
Section 5.7.4 (Aspects of Transaction Locks)
Section 5.7.5 (Tuning Your Use of Locks)

DB2 for OS/390 Version 6: Administration Guide
Section 5.7.4 (Aspects of Transaction Locks)
Section 5.7.5 (Tuning Your Use of Locks)

©Copyright 1999, Computer Management Sciences, Inc.

Rule DB2-302: Locks were escalated to shared mode

Finding: Locks were escalated to shared mode.

Impact: This finding can have a LOW IMPACT, MEDIUM IMPACT, or HIGH IMPACT on the performance of the DB2 subsystem. The level of impact depends on the number of locks that were escalated to shared mode.

Discussion: Lock escalation is the act of releasing a large number of page or row locks, held by an application process on a single table or table space, to acquire a table or table space lock, or a set of partition locks, of lock Mode S or lock Mode X[1].

Lock escalation balances concurrency[2] with performance by using page or row locks while a process accesses relatively few pages or rows, then changing the lock to table space, table, or partition locks when the process accesses many pages or rows. Thus, DB2 allows applications to concurrently access information so long as the performance cost of locking is not prohibitive.

The page or row locking uses *significant* processing time. Once the performance cost of locking is unacceptable, DB2 changes (or escalates) the locks to lock the entire table space, table, or partition (thus holding a single lock) and releases all the individual page or row locks. This lock escalation can reduce concurrency because other applications *may* not be able to access the table space, table, or partition (depending on the locking required). However, the lock escalation improves performance because the overhead of managing many locks is exchanged for the overhead of managing only a single lock.

When it occurs, lock escalation varies by table space, depending on the values of LOCKSIZE and LOCKMAX, as described below.

- The *LOCKSIZE* clause on the CREATE TABLESPACE and ALTER TABLESPACE statements specifies the size of a lock held on a table or table space by any application process that accesses it. The default specification of LOCKSIZE=ANY allows DB2 to chose the size of the lock. Other options can be specified in unique situations to explicitly restrict the size of the lock. These options are

[1] Please refer to Rule DB2-301 for a discussion of different lock modes.

[2] Concurrency is the ability of more than one application process to access the same data at essentially the same time.

LOCKSIZE=TABLESPACE, LOCKSIZE=TABLE, LOCKSIZE=PAGE, and LOCKSIZE=ROW. However, the default LOCKSIZE=ANY should be used unless the situation is unique.

- *LOCKMAX* is an option, available with DB2 Version 4, that is specified on the CREATE TABLESPACE and ALTER TABLESPACE statements. The LOCKMAX option defines the maximum number of page or row locks that an application process can hold simultaneously in the table space. If a program requests more than that number, locks are escalated. The page or row locks are released, and the intent lock on the table space or segmented table is promoted to S or X.

  Prior to DB2 Version 4, the maximum number of page or row locks that an application process can hold simultaneously in the table space is applied to **all** table spaces, via the LOCKS PER TABLE(SPACE) field of panel DSNTIPJ. This specification on panel DSNTIPJ is still made with DB2 Version 4, but the specification can be over-ridden for individual table spaces using the LOCKMAX clause.

DB2 Version 4 extended the lock escalation process by:

- Applying the count of locks to row locks in the same way as to page locks.

- Lock escalation applies not only to table spaces defined with LOCKSIZE ANY, but also to those defined with LOCKSIZE PAGE or ROW.

- A different lock escalation cutoff point can be selected for a specific table space by specifying LOCKMAX for that table space. Lock escalation of a table space can be disabled by setting LOCKMAX to 0.

Lock escalations are generally undesirable and are caused by processes that use a large number of page or row locks. In some cases, it is possible to improve system performance by using table locks or table space locks.

CPExpert compares the QTXALES variable in DB2STATS (the number of locks escalated to shared mode) with **the QTXALES** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-302 when the number of locks escalated to shared mode exceeds the value specified by **the QTXALES** guidance variable.

The default value for **the QTXALES** guidance variable is 0, indicating that CPExpert should produce Rule DB2-302 when any locks were escalated to shared mode.

The following example illustrates the output from Rule DB2-302:

```
RULE DB2-302: LOCKS WERE ESCALATED TO SHARED MODE

   Lock escalation releases a large number of page or row locks, held by
   an application process on a single table or table space, to acquire a
   table or table space lock, or a set of partition locks, of mode S or X.
   Escalation can cause unpredictable response times.  Lock escalation
   should happen only when an application process updates or references
   (if repeatable read is used) more pages or rows than it normally does.
   This situation occurred during the intervals shown below:

                                     NUMBER OF          NUMBER OF LOCKS
   MEASUREMENT INTERVAL            LOCK REQUESTS      ESCALATED TO SHARED
   17:14-17:44, 16SEP1999              279,104                  1
```

Suggestion: If Rule DB2-302 is produced frequently, you should consider the following alternatives:

- You might alter the table to increase LOCKMAX and thus decrease the number of escalations.

- Alternatively, you might revise the application to ensure that it commits more frequently.

  Many locks can be taken with applications that use ISOLATION(RR) or ISOLATION(RS) . With these repeatable read or read stability applications referencing many rows or pages, lock escalation might take place. Frequent commits would release the locks and could help avoid lock escalation.

- Alternatively, you might let the process that is causing the lock escalations begin by locking the entire table space, using the statement LOCK TABLE. This would prevent concurrency, but it is a reasonable solution for some end-of-month or end-of-year situations when a process updates more pages than it normally does.

- Alternatively, you can alter CPExpert's analysis by modifying the **QTXALES** guidance variable in USOURCE(DB2GUIDE).

Reference: DB2 for OS/390 Version 3: Administration Guide
Section 2.4.1.23 (IRLM Panel 2: DSNTIPJ)
Section 5.7.4.5.3 (Lock Escalation)

DB2 for OS/390 Version 4: Administration Guide
Section 5.7.4.5.3 (Lock Escalation)

Section 5.7.6.3.3 (LOCKMAX Clause)

DB2 for OS/390 Version 5: Administration Guide
Section 5.7.4.5.3 (Lock Escalation)
Section 5.7.5.3.3 (LOCKMAX Clause)

DB2 for OS/390 Version 6: Administration Guide
Section 5.7.4.5.3 (Lock Escalation)
Section 5.7.5.3.3 (LOCKMAX Clause)

©Copyright 1999, Computer Management Sciences, Inc.

Rule DB2-303: Locks were escalated to exclusive mode

Finding: Locks were escalated to exclusive mode.

Impact: This finding can have a LOW IMPACT, MEDIUM IMPACT, or HIGH IMPACT on the performance of the DB2 subsystem. The level of impact depends on the number of locks that were escalated to exclusive mode.

Discussion: Lock escalation is the act of releasing a large number of page or row locks, held by an application process on a single table or table space, to acquire a table or table space lock, or a set of partition locks, of lock Mode S or lock Mode X[1].

Lock escalation balances concurrency[2] with performance by using page or row locks while a process accesses relatively few pages or rows, then changing the lock to table space, table, or partition locks when the process accesses many pages or rows. Thus, DB2 allows applications to concurrently access information so long as the performance cost of locking is not prohibitive.

The page or row locking uses *significant* processing time. Once the performance cost of locking is unacceptable, DB2 changes (or escalates) the locks to lock the entire table space, table, or partition (thus holding a single lock) and releases all the individual page or row locks. This lock escalation can reduce concurrency because other applications *may* not be able to access the table space, table, or partition (depending on the locking required). However, the lock escalation improves performance because the overhead of managing many locks is exchanged for the overhead of managing only a single lock.

When it occurs, lock escalation varies by table space, depending on the values of LOCKSIZE and LOCKMAX, as described below.

- The *LOCKSIZE* clause on the CREATE TABLESPACE and ALTER TABLESPACE statements specifies the size of a lock held on a table or table space by any application process that accesses it. The default specification of LOCKSIZE=ANY allows DB2 to chose the size of the lock. Other options can be specified in unique situations to explicitly restrict the size of the lock. These options are

[1] Please refer to Rule DB2-301 for a discussion of different lock modes.

[2] Concurrency is the ability of more than one application process to access the same data at essentially the same time.

LOCKSIZE=TABLESPACE, LOCKSIZE=TABLE, LOCKSIZE=PAGE, and LOCKSIZE=ROW. However, the default LOCKSIZE=ANY should be used unless the situation is unique.

- *LOCKMAX* is an option, available with DB2 Version 4, that is specified on the CREATE TABLESPACE and ALTER TABLESPACE statements. The LOCKMAX option defines the maximum number of page or row locks that an application process can hold simultaneously in the table space. If a program requests more than that number, locks are escalated. The page or row locks are released, and the intent lock on the table space or segmented table is promoted to S or X.

  Prior to DB2 Version 4, the maximum number of page or row locks that an application process can hold simultaneously in the table space is applied to **all** table spaces, via the LOCKS PER TABLE(SPACE) field of panel DSNTIPJ. This specification on panel DSNTIPJ is still made with DB2 Version 4, but the specification can be over-ridden for individual table spaces using the LOCKMAX clause.

DB2 Version 4 extended the lock escalation process by:

- Applying the count of locks to row locks in the same way as to page locks.

- Lock escalation applies not only to table spaces defined with LOCKSIZE ANY, but also to those defined with LOCKSIZE PAGE or ROW.

- A different lock escalation cutoff point can be selected for a specific table space by specifying LOCKMAX for that table space. Lock escalation of a table space can be disabled by setting LOCKMAX to 0.

Lock escalations are generally undesirable and are caused by processes that use a large number of page or row locks. In some cases, it is possible to improve system performance by using table locks or table space locks.

CPExpert compares the QTXALEX variable in DB2STATS (the number of locks escalated to exclusive mode) with **the QTXALEX** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-303 when the number of locks escalated to exclusive mode exceeds the value specified by **the QTXALEX** guidance variable.

The default value for **the QTXALEX** guidance variable is 0, indicating that CPExpert should produce Rule DB2-303 when any locks were escalated to exclusive mode.

The following example illustrates the output from Rule DB2-303:

```
RULE DB2-303: LOCKS WERE ESCALATED TO EXCLUSIVE MODE

   Lock escalation releases a large number of page or row locks, held by
   an application process on a single table or table space, to acquire a
   table or table space lock, or a set of partition locks, of mode S or X.
   Escalation can cause unpredictable response times.  Lock escalation
   should happen only when an application process updates or references
   (if repeatable read is used) more pages or rows than it normally does.
   This situation occurred during the intervals shown below:

                                     NUMBER OF         NUMBER OF LOCKS
   MEASUREMENT INTERVAL            LOCK REQUESTS    ESCALATED TO EXCLUSIVE
   17:14-17:44, 16SEP1999              121,040                  6
```

Suggestion: If Rule DB2-303 is produced frequently, you should consider the following alternatives:

- You might alter the table to increase LOCKMAX and thus decrease the number of escalations.
- Alternatively, you might revise the application to ensure that it commits more frequently.
- Alternatively, you might let the process that is causing the lock escalations begin by locking the entire table space, using the statement LOCK TABLE. This would prevent concurrency, but it is a reasonable solution for some end-of-month or end-of-year situations when a process updates more pages than it normally does.
- Alternatively, you can alter CPExpert's analysis by modifying the **QTXALEX** guidance variable in USOURCE(DB2GUIDE).

Reference: DB2 for OS/390 Version 3: Administration Guide
Section 2.4.1.23 (IRLM Panel 2: DSNTIPJ)
Section 5.7.4.5.3 (Lock Escalation)

DB2 for OS/390 Version 4: Administration Guide
Section 5.7.4.5.3 (Lock Escalation)
Section 5.7.6.3.3 (LOCKMAX Clause)

DB2 for OS/390 Version 5: Administration Guide
Section 5.7.4.5.3 (Lock Escalation)
Section 5.7.5.3.3 (LOCKMAX Clause)

DB2 for OS/390 Version 6: Administration Guide |
Section 5.7.4.5.3 (Lock Escalation) |
Section 5.7.5.3.3 (LOCKMAX Clause) |

Rule DB2-305: Lock escalation was not effective

Finding: Lock escalation was not effective because a relatively large number of lock escalations resulted in timeout suspensions or deadlocks.

Impact: This finding can have a MEDIUM IMPACT, or HIGH IMPACT on the performance of the DB2 subsystem.

Discussion: Lock escalation is the promotion of a lock from a row or page lock to a table space lock because the number of page locks concurrently held on a given resource exceeds a preset limit.

Lock escalation releases a large number of page or row locks, held by an application process on a single table or table space, to acquire a table or table space lock, or a set of partition locks, of mode S or X.

Lock escalation balances concurrency with performance by using page or row locks while a process accesses relatively few pages or rows, then changing to table space, table, or partition locks when the process accesses many pages or rows.

- In individual application processes lock only at the row or page level, other application processes can access the non-locked parts of the tablespace or table. This allows concurrency of access.

- Page or row locking uses *significant* processing time. Once the performance cost of locking is unacceptable, DB2 changes (or escalates) the locks to lock the entire table space, table, or partition (thus holding a single lock) and releases all the individual page or row locks. This lock escalation can reduce concurrency because other applications *may* not be able to access the table space, table, or partition (depending on the locking required). However, the lock escalation improves performance because the overhead of managing many locks is exchanged for the overhead of managing only a single lock.

IBM states that, as a rough estimate, lock escalation is not effective if more than one quarter of the lock escalations cause timeouts or deadlocks.

CPExpert computes the percent of ineffective lock escalations (PCTESC) by the following algorithm:

$$PCTESC = \frac{Timeouts + Deadlocks}{Shared\ locks\ escalated + Exclusive\ locks\ escalated}$$

©Copyright 1999, Computer Management Sciences, Inc.

CPExpert compares the computed PCTESC with the **PCTESC** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-305 when the percent of lock escalations that were not effective exceeds the value specified by the **PCTESC** guidance variable.

The default value for the **PCTESC** guidance variable is 25, indicating that CPExpert should produce Rule DB2-305 when more than 25% of the lock escalations were not effective.

The following example illustrates the output from Rule DB2-305:

```
RULE DB2-305: LOCK ESCALATION WAS NOT EFFECTIVE

   Lock escalation releases a large number of page or row locks, held by
   an application process on a single table or table space, to acquire a
   table or table space lock, or a set of partition locks, of mode S or X.
   Lock escalation balances concurrency with performance by using page or
   row locks while a process accesses relatively few pages or rows, then
   changing to table space, table, or partition locks when the process
   accesses many pages or rows.  IBM states that, as a rough estimate,
   escalation is not effective if more than one quarter of the lock
   escalations cause timeouts or deadlocks.  CPExpert detected that more
   than 25% of the escalations caused timeouts or deadlocks, during the
   intervals shown below:

                              NUMBER OF LOCKS       TIMEOUTS OR
   MEASUREMENT INTERVAL          ESCALATED            DEADLOCKS    PERCENT
    2:47- 3:17, 16SEP1999             2                  1          50.0
   11:15-11:45, 16SEP1999             2                  1          50.0
   11:45-12:15, 16SEP1999             2                  1          50.0
```

Suggestion: If Rule DB2-305 is produced regularly, CPExpert suggests that you consider the following alternatives:

- You might alter the table to increase the LOCKMAX specification. This would decrease the number of escalations. You should set the value of LOCKMAX high enough that, when lock escalation occurs, one application already holds so many locks that it significantly interferes with others. For example, if an application holds half a million locks on a table with a million rows, it probably already locks out most other applications. Yet lock escalation can prevent it from potentially acquiring another half million locks.

- You might let the process that is causing the lock escalations initially lock the entire table space, using the statement LOCK TABLE. This action would prevent concurrency, but it is a reasonable solution for some end-of-month or end-of-year situations when a process updates more pages than it normally does.

- You can alter CPExpert's analysis by modifying the **PCTESC** guidance variable in USOURCE(DB2GUIDE).

Reference: DB2 for OS/390 Version 3: Administration Guide
Section 2.4.1.23 (IRLM Panel 2: DSNTIPJ)
Section 5.7.4.5.3 (Lock Escalation)

DB2 for OS/390 Version 4: Administration Guide
Section 5.7.4.5.3 (Lock Escalation)
Section 5.7.6.3.3 (LOCKMAX Clause)

DB2 for OS/390 Version 5: Administration Guide
Section 5.7.4.5.3 (Lock Escalation)
Section 5.7.5.3.3 (LOCKMAX Clause)

DB2 for OS/390 Version 6: Administration Guide
Section 5.7.4.5.3 (Lock Escalation)
Section 5.7.5.3.3 (LOCKMAX Clause)

Rule DB2-310: Work was suspended for longer than time-out value

Finding: CPExpert detected that application process was suspended for longer than the time-out value.

Impact: This finding can have a MEDIUM IMPACT, or HIGH IMPACT on the performance of the DB2 subsystem.

Discussion: The Internal Resource Lock Manager (IRLM)[1] is responsible for managing all requests for locks and for controlling access to DB2 databases. The negative effects of locks on application processes are: *suspension*, *timeout*, and *deadlock*.

- The IRLM *suspends* an application process if it requests a lock that is already owned by another application process and the lock cannot be shared. The suspended process is said to be "waiting" for the lock.

  The suspended process resumes running (1) when the lock is available, (2) when the request is denied because of a *timeout* or *deadlock*, or (3) if some other interruption occurs. In the case of a timeout or deadlock, the process resumes to deal with the timeout or deadlock error condition.

- An application process is said to *time out* when it is terminated because it has been suspended for longer than a preset interval. This interval is determined by the value of the RESOURCE TIMEOUT parameter on the installation panel DSNTIPI. The RESOURCE TIMEOUT specifies a minimum number of seconds before a timeout can occur. A small value can cause a large number of timeouts. With a larger value, suspended processes more often resume normally, but they remain inactive for longer periods.

  When an application process times out, DB2 terminates the process, issues two messages to the console, and returns SQLCODE -911 or -913 to the process.

- A *deadlock* is an unresolvable contention for the use of a resource such as a table or an index. A deadlock occurs when two or more application processes each hold locks on resources that the others need and without which they cannot proceed. Deadlocks are analyzed by Rule DB2-311.

[1] Also known as the IMS Resource Lock Manager, IMS/VS Resource Lock Manager, or Inter-Region Lock Manager depending on the IBM document you reference.

©Copyright 1999, Computer Management Sciences, Inc.

CPExpert compares the QTXATIM variable in DB2STATS (the number of times work was suspended for longer than the time-out value) with the **QTXATIM** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-310 when the number of times work timed out exceeds the value specified by **the QTXATIM** guidance variable.

The default value for the **QTXATIM** guidance variable is 0, indicating that CPExpert should produce Rule DB2-310 when any time-outs occurred.

The following example illustrates the output from Rule DB2-310:

```
RULE DB2-310: WORK WAS SUSPENDED FOR LONGER THAN TIMEOUT VALUE

   A timeout occurs when an application process is suspended for longer
   than a preset value (the value is specified as the RESOURCE TIMEOUT
   on the DSNTIPI panel).  There is a trade off between having a small
   time-out value (and experiencing a relatively large number of suspends
   that time-out), versus having a large time-out value with the belief
   that suspended processes will resume normally (but the processes will
   remain inactive for longer periods). This situation occurred during
   the intervals shown below:

                                 NUMBER OF        LOCK          LOCK
   MEASUREMENT INTERVAL        LOCK REQUESTS   SUSPENSIONS    TIMEOUTS
   21:44-21:59, 08SEP1998        2,819,800           43           19
```

Suggestion: IBM provides the following suggestions on ways to prevent timeouts or to minimize the effect of timeouts:

- If statistics trace class 3 is active, DB2 writes a “Lock timeout details” trace record (IFCID 0196). Timeouts do not occur often, so this trace record should be analyzed to identify the involved in the timeout. This information is located in MXG file T102S196.

- If a task is waiting or is swapped out and the unit of work has not been committed, then it still holds locks. When a system is heavily loaded, contention for processing, I/O, and storage can cause waiting. Consider reducing the number of initiators, increasing the priority for the DB2 tasks, and providing more processing, I/O, or storage resources.

- Make sure that the IRLM has a high MVS-dispatching priority. IBM recommends that IRLM come next after VTAM and before DB2. If you are running in Goal Mode, you should assign IRLM to SYSSTC, so it will have high dispatching priority.

- Access data in a consistent order: When different applications access the same data, try to make them do so in the same sequence. For example, make both access rows 1,2,3,5 in that order. In that case, the first application to access the data delays the second, but the two applications cannot deadlock. For the same reason, try to make different applications access the same tables in the same order.

- Commit work as soon as Is practical. To avoid unnecessary lock contentions, issue a COMMIT statement as soon as possible after reaching a point of consistency, even in read-only applications. To prevent unsuccessful SQL statements (such as PREPARE) from holding locks, issue a ROLLBACK statement after a failure. Statements issued through SPUFI can be committed immediately by the SPUFI autocommit feature.

- Retry an application after deadlock or timeout. Include logic in a batch program so that it retries an operation after a deadlock or timeout. That could help you recover from the situation without assistance from operations personnel. Field SQLERRD(3) in the SQLCA returns a reason code that indicates whether a deadlock or timeout occurred.

- Close cursors. If you define a cursor using the WITH HOLD option, the locks it needs can be held past a commit point. Use the CLOSE CURSOR statement as soon as possible in your program, to release those locks and free the resources they hold.

- Bind plans with ACQUIRE(USE): That choice is best for concurrency. Packages are always bound with ACQUIRE(USE), by default. ACQUIRE(ALLOCATE) gives better protection against deadlocks for a high-priority job; if you need that option, you might want to bind all DBRMs directly to the plan.

- Bind with ISOLATION(CS) and CURRENTDATA(NO) Typically: ISOLATION(CS) lets DB2 release acquired locks as soon as possible. CURRENTDATA(NO) lets DB2 avoid acquiring locks as often as possible. After that, in order of decreasing preference for concurrency, use these bind options:

- ISOLATION(CS) with CURRENTDATA(YES), when data you have accessed must not be changed before your next FETCH operation.

- ISOLATION(RS), when rows you have accessed must not be changed before your application commits or rolls back. However, you do not care if other application processes insert additional rows.

- ISOLATION(RR), when rows you have accessed must not be changed before your application commits or rolls back. New rows cannot be inserted into the answer set.

- Use ISOLATION(UR) Cautiously: UR isolation acquires almost no locks. It is fast and causes little contention, but it reads uncommitted data. Do not use it unless you are sure that your applications and end users can accept the logical inconsistencies that can occur.

- If you can define more ECSA, then start the IRLM with PC=NO rather than PC=YES. You can make this change without changing your application process. This change can also reduce processing time.

- Alternatively, you can revise the application so that it issues COMMIT more frequently.

You should review the sections titled "Tuning Your Use of Locks" in IBM's DB2 *Administration Guides* for more suggestions from IBM on how to minimize or eliminate the negative effects of locks.

You can alter CPExpert's analysis by modifying the **QTXATIM** guidance variable in USOURCE(DB2GUIDE).

Reference: DB2 for OS/390 Version 3: Administration Guide
Section 7.8 (Improving Concurrency)

DB2 for OS/390 Version 4: Administration Guide
Section 5.7 (Improving Concurrency)

DB2 for OS/390 Version 5: Administration Guide
Section 5.7 (Improving Concurrency)

DB2 for OS/390 Version 6: Administration Guide
Section 5.7 (Improving Concurrency)

Rule DB2-311: Deadlocks were detected

Finding: Deadlocks were detected in the DB2 interval statistics data.

Impact: This finding can have a MEDIUM IMPACT, or HIGH IMPACT on the performance of the DB2 subsystem.

Discussion: A deadlock is an unresolvable contention for the use of a resource such as a table or an index. A deadlock occurs when two or more application processes each hold locks on resources that the others need and without which they cannot proceed.

The *DB2 Administration Guides* give the following example of a deadlock situation:

- Jobs EMPLJCHG and PROJNCHG are two transactions. Job EMPLJCHG accesses table M, and acquires an exclusive lock for page B, which contains record 000300.

- Job PROJNCHG accesses table N, and acquires an exclusive lock for page A, which contains record 000010.

- Job EMPLJCHG requests a lock for page A of table N while still holding the lock on page B of table M. The job is suspended, because job PROJNCHG is holding an exclusive lock on page A.

- Job PROJNCHG requests a lock for page B of table M while still holding the lock on page A of table N. The job is suspended, because job EMPLJCHG is holding an exclusive lock on page B. The situation is a deadlock.

After a preset time interval (the value of DEADLOCK TIME), DB2 will detect the deadlock situation. DB2 can roll back the current unit of work for one of the processes, or DB2 can request a process to terminate.

It is possible for two processes to be running on separate DB2 subsystems, each trying to access a resource at the other location. In that case, neither subsystem can detect that the two processes are in deadlock; the situation resolves only when one process times out.

Either of these actions frees the locks and allows the remaining processes to continue. However, the rollback of the current unit or work, or termination of a process can seriously effect performance. Not only does

©Copyright 1999, Computer Management Sciences, Inc.

the performance of the deadlock processes suffer, but significant system overhead can be incurred.

CPExpert compares the QTXTDEA variable in DB2STATS (the number of deadlocks encountered) with **the QTXTDEA** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-311 when the number of deadlocks encountered exceeds the value specified by **the QTXTDEA** guidance variable.

The default value for **the QTXTDEA** guidance variable is 0, indicating that CPExpert should produce Rule DB2-311 when any deadlocks were detected.

The following example illustrates the output from Rule DB2-311:

```
RULE DB2-311: DEADLOCKS WERE DETECTED

   A deadlock occurs when two or more application processes each hold
   locks on resources that the others need and without which they cannot
   proceed.  DB2 scans for deadlocked processes at regular intervals,
   with the scan interval set by the DEADLOCK TIME on Installation Panel
   DSNTIPJ.  Upon detecting a deadlock, DB2 can roll back the current unit
   of work for one of the processes or request a process to terminate.
   Deadlocks can significantly affect performance and should be avoided.
   This situation occurred during the intervals shown below:

                                  LOCK       LOCK       LOCK
   MEASUREMENT INTERVAL          REQUESTS  SUSPENSION  TIMEOUTS  DEADLOCKS
   21:44-21:59, 08SEP1998       2,819,800       43         19        5
```

Suggestion: If application processes encounter deadlocks, CPExpert suggests that you consider the following alternatives:

- If statistics trace class 3 is active, DB2 writes a “Unit of Work involved in deadlock” trace record (IFCID 0172). Deadlocks do not occur often, so little overhead is involved in producing this trace record. This trace record should be analyzed to identify the unit of work involved in the deadlocks. This information is located in MXG file T102S172.

- As mentioned earlier, DB2 scans for deadlocked processes at regular intervals, as specified by the DEADLOCK TIME on Installation Panel DSNTIPJ. The default for the DEADLOCK TIME is 5 seconds.

  IBM recommends that this default be adjusted so DB2 can detect deadlocks as quickly as possible. IBM recommends that, in most cases, the default be changed to specify a value of 1.

- If a task is waiting or is swapped out and the unit of work has not been committed, then it still holds locks. When a system is heavily loaded, contention for processing, I/O, and storage can cause waiting. Consider reducing the number of initiators, increasing the priority for the DB2 tasks, and providing more processing, I/O, or storage resources.

- Make sure that the IRLM has a high MVS-dispatching priority. IBM recommends that IRLM come next after VTAM and before DB2. If you are running in Goal Mode, you should assign IRLM to SYSSTC, so it will have high dispatching priority.

- Access data in a consistent order: When different applications access the same data, try to make them do so in the same sequence. For example, make both access rows 1,2,3,5 in that order. In that case, the first application to access the data delays the second, but the two applications cannot deadlock. For the same reason, try to make different applications access the same tables in the same order.

- Commit work as soon as Is practical. To avoid unnecessary lock contentions, issue a COMMIT statement as soon as possible after reaching a point of consistency, even in read-only applications. To prevent unsuccessful SQL statements (such as PREPARE) from holding locks, issue a ROLLBACK statement after a failure. Statements issued through SPUFI can be committed immediately by the SPUFI autocommit feature.

- Retry an application after deadlock or timeout. Include logic in a batch program so that it retries an operation after a deadlock or timeout. That could help you recover from the situation without assistance from operations personnel. Field SQLERRD(3) in the SQLCA returns a reason code that indicates whether a deadlock or timeout occurred.

- Close cursors. If you define a cursor using the WITH HOLD option, the locks it needs can be held past a commit point. Use the CLOSE CURSOR statement as soon as possible in your program, to release those locks and free the resources they hold.

- Bind plans with ACQUIRE(USE): That choice is best for concurrency. Packages are always bound with ACQUIRE(USE), by default. ACQUIRE(ALLOCATE) gives better protection against deadlocks for a high-priority job; if you need that option, you might want to bind all DBRMs directly to the plan.

- Bind with ISOLATION(CS) and CURRENTDATA(NO) Typically: ISOLATION(CS) lets DB2 release acquired locks as soon as possible. CURRENTDATA(NO) lets DB2 avoid acquiring locks as often as

possible. After that, in order of decreasing preference for concurrency, use these bind options:

- ISOLATION(CS) with CURRENTDATA(YES), when data you have accessed must not be changed before your next FETCH operation.

- ISOLATION(RS), when rows you have accessed must not be changed before your application commits or rolls back. However, you do not care if other application processes insert additional rows.

- ISOLATION(RR), when rows you have accessed must not be changed before your application commits or rolls back. New rows cannot be inserted into the answer set.

- Use ISOLATION(UR) Cautiously: UR isolation acquires almost no locks. It is fast and causes little contention, but it reads uncommitted data. Do not use it unless you are sure that your applications and end users can accept the logical inconsistencies that can occur.

- If you can define more ECSA, then start the IRLM with PC=NO rather than PC=YES. You can make this change without changing your application process. This change can also reduce processing time.

- Alternatively, you can revise the application so that it issues COMMIT more frequently.

You should review the sections titled *"Tuning Your Use of Locks"* in IBM's DB2 *Administration Guides* for more suggestions from IBM on how to minimize or eliminate the negative effects of locks.

- You can alter CPExpert's analysis by modifying the **QTXTDEA** guidance variable in USOURCE(DB2GUIDE).

Reference: DB2 for OS/390 Version 3: Administration Guide
Section 7.8.1 (Aspects of Transaction Locking)
Section 7.8.4 (Tuning Your Use of Locks)

DB2 for OS/390 Version 4: Administration Guide
Section 5.7.3 (Basic Recommendations to Promote Concurrency)
Section 5.7.4 (Aspects of Transaction Locks)
Section 5.7.5 (Tuning Your Use of Locks)

DB2 for OS/390 Version 5: Administration Guide
Section 5.7.3 (Basic Recommendations to Promote Concurrency)
Section 5.7.4 (Aspects of Transaction Locks)

©Copyright 1999, Computer Management Sciences, Inc.

Section 5.7.5 (Tuning Your Use of Locks)

DB2 for OS/390 Version 6: Administration Guide
Section 5.7.3 (Basic Recommendations to Promote Concurrency)
Section 5.7.4 (Aspects of Transaction Locks)
Section 5.7.5 (Tuning Your Use of Locks)

Rule DB2-401: Resynchronization connections attempted

Finding: DB2 attempted to resynchronize indoubt units of work with remote DBMS.

Impact: This finding can have a MEDIUM IMPACT, or HIGH IMPACT on the performance of the DB2 subsystem.

Discussion: When communicating with a remote DBMS, indoubt units of recovery can result from failure with either the participant or coordinator or with the communication link between them even if the systems themselves have not failed.

If DB2 loses its connection to another system, it normally attempts to recover all inconsistent objects after restart. The information needed to resolve indoubt units of recovery must come from the coordinating system.

A large number of resynchronization connection attempts often indicate that there are network or system problems.

CPExpert compares the QDSTRSAT variable in DB2STATS (the number of resynchronization connections attempted) with the **QDSTRSAT** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-401 when the number of resynchronization connections attempted exceeds the value specified by the **QDSTRSAT** guidance variable.

The default value for the **QDSTRSAT** guidance variable is 0, indicating that CPExpert should produce Rule DB2-401 when any resynchronization connections attempted.

Suggestion: You should consider the following alternatives if Rule DB2-401 is produced regularly:

- Review the statistics related to system failure on either the participant or coordinator. If no system failures occurred, the problems are related to the network.
- Review the network and system statistics to determine if network or problems exist. It is likely that you will need to review the statistics at both the coordinator and participant locations.
- You can alter CPExpert's analysis by modifying the **QDSTRSAT** guidance variable in USOURCE(DB2GUIDE).

©Copyright 1999, Computer Management Sciences, Inc.

Reference: DB2 for OS/390 Version 3: Administration guide
Section 6.7.5.3 (Resolution of Indoubt Units of Recovery)
Section 6.7.16 (Resolving Indoubt Threads)

DB2 for OS/390 Version 4: Administration guide
Section 4.5.2 (Resolving Indoubt Units of Recovery)
Section 4.7.17 (Resolving Indoubt Threads)

DB2 for OS/390 Version 5: Administration Guide
Section 4.5.2 (Resolving Indoubt Units of Recovery)
Section 4.7.18 (Resolving Indoubt Threads)

DB2 for OS/390 Version 6: Administration Guide
Section 4.5.2 (Resolving Indoubt Units of Recovery)
Section 4.7.18 (Resolving Indoubt Threads)

©Copyright 1999, Computer Management Sciences, Inc.

Rule DB2-402: Network problems might exist

Finding: Based on the number of resynchronization connections attempted and resynchronization connections that succeeded, CPExpert believes that network problems existed.

Impact: This finding can have a MEDIUM IMPACT, or HIGH IMPACT on the performance of the DB2 subsystem.

Discussion: When communicating with a remote DBMS, indoubt units of recovery can result from failure with either the participant or coordinator or with the communication link between them even if the systems themselves have not failed.

If DB2 loses its connection to other systems, it normally attempts to recover all inconsistent objects after restart. The information needed to resolve indoubt units of recovery must come from the coordinating system.

When the number of successful resynchronization connections exceeds the number of resynchronization attempts, this indicates that there are network problems.

CPExpert computes the percent of reconnection attempts that succeeded. The calculation uses data in DB2STATB, and is performed as shown below:

$$\textit{Percent successful reconnection attempts} = \frac{\textit{Successful reconnection attempts}}{\textit{Reconnection attempts}}$$

CPExpert compares the computed percent successful reconnection attempts with the **PCTRSAT** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-402 when the percent of reconnection attempts that succeeded is **greater than** the value specified by the **PCTRSAT** guidance variable.

The default value for the **PCTRSAT** guidance variable is 75%, indicating that Rule DB2-402 should be produced when the percent reconnection attempts that succeeded was **greater than 75%** of the reconnection attempts.

CPExpert compares the QDSTRSSU variable in DB2STATS (the number of resynchronization connection that succeeded with all remote locations) with the QDSTRSAT variable in DB2STATS (the number of

resynchronization connections attempted with all remote locations). CPExpert produces Rule DB2-402 when the number of successful resynchronization exceeds the resynchronization connections attempted.

Suggestion: You should consider the following alternatives if Rule DB2-402 is produced regularly:

- Review the network statistics to determine if network problems exist. It is likely that you will need to review the statistics at both the coordinator and participant locations.

- You can alter CPExpert's analysis by "turning off" this rule using the procedures outlined in Section 3 of this document.

Reference: DB2 for OS/390 Version 3: Administration guide
Section 6.7.5.3 (Resolution of Indoubt Units of Recovery)
Section 6.7.16 (Resolving Indoubt Threads)

DB2 for OS/390 Version 4: Administration guide
Section 4.5.2 (Resolving Indoubt Units of Recovery)
Section 4.7.17 (Resolving Indoubt Threads)

DB2 for OS/390 Version 5: Administration Guide
Section 4.5.2 (Resolving Indoubt Units of Recovery)
Section 4.7.18 (Resolving Indoubt Threads)

DB2 for OS/390 Version 6: Administration Guide
Section 4.5.2 (Resolving Indoubt Units of Recovery)
Section 4.7.18 (Resolving Indoubt Threads)

Rule DB2-403: Threads became indoubt with the remote location

Finding: Threads became indoubt with the remote location, indicating possible network problems.

Impact: This finding can have a MEDIUM IMPACT, or HIGH IMPACT on the performance of the DB2 subsystem.

Discussion: When communicating with a remote DBMS, indoubt units of recovery can result from failure with either the participant or coordinator or with the communication link between them even if the systems themselves have not failed.

If DB2 loses its connection to other systems, it normally attempts to recover all inconsistent objects after restart. The information needed to resolve indoubt units of recovery must come from the coordinating system.

When the number of successful resynchronization connections exceeds the number of resynchronization attempts, this indicates that there are network problems.

CPExpert compares the QLSTINDT variable in DB2STATS (the number of threads that became indoubt with the remote location as coordinator) with the **QLSTINDT** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-403 when the number of (the number of threads that became indoubt exceeds the value specified by the **QLSTINDT** guidance variable.

The default value for the **QLSTINDT** guidance variable is 0, indicating that CPExpert should produce Rule DB2-403 when any (the number of threads that became indoubt with the remote location as coordinator.

Suggestion: You should consider the following alternatives if Rule DB2-403 is produced regularly:

- Review the network statistics to determine if network problems exist.
- You can alter CPExpert's analysis by "turning off" this rule using the procedures outlined in Section 3 of this document.

©Copyright 1999, Computer Management Sciences, Inc.

Reference: DB2 for OS/390 Version 3: Administration guide
Section 6.7.5.3 (Resolution of Indoubt Units of Recovery)
Section 6.7.16 (Resolving Indoubt Threads)

DB2 for OS/390 Version 4: Administration guide
Section 4.5.2 (Resolving Indoubt Units of Recovery)
Section 4.7.17 (Resolving Indoubt Threads)

DB2 for OS/390 Version 5: Administration Guide
Section 4.5.2 (Resolving Indoubt Units of Recovery)
Section 4.7.18 (Resolving Indoubt Threads)

DB2 for OS/390 Version 6: Administration Guide
Section 4.5.2 (Resolving Indoubt Units of Recovery)
Section 4.7.18 (Resolving Indoubt Threads)

©Copyright 1999, Computer Management Sciences, Inc.

Rule DB2-421: Remote create thread requests rejected - ZPARM limit reached

Finding: The DB2 statistics showed that remote create thread requests were rejected by DB2, and the conversation was deallocated, because the MAX REMOTE CONNECTED limit had been reached.

Impact: This finding can have a MEDIUM IMPACT, or HIGH IMPACT on the performance of the DB2 subsystem.

Discussion: The DB2 DSNTIPE panel is used to limit the number of allied threads and database access threads that can be allocated concurrently.

- The MAX USERS value is used to specify the maximum number of allied threads. An allied thread is each active CICS transaction that can access DB2, each TSO user (whether running a DSN command or a DB2 request from QMF), each batch job (whether running a DSN command or a DB2 utility), each IMS region that can access DB2, each active CICS transaction that can access DB2, and each task connected to DB2 through the call attachment facility.

- The MAX REMOTE ACTIVE value is used to specify the maximum number of active database access threads (DBATs) that can be allocated concurrently. DBATs are allocated on demand on behalf of an SQL request started at a remote subsystem.

The **total** number of active threads[1] accessing data that can be allocated concurrently is the sum of the MAX USERS value and the MAX REMOTE ACTIVE. The maximum allowable value for this sum is 2000. When the number of users attempting to access DB2 exceeds the number specified, excess plan allocation requests are queued.

The MAX REMOTE CONNECTED value is used to specify the maximum number of database access threads, both active and inactive, that can be connected to the DB2 subsystem concurrently.

During connection or signon, DB2 checks the MAX REMOTE CONNECTED limit you specified on panel DSNTIPE to see if it has been reached. If the limit has been reached, DB2 does not create an active or an

[1] Database access threads can be active or both active and inactive, depending on what you specified for the DDF THREADS field on installation panel DSNTIPR. If the DDF THREADS INACTIVE option is selected, up to 25,000 can be specified as the maximum number of active and inactive database access threads that can concurrently exist within DB2, using the MAX REMOTE CONNECTED field of panel DSNTIPE.

©Copyright 1999, Computer Management Sciences, Inc.

inactive thread; the create thread request is rejected, and the conversation is deallocated. If the MAX REMOTE CONNECTED limit has not been reached, the thread creation process continues[2].

CPExpert compares the QDSTQCRT variable in DB2STATS (the number of conversations that were deallocated because ZPARMS limit was reached) with the **QDSTQCRT** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-421 when the number of conversations that were deallocated exceeds the value specified by the **QDSTQCRT** guidance variable.

The default value for the **QDSTQCRT** guidance variable is 0, indicating that CPExpert should produce Rule DB2-421 when any conversations that were deallocated because ZPARMS limit was reached.

Suggestion: You should consider the following alternatives if Rule DB2-421 is produced regularly:

- Consider increasing the MAX REMOTE CONNECTED value specified on the DSNTIPE panel. This value should be increased unless real storage is the limiting factor.

- The MAX REMOTE CONNECTED value on the DSNTIPE panel normally should be used along with the DDF THREADS value on the DSNTIPR panel.

  - The DDF THREADS ACTIVE specification is the *default* on the DSNTIPR panel. If ACTIVE is specified, then the thread remains active. This provides the best performance but consumes system resources.

  - IBM recommends that the DDF THREADS INACTIVE specification should normally be used if your installation must support a large number of connections. The DDF THREADS INACTIVE specification is used to specify whether to make a thread active or inactive after it successfully commits or rolls back and holds no database locks or cursors.

    A database access thread that does not hold any cursors or database resources (such as storage) is known as an inactive thread. There are many advantages of inactive (or :sometimes active") threads:

[2] Note that the remote thread might be queued if other limits have been exceeded. Queuing because of these limits are described in Rule DB2-422 and Rule DB2-423.

- You can leave an application that is running on a workstation connected to DB2 from the time the application is first activated until the workstation is shut down, thus avoiding the delay of repeated connections.

- DB2 can support a larger number of DDF threads (a total of 25,000 active and inactive threads can be supported instead of 2,000).

- Less storage is used for each DDF thread.

- You get an accounting trace record (IFCID 0003) each time a thread becomes inactive rather than once for the entire time you are connected. When an inactive thread becomes active, the accounting fields for that thread are initialized again. As a result, the accounting record contains information about active threads only. This makes it easier to study how distributed applications are performing.

- Each time a thread becomes inactive, workload manager resets the information it maintains on that thread. The next time that thread is activated, workload manager begins managing to the goals you have set for transactions that run in that service class. If you use multiple performance periods, it is possible to favor short-running units of work that use fewer resources while giving fewer resources over time to long running units of work

- You can alter CPExpert's analysis by modifying the **QDSTQCRT** guidance variable in USOURCE(DB2GUIDE).

Reference: DB2 for OS/390 Version 3: Administration Guide
Section 2.4.1.13 (Storage Sizes Panel: DSNTIPE)
Section 2.4.1.29 (Distributed Data Facility: DSNTIPR)
Section 7.7.2.3 (How a Database Access Thread Is Created)

DB2 for OS/390 Version 4: Installation Guide
Section 2.5.1.19 (Thread Management Panel: DSNTIPE)
Section 2.5.1.32 (Distributed Data Facility: DSNTIPR)

DB2 for OS/390 Version 4: Administration Guide
Section 5.6.3.4 (How a Database Access Thread Is Created)

DB2 for OS/390 Version 5: Installation Guide
Section 2.5.13 (Thread Management Panel: DSNTIPE)
Section 2.5.27 (Distributed Data Facility: DSNTIPR)

©Copyright 1999, Computer Management Sciences, Inc.

DB2 for OS/390 Version 5: Administration Guide
Section 5.6.3.4 (How a Database Access Thread Is Created)

DB2 for OS/390 Version 6: Installation Guide
Section 2.5.12 (Thread Management Panel: DSNTIPE)
Section 2.5.27 (Distributed Data Facility: DSNTIPR)

DB2 for OS/390 Version 6: Administration Guide
Section 5.6.2 (Allied thread allocation)

Rule DB2-422: Remote threads queued by DDF - ZPARMS limit reached

Finding: The DB2 statistics showed that remote threads were queued by DDF waiting for allocation, because the MAX REMOTE ACTIVE limit had been reached.

Impact: This finding can have a MEDIUM IMPACT, or HIGH IMPACT on the performance of the DB2 subsystem.

Discussion: The DB2 DSNTIPE panel is used to limit the number of allied threads and database access threads that can be allocated concurrently.

- The MAX USERS value is used to specify the maximum number of allied threads. An allied thread is each active CICS transaction that can access DB2, each TSO user (whether running a DSN command or a DB2 request from QMF), each batch job (whether running a DSN command or a DB2 utility), each IMS region that can access DB2, each active CICS transaction that can access DB2, and each task connected to DB2 through the call attachment facility.

- The MAX REMOTE ACTIVE value is used to specify the maximum number of active database access threads (DBATs) that can be allocated concurrently. DBATs are allocated on demand on behalf of an SQL request started at a remote subsystem.

The **total** number of active threads[1] accessing data that can be allocated concurrently is the sum of the MAX USERS value and the MAX REMOTE ACTIVE. The maximum allowable value for this sum is 2000. When the number of users attempting to access DB2 exceeds the number specified, excess plan allocation requests are queued.

The MAX REMOTE CONNECTED value is used to specify the maximum number of database access threads, both active and inactive, that can be connected to the DB2 subsystem concurrently.

During connection or signon, DB2 checks the MAX REMOTE CONNECTED limit you specified on panel DSNTIPE to see if it has been reached. If the limit has been reached, DB2 does not create an active or an inactive thread; the create thread request is rejected, and the conversation

[1] Database access threads can be active or both active and inactive, depending on what you specified for the DDF THREADS field on installation panel DSNTIPR. If the DDF THREADS INACTIVE option is selected, up to 25,000 can be specified as the maximum number of active and inactive database access threads that can concurrently exist within DB2, using the MAX REMOTE CONNECTED field of panel DSNTIPE.

©Copyright 1999, Computer Management Sciences, Inc.

is deallocated. If the MAX REMOTE CONNECTED limit has not been reached, the thread creation process continues.

DB2 compares the MAX REMOTE ACTIVE limit specified on panel DSNTIPE with the current number of active database access threads. If the MAX REMOTE ACTIVE limit is reached, DB2 queues the thread request until the MAX REMOTE ACTIVE value falls below the limit. Until that happens, the thread is an inactive thread. When the number of active threads falls below this MAX REMOTE ACTIVE limit, the queued inactive thread completes the thread creation process and becomes an active thread.

CPExpert compares the QDSTQDBT variable in DB2STATS (the number of times a database access thread was queued because it reached the ZPARM maximum for active remote threads) with the **QDSTQDBT** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-422 when the number of database access threads queued exceeds the value specified by the **QDSTQDBT** guidance variable.

The default value for the **QDSTQDBT** guidance variable is 0, indicating that CPExpert should produce Rule DB2-422 when any requests were queued because of the MAX REMOTE ACTIVE value.

Suggestion: You should consider the following alternatives if Rule DB2-422 is produced regularly:

- Consider increasing the MAX REMOTE ACTIVE value specified on the DSNTIPE panel. This value should be increased unless real storage is the limiting factor. When altering the MAX REMOTE ACTIVE value, you should also consider the MAX USERS value.

  As mentioned earlier, the **total** number of active threads accessing data that can be allocated concurrently is the sum of the MAX USERS value and the MAX REMOTE ACTIVE. The maximum allowable value for this sum is 2000. When the number of users attempting to access DB2 exceeds the number specified, excess plan allocation requests are queued.

  The MAX USERS and MAX REMOTE ACTIVE values should be set to provide good response time without wasting resources (such as virtual and real storage).

  - Fewer threads than needed under-utilize the processor and cause queuing for threads.

- More threads than needed do not improve the response time. They require more real storage for the additional threads and might cause more paging with resulting performance degradation.

If real storage is the limiting factor, set MAX USERS and MAX REMOTE ACTIVE according to the available storage. For more information on storage, refer to Section 2 of the *DB2 Installation Guide.*

- Please refer to Rule DB2-421 for a discussion of the advantages of specifying a relatively large value for the MAX REMOTE CONNECTED value on the DSNTIPE panel, which normally should be used along with the DDF THREADS value on the DSNTIPR panel.

- You can alter CPExpert's analysis by modifying the **QDSTQDBT** guidance variable in USOURCE(DB2GUIDE).

Reference: DB2 for OS/390 Version 3: Administration Guide
Section 2.4.1.13 (Storage Sizes Panel: DSNTIPE)
Section 2.4.1.29 (Distributed Data Facility: DSNTIPR)
Section 7.7.2.3 (How a Database Access Thread Is Created)

DB2 for OS/390 Version 4: Installation Guide
Section 2.5.1.19 (Thread Management Panel: DSNTIPE)
Section 2.5.1.32 (Distributed Data Facility: DSNTIPR)

DB2 for OS/390 Version 4: Administration Guide
Section 5.6.3.4 (How a Database Access Thread Is Created)

DB2 for OS/390 Version 5: Installation Guide
Section 2.5.13 (Thread Management Panel: DSNTIPE)
Section 2.5.27 (Distributed Data Facility: DSNTIPR)

DB2 for OS/390 Version 5: Administration Guide
Section 5.6.3.4 (How a Database Access Thread Is Created)

DB2 for OS/390 Version 6: Installation Guide
Section 2.5.12 (Thread Management Panel: DSNTIPE)
Section 2.5.27 (Distributed Data Facility: DSNTIPR)

DB2 for OS/390 Version 6: Administration Guide
Section 5.6.2 (Allied thread allocation)

Rule DB2-423: Database access threads were queued - ZPARM limit reached

Finding: The DB2 statistics showed that database access threads were queued because the MAX REMOTE ACTIVE limit had been reached.

Impact: This finding can have a MEDIUM IMPACT, or HIGH IMPACT on the performance provided to remote users of DB2.

Discussion: The DB2 DSNTIPE panel is used to limit the number of allied threads and database access threads that can be allocated concurrently.

- The MAX USERS value is used to specify the maximum number of allied threads. An allied thread is each active CICS transaction that can access DB2, each TSO user (whether running a DSN command or a DB2 request from QMF), each batch job (whether running a DSN command or a DB2 utility), each IMS region that can access DB2, each active CICS transaction that can access DB2, and each task connected to DB2 through the call attachment facility.

- The MAX REMOTE ACTIVE value is used to specify the maximum number of active database access threads (DBATs) that can be allocated concurrently. DBATs are allocated on demand on behalf of an SQL request started at a remote subsystem.

The **total** number of active threads[1] accessing data that can be allocated concurrently is the sum of the MAX USERS value and the MAX REMOTE ACTIVE. The maximum allowable value for this sum is 2000. When the number of users attempting to access DB2 exceeds the number specified, excess plan allocation requests are queued.

During thread creation for remote users, DB2 compares the MAX REMOTE ACTIVE limit specified on panel DSNTIPE with the current number of active database access threads. If the MAX REMOTE ACTIVE limit is reached, DB2 queues the thread request until the MAX REMOTE ACTIVE value falls below the limit. When the number of active threads falls below this MAX REMOTE ACTIVE limit, the queued inactive thread completes the thread creation process and becomes an active thread.

[1] Database access threads can be active or both active and inactive, depending on what you specified for the DDF THREADS field on installation panel DSNTIPR. If the DDF THREADS INACTIVE option is selected, up to 25,000 can be specified as the maximum number of active and inactive database access threads that can concurrently exist within DB2, using the MAX REMOTE CONNECTED field of panel DSNTIPE.

©Copyright 1999, Computer Management Sciences, Inc.

CPExpert compares the QDSTQDBT variable in DB2STATS (the number of times a database access thread was queued because it had reached the MAX REMOTE ACTIVE value) with the **QDSTQDBT** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-423 when the number of database access threads queued exceeds the value specified by the **QDSTQDBT** guidance variable.

The default value for the **QDSTQDBT** guidance variable is 0, indicating that CPExpert should produce Rule DB2-423 when any requests were queued because of the CTHREAD value.

Suggestion: You should consider the following alternatives if Rule DB2-423 is produced regularly:

- The MAX USERS and MAX REMOTE ACTIVE values should be set to provide good response time without wasting resources (such as virtual and real storage).

  - Fewer threads than needed under-utilize the processor and cause queuing for threads.

  - More threads than needed do not improve the response time. They require more real storage for the additional threads and might cause more paging with resulting performance degradation.

  If real storage is the limiting factor, set MAX USERS and MAX REMOTE ACTIVE according to the available storage. For more information on storage, refer to Section 2 of the *DB2 Installation Guide.*

- You can alter CPExpert's analysis by modifying the **QDSTQDBT** guidance variable in USOURCE(DB2GUIDE).

Reference: DB2 for OS/390 Version 3: Administration Guide
Section 2.4.1.29 (Distributed Data Facility: DSNTIPR)
Section 7.7.2.3 (How a Database Access Thread Is Created)

DB2 for OS/390 Version 4: Installation Guide
Section 2.5.27 (Distributed Data Facility: DSNTIPR)

DB2 for OS/390 Version 4: Administration Guide
Section 5.6.3.4 (How a Database Access Thread Is Created)

DB2 for OS/390 Version 5: Installation Guide
Section 2.5.27 (Distributed Data Facility: DSNTIPR)

DB2 for OS/390 Version 5: Administration Guide
Section 5.6.3.4 How a Database Access Thread Is Created

DB2 for OS/390 Version 6: Installation Guide
Section 2.5.12 (Thread Management Panel: DSNTIPE)
Section 2.5.27 (Distributed Data Facility: DSNTIPR)

DB2 for OS/390 Version 6: Administration Guide
Section 5.6.2 (Allied thread allocation)

©Copyright 1999, Computer Management Sciences, Inc.

Rule DB2-501: Archive log read allocations exceeded guidance

Finding: Archive log read allocations exceeded guidance

Impact: This finding can have a MEDIUM IMPACT, or HIGH IMPACT on the performance of the DB2 subsystem.

Discussion: One of the major advantages of DB2 is its ability to recover in case of restart after a shutdown, in case of failure, or for other reasons. The recovery in chase of DB2 shutdown (STOP DB2) relies on checkpoint information that is created during the shutdown operation. In case of failure, application process timeout, or application process deadlock, DB2 recovers or performs a rollback based on logging it has done of changes to data and significant events.

A *unit of recovery* is the work, done by a single DB2 DBMS for an application, that changes DB2 data from one *point of consistency* to another. A point of consistency (also called, sync point or commit point) is a time when all recoverable data that an application program accesses is consistent with other data.

If failure occurs within a unit of recovery, DB2 backs out any changes to data, returning the data to its state at the start of the unit of recovery. That is, DB2 undoes the work. The SQL ROLLBACK statement, and deadlocks and timeouts cause DB2 to perform the same actions.

DB2 is able to back out changes by registering changes to data and significant events in *recovery log records*. In case of a recovery requirement, DB2 can use the recovery log records to recover or rollback any unit of work that has not been committed.

The recovery log records are placed sequentially in *output log buffers*, which are formatted as VSAM control intervals (CIs). The CIs are written to a set of predefined DASD active log data sets, which are used sequentially and recycled. Output log buffers hold recovery log records waiting to be written to the *active log*.

The output log buffers are written to an *active log data set* when the buffers become full, when the output log buffer write threshold is reached (as specified on the DSNTIPL panel), or, more often, when the DB2 subsystem forces the log buffer to be written (such as, at commit time).

©Copyright 1999, Computer Management Sciences, Inc.

Under certain conditions, DB2 copies the contents of the active log to a DASD or magnetic tape data set called the *archive log*. The process of copying active logs to archive logs is called off-loading. An off-load of an active log to an archive log can be triggered by several events. The most common[1] are when:

- An active log data set is full.
- Starting DB2 and an active log data set is full
- The command ARCHIVE LOG is issued.

When a rollback or recovery action is required and data is backed out, for optimal performance the data should be available in the output buffer or in the active log. If the data has already been off-loaded to the archive log, DB2 must read the archive log for the data set(s) being rolled back or recovered. The application process thread will be suspended for an Archive Log Read Suspension during this interval.

CPExpert compares the QJSTALR variable in DB2STATS (the number of archive log read allocations) with the **QJSTALR** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-501 when the number of archive log read allocations exceeds the value specified by the **QJSTALR** guidance variable.

The default value for the **QJSTALR** guidance variable is 0, indicating that CPExpert should produce Rule DB2-501 when any archive log read allocations occur.

Suggestion: IBM's *DB2 Administration Guides* state that for optimal performance when data is backed out, the data should still be available in the output buffer or in the active log.

- If the data has already been off-loaded to the archive log, the active log is probably too small. You should consider increasing the size of the active log.
- Alternatively, you should consider increasing the number of active log data sets.

You can alter CPExpert's analysis by modifying the **QJSTALR** guidance variable in USOURCE(DB2GUIDE).

[1] Off-load is also triggered by two uncommon events (1) an error occurring while writing to an active log data set, and (2) filling of the last unarchived active log data set.

Reference: DB2 for OS/390 Version 3: Administration Guide
Section 6.3.1 (How Database Changes Are Made)
Section 7.6.5 (DB2 Logging)

DB2 for OS/390 Version 4: Administration Guide
Section 4.3.1 (How Database Changes Are Made)
Section 5.5.5 (DB2 Logging)

DB2 for OS/390 Version 5: Administration Guide
Section 4.3.1 (How Database Changes Are Made)
Section 5.5.5 (DB2 Logging)

DB2 for OS/390 Version 6: Administration Guide
Section 4.3.1 (How Database Changes Are Made)
Section 5.5.5 (DB2 Logging)

©Copyright 1999, Computer Management Sciences, Inc.

Rule DB2-502: Archive log write allocations exceeded guidance

Finding: Archive log write allocations exceeded guidance

Impact: This finding can have a LOW IMPACT, or MEDIUM IMPACT on the performance of the DB2 subsystem.

Discussion: Please refer to Rule DB2-501 for a discussion of DB2 archive logs.

The output log buffers are written to an *active log data set* when the buffers become full, when the output log buffer write threshold is reached (as specified on the DSNTIPL panel), or, more often, when the DB2 subsystem forces the log buffer to be written (such as, at commit time).

Under certain conditions, DB2 copies the contents of the active log to a DASD or magnetic tape data set called the *archive log*. The process of copying active logs to archive logs is called off-loading. An off-load of an active log to an archive log can be triggered by several events. The most common[1] are when:

- An active log data set is full.
- Starting DB2 and an active log data set is full
- The command ARCHIVE LOG is issued.

CPExpert compares the QJSTALW variable in DB2STATS (the number of archive log write allocations) with the **QJSTALW** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-502 when the number of archive log write allocations exceeds the value specified by the **QJSTALW** guidance variable.

The default value for the **QJSTALW** guidance variable is 0, indicating that CPExpert should produce Rule DB2-502 when there were any archive log write allocations.

The following example illustrates the output from Rule DB2-502:

[1] Off-load is also triggered by two uncommon events (1) an error occurring while writing to an active log data set, and (2) filling of the last unarchived active log data set.

©Copyright 1999, Computer Management Sciences, Inc.

```
RULE DB2-502: ARCHIVE LOG WRITE ALLOCATIONS EXCEEDED GUIDANCE

   The archive log open and close activity for write allocations was higher
   than the guidance provided to CPExpert.  This finding might indicate a
   need for more or larger active log data sets.  Archive log write open
   and close activity was high during the intervals shown below:

                                            ARCHIVE LOG OPEN
   MEASUREMENT INTERVAL                  AND CLOSE WRITE ACTIVITY
    2:23- 2:53, 15SEP1999                          4
    2:53- 3:22, 15SEP1999                          2
    7:22- 7:51, 15SEP1999                          6
```

Suggestion: If Rule DB2-502 is produced regularly, you should consider the following alternatives:

- You should consider increasing the size of the active log.
- You should consider increasing the number of active log data sets.
- You can alter CPExpert's analysis by modifying the **QJSTALW** guidance variable in USOURCE(DB2GUIDE).

Reference: DB2 for OS/390 Version 3: Administration Guide
Section 6.3.1 (How Database Changes Are Made)
Section 7.6.5 (DB2 Logging)

DB2 for OS/390 Version 4: Administration Guide
Section 4.3.1 (How Database Changes Are Made)
Section 5.5.5 (DB2 Logging)

DB2 for OS/390 Version 5: Administration Guide
Section 4.3.1 (How Database Changes Are Made)
Section 5.5.5 (DB2 Logging)

DB2 for OS/390 Version 6: Administration Guide
Section 4.3.1 (How Database Changes Are Made)
Section 5.5.5 (DB2 Logging)

©Copyright 1999, Computer Management Sciences, Inc.

Rule DB2-511: Waits were caused by unavailable output log buffer

Finding: The DB2 Log Manager statistics revealed that waits were caused by an unavailable output log buffer.

Impact: This finding can have a MEDIUM IMPACT, or HIGH IMPACT on the performance of the DB2 subsystem.

Discussion: DB2 writes each log record to a DASD data set called the active log. When the active log is full, DB2 copies the contents of the active log to a DASD or magnetic tape data set called the archive log. In the case of failure, DB2 uses this data to recover.

The archive log can consist of up to 1000 data sets, each of which is a sequential data set (physical sequential) that resides on a DASD or magnetic tape volume. An archive log data set is created during the log off-load process (when an active log data set is copied to an archive log data set). The archive log can be cataloged in an integrated catalog facility catalog and protected with an MVS data set password or with resource access control facility (RACF).

DB2 logs changes made to data, and other significant events, as they occur. DB2 initially writes the changes to the log buffer, the changes are then written to the active logs, and eventually off loaded to the archive logs.

Log buffers are buffers that hold writes before they written to the active logs. If the log buffers are too small, the Log Manager must wait for buffers to become available. This means that both DB2 and the application must wait for an available log buffer.

DB2 uses two different types of requests when it logs data and event

- **NOWAIT**. NOWAIT requests are the most frequent. When data is updated, before- and after-image records are usually moved to the log buffer, and control is returned to the application. However, if no log buffer is available, the application must wait for one. When the number of log buffers used reaches the WRITE THRESHOLD value, the data is written; however, the application does not wait for the write.

  Batch jobs might log a large amount of data before they commit. The amount of data logged between FORCE requests can be larger than the total buffer size.

©Copyright 1999, Computer Management Sciences, Inc.

- FORCE: FORCE requests occur at commit time, when an application has performed database updates. If the log data set is not busy, all log buffers are written to disk. The application must wait until the write is completed. If the log data set is busy, the requests are queued until it is freed.

Waits for log buffers can significantly effect harm performance, depending on the length of the wait.

CPExpert compares the QJSTWTB variable in DB2STATS (the number of waits caused by unavailable output log buffer) with **the QJSTWTB** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-511 when the number of waits caused by unavailable output log buffer exceeds the value specified by **the QJSTWTB** guidance variable.

The default value for **the QJSTWTB** guidance variable is 0, indicating that CPExpert should produce Rule DB2-511 when any log writes had to wait for log buffers.

The following example illustrates the output from Rule DB2-511:

```
RULE DB2-511: WAITS WERE CAUSED BY UNAVAILABLE OUTPUT LOG BUFFER

   The DB2 Log Manager statistics revealed that waits were caused by an
   unavailable output log buffer.  When DB2 wants to write a log record
   and the log buffer is not available, DB2 and the application must
   wait for an available log buffer.  This finding means that you should
   increase the number of output buffers being used to write to the active
   log data set.  This finding occurred during the intervals shown below:

                                          WAITS CAUSED BY
   MEASUREMENT INTERVAL              UNAVAILABLE OUTPUT LOG BUFFER
    2:23- 2:53, 15SEP1999                            4
```

Suggestion: CPExpert suggests that you consider the following alternatives:

- You should review the specifications that were entered on the Log Data Sets panel DSNTIPL. On this panel, the most important options for tuning performance are OUTPUT BUFFER and WRITE THRESHOLD.

  - The OUTPUT BUFFER field specifies the size of the output buffer used for writing active log data sets. The maximum size of OUTBUFF is 4000 KB. The buffer must be large enough to prevent buffer shortages in order to decrease the number of forced I/O operations (forced because there are no more buffers) or wait conditions. IBM recommends that the maximum allowed value for this field be used.

- The WRITE THRESHOLD field indicates the number of contiguous 4KB output buffer pages that are allowed to fill before data is written to the active log data set. The default is 20, and IBM recommends that this default be used. IBM further recommends that you never choose a value that is greater than 20% of the number of buffers in the output buffer.

The above advice is somewhat inconsistent, although both suggestions were taken from the IBM's *DB2 Administration Guides*! Assume that the default values were used; 400 KB is the default for OUTPUT BUFFER and 20% is the default for the WRITE THRESHOLD. With these defaults, DB2 would start writing data from the output buffer to the active log when 80 KB of the buffer were filled (400 * .20). This low threshold would allow DB2 to perform I/O operations with the remaining 80% (or 320 KB) of the output buffer in reserve to receive new log requests.

If the IBM recommendation to use the maximum value for the OUTPUT BUFFER (4000 KB) and retain the 20% value for the WRITE THRESHOLD, DB2 would start writing data from the output buffer to the active log when 800 KB of the buffer were filled (4000 * .20), with 3200 KB of the output buffer in reserve to receive new log requests - these are significantly larger values than the values used with the defaults.

While the 4000 KB buffer would be excellent for performance aspects, be aware that an increase in the OUTBUFF value can increase the amount of DB2 data lost if DB2 ABENDs. Consequently, you should balance the performance benefits of not incurring I/O operations against the risk of lost data if DB2 ABENDs.

- Whenever you allocate new active log data sets, preformat them using the DSNJLOGF utility described in Section 3 of Utility Guide and Reference. This avoids the overhead of preformatting the log, which normally occurs at unpredictable times.

- Review and apply IBM's suggestions in the "Guidelines for Controlling Logging" subsection (in the "DB2 Logging" section referenced below) of the *DB2 Administration Guides*.

- You can design the DB2 log for better performance. In particular, try to:

  - Minimize device contention on the log data sets by placing data sets correctly.

  - Avoid waits that occur because no log buffer is available.

- Define enough active log data sets to prevent DB2 from waiting while a log is archived.

- Make the active log large enough that backouts do not have to use the archive log.

- Place the copy of the bootstrap data set and, if using dual active logging, the copy of the active log data sets, on volumes that are accessible on a path different than that of their primary counterparts. DB2 writes serially or in parallel to the log data sets. If a 4KB log control interval is written for the first time to DASD, the write I/Os to the log data sets are done in parallel. If the same 4KB log control interval is again written to DASD, then the write I/Os to the log data sets must be done serially to prevent any possibility of losing log data in case of I/O errors on both copies simultaneously. This improves system integrity; there is no I/O overlap in dual logging, except when the WRITE THRESHOLD value is reached, triggering full output log buffers to be written. .

- You can alter CPExpert's analysis by modifying **the QJSTWTB** guidance variable in USOURCE(DB2GUIDE).

Reference: DB2 for OS/390 Version 3: Administration Guide
Section 7.6.5 (DB2 Logging)

DB2 for OS/390 Version 4: Administration Guide
Section 5.5.5 (DB2 Logging)

DB2 for OS/390 Version 5: Administration Guide
Section 5.5.5 (DB2 Logging)

DB2 for OS/390 Version 6: Administration Guide
Section 5.5.5 (DB2 Logging)

Rule DB2-512: Log reads satisfied from active log data set

Finding: A relatively large percent of log reads were satisfied from active log data set.

Impact: This finding can have a MEDIUM IMPACT, or HIGH IMPACT on the performance of the DB2 subsystem.

Discussion: DB2 is able to back out changes by registering changes to data and significant events in *recovery log records*. In case of a recovery requirement, DB2 can use the recovery log records to recover or rollback any unit of work that has not been committed.

The recovery log records are placed sequentially in *output log buffers*, which are formatted as VSAM control intervals (CIs). The CIs are written to a set of predefined DASD active log data sets, which are used sequentially and recycled. Output log buffers hold recovery log records waiting to be written to the *active log*.

The output log buffers are written to an *active log data set* when the buffers become full, when the output log buffer write threshold is reached (as specified on the DSNTIPL panel), or, more often, when the DB2 subsystem forces the log buffer to be written (such as, at commit time).

Under certain conditions, DB2 copies the contents of the active log to a DASD or magnetic tape data set called the *archive log*. The process of copying active logs to archive logs is called off-loading.

When a rollback or recovery action is required and data is backed out, for optimal performance the data should be available in the output buffer or in the active log. It is preferable that the data be in the output buffer, but this is not always possible with an active DB2 environment. However, if a large percent of reads are satisfied from the active log, you should ensure that the output buffer is as large as possible.

CPExpert computes the percent of reads satisfied from the active log by the following algorithm:

$$\textit{Percent reads from active} \text{ log} = \frac{\textit{Reads from active} \text{ log } (\textit{QBSTRACT})}{\textit{Total reads from} \text{ log}}$$

CPExpert compares the resultant percent) with the **PCTRACT** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-512 when the percent of reads satisfied from the active log exceeds the value specified by the **PCTRACT** guidance variable.

The default value for the PCTRACT guidance variable is 25, indicating that CPExpert should produce Rule DB2-512 when more than 25% of the log reads were satisfied from the active log.

The following example illustrates the output from Rule DB2-512:

```
RULE DB2-512: LOG READS WERE SATISFIED FROM ACTIVE LOG DATA SET

   The DB2 Log Manager statistics revealed that more than 25% of the
   log reads were satisfied from the active log data set. It is
   preferable that the data be in the output buffer, but this is not
   always possible with an active DB2 environment.  However, if a large
   percent of reads are satisfied from the active log, you should ensure
   that the output buffer is as large as possible. This finding occurred
   during the intervals shown below:

                              TOTAL LOG       LOG READS FROM
   MEASUREMENT INTERVAL         READS      ACTIVE LOG DATA SET   PERCENT
   21:32-21:47, 08SEP1998      56,393             56,391          100.0
```

Suggestion: As mentioned above, it is preferable that the data be in the output buffer when a log read occurs. While this is not always possible with an active DB2 environment, you should attempt to minimize the reads to the active log or to the archive log. Reads to the output buffer do not require an I/O operation, while I/O is required for reads to the active log or archive log.

If a large percent of reads are satisfied from the active log (and thus DB2 is incurring I/O operations), you should consider the following alternatives:

- Ensure that the output buffer is as large as possible. The larger the output buffer, the more likely a requested RBA can be found without a read request to either the active log or the archive log.

  - The OUTPUT BUFFER field on panel DSNTIPL specifies the size of the output buffer used for writing active log data sets. The default size of OUTBUF is 400 KB, and the maximum size is 4000 KB. The buffer must be large enough to prevent buffer shortages in order to decrease the number of forced I/O operations (forced because there are no more buffers) or wait conditions.

    IBM recommends that the maximum allowed value (4000 KB) be specified for this field, rather than accepting the default.

- The WRITE THRESHOLD field indicates the number of contiguous 4KB output buffer pages that are allowed to fill before data is written to the active log data set. The default is 20, and IBM recommends that this default be used. IBM further recommends that you never choose a value that is greater than 20% of the number of buffers in the output buffer.

The above advice is somewhat inconsistent, although both suggestions were taken from the IBM's *DB2 Administration Guides*! Assume that the default values were used; 400 KB is the default for OUTPUT BUFFER and 20% is the default for the WRITE THRESHOLD. With these defaults, DB2 would start writing data from the output buffer to the active log when 80 KB of the buffer were filled (400 * .20). This low threshold would allow DB2 to perform I/O operations with the remaining 80% (or 320 KB) of the output buffer in reserve to receive new log requests.

If the IBM recommendation to use the maximum value for the OUTPUT BUFFER (4000 KB) and retain the 20% value for the WRITE THRESHOLD, DB2 would start writing data from the output buffer to the active log when 800 KB of the buffer were filled (4000 * .20), with 3200 KB of the output buffer in reserve to receive new log requests - these are significantly larger values than the values used with the defaults.

While the 4000 KB buffer would be excellent for performance aspects, be aware that an increase in the OUTBUFF value can increase the amount of DB2 data lost if DB2 ABENDs. Consequently, you should balance the performance benefits of not incurring I/O operations against the risk of lost data if DB2 ABENDs.

- Whenever you allocate new active log data sets, preformat them using the DSNJLOGF utility described in Section 3 of Utility Guide and Reference. This avoids the overhead of preformatting the log, which normally occurs at unpredictable times.

- Review and apply IBM's suggestions in the "Guidelines for Controlling Logging" subsection (in the "DB2 Logging" section referenced below) of the *DB2 Administration Guides*.

- You can design the DB2 log for better performance. In particular, try to:

  - Minimize device contention on the log data sets by placing data sets correctly.

  - Avoid waits that occur because no log buffer is available.

- Define enough active log data sets to prevent DB2 from waiting while a log is archived.

- Make the active log large enough that backouts do not have to use the archive log.

- Place the copy of the bootstrap data set and, if using dual active logging, the copy of the active log data sets, on volumes that are accessible on a path different than that of their primary counterparts. DB2 writes serially or in parallel to the log data sets. If a 4KB log control interval is written for the first time to DASD, the write I/Os to the log data sets are done in parallel. If the same 4KB log control interval is again written to DASD, then the write I/Os to the log data sets must be done serially to prevent any possibility of losing log data in case of I/O errors on both copies simultaneously. This improves system integrity; there is no I/O overlap in dual logging, except when the WRITE THRESHOLD value is reached, triggering full output log buffers to be written. .

- You can alter CPExpert's analysis by modifying the PCTRACT guidance variable in USOURCE(DB2GUIDE).

Reference: DB2 for OS/390 Version 3: Administration Guide
Section 2.4.1.26 (Active Log Data Set Parameters: DSNTIPL)

DB2 for OS/390 Version 4: Administration Guide
Section 5.5.5 (DB2 Logging)

DB2 for OS/390 Version 5: Administration Guide
Section 5.5.5 (DB2 Logging)

DB2 for OS/390 Version 6: Administration Guide
Section 5.5.5 (DB2 Logging)

Rule DB2-513: Log reads satisfied from archive log data set

Finding: The DB2 Log Manager statistics revealed that log reads were satisfied from the archive log data set.

Impact: This finding can have a MEDIUM IMPACT or HIGH IMPACT on the performance of the DB2 subsystem.

Discussion: DB2 logs changes made to data, and other significant events, as they occur. DB2 initially writes the changes to the log buffer, the changes are then written to the active logs, and eventually off loaded to the archive logs.

For optimal performance, when data is backed out, it should still be available in the output buffer or in the active log. If the data has already been off-loaded to the archive log, the active log is probably too small.

The capacity you specify for the active log affects DB2 performance significantly. If you specify a capacity that is too small, DB2 might need to access data in the archive log during rollback and restart. This takes a considerable amount of time. When you are deciding how many active log data sets to use and how large each must be, consider that:

- Performance is negatively affected when DB2 must access data on an archive log data set. This can occur during rollback or restart. Access to archived information can be delayed for a considerable length of time if a unit is unavailable or if a volume mount is required (for example, a tape mount).

- During rollback, DB2 does not share access to an archive log data set with multiple units of work. In other words, one unit of work retrieves all records from an archive log data set before another unit of work is allowed to access that archive log data set. Database locks, virtual storage, and other resources are not freed until the rollback completes. DB2 is unavailable for new work until restart processing is complete. However, if the archive log data set resides on DASD, it can be shared by many units of work.

CPExpert compares the QJSTRARH variable in DB2STATS (the number of log reads satisfied from the archive log data set) with the **QJSTRARH** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-513 when the number of log reads satisfied from the archive log data set exceeds the value specified by the **QJSTRARH** guidance variable.

©Copyright 1999, Computer Management Sciences, Inc.

The default value for the **QJSTRARH** guidance variable is 0, indicating that CPExpert should produce Rule DB2-513 when **any** log reads were satisfied from the archive log data set.

The following example illustrates the output from Rule DB2-513:

```
RULE DB2-513: LOG READS WERE SATISFIED FROM ARCHIVE LOG DATA SET

   The DB2 Log Manager statistics revealed that some of the log reads
   were satisfied from the archive log data set. If this finding is
   produced more than infrequently or if it is produced for intervals
   that are during prime-time, you should consider (1) increasing the
   number of active log data sets or (2) increasing the size of the
   active log data sets.  This finding occurred during the intervals
   shown below:

                              TOTAL LOG   --LOG READS FROM INDICATED LOG--
   MEASUREMENT INTERVAL         READS     OUTPUT       ACTIVE      ARCHIVE
   21:32-21:47, 08SEP1998      56,393          1       56,391          1
```

Suggestion: If Rule DB2-513 is produced more than infrequently or if it is produced for intervals that are during prime-time, you should consider (1) increasing the number of active log data sets or (2) increasing the size of the active log data sets.

You can alter CPExpert's analysis by modifying the **QJSTRARH** guidance variable in USOURCE(DB2GUIDE).

Reference: DB2 for OS/390 Version 3: Administration Guide
Section 2.4.1.26 (Active Log Data Set Parameters: DSNTIPL)

DB2 for OS/390 Version 4: Administration Guide
Section 5.5.5 (DB2 Logging)

DB2 for OS/390 Version 5: Administration Guide
Section 5.5.5 (DB2 Logging)

DB2 for OS/390 Version 6: Administration Guide
Section 5.5.5 (DB2 Logging)

Rule DB2-515: Failed look-ahead tape mounts

Finding: The DB2 Log Manager statistics revealed that look-ahead tape mounts failed.

Impact: This finding can have a MEDIUM IMPACT or HIGH IMPACT on the performance of the DB2 subsystem.

Discussion: DB2 logs changes made to data, and other significant events, as they occur. DB2 initially writes the changes to the log buffer, the changes are then written to the active logs, and eventually off loaded to the archive logs. The archive logs can reside on either tape or DASD.

If failure occurs within a unit of recovery, DB2 backs out any changes to data, returning the data to its state at the start of the unit of recovery. That is, DB2 undoes the work. The SQL ROLLBACK statement, and deadlocks and timeouts cause DB2 to perform the same actions.

Any log records not found in the active logs are read from the archive log data sets, which are dynamically allocated to satisfy the requests. The type of storage used for archive log data sets is a significant factor in the performance.

If the archive log must be read from tape, DB2 optimizes access by means of ready-to-process and look-ahead mount requests. DB2 also permits delaying the deallocation of a tape drive if later RECOVER jobs require the same archive log tape. Those methods are described in more detail below.

- **Ready-to-process**: The current job needs this tape immediately. As soon as the tape is loaded, DB2 allocates and opens it.

- **Look-ahead**: This is the next tape volume required by the current job. Responding to this request enables DB2 to allocate and open the data set before it is needed, thus reducing overall elapsed time for the job.

CPExpert computes the effectiveness of DB2's look-ahead tape mount algorithm by the following algorithm:

$$\textit{Failed Look-ahead mounts} = \frac{\textit{Look-ahead attempts} - \textit{Successful look-aheads}}{\textit{Look-ahead attempts}}$$

©Copyright 1999, Computer Management Sciences, Inc.

CPExpert compares the resulting percent with the **PCTFLAM** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-515 when the number of log reads satisfied from the archive log data set exceeds the value specified by the **PCTFLAM** guidance variable.

The default value for the **PCTFLAM** guidance variable is 0, indicating that CPExpert should produce Rule DB2-515 when **any** log reads were satisfied from the archive log data set.

The following example illustrates the output from Rule DB2-515:

```
RULE DB2-515: FAILED LOOK-AHEAD TAPE MOUNTS

  The number of failed "look-ahead" tape mounts was greater than 0% of
  the total "look-ahead" tape mounts attempted.  Too many failed attempts
  negate potential performance gains from "look-ahead" tape mounts, and
  can be caused by not having enough tape units available.  This finding
  occurred during the intervals shown below:

                            TOTAL "LOOK-AHEAD"      FAILED     PERCENT
  MEASUREMENT INTERVAL      TAPE MOUNT ATTEMPTS    ATTEMPTS     FAILED
  21:44-21:59, 08SEP1998              10               5          50
```

Suggestion: If Rule DB2-515 is produced more than infrequently or if it is produced for intervals that are during prime-time, you should consider the following alternatives:

- If system resources permit, allocate the archive log data sets to DASD. Keeping archive logs on DASD provides the best possible performance.

- Controlling archive log data sets by DFSMShsm is next best alternative. DB2 optimizes recall of the data sets; after being recalled, the data set is read from DASD. Recall for the first DFSMShsm archive log data set starts automatically when the LOGAPPLY phase starts. When recall is complete and the first log record is read, recall for the next archive log data set starts. This process is know as *look-ahead recalling*. Its object is to recall the next data set in parallel with reading the preceding one.

- Consider staging cataloged tape data sets to DASD before allocation by the log read process.

- If the data sets are read from tape, set both the COUNT and the TIME value on the -SET ARCHIVE command to the maximum allowable within the system constraints.

You can alter CPExpert's analysis by modifying the **PCTFLAM** guidance variable in USOURCE(DB2GUIDE).

Reference: DB2 for OS/390 Version 3: Administration Guide
Section 2.4.1.26 (Active Log Data Set Parameters: DSNTIPL)

DB2 for OS/390 Version 4: Administration Guide
Section 5.5.5 (DB2 Logging)

DB2 for OS/390 Version 5: Administration Guide
Section 5.5.5 (DB2 Logging)

DB2 for OS/390 Version 6: Administration Guide
Section 5.5.5 (DB2 Logging)

Rule DB2-521: Log reads were delayed because of unavailable resources

Finding: The DB2 Log Manager statistics revealed that log reads were delayed because of unavailable resources.

Impact: This finding can have a MEDIUM IMPACT or HIGH IMPACT on the performance of the DB2 subsystem.

Discussion: DB2 logs changes made to data, and other significant events, as they occur. DB2 initially writes the changes to the log buffer, the changes are then written to the active logs, and eventually off loaded to the archive logs. The archive logs can reside on either tape or DASD.

If failure occurs within a unit of recovery, DB2 backs out any changes to data, returning the data to its state at the start of the unit of recovery. That is, DB2 undoes the work. The SQL ROLLBACK statement, and deadlocks and timeouts cause DB2 to perform the same actions.

Any log records not found in the active logs are read from the archive log data sets, which are dynamically allocated to satisfy the requests. If the archive log is on tape, the rollback is a sequential process. Any agent attempting to access the archive log must wait until the tape volume is free. In this case, the read accesses might be delayed because of unavailable resources.

CPExpert compares the QJSTWUR (read accesses for archive log delayed because of unavailable resources) with the **QJSTWUR** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-521 when the number of read accesses for archive log delayed because of unavailable resources) exceeds the value specified by the **QJSTWUR** guidance variable.

The default value for the **QJSTWUR** guidance variable is 0, indicating that CPExpert should produce Rule DB2-521 when **any** log reads delayed because of unavailable resources)

The following example illustrates the output from Rule DB2-521:

©Copyright 1999, Computer Management Sciences, Inc.

```
RULE DB2-521: READ ACCESSES DELAYED BECAUSE OF UNAVAILABLE RESOURCES

  Read accesses were delayed because of unavailable resources.  This can
  be caused by not having enough tape units allocated.  This finding
  occurred during the intervals shown below:

                                  READ ACCESSES DELAYED
  MEASUREMENT INTERVAL            UNAVAILABLE RESOURCES
  19:44-19:59, 08SEP1998                     15
```

Suggestion: If Rule DB2-521 is produced more than infrequently or if it is produced for intervals that are during prime-time, you should consider the following alternatives:

- If system resources permit, allocate the archive log data sets to DASD. Keeping archive logs on DASD provides the best possible performance.

- Controlling archive log data sets by DFSMShsm is next best alternative. DB2 optimizes recall of the data sets; after being recalled, the data set is read from DASD. Recall for the first DFSMShsm archive log data set starts automatically when the LOGAPPLY phase starts. When recall is complete and the first log record is read, recall for the next archive log data set starts. This process is know as *look-ahead recalling*. Its object is to recall the next data set in parallel with reading the preceding one.

- Consider staging cataloged tape data sets to DASD before allocation by the log read process.

- If the data sets are read from tape, set both the COUNT and the TIME value on the -SET ARCHIVE command to the maximum allowable within the system constraints.

You can alter CPExpert's analysis by modifying the **QJSTWUR** guidance variable in USOURCE(DB2GUIDE).

Reference: DB2 for OS/390 Version 3: Administration Guide
Section 2.4.1.26 (Active Log Data Set Parameters: DSNTIPL)

DB2 for OS/390 Version 4: Administration Guide
Section 5.5.5 (DB2 Logging)

DB2 for OS/390 Version 5: Administration Guide
Section 5.5.5 (DB2 Logging)

DB2 for OS/390 Version 6: Administration Guide
Section 5.5.5 (DB2 Logging)

Rule DB2-522: Log reads were delayed because of tape volume contention

Finding: The DB2 Log Manager statistics revealed that log reads were delayed because of tape volume contention.

Impact: This finding can have a MEDIUM IMPACT or HIGH IMPACT on the performance of the DB2 subsystem.

Discussion: DB2 logs changes made to data, and other significant events, as they occur. DB2 initially writes the changes to the log buffer, the changes are then written to the active logs, and eventually off loaded to the archive logs. The archive logs can reside on either tape or DASD.

If failure occurs within a unit of recovery, DB2 backs out any changes to data, returning the data to its state at the start of the unit of recovery. That is, DB2 undoes the work. The SQL ROLLBACK statement, and deadlocks and timeouts cause DB2 to perform the same actions.

Any log records not found in the active logs are read from the archive log data sets, which are dynamically allocated to satisfy the requests. If the archive log is on tape, the rollback is a sequential process. Any agent attempting to access the archive log must wait until the tape volume is free. In this case, the read accesses might be delayed because of tape volume contention with another agent that is reading from the archive log..

CPExpert compares the QJSTTVC (read accesses for archive log delayed because of unavailable resources) with the **QJSTTVC** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-522 when the number of read accesses for archive log delayed because of unavailable resources) exceeds the value specified by the **QJSTTVC** guidance variable.

The default value for the **QJSTTVC** guidance variable is 0, indicating that CPExpert should produce Rule DB2-522 when **any** log reads delayed because of unavailable resources)

The following example illustrates the output from Rule DB2-522:

©Copyright 1999, Computer Management Sciences, Inc.

```
RULE DB2-522: READ ACCESSES WERE DELAYED, TAPE VOLUME CONTENTION

  Read accesses were delayed because of tape volume contention.  This can
  occur in a situation where only one reader per tape is possible. This
  finding occurred during the intervals shown below:

                                   READ ACCESSES DELAYED
  MEASUREMENT INTERVAL             TAPE VOLUME CONTENTION
  21:44-21:59, 08SEP1998                      2
```

Suggestion: If Rule DB2-522 is produced more than infrequently or if it is produced for intervals that are during prime-time, you should consider the following alternatives:

- If system resources permit, allocate the archive log data sets to DASD. Keeping archive logs on DASD provides the best possible performance.

- Controlling archive log data sets by DFSMShsm is next best alternative. DB2 optimizes recall of the data sets; after being recalled, the data set is read from DASD. Recall for the first DFSMShsm archive log data set starts automatically when the LOGAPPLY phase starts. When recall is complete and the first log record is read, recall for the next archive log data set starts. This process is know as *look-ahead recalling*. Its object is to recall the next data set in parallel with reading the preceding one.

- Consider staging cataloged tape data sets to DASD before allocation by the log read process.

You can alter CPExpert's analysis by modifying the **QJSTTVC** guidance variable in USOURCE(DB2GUIDE).

Reference: DB2 for OS/390 Version 3: Administration Guide
Section 2.4.1.26 (Active Log Data Set Parameters: DSNTIPL)

DB2 for OS/390 Version 4: Administration Guide
Section 5.5.5 (DB2 Logging)

DB2 for OS/390 Version 5: Administration Guide
Section 5.5.5 (DB2 Logging)

DB2 for OS/390 Version 6: Administration Guide
Section 5.5.5 (DB2 Logging)

Rule DB2-601: Coupling facility read requests could not complete

Finding: DB2 was unable to register a page or a list of pages in the coupling facility because of a lack of directory entry storage.

Impact: This finding can have a MEDIUM IMPACT or HIGH IMPACT on the performance of the DB2 subsystem.

Discussion: When DB2 needs to reference a page, it looks for the page in the following order:

- **Step 1: Search the virtual buffer pool**. The page might exist in the virtual buffer pool both with a non-data sharing environment and with a data sharing. However, with a data sharing environment, the page might be marked invalid in the virtual buffer pool. In this case, DB2 skips to Step 4.

- **Step 2: Search the hiperpool buffer pool**. If a hiperpool exists for the virtual buffer pool, DB2 next looks for the page In the hiperpool. The page might exist in the hiperpool both with a non-data sharing environment and with a data sharing (unless GBPCACHE ALL has been specified for the page set, in which case the hiperpool is not used). However, with a data sharing environment, the page might be marked invalid in the hiperpool. In this case, DB2 skips to Step 4.

- **Step 3: Search the group buffer pool**. With a data sharing environment, DB2 next checks the group buffer pool for the page. This is done if the page set or the partition is group buffer pool dependent or if the page set is defined as GBPCACHE ALL. This step is skipped if one of the following conditions is true:

  - The group buffer pool is defined as GBPCACHE(NO).

  - The page set is defined as GBPCACHE NONE.

  - The page set is defined as GBPCACHE SYSTEM and the page being read is not a space map page.

- **Step 4: Read the page from DASD**. DB2 reads the page from DASD if the page is not found after applying the above search.

DB2 reads a page from DASD using either random read, sequential read, or sequential prefetch read. In either case, with data sharing and a shared

page set, DB2 must register the page with the coupling facility (group buffer pool) if there is inter-DB2 R/W interest[1] in the page set. This is done so DB2 can maintain integrity of the data. For example, if another DB2 should change the same page, the page must be invalidated in the virtual buffer pool or hiperpool of *this* DB2.

The page is registered with the group buffer pool before it is read from DASD. If a directory entry exists for the page, the directory entry is updated to reflect that *this* DB2 has an interest in the page. If a directory entry does not exist for the page, one is created and updated to reflect that *this* DB2 has an interest in the page.

As just described, a directory entry must be created if one does not exist for the page. Creating a directory entry means that storage must be available in the group buffer pool to hold the directory entry. Under certain situations, there may be no directory entry storage available in the group buffer pool.

Group buffer pool storage is divided into directory entry storage and data entry storage. This means that, of the total storage allocated to the group buffer pool, some of the storage is reserved for directory entries and some is reserved for data (page) entries. This division of storage is done when the group buffer pool is created and can be changed via an ALTER GROUPBUFFERPOOL command. The division is based on a specified **ratio** of directory entries to data entries. The default ratio is 5:1, meaning that there are 5 directory entries for each data entry in the group buffer pool.

There are more directory entries than data entries because (unless GBPCACHE ALL is specified), only *changed* pages from a virtual buffer pool are placed in the group buffer pool. Unless there is extensive update activity, changed pages account for a relatively small percent of the total number of pages in a virtual buffer pool. However, a directory entry must be created to account for each page from a shared page set that is in any virtual buffer pool or hiperpool of the sharing DB2 members. Consequently, there normally would be far more directory entries than there would be data entries. IBM has decided that, as a default, there should be five times as many directory entries as there are data entries.

As mentioned above, when a DB2 wants to read a page from DASD, it must register the page with the group buffer pool if there is inter-DB2 R/W interest in the page set. If many pages are registered (or if the ratio of

[1] Inter-DB2 R/W interest is a property of data in a table space, index, or partition that has been opened by more than one member of a data sharing group and that has been opened for writing by at least one of those members. When there is inter-DB2 R/W interest in a particular table space, index, or partition, it is dependent on the group buffer pool associated with the object. The object is then said to be *GBP-dependent* (group buffer pool dependent).

directory entries to data entries is too low), there might be no storage in the group buffer pool that is available for creating a directory entry. In this case, the read fails and directory entries must be **reclaimed** in the group buffer pool.

This situation can cause serious performance problems for the following reasons:

- When a DASD read fails, the DB2 application must wait for the page if the read were a synchronous read (a random page read or a sequential page read that had not been read as part of a prefetch operation).

- When a DASD read fails for a prefetch operation, the prefetch operation is aborted. This means that pages that otherwise would be prefetched via an asynchronous I/O operation must be read via a synchronous I/O operation when the pages are required.

- When directory entries are reclaimed, the associated pages must be invalidated in each DB2 registered for the pages. This page invalidation process creates overhead and uses system resources.

- When a page has been invalidated in each DB2 registered for the page, that DB2 must re-read the page from DASD if it again wishes to reference the page. Re-reading the page requires re-registration of the page (including creating a directory entry). Reclaiming pages because of a shortage of directory entry storage can cause “thrashing” of pages between buffer pools and DASD, and can generate significant overhead.

CPExpert compares the QBGLRF variable in DB2GBPST (the number of failed reads because of unavailable storage in the group buffer pool) with the **QBGLRF** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-601 when the number of failed reads because of unavailable storage in the group buffer pool exceeds the value specified by the **QBGLRF** guidance variable.

The default value for the **QBGLRF** guidance variable is 0, indicating that CPExpert should produce Rule DB2-601 when any read failed because of unavailable storage in the group buffer pool.

The following example illustrates the output from Rule DB2-601:

```
RULE DB2-601: COUPLING FACILITY READ REQUESTS COULD NOT COMPLETE

   Group Buffer Pool 6: Coupling facility read requests could not be
   completed because of a lack of coupling facility storage resources.
   This situation occurred for Group Buffer Pool 6 during the intervals
   shown below:

                                GROUP BUFFER POOL      TIMES CF READ
   MEASUREMENT INTERVAL          ALLOCATED SIZE     REQUESTS NOT COMPLETE
   11:01-11:31, 14OCT1999              38M                   130
```

Suggestion: If Rule DB2-601 is produced more than occasionally, you should consider the following alternatives:

- **Increase the directory entry/data entry ratio**. The easiest solution to this problem is to simply increase the directory entry/data entry ratio. This can be done by ALTER GROUPBUFFERPOOL.

  Recall that the directory entry size is small relative to the size of the data entry (directory entries are normally about 200 bytes while data entries are the size of the local buffer pool page, which is a minimum of 4096 bytes). Consequently, increasing the directory entry/data entry ratio might yield a relatively large increase in the number of directory entries, while not dramatically reducing the number of data entries.

- **Increase the size of the group buffer pool**. If the above solution is not feasible because of a relative shortage of data entries, you should consider increasing the size of the group buffer pool. Increasing the size of the group buffer pool will, of course yield more directory entries (since part of the increase would be used for directory entries and part would be used for data entries).

  If you should increase the size of the group buffer pool, you might consider increasing the directory entry/data entry ratio at the same time. This would cause the increased group buffer pool storage to be used for directory entries.

- **Reduce the data sharing level**. In some instances, conflicts between applications might be so great that it might be impossible to obtain satisfactory DB2 data sharing performance. In this case, it might be necessary to decrease the data sharing level associated with specific group buffer pools. To accomplish this, you could reassign page sets or partitions to different virtual buffer pools (with corresponding reassignment to different group buffer pools), create additional virtual buffer pools (with corresponding additional group buffer pools), or move some application between data sharing members and even to another processing time.

- You can alter CPExpert's analysis by modifying the **QBGLRF** guidance variable in USOURCE(DB2GUIDE).

Reference: DB2 for OS/390 Version 4: Data Sharing Planning and Administration Section 6.4.3.1 (Where DB2 looks for a page)

DB2 for OS/390 Version 5: Data Sharing Planning and Administration Section 7.5.4.1 (Where DB2 looks for a page)

DB2 for OS/390 Version 6: Data Sharing Planning and Administration Section 7.6.4.1 (Where DB2 looks for a page)

DB2 for MVS/ESA Version 4 Data Sharing Performance Topics Redbook (SG24-4611)

©Copyright 1999, Computer Management Sciences, Inc.

Rule DB2-602: Coupling facility write requests could not complete

Finding: Coupling facility write requests could not complete because of a shortage of available storage in the coupling facility structure.

Impact: This finding can have a MEDIUM IMPACT or HIGH IMPACT on the performance of the DB2 subsystem.

Discussion: Rule DB2-220 through Rule DB2-222 describe how DB2 manages pages in a virtual buffer pool. Included in these descriptions is the concept of *deferred writes* of pages. With deferred writes, DB2 retains changed pages in the virtual buffer pool expecting that there is some probability that the updated pages will be referenced again. These deferred write pages are queued by data set until they are written when:

- A DB2 *checkpoint* is taken.
- The percentage of unavailable pages in a virtual buffer pool exceeds a preset limit called the Deferred Write Threshold (DWQT).
- The percentage of updated pages in a virtual buffer pool for a single data set exceeds a preset limit called the Vertical Deferred Write Threshold (VDWQT).

With data sharing, DB2 still performs deferred writes for DB2 table spaces, indexes or partitions. However, when an update is to a page set that has inter-DB2 R/W interest[1], DB2 writes the updated pages from the virtual buffer pool to the group buffer pool when:

- The transaction commits. When committing an updating transaction, pages that were updated but not yet written to the group buffer pool are synchronously written to the group buffer pool.
- The Deferred Write Threshold (DWQT) or the Vertical Deferred Write Threshold (VDWQT)[2] is reached. Deferred write pages in the virtual buffer pool that are synchronously written to the group buffer pool.

---

[1] Inter-DB2 R/W interest is a property of data in a table space, index, or partition that has been opened by more than one member of a data sharing group and that has been opened for writing by at least one of those members. When there is inter-DB2 R/W interest in a particular table space, index, or partition, it is dependent on the group buffer pool associated with the object. The object is then said to be *GBP-dependent* (group buffer pool dependent).

[2] DB2 for MVS/ESA Version 4 Data Sharing Performance Topics Redbook (SG24-4611) states that DB2 writes updated pages to the group buffer pool when 50% of VDWQT is reached, and it makes no mention of a percentage of DWQT. Other documents shown in the references do not discuss a percentage of these thresholds being used to trigger writing updated pages to the group buffer pool. I've not been able to verify this apparent conflict between the two sets of IBM documents.

©Copyright 1999, Computer Management Sciences, Inc.

- The buffer pool is short of reassignable buffers because writes to the group buffer pool cannot keep up with update activity in the buffer pool. The shortage of buffers also can occur when the deferred write thresholds are too high, or if the application is not committing frequently.

- An updated page has stayed in the buffer pool for a long period of time since it was last referenced or updated. This can happen in situations such as with a long running transaction that doesn't issue frequent commits. In this case, a system checkpoint will cause deferred write pages in the virtual buffer pool to be synchronously written to the group buffer pool.

- A page is required for update by another system because there is no conflict on transaction locking (such as page sets that are using row locking, index pages, space map pages, etc.).

When a page of data is written to the group buffer pool, all copies of that page cached in other members' buffer pool are invalidated. This means that the next time one of those members needs that page, the page must be refreshed from the group buffer pool (or DASD).

A data entry must be available in the group buffer pool when a page of data is written to the group buffer pool. Under certain conditions, the group buffer pool might not have a data entry available, and the write is rejected because of a lack of storage.

Message DSNB319A (*THERE IS A SHORTAGE OF SPACE IN GROUP BUFFER POOL gbpname*) is issued when the group buffer pool is 75% full. Message DSNB325A (*THERE IS A CRITICAL SHORTAGE OF SPACE IN GROUP BUFFER POOL gbpname)* is issued when the group buffer pool is 90% full. Message DSNB228I *(GROUP BUFFER POOL gbpname CANNOT BE ACCESSED*) is issued when the group buffer pool is full.

When a write is rejected, DB2 initiates castout processing if it isn't already in progress. The DB2 member waits for 3 seconds after triggering the castout, and it retries the write operation for up to four times. For simplexed group buffer pools, or for the primary of a duplexed group buffer pool,[3] pages that cannot be written to the group buffer pool are added to the Logical Page List and message DSNB250E is issued. This action would require a recovery of the page set before the pages could be used.

.

[3] If it is the secondary group buffer pool that is too full, DB2 does not add pages to the logical page list, but takes the structure out of duplexing mode

CPExpert compares the QBGLWF variable in DB2GBPST (the number of failed writes because of unavailable storage in the group buffer pool) with the **QBGLWF** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-602 when the number of failed writes because of unavailable storage in the group buffer pool exceeds the value specified by the **QBGLWF** guidance variable.

The default value for the **QBGLWF** guidance variable is 0, indicating that CPExpert should produce Rule DB2-602 when any writes failed because of unavailable storage in the group buffer pool.

The following example illustrates the output from Rule DB2-602:

```
RULE DB2-602: COUPLING FACILITY WRITE REQUESTS COULD NOT COMPLETE

   Group Buffer Pool 8: Coupling facility write requests could not be
   completed because of a lack of coupling facility storage resources.
   DB2 was unable to write a page in the coupling facility because of a
   lack of data entry storage.  Failed write requests can cause serious
   DB2 performance problems.  This situation occurred for Group Buffer
   Pool 8 during the intervals shown below:

                                GROUP BUFFER POOL      TIMES CF WRITE
   MEASUREMENT INTERVAL          ALLOCATED SIZE     REQUESTS NOT COMPLETE
   11:01-11:31, 14OCT1999              21M                    14
```

Suggestion: If Rule DB2-602 is produced, you should consider the following alternatives:

- **Reduce the local buffer pool thresholds**. Failed writes typically result from castout processing not being able to keep up with the writes. There are several causes, but the most common is that a transaction commits and "floods" the group buffer pool with write requests. This "flood" of write requests can be reduced by reducing the local buffer pool thresholds (VDWQT and DWQT). This would spread the write requests more evenly over the life of the transaction.

  - Lower the Vertical Deferred Write Threshold (VDWQT) for the virtual buffer pool. A low (or even zero[4]) VDWQT value will cause DB2 to write updated pages more frequently to the group buffer pool. As a result, there would not be a large number of pages required to be written at checkpoint or commit.

  - Lower the Deferred Write Threshold (DWQT) for the virtual buffer pool. A low DWQT value will cause DB2 to write updated pages

[4] With VDWQT specification of zero, DB2 will write 32 pages at a time to the group buffer pool.

more frequently to the group buffer pool. As a result, there would not be a large number of pages required to be written at checkpoint or commit.

- **Increase frequency of castout**. By increasing the frequency of castout processing, pages in the group buffer pool would be castout to DASD more often. Consequently, more pages would normally be available. This is accomplished by reducing the group buffer pool checkpoint interval (the default is eight minutes), reducing the group buffer pool castout threshold, or reducing the group buffer pool class castout threshold.

- **Increase the size of the group buffer pool**. As mentioned above, the basic cause of the failed write is that a data entry was not available. You should consider increasing the size of the group buffer pool if the above actions do not yield satisfactory results.

- **Issue COMMIT more frequently**. If the application can issue COMMIT more frequently, few pages would normally be written to the group buffer pool at each COMMIT. Consequently, there would be a greater opportunity for the castout processing to castout pages between COMMIT operations.

- **Verify GBPCACHE ALL is required**. Some applications might have specified GBPCACHE ALL, and DB2 could be “flooding” the group buffer pool during periods of heavy prefetch activity. In this case, investigate these applications to determine whether this specification is necessary.

- **Review I/O configuration**. You should check your I/O configuration (and the coupling facility performance) to ensure that castout processing is not being delayed by I/O waits.

- **Reduce the directory entry/data entry ratio**. It is possible that the directory entry/data entry ratio is too high, resulting in too much storage reserved for directory entries rather than being available for data entries.

  However, the directory entry is small relative to the size of the data entry (directory entries are normally about 200 bytes while data entries are the size of the local buffer pool page, which is a minimum of 4096 bytes). Consequently, adjusting the directory entry/data entry ratio usually would have little positive effect.

- **Reduce the data sharing level**. In some instances, conflicts between applications might be so great that it might be impossible to obtain satisfactory DB2 data sharing performance. In this case, it might be necessary to decrease the data sharing level associated with specific

group buffer pools. To accomplish this, you could reassign page sets or partitions to different virtual buffer pools (with corresponding reassignment to different group buffer pools), create additional virtual buffer pools (with corresponding additional group buffer pools), or move some application between data sharing members and even to another processing time.

Reference: DB2 for OS/390 Version 4: Data Sharing Planning and Administration
Section 5.7.7.7 (Problem: Storage Shortage in the group buffer pool)
Section 6.4.4.1 (Writing to the Group Buffer Pool)

DB2 for OS/390 Version 5: Data Sharing Planning and Administration
Section 6.7.7.7 (Problem: Storage Shortage in the group buffer pool)
Section 7.5.5.1 (Writing to the Group Buffer Pool)

DB2 for OS/390 Version 6: Data Sharing Planning and Administration
Section 6.7.7.7 (Problem: storage shortage in the group buffer pool)
Section 7.6.5.2 (Writing to the group buffer pool)

DB2 for MVS/ESA Version 4 Data Sharing Performance Topics Redbook (SG24-4611)

©Copyright 1999, Computer Management Sciences, Inc.

Rule DB2-603: There was low read hit percent for cross-invalidated pages

Finding: The group buffer pool had a low read hit percentage for cross-invalidated pages, meaning that the cross-invalidated pages were not present in the group buffer pool when needed.

Impact: This finding can have a SMALL IMPACT, MEDIUM IMPACT, or HIGH IMPACT on the performance of the DB2 subsystem. The level of impact depends on how often data was not returned from read requests when the buffer had been marked invalid.

Discussion: Pages of GBP-dependent[1] page sets are registered in the group buffer pool when they are read from DASD by any DB2 member of the data sharing group. When a changed page is written to the group buffer pool, all DB2 subsystems that have this page cached in their buffer pools are notified that the page has been invalidated. This notification process is called cross-invalidation.

When a DB2 member needs a page of data and finds the page in its local buffer pool (either a virtual buffer pool or a hiperpool), it tests to see if the buffer contents are still valid. If the buffer contents are not valid, the page must be refreshed, either from the group buffer pool or DASD[2].

Under many conditions, the page will reside in the group buffer pool. This is because the page was written to the group buffer pool by another DB2 member when it was changed and cross-invalidated. But changed pages do not stay in the group buffer pool indefinitely. Eventually, the page is *castout* (written to DASD). Even after the page has been castout, it remains in the group buffer pool, and can be read by any DB2 member. However, the status of a castout page is altered from “changed” to “unchanged”, since it has been written to DASD[3].

Data entries in the group buffer pool that hold unchanged pages can be *reclaimed* if required to hold a new page to be placed in the group buffer

---

[1] Inter-DB2 R/W interest is a property of data in a table space, index, or partition that has been opened by more than one member of a data sharing group and that has been opened for writing by at least one of those members. When there is inter-DB2 R/W interest in a particular table space, index, or partition, it is dependent on the group buffer pool associated with the object. The object is then said to be *GBP-dependent* (group buffer pool dependent).

[2] See Rule DB2-601 for further discussion of the actions DB2 takes when it needs to reference a page of data.

[3] ”Unchanged” pages that are eligible for reclaim would always be those changed pages that had been castout, unless GBPCACHE ALL had been specified for a page set. If GBPCACHE ALL had been specified , “unchanged” pages would include all pages that had been read from the page set, regardless of whether they had been changed and castout, or had simply been read as a part of GBPCACHE ALL.

©Copyright 1999, Computer Management Sciences, Inc.

pool. Unchanged pages are reclaimed on a least recently used (LRU) basis. Once a data entry has been reclaimed, the previous page is no longer available in the group buffer pool. If any DB2 member needs to read the page, the page must be read from DASD.

As a separate issue, directory entries can be reclaimed[4]. When directory entries are reclaimed, the associated pages are invalidated in each DB2 registered for the pages.

If the group buffer pool is too small, previously changed pages that had been castout are quickly reclaimed, and an attempted reread of a reclaimed page would fail. Alternatively, if the directory entry to data entry is too low, directory entries describing castout pages can be reclaimed, and an attempted reread of the page associated with the reclaimed directory entries would fail.

CPExpert computes a read hit percentage for **cross-invalidated** pages[5], based on the following algorithm:

$$\textit{read hit percentage} = \frac{QBGLXD}{QBGLXD + QBGLXR}$$

where:

QBGLXD = Reads due to cross-invalidation, data returned**.** This means that the page was in the group buffer pool and DB2 did not have refresh the page from DASD. This is the ideal situation.

QBGLXR = Reads due to cross-invalidation, data not returned, directory entry existed or created. This means that the page was not in the group buffer pool, but another DB2 still had read/write interest in the page set or partition. The page had to be refreshed from DASD. This is an undesirable read-miss condition.

To avoid spurious findings, CPExpert ignores any measurement interval in which there were less than 1000 reads because of cross-invalidation.

A low percentage of read hits for cross-invalidated pages for any particular DB2 member normally indicates that the average residency time for a cached page in the group buffer pool is too short, which might indicate that the group buffer pool is too small or the directory entry to data entry is too low. The reasoning here is that:

[4] See Rule DB2-601 for a discussion of directory entries, directory entry to data entry ratio, and directory entry reclaim.

[5] Note that the *DB2 Data Sharing and Administration Guides* shows a read hit percentage calculation for all pages read. However, the *Guides* immediately caution that *only* read misses for cross-invalidated pages would indicate a serious performance problem. CPExpert simply computes the read hit percentage based on cross-invalidated pages and uses that percentage as a direct indicator of a performance problem.

- The DB2 member having a low read hit percentage did have the page in its local buffer pool (either virtual buffer pool or hiperpool).

- The page in the local buffer pool had been cross-invalidated by some other member (which means that the other member had placed the changed page in the group buffer pool).

- The page was no longer in the group buffer pool when this member wanted to reference it, which means that the page had been castout, marked “unchanged” in the group buffer pool, and had been reclaimed because data entries or directory entries were needed by the group buffer pool.

- Thus, the residency time for the changed page in the group buffer pool was too short, which probably means that pages were being castout and reclaimed too often because the group buffer pool was too small or the directory entry to data entry was too low.

CPExpert compares the computed read hit percentage for cross-invalidated pages with the **PCTHITGB** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-603 when the computed read hit percentage for cross-invalidated pages is less than the value specified by the **PCTHITGB** guidance variable.

The default value for the **PCTHITGB** guidance variable is 95, indicating that CPExpert should produce Rule DB2-603 when less than ninety-five percent of the reads for cross-invalidated buffers did not find the page in the group buffer pool.

The following example illustrates the output from Rule DB2-603:

```
RULE DB2-603: LOW READ HIT PERCENTAGE FOR CROSS-INVALIDATED PAGES

 Group Buffer Pool 5:  The read hit percentage for cross-invalidated
 pages was less than 95% for this group buffer pool.  This means that
 the pages that had been cross-invalidated by another DB2 member were
 not present in the group buffer pool when needed.  The pages had been
 castout and their data entries had been reclaimed.  This situation
 occurred for Group Buffer Pool 5 during the intervals shown below:

                                                CROSS-INVALIDATED BUFFERS
                                GROUP BUFFER POOL            READ    READ HIT
   MEASUREMENT INTERVAL          ALLOCATED SIZE     READS   MISSES    PERCENT
   11:00-11:30, 14OCT1999             38M           3,026      331      89.1
```

Suggestion: If Rule DB2-603 is regularly produced, you should consider the following alternatives:

- **Increase Class Castout Threshold or Group Buffer Pool Castout Threshold**. Pages normally are castout when a group buffer pool checkpoint occurs, or when the Class Castout Threshold or Group Buffer Pool Castout Threshold is reached. If only a few page sets or partitions are assigned to the group buffer pool, castout might occur prematurely because of the castout thresholds. If a low total read hit percentage occurs, you should consider increasing the thresholds.

  Since all changed pages are castout at group buffer pool checkpoint, you might encounter "castout engine not available" if the thresholds are excessively high. This situation is described in Rule DB2-605.

- **Verify GBPCACHE ALL is required**. Some application may have specified GBPCACHE ALL and DB2 was writing pages to the group buffer pool regardless of whether they are changed. In this case, you should review the total read hit percentage (changed and unchanged pages) to see if GBPCACHE ALL is productive.

  Once changed pages have been castout, their status is altered to "unchanged" and they are susceptible to the same LRU stealing as would unchanged pages that were placed in the group buffer pool by a member that specified GBPCACHE ALL. It is possible that high read activity on a GBPCACHE ALL page set could flood the group buffer pool with unchanged pages, effectively causing castout changed pages to be stolen earlier than otherwise. This particularly would be the case with a relatively small group buffer pool and a relatively large number of pages read from a page set with GBPCACHE ALL specified.

  If you verify that GBPCACHE ALL is the correct specification, you might consider moving the page set to a group buffer pool different from the one that is experiencing a low read hit percent for cross-invalidated pages.

- **Consider GBPCACHE NONE**. One or more of the page sets assigned to the group buffer pool might be a candidate for GBPCACHE NONE (if you are operating with DB2 Version 6). This alternative would be considered, of course, only if total read hit percentage was very low (for example, the percentage was less than 5%).

- **Increase the size of the group buffer pool**. As mentioned above, the basic cause of the low read hit percentage for cross-invalidate pages is that the "residency time" of the pages is too short before the pages are castout. You should consider increasing the size of the group buffer pool if the above actions do not yield satisfactory results.

©Copyright 1999, Computer Management Sciences, Inc.

- **Reduce the data sharing level**. In some instances, conflicts between applications might be so great that it might be impossible to obtain satisfactory DB2 data sharing performance. In this case, it might be necessary to decrease the data sharing level associated with specific group buffer pools. To accomplish this, you could reassign page sets or partitions to different virtual buffer pools (with corresponding reassignment to different group buffer pools), create additional virtual buffer pools (with corresponding additional group buffer pools), or move some application between data sharing members and even to another processing time.

- You can alter CPExpert's analysis by modifying the **PCTHITGB** guidance variable in USOURCE(DB2GUIDE).

Reference:

DB2 for OS/390 Version 4: Data Sharing Planning and Administration
Section 6.4.7 (Making Sure Group Buffer Pools are the Right Size and Ratio)
Section 7.5.5.1 (Writing to the Group Buffer Pool)

DB2 for OS/390 Version 5: Data Sharing Planning and Administration
Section 7.5.8 (Making Sure Group Buffer Pools are the Right Size and Ratio)
Section 7.6.5.2 (Writing to the group buffer pool)

DB2 for OS/390 Version 6: Data Sharing Planning and Administration
Section 6.4.4.1 (Writing to the Group Buffer Pool)
Section 7.6.8 (Determining the correct size and ratio)

DB2 for MVS/ESA Version 4 Data Sharing Performance Topics Redbook (SG24-4611)

©Copyright 1999, Computer Management Sciences, Inc.

Rule DB2-605: Castout engine was not available

Finding: The DB2 statistics revealed that a castout engine was not available during the statistics interval(s) shown.

Impact: This finding can have a MEDIUM IMPACT or HIGH IMPACT on the performance of the DB2 subsystem.

Discussion: With data sharing, a database page can reside in the following locations:

- In a virtual buffer pool
- In a hiperpool
- In a group buffer pool (GBP)
- On DASD

Database pages continue to be cached in each sharing member's virtual buffer pool before they can be referenced or updated. However, if there is inter-DB2 R/W interest[1] in the data, the group buffer pool is also used for caching changed pages[2].

Pages of GBP-dependent page sets are registered in the group buffer pool. When a changed page is written to the group buffer pool, all DB2 subsystems that have this page cached in their buffer pools are notified that the page has been invalidated (this notification does not cause a processing interruption on those systems). This is called cross-invalidation. When a member needs a page of data and finds it in its buffer pool, it tests to see if the buffer contents are still valid. If not, then the page must be refreshed, either from the group buffer pool or DASD.

DB2 uses a castout process to write changed data to DASD from a group buffer pool. The castout process normally is controlled by the value specified for two group buffer pool thresholds:

- **Class castout threshold**. There is a fixed number of castout class queues in each group buffer pool. This number is an internal value set by DB2. DB2 internally maps updated pages that belong to the same

[1] Inter-DB2 R/W interest is a property of data in a table space, index, or partition that has been opened by more than one member of a data sharing group and that has been opened for writing by at least one of those members. When there is inter-DB2 R/W interest in a particular table space, index, or partition, it is dependent on the group buffer pool associated with the object. The object is then said to be *GBP-dependent* (group buffer pool dependent).

[2] It is also possible to cache unchanged pages in the group buffer pool as a mutually exclusive alternative to hiperpools. This is done by specifying GBPCACHE(ALL).

©Copyright 1999, Computer Management Sciences, Inc.

page sets or partitions to the same castout class queues.. This internal mapping scheme is the same across all sharing subsystems.

When DB2 writes changed pages to the group buffer pool, it determines how many changed pages are on a particular class castout queue. If the number of changed pages on a specified castout class queue exceeds the threshold, DB2 casts out a number of pages from that queue.

The default for the class castout is 10, which means that castout is initiated for a particular page set or partition when 10 percent of the group buffer pool contains changed pages for the class.

- **Group buffer pool castout threshold**. The group buffer pool castout threshold is a percentage of the total number of pages in the group buffer pool. This threshold determines the total number of changed pages that can exist in the group buffer pool before castout occurs.

  When DB2 writes changed pages to the group buffer pool, it determines how many changed pages are in the group buffer pool. DB2 casts out enough class castout queues to bring the number of changed pages below the threshold.

  The default value for the group buffer pool castout threshold is 50, which means that castout is initiated when the group buffer pool is 50 percent full of changed pages.

The castout process also is triggered by (1) a group buffer pool checkpoint, (2) when there is no more inter-DB2 R/W interest in the page set, and (3) when the group buffer pool is being rebuilt, but the alternate group buffer pool is not large enough to contain the pages from the group buffer that is being rebuilt.

Pages that are cast out as a result of meeting a threshold remain cached in the group buffer pool, and the buffers are available for stealing. Pages that are cast out because there is no more shared interest in the page set are purged from the group buffer pool.

When data is cast out from a group buffer pool to DASD, that data must first pass through a DB2's address space because there is no direct connection from a coupling facility to DASD. This data passes through a private buffer, not DB2's virtual buffer pools.

The DB2 that is assigned ownership of castout is the DB2 subsystem that had the first update intent (except during restart) on the page set or partition. After the castout ownership is assigned, subsequent updating DB2 subsystems become backup owners. One of the backup owners

becomes the castout owner when the original castout owner no longer has read-write interest in the page set.

When a castout operation is scheduled, a *castout engine* is used for the castout process. DB2 has up to 300 castout engines, depending on the castout workload and the amount of DBM1 storage available. When castout is scheduled, there might not be an available engine, normally because the castout rate occurs too quickly for the DASD system to handle the I/O. In this situation, castout cannot commence for the castout process scheduled, and castout waits for an engine to become available.

The castout class owner retries to schedule the castout three times. When the GBP is used by more than one page set (castout class), other data sharing members also castout from that GBP. If the changed pages in the GBP cannot be cast out, the number of free and clean pages in the GBP is depleted over time..

To allow force-at-commit policy to write changed pages to the coupling facility, the coupling facility reclaims the changed pages. For the changed pages to be reclaimed, they must be cross invalidated, then put on the logical page list (LPL). Once a page is put on the LPL, the page set must be recovered to remove the page from the LPL.

CPExpert compares the QBGLCN variable in DB2GBPST (the number of times a castout engine was not available) with the **QBGLCN** guidance variable in USOURCE(DB2GUIDE).

CPExpert produces Rule DB2-605 when the number of times a castout engine was not available exceeds the value specified by the **QBGLCN** guidance variable.

The default value for the **QBGLCN** guidance variable is 0, indicating that CPExpert should produce Rule DB2-605 whenever a castout engine was not available.

The following example illustrates the output from Rule DB2-605:

```
RULE DB2-605: CASTOUT ENGINE WAS NOT AVAILABLE

   Group Buffer Pool 4: The DB2 statistics revealed that a castout engine
   was not available during the intervals shown below.  You should consider
   increasing the size of the group buffer pool if these numbers are large
   and are not reflecting just a momentary surge in activity.  This
   situation occurred for Group Buffer Pool 4 during the intervals shown
   below:

                                GROUP BUFFER POOL  TIMES CASTOUT ENGINE
   MEASUREMENT INTERVAL          ALLOCATED SIZE       WAS UNAVAILABLE
   10:34-11:04, 14OCT1999              38M                   206
   11:04-11:34, 14OCT1999              38M                   206
```

Suggestion: If Rule DB2-605 is produced more than occasionally, you should consider the following alternatives:

- **Review I/O configuration**. Another common reason that a castout engine is not available is that the castout rate occurs too quickly for the coupling facility and the DASD system to handle the I/O, or that the DB2 "owning" the group buffer pool (and responsible for castout processing) is overloaded with higher importance work. You should verify that the coupling facility and the DASD I/O system are not experiencing unnecessary delay, and that sufficient capacity exists for the DB2 that owns the group buffer pool.

- **Reduce the group buffer pool castout thresholds**. The Class Castout Threshold or the Group Buffer Pool Threshold might be too high, with the result that a burst of write activity to the group buffer pool could cause a corresponding burst of castout activity. Decreasing these thresholds could reduce the bursts of castout.

- **Increase the size of the group buffer pool**. Increasing the size of the group buffer pool might help this situation, but only if the bursts of activity described above can be managed.

- **Change the castout owner for selected page sets or partitions**. The DB2 that is assigned ownership of castout is the DB2 subsystem that had the first update intent (except during restart) on the page set or partition. If a particular DB2 becomes overloaded and cannot provide castout engines, it might be feasible to change the order in which DB2 subsystems register update intent for selected page sets or partitions. This could remove some castout overhead from the DB2 that exhausts castout engines, and transfer the castout overhead to a different DB2 subsystem.

- **Reduce the data sharing level**. In some instances, conflicts between applications might be so great that it might be impossible to obtain satisfactory DB2 data sharing performance. In this case, it might be necessary to decrease the data sharing level associated with specific group buffer pools. To accomplish this, you could reassign page sets or partitions to different virtual buffer pools (with corresponding reassignment to different group buffer pools), create additional virtual buffer pools (with corresponding additional group buffer pools), or move some application between data sharing members and even to another processing time.

- You can alter CPExpert's analysis by modifying the **QBGLCN** guidance variable in USOURCE(DB2GUIDE).

Reference: DB2 for OS/390 Version 4: Data Sharing Planning and Administration
Section 6.4.7.1 (Group Buffer Pool Size Too Small)
Section 6.4.7.3 (Too Few Data Entries)
Section 6.4.7.4 (What to Watch for in a DB2 PM Statistics Report)

DB2 for OS/390 Version 5: Data Sharing Planning and Administration
Section 7.5.8.1 (Group Buffer Pool Size Too Small)
Section 7.5.8.3 (Too Few Data Entries)
Section 7.5.8.4 (What to Watch for in a DB2 PM Statistics Report)

DB2 for OS/390 Version 6: Data Sharing Planning and Administration
Section 7.6.8.1 (Group Buffer Pool Size Too Small)
Section 7.6.8.3 (Too Few Data Entries)
Section 7.6.8.4 (What to Watch for in a DB2 PM Statistics Report)

DB2 for MVS/ESA Version 4 Data Sharing Performance Topics Redbook (SG24-4611)

©Copyright 1999, Computer Management Sciences, Inc.

Rule DB2-606: Coupling facility write engine was not available

Finding: A coupling facility write engine was not available for coupling facility writes.

Impact: This finding can have a MEDIUM IMPACT or HIGH IMPACT on the performance of the DB2 subsystem.

Discussion: Rule DB2-220 through Rule DB2-222 describe how DB2 manages pages in a virtual buffer pool. Included in these descriptions is the concept of *deferred writes* of pages. With deferred writes, DB2 retains changed pages in the virtual buffer pool expecting that there is some probability that the updated pages will be referenced again. These deferred write pages are queued by data set until they are written when:

- A DB2 *checkpoint* is taken.
- The percentage of unavailable pages in a virtual buffer pool exceeds a preset limit called the Deferred Write Threshold (DWQT).
- The percentage of updated pages in a virtual buffer pool for a single data set exceeds a preset limit called the Vertical Deferred Write Threshold (VDWQT).

With data sharing, DB2 still performs deferred writes for DB2 table spaces, indexes or partitions. However, when an update is to a page set that has inter-DB2 R/W interest[1], DB2 writes the updated pages from the virtual buffer pool to the group buffer pool when:

- The transaction commits. When committing an updating transaction, pages that were updated but not yet written to the group buffer pool are synchronously written to the group buffer pool.
- The Deferred Write Threshold (DWQT) or the Vertical Deferred Write Threshold (VDWQT)[2] is reached. Deferred write pages in the virtual buffer pool are synchronously written to the group buffer pool.

---

[1] Inter-DB2 R/W interest is a property of data in a table space, index, or partition that has been opened by more than one member of a data sharing group and that has been opened for writing by at least one of those members. When there is inter-DB2 R/W interest in a particular table space, index, or partition, it is dependent on the group buffer pool associated with the object. The object is then said to be *GBP-dependent* (group buffer pool dependent).

[2] DB2 for MVS/ESA Version 4 Data Sharing Performance Topics Redbook (SG24-4611) states that DB2 writes updated pages to the group buffer pool when 50% of VDWQT is reached, and it makes no mention of a percentage of DWQT. Other documents shown in the references do not discuss a percentage of these thresholds being used to trigger writing updated pages to the group buff er pool. I've not been able to verify this apparent conflict between the two sets of IBM documents.

©Copyright 1999, Computer Management Sciences, Inc.

- The buffer pool is short of reassignable buffers because writes to the group buffer pool cannot keep up with update activity in the buffer pool. The shortage of buffers also can occur when the deferred write thresholds are too high, or if the application is not committing frequently.

- An updated page has stayed in the buffer pool for a long period of time since it was last referenced or updated. This can happen in situations such as with a long running transaction that doesn't issue frequent commits. In this case, a system checkpoint will cause deferred write pages in the virtual buffer pool to be synchronously written to the group buffer pool.

- A page is required for update by another system because there is no conflict on transaction locking (such as page sets that are using row locking, index pages, space map pages, etc.).

When a page of data is written to the group buffer pool, all copies of that page cached in other members' buffer pool are invalidated. This means that the next time one of those members needs that page, the page must be refreshed from the group buffer pool (or DASD).

The coupling facility write operations are implemented by DB2 subtasks, called *write engines*. The maximum number of concurrent write subtasks is a constant designed into DB2 (the maximum *write engines*). Coupling facility write operations are suspended if there are no more write engines when DB2 wishes to write pages to the coupling facility

When a "must complete" operation to the GBP fails (write at commit or rollback, read for rollback or restart), the page is put in the *logical page list* (LPL) and becomes unavailable. The START DATABASE command must be used to recover pages from the LPL.

The most likely cause of unavailable coupling facility write engines is that either a DB2 checkpoint was taken or a transaction commit occurred, and a large number of deferred write pages were written from the virtual buffer pool to the coupling facility.

CPExpert compares the QBGLSU variable in DB2GBPST (the number of times a coupling facility write engine was not available) with the **QBGLSU** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-606 when the number of times a coupling facility write engine was not available exceeds the value specified by the **QBGLSU** guidance variable.

The default value for the **QBGLSU** guidance variable is 0, indicating that CPExpert should produce Rule DB2-606 whenever a coupling facility write engine was not available.

The following example illustrates the output from Rule DB2-606:

```
RULE DB2-606: COUPLING FACILITY WRITE ENGINE WAS NOT AVAILABLE

   Group Buffer Pool 3: A coupling facility write engine was not available
   for coupling facility writes during the intervals shown below. You
   should consider increasing the size of the group buffer pool if these
   numbers are large and are not reflecting just a momentary surge in
   activity.  This situation occurred for Group Buffer Pool 3 during the
   intervals shown below:

                              GROUP BUFFER POOL       TIMES CF WRITE
   MEASUREMENT INTERVAL        ALLOCATED SIZE     ENGINE WAS UNAVAILABLE
   21:44-21:44, 08SEP1998            38M                   135
```

Suggestion: If Rule DB2-606 is produced more than occasionally, you should consider the following alternatives:

- **Reduce the local buffer pool thresholds**. Local buffer pool write activity can occur in bursts when a transaction commits and “floods” the group buffer pool with write requests for changed pages. This “flood” of write requests can cause the pool of write engines to be exhausted. The “flood” of write requests can be reduced by lowering the local buffer pool thresholds (DWQT or VDWQT), thus spreading the write requests more evenly over the life of the transaction.

  - Lower the Vertical Deferred Write Threshold (VDWQT) for the virtual buffer pool. A low (or even zero[3]) VDWQT value will cause DB2 to write updated pages more frequently to the group buffer pool. As a result, there would not be a large number of pages required to be written at checkpoint or commit.

  - Lower the Deferred Write Threshold (DWQT) for the virtual buffer pool. A low DWQT value will cause DB2 to write updated pages more frequently to the group buffer pool. As a result, there would not be a large number of pages required to be written at checkpoint or commit.

- **Verify GBPCACHE ALL is required**. Some application may have specified GBPCACHE ALL and DB2 was writing pages to the group buffer pool regardless of whether they are changed. In this case, you should review the total read hit percentage (changed and unchanged pages) to see if GBPCACHE ALL is productive.

[3] With VDWQT specification of zero, DB2 will write 32 pages at a time to the group buffer pool.

- **Issue COMMIT more frequently**. If the application can issue COMMIT more frequently, few pages would normally be written to the group buffer pool at each COMMIT. This would minimize the number of changed pages written from the local buffer pool when the application does issue COMMIT. Since updated pages in the virtual buffer pool are written to the group buffer pool at commit, increasing the frequency of commit would decrease the number of updated pages between commits.

- **Lower the checkpoint frequency for DB2.** This alternative should be carefully considered, however, as there are other implications that might cause performance problems (for example, DB2 writes all updated pages in the group buffer pool to DASD at checkpoint).

- **Reduce the data sharing level**. In some instances, conflicts between applications might be so great that it might be impossible to obtain satisfactory DB2 data sharing performance. In this case, it might be necessary to decrease the data sharing level associated with specific group buffer pools. To accomplish this, you could reassign page sets or partitions to different virtual buffer pools (with corresponding reassignment to different group buffer pools), create additional virtual buffer pools (with corresponding additional group buffer pools), or move some application between data sharing members and even to another processing time.

- You can alter CPExpert's analysis by modifying the **QBGLSU** guidance variable in USOURCE(DB2GUIDE).

Reference: DB2 for OS/390 Version 4: Data Sharing Planning and Administration Section 6.4.4.1 (Writing to the Group Buffer Pool)

DB2 for OS/390 Version 5: Data Sharing Planning and Administration Section 7.5.5.1 (Writing to the Group Buffer Pool)

DB2 for OS/390 Version 6: Data Sharing Planning and Administration Section 7.6.5.2 (Writing to the group buffer pool)

DB2 for MVS/ESA Version 4 Data Sharing Performance Topics Redbook (SG24-4611)

©Copyright 1999, Computer Management Sciences, Inc.

Rule DB2-641: Parallelism coordinator had to bypass a DB2 (buffers)

Finding: The parallelism coordinator in a data sharing group had to bypass a DB2 as an Assistant because of a buffer shortage in the assisting DB2.

Impact: This finding can have a MEDIUM IMPACT, or HIGH IMPACT on the performance of the DB2 subsystem. This finding is not applicable to DB2 Version 4 and earlier versions of DB2.

Discussion: DB2 can initiate multiple parallel operations to speed the response time of a query. The type of parallel operations differs depending on the version of DB2 that is installed.

- With DB2 Version 3, when DB2 plans to use sequential prefetch to access data from a table in a partitioned table space, it can initiate parallel I/O operations. *Query I/O parallelism* manages concurrent I/O requests for a single query, fetching pages into the buffer pool in parallel. For queries whose response time is dominated by I/O operations, the response time can be significantly reduced.

- With DB2 Version 4, parallel operation was expanded to enable true multi-tasking within a query, by including *Query CP parallelism*. With Query CP parallelism, a large query can be broken into multiple smaller queries. These smaller queries run simultaneously on multiple processors accessing data in parallel. The response time for I/O-intensive or processor-intensive queries can be significantly reduced.

- With DB2 Version 5, parallel operation was expanded by adding *Sysplex query parallelism* to take advantage of the sysplex processing capacity. With Sysplex query parallelism, DB2 can split a large query across different DB2 members in a data sharing group. Sysplex query parallelism is similar to Query CP Parallelism, with the difference being that DB2 can spread the queries across multiple systems in the sysplex.

With Sysplex query parallelism, one DB2 subsystem acts as a *Coordinator* while other DB2 subsystems on other systems in the data sharing group act as *Assistants*.

- The DB2 acting as a Coordinator is designated on the DSNTIPK panel by specifying YES in the COORDINATOR field. This designation allows that DB2 to send query work to other members of the data sharing group. If the COORDINATOR parameter is not YES at run time, the query runs within a single DB2.

©Copyright 1999, Computer Management Sciences, Inc.

- To be considered as a candidate for assisting the parallelism coordinator, YES must be specified in the ASSISTANT field of installation panel DSNTIPK. The ASSISTANT subsystem parameter controls whether this DB2 can receive parallel tasks from another DB2 in the data sharing group.

  Additionally, the Virtual Buffer Pool Assisting Parallel Sequential Threshold (VPXPSEQT) value must **not** be zero (either at bind time or at run time) for a DB2 to act as an Assistant. The VPXPSEQT threshold specifies the portion of the virtual buffer pool that might be used to assist with parallel operations initiated from another DB2 in the data sharing group.

  The VPXPSEQT percent is specified as a percentage of the Virtual Buffer Pool Parallel Sequential Threshold (VPPSEQT) value. Because VPXPSEQT is a subset of VPPSEQT and VPPSEQT is a subset of VPSEQT, the VPPSEQT and VPSEQT values must be greater than 0.

  The above statements simply mean that there must be buffers available at run time for a DB2 to act as an Assistant. Buffer availability is a function of:

  - The basic allocation of buffers to the buffer pool.

  - The percent of the buffer pool that can be used for sequentially accessed requests (controlled by the VPSEQT parameter).

  - The percent of the sequentially accessed pages that can be used for parallel I/O operations (controlled by the VPPSEQT parameter).

  - The percent of the pages that can be used for parallel I/O operations that can be used to support sysplex query as an Assistant (controlled by the VPXPSEQT parameter).

A shortage of buffer pool resources can cause Assistant members to be bypassed when distributing tasks around the data sharing group. This could lead to a less-than-optimal degree of distribution of the queries. It can also cause a query to run at a degree that is less than the optimal degree determined at bind time.

CPExpert compares the QZZCSKIP variable in DB2STATS (the number of times DB2 skipped an Assistant because of a buffer shortage) with the **QZZCSKIP** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-641 when the number of times DB2 skipped an

Assistant because of a buffer shortage exceeds the value specified by the QZZCSKIP guidance variable.

The default value for the QZZCSKIP guidance variable is 0, indicating that CPExpert should produce Rule DB2-641 whenever DB2 skipped an Assistant because of a buffer shortage.

Please note that this finding (Rule DB2-641) is **not** produced if VPSEQT, VPPSEQT, or VPXPSEQT parameters were zero[1]. The count in QZZCSKIP is not incremented if these parameters were zero. Rather, the QZZCSKIP count is incremented only when these values were **non-zero**, but DB2 had to bypass a member in the data sharing group because sufficient buffers were not available.

The following example illustrates the output from Rule DB2-641:

```
RULE DB2-641: PARALLELISM COORDINATOR HAD TO BYPASS A DB2 (BUFFERS)

   The parallelism coordinator in a data sharing group had to bypass a
   DB2 as an Assistant because of a buffer shortage in the assisting DB2.
   This situation occurred during the intervals shown below:

                                     TIMES ASSISTING DB2
   MEASUREMENT INTERVAL             WAS BYPASSED: BUFFERS
   11:59-12:29, 14OCT1999                     121
```

Suggestion: If Rule DB2-641 is regularly produced, you should consider the following alternatives:

- There is no information in the DB2 interval statistics that can be used to directly identify which buffers that are causing DB2 to skip an Assistant because buffers were unavailable. You can use performance class 8 and inspect the IFCID 0221 trace record to pinpoint which buffer pools cause DB2 members to be skipped.

- Examine the buffer pools for assisting DB2 members in the data sharing group to determine (1) which buffer pools are over-used and(2) which buffer pools have VPSEQT, VPPSEQT, or VPXPSEQT parameters that should be adjusted.

- Consider increasing the size of the buffer pool(s) identified as causing DB2 to skip the member, using the ALTER BUFFERPOOL VPSIZE parameter.

[1] Rule DB2-644 will be produced if a DB2 is designated an Assistant (ASSISTANT=YES), but either the VPSEQT, VPPSEQT, or VPXPSEQT parameter is zero at run time.

- Make sure that the sequential thresholds (VPSEQT, VPPSEQT, and VPXPSEQT values) are set properly for work file buffer pools. DB2 assumes that you have all your work files of the same size (4KB or 32KB) in the same buffer pool number on all DB2s participating in the data sharing group. DB2 makes run time decisions based on a single buffer pool. A lack of buffer pool resources for the work files can lead to a reduced degree of parallelism or cause the query to run sequentially.

- You can alter CPExpert's analysis by modifying the QZZCSKIP guidance variable in USOURCE(DB2GUIDE).

Reference: DB2 for OS/390 Version 5: Administration Guide
Section 5.4.1.7.2 (Variable Thresholds)
Section 5.10.8.7 (Tuning Parallel Processing)

DB2 for OS/390 Version 5: Installation Guide
Section 2.5.3 (Define Group or Member Panel: DSNTIPK)

DB2 for OS/390 Version 5 Data Sharing Planning and Administration
Section 7.3.3 (Monitoring and Tuning Parallel Queries)

DB2 for OS/390 Version 6 Data Sharing Planning and Administration
Section 7.4.3 (Monitoring and Tuning Parallel Queries)

Rule DB2-642: Parallel groups executed on a single DB2 (RR/RS)

Finding: Parallel groups executed on a single DB2 because the plan or package was bound with an isolation value of repeatable read or read stability.

Impact: This finding can have a MEDIUM IMPACT, or HIGH IMPACT on the performance of the DB2 subsystem. This finding is not applicable to DB2 Version 4 and earlier versions of DB2.

Discussion: DB2 can initiate multiple parallel operations to speed the response time of a query. The type of parallel operations differs depending on the version of DB2 that is installed.

- With DB2 Version 3, when DB2 plans to use sequential prefetch to access data from a table in a partitioned table space, it can initiate parallel I/O operations. *Query I/O parallelism* manages concurrent I/O requests for a single query, fetching pages into the buffer pool in parallel. For queries whose response time is dominated by I/O operations, the response time can be significantly reduced.

- With DB2 Version 4, parallel operation was expanded to enable true multi-tasking within a query, by including *Query CP parallelism*. With Query CP parallelism, a large query can be broken into multiple smaller queries. These smaller queries run simultaneously on multiple processors accessing data in parallel. The response time for I/O-intensive or processor-intensive queries can be significantly reduced.

- With DB2 Version 5, parallel operation was expanded by adding *Sysplex query parallelism* to take advantage of the sysplex processing capacity. With Sysplex query parallelism, DB2 can split a large query across different DB2 members in a data sharing group. Sysplex query parallelism is similar to Query CP Parallelism, with the difference being that DB2 can spread the queries across multiple systems in the sysplex.

Not all queries are eligible to participate in sysplex query parallel operations. Even if parallelism is turned on, not all read-only queries are eligible. For example, DB2 does not choose sysplex query parallelism if the plan or package is bound with RR or RS isolation.

CPExpert compares the QXISORR variable in DB2STATS (the number of parallel groups that executed on a single DB2 because the plan or package was bound with an isolation value of repeatable read or read stability) with the **QXISORR** guidance variable in USOURCE(DB2GUIDE). CPExpert

©Copyright 1999, Computer Management Sciences, Inc.

produces Rule DB2-642 when the number of parallel groups executed on a single DB2 because the plan or package was bound with an isolation value of repeatable read or read stability exceeds the value specified by the **QXISORR** guidance variable.

The default value for the **QXISORR** guidance variable is 0, indicating that CPExpert should produce Rule DB2-642 when any parallel groups executed on a single DB2 because the plan or package was bound with an isolation value of repeatable read or read stability.

Suggestion: If Rule DB2-223 is regularly produced, you should consider the following alternatives:

- If you want to use RR or RS isolation with sysplex query parallelism, you should bind with CS or UR. You then should use a LOCK TABLE table-name IN SHARE MODE statement before the query.

- You can alter CPExpert's analysis by modifying the **QXISORR** guidance variable in USOURCE(DB2GUIDE).

Reference: DB2 for OS/390 Version 5 Data Sharing Planning and Administration
Section 7.3.2 (Enabling Sysplex Query Parallelism)
Section 7.3.3 (Monitoring and Tuning Parallel Queries)

DB2 for OS/390 Version 6 Data Sharing Planning and Administration
Section 7.4.2 (Enabling Sysplex Query Parallelism)
Section 7.4.3 (Monitoring and Tuning Parallel Queries)

Rule DB2-643: Parallel groups executed on a single DB2

Finding: Parallel groups executed on a single DB2 because the DB2 system on which the plan or package was executed had COORDINATOR=NO specified.

Impact: This finding can have a MEDIUM IMPACT, or HIGH IMPACT on the performance of the DB2 subsystem. This finding is not applicable to DB2 Version 4 and earlier versions of DB2.

Discussion: DB2 can initiate multiple parallel operations to speed the response time of a query. The type of parallel operations differs depending on the version of DB2 that is installed.

- With DB2 Version 3, when DB2 plans to use sequential prefetch to access data from a table in a partitioned table space, it can initiate parallel I/O operations. *Query I/O parallelism* manages concurrent I/O requests for a single query, fetching pages into the buffer pool in parallel. For queries whose response time is dominated by I/O operations, the response time can be significantly reduced.

- With DB2 Version 4, parallel operation was expanded to enable true multi-tasking within a query, by including *Query CP parallelism*. With Query CP parallelism, a large query can be broken into multiple smaller queries. These smaller queries run simultaneously on multiple processors accessing data in parallel. The response time for I/O-intensive or processor-intensive queries can be significantly reduced.

- With DB2 Version 5, parallel operation was expanded by adding *Sysplex query parallelism* to take advantage of the sysplex processing capacity. With Sysplex query parallelism, DB2 can split a large query across different DB2 members in a data sharing group. Sysplex query parallelism is similar to Query CP Parallelism, with the difference being that DB2 can spread the queries across multiple systems in the sysplex.

Before you can use Sysplex query parallelism, you must configure the members of the data sharing group to allow it.

- Specifying YES in the COORDINATOR field of installation panel DSNTIPK to allow a DB2 to send query work to other members. The COORDINATOR subsystem parameter controls whether this DB2 can send parallel tasks out to other DB2s in the data sharing group. If the

©Copyright 1999, Computer Management Sciences, Inc.

COORDINATOR parameter is not YES **at run time**, the query runs within a single DB2.

- To be considered as candidates for assisting the parallelism coordinator, YES must be specified in the ASSISTANT field of installation panel DSNTIPK for other DB2 members in the data sharing group. The ASSISTANT subsystem parameter controls whether this DB2 can receive parallel tasks from another DB2 in the data sharing group[1].

When a plan or package is bound on a DB2 in which YES was specified for the COORDINATOR subsystem parameter, DB2 will decided whether to plan on allowing the plan or package to participate in sysplex parallel query. However, it is possible that when the plan or package actually runs, the COORDINATOR subsystem parameter might not be YES. There are two conditions where this can occur:

- The COORDINATOR parameter had been **changed** to NO when the program runs.

- The plan or package was bound on a DB2 with the COORDINATOR subsystem parameter set to YES, but the program is being run on a **different** DB2 that had the COORDINATOR value set to NO.

Regardless of the reason, the plan or package runs on a single DB2 If COORDINATOR is NO at run time.

CPExpert compares the QXCOORNO variable in DB2STATS (the number of number of parallel groups executed on a single DB2 because COORDINATOR was NO) with the **QXCOORNO** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-643 when the of parallel groups executed on a single DB2 because COORDINATOR was NO exceeds the value specified by the **QXCOORNO** guidance variable.

The default value for the **QXCOORNO** guidance variable is 0, indicating that CPExpert should produce Rule DB2-643 when any parallel groups executed on a single DB2 because COORDINATOR was NO.

Suggestion: You should consider the following alternatives if Rule DB2-643 is produced:

- Verify whether the plan or package should be participating in Sysplex query parallelism. If so, determine why the COORDINATOR value on the

[1] If this DB2 is the coordinator for a particular query, then its ASSISTANT parameter is not relevant.

DB2 processing the plan or package does not have YES for the COORDINATOR subsystem parameter.

- You can alter CPExpert's analysis by modifying the **QXCOORNO** guidance variable in USOURCE(DB2GUIDE).

Reference: DB2 for OS/390 Version 5: Installation Guide
Section 2.5.3 (Define Group or Member Panel: DSNTIPK)

DB2 for OS/390 Version 6: Installation Guide
Section 2.5.3 (Define Group or Member Panel: DSNTIPK)

Rule DB2-644: Assisting DB2 had VPSEQT, VPPSEQT, or VPXPSEQT = zero

Finding: CPExpert noticed that a DB2 had ASSISTANT=YES specified from the DSNTIPK panel, however a buffer pool had zero specified for the VPSEQT, VPPSEQT, or VPXPSEQT parameter. The parallelism coordinator in a data sharing group would bypass any DB2 as an Assistant if either of these parameters were zero.

Impact: This finding can have a MEDIUM IMPACT, or HIGH IMPACT on the performance of the DB2 subsystem. This finding is not applicable to DB2 Version 4 and earlier versions of DB2.

Discussion: With Sysplex query parallelism, one DB2 subsystem acts as a *Coordinator* while other DB2 subsystems on other systems in the data sharing group act as *Assistants*.

- The DB2 acting as a Coordinator is designated on the DSNTIPK panel by specifying YES in the COORDINATOR field. This designation allows that DB2 to send query work to other members of the data sharing group. If the COORDINATOR parameter is not YES at run time, the query runs within a single DB2.

- To be considered as a candidate for assisting the parallelism coordinator, YES must be specified in the ASSISTANT field of installation panel DSNTIPK. The ASSISTANT subsystem parameter controls whether this DB2 can receive parallel tasks from another DB2 in the data sharing group.

  Additionally, the Virtual Buffer Pool Assisting Parallel Sequential Threshold (VPXPSEQT) value must **not** be zero (either at bind time or at run time) for a DB2 to act as an Assistant. The VPXPSEQT threshold specifies the portion of the virtual buffer pool that might be used to assist with parallel operations initiated from another DB2 in the data sharing group.

  The VPXPSEQT percent is specified as a percentage of the Virtual Buffer Pool Parallel Sequential Threshold (VPPSEQT) value. Because VPXPSEQT is a subset of VPPSEQT and VPPSEQT is a subset of VPSEQT, the VPPSEQT and VPSEQT values must be greater than 0.

  The above statements simply mean that there must be buffers available at run time for a DB2 to act as an Assistant. Buffer availability is a function of:

©Copyright 1999, Computer Management Sciences, Inc.

- The basic allocation of buffers to the buffer pool.

- The percent of the buffer pool that can be used for sequentially accessed requests (controlled by the VPSEQT parameter).

- The percent of the sequentially accessed pages that can be used for parallel I/O operations (controlled by the VPPSEQT parameter).

- The percent of the pages that can be used for parallel I/O operations that can be used to support sysplex query as an Assistant (controlled by the VPXPSEQT parameter).

When the DB2 parallelism coordinator is distributing queries to members in the data sharing group, it will not consider an assisting DB2 member if the required buffer has a zero value for either VPSEQT, VPPSEQT, or VPXPSEQT. Since the VPXPSEQT threshold reflects the percent of the pages that can be used for parallel I/O operations that can be used to support sysplex query as an Assistant, a zero in any of these three parameters means that there can not be buffers available for use in support of sysplex queries. Consequently, the DB2 parallelism coordinator simply ignores the DB2 member even though ASSISTANT=YES has been specified for the DB2 member.

CPExpert selects DB2 subsystems that have specified ASSISTANT=YES (the QWPAASST parameter in T102S106 is “Y’). CPExpert tests whether either VPSEQT, VPPSEQT, or VPXPSEQT is zero for the selected member (variables QDBPVPSH, QDBPPSQT, and QDBPXSQT from DB2STAT2, respectively). CPExpert produces Rule DB2-644 when the selected member has a zero value for either of these parameters.

The following example illustrates the output from Rule DB2-644:

```
RULE DB2-644: ASSISTING DB2 HAD VPSEQT, VPPSEQT, OR VPXPSEQT = ZERO

   Buffer Pool 3:  This DB2 was specified as an assisting DB2 in a data
   sharing group (ASSISTANT = YES on panel DSNTIPK).  However, Buffer Pool
   3 had zero specified for VPXPSEQT, VPPSEQT, or VPSEQT.  Consequently,
   this buffer pool could not be used to support data sharing.  Unless
   relevant buffer pools have non-zero values for these parameters, the
   coordinating DB2 will bypass this DB2 when distributing work, even
   though this DB2 had ASSISTANT=YES specified on panel DSNTIPK.  Please
   "turn off" this rule for this buffer pool if you do not intend the
   buffer pool to be used in support of a data sharing group. This
   situation occurred for Buffer Pool 3 during the intervals shown below:

                                BUFFERS
   MEASUREMENT INTERVAL       ALLOCATED     VPSEQT    VPPSEQT    VPXPSEQT
   21:44-21:59, 08SEP1998       2,000         80         50          0
```

Suggestion: If Rule DB2-644 is produced, you should consider the following alternatives:

- Review the DB2 subsystem being analyzed by CPExpert to determine whether the DB2 subsystem should be participating as an Assistant in a data sharing group. If the DB2 subsystem should not be participating as an Assistant, correct the ASSISTANT=YES parameter on panel DSNTIPK.

- If the DB2 is to be participating as an Assistant in a data sharing group, review the buffer pool that is identified by Rule DB2-641. Determine whether this buffer pool, for this DB2 subsystem, is intended to be used in supporting sysplex parallel queries.

  - If the buffer pool **is to be used** in supporting sysplex parallel queries, correct the specification of VPSEQT, VPPSEQT, or VPXPSEQT parameters. These parameters should reflect a proper amount of buffers that can be used to support sequential requests, support parallel I/O operations, and support sysplex assisting parallel I/O operations, respectively.

  - Please "turn off" this rule for this buffer pool, using the procedures outlined in Section 3 of this document if the buffer pool identified by this rule is **not to be used** in supporting sysplex parallel queries.

    There is no information in the DB2 interval statistics to indicate whether a specific buffer pool is to be used in support of sysplex parallel queries. Consequently, CPExpert can only identify each buffer pool that could be participating (because the ASSISTANT=YES parameter has been specified for this DB2). CPExpert then verifies that all buffer pools in the DB2 subsystem have non-zero values for the critical parameters to allow the buffer pools to participate in sysplex parallel queries.

    It is likely that you do not intend all buffers to participate in sysplex parallel queries. However, CPExpert cannot identify these buffers from the data. Thus, you should "turn off' this rule for non-participating buffer pools.

Reference: DB2 for OS/390 Version 5: Administration Guide
Section 5.4.1.7.2 (Variable Thresholds)
Section 5.10.8.7 (Tuning Parallel Processing)

DB2 for OS/390 Version 5: Installation Guide
Section 2.5.3 (Define Group or Member Panel: DSNTIPK)

DB2 for OS/390 Version 5 Data Sharing Planning and Administration
Section 7.3.3 (Monitoring and Tuning Parallel Queries)

DB2 for OS/390 Version 6: Administration Guide
Section 5.4.1.6.2 (Thresholds you can change)
Section 5.11.7 (Tuning Parallel Processing)

DB2 for OS/390 Version 6: Installation Guide
Section 2.5.3 (Define Group or Member Panel: DSNTIPK)

DB2 for OS/390 Version 6 Data Sharing Planning and Administration
Section 7.4.3 (Monitoring and Tuning Parallel Queries)

©Copyright 1999, Computer Management Sciences, Inc.

Rule DB2-651: Lock Contention was high

Finding: The lock contention for DB2 group buffer pools was higher than guidance provided by IBM for normal lock contention.

Impact: This finding can have a LOW IMPACT, MEDIUM IMPACT, or IMPACT on the signaling performance of the sysplex. The level of impact depends on the amount of lock contention.

Discussion: *Locking* is a means of controlling concurrent events or access to data. DB2 uses locks to ensure that no process accesses data that has been changed (but not yet committed) by another process. Locking prevent concurrent users from accessing inconsistent data. A lock associates a DB2 resource with an application process in a way that affects how other processes can access the same resource. The resource can be a table space, a table (if table spaces are segmented), pages, and rows. The process associated with the resource is said to "hold" or "own" the lock. Please refer to rules in the DB2-3(series) for a discussion of normal DB2 locking.

With DB2 data sharing, DB2 must control concurrent events or access to data *within each member* of the DB2 data sharing group, and *between the DB2 data sharing group members*. Consequently, DB2 uses special data sharing locking and caching mechanisms to ensure data consistency.

When multiple members of a data sharing group have opened the same table space, index space, or partition, **and** at least one of them has opened it for writing, the data is said to be of *inter-DB2 read/write interest* to the members of the data sharing group. To control access to data that is of inter-DB2 interest, whenever the data is changed, DB2 caches it in a group buffer pool located in a coupling facility. This means that locks used in data sharing are global in scope.

DB2 maintains concurrency control between members of the DB2 data sharing group by using a lock structure in the coupling facility.. Consequently, many global locks are processed not only by the local IRLM but also by MVS's cross-system extended services (XES) and the lock structure in the coupling facility. Of course, there additional overhead is (potentially, there is **significant** additional overhead) related to concurrency control.

Lock contention occurs when one user wishes to access data and some other user has placed a lock on the data. The user wishing to access the

data usually is suspended until the data is available (that is, until the lock is released).

The coupling facility lock structure contains information used to determine cross-system contention on a particular resource. IRLM assigns (or "hashes") locked resources to an entry value in the lock structure in the coupling facility. IRLM uses the lock table to determine whether a resource is locked. If the lock structure defined on the coupling facility is too small, the hashing algorithm can select the same lock table entry for two different locks. This situation is termed *false lock contention*. The user wishing to access the locked data is suspended until it is determined that there is no real lock contention on the resource.

For good DB2 performance, it is essential to control the volume of global lock requests that are propagated to the coupling facility and to control the amount of real lock contention and false lock contention. Additionally, it is essential to make sure that any locking problems are not caused by inadequate or over-stressed data sharing resources, such as an undersized lock structure, or the over-utilization of the coupling facility or coupling facility channels (links).

SMF Type 74 (Subtype 4 - Coupling Facility Activity) records contain information describing the requests for data, the number of requests that were delayed because of lock contention, and the number of requests that encountered false lock contention. CPExpert selects information relating to the DB2 lock structure (the structure name contained in the R744SNAM variable ends with "_LOCK1"). CPExpert analyzes the selected information to determine whether an excessive percentage of requests encountered lock contention.

CPExpert computes the percent of lock requests that experienced lock contention by the following algorithm:

$$\textit{Percent lock contention} = \frac{\textit{Lock requests encountering lock contention}}{\textit{Lock requests}} \quad \textit{or} \quad \frac{\textit{R744SCN}}{\textit{R744STRC}}$$

CPExpert compares the computed percent of requests that experienced lock contention with the **LOCKCONT** guidance variable in USOURCE(DB2GUIDE).

CPExpert produces Rule DB2-651 when the percent of lock contention exceeds the value specified by the LOCKCONT variable. To avoid spurious findings, CPExpert ignores any measurement interval in which there were less than 1000 lock requests.

The default value for the LOCKCONT variable is 1%, indicating that CPExpert should produce Rule DB2-651 when more than one percent of the requests were delayed because of lock contention. IBM's *Data Sharing: Planning and Administration* document suggests that DB2 global lock contention should be less than five percent, and preferably less than one percent. Consequently, the default value for the LOCKCONT guidance variable for DB2 lock structure is 1%. You should adjust the variable if you feel that Rule DB2-651 is produced too often and you are unable to take corrective action.

The following example illustrates the output from Rule DB2-651:

```
RULE DB2-651: LOCK CONTENTION WAS HIGH

   DSNDB1G_LOCK1:  The lock contention for this structure was higher than
   normal.  Higher lock contention can result in an increase in central
   processor utilization and a reduction in throughput.  If this finding
   continues to occur, you should review the alternatives listed in the
   DB2 Component User Manual.  If you are unable to take action, you
   should consider increasing the LOCKCONT guidance variable, located
   in USOURCE(DB2GUIDE).  The LOCKCONT variable currently is 1%.

                                 TOTAL LOCK   REQUESTS WITH   PERCENT LOCK
   MEASUREMENT INTERVAL           REQUESTS   LOCK CONTENTION   CONTENTION
   11:00-11:30,14OCT1999          208,623        10,941             5
   11:30-12:00,14OCT1999          199,210         9,507             5
```

Suggestion: If Rule DB2-651 is produced regularly, you should consider the following alternatives:

- Determine whether there is excessive time needed to resolve lock contention. When there is contention on a hash class, MVS uses XCF messages to resolve the conflict. This is how it determines which specific resources are involved in the contention, or if the contention is false. For speedy resolution of contention situations, make sure there is no queuing of messages for XCF message buffers. You can use the MVS Component or WLM Component of CPExpert to detect whether queuing is excessive.

- Avoid false lock contention when possible. Rule DB2-652 analyzes excessive false lock contention, and suggests alternatives if there was an excess of false lock contention.

- With DB2 Version 4 or DB2 Version 5, you can use Type 2 Indexes rather than Type 1 indexes (Type 1 indexes are not available with DB2 Version 6). Type 2 indexes are a better choice than Type 1, because Type 2 indexes rely on data-only locking, which can reduce the total

number of locks taken. The fewer locks that are needed, the fewer locks that need to be propagated beyond the local IRLM.

- Beginning with DB2 Version 5, if you are purposefully creating affinity between data partitions and DB2 members, you can use the LOCKPART option of CREATE and ALTER TABLESPACE to indicate that you want individual partitions locked only as they are accessed. Please refer to the “Tuning your use of Locks” section of IBM’s *Data Sharing: Planning and Administration* documents (see Reference in this document) for a discussion of the LOCKPART option.

- A frequent cause of high lock contention is batch jobs running against shared databases. If this is the cause, you may be able to reschedule the batch jobs to resolve the lock contention problem.

- Review the use of control records; that is, records that are accessed by most applications. If they have to be updated, it is important to remember that the CI or physical block is locked from other subsystems until the updates are committed.

- Refer to the "Improving Concurrency" section in the DB2 Administration Guide for general recommendations on how to reduce lock contention..

- If you decide that the DB2 application design is causing lock contention, you should refer to the DB2 Application Programming and SQL Guide for detailed suggestions about how to avoid or minimize lock contention.

- As mentioned earlier, the default value of 1% for the LOCKCONT guidance variable for DB2 lock structure is based on IBM’s guidance. You should adjust the variable if you feel that Rule DB2-651 is produced too often and you are unable to take corrective action.

Reference: OS/390: Setting Up a Sysplex (GC28-1779)
Section 6.3.1: Lock Contention

DB2 for MVS/ESA Version 4 Data Sharing: Planning and Administration
Section 6.3.2: Tuning Your Use of Locks

DB2 for MVS/ESA Version 5 Data Sharing: Planning and Administration
Section 7.4.2: Tuning Your Use of Locks

DB2 for MVS/ESA Version 6 Data Sharing: Planning and Administration
Section 7.5.2: Tuning your use of locks

"Parallel Sysplex Performance: tuning tips and techniques,"
Kelley, Joan (IBM, Poughkeepsie, NY), SHARE 86, February 1996.

Rule DB2-652: False Lock Contention was high

Finding: The false lock contention for the DB2 lock structure was higher than guidance provided by IBM.

Impact: This finding can have a LOW IMPACT, MEDIUM IMPACT, or IMPACT on the performance of the data sharing group. The level of impact depends on the amount of false lock contention. However, when analyzing the impact of this finding, you should keep in mind that (1) false lock contention requires overhead and (2) false lock contention is unnecessary as it normally can be eliminated.

Discussion: *Locking* is the mechanism used to reserve all or part of a database so that other programs will not be able to update the data until the user placing the lock has finished processing the data. By locking the data, users can be sure that the information they are processing is current. Without locking, users might lose updates or access invalid or incomplete data. Locking is necessary, of course, only if one or more of the users of the data will be performing updates. If no updating of the data is performed, locking is unnecessary; the data may be concurrently accessed by any number of user without worry that the data is incomplete or invalid.

Lock contention occurs when one user wishes to access data and some other user has placed a lock on the data. The user wishing to access the data usually is suspended until the data is available (that is, until the lock is released). Techniques such as separating data, choosing locking parameters, and monitoring for contention can be used to provide a balance between concurrency of access, isolation and integrity of data, and efficient use of system resources..

The coupling facility lock structure contains information used to determine cross-system contention on a particular resource. IRLM assigns (or "hashes") locked resources to an entry value in the lock structure in the coupling facility. IRLM uses the lock table to determine whether a resource is locked. If the lock structure defined on the coupling facility is too small, the hashing algorithm can select the same lock table entry for two different locks. This situation is termed *false lock contention*. The user wishing to access the locked data is suspended until it is determined that there is no real lock contention on the resource.

SMF Type 74 (Subtype 4 - Coupling Facility Activity) records contain information describing the requests for data, the number of requests that were delayed because of lock contention, and the number of requests that

©Copyright 1997, Computer Management Sciences, Inc.

encountered false lock contention. CPExpert selects information relating to the DB2 lock structure (the structure name contained in the R744SNAM variable ends with "_LOCK1"). CPExpert analyzes the selected information to determine whether an excessive percentage of requests encountered false lock contention.

CPExpert computes the percent of lock requests that experienced false lock contention by the following algorithm:

$$\textit{Percent false lock contention} = \frac{\textit{Lock requests encountering false lock contention}}{\textit{Lock requests}} \quad \textit{or} \quad \frac{\textit{R744SFCN}}{\textit{R744STRC}}$$

CPExpert produces Rule DB2-652 when more than 0.1% of the total requests encountered false lock contention. This threshold (0.1% of total requests) is based on the advice from several IBM documents listed in the Reference section of this discussion.

To avoid spurious findings, CPExpert ignores any measurement interval in which there were less than 1000 lock requests. CPExpert also checks that the overall lock contention was at least as high as 25% of the value specified in the **LOCKCONT** guidance variable.

The following example illustrates the output from Rule DB2-652:

```
RULE DB2-652: FALSE LOCK CONTENTION WAS HIGH

   DB2DBP2_LOCK1:  The number of locks with false contention should be less
   than 0.1 of the total requests, based on IBM's guidance.  The false
   lock contention for this structure exceeded the guideline.  False lock
   contention occurs when the hashing algorithm hashes to the same lock
   table entry (hash value) for two different locks.  Please refer to Rule
   DB2-652 in the DB2 Component User Manual for suggestions on ways to
   reduce false lock contention.

                                TOTAL LOCK    FALSE LOCK     PERCENT FALSE
   MEASUREMENT INTERVAL          REQUESTS     CONTENTION    LOCK CONTENTION
   11:15-11:30,06EP1999          14,676         2,541             17
```

Suggestion: IBM's *Data Sharing: Planning and Administration Guide* suggests the following ways to reduce false lock contention:

- As much as possible, reduce the amount of real lock contention in your applications. Please refer to Rule DB2-651 for suggestions of ways to reduce real lock contention in your environment.

- Specify a larger size for the lock structure and manually rebuild the lock structure.

- Make sure the value for LOCK ENTRY SIZE is not too large for the number of members in your group. The LOCK ENTRY SIZE parameter for the first IRLM to join the group determines the size of each lock entry in the lock table. Please note that if you do increase the size of the lock structure, you should increase by a power of 2 to avoid wasting storage. A lock entry size of 2 allows twice as many lock entries as compared to a lock entry size of 4.

- Decrease IRLM MAXUSRS from 32 to the actual number of users (IRLM subsystems). MAXUSRS is a parameter on the IRLM procedure. It specifies the number of users of the global lock structure. The users are online systems and any additional DLI batch jobs that could connect to the global lock structure. By reducing MAXUSRS to the number of actual users, the size of each lock table entry may be reduced, allowing more locks in the table and hence less false lock contention. For seven or fewer MAXUSRS, the lock entries are two bytes. For 8 to 23 MAXUSRS, the lock entries are four bytes. For 24 to 32 MAXUSRS, the lock entries are eight bytes.

Reference: OS/390: Setting Up a Sysplex (GC28-1779)
Section 6.3.1: Lock Contention

DB2 for MVS/ESA Version 4 Data Sharing: Planning and Administration
Section 6.3.2.3: Avoid False Contention

DB2 for MVS/ESA Version 5 Data Sharing: Planning and Administration
Section 7.4.2.3: Avoid False Contention

DB2 for MVS/ESA Version 6 Data Sharing: Planning and Administration
Section 7.5.2.2: Avoid false contention

"Parallel Sysplex Performance: tuning tips and techniques,"
Kelley, Joan (IBM, Poughkeepsie, NY), SHARE 86, February 1996.

©Copyright 1997, Computer Management Sciences, Inc.

Rule DB2-660: Synchronous service time was high for the indicated structure

Finding: The synchronous service time for the indicated structure exceeded the guidance provided to CPExpert.

Impact: This finding can have a LOW IMPACT, MEDIUM IMPACT, or HIGH IMPACT on the performance of the DB2 data sharing group. The level of impact depends on the amount of delay to synchronous requests.

Discussion: Most coupling facility accesses in support of DB2 applications are performed synchronously. This is done to avoid the overhead of suspend and resume processing of the application or DB2 system task and avoid passing of messages between members of the DB2 data sharing group. When coupling facility accesses are made synchronously, the task on the MVS sender uses CPU instruction cycles[1] waiting for the response from the coupling facility (this waiting for the response is called a “CPU spin-loop”).

Signalling requests to a coupling facility can occur only if a subchannel to the coupling facility is available. If no subchannel is available, the cross-system extended services (XES) will either enter a CPU "spin loop" waiting for a subchannel to become available or queue the request until a subchannel is available. The type of action taken by XES depends on whether the request was specified as synchronous or asynchronous.

- Synchronous requests require that a response be received from the coupling facility before the requesting application continues execution. Synchronous requests would be used, for example, to request a lock. In this example, the application cannot proceed until the lock is granted.

  For synchronous requests, XES will either (1) satisfy the request if a subchannel is available, (2) enter CPU "spin-looping" until a subchannel is available[2] and the request is satisfied, or (3) convert the synchronous request to an asynchronous request if the type of request permits the conversion.

---

[1] This CPU processing is charged back to the application task or the DB2 system service unit.

[2] Please note that the coupling facility is simply another processor that acts as a server in a parallel sysplex. If the central processor (CP) is significantly fast than the coupling facility, synchronous delays can significantly degrade performance. This is because a faster processor is waiting on a slower processor (although the faster processor appears “busy” while it is in a CPU spin loop). Consequently, systems that are faster than the coupling facility effectively lose more potential capacity during synchronous processing of requests by the coupling facility than do systems of the same speed as the coupling facility. For example, suppose an IRLM requests a global logical lock and the request is processed synchronously in the coupling facility. This example is more costly for an IRLM on a 9021 processor than on a 9672 processor, because the 9021 can process approximately 4 to 6 times more instructions than a 9672-Rx1 in the same time.

©Copyright 1997, Computer Management Sciences, Inc.

- Asynchronous requests allow the requesting application to continue processing and be notified when the request is completed. For asynchronous requests, XES either starts or queues the request and returns control to the application issuing the request.

The type (synchronous or asynchronous) of request that is issued generally depends on the type of structure.

- Some requests can be satisfied only by synchronous requests. For example, signals generated by XES itself will always be synchronous and will not be converted to asynchronous requests. Lock requests for DB2 locks also will always be synchronous and will not be converted to asynchronous requests.

- Some requests can be issued as either synchronous or asynchronous requests, depending on the application's use of the structure (for example, cache structure requests can be issued as either synchronous or asynchronous).

- Some requests can be issued as synchronous but will be converted to asynchronous if the subchannels are busy[3] unless the application has indicated that the synchronous cannot be converted.

The time spent waiting for subchannels to become free for synchronous requests not only delays the request (and consequently delays the application waiting on the request), but wastes processor resources since the processor is in a CPU "spin-loop" waiting for the synchronous request to be satisfied. As mentioned earlier in a footnote, the cost of a spin loop can be **significant** if a faster central processor is connected to a slower coupling facility processor.

The service time represents the time from when MVS issues a command for the coupling facility until the return from the command is recognized by MVS. The time includes time spent on the coupling facility links, the coupling facility processing time, and the time for MVS to recognize that the command was completed. The service time varies based on whether subchannels are available, the activity level of the coupling facility itself, and on the amount of data being processed.

IBM suggests that the service time for synchronous requests should be less than 250-350 microseconds, depending upon the length of the request. The service time for lock structures should be considerably less than 250 microseconds, since lock structure requests are small.

[3]The application can specify which requests must be satisfied as synchronous and which can be converted to asynchronous. XES will automatically convert requests from synchronous to asynchronous if all signalling paths are busy, unless the application specifies that the conversion is not to be done.

©Copyright 1997, Computer Management Sciences, Inc.

CPExpert compares the synchronous service time (R744SSTM) against the **SYNCSRV** variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-660 when the synchronous service time is greater than the SYNCSRV guidance variable.

The default value for the SYNCSRV variable is 350, indicating that CPExpert should produce Rule DB2-660 when synchronous service time is more than 350 microseconds. CPExpert subtracts 100 microseconds from the SYNCSRV guidance variable if evaluating a lock structure. Thus, CPExpert will produce Rule DB2-660 when the service time for lock structures is greater than 250 microseconds.

To avoid spurious findings, CPExpert ignores any measurement interval in which there were less than 1000 synchronous requests.

The following example illustrates the output from Rule DB2-660:

```
RULE DB2-660: SERVICE TIME WAS HIGH FOR SYNCHRONOUS REQUESTS

   DSNDB1G_GBP3:  The service time for this structure has exceeded the
   guidelines for synchronous requests.  Service time is accumulated from
   the time MVS issues a command for the coupling facility until the return
   from the command is recognized by MVS.  Service time is recorded for
   each structure used by each system.  You can alter the times used by
   CPExpert in making this finding by altering the SYNCSRV guidance
   variable in USOURCE(DB2GUIDE) if you are unable to make changes to
   reduce service time for the structure. The SYNCSRV guidance variable
   currently is 200 and the LOCKSRV guidance variable is 250.

                                 TOTAL SYNC      AVERAGE SERVICE
   MEASUREMENT INTERVAL           REQUESTS     TIME (MICROSECONDS)
   11:00-11:30,14OCT1999          27,774                  410
   11:30-12:00,14OCT1999          24,322                  603
```

Suggestion: CPExpert suggests that you consider the following alternatives if Rule DB2-660 is produced:

- Synchronous command processing is performed by the central processor. You should make certain that sufficient CPU resources have been allocated to the coupling facility LPAR.

- Examine whether the structure activity is balanced between coupling facilities. You may wish to consider redistributing the structures among the coupling facilities if a significant imbalance exists.

- You should consider whether additional coupling facility links should be added between the MVS processor and the coupling facility. Each coupling facility link will contribute two subchannels.

- If possible, you should consider influencing the exploiters of the coupling facilities to lower the activity rate to the coupling facilities. Taking other tuning actions (especially if indicated by other rules produced by CPExpert) may reduce the number of XCF signals. For example, signal activity can be lowered by (1) reducing lock contention, (2) reducing false lock contention, or (3) tuning the XCF to eliminate signals related to the expansion of a transport class size.

- If none of the above alternatives are appealing, you may wish to change the guidance to CPExpert by altering the **SYNCSRV** guidance variable in USOURCE(DB2GUIDE).

  IBM provides the following example service times based on measurements of coupling facility lock requests for various combinations of sender CPCs and CFs. These measurements were reported in *S/390 MVS Parallel Sysplex Configuration, Volume 2: Cookbook*, document Number SG24-2076-00.

Central Processor	**Coupling facility**	**Service Time**, **Lock Requests** (microseconds)
9672R1 based	9674C01	250
9672R2/R3 based	9674C02/3	180
9672G3	9674C04	140
9672G4	9674C04	100
9672G4	9674C05	70
9021 711 based	9674C01	160
9021 711 based	9674C02/3	130
9021 711 based	9674C04	100
9021 711 based	9021 711 based	80

Reference: Washington System Center Flash 9609 ("CF Reporting Enhancements to RMF 5.1")

S/390 MVS Parallel Sysplex Configuration, Volume 2: Cookbook, document Number SG24-2076-00.

"Parallel Sysplex Performance: tuning tips and techniques," Kelley, Joan (IBM, Poughkeepsie, NY), SHARE 86, February 1996.

©Copyright 1997, Computer Management Sciences, Inc.

Rule DB2-661: Asynchronous service time was high for group buffer pool

Finding: The asynchronous service time for the indicated group buffer pool exceeded the guidance provided to CPExpert.

Impact: This finding can have a LOW IMPACT, MEDIUM IMPACT, or HIGH IMPACT on the signalling performance of the sysplex. The level of impact depends on the amount of delay to asynchronous requests and how important the requests are.

Discussion: Signalling requests to a coupling facility can occur only if a subchannel to the coupling facility is available. If no subchannel is available, the cross-system extended services (XES) will either enter a CPU "spin loop" waiting for a subchannel to become available or queue the request until a subchannel is available. The type of action taken by XES depends on whether the request was specified as synchronous or asynchronous, and a member of a DB2 data sharing group makes both synchronous and asynchronous requests.

- Synchronous requests require that a response be received from the coupling facility before the requesting application continues execution. Synchronous requests would be used, for example, to request a lock. In this example, the application cannot proceed until the lock is granted.
  For synchronous requests, XES will either (1) satisfy the request if a subchannel is available, (2) enter CPU "spin-looping" until a subchannel is available and the request is satisfied, or (3) convert the synchronous request to an asynchronous request if the type of request permits the conversion.

- Asynchronous requests allow the requesting application to continue processing and be notified when the request is completed. For asynchronous requests, XES either starts or queues the request and returns control to the application issuing the request.

The type (synchronous or asynchronous) of request that is issued generally depends on the type of structure.

- Some requests can be satisfied only by synchronous requests (for example, signals generated by XES itself will always be synchronous and will not be converted to asynchronous requests). DB2 lock requests also will always be only synchronous requests.

©Copyright 1997, Computer Management Sciences, Inc.

- Some requests can be issued as either synchronous or asynchronous requests, depending on the application's use of the structure (for example, DB2 group buffer pool requests can be issued as either synchronous or asynchronous).

- Some requests can be issued as synchronous but will be converted to asynchronous if the subchannels are busy[1] unless the application has indicated that the synchronous cannot be converted.

Each coupling link has two subchannels. Most interactions with the coupling facility are synchronous. When a group buffer pool access has to wait for a subchannel, it is converted into an asynchronous request. The conversion involves a suspension and resumption of the original task with the corresponding overhead. To reduce this overhead, IBM suggests that coupling facilities should be configured with two links to each processor.

Additionally, the time spent waiting for subchannels to become free for asynchronous requests delays the request (and consequently delays the application waiting on the request).

The service time represents the time from when MVS issues a command for the coupling facility until the return from the command is recognized by MVS. The time includes time spent on the coupling facility links, the coupling facility processing time, any delay time while the request is queued, and the time for MVS to recognize that the command was completed. The service time varies based on whether subchannels are available, the activity level of the coupling facility itself, and on the amount of data being processed.

IBM suggests that the service time for asynchronous requests should be less than 5000 microseconds (or 5 milliseconds).

CPExpert compares the asynchronous service time (R744ASTM) against the **ASYNCSRV** variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-661 when the asynchronous service time is greater than the ASYNCSRV guidance variable.

The default value for the ASYNCSRV variable is 5000, indicating that CPExpert should produce Rule DB2-661 when asynchronous service time is more than 5000 microseconds.

The following example illustrates the output from Rule DB2-661:

[1]The application can specify which requests must be satisfied as synchronous and which can be converted to asynchronous. XES will automatically convert requests from synchronous to asynchronous if all signalling paths are busy, unless the application specifies that the conversion is not to be done.

```
RULE DB2-661: SERVICE TIME WAS HIGH FOR ASYNCHRONOUS REQUESTS

   DSNDB1G_GBP3:  The service time for this structure has exceeded the
   guidelines for asynchronous requests.  Service time is accumulated from
   the time MVS issues a command for the coupling facility until the return
   from the command is recognized by MVS.  Service time is recorded for
   each structure used by each system.  You can alter the times used by
   CPExpert in making this finding by altering the ASYNCSRV guidance
   variable in USOURCE(DB2GUIDE) if you are unable to make changes to
   reduce service time for the structure.  The ASYNCSRV variable currently
   is 5000 microseconds.

                                TOTAL  ASYNC      AVERAGE SERVICE
   MEASUREMENT INTERVAL           REQUESTS      TIME (MILLISECONDS)
   11:00-11:30,14OCT1999             246               5.362
```

Suggestion: CPExpert suggests that you consider the following alternatives if Rule DB2-661 is produced:

- Asynchronous command processing is performed primarily by the I/O processor. You should make certain that sufficient CPU resources have been allocated to the coupling facility LPAR.

- Examine whether the structure activity is balanced between coupling facilities. You may wish to consider redistributing the structures among the coupling facilities if a significant imbalance exists.

- You should consider whether additional coupling facility links should be added between the MVS processor the coupling facility. Each coupling facility link will contribute two subchannels.

- If possible, you should consider influencing the exploiters of the coupling facilities to lower the activity rate to the coupling facilities. Taking other tuning actions (especially if indicated by other rules produced by CPExpert) may reduce the number of XCF signals. For example, signal activity can be lowered by (1) reducing lock contention, (2) reducing false lock contention, or (3) tuning the XCF to eliminate signals related to the expansion of a transport class size.

If none of the above alternatives are appealing, you may wish to change the guidance to CPExpert by altering the **ASYNCSRV** guidance variable in USOURCE(DB2GUIDE).

©Copyright 1997, Computer Management Sciences, Inc.

Reference: Washington System Center Flash 9609 ("CF Reporting Enhancements to RMF 5.1")

"Parallel Sysplex Performance: tuning tips and techniques,"
Kelley, Joan (IBM, Poughkeepsie, NY), SHARE 86, February 1996.

Rule DB2-665: Too many synchronous requests were changed to asynchronous requests

Finding: An unacceptably large number of synchronous requests were changed to asynchronous requests.

Impact: This finding can have a MEDIUM IMPACT, or HIGH IMPACT on the signalling performance of the sysplex.

Discussion: Each coupling link has two subchannels. Signalling requests to a coupling facility can occur only if a subchannel to the coupling facility is available. If no subchannel is available, the cross-system extended services (XES) will either enter a CPU "spin loop" waiting for a subchannel to become available or queue the request until a subchannel is available. The type of action taken by XES depends on whether the request was specified as synchronous or asynchronous.

- Synchronous requests require that a response be received from the coupling facility before the requesting application continues execution. Synchronous requests would be used, for example, to request a lock. In this example, the application cannot proceed until the lock is granted.

  For synchronous requests, XES will either (1) satisfy the request if a subchannel is available, (2) enter CPU "spin-looping" until a subchannel is available and the request is satisfied, or (3) convert the synchronous request to an asynchronous request if the type of request permits the conversion.

- Asynchronous requests allow the requesting application to continue processing and be notified when the request is completed. For asynchronous requests, XES either starts or queues the request and returns control to the application issuing the request.

- Some requests can be issued as synchronous but will be converted to asynchronous if the subchannels are busy[1] unless the application has indicated that the synchronous cannot be converted. Most DB2 interactions with the coupling facility are synchronous. When a group buffer pool access has to wait for a subchannel, it is converted into an asynchronous request. The conversion involves a suspension and

[1]The application can specify which requests must be satisfied as synchronous and which can be converted to asynchronous. XES will automatically convert requests from synchronous to asynchronous if all signalling paths are busy, unless the application specifies that the conversion is not to be done.

©Copyright 1997, Computer Management Sciences, Inc.

resumption of the original task with the corresponding overhead. To reduce this overhead, coupling facilities should be configured with two links to each processor.

There is a significant overhead associated with changing synchronous requests to asynchronous requests. XES must initially detect that the synchronous request is not going to be satisfied, the request must be changed to asynchronous, the request is queued, XES must detect when a subchannel is available, de-queue the asynchronous request, and process the asynchronous request. Not only is this overhead expensive in terms of resource consumption, but it is expensive in terms of delay to the application issuing the synchronous request.

The number of synchronous requests changed to asynchronous should be very low, to minimize the overhead and the delay to applications. IBM suggests that action should be taken when more than 10% of the synchronous requests are changed to asynchronous requests. This percentage is, obviously, dependent upon the application and the importance of the requests.

CPExpert computes the percent of synchronous requests changed to asynchronous requests by the following algorithm:

$$\textit{Percent synchronous changed} = \frac{\textit{Synchronous requests changed}}{\textit{Synchronous requests}} \quad \textit{or} \quad \frac{\textit{R742SSTA}}{\textit{R742SSRC}}$$

CPExpert produces Rule DB2-665 when this percent is greater than the SYNCCHG guidance variable.

The default value for the SYNCCHG guidance variable is 10, indicating that CPExpert should produce Rule DB2-665 when more than 10% of the synchronous requests are changed to asynchronous requests.

The following example illustrates the output from Rule DB2-665:

```
RULE DB2-665: TOO MANY SYNCHRONOUS REQUESTS WERE CHANGED TO ASYNCHRONOUS

   DSNDB1G_GBP8:  The structure experienced too many requests being changed
   from synchronous to asynchronous.  If MVS determines that a synchronous
   request will be significantly delayed (perhaps because subchannels are
   busy), MVS will change the request to an asynchronous request (note that
   synchronous lock requests are not changed).  This finding could indicate
   that you need additional coupling facility links. You can alter the
   analysis by CPExpert in making this finding by altering the SYNCCHG
   guidance variables in USOURCE(DB2GUIDE).  The SYNCCHG variable
   currently is 10%.

                                TOTAL SYNC   SYNCH REQUESTS     PERCENT
   MEASUREMENT INTERVAL          REQUESTS   CHANGED TO ASYNCH   CHANGED
   11:00-11:30,14OCT1999          20,126          9,308           46.2
   11:30-12:00,14OCT1999          26,523          5,611           21.2
```

Suggestion: Changed requests normally are caused by subchannel unavailable conditions. CPExpert suggests that you consider the following alternatives if Rule DB2-665 is produced:

- You should make certain that sufficient CPU resources have been allocated to the coupling facility LPAR.

- You should consider whether additional coupling facility links should be added between the MVS processor the coupling facility. Each coupling facility link will contribute two subchannels.

- You should examine whether the structure activity is balanced between coupling facilities. You may wish to consider redistributing the structures among the coupling facilities if a significant imbalance exists.

- If possible, you should consider influencing the exploiters of the coupling facilities to lower the activity rate to the coupling facilities. Taking other tuning actions (especially if indicated by other rules produced by CPExpert) may reduce the number of XCF signals. For example, signal activity can be lowered by (1) reducing lock contention, (2) reducing false lock contention, or (3) tuning the XCF to eliminate signals related to the expansion of a transport class size.

- If none of the above alternatives are appealing, you may wish to change the guidance to CPExpert by altering the **SYNCCHG** guidance variable in USOURCE(DB2GUIDE).

Reference: Washington System Center Flash 9609 ("CF Reporting Enhancements to RMF 5.1")

"Parallel Sysplex Performance: tuning tips and techniques,"
Kelley, Joan (IBM, Poughkeepsie, NY), SHARE 86, February 1996.

©Copyright 1997, Computer Management Sciences, Inc.

Rule DB2-671: Excessive cross-invalidations due to directory entry reclaim

Finding: Directory entry reclaims resulted in a large number of cross-invalidation actions for members of the data sharing group.

Impact: This finding can have a MEDIUM IMPACT, or HIGH IMPACT on the performance of the DB2 subsystem. This finding can be made only if SMF Type 74 (Subtype 4 - Coupling Facility Activity, Cache Data Section) information is available for DB2 group buffer pools.

Discussion: The space allocated for a group buffer pool is divided into two parts: (1) directory entries and (2) data entries.

- *Directory entries* are used by the coupling facility to identify pages for which there is inter-DB2 R/W interest[1] and to identify the DB2 systems that have interest in the page.

  When there is inter-DB2 R/W interest in a table space or partition, DB2 registers pages when the pages are read into a local buffer pool. The registration is done in the page directory of the group buffer pool associated with the local buffer pool[2].

  There is one directory entry per page, regardless of the number of DB2 systems that have cached the page in their local buffer pool. This is because the directory entry simply identifies the page and indicates the specific DB2 systems that have an interest in the page.

  For example, suppose there were five DB2 systems in the sysplex (DB2A, DB2B, DB2C, DB2D, and DB2E). The directory entry for any particular page would have indicator information for all five DB2 systems. This is simply a 32-bit vector in the Directory Entry Information Block (DEIB). Each bit corresponds to one of the 32 possible systems in a sysplex. The bit is set to “1" if the associated system connected to the coupling facility has a registered interest in the page and set to “0" if the associated system does not have a registered interest in the page.

---

[1] Inter-DB2 R/W interest is a property of data in a table space, index, or partition that has been opened by more than one member of a data sharing group and that has been opened for writing by at least one of those members. When there is inter-DB2 R/W interest in a particular table space, index, or partition, it is dependent on the group buffer pool associated with the object. The object is then said to be *GBP-dependent* (group buffer pool dependent).

[2] The cached state of the page set or partition P-lock tells DB2 which data sharing protocols must be used to maintain inter-DB2 buffer coherency for the page set or partition. For example, a cached state of IS tells DB2 that whenever a page belonging to that page set or partition is read into the local buffer pool, the page must be registered to the coupling facility for cross-invalidation purposes. If the cached state were SIX, the coupling facility page registration would not be necessary.

©Copyright 1999, Computer Management Sciences, Inc.

Suppose further that only DB2A and DB2C had an interest in the particular page. The directory entry indicators (the bits in the DEIB vector) would reflect that only DB2A and DB2C had the interest. Thus, the coupling facility would send cross-invalidation signals to only DB2A and DB2C when the page was changed or when the directory entry must be reused.

- Data entries contain the actual page of data that is subject to inter-DB2 read/write interest. The data entries normally would contain only changed pages as these are the most common pages written to the coupling facility. The data entries might contain unchanged pages if GBPCACHE ALL were specified.

There is not normally a one-to-one ratio between directory entries and data pages. This is because directory entries reflect information about *any* page in *any* DB2 local buffer pool for a table space or partition in which there is R/W interest. Many (perhaps most) of these pages will be read-only pages. Consequently, only changed pages will be in the group buffer pool (unless GBPCACHE ALL were specified). Thus, normally there will be more directory entries than there are data entries. This ratio of directory entries to data entries can be is specified to DB2 (the default is 5 directory entries per data entry).

When DB2 attempts to register a page in the group buffer pool, either a directory entry already exists for the page, or a directory entry must be available to hold the information about the page.

- A directory entry for the page would already exist if another DB2 subsystem has registered the page. In this case, the coupling facility simply updates the directory entry to reflect that the DB2 attempting to register the page has an interest in the page.

- A directory entry must be available, if a directory entry does not exist for the page. In this case, either there is a free directory entry in the group buffer pool or a directory entry must be made available via the directory entry reclaim process.

  Directory entries in the group buffer pool can be free because they have never been used, they can be free because they have freed by the castout process, or they can be free because there is no more inter-DB2 R/W interest in the page set, or they can be free because all DB2 members had “unregistered” interest in the corresponding page.

If there are no free directory entries when DB2 attempts to register a page, directory entries must be reclaimed[3]. Directory entries can be *reclaimed* if they describe unchanged pages or if they describe pages that are not in a local buffer pool.

The unchanged pages could be in the group buffer pool because they originally were changed pages but have been castout (these pages remain cached in the group buffer pool until they are reclaimed). Additionally, the unchanged pages could be in the group buffer pool because they were placed there by a GBPCACHE ALL specification for the page set.

When directory entries are reclaimed to handle new work, cross-invalidation must occur for all members of the DB2 data sharing group that have those pages in their buffer pools, even when the data has not actually changed. This cross-invalidation process requires overhead, as all members must be notified that their copy of the data item is invalid.

When another member of the DB2 data sharing group needs access to the invalidated data item, the item must be re-read from DASD and registered with the coupling facility. Buffer pool pages must be continually refreshed from DASD when high cross-invalidation levels are reached. These processes can cause a significant overhead in DB2 data sharing. When there are insufficient directory entries, the directory reclaim activity can lead to "thrashing" and significant overhead and corresponding poor performance.

Directory reclaim activity can result in the following:

- Increased read I/O activity to the data base to re-acquire a referenced data item.

- Increased CPU utilization caused by registered interest in data items having to be re-acquired.

- Elongated transaction response times whenever a read miss occurs in the local buffer pool.

IBM states that the usual causes for directory reclaim activity are:

- The initial structure size is insufficient to support the amount of buffering in the local buffer pool.

[3] If no directory entries can be reclaimed, a *structure full* condition exists. This situation is analyzed by Rule DB2-601 and Rule DB2-602.

- Installations have increased the size of the local buffer pools without an accompanying increase in structure size.

- Installations have increased the number of users of the structure without a corresponding increase in the size of the group buffer pool.

CPExpert computes the percentage of **cross-invalidated** pages caused by directory reclaims relative to the total directory entry reclaims, based on the following algorithm:

$$\textit{percent cross invalidated pages} = \frac{R744CXDR}{R744CDER}$$

where:

R744CXDR = Cross-invalidate directory reclaim counter. This is the number of pages that were cross-invalidated because the corresponding directory entry had been reclaimed.

R744CDER = Directory entry reclaim counter. This is the total number of directory entries that were reclaimed.

CPExpert compares the percentage for cross-invalidated pages because of directory reclaims with the **PCTXIGBP** guidance variable in USOURCE(DB2GUIDE). CPExpert produces Rule DB2-671 when the computed percentage of cross-invalidated pages because of directory reclaims is less than the value specified by the **PCTXIGBP** guidance variable.

IBM states that for **best** DB2 performance, there should no cross-invalidate actions due to directory reclaims. Consequently, the default value for the **PCTXIGBP** guidance variable is 0, indicating that CPExpert should produce Rule DB2-671 when any cross-invalidate actions were taken because of directory reclaims

The following example illustrates the output from Rule DB2-671:

```
RULE DB2-671: EXCESSIVE CROSS-INVALIDATIONS DUE TO DIRECTORY RECLAIMS

   DSNDB1G_GBP4: More than 0.1% of the directory entry reclaims resulted
   in cross-invalidation action for members of the data sharing group.
   This could mean that the group buffer pool was too small or that the
   directory entry to data entry ratio was too low. The directory entry
   to data entry ratio was specified as 5:1

                              CROSS-INVALIDATED    DIRECTORY   PCT DIR RECL
   MEASUREMENT INTERVAL     (DIRECTORY RECLAIM)    RECLAIM     CAUSING XI
    9:30-10:00, 29AUG2000          1,636            30,759         5.3
```

Suggestion: If Rule DB2-671 is produced, you should consider the following alternatives (particularly if the percent of cross-invalidated pages shows an increasing trend):

- **Increase the directory entry/data entry ratio**. Excessive cross-invalidations due to directory reclaims indicate that there are too few directory entries. The number of directory entries can be increased by increasing the directory entry to data entry ratio. Implementing this option would reduce the number of data entries, but changing the directory entry to data entry ratio is relatively easy to implement. A small increase in the directory entry to page entry ratio could result in a large increase in the number of directory entries, with only a small decrease in the number of data entries. This is because the directory entry is so small (about 200 bytes) relative to the data entry (4096 bytes or more, depending on the page size of the corresponding local buffer pool).

- **Increase the size of the group buffer pool**. Since the group buffer pool is divided into directory entries and data entries, increasing the size of the group buffer pool would provide more directory entries. This option is not likely to be satisfactory, however, unless you either (1) significantly increase the size of the group buffer pool, or (2) increase the directory entry to data entry ratio.

- **Reduce the data sharing level**. In some instances, conflicts between applications might be so great that it might be impossible to obtain satisfactory DB2 data sharing performance. In this case, it might be necessary to decrease the data sharing level associated with specific group buffer pools. To accomplish this, you could reassign page sets or partitions to different virtual buffer pools (with corresponding reassignment to different group buffer pools), create additional virtual buffer pools (with corresponding additional group buffer pools), or move some application between data sharing members and even to another processing time.

- You can alter CPExpert's analysis by modifying the **PCTXIGBP** guidance variable in USOURCE(DB2GUIDE).

Reference: DB2 for OS/390 Version 4: Data Sharing Planning and Administration
Section 5.7.7.7 (Problem: Storage Shortage in the group buffer pool)
Section 6.4.4.1 (Writing to the Group Buffer Pool)

DB2 for OS/390 Version 5: Data Sharing Planning and Administration
Section 6.7.7.7 (Problem: Storage Shortage in the group buffer pool)
Section 7.5.5.1 (Writing to the Group Buffer Pool)

©Copyright 1999, Computer Management Sciences, Inc.

DB2 for OS/390 Version 6: Data Sharing Planning and Administration
Section 6.7.7.7 (Problem: storage shortage in the group buffer pool)
Section 7.6.5.2 (Writing to the group buffer pool)

DB2 for MVS/ESA Version 4 Data Sharing Performance Topics Redbook (SG24-4611)

IBM Washington System Center Flash 9609.

Your turn:

This manual has described how to use the DB2 Component to analyze performance constraints with IBM's DB2.

We would appreciate receiving any comments you have regarding this document (style, content, clarity, etc.), or suggestions for improving the DB2 Component (ease of use, new rules, changes to rules, etc.). Please send your comments to:

Don Deese
Computer Management Sciences, Inc.
6076-D Franconia Road
Alexandria, VA 22310
www.cpexpert.com

Comments: