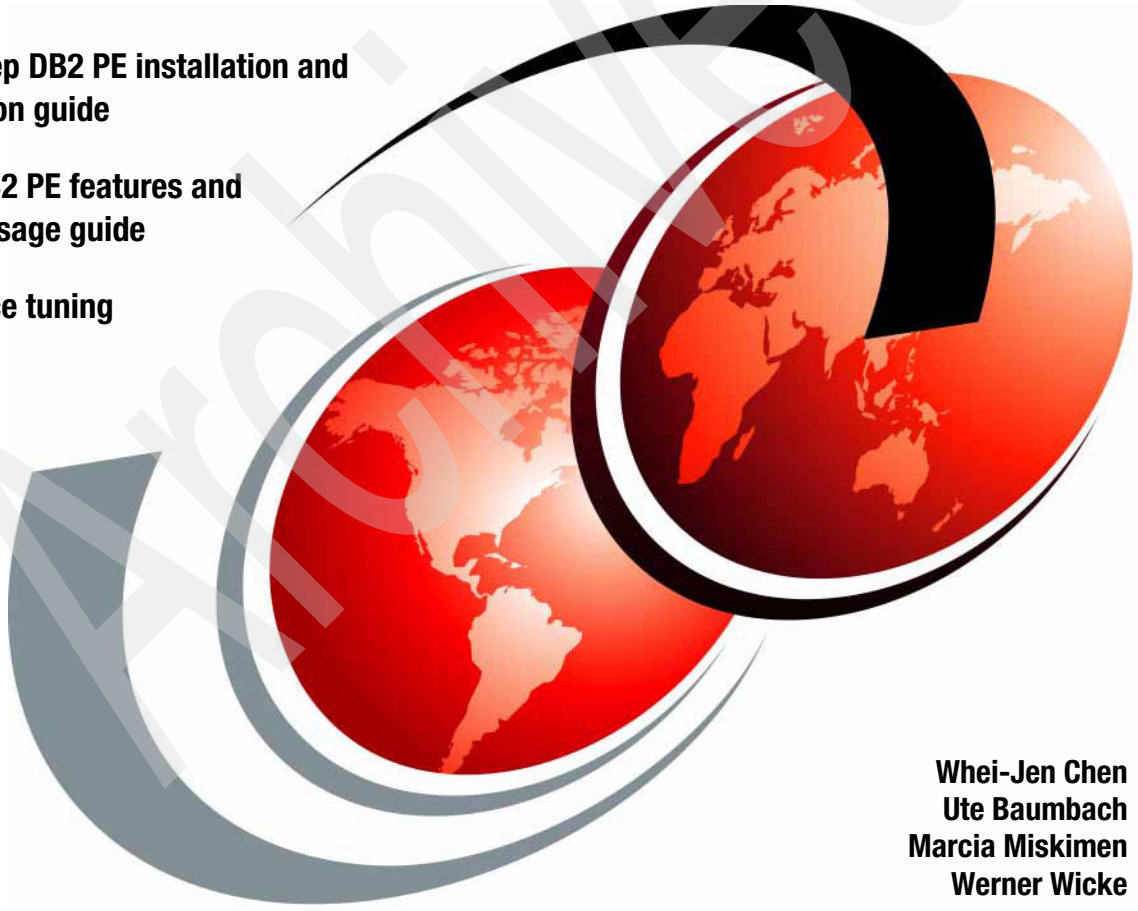
DB2 Information Management Software

IBM

DB2 Performance Expert for Multiplatforms V2.2

- Step-by-step DB2 PE installation and configuration guide
- Detailed DB2 PE features and functions usage guide
- Performance tuning scenarios



Whei-Jen Chen
Ute Baumbach
Marcia Miskimen
Werner Wicke

Redbooks

ibm.com/redbooks

International Technical Support Organization

DB2 Performance Expert for Multiplatforms V2.2

March 2006

SG24-6470-01

Note: Before using this information and the product it supports, read the information in “Notices” on page vii.

Second Edition (March 2006)

This edition applies to DB2 Performance Expert Version 2.1.1, 2.1.2, and Version 2.2 Fix Pack 1, DB2 UDB Version 8.1 and Version 8.2, AIX Version 5.0, Microsoft Windows 2000 Server, and Microsoft Windows 2000 Professional and XP.

Contents

Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:
IBM Director of Licensing, IBM Corporation, North Castle Drive Armonk, NY 10504-1785 U.S.A.

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:
This information contains sample application programs in source language, which illustrates programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. You may copy, modify, and distribute these sample programs in any form without payment to IBM for the purposes of developing, using, marketing, or distributing application programs conforming to IBM's application programming interfaces.

Trademarks

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

AIX 5L™	IBM®	Tivoli®
AIX®	ibm.com®	WebSphere®
DB2 Connect™	OS/390®	xSeries®
DB2 Universal Database™	pSeries®	z/OS®
DB2®	Redbooks™	zSeries®
DRDA®	Redbooks (logo)  ™	

The following terms are trademarks of other companies:

Java, JDBC, Solaris, Sun, and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Excel, Microsoft, Windows, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

Pentium, Intel logo, Intel Inside logo, and Intel Centrino logo are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Linux is a trademark of Linus Torvalds in the United States, other countries, or both.

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

Preface

This IBM® Redbook is an informative guide for installing, configuring, and using DB2® Performance Expert Version 2.2 Fix Pack 1. This redbook is organized as follows:

- Chapter 1 provides a general overview of DB2 Performance Expert Version 2.2 and a description of the Performance Expert architecture.
- Chapter 2 discusses the installation planning and topology. It provides a step-by-step guide for installing DB2 Performance Expert Server and Client.
- Chapter 3 discusses Performance Expert configuration and installation verification. It also shows the troubleshooting procedures for both PE Server and Client.
- Chapter 4 goes over DB2 Performance Expert online and short-term monitoring features and functions in detail. Along with the description of features and functions, DB2 performance tuning using Performance Expert is also discussed.
- Chapter 5 goes over DB2 Performance long-term monitoring Expert features and functions in detail. Along with the description of features and functions, DB2 performance tuning using Performance Expert is also discussed.
- Chapter 6 provides performance tuning scenarios with Performance Expert. The examples includes trend analysis for buffer pool utilization, lock escalation, and database connection, reported problem investigation, and performance verification in a test environment.
- Appendix A provides Rule of Thumb definition examples. These samples are valuable criteria in measuring the performance of applications in a DB2 system.
- Appendix B provides guidelines for using Performance Expert in different CM performance monitoring and analyzing tasks.
- Appendix C focuses on the typical and most important performance issues that occur in Business Intelligence (BI) environments and how DB2 Performance Expert (PE) can be used to analyze and fix them.

The team that wrote this redbook

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

Whei-Jen Chen is a Project Leader at the International Technical Support Organization, San Jose Center. She has extensive experience in application development, database design and modeling, and DB2 system administration. Whei-Jen is an IBM Certified Solutions Expert in Database Administration and Application Development as well as an IBM Certified IT Specialist.

Ute Baumbach has been a software developer at the IBM lab in Germany for 15 years, where she works in different software development projects and different roles. Most of her projects were based on DB2. for two and half years, she has worked as a member of the DB2 Performance Expert development team. In addition, Ute supports customers in setting up DB2 Performance Expert and utilizing it to its best advantage. She is an IBM Certified Database Administrator and a Certified Application Developer for DB2 UDB.

Marcia Miskimen is a Software Engineer in the USA at IBM Software Group's Silicon Valley Lab, where she is currently a Data Management Tools specialist. She has over 20 years of experience in the IT industry, including ten years as an IT Specialist in IBM Global Services. Her areas of expertise and interest include the application development life cycle, software testing, and tools of all sorts. She has a Bachelor of Science in Business Administration from Ohio State University.

Werner Wicke is a software engineer at IBM Lab in Boeblingen, Germany, where he has been working in different projects for almost 20 years, including product assurance for VSE and several development and project management tasks on z/OS® and multiplatforms. He has been a member of the DB2 Performance Expert development team for three and half years, where he is responsible for the design and implementation of major parts of the Performance Expert Server. He holds a Master's degree (diploma in Germany) in Computer Science for Medicine and has a certification as a Project Management Professional at the Project Management Institute.

Acknowledgements

The authors express their deep gratitude for the help received from Torsten Steinbach who contributed advice, support, and written content:

Torsten Steinbach works for IBM Germany since 1997. He is the architect of DB2 Performance Expert for Multiplatform. Before joining the Performance Expert team, he worked as an expert for DB2 performance and application development in joint projects with SAP and other IBM partners and for development of IBM WebSphere® Portal database access component.

We also thank the following people for their support and contributions to this project:

Steve Mink
John Bowen
Joel Farber
IBM Silicon Valley Laboratory

Surendra Parlapalli
Rao Chinnam
Randy Holmes
Ron Sherman
IBM USA

Michael Reichert
Birger Boyens
Oliver Draese
IBM Germany

Alice Ma
Andy Markovic
Rakesh Midha
Marcia Miskimen
Ken Siders
Kevin Taylor
Maria Weinerth
Authors of "DB2 Performance Expert for Multiplatforms V2"

Emma Jacobs
Sangam Racherla
International Technical Support Organization, San Jose Center

Become a published author

Join us for a two- to six-week residency program! Help write an IBM Redbook dealing with specific products or solutions, while getting hands-on experience with leading-edge technologies. You'll team with IBM technical professionals, Business Partners, and customers.

Your efforts will help increase product acceptance and customer satisfaction. As a bonus, you'll develop a network of contacts in IBM development labs, and increase your productivity and marketability.

Discover more about the residency program, browse the residency index, and apply online at:

ibm.com/redbooks/residencies.html

Comments welcome

Your comments are important to us!

We want our Redbooks™ to be as helpful as possible. Send us your comments about this or other Redbooks in one of the following ways:

- Use the online **Contact us** review redbook form found at:

  ibm.com/redbooks

- Send your comments in an e-mail to:

  redbook@us.ibm.com

- Mail your comments to:

  IBM Corporation, International Technical Support Organization
  Dept. QXXE Building 80-E2
  650 Harry Road
  San Jose, California 95120-6099

Summary of changes

This section describes the technical changes made in this edition of the book and in previous editions. This edition may also include minor corrections and editorial changes that are not identified.

Summary of Changes
for SG24-6470-01
for DB2 Performance Expert for Multiplatforms V2.2
as created or updated on March 22, 2006.

March 2006, Second Edition

This revision reflects the addition, deletion, or modification of new and changed information described below.

New information

- Performance Expert for Multiplatforms V2.2 and V2.2 Fix Pack 1 enhancements
- IBM Content Manager performance monitoring and tuning using Performance Expert
- Using Performance Expert to monitor and tune a business intelligent environment

1

Introduction

The IBM DB2 Performance Expert (PE) Version 2 for Multiplatform is a client/server performance analysis and tuning tool for managing a heterogeneous mix of DB2 systems using a single end-user interface. In this chapter, we provide a general overview of IBM DB2 Performance Expert for Multiplatforms and IBM DB2 Performance Expert for Workgroups.

The following topics are discussed in this chapter:

- Performance Expert overview
- Performance Expert architecture
- Availability of Performance Expert

1.1 Performance Expert overview

Performance monitoring can be a DBA's most challenging and critical task. In today's marketplace, data, information, and assets need to be available now. Your company's data, information, and assets are only valuable if your users can access them in a timely fashion. The performance of the database server therefore is critical to the success of the business.

IBM Performance Expert (PE) can be your DBA's most powerful tool to resolve your DB2 performance problems. PE can help streamline the diagnostic, analytical, and detection process to resolve performance problems in a timely fashion. The performance data PE provides to resolve performance problems is based on data from the DB2 snapshot monitor, DB2 event monitor, and the underlying operating system.

In this redbook, we cover IBM DB2 Performance Expert for Multiplatforms and IBM DB2 Performance Expert for Workgroups. The functions discussed in this redbook applies to both IBM DB2 Performance Expert for Multiplatforms and IBM DB2 Performance Expert for Workgroups work unless stated.

For information regarding Performance Expert for z/OS, please refer to *IBM DB2 Performance Expert for z/OS Version 2,* SG24-6867.

1.1.1 Performance monitor

PE provides four essential levels of performance monitoring:

- Online monitoring
- Short-term history monitoring
- Long-term history monitoring
- Exception processing

Online monitoring

Online monitoring is used to monitor the current operation of your DB2 system and the underlying operating system at a point in time when the DB2 instance is being monitored by the DBA sitting in front of PE. PE can help you gather current DB2 system information, current applications, and SQL workload, and since some DB2 performance problems are caused by bottlenecks in the underlying operating system, this information is gathered as well. PE online monitoring features and functions can help you detect problems such as long waits and timeouts, deadlocks, and long running SQL statements.

These features and functions provide the DBA with the ability to drill down to get more detailed information, such as set filters to isolate the problem, customize graphical health charts to visualize how activity and performance evolves over

time, trace SQL activities for a single application or the whole database, and view and analyze the trace to identify, for example, heavy hitter SQL statements that need further tuning.

Short-term history monitoring

Short-term history data can provide information to help a DBA look at specific events that occurred in a short interval of time. PE allows the user to configure the number of hours PE stores short-term history information.

Using PE short-term history monitoring mode can help a DBA diagnose deadlocks, long running SQL, timeouts, and lock escalations that happened minutes, hours, or days ago without the need to reproduce the problems, and monitor other aspects, such as UOW or buffer pool, tablespace, and file system usage.

Also, for short-term history data, the graphical health charts can be used to visualize performance metrices over time in history either to diagnose problems or identify trends.

For online and short-term monitoring, PE provides the users the ability to see detailed information for the following items:

- Application Summary/Details:
  - Times
  - Locking
  - SQL activity
  - SQL statements
  - Buffer pools
  - Caching
  - Sorting
  - Memory pools
  - Agents
- Statistic Details
  - Instance information
  - Database (usage, caches, high water marks, locks, reads, writes, and so on)
  - Tablespaces (Space management, read/write and I/O, containers and so on)
  - Tables

  - Buffer pool (read, write, I/O, and so on)
  - Memory pools
  - Dynamic SQL statement cache details
  - Utility Information
- Applications in Lock Conflicts/Locking Conflicts
- Locking Conflicts
- System Health: View DB2 performance information in a graphical format
- System Parameters - Instance
- System Parameters - Database
- Operating System Information
  - Memory and process configuration, processor status
  - File systems
  - Disks (Solaris™ only)
- Operating System Status
  - Memory and CPU usage
  - Running processes
  - Disk utilization

Long-term history monitoring

Long-term history data is collected over a period of time. The collected data is used for trend analysis. PE can help you collect trend analysis data that can be used to develop a performance baseline for your system. Using trend analysis data can also help you understand how your system will:

- React during normal and peak periods to help you set realistic performance goals.
- Resolve potential performance problems before they become an issue.
- Grow over a period of time.

PE provides long-term monitoring capability under the following functions:

- Performance Warehouse and Rules of Thumb

  PE includes Performance Warehouse, which allows you to quickly and easily identify potential performance problems. Performance Warehouse collects performance data for SQL, database, buffer pool activity, and the operating system. This performance data is used for generating reports. These reports can be used for further investigation and trend analysis. Performance

Warehouse data can also be used for Rules of Thumb (RoT), which is included in Performance Warehouse.

RoT can help a DBA by being proactive in making suggestions on how to improve performance. Performance Warehouse provides RoT queries for SQL, database, tablespace, and buffer pool activity.

- Buffer Pool Analysis

  Buffer pools are one of the most important aspects for tuning. PE Buffer Pool Analysis gathers detailed information regarding current buffer pool activity using snapshots. Buffer Pool Analysis allows the database administrator to view buffer pool information in a variety of formats, including tables, pie charts, and diagrams. Providing these different formats to view buffer pool information will enable the database administrator to quick identify potential problems and do trend analysis.

Exception processing

Exception process monitoring is another PE feature that allows DBAs to monitor a database server proactively. DBAs can use the exception processing function to activate predefined alert sets for OLTP or BI workloads or to configure their own alerts both to notify them when a particular situation has occurred. PE provides two types of alerts: *deadlock* and *periodic*. The alert message can be sent to specified e-mail addresses or a user exit can be called that allows you to exchange the alert message and details with other applications or to execute actions. Additionally, signals on the PE client indicate the occurrence of an exception together with drill down options.

1.2 PE architecture

PE's primary design goal was to provide an "end to end" solution for performance monitoring of DB2 systems. PE for Multiplatforms has three parts, as shown in Figure 1-1 on page 6:

- PE Server
- PE Client
- PE Agent

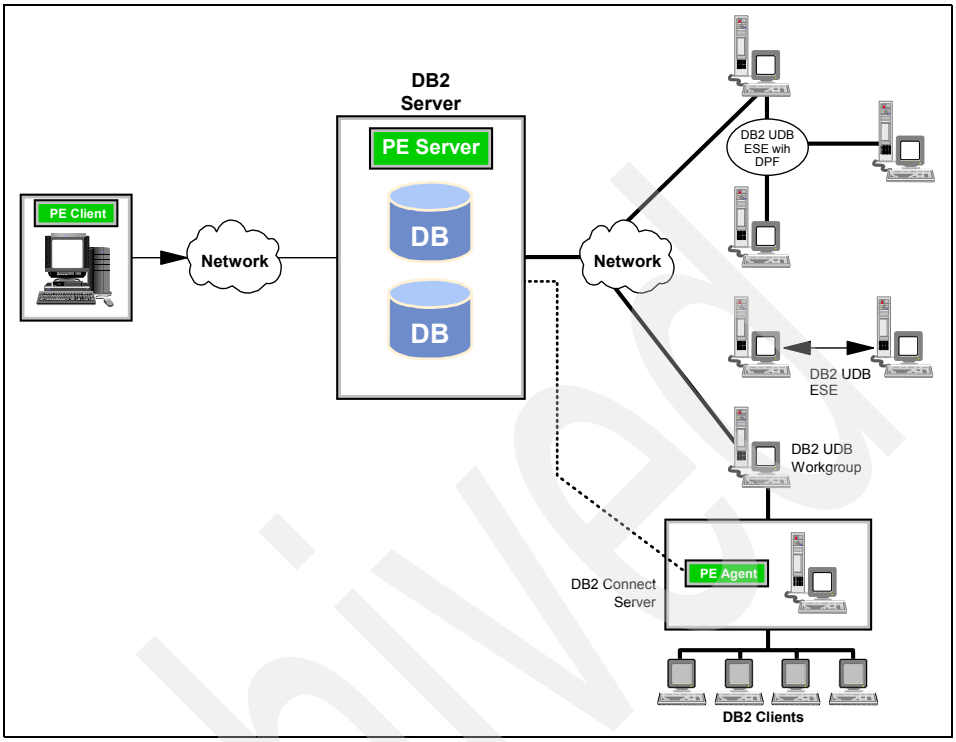


Figure 1-1 PE architecture

PE Server

The PE Server collects and stores the performance data of the monitored DB2 instance. In a multiplatform environment, one PE Server can monitor multiple DB2 instances in the network.

Figure 1-1 shows that one PE Server is installed on its own DB2 instance and remotely monitors the other DB2 instances. However, the PE Server can be installed on the same DB2 instance as the DB2 instance you wish to monitor.

The PE Server supports monitoring of DB2 UDB Enterprise Server Edition (ESE), Data Partitioning Feature (DPF), and Workgroup Edition. The PE Server for Multiplatforms comes in two flavors:

- DB2 Performance Expert for Multiplatforms
- DB2 Performance Expert for Workgroups

For information regarding the supported DB2 environments, please refer to the IBM PE manual *IBM DB2 Performance Expert for Multiplatforms IBM DB2 Performance Expert for Workgroups Installation and Configuration*, SC18-9191.

The PE Server is architected and designed to store all of the monitored DB2 instance information in the PE database. As shown in Figure 1-2, PE Server has two types of databases: master database and performance database.

- Master database

  The master database is used to store the PE Server metadata and configuration information. PE creates one master database called DB2PM during configuration time. There is only one DB2PM database per PE Server.

- Performance database

  Performance data of the monitored instance is stored in a database on the PE Server instance. For each monitored instance, PE Server creates one performance database. In Figure 1-2, you see three DB2 instances that the PE Server is monitoring. Each monitored instance has a performance database on the PE Server instance.

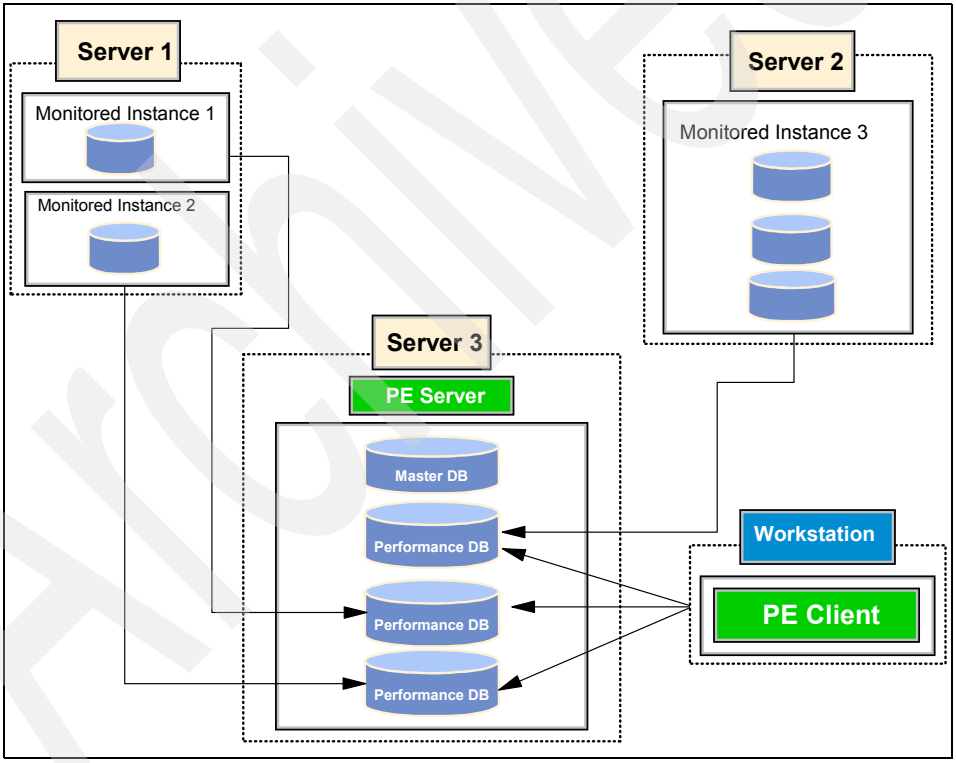


Figure 1-2 PE Server

The PE Server uses DB2 Snapshot and Event Monitors to collect DB2 performance data for the online monitoring, short-term history, long-term history,

and exception processing. To reduce overhead on the monitored DB2 instance, PE Server uses Snapshot instead of Event Monitoring whenever possible.

The PE Server uses *Common Information Model Object Manager* (*CIMOM*) to additionally collect operating system performance data for the online monitoring, short-term history, long-term history, and exception processing. Collecting operating system performance data is currently limited to monitored instances residing on AIX®, Solaris, or Linux®.

CIM is a common information model for describing management properties that are not bound to a particular implementation. This allows the interchange of management information between management systems and applications through CIMOM. CIMOM is an object management engine that exists between the managed system and the management application.

CIMOM is delivered as operating system extensions. On AIX, CIMOM is part of the *AIX Expansion Pack*, on Solaris, CIMOM is part of the *WBEM Services,* and on Linux, CIMOM is downloadable from the Web.

Figure 1-3 shows how CIMON is used with PE. The CIM Client, which is included in PE Server, requests the CIM Server through TCP/IP to collect and return operating system data of the AIX or Solaris system on which the monitored instance runs. The collected operating system data and the DB2 performance data is stored in a performance database.

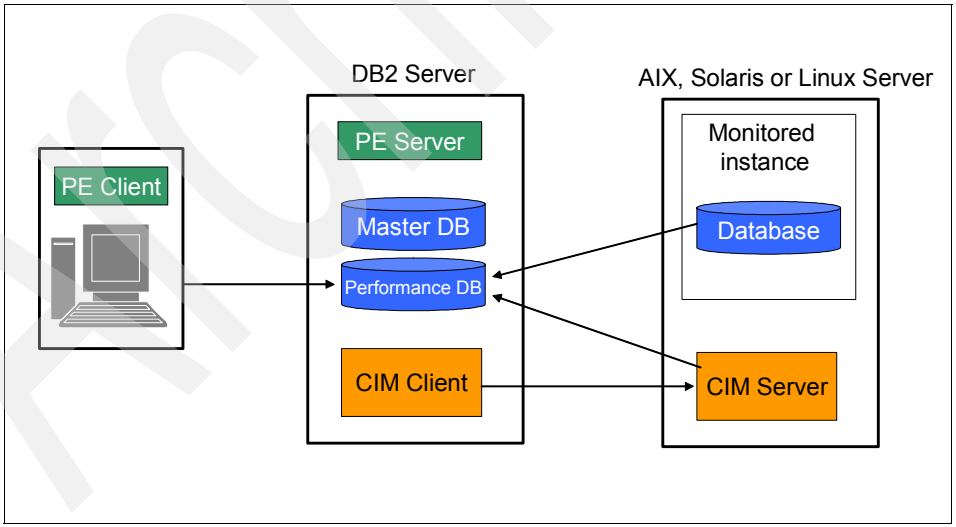


Figure 1-3 PE and CIMOM

PE Agent

The PE Agent is used to monitor the Database Connection Services (DCS) applications (also known as the DB2 gateway). Information collected by the PE Agent is stored on a database located on the PE Server. The PE Server can either be installed on any supported platform (AIX, Sun™ Solaris, HP-UX, Linux, Linux on z/Os, Windows®, or z/Os). Information can be accessed from the PE Client. As shown in Figure 1-1 on page 6, the PE Client communicates with the PE Server directly to obtain this information. The Client does not communicate with the PE Agent directly. For more information about Performance Agent, please refer to *IBM DB2 Performance Expert for z/OS Version 2,* SG24-6867.

PE Client

PE Client provides a graphical front-end interface to view the performance data collected by the PE Server and PE Agent data. The PE Client is used to configure the PE Server tracing, report PE Server history data, set up exception monitoring, set up buffer pool analyzer reports, set up performance warehouse reports, configure the Rules of Thumb, and more.

Prior to PE V2.1.1, the PE Client only communicates with the PE Server. It never communicates directly with the monitored DB2 instances. On PE V2.1.2, the Visual Explain launch was introduced on the PE Client. A catalog entry of the monitored database on the PE Client machine is required for using Visual Explain. The PE Client uses JDBC™ to communicate with the PE Server.

1.3 Ordering information

PE can be purchased in the following ways:

- IBM Web Site

  You can purchase PE on the IBM Web site:

  `http://www.ibm.com/software/data/db2imstools/db2tools/db2pe/index.html`

- Through e-mail

  The IBM Web site to send us e-mail regarding purchasing PE is:

  `https://www6.software.ibm.com/reg/swgmail/swgmail-r?S_TACT=1O4CBW68`

- Over the phone

  The IBM Phone number to order PE is:

  1-877-426-3774

- From an IBM Sales Representative

  PE can be purchased from your IBM Sales Account Representative. If you do not have an IBM Sales Account Representative yet, you can contact IBM by going to the above Web site, then selecting the **Contact IBM** button.

2

Planning and installation

A successful DB2 Performance Expert implementation includes planning, installation, and configuration. This chapter describes the steps necessary for a successful DB2 Performance Expert installation. We discuss planning considerations, sizing guidelines, and suggestions for placement of a PE Server. Installation steps for both server and client are described in detail.

The information contained in this chapter is meant to complement the material found in the DB2 Performance Expert for Multiplatforms product documentation, *IBM DB2 Performance Expert for Multiplatforms IBM DB2 Performance Expert for Workgroups Installation and Configuration,* SC18-9191.

This chapter discusses the following:

- Planning: What do you need to know before you start the install. This effort requires input from both the system administration and DBA side of the house.
- Installation: PE Server on AIX and Windows, the PE Client on Windows. This is performed by the system administrator.

2.1 Planning

It is important that you understand what information is required before you begin the installation of the components of DB2 Performance Expert. In the following discussion, we touch upon the hardware requirements, supported software, and configuration information required to smoothly install and configure DB2 Performance Expert.

2.1.1 Information gathering

It is wise to review the prerequisite hardware and software requirements before you start to install DB2 Performance Expert (PE).

Hardware capacity

This discussion is a summary of current multiple sources and is intended to clarify what hardware resources are required and why. Additional discussion can be found in Appendix E, “Space requirements for Performance Expert Server”, in *IBM DB2 Performance Expert for Multiplatforms IBM DB2 Performance Expert for Workgroups Installation and Configuration*, SC18-9191.

The following list pertains to the PE Server requirements.

- Temporary space

  InstallShield needs about 100 MB as temporary space. The temporary space on UNIX® systems is in the home directory of the root user. The temporary directory can be changed by the command-line switch `-is:tempdir <temporary directory>` when starting the installation. For the Windows platform, the file system defined by the TEMP environment variable is used.

- PE Server program files

  The UNIX PE Server program files will require about 25 MB in the install directory. An additional amount of about 52 MB is required for each Java™ runtime library (32-bit and 64-bit). Table 2-1 shows which Java runtime libraries are installed depending on the platform.

Table 2-1 Summary of Java runtime library installations

Platform	32-bit Java runtime	64-bit Java runtime	Notes
AIX	x	(x)	64-bit runtime only if platform supports 64-bit.
Linux on xSeries®	x	(x)	64-bit runtime only if platform supports 64-bit.

Platform	32-bit Java runtime	64-bit Java runtime	Notes
Linux on pSeries®		x	Only 64-bit is supported.
Linux on zSeries®	x		Only 31-bit is supported.
Solaris	x	x	Both are included.
HP/UX	x	x	Both are included.

For the PE Server on the Windows platform, 72 MB is required.

- DB2PM master database

  The PE Server has a single database named DB2PM for its metadata. The size of the DB2PM master database does not grow much. We recommend you initially allocate 25 MB for this database. The DB2PM master database only contains the configuration data, that is, the data for the monitored instances and their databases. If you find you need to add file system space for this database to accommodate adding more monitored instances, for each instance, add approximately 4 KB, and for each database to be monitored, add an additional 2 KB.

- PE performance database

  Each monitored DB2 instance has a performance database that stores history and Performance Warehouse data for the databases in that instance. This performance database is owned by the Performance Expert Server, and resides in the same instance as the PE Server and grows over time. The rate of growth depends on a number of factors. Refer to Appendix E, “Space requirements for Performance Expert Server”, in *IBM DB2 Performance Expert for Multiplatforms IBM DB2 Performance Expert for Workgroups Installation and Configuration*, SC18-9191 for a technical discussion of what influences the growth of the performance database. One recommendation is that you initially allocate 1.5 GB. The space needed for Performance database heavily depends on the workload and monitored objects. Calculations should be done before installing PE using Chapter 20, “Space requirements for Performance Expert Server”, *IBM DB2 Performance Expert for Multiplatforms IBM DB2 Performance Expert for Workgroups Installation and Configuration*, SC18-9191. Another recommendation is that if you wish to add another monitored instance, ensure there is initially about 40 MB available for each instance to be added. We believe this is also a minimum value and any real PE activity for the newly configured monitored database will require additional space well beyond the suggested 40 MB.

- DB2 transaction log space

  We recommend the transaction log space for each PE performance database be initially allocated to 100 MB. The PE performance database runs in circular logging. The log directory is initially set to the default, but can be changed in the database configuration using normal database commands.

- Exchange of Event Monitor data

  In the past, if you wanted to use the data from DB2 deadlock and statement event monitors, it was a requirement to have a shared file system between the monitored database and the PE Server. In some shops, shared file systems conflict with database or network security. In Performance Expert V2 Fix Pack 2, the requirement for a shared file system was lifted. You must decide which approach to take before you configure the PE Server, but you can later change this data exchange structure if you wish. Refer to Chapter 4, “Configuring Performance Expert Server on Windows systems” and Chapter 10, “Configuring Performance Expert Server on UNIX and Linux systems” in *IBM DB2 Performance Expert for Multiplatforms IBM DB2 Performance Expert for Workgroups Installation and Configuration*, SC18-9191 for the Windows PE Server configuration and the UNIX PE Server configuration, respectively.

  – Shared File system

    We recommend you start with 300 MB of file system space for each monitored database. As you identify what data you wish to monitor, your real file system needs will become more apparent. Before you start to configure the PE Server, you need to define the access path from the PE Server system to the shared file system and the access path from the monitored database to the shared file system. For the Windows PE Server, the shared file system must reside upon the same machine as the monitored database and you must explicitly enable each of the users that need access to the shared file system. For UNIX, **`CHMOD 777`** will be used to grant shared file system permissions.

  – No Shared File system

    If you decide a shared file system will not work for your system, you can use database objects to collect event monitor data. Before you configure the PE Server, you need to have identified the two directories that represent the local and remote directories. The local is the directory that you will create on the PE Server and the remote directory is the path you will create on the monitored database. You should consider allocating 300 MB for the local directory.

- Working directory

  When running the **pecentralize** utility, it refers to the DB2PE working directory. This is used for PE Server application logs, traces, and some DDL files created at startup. As a rough guideline, a maximum of 18 MB is needed for the PE Server and 16 MB is needed for each monitored instance. The default path for the UNIX systems is <user home directory>/db2pe/v22/<instance name>, and you are allowed to change this path. Although there is no **pecentralize** command for Windows, the default path for the working directory on Windows is C:\Program Files\IBM\DB2 Performance Expert Server v2\instances\<instance name>. For the Windows PE Server, you cannot change the path of the PE working directory.

- Memory

  We recommend you start with 60 MB of memory for the UNIX PE Server and 100 MB of memory for the Windows PE Server. Add 40 MB of memory for each additional monitored instance. Please note that since the PE Server must run under DB2, you should also make sure that the PE Server system has main memory that meets the memory retirement specified in the DB2 documentation for the appropriate DB2 version.

- Processor

  A 400 MHz processor clock is the minimum recommended for the Windows PE Server. For the UNIX PE Server, no specific CPU clock is stated.

Table 2-2 is a summary of the PE Server space requirements.

Table 2-2 Summary of minimum PE Server disk space requirements

	AIX	Linux on xSeries	Linux on pSeries	Linux of zSeries	Solaris	HP/UX	Windows
Install temporary space	100 MB	100 MB	100 MB	100 MB	100 MB	100 MB	100 MB
PE Server program file	78 MB (32-bit), 130 MB (64-bit)	78 MB (32-bit), 130 MB (64-bit)	78 MB	78 MB	130 MB	130 MB	72 MB
PE DB2PM database	25 MB	25 MB	25 MB	25 MB	25 MB	25 MB	25 MB

The space requirements for some of the PE Server items, such as performance database or event monitor files, heavily depend on the workload and monitored objects. Table 2-3 shows the starting disk space size recommended for these items. The numbers apply to all platforms. Calculations should be done before installing PE using Chapter 20, “Space requirements for Performance Expert Server”, *IBM DB2 Performance Expert for Multiplatforms IBM DB2 Performance Expert for Workgroups Installation and Configuration*, SC18-9191.

Table 2-3 Summary of recommended starting disk space size for PE Server

	Recommended disk space
Performance database for each monitored instance	1.5 GB
Shared file system /no shared file system	300 MB
Performance DB2 transaction log space	100 MB
Working directory	34 MB

The following discussion pertains to the Windows PE Client.

- PE Client program files

  The PE Client for Windows will use 110 MB of disk space. The default install path is C:\Program Files\IBM\IBM DB2 Performance Expert V2.

- PE Client work area and file storage

  There is an area on the file system that the PE Client uses for storing various files and as a work area. The default path is C:\Documents and Settings\Administrator\.db2pev2\. Such files as Buffer Pool Analysis reports, Performance Warehouse outputs, PE Client screen settings, and work files will be stored there. This space can become very large and there is no automatic purge of old material. It is important to note that this area also exists on the UNIX PE Client in the ~home/.db2pev2 directory. Care should be taken to monitor and manage this area.

- PE Client memory

  We recommend that you allow 70 MB of memory for the PE Client’s needs. We suspect this will work as a minimum. But if you intend to really use the PE Client, consider having at least 512 MB memory.

- Console

  The PE Client GUI is displayed better if the console resolution is 1024X768 or higher. The higher resolution will facilitate viewing the large amount of data the PE Client can display.

- Processor

  A Pentium® with a clock rate of 400 MHz is considered the minimum needed.

Software product levels

It is important that you understand the supported software requirements. *IBM DB2 Performance Expert for Multiplatforms IBM DB2 Performance Expert for Workgroups Installation and Configuration*, SC18-9191 has lists of the minimum software located in each chapter that discusses the installation of PE components. We suggest that you first review *IBM DB2 Performance Expert for Multiplatforms IBM DB2 Performance Expert for Workgroups Installation and Configuration*, SC18-9191 and then the product readme for more current requirements. Rather than list everything that is already documented, the following is a summary of the software requirements and is based upon PE V2 Release 2:

- Operating Systems
  - Windows 2000 Service Pack 2, XP, or 2003
  - Minimum UNIX operating systems
    - AIX 5L™ Version 5.1
    - HP-UX Version 11i
    - Linux on zSeries Kernel Version 2.4.19
    - Linux on xSeries Kernel Version 2.4.19
    - Linux on pSeries Kernel Version 2.4.19
    - Solaris Version 8

  For a detailed list of supported Linux distributions, please see Chapter 9, “Installing Performance Expert Server on UNIX and Linux systems”, in *IBM DB2 Performance Expert for Multiplatforms IBM DB2 Performance Expert for Workgroups Installation and Configuration*, SC18-9191.

- Database

  DB2 UDB Enterprise or Workgroup Edition Version 8 Fix Pack 6b is the minimum level supported. Refer to *IBM DB2 Performance Expert for Multiplatforms IBM DB2 Performance Expert for Workgroups Installation and Configuration*, SC18-9191 for more specific details. Be advised that PE does not support all database editions on all operating systems.

- Network

  TCP/IP is expected to be installed.

- Java Runtime (JRE)

  Java is shipped and installed with the PE code. JRE Version 1.4.2 is needed for PE runtime. The PE installer itself needs a Java 1.4 to start installation. If this is not already available, the shipped IBM JRE V1.4.2 can be extracted from the installation disks and used to start the installation. Chapter 9, “Installing Performance Expert Server on UNIX and Linux systems”, in *IBM DB2 Performance Expert for Multiplatforms IBM DB2 Performance Expert for Workgroups Installation and Configuration*, SC18-9191 describes the installation.

- Requirements to integrate operating system data in Performance Expert

  This function is only supported for monitored DB2 instances running on AIX 5.2 ML03 or later, on Solaris 9 or later, or one of the Linux versions mentioned above.

  – On AIX, Pegasus Version 2.3.2 or later, which is part of AIX Expansion Pack 5.2.6 or AIX Expansion Pack 5.3.2, must be installed and running.

  – On Solaris, WBEM Services 2.5 of Solaris 9 or later must be installed and running. Additionally, the minimum level of DB2 for the Performance Expert Server is DB2 UDB Enterprise Server or Workgroup Server Edition Version 8.2 with Fix Pack 8.

  – On Linux platforms, only OpenPegasus 2.5 Common Information Model (CIM) server, or higher, must be installed.

- Additional PE Client requirements

  If the PE Client is not on the same machine as the PE Server, we recommend that you use the DB2 Administration Client to install the PE Client. The PE Client also requires a Web browser.

Users, groups, and environmental considerations

There are several decisions you must make before you begin the install of Performance Expert.

- Will you use the default install directory? If not, be prepared to identify that path in which you want the PE code to be installed.
- If this is a Windows PE Server install, what will be the PE database administrator name? For the Windows PE Server, you can use either the default DBA ID or an ID defined for this purpose. This ID will configure and start the PE Server.
- What DB2 Group name will you use for the PE instance? Windows DB2 has a default group, but you can create another one if you like. For UNIX, you are directed to create a group. Note that the group name cannot exceed eight characters and contain blanks. If you have DB2 V8.2 with PE V2 Fix Pack 2, the group name can be up to 30 characters.

- You should make a list of the users that you want to give initial access to the PE Server through the PE Client. This list of users and the IDs that run PE Server should be added to the group.
- If this is a UNIX PE Server, what name will you assign the PE DB2 instance owner? This ID will configure and start the PE Server.
- What will you use for the name of this DB2 fenced ID for PE Server instance?
- A PE DB2 instance port number and name are required for both Windows and UNIX PE Servers. Typically, the PE configuration utility updates the Windows etc\services file, but Chapter 4, “Configuring Performance Expert Server on Windows systems”, in *IBM DB2 Performance Expert for Multiplatforms IBM DB2 Performance Expert for Workgroups Installation and Configuration*, SC18-9191 directs you to verify that the entry exists. Chapter 9, “Installing Performance Expert Server on UNIX and Linux systems”, in *IBM DB2 Performance Expert for Multiplatforms IBM DB2 Performance Expert for Workgroups Installation and Configuration*, SC18-9191 instructs you to edit the UNIX etc/services file with the port number and a port name, but if you wish, you can use one of the ports that are automatically created when the instance is created.
- Will you use the default PE working directory, that is, the home directory of the instance owner? During the UNIX install, you are offered a chance to change this path. This is not offered for the Windows PE Server install.
- Do you have Java 1.4 installed on your system and do you know what the path is? The installation may fail because it does not find a sufficient Java version. Don’t worry, if you don’t know, or if you don’t have Java 1.4 installed. If you know, you can use the installation command-line option “*-is:javahome <Java directory>*” to specify the path. If you don’t have a Java 1.4 installed, you can extract the installation files of IBM JRE Version 1.4.2 from the installation disks. This is described in detail in *IBM DB2 Performance Expert for Multiplatforms IBM DB2 Performance Expert for Workgroups Installation and Configuration*, SC18-9191, Appendix M. Troubleshooting.
- You must decide if you will use a shared file system or database objects for the exchange of event monitor data.

2.1.2 Topology

Performance Expert can be implemented in several ways. There are three basic configurations. You must decide which of the three you intend to use before you start installing DB2 PE. We discuss these three configurations and their strenghts and weaknesses. This discussion is intended to augment the existing material in Appendix B, “Scenarios for the installation and configuration of Performance Expert Server”, *IBM DB2 Performance Expert for Multiplatforms IBM DB2 Performance Expert for Workgroups Installation and Configuration*, SC18-9191.

PE Server and the monitored DB2 instance are on the same instance

Why would you want to use this approach?

- Hardware requirements are minimal.
- The event monitor data exchange is made simple by this approach.
- It is good for learning, testing, demonstrations, and for a starter system where system resources are not a concern.

Why would you NOT want to use this approach?

- The PE Server and monitored database will compete for the same resources.
- You would not want to use this for a production system.
- The performance of PE as an application cannot be used for setting expectations on how it will perform in your production system.
- If you have 32-bit DB2 on AIX, PE Server will want EXTSHM turned on. Can the other database(s) in the same instance allow it?

PE Server and the monitored DB2 instance on the same system but in separate instances

Why would you want to use this approach?

- This would make a good learning, test, or development system.
- For a small production system, this setup would function.
- Because you do not want to share DB2 resources between the PE Server instance and the monitored database instance.
- If you have 32-bit DB2 on AIX, PE can set EXTSHM on without affecting the other instance.
- The monitored data exchange file system is local to both PE Server and the monitored database.

- From this initial setup, you can add other remote database instances to be monitored.
- The performance of PE as an application will be closer to what you can expect in your production environment.
- This configuration is the minimum we recommended.

Why would you NOT want to use this approach?

- Although database resources are not shared within the instance, there will still be competition for system resources.
- If you continue to add instances to monitor, eventually the other DB2 instances on the same system will begin to feel the competition for resources.

PE Server and the monitored DB2 instance on different machines

Why would you want to use this approach?

- You can add monitored databases without being concerned with impacting other applications as the resource demands increase for the PE Server.
- You have many remote instances you wish to monitor.
- The PE Server databases do not compete for resources with another database.
- The PE Server can be running on a smaller system than the production system.
- If you have 32 bit DB2 on AIX, PE Server can set EXTSHM on.

Why would you NOT want to use this approach?

- DB2 monitored data exchange is required if you wish to collect event data.
- Another DB2 license may be required for the PE database.
- A separate machine is required for the PE Server.

2.2 Installation

With the installation decisions made and the information gathered, the next step is to install the Performance Expert components. This discussion is intended to be used with *IBM DB2 Performance Expert for Multiplatforms IBM DB2 Performance Expert for Workgroups Installation and Configuration*, SC18-9191 and the product readme.

In the following scenarios, we remind you of what you need to know and briefly show the steps required to install the PE Server on AIX and Windows and the PE Client on Windows. This discussion is intended to augment the product documentation. Please refer to the product documentation as you read this section.

2.2.1 Lab environment

Figure 2-1 on page 22 shows the systems used to demonstrate the installation and configuration for PE Server and Client in this chapter.

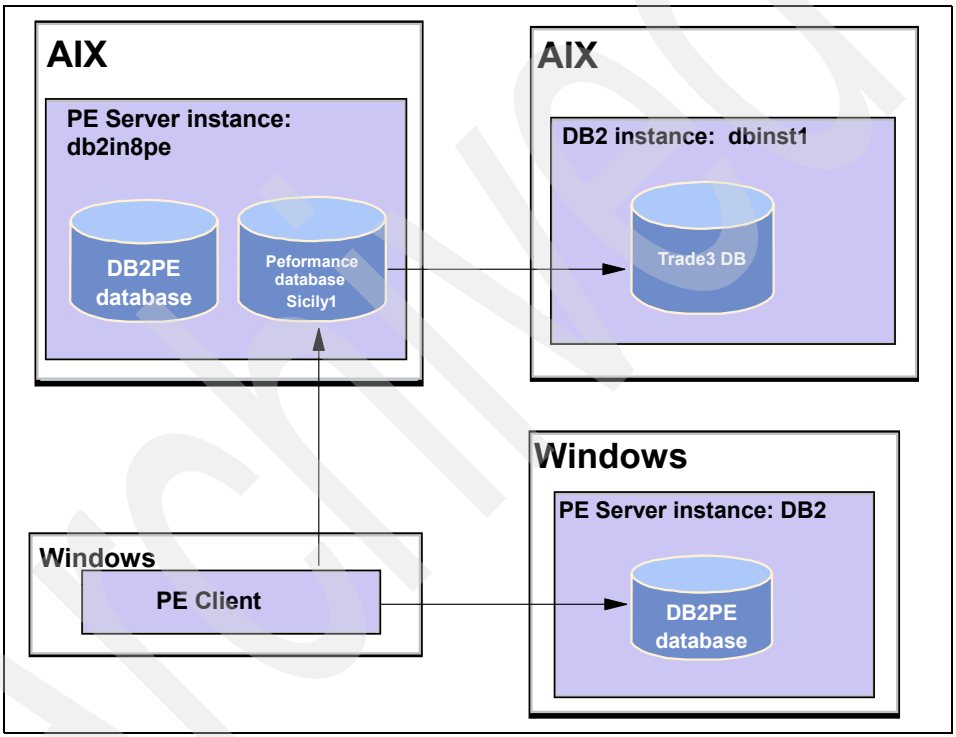


Figure 2-1 Installation laboratory environment

PE Server

For the following PE Server for AIX install process, we are using telnet from a Windows platform to a Model P609 running AIX 5L V5.2, DB2 V8 FP10, and Java 1.4, with a host name of jamacia, 16 processors, 4 TB of disk storage, and 1208 GB of memory.

For the PE Server install on Windows, we are running Windows 2000 and DB2 V8 Fix Pack 9 on a NetVista with 40 GB of disk space and 1 GB of memory.

PE Client

For this example, PE Windows Client install, we are running Windows 2000 and DB2 V8 Fix Pack 9 on a NetVista with 40 GB of disk space and 1 GB of memory.

Monitored DB2 server

The monitored database is Trade3, which is located on a Model F80 running AIX 5L V5.2, DB2 V8 Fix Pack 6, and Java 1.4, with a host name of sicily, 4 processors, 1 GB of memory, and two 9 GB drives.

Table 2-4 shows the value used for the PE installation example.

Table 2-4 Installation information for installation example

Description	PE AIX Server	PE Windows Server
Install Directory	default	default
Group name	db2pe	db2admin
DBA/Instance Owner	db2in8pe	markovic
Instance name	db2in8pe	DB2
Fenced Instance Owner	db2fn8pe	N/A
Instance Port Number	60004	default
PE Working Directory	default	default
JRE Path	/usr/java142_64	N/A
EXTSHM can be turned on?	Yes	N/A

2.2.2 Installing Performance Expert Server on AIX

We present in this section what you should expect while performing a successful install of the PE AIX server. The install steps assume that the client's shop requires an AIX system administrator to perform the root file system install and the DBA to perform the PE configuration. We prefer the console mode for installing the server on AIX.

Using the *root* user ID, we copy PE Server GA code to a temporary directory. If you use an install CD, insert the CD-ROM and mount the drive. Before invoking the installer, make sure that the umask is set to 022 by entering the command. A wrong umask (like 077) will result in wrong permissions of the installed files. As root, enter the command **`./db2pe.server.v2.2.install-on-aix -console`** to start the installation process.

Figure 2-2 on page 24 shows the InstallShield initializing. When prompted, we enter 1 to select Next.

```
# ./db2pe.server.v2.2.install-on-aix -console



             Initializing InstallShield Wizard........

             Launching InstallShield Wizard........


------------------------------------------------------------------------------
IBM DB2 Performance Expert Server V2 Setup

Welcome to the InstallShield Wizard for IBM DB2 Performance Expert Server V2

The InstallShield Wizard will install IBM DB2 Performance Expert Server V2
on your computer.
To continue, choose Next.

DB2 Performance Expert Server V2
IBM
http://www.ibm.com

Press 1 for Next, 3 to Cancel or 5 to Redisplay [1] 1
```

Figure 2-2 Install Shield initialization

In Figure 2-3 on page 25, the license agreement is displayed. The screen first shows that the default 'I do not accept....' is selected. Once you read the agreement and agree to the terms, select the 'I accept...' option by entering 1. Your choice is then displayed. Enter 0 to signal that you have made a choice. You are given a chance to back up and change your answer. We enter 1 for Next to continue. The install now starts writing files to the file system.

```
Please read the following license agreement carefully.
.... The licence terms and agreements are displayed.......
Please choose from the following options:

[ ] 1 - I accept the terms of the license agreement.
[X] 2 - I do not accept the terms of the license agreement.

To select an item enter its number, or 0 when you are finished: [0] 1

[X] 1 - I accept the terms of the license agreement.
[ ] 2 - I do not accept the terms of the license agreement.

To select an item enter its number, or 0 when you are finished: [0] 0

Press 1 for Next, 2 for Previous, 3 to Cancel or 4 to Redisplay [1] 1
```

Figure 2-3 License agreement

Figure 2-4 gives you the chance to override the default PE Server install directory. We accept the default by entering 1 for Next.

```
IBM DB2 Performance Expert Server V2 Setup

IBM DB2 Performance Expert Server V2 Installation Location

Please specify a directory or press Enter to accept the default directory.

Destination Directory [/opt/IBM/db2pesrv/V2.2]

Press 1 for Next, 2 for Previous, 3 to Cancel or 5 to Redisplay [1] 1
```

Figure 2-4 Install directory confirmation

Figure 2-5 on page 26 shows the verification of the size of the file system required and gives you another chance to change the install directory. We enter 1 for Next.

```
IBM DB2 Performance Expert Server V2 Setup

IBM DB2 Performance Expert Server V2 will be installed in the following
location:

/opt/IBM/db2pesrv/V2.2

for a total size:

 130.7 MB

Press 1 for Next, 2 for Previous, 3 to Cancel or 4 to Redisplay [1] 1
```

Figure 2-5 Install directory confirmation

Figure 2-6 presents the status of the files being written to the file system followed by the final option to finish. You should select 3 to Finish. If you encountered errors, you should see the details on the screen; but there is a log file db2pesrv.log in the directory given by the installation process that may contain additional data.

```
IBM DB2 Performance Expert Server V2 Setup

Installing DB2 Performance Expert Server V2. Please wait...

|-----------|-----------|-----------|------------|
0%         25%         50%         75%        100%
||||||||||||||||||||||||||||||||||||||||||||||||||

Updating the inventory ...

Creating uninstaller...

-------------------------------------------------------------------------------
IBM DB2 Performance Expert Server V2 Setup

The InstallShield Wizard has successfully installed IBM DB2 Performance
Expert Server V2. Choose Finish to exit the wizard.

Look at the log file /var/adm/sw/db2pesrv.log for details.

Press 3 to Finish or 5 to Redisplay [3] 3
```

Figure 2-6 File installation progress

As *IBM DB2 Performance Expert for Multiplatforms IBM DB2 Performance Expert for Workgroups Installation and Configuration*, SC18-9191 states, the system administrator, as *root*, will perform the following actions. You must perform these tasks before going further. These steps are required for the **pecentralize** utility to function. If your intent is to put the PE Server in a separate instance, you will want to perform all of the following steps. Refer to Chapter 9, “Installing Performance Expert Server on UNIX and Linux systems”, in *IBM DB2 Performance Expert for Multiplatforms IBM DB2 Performance Expert for Workgroups Installation and Configuration*, SC18-9191 for more details.

- Create a user group and add the PE users to this group. We named our group db2pe.
- Create a DB2 instance owner user. The instance owner we created is db2in8pe.
- Create a DB2 instance owner fenced user. We created db2fn8pe.
- Create an instance for the PE Server database. Our instance is named db2in8pe.
- Edit the etc/services file to add the port name and number. While you can do this, we elected to use one of the default ports created with the instance. How you decide to do this will be your choice.

With the underlying structure ready, we are ready to set up the PE database.

We are still using the root user. From the /<pe install path>/bin directory, issue the command **./pecentralize <pe_instance_name>** to start the next step in the installation. This utility will perform the following tasks:

- Checks that the installation appears correct.
- Provides a list of what information you need to complete this utility and shows what values were gathered.
  - DB2PE Group: The name of the group of users that will log on to the PE Server.
  - JRE Root Path: The default JRE root path is /<install dir>/java142 or /<install dir>/java142_64, depending on whether a 32-bit or 64_bit DB2 instance is used for PE server. You must change this if you are using a JRE other than the default. You should also verify if the correct 32 or 64 bit version is taken.
  - DB2PE Working Directory: Change this if you want to use a different path.
  - EXTSHM_IS_ON: You must answer Yes or the utility will stop.
- Creates <user home directory>/db2pe/v22/<instance name>.

Figure 2-7 shows the first question you need to answer. If you do not want to use one or more of the defaults, enter *n*. In the next screen, you can change those values you wish to override.

```
Parameters for the DB2 instance db2in8pe:

DB2PE Group:              [db2pegrp]
JRE Root Path:            [/opt/IBM/db2pesrv/V2.2/java142]<-- recommended
for Performance Expert
DB2PE Working Directory:  [/home/db2in8pe/db2pe/v22]
EXTSHM=ON? (yes or no):   [no]

Take these default values (y|n) [y]  n
```

Figure 2-7 The list of defaults to be taken for the PE instance

In Figure 2-8, we are using our own DB2 group (db2pe) and allowing EXTSHM to be turned on for the PE instance. Once you are satisfied with these options, enter y. On your screen, you should see various operations as **pecentralize** checks and changes your system. All of these operations should result in success and be tagged as such.

```
Type the value you want to take or press Enter to take the default.

DB2PE Group:              [db2pegrp] db2pe
JRE Root Path:            [/opt/IBM/db2pesrv/V2.2/java142]
DB2PE Working Directory:  [/home/db2in8pe/db2pe/v22]
EXTSHM=ON? (yes or no):   [no] yes

Take these values (y|n) [y]  y
```

Figure 2-8 PE instance options selection screen

At the point where you are asked whether you wish to configure the PE DB2 instance, select 1 for Run the configuration later and turn the PE installation over to the DBA. PE Configuration is discussed in Chapter 3, “Configuration and verification” on page 45.

If there are errors while running **pecentralize**, an install error log, pecentralize.log, will be created in the /tmp directory.

The configuration parameters taken above are saved in a configuration file located in /var/db2pe/v22/db2pesrv.cfg. This file stores the configuration information for all DB2 instances where DB2 Performance Expert Server has been installed. Thus if you are installing Fix Pack 1 of DB2 Performance Expert Server V2.2, the same values are presented to you when running **pecentralize** again.

The following configuration information is saved:

```
db2in8pe.db2pe_group=dbpe
db2in8pe.jre_path=/opt/IBM/db2psrv/V2.2/java142
db2in8pe.db2pe_homedir=/home/db2in8pe/db2pe/v22
db2in8pe.extshm_is_on=yes
```

Installation verification

After installation, you can do a simple installation verification by starting/stopping the PE Server. Before starting the PE Server, first make sure that the DB2 instance on which PE Server is running has been started.

- Start PE Server

  Log in as the DB2 instance owner of the instance on which PE Server is running and execute **pestart** from the bin subdirectory of your installation directory.

The PE Server will immediately terminate if the master database DB2PM has not yet been created. You will get the actual version and code level information of PE server that has been installed.

The master database is automatically created when you start to configure the PE Server by running the **peconfig** tool for the first time (see Chapter 3, “Configuration and verification” on page 45).

- Stop PE Server

  If you need to stop the PE Server, log in as the DB2 instance owner of the instance on which PE Server is running and execute **pestop** from the bin subdirectory of your installation directory. If the master database DB2PM is available, the first time you start PE Server, it may take longer due to the initialization process. You should wait until PE is completely started before stopping the PE.

Check also the server-level db2pesrv.log file described in 3.2, “Installation verification and support” on page 122 to be sure the server started/stopped properly.

2.2.3 Installing Performance Expert Server on Windows

The install of the PE Server on Windows is accomplished by using the InstallShield GUI. Our objective is to provide a preview of the screens to which you have to respond. This discussion is intended to supplement Chapter 3, “Installing Performance Expert Server on Windows systems”, *IBM DB2 Performance Expert for Multiplatforms IBM DB2 Performance Expert for Workgroups Installation and Configuration*, SC18-9191.

Before you start, you must have answers for the values listed in Table 2-4 on page 23:

- You must decide whether to use the default install path or not.
- You must use the default DB2 Administrator or you must create or select a user with the correct DB2 authority and Windows privileges.
- While it is optional, it is recommended you place PE in a separate DB2 instance. This separate instance must exist before you start installation if you plan to place PE in its own instance.
- The PE administration group has to exist.
  - With PE V2.2, you can also create this group during installation.
  - The group name cannot exceed eight characters. With DB2 UDB V8.2 on Windows, the group name limitation is 30 characters.
  - The users that will access the PE Server from PE Clients should also be included in the group.

- You must decide whether the PE Server service should be started at system initialization or be started manually. For a production system, it would make sense to start PE at system initialization. For a test system, it would depend on how much you intend using the PE Server.

The Windows install of the Performance Expert Server (PE) starts with entering **db2pe.server.v2.2.install-on-win.exe** from a command line. The sequence of GUI screen titles follows with a comment about each. We placed PE Server installable files in a temporary directory before starting the install. If you use an install CD, insert the CD-ROM and mount the drive.

- InstallShield welcome screen

  This screen is the initial welcome screen. Select **Next** to continue.

- License Agreement

  After reading the License agreement, select **Yes** to continue.

- Choose Destination Location

  The default path is C:\Program Files\IBM\DB2 Performance Expert Server v2. If you have decided to not use the default path, you should identify the path you want to use now. Select **Next** when you are done. We selected the default path.

- Select Service Account

  In this screen, you are requested to select the database administrator for the PE database. Notice that the screen also states this same user should be used when running the Performance Expert configuration utility. We elected to use the default DB2 administrator. Figure 2-9 is an example of what the install GUI looks like; in this case, it is the Select Services Account screen. You are offered the default DB2 administrator or, from a list, you can select a user you created for this purpose. Select **Next** when you are ready.

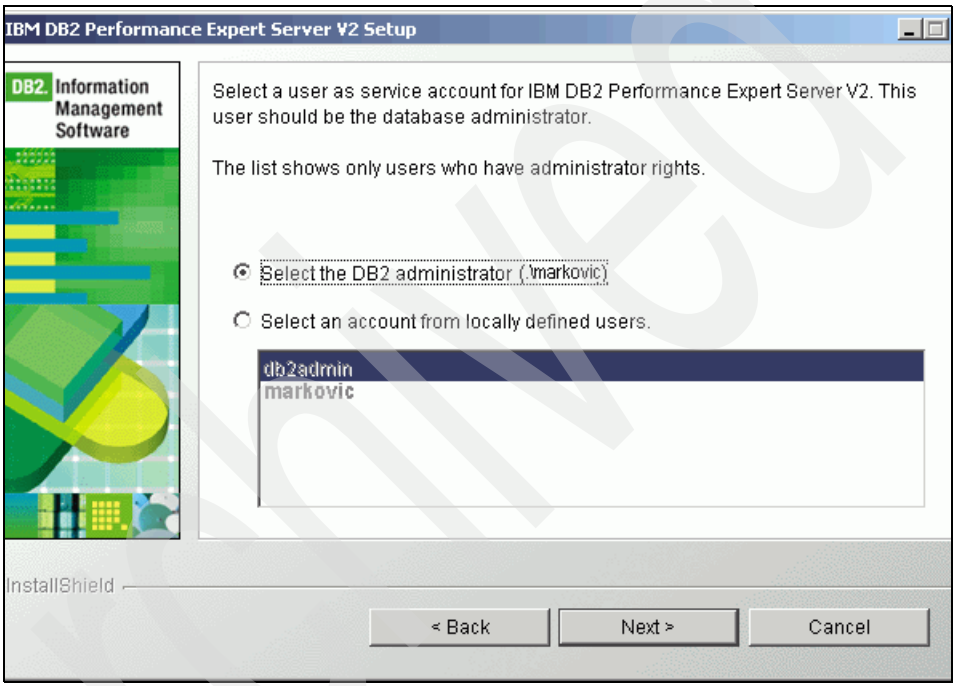


Figure 2-9 Select Service Account window

- Type password

  Enter the password for the user selected on the Select Service Account screen and displayed on this screen. Select **Next** when you are done.

- Select User Group

  On this window is a list of all the user groups for this system. You can also create a group and add users from this panel. From the list of user groups, select the group name you planned for this purpose. Select **Next** to continue.

- Select DB2 Instances

  Select the instance you have decided to use for the PE database. On this screen, it is recommended you install PE on a separate DB2 instance.

Figure 2-10 is the selection area of the Select DB2 Instance screen. Choose an instance and select **Next** to continue. We are selecting the default DB2 instance, DB2.

Figure 2-10 Select DB2 Instances

- Setup will now start the installation

  The installation path and the instance name are displayed. You should verify they are correct before selecting **Next**.

- Setup Status

  This screen shows the status of the files being written to disk. When this is complete, the next screen automatically pops up.

- Start of Performance Expert Server Service

  This pop-up wants you to select whether the PE Server service will be started at system initialization or manually. We elected to start the PE Server service automatically.

  Figure 2-11 shows the selection area of the Start of Performance Expert Server service screen. Once you have made your selection, hit **Next** to continue.

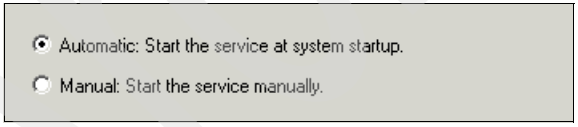


Figure 2-11 Start of PE Server service selection

- Setup has finished

  On this screen, you are given the option to continue with the Configuration of Performance Expert Server or to leave the install process so you can return later to complete the server configuration. The options are **Configure Performance Expert Server** or **Finish installation**. Choose one to continue. We elected to exit here and configure PE later.

- InstallShield Wizard Complete screen

  If you decided to exit the install on the prior screen, the InstallShield Wizard Complete screen is displayed. The install is complete; select **Finish** to exit InstallShield.

Near the end of Chapter 4, “Configuring Performance Expert Server on Windows systems”, *IBM DB2 Performance Expert for Multiplatforms IBM DB2 Performance Expert for Workgroups Installation and Configuration*, SC18-9191 is a discussion regarding the enablement of the client’s TCP/IP connection, which needs your verification.

Note that errors are presented as pop-up windows and are logged to the file db2pesrv.log. The name of the path is presented in the InstallShield Wizard Complete window and is usually C:\Documents and Settings\Administrator.

Installation verification

After installation, you can do a simple installation verification by starting/stopping the PE Server. Before start the PE Server, first make sure that the DB2 instance on which PE Server is running has been started. There are several ways to start/stop PE Server in a Windows environment. We have a detailed discussion on this topic in Chapter 3, “Configuration and verification” on page 45. Use the following method to do the initial installation verification:

- Start PE Server

  Select **Start** → **Programs** → **IBM DB2 Performance Expert Server** → **Start Performance Expert Server**. A command prompt window will open containing status information about PE Server, and this window needs to remain open while PE Server operates.

- Stop PE Server

  Select **Start** → **Programs** → **IBM DB2 Performance Expert Server** → **Stop Performance Expert Server**.

Check the server-level db2pesrv.log file described in 3.2, “Installation verification and support” on page 122 to be sure the server started properly.

Analogous to the installation verification of PE Server on UNIX, PE Server will terminate when the master database DB2PM has not been created yet. The master database will be created when running the PE Server configuration command-line tool or the PE Server configuration GUI for the first time. This is described in detail in Chapter 3, “Configuration and verification” on page 45.

2.2.4 Installing Performance Expert Client on Windows

The installation of the Performance Expert Client on Windows is very similar to the install of the Server on Windows. The GUI looks the same and the sequence of screens is very similar. This discussion is intended to supplement the information found in Chapter 5, “Installing Performance Expert Client on Windows systems”, in *IBM DB2 Performance Expert for Multiplatforms IBM DB2 Performance Expert for Workgroups Installation and Configuration*, SC18-9191 and the product readme.

The only decisions you need to make before installing the client are whether you will use the default path and the client type (on the Setup Type screen). For this discussion, we placed the Client installable files in a temporary directory on the client machine.

Start the PE Client installation by issuing the command **db2pe.client.v2.2.install-on-win.exe**. A sequence of screens follows:

- InstallShield Welcome Screen

  Select **Next** to continue.

  If you have the PE Version 1 Client installed, the next window will ask you to select the software to be uninstalled. The PE Client install code will uninstall the selected client code and migrate the PE Version 1 system configuration.

- License Agreement

  After you read the license agreement, select **Yes** to continue.

- Select Performance Expert Client

  There are four clients to choose from. Three are for the Windows PE Client that works with the zOS database. We are installing the Performance Expert for Multiplatforms. Select **Next** to continue with the install.

- Choose Destination Location

  The default path is C:\Program Files\IBM\IBM DB2 Performance Expert V2. If you wish to install the PE Client in a different path, select that path and click **Next** to proceed.

- Setup Type

  Be aware that when you hit **Next** on this screen, the client files of the client type you selected will be installed. There are three client types from which to choose:

  – Typical: This is the option we are using for this discussion. The complete client is installed.

– Custom: Allows you to select what you wish to install. You can select this to see what the options are and return to the Setup Type window without installing the client code.

Once you have made your choice and selected **Next**, the Setup Status screen is displayed. This status screen closes automatically when the file copy is completed and the next screen is displayed.

- InstallShield Wizard Complete

  Select **Finish** to exit the install process.

An icon for the PE Client is placed upon the desktop.

Install errors are presented in the form of a pop-up window and are logged to the db2pecli.log file. The name of the path is presented in the InstallShield Wizard Complete window and is usually C:\Documents and Settings\Administrator.

At this point, you are ready to configure the Client.

2.2.5 Installing Performance Expert Server V2.2 Fix Pack on AIX

To install a Fix Pack for the PE AIX Server, the procedure is almost identical to the installation of the base code. For this procedure, we are using V2.2 Fix Pack 1. We strongly recommend that you always download the Fix Pack readme from the Web at `http://www.ibm.com/software/data/db2imstools/support/fixpaks.html` and be familiar with Chapter 9, “Installing Performance Expert Server on UNIX and Linux systems”, in *IBM DB2 Performance Expert for Multiplatforms IBM DB2 Performance Expert for Workgroups Installation and Configuration*, SC18-9191. To start the Fix Pack installation:

- Log in to the AIX system as root.
- Copy the Fix Pack 1 file to the AIX system and unpack it.
- When you are ready to begin, stop the PE Server.
- To start the Fix Pack install process, enter the command **`./db2pe.server.v2.2.fp1.install-on-aix -console`**.
- The license agreement is displayed just like in Figure 2-3 on page 25 and you should respond in the same manner.
- The status of the files being written is displayed, as shown in Figure 2-6 on page 27. When the file transfer is complete, you should select 1 for Finish.

2.2.6 Installing Performance Expert Server V2.2 Fix Pack on Windows

Installing PE Server Fix Packs on the Windows platform is quick and simple. It is assumed you have the base code already installed. We have downloaded the PE V2.2 Fix Pack 1 file and unzipped it to a temporary directory. We stop the PE Server before we begin as specified in the readme file. To start the install, issue **db2pe.server.v2.2.fp1.install-on-win.exe**. The following sequence of Fix Pack install GUIs is brief and generally looks just like the install GUIs presented in 2.2.3, “Installing Performance Expert Server on Windows” on page 30. Please reference Chapter 3, “Installing Performance Expert Server on Windows systems“, in *IBM DB2 Performance Expert for Multiplatforms IBM DB2 Performance Expert for Workgroups Installation and Configuration*, SC18-9191.

- The InstallShield initializing screen will flash by as it sets up.
- Welcome to the InstallShield Wizard for IBM DB2 Performance Expert Server

  The Welcome screen is displayed. You should select **Next** to continue.
- Setup will now start the installation

  The path and DB2 instance where the PE Server is currently installed is displayed. You are provided with the usual **Back**, **Next**, or **Cancel** options. We will assume you are continuing by selecting **Next**.
- Setup Status

  The status bar of the files being written to the install path is displayed. When the file transfer is complete.
- Update Complete

  If you have not seen an error presented in a pop-up, then the update is complete. Select **Finish** to exit the Fix Pack install. If an error does occur, the PESService log is created in the <install path>/bin directory.

2.2.7 Installing Performance Expert Client V2.2 Fix Pack on Windows

The process for installing maintenance on the PE Client is faster than the Server. We have downloaded the Fix Pack and unzipped the file into a temporary directory. To start the install, issue **db2pe.client.v2.2.fp1.install-on-win.exe**.

- Welcome to Performance Expert Screen

  The Welcome screen displays the current PE maintenance level and the Fix Pack maintenance level. You should select **Next** to continue.
- License Agreement

  The license agreement is displayed and you must select **Yes** to continue.

- Select Performance Expert Client

  The default is the PE Client for Multiplatforms, which is what we are currently updating. You must select **Next** to continue.

- Installation summary information

  The installation path, additional features, and the total size of the installation is displayed. You must select **Install** to start the installation.

- Setup Status

  The file transfer status is displayed and then disappears when complete.

- Update Complete

  The files are transferred and the install complete. You must select **Finish** to exit.

Unlike the initial install of the base client code, you do not have to identify the install path or install type. If an error occurs, a pop-up window should be presented at this point in the Fix Pack install.

2.2.8 Migrating Performance Expert Client from Version 2.1 to Version 2.2 on Windows

PE Clients support all servers that are the same level or downlevel (except V1). Before you start migrating your Performance Expert Server, please make sure that all your Performance Expert Clients have been migrated to the same level in advance.

To migrate Performance Expert Client from Version 2.1 to Version 2.2, do the following steps:

- Uninstall Performance Expert Client V2.1.

  You will not lose your configuration information; just the program files are removed.

- Install Performance Expert Client V2.2.

If you do not uninstall V2.1 first, you can start and run V2.2 without any problem, but a second program entry is created in the Windows Add/Remove Programs dialog for the Performance Expert Client. You will not be able to uninstall the V2.1 entry. To remove the V2.1 entry, you would first have to uninstall V2.2 and then remove the V2.1 entry.

2.2.9 Migrating Performance Expert Server from Version 2.1 to Version 2.2 on Windows

Performance Expert Server V2.2 comes with a new installation wizard, InstallShieldX. The installation wizard will not automatically replace V2.1 with V2.2. The installation of V2.2 will be rejected if V2.1 still exists. In order to preserve your current configuration settings and to keep the performance data in the performance databases, you should follow some hints when uninstalling the previous version, which we describe in the following paragraphs.

In summary, the following steps must be performed:

- Back up Performance Expert Server databases (optional).
- Uninstall V2.1, but do not unconfigure it.
- Install V2.2 reusing the V2.1 installation parameter
- Migrate the databases by starting Performance Expert Server V2.2.

A detailed description is given below:

- Start the V2.2 installation to gather the needed information provided in the installation rejection message, as shown in Figure 2-12, when the V2.2 detected that an older version exists. This data must be reused when installing the new version.

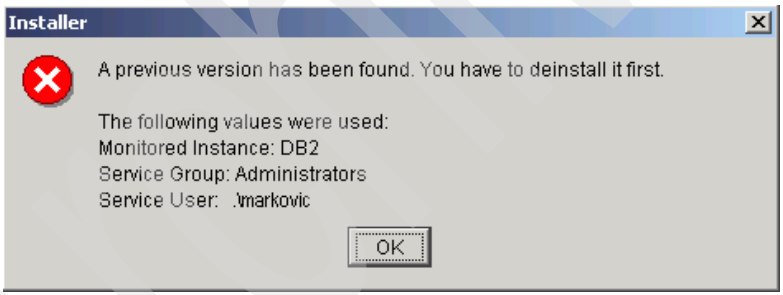


Figure 2-12 Version 2.2 Installer rejection window

- Back up databases.

  You should back up all databases created by Performance Expert Server V2.1. You will then be able to rollback if migration to V2.2 fails for some reason.

  These databases are:

  – The master database DB2PM, which owns the configuration data.

  – One performance database for every instance that has been monitored in the past. If you are not sure about the performance databases, you can start the Performance Expert Server configuration tool and list the

configuration data of each instance that shows the name of the performance database.

- Uninstall Performance Expert Server V2.1.

  Start the Windows Control Panel, double-click **Add/Remove Programs**, select **IBM DB2 Performance Expert Server for Multiplatforms**, and click the **Change/Remove** button.

  When the InstallShield wizard starts, select **Remove** and click the **Next** button.

  Confirm the question with **Yes** when you are asked whether you want to completely remove the selected application and all of its components.

  Next you are asked to unconfigure Performance Expert Server before removing it. *Do not invoke the unconfiguration utility*. Otherwise, you will loose your configuration and performance data. Instead, select **The unconfiguration is already complete** radio button. See Figure 2-13 for details. Then click the **Next** button.

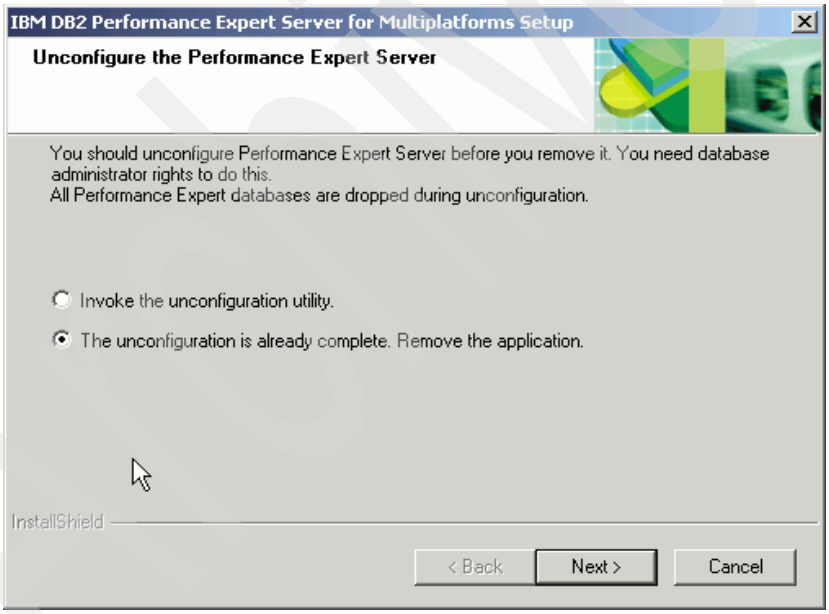


Figure 2-13 Unconfigure Performance Expert Server window

- The uninstallation process deletes the program files. You are then asked whether you want to remove files that are still left in the installation directory.

Chapter 3, “Migrating Performance Expert Server from Version 2.1 to Version 2.2”, in *IBM DB2 Performance Expert for Multiplatforms IBM DB2 Performance Expert for Workgroups Installation and Configuration*, SC18-9191 states that you should select **No** to keep the files. No files need to be kept for migrating to Version 2.2, but the master database might have defined some working directories for event monitoring files as subfolders of the installation directory. These will be deleted if you select **Yes**. If the files and folders are deleted, the Performance Expert Server Version 2.2 will stop with an error that states that the folders are missing. Therefore, if you have instructed the InstallShield to delete the folders, you have to change the configuration of the event monitor local path. V2.2 FP1 Server will recreate these folders automatically at startup.

You have two alternatives:

- Keep the files in any case; you do not have to adapt your configuration afterwords.
- Remove all files and folders. After installing V2.2, you might then need to check for the event monitoring local path in your master database and change it to another directory that already exists.

We recommend keeping them, but it is okay if you deleted the files and folders.

- Install Performance Expert Server V2.2. Reuse the installation parameters given in the initial installation rejection window shown in Figure 2-12 on page 39, that is, use “Service User” for the user to be used as service account, use “Service Group” for the user group, and use (misleadingly named) “Monitored Instance” for the DB2 instance on which Performance Expert Server is to run.
- Start the Performance Expert Server.

  During startup, the server migrates all the required databases. If the sever stops because some working directories are not available, just change the configuration or recreate the required directories as described above. Then start the server again.

  During the migration process, tables are moved from tablespaces of 4 KB in size to tablespaces of 8 KB in size. Ensure that enough space is available in the tablespace directory of the performance database.

  Three tables are moved:

  - DB2PM.STATEMENT from tablespace HISTTSP to HISTTSP8K
  - DB2PM.DBCFG from tablespace HISTTSP to HISTTSP8K
  - PWH.DBCFG from tablespace PWHTSP to PWHTSP8K

2.2.10 Migrating Performance Expert Server from Version 2.1 to Version 2.2 on AIX

To migrate Performance Expert Server from V2.1 to V2.2, do the following steps:

- Back up databases.

  You should back up all databases created by Performance Expert Server V2.1 in order to be able to rollback if the migration fails for any reason.

  The applicable databases are the master database DB2PM, and one performance database for every instance that has been monitored in the past. The list of the applicable databases can be determined by the Performance Expert Server configuration tool. Display the configuration data for every instance that shows the performance database used for the instance.

- Install Performance Expert Server V2.2 in a directory separate from Performance Expert Server V2.1.

  See 2.2.2, “Installing Performance Expert Server on AIX” on page 23 for how to install.

- Run the script **`pecentralize`** from the /<pe install path>/bin directory for each DB2 instance on which Performance Expert Server V2.1 ran before.

- Start the Performance Expert Server.

  Enter the command **`./pestart`** from the /<pe install path>/bin directory. During startup, the server migrates all the required databases.

  During the migration process, tables are moved from tablespaces of 4 KB in size to tablespaces of 8 KB in size. Ensure that enough space is available in the tablespace directory of the performance database.

  Three tables are moved:

  - DB2PM.STATEMENT from tablespace HISTTSP to HISTTSP8K
  - DB2PM.DBCFG from tablespace HISTTSP to HISTTSP8K
  - PWH.DBCFG from tablespace PWHTSP to PWHTSP8K

- If the migration was successful, you may then decide whether you want to remove Performance Expert Server Version 2.1. But be aware not to unconfigure it.

2.2.11 Enabling CIM

In order to collect operating system data, you must install and configure a Common Information Model Object Manager (CIMOM) on the monitored system before you configure Performance Expert Server. The following versions are currently supported by Performance Expert Server:

- CIMOM Pegasus Version 2.3.2 or later for AIX
- WBEM Services 2.5 of Solaris 9 or later, part of which is CIMOM for the Solaris Operating environment
- OpenPegasus 2.5 and SBLIM providers for Linux

The following sections describe the installation of a Common Information Model (CIM) server on AIX. For the Solaris Operating System environment and for Linux, see Chapter 10, “Configuring Performance Expert Server on Unix and Linux systems”, in *IBM DB2 Performance Expert for Multiplatforms IBM DB2 Performance Expert for Workgroups Installation and Configuration*, SC18-9191.

You must install:

- CIMOM Pegasus Version 2.3.2 or later

  It is part of AIX Expansion Pack 5.2.6 or later. Install the following file sets by using the System Management Interface Tool (SMIT) or the **`installp`** command:

  – sysmgt.pegasus.cimserver

    Installs the Pegasus CIM Server file sets in the /opt/freeware/cimom/pegasus directory.

  – sysmgt.pegasus.osbaseproviders

    Installs the base providers for AIX file sets in the /usr/pegasus/provider directory.

  To install the packages using SMIT, complete the following:

  a. At the command line, type `smitty`.

  b. Select Software Installation and Maintenance → Install and Update Software → Install Software.

  c. At the Input Device/directory for software field, press the F4 key to view a list of options.

  d. Select the option that reflects the location or media that contains the CIM packages.

  e. At the Software to Install field, press the F4 key to view a list of package options.

f. Select the sysmgt.pegasus.cimserver and sysmgt.pegasus.osbaseproviders packages by pressing the F7 key.

g. Accept the new license agreement before continuing with the installation.

h. Start the installation.

To verify that the CIM Server file sets were installed correctly, use the **lslpp** command as follows:

```
lslpp -al sysmgt.pegasus.cimserver.rte
```

- RPM file OpenSSL

It must be installed in order for the CIM server to run. However, Performance Expert does not yet support securing the CIM server, that is, data transferred from the CIM server to the Performance Expert Server is not yet encrypted. This feature will likely come with one of the next releases or Fix Packs.

The RPM file is contained on the AIX Linux Toolbox CD. You can also download it from the following Web site:

http://www-1.ibm.com/servers/aix/products/aixos/linux/download.html

On this Web site, select **AIX Toolbox Cryptographic Content** under the sorted download heading on the right of the page. After you have registered, you can download openssl-0.9.6k-1.aix4.3.ppc.rpm or a later version.

To determine if the RPM file is installed on your system, run the following commands:

```
rpm -q -f /opt/freeware/lib/libssl.a
rpm -qa |grep -i openssl
```

If both the libssl.a library and the openssl-0.9.*6XXX* RPM, where *XXX* indicates the build level, are found, then OpenSSL is installed on your system.

To install the OpenSSL RPM file, run the command:

```
rpm -ivh openssl-0.9.6 XXX .rpm
```

where *XXX* indicates the build level.

For more information about installation of CIMOM and OpenSSL, refer to the release notes of the expansion packs and to the corresponding Common Information Model Guides on the following Web sites:

For AIX Expansion Pack 5.2.6, refer to:

http://www.ibm.com/pseries/en_US/aixbman/cim/cimtfrm.htm

For AIX Expansion Pack 5.3.2, refer to:

http://publib.boulder.ibm.com/infocenter/pseries/topic/com.ibm.aix.doc/aixbman/cim/cim.pdf||

3

Configuration and verification

Configuration is the next step after installation. This chapter describes the steps necessary for a successful DB2 Performance Expert configuration. The information contained in this chapter is meant to complement the material found in the DB2 Performance Expert for Multiplatforms product documentation, *Installation and Configuration,* SC18-9191.

This chapter discusses the following:

- Configuration: This required PE utility is performed by the DBA, but information determined during planning is required to perform this task.
- Verification: Guides for installation verification and troubleshooting tips.

3.1 Configuration

In this section, we discuss how to configure PE Server to monitor DB2 instances and how to configure PE Client to view performance information for those instances. We include a worksheet for the PE Server configuration and step through a complete example using the worksheet. We also describe some preliminary steps for using PE Client.

Note: The PE Server, Client, and documentation often use the term “DB2 Systems” to refer to DB2 instances (multiplatforms), DB2 subsystems (z/OS), and DB2 Connect™ gateways. This redbook focuses on multiplatform environments, so we refer here to DB2 instances.

After product installation, perform the following steps before monitoring DB2 instances:

- PE Server configuration
  - Select the desired configuration method.
  - Select the desired event monitoring configuration (shared file system or not).
    - Install the shared library on the monitored instance, if necessary.
  - Prepare the information worksheet.
  - Configure monitoring.
    - Add DB2 instances to monitor.
    - Add databases to monitor.
    - Enable monitoring for added DB2 instances.
    - Enable event monitoring.
    - Verify configuration.
  - Start or restart PE Server
- DB2 instance configuration
  - Enable desired default monitor switches on monitored instance.
  - DB2 instance configuration for monitored instance and PE Server instance.
  - Configure TCP/IP for client communication on the PE DB2 instance.
- PE Client configuration
  - Add DB2 instances to monitor.
  - Change PE Client configuration settings.

– Change PE Server properties.
– Customize monitoring for monitored DB2 instances as desired.

3.1.1 PE Server configuration and related tasks

In this section, we discuss the three available methods of PE Server configuration, recommended DB2 instance configuration changes, event monitoring considerations, and starting and stopping PE Server. PE Server may be started or stopped during configuration, but you will need to stop and restart PE Server after most configuration changes in order for them to take effect. This is true for Performance Expert Server until V2.2. With PE Server V2.2 Fix Pack 1, this obstacle has been removed and you can dynamically activate the configuration changes while the PE Server is running without needing to restart it. We discuss this enhancement in more detail in 3.1.5, “New PE server configuration features of V2.2 Fix Pack 1” on page 79.

Before configuring PE Server, we recommend that you read all the applicable sections of the version of the PE manual *IBM DB2 Performance Expert for Multiplatforms IBM DB2 Performance Expert for Workgroups Installation and Configuration*, SC18-9191 corresponding to the version and Fix Pack you have installed.

The following configuration methods are available for PE Server:

- Interactive mode (command line)

  Use this method if you want to use the command line to manually configure a PE Server to monitor one or multiple DB2 instances.

- Interactive mode

  Use this method if you want to use a graphical interface to manually configure a PE Server to monitor one or multiple DB2 instances. This method contains the same functionality as the command-line method.

- Silent mode

  Use this method if you want to configure a PE Server to monitor a large number of DB2 instances in a single step. This method uses a response file in the installation. It may save time if the monitored instances have multiple parameter values in common. The silent mode contains the same functionality as the interactive methods with the exception that you cannot change your existing instance configuration in silent mode. The monitored instance alias can be customized in a silent configuration. With V2.2 Fix Pack 1, configuration definitions are activated immediately, that is, monitoring is started automatically, when the PE server is running (see 3.1.5, “New PE server configuration features of V2.2 Fix Pack 1” on page 79).

Each configuration method must be performed while logged in as the DB2 instance owner of the instance on which PE Server runs (UNIX) or the user defined during product installation (Windows).

After PE Server configuration is complete, the next step is to open PE Client and configure it to connect to the desired monitored instances.

Important: Be aware of the monitoring overhead incurred and table and disk space needed for monitoring information immediately after you have configured PE Server to monitor one or more DB2 instances and have restarted PE Server. At this time, PE Server will begin immediately to take snapshots on monitored databases, regardless of whether you have logged on in the PE Client or not. These snapshots will be taken using default values (multiple snapshots per minute for certain components, with the default 50 hours of data stored in tables in the performance database for each monitored instance) unless you change them from the PE Client.

Command-line interactive configuration

Invoke the command-line configuration utility for PE Server as described below.

- In Windows, use one of the following methods:
  - Click **Start** → **IBM DB2 Performance Expert Server V2** → **Configure Performance Expert Server in console mode**.
  - Run **`peconfig.bat -console`** from the bin subdirectory of the installation directory
- In UNIX, run **`peconfig -console`** from the bin subdirectory of the installation directory.

The `peconfig =>` prompt indicates you are working in the configuration utility. For a complete command reference for this utility, use the **`help`** command at this prompt, or see Appendix A, “Command reference for the configuration of Performance Expert Server by using the command-line utility”, in *Installation and Configuration,* SC18-9191. Perform the following steps:

1. Use ADDLOCINST or ADDREMINST to register one or more local or remote DB2 instances for monitoring, or use DROPINST to unregister an instance for monitoring.

   Note: The term “local instance” in this context refers to the DB2 instance on which PE Server is running, and the term “remote instance” refers to any other instance, regardless of whether it is running on the local server or a remote server.

2. Use ADDDB to add one database on a local or remote monitored DB2 instance for monitoring, or use REMOVEDB to remove a database on a monitored instance from monitoring. Use ADDALLDBS to add all databases for monitoring on a local or remote DB2 instance (Adding all databases to a remote instance is new with V2.2. The prerequisite is that all remote databases are cataloged to the local instance.). Repeat as necessary to add multiple databases.
3. Use ENABLE to enable monitoring of a registered DB2 instance, or use DISABLE to disable monitoring of a previously enabled instance.
4. If desired, use EVMON to enable event exception monitoring for a registered database, or use EVMOFF to disable event exception monitoring for a registered database.

At any point during or after configuration, use the LIST command to list details about monitored DB2 instances, or LIST InstanceID to view detailed information about one monitored instance.

Use CHANGE to alter any of the following parameters for a monitored instance:

- Port number
- User login name or password
- Time zone difference between PE Server and monitored DB2 instance
- Local and remote paths to shared file system for event monitoring data
- Whether or not a shared file system for event monitoring files will be used
- Whether to configure access to a CIM server (V2.2 and later)

Type **`exit`** or **`quit`** at the prompt to leave the utility.

GUI interactive configuration

PE Server V2.1 Fix Pack 2 or higher must be installed to use this feature. Invoke the GUI configuration utility for PE Server as described below.

- In Windows, use one of the following methods:
  - Click **Start** → **IBM DB2 Performance Expert Server V2** → **Configure Performance Expert Server in graphical mode**.
  - Run **`peconfig.bat`** from the bin subdirectory of the installation directory.
- In UNIX, run **`peconfig`** from the bin subdirectory of the installation directory. UNIX systems must also be configured to run GUIs to use this configuration method.

See Appendix D, “Configuring Performance Expert Server by using the configuration GUI”, of *Installation and Configuration,* SC18-9191, for a complete discussion of the functionality of this utility. Use the graphical interface to perform the following tasks:

- Create a PE master database if one does not already exist. The PE master database is created implicitly by the GUI, as in the command-line mode.
- Add or remove local or remote DB2 instances for monitoring.
- Enable or disable a DB2 instance for monitoring.
- Change any of the alterable parameters for a monitored instance.
- Add or remove individual databases on local or remote monitored DB2 instances for monitoring, or add all databases on a local DB2 instance for monitoring.
- If desired, switch event monitoring on or off for a database that is registered for monitoring.

The GUI configuration utility requires you to enter the same information that the command-line utility requires, but the fields are presented in a different order in each utility. Note that some menu items in the GUI configuration tool are context sensitive - they only appear when they are relevant to the object currently selected.

Silent configuration

To configure PE Server in silent mode, perform the following steps:

- Prepare a response file.The sample response file, peresp.txt, can be found in the bin subdirectory of the installation directory for both Windows and UNIX.

  This file contains detailed information about performing a configuration in silent mode, including instructions, syntax, a template, and examples.

  Note: With PE V2.2, the format of the response file changed. PE V2.2 does not support the format of the response file of PE V2.1

- Execute the silent configuration by running the following command from the bin subdirectory of the installation directory:

  ```
  peconfig -silent peresp.txt
  ```

  where peresp.txt represents the name (proceeded by the path, if necessary) of your response file. Performing silent configuration will create the log file pesilent.trc in the following location:

  – In Windows, in <install_dir>\instances\<InstName>, where InstName represents the name of the DB2 instance on which PE Server runs.

– In UNIX, in <working_dir>/<InstName>, where InstName represents the name of the DB2 instance on which PE Server runs.

Event monitoring considerations

As previously mentioned, you can configure PE Server to use or not use a shared file system for event monitoring data. If you choose to use a shared file system, both deadlock event data and SQL activity data will be written to files in this file system. If you choose not to use a shared file system, then deadlock event data will be written to and read from tables located in tablespace USERSPACE1 or in a tablespace that the user specifies in db2pesrv.prop of the monitored database, and SQL activity data will be written to files in a file system local to the monitored instance and accessed by PE Server through a shared library that you must install. You should decide which method to use and create the necessary file systems, and share them if necessary, before beginning configuration. For a more detailed description of defining the exchange of event monitoring data, see *Installation and Configuration,* SC18-9191.

DB2 instance configuration

The snapshot data available to PE Server is dependent upon the settings of the default monitor switches on the monitored DB2 instance. The settings for these switches may be made in the DB2 Control Center or by issuing the appropriate DB2 command. These parameters are configurable online, so you do not need to restart the monitored DB2 instance after changing them. When determining which switches to enable, also consider the overhead on the monitored DB2 instance that may be incurred by snapshots taken by PE. For example, the following command enables all default monitor switches:

```
update dbm cfg using DFT_MON_SORT ON DFT_MON_LOCK ON DFT_MON_TABLE ON
DFT_MON_BUFPOOL ON DFT_MON_UOW DFT_MON_STMT ON
```

To avoid potential problems, you should set the MON_HEAP_SZ parameter of each monitored instance to at least a value of 512. This parameter is not currently configurable online, so you will need to restart the instance after making the change. You may change this parameter through the Control Center or with the following DB2 command:

```
update dbm cfg using MON_HEAP_SZ 512
```

A few other database manager parameters are set during PE Server installation and configuration. You should review the values of these parameters and determine if they are appropriate for your environment, especially if the PE Server runs on the same DB2 instance you are monitoring.

Configuring TCP/IP communication on the PE DB2 instance

TCP/IP communication must be configured in order for any PE Client to log on and have access to the databases maintained by a PE Server. Perform the following steps to configure TCP/IP communication:

- If not already done during instance creation, update the TCP/IP services file with the port number to be used for the DB2 instance to accept client requests.
- Set the DB2 registry variable DB2COMM to TCPIP by executing the following command:

  ```
  db2set db2comm=tcpip
  ```

- Update the SVCENAME parameter in the database manager configuration file with the service name specified in the services file with the following command:

  ```
  db2 update dbm cfg using SVCENAME ServiceName
  ```

- Stop and restart the DB2 instance.

PE configuration differences between Windows and UNIX

The following is a summary of significant PE configuration differences between Windows and UNIX environments:

- The pecentralize script is run on UNIX after installation, but is not run on Windows, where the necessary tasks are performed during product installation.
- In UNIX, PE Server configuration must be performed by the instance owner of the DB2 instance on which PE Server runs. In Windows, configuration must be performed by the ID with administrative privileges that was identified during product installation.
- When using a shared file system for event monitoring data, it must be on the server of monitored instance for Windows environments. For UNIX, this is also recommended, but it also works mostly if the shared file system resides on the PE Server and is mounted on the monitored instance system.

Starting/stopping PE Server

The DB2 instance on which PE Server is running needs to be started before starting PE Server.

To start PE Server:

- In Windows, use one of the following methods:
  - Click **Start** → **Programs** → **IBM DB2 Performance Expert Server V2** → **Start Performance Expert Server**. A command prompt window will open

containing status information about PE Server, and this window needs to remain open while PE Server operates. This method does *not* update the Status column in the Services window. This method calls the script pestart.bat in the bin subdirectory of the installation directory, which can also be run directly.

Important: If you start PE Server on Windows by using the Start menu or pestart.bat and then either close the command window that contains the status information or terminate the batch job it contains, then the PE Server stops, even though the db2pesrv.log file may not log the termination.

– Start PE Server from the Services window. Depending on the operating system level and configuration, this can be accomplished by performing the following steps:

  - Click **Start** → **Settings** → **Control Panel**, double-click **Administrative Tools**, double-click **Services**, and start the service named DB2 Performance Expert Server v2.
  - Click **Start** → **Computer Management**, expand the Services and Applications tree, click **Services**, and start the service named DB2 Performance Expert Service.

  Using this method, the window will reflect that the service is started in the Status column, but no additional information will appear and a command prompt window will not be opened. This service entry calls the executable PESService.exe in the bin subdirectory of the installation directory, which can also be called manually from a command prompt. Run this command without any parameters to see a complete syntax description. Executing the **`net start`** command for the PE Server service from a command prompt also has the same effect. With either of these alternate methods, use the service name that appears in the properties window of the service. This name defaults to PES*inst*, where inst is the name of the DB2 instance on which PE Server runs. For example, you would use the commands in Table 3-1 to start and stop a PE Server with service name PESDB2.

Table 3-1 Syntax for alternate methods of starting/stopping PE Server

Method	Start syntax	Stop syntax
PESService.exe	`PESService -r PESDB2`	`PESService -s PESDB2`
net start/stop	`net start PESDB2`	`net stop PESDB2`

Note: Attempting to start PE Server on Windows through either the Service window or the Start menu when it has already been started by the other method will result in an error, but the PE Server will remain started.

- In UNIX, log in as the DB2 instance owner of the instance on which PE Server is running and execute **pestart** from the bin subdirectory of your installation directory.

You should stop PE Server using the same method you used to start it. To stop PE Server:

- In Windows, use one of the following methods:
  - If PE Server was started using the Start menu, then stop it by clicking **Start → Programs → IBM DB2 Performance Expert Server V2 → Stop Performance Expert Server**. This method calls the script `pestop.bat` in the bin subdirectory of the installation directory, which can also be run directly.
  - If PE Server was started using the Services window, then stop it by accessing the Services window through one of the methods described above in the discussion on starting PE Server.
- In UNIX, log in as the DB2 instance owner of the instance on which PE Server is running and execute **pestop** from the bin subdirectory of your installation directory.

Tip: If you want to schedule a script that includes stopping and starting PE Server on Windows (for example, to cleanly stop PE Server before taking an offline database backup of a monitored database, the PE master database, or a PE performance database), you may use any of the three command-line methods mentioned previously, depending on the method with which PE Server was previously started.

3.1.2 Worksheet

We recommend you prepare values for all parameters before beginning PE Server configuration. Table 3-2 on page 55 shows the parameters for which values you will need to input values when you configure PE Server to monitor DB2 instances.

The PE Server and each monitored DB2 instance will either be:

- On different machines
- In separate DB2 instances on the same machine
- In the same DB2 instance

Different topologies will require different parameters. Parameters marked with the * (asterisk) symbol do not apply when the PE Server shares a DB2 instance with the monitored instance. The parameter marked with the ** (double asterisk) symbol only applies when the PE Server shares a DB2 instance with the monitored instance. Parameters marked with the $ symbol are new in V2.1 Fix Pack 2. Parameters marked with $$ are new with V2.2.

We use the UNIX example values in a command-line configuration example in the next section, and we use the Windows example values in a GUI configuration example. Due to the configuration methods and topologies chosen, certain parameters do not apply to each example and are designated as N/A (not applicable).

Note: DB2 database, database alias, and instance names can have up to eight characters (letters or numbers), but cannot begin with a number. DB2 documentation recommends not using the special characters @, #, and $ in these names to avoid potential communications problems, but PE does not allow any special characters (including the underscore character) in the names of DB2 objects it creates.

Table 3-2 PE Server configuration worksheet - enabling monitoring for local or remote DB2 instances

Parameter	Description	Example value - UNIX - interactive method	Example value - Windows - GUI method
*DB2 instance node name	A name of your choice to be used locally for the **catalog tcpip node** command used by PE Server for the monitored remote DB2 instance.	SICILY1	WISLADB2
*Host name or IP address	The host name or IP address of the monitored remote DB2 instance, or optionally localhost if the PE Server and the monitored DB2 instance are on the same machine.	sicily.almaden.ibm.com®	wisla.almaden.ibm.com

Parameter	Description	Example value - UNIX - interactive method	Example value - Windows - GUI method
*Port number	The port number of the monitored remote DB2 instance. This number is found in the entry in the operating system's services file that contains the service name used for the SVCENAME parameter in the database manager configuration of the remote DB2 instance.	60000	50000
*Monitored DB2 instance alias	A name of your choice to be used within PE Server for the monitored remote DB2 instance. PE Client will show this value as a default during client configuration. This name may be longer than eight characters and may contain special characters or spaces.	SICILY1	WISLA_DB2
User ID	The user ID with administrator rights for the remote DB2 instance.	db2inst1	db2admin
User password	The password for the user ID above.	********	********
Performance database name	A name of your choice for the performance database in which PE Server stores the collected data for each monitored DB2 instance and each monitored database.	PWHINST1	WSDB2PWH
Performance database path	The path where PE Server creates the files for the performance database.	/db2pe	C
Tablespace path	A path of your choice in which to place the containers for the tablespaces of the performance database. You will be prompted to create this directory if it does not already exist.	/db2pe/db2in8pe /pwhinst1	C:\DB2\WISLA

Parameter	Description	Example value - UNIX - interactive method	Example value - Windows - GUI method
*Same time zone (Y/N)?	This parameter refers to the time zones of the server on which the remote DB2 instance runs and the server on which the local DB2 instance used by PE Server runs. The use of this parameter ensures that time stamps will display correctly in PE Client. If no, also enter the code of the time zone for the monitored remote DB2 instance. This value may be found in the file timezone_codes.txt in the bin subdirectory of the installation directory.	Y	Y
*Local path to shared file system for event monitoring files	A path of your choice on the local system to be used for event monitoring data. On Windows, define the shared file system on the server where the monitored DB2 instance is located.	/db2pe/peevm	C:\peevm
*Remote path to shared file system for event monitoring files	A path of your choice from the remote server to the shared file system that will be used for event monitoring data. This path should be defined to reconnect after a server reboot, if that behavior is desirable.	/jampeevm	C:\DB2\peevm
**Path for event monitoring files	A path of your choice for event monitoring files, which contain event monitoring data that PE Server collects for monitored DB2 instances.	N/A	N/A

Table 3-3 shows the information required to register databases on the remote DB2 instance for monitoring.

Table 3-3 PE Server configuration worksheet - adding databases to monitor

Parameter	Description	Example value - UNIX	Example value - Windows
Instance ID (command-line method only)	The number defined by PE Server for the DB2 instance that is defined for monitoring. This number can be obtained by using the LIST command when using peconfig.	1	N/A
Database alias	A name of your choice used as the alias in a **catalog database** command when PE catalogs the database you want to monitor.	SICTRD3	WSPEDEMO
Database name	The name of the database to be monitored on the monitored DB2 instance.	TRADE3DB	PEDEMO
*$Remote database alias	The alias with which the database is cataloged on the remote monitored DB2 instance in which it was created. This value is important when you have databases cataloged as an alias name. The default alias name is the same as the real database name. PE Server uses this alias as the database name in the **catalog database** command.	N/A	PEDEMO
$$CIM server port number	PE Server uses this port number to access the CIM server on the monitored instance machine(s). PE server asks for the port number during configuration, if you selected to collect operating system information from the monitored system. Note that for collecting this data a CIM server must run on the monitored system. See 3.1.3, “Configuring the CIM server” on page 59 for how to determine the port number and how to start the CIM server.	5988 Default HTTP port	N/A

Tip: For ease of administration, consider establishing a naming convention for your PE environment before you begin configuration. Consider the naming of objects, such as the monitored DB2 instance alias, DB2 instance node name, and performance database name, which are included in the PE Server configuration, and also the monitored DB2 instance alias and DB2 connection alias, which are included in the PE Client configuration. For example, you may wish to consider the following options:

- Include characters in each performance database name to help administrators and users to easily identify them as such, such as PWH (performance warehouse), PE, PERF, and so on. You may choose to also identify the monitored instance and the server on which it runs, within the eight-character limit. For example, we use the value WSDB2PWH for the Windows example value in the worksheet for the name of the performance database of instance DB2 on server Wisla.
- You cannot change the monitored instance alias at the server after configuration, but you may change it in the PE Client when it is configured. If you want to keep it the same in all locations, you may choose to use a naming convention for it that reflects the monitored instance name and the server on which it is located, and possibly the name of the server on which the PE Server is installed. Or you might choose to identify DB2 instances used for production, development, testing, or other purposes by beginning the monitored instance alias with a certain letter (p, d, t, and so on).
- If your environment is small, you may choose to use shorter or simpler object names.

3.1.3 Configuring the CIM server

In the configuration examples later in this chapter, we configure an instance for collecting operating system information using a CIM sever on AIX. Before we can start configuring the instance and start monitoring, we need to collect some CIM server related values and prepare the monitored instance to run a CM server. The following steps are prerequisites for configuring a CIM server:

- Start the CIM server and determine the port number used by the CIM server.
- Add a user ID to the CIM server.
- Add a CIM user authorization.

The following section describes the steps for a CIM server running on AIX. For details on how to configure CIM servers, please see Chapter 4, “Configuring Performance Expert Server on Windows systems” and Chapter 9 “Configuring Performance Expert Server on UNIX and Linux systems”, in *IBM DB2 Performance Expert for Multiplatforms IBM DB2 Performance Expert for Workgroups Installation and Configuration*, SC18-9191.

Starting the CIM server

Follow the steps below to start the CIM server:

1. Log in as root at the system that owns the monitored instance.
2. Change to the /opt/freeware/cimom/pegasus/bin directory.
3. Enter the command:

   ```
   cimserver
   ```

   (To stop the CIM server, enter **`cimserver -s`**.)

When the CIM server starts, you will get an output similar to Figure 3-1. The output lists the port number used by the CIM server. Record the port number in the worksheet; you will need it when you configure the instance for monitoring. The default port number is 5988.

```
# cimserver
PGS10020: Logs directory = /opt/freeware/cimom/pegasus/logs
CIM Server 2.3.2
--- CMPI Provider Manager activated
PGS10025:  The CIM server is listening on HTTP port 5,988.
Started.
#
```

Figure 3-1 Starting the CIM server on AIX

Adding a user ID to the CIM server

You must add a user ID to the CIM server. Use the user ID prepared in the worksheet during configuration of the Performance Expert Server. This is to authorize the user to access the CIM server. Note that CIM users must be valid users at the local system. Run the following command:

```
cimuser -a -u db2inst1
```

You are asked to enter the password:

```
Please enter your password:
Please re-enter your password:
User added successfully.
```

Note that there is no default authorization permissions set for a newly added user. Verify that the user was successfully added by entering the following command:

```
cimuser -l
```

The output should list the user ID *db2inst1*.

Note: With CIM server V2.5 or later, adding the user ID is no long a required step and the **cimuser** command is not available any more.

Adding a CIM authorization

CIM manages authorizations by using namespaces. A namespace is a logical unit for grouping to control scope and visibility. When a user submits a request for a namespace, the CIM server checks whether he is authorized for that namespace. By default, basic authentication is enabled.

You may disable authentication. Then you have to stop the CIM server first and restart it with:

```
cimserver enableAuthentication=false
```

If you leave authentication enabled, you have to add a CIM authorization for the user added above. This allows Performance Expert Server to access the CIM server. Enter the following command:

```
cimauth -a -u db2inst1 -n root/cimv2
```

Verify that the authorization was added successfully. Enter the command:

```
cimauth -l
```

The output will look as follows:

```
db2inst1, root/cimv2, "r"
```

You are now prepared to start configuration of the instance in Performance Expert Server.

3.1.4 Server configuration examples

In this section, we walk through complete examples of configuring a PE Server to monitor a remote DB2 instance using both the GUI configuration utility and **peconfig** command-line utility methods.

The example values provided in Table 3-2 on page 55 are used in these examples. Note that these two configuration methods require the information from the worksheet to be entered in a different sequence.

Windows GUI Server configuration example

This example uses the Windows values provided in Table 3-2 on page 55 in the example value column. Most options may be selected either from the drop-down menus or from the context menus that appear after right-clicking on an object; we use the latter method in this example. As shown in the topology diagram in Figure 3-2, the PE Server and the monitored DB2 instance are on separate servers, and we have chosen not to use a shared directory for event monitoring information.

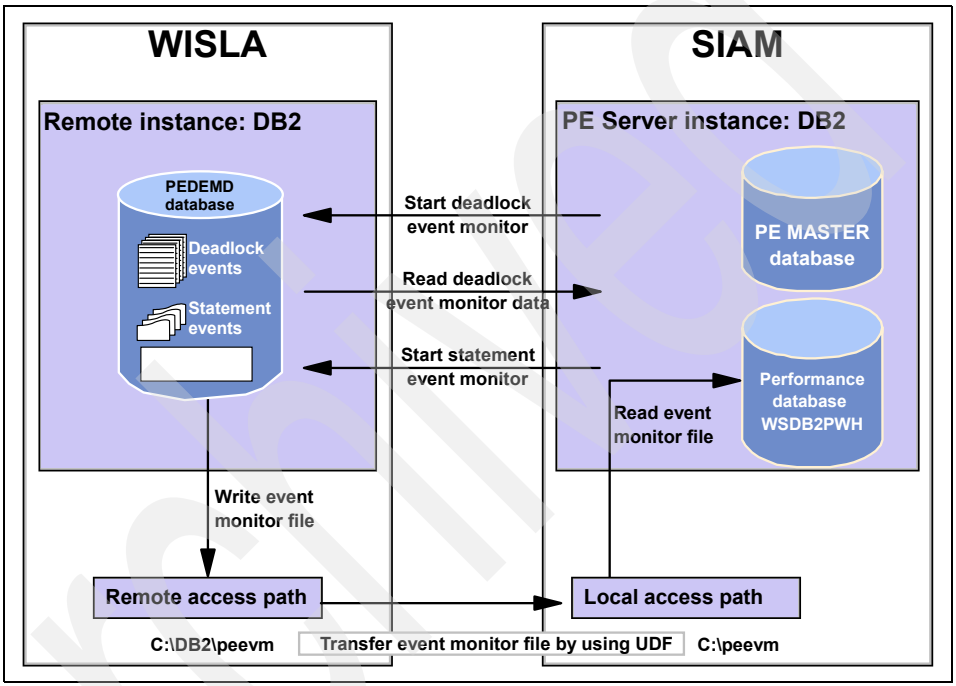


Figure 3-2 Topology for PE Server GUI configuration example

First, we open the GUI configuration utility for PE Server. Because PE Server has not yet been configured on this server, we next see the Create Master Database window in Figure 3-3 on page 63 open in front of the main DB2 Performance Expert Server Configuration window.

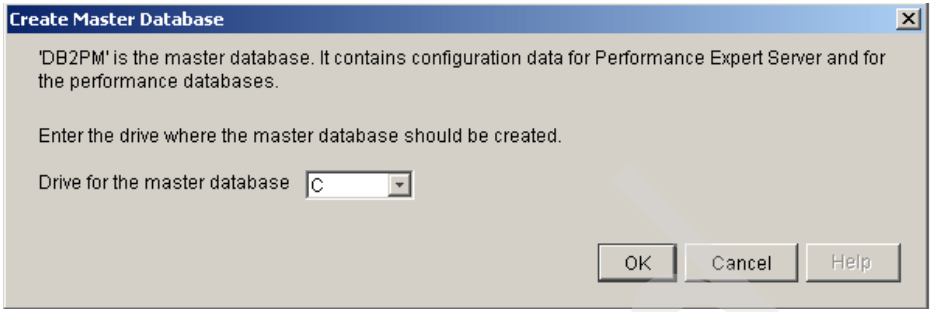


Figure 3-3 Create Master Database window

After leaving the default drive selection of C: and clicking **OK**, the master database is created in C:\DB2. When the master database creation completes, we see a success message at the bottom of the main server configuration window. To add a remote instance for monitoring, we right-click the **Monitored Instances** folder and select **Add remote instance**, as shown in Figure 3-4.

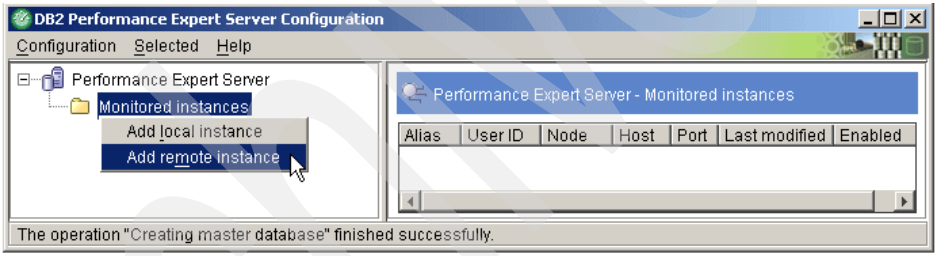


Figure 3-4 DB2 Performance Expert Server Configuration main window

In the Add Remote Instance window, we enter the appropriate values from the worksheet on the DB2 Instance Node tab, as shown in Figure 3-5, and click **Next**.

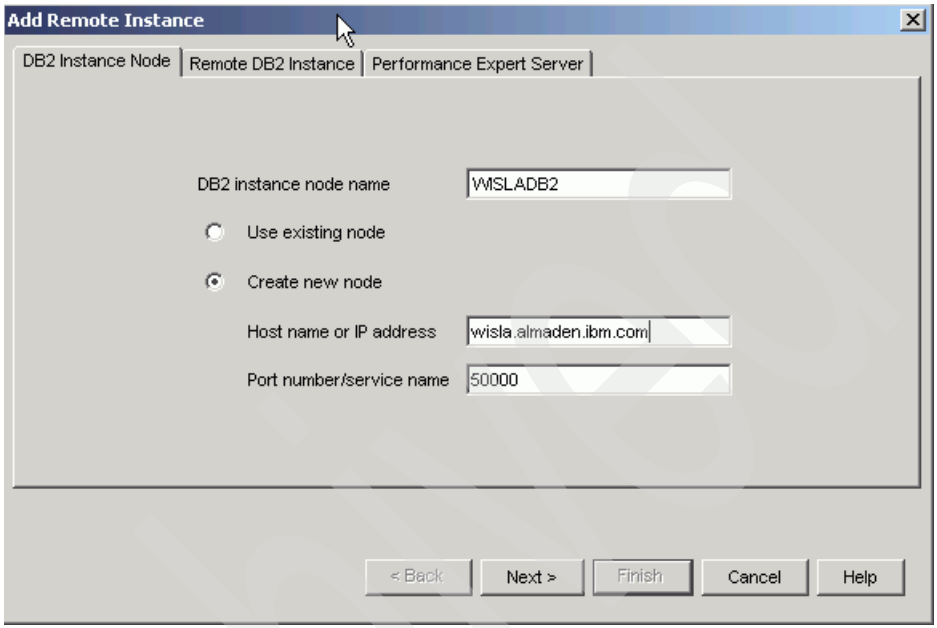


Figure 3-5 Add Remote Instance window - DB2 Instance Node tab

Next, we enter the appropriate information from the worksheet in the Remote DB2 Instance tab fields, as shown in Figure 3-6 on page 65, and click **OK**.

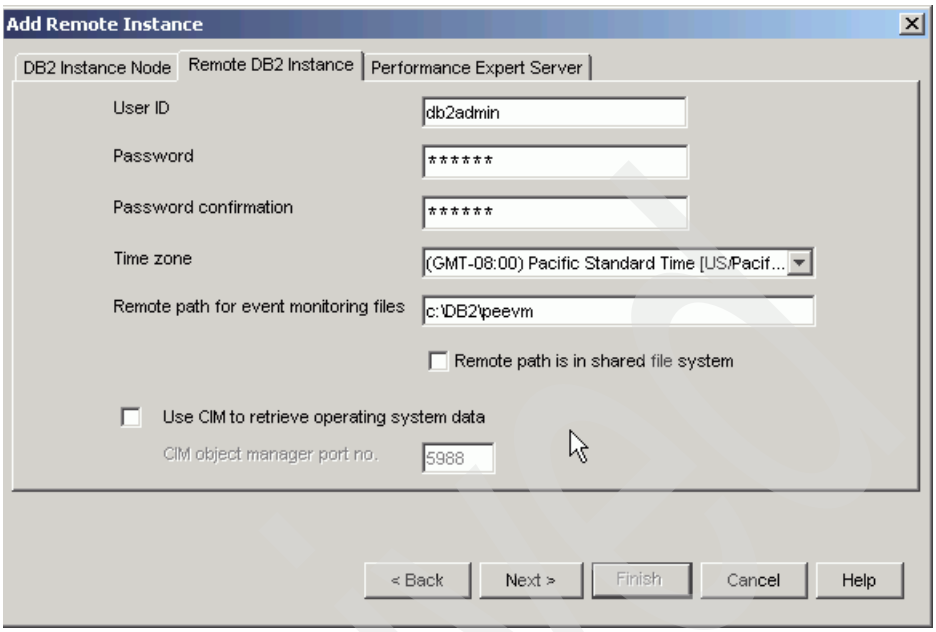


Figure 3-6 Add Remote Instance window - Remote DB2 Instance tab

We have chosen not to use a shared file system for event monitoring files, so we leave the “Remote path is in shared file system” check box unchecked. We also copy the DLL file fpeevm.dll from C:\Program Files\IBM\DB2 Performance Server v2\resources\win\V8\lib32 on server SIAM to the path we have specified on server WISLA.

We have furthermore chosen not to use CIM to retrieve operating system data, so we leave the corresponding check box unchecked too.

Next, we enter the information from the worksheet in the Performance Expert Server tab fields, as shown in Figure 3-7, and click **OK**.

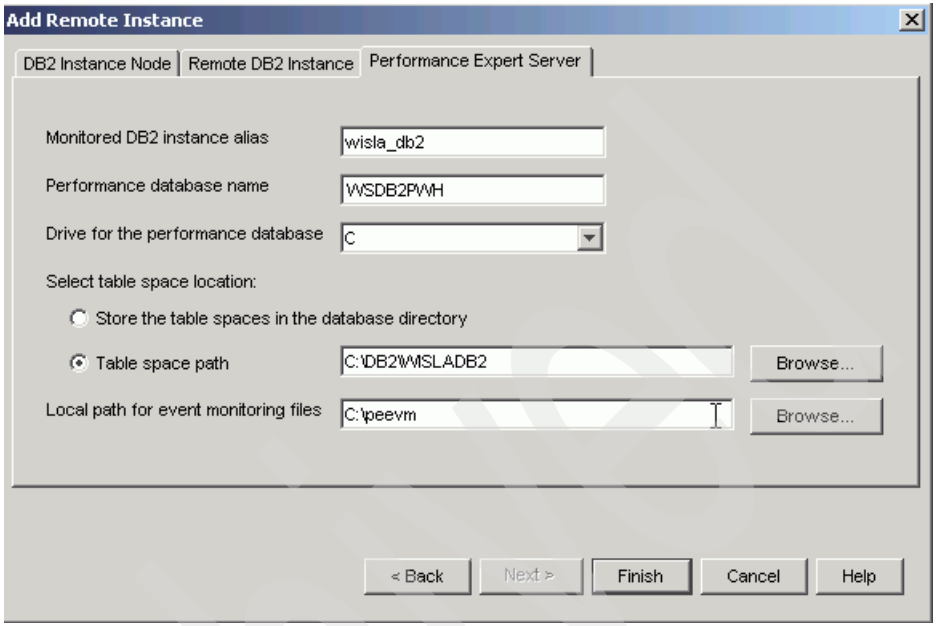


Figure 3-7 Add Remote Instance Window - Performance Expert Server tab

Note that no matter the folder for event monitoring files is shared or not, the remote and local paths entered do not need to match. The utility warns the user during entry of the “Tablespace path” or “Local path for event monitoring files” data if the path does not exist, and creates new directories, if necessary.

A success message and the newly added monitored DB2 instance now appears in the main window. We verify that the detail information on the right side of the window is correct, as seen in Figure 3-8 on page 67.

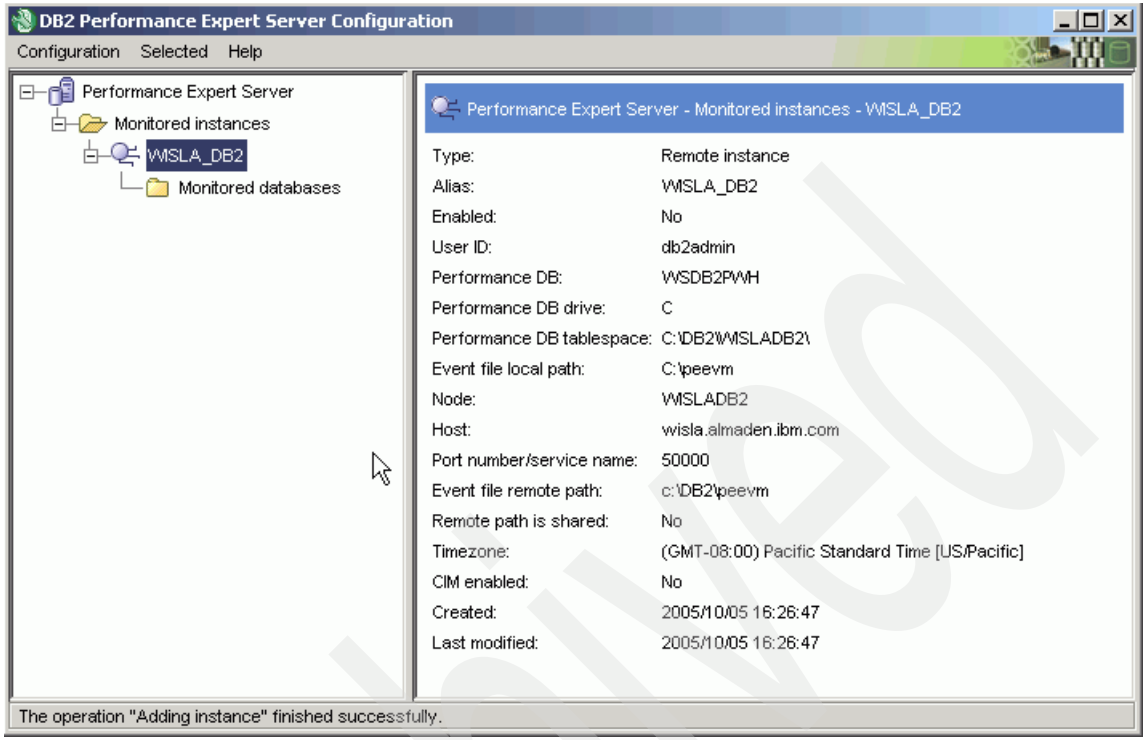


Figure 3-8 PE Server Configuration - instance added successfully

The utility requires that you add databases for monitoring before you enable the instance for monitoring or enable event monitoring. Next, we right-click the monitored databases folder and select **Add Database**. We enter the appropriate information from the worksheet, as shown in Figure 3-9, and click **OK**.

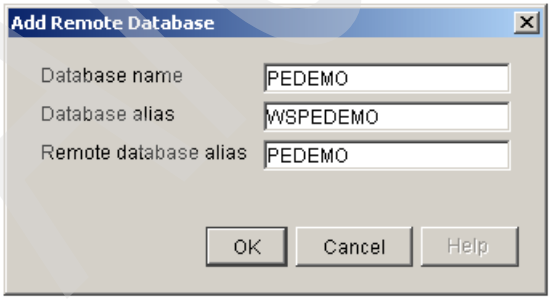


Figure 3-9 Add Remote Database window

A success message and the newly added database appears in the main window.

The order in which we enable the DB2 instance for monitoring and switch on event monitoring for the database does not matter. Next, we switch on event monitoring by right-clicking on the database name and selecting **Switch on event monitor**, as shown in Figure 3-10.

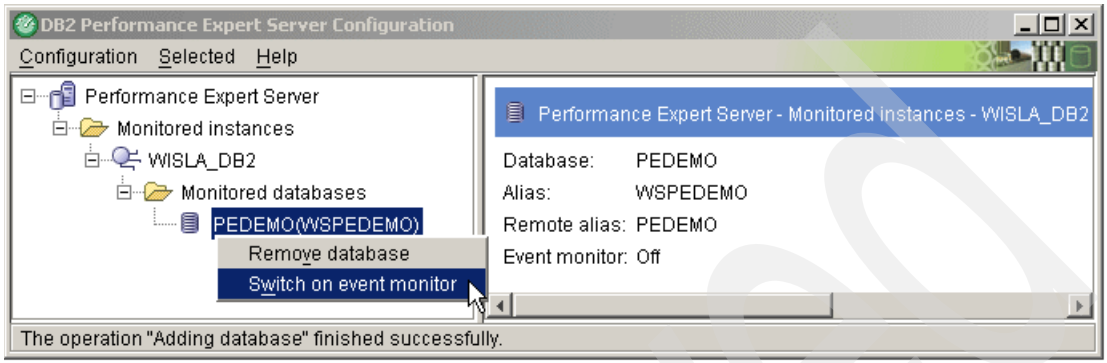


Figure 3-10 PE Server Configuration - database information

A success message appears, and we verify in the right pane that the event monitor status for the database changes to On. Finally, we enable the instance for monitoring by right-clicking on the instance name and selecting **Enable instance**. When we enable an instance for monitoring for the first time, the performance database with the configured name is created. Thus, enabling an instance for the first time may take a while. A success message appears, and we verify in the details pane that the instance is enabled.

After we exit the utility, we check and see that it has created the node and database catalog entries in the local DB2 instance used by PE Server, as shown in Example 3-1.

Example 3-1 Sample DB2 directory entries created by GUI configuration

```
Node 1 entry:

 Node name                      = WISLADB2
 Comment                        = Performance Expert Server
 Directory entry type           = LOCAL
 PRoTocol                       = TCPIP
 Hostname                       = wisla.almaden.ibm.com
 Service name                   = 50000

Database 1 entry:

 Database alias                       = DB2PM
 Database name                        = DB2PM
 Database drive                       = C:\DB2
```

```
 Database release level              = a.00
 Comment                             = Performance Expert Server
 Directory entry type                = Indirect
 Catalog database partition number   = 0

Database 2 entry:

 Database alias                      = WSDB2PWH
 Database name                       = WSDB2PWH
 Database drive                      = C:\DB2
 Database release level              = a.00
 Comment                             = Performance Expert Server
 Directory entry type                = Indirect
 Catalog database partition number   = 0

Database 3 entry:

 Database alias                      = WSPEDEMO
 Database name                       = PEDEMO
 Node name                           = WISLADB2
 Database release level              = a.00
 Comment                             = Performance Expert Server
 Directory entry type                = Remote
 Authentication                      = SERVER
 Catalog database partition number   = -1
```

AIX Server configuration example - peconfig

This example uses the UNIX values provided in Table 3-2 on page 55 in the example value column. As shown in the topology diagram in Figure 3-11, the PE Server and the monitored DB2 instance are on separate servers, and a shared directory is used for event monitoring information.

Run **peconfig.bat -console** from the bin subdirectory of the installation directory for running in interactive mode.

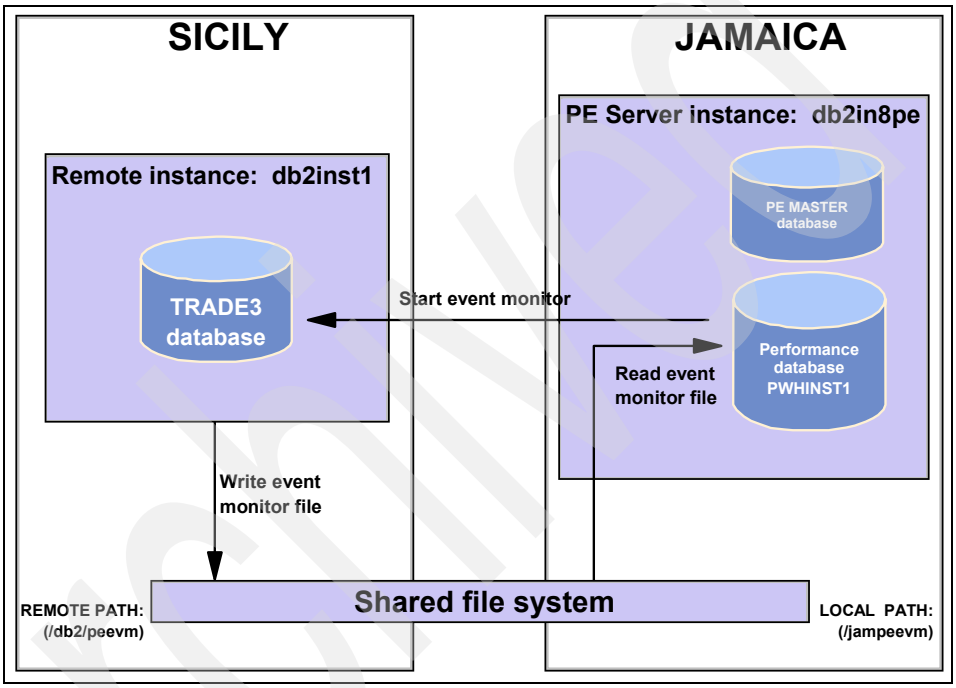


Figure 3-11 Topology for PE command-line configuration example

Example 3-2 contains a transcript of the peconfig session in which we performed the configuration. We executed the **list** command after several commands to show the intermediate and final results.

Example 3-2 Transcript from peconfig utility

```
peconfig => addreminst

You now register a remote DB2 instance to be monitored. The remote DB2 instance to be monitored
is different to the DB2 instance on which Performance Expert Server runs.
The user login name for the remote DB2 instance must have DB2 administrator rights on the
remote DB2 instance to register it.
```

```
Within this command you can type:
CANCEL      : Cancels the command and returns to the configuration tool.
Press Enter : Keeps the offered default value [default].

Enter a name of your choice that is used as a node name for the DB2 instance that is to be
monitored.

ADDREMINST - DB2 instance node name [NODE0001] => SICILY1

Enter the host name or IP address of the remote DB2 instance that is to be monitored.

ADDREMINST - Host name or IP address [localhost] => sicily.almaden.ibm.com

Enter the port number of the remote DB2 instance you want to monitor.
You can change the value after you register the DB2 instance by using CHANGE.

ADDREMINST - Port number [50000] => 60000

Enter an alias name of your choice for the remote DB2 instance that is to be monitored. This
alias name is also used by Performance Expert Client.

ADDREMINST - Monitored DB2 instance alias [sicily.almaden.ibm.com_60000_in] => SICILY1

Enter the user name for the DB2 instance to be monitored. This user name must have
administrator rights for the DB2 instance.
You can change the value after you register the DB2 instance by using CHANGE.

ADDREMINST - User login name [db2in8pe] => db2inst1
The default value is used.

Enter the user password for the DB2 instance to be monitored.

ADDREMINST - User password =>
Repeat the user password for the DB2 instance.

ADDREMINST - User password =>

Enter a name of your choice for the performance database. Performance Expert Server stores the
data that is collected for each monitored DB2 instance in a separate performance database.

ADDREMINST - Performance database name [D6169289] => PWHINST1

Enter the path where the performance database will be created.

ADDREMINST - Path for the performance database [/home/db2in8pe] => /db2pe

Enter a path of your choice for the tablespace of the performance database 'PWHINST1'.

ADDREMINST - PWHINST1 tablespace path [/db2pe/db2in8pe/SICILY1/tablespace/] =>
```

```
/db2pe/db2in8pe/pwhinst1

The folder '/db2pe/db2in8pe/pwhinst1' does not exist.

ADDREMINST - Do you want to create a new folder '/db2pe/db2in8pe/pwhinst1' [Y/N] ? [Yes] => y

The time zone of the local Performance Expert Server instance is [America/Los_Angeles].

ADDREMINST - Is the time zone of the remote DB2 instance identical to the time zone of the
local DB2 instance on which Performance Expert Server runs (Y/N)? [Yes] => y

Performance Expert Server collects DB2 performance event monitoring (EVM) information and makes
it available for Exception Processing and the Performance Warehouse. To exchange EVM
information between the local and the remote system, you must do one of the following:
- Specify a shared file system.
- Install a shared library on the remote system.

ADDREMINST - Do you want to specify a shared file system (Y/N) ? [Yes] => y

Enter the access path from your local system to the shared file system. You can change the
value after you register the DB2 instance by using CHANGE.

ADDREMINST - Local access path to the shared file system for event monitoring data =>
/db2pe/peevm

Enter the access path from the remote system to the shared file system.
You can change the value after you register the DB2 instance by using CHANGE.

ADDREMINST - Remote access path to the shared file system for event monitoring data =>
/jampeevm

To retrieve performance data of the monitored operating system, you must specify the port
number to access the Common Information Model Object Manager (CIMOM). Note: Not all monitored
platforms are currently supported for CIM based monitoring. Refer to documentation for
supported platforms.

ADDREMINST - Do you want to configure access to CIMOM (Y/N) ? [No] => y

ADDREMINST - Specify the port number to be used to access the CIMOM [5988] => 5988

You have successfully configured a DB2 instance to be monitored by Performance Expert Server.
You now need to add DB2 databases to be monitored by using ADDDB.

Use the ENABLE command to enable the Perfomance Expert Server DB2 instance for Performance
Expert Server.

peconfig => list
```

```
------------------------------------------------------------------------

Instance ID : 1
-----------------
   Enabled : No
   Monitored Instance Alias : SICILY1
   Node, Host, Port : SICILY1, sicily.almaden.ibm.com, 60000
   No databases are registered.

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

peconfig => adddb 1 sictrd3 trade3db trade3db

Testing the connection to the 'SICTRD3' database alias...
The monitored database is successfully added.

You have successfully added the database to be monitored by Performance Expert Server.
If the DB2 instance is not enabled use the ENABLE command followed by the number of the DB2
instance to enable the DB2 instance and its DB2 databases for monitoring.
Start or restart Performance Expert Server so that performance data for enabled databases are
collected.
By default, all configured databases are not monitored when Performance Expert Server is
started. You can control the databases that you want to monitor by using the EVMON and EVMOFF
commands.

peconfig => list

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

Instance ID : 1
-----------------
   Enabled : No
   Monitored Instance Alias : SICILY1
   Node, Host, Port : SICILY1, sicily.almaden.ibm.com, 60000
   Database, remote alias, local alias, EVM: TRADE3DB, TRADE3DB, SICTRD3, No

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

peconfig => enable 1

Before the DB2 instance can be enabled for monitoring, the DB2 instance 'db2in8pe' on which
Performance Expert Server runs must be restarted.

ENABLE - Do you want to continue (Y/N) ? [Yes] => y

Performance Expert Server now creates the performance database for the 'SICILY1' DB2 instance
and sets configuration parameters.
```

```
Creating the 'PWHINST1' database in the DB2 instance 'db2in8pe' on which Performance Expert
Server DB2 runs.
This might take a while. [      ]
Updating parameters for 'PWHINST1' database... [      ]
Creating buffer pools...
Creating tablespaces...

The DB2 instance 'SICILY1' is successfully enabled for monitoring.
Start or restart Performance Expert Server so that performance data for all enabled DB2
instances are collected.

peconfig => list

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

Instance ID : 1
-----------------
   Enabled : Yes
   Monitored Instance Alias : SICILY1
   Node, Host, Port : SICILY1, sicily.almaden.ibm.com, 60000
   Database, remote alias, local alias, EVM: TRADE3DB, TRADE3DB, SICTRD3, No

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

peconfig => evmon 1 sictrd3
The database is successfully updated.

peconfig => list

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

Instance ID : 1
-----------------
   Enabled : Yes
   Monitored Instance Alias : SICILY1
   Node, Host, Port : SICILY1, sicily.almaden.ibm.com, 60000
   Database, remote alias, local alias, EVM: TRADE3DB, TRADE3DB, SICTRD3, Yes
------------------------------------------------------------------------

peconfig => list 1

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

DB2 instance ID = 1
   Active DB2 instance = Yes
   Monitored instance alias = SICILY1
   User login name = db2inst1
   Node name = SICILY1
   Host name = sicily.almaden.ibm.com
```

```
   Port number = 60000
   Performance database = PWHINST1
   Performance database path = /db2pe
   Performance db tablespace = /db2pe/db2in8pe/db2inst1/
   Time zone = America/Los_Angeles
   Shared event monitoring path = Yes
   Event monitoring local path = /db2pe/peevm/
   Event monitoring remote path = /jampeevm
   CIM Object Manager enabled = Yes
   CIM Object Manager port number = 5988
   Creation time = 2005-10-06 08:50:02.509643
   Last modification time = 2005-10-28 19:26:27.304438
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
Registered databases:
   Local alias=SICTRD3; Name =TRADE3DB; Remote alias=TRADE3DB; EVM=Yes;

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

peconfig => exit
```

We check and see that the utility has created the node and database catalog entries in the local DB2 instance used by PE Server, as shown in Example 3-3.

Example 3-3 Sample DB2 directory entries created by command-line configuration

```
Node 1 entry:

 Node name                      = SICILY1
 Comment                        = Performance Expert Server
 Directory entry type           = LOCAL
 PRoTocol                       = TCPIP
 Hostname                       = sicily.almaden.ibm.com
 Service name                   = 60000

Database 1 entry:

 Database alias                       = DB2PM
 Database name                        = DB2PM
 Local database directory             = /db2pe
 Database release level               = a.00
 Comment                              = Performance Expert Server
 Directory entry type                 = Indirect
 Catalog database partition number    = 0

Database 2 entry:

 Database alias                       = PWHINST1
 Database name                        = PWHINST1
 Local database directory             = /db2pe
```

```
 Database release level                = a.00
 Comment                               = Performance Expert Server
 Directory entry type                  = Indirect
 Catalog database partition number     = 0

Database 3 entry:

 Database alias                        = SICTRD3
 Database name                         = TRADE3DB
 Node name                             = SICILY1
 Database release level                = a.00
 Comment                               = Performance Expert Server
 Directory entry type                  = Remote
 Authentication                        = SERVER
 Catalog database partition number     = -1
```

WIN Server example of a silent configuration for a remote AIX instance

This example uses the UNIX values provided in Table 3-2 on page 55 in the example value column. As shown in the topology diagram in Figure 3-12 on page 77, the PE Server is on a Windows platform (SIAM) and the monitored DB2 instance is on an AIX platform (SICILY). Due to the heterogeneous environment, no shared directory can be used for event monitoring information, but exchanging data using database objects is used.

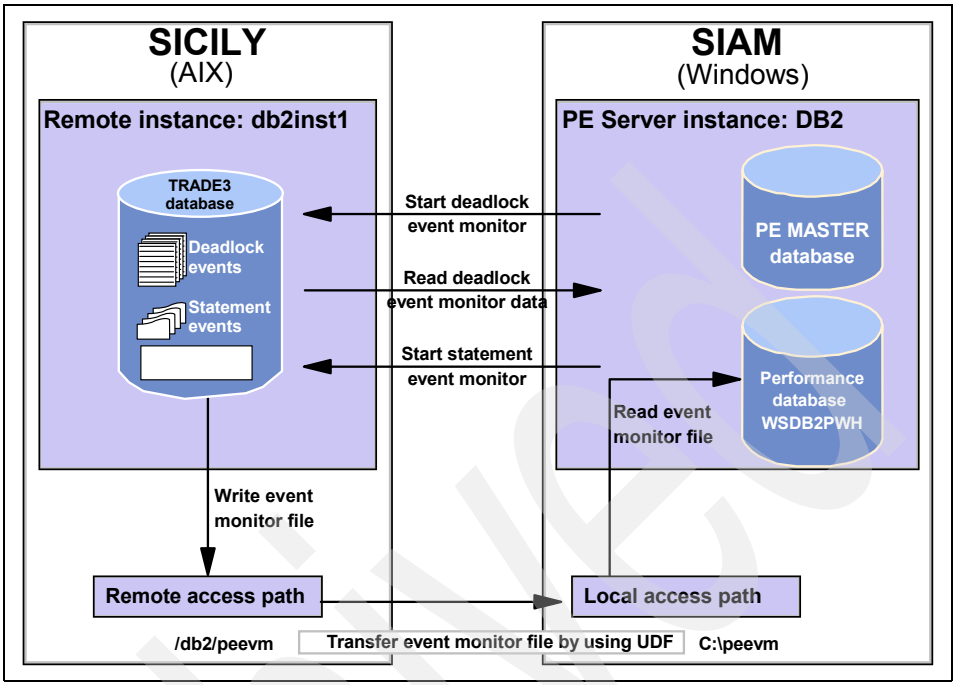


Figure 3-12 Topology for PE silent configuration example

Before we can start silent configuration, we need to prepare the response file that is used as input to **peconfig**. For a description of the complete list of parameters and the syntax of the response file, please refer to the sample response file peresp.txt, which you can find in the bin subdirectory of the installation directory. We use the sample response file as a template to create a new response file called myperesp.txt.

For the topology shown in Figure 3-12, we can prepare the response file myperesp.txt, as shown in Example 3-4 on page 78. Note that the lines starting with a "#" are comment lines and the default values are taken for those parameters that are omitted. Please see the sample response file peresp.txt for more details.

In our example, the first line defines an identifier, i1, which is used to uniquely mark the parameters for a dedicated instance definition. We do not specify a node name, but a host name and a port number. The **peconfig** command will create the node name node0001 itself (or node0002 and so on if node0001 already exists). If we had the host SICICLY already cataloged, we could explicitly use this node name instead and would omit the host name and port number. Both host name and port number would then be taken from the catalog entry. Subsequently, we define the databases to be monitored. Again, the database

gets a unique identifier first. In our example, we explicitly define database TRADE3. If we would omit the database section, **peconfig** would register all databases for monitoring that are cataloged for that node.

Example 3-4 Response file for a silent configuration of a remote instance

```
instance                          = i1
i1.local_instance                 = N
i1.instance_alias                 = SICILY
# i1.node_name                       = node0001
i1.host_name                      = sicily.almaden.ibm.com
i1.port_number_service_name          = 60000
i1.user_login_name                = db2inst1
i1.user_password                  = <...password... >
i1.performance_database_name      = WSDB2PWH
# i1.performance_database_path    = D
# i1.table_space_path                = d:\peserver\ts\
# i1.time_zone_code                  = 143
i1.shared_evm_path_flag           = N
i1.local_evm_access_path          = c:\peevm
i1.remote_evm_access_path         = /db2/peevm
i1.cim_object_manager_enabled     = Y

i1.database                       = d1
i1.d1.database_name               = TRADE3
i1.d1.local_database_alias        = TRDE3SIC
i1.d1.remote_database_alias       = TRADE3
i1.d1.event_monitor_flag          = Y
```

Now let us start the silent configuration by executing the following command from the bin subdirectory of the installation path:

```
peconfig -silent myperesp.txt
```

A command prompt will pop up as shown in Example 3-5.

Example 3-5 Transcript of peconfig output of a silent configuration

```
IBM(c) DB2 Performance Expert Server.

Use this configuration tool to set up DB2 Instances to be monitored by
Performance Expert Server.

Entering Silent configuration mode...
Refer to the 'C:\Program Files\IBM\IBM DB2 Performance Expert Server
v2\instances\DB2\pesilent.trc log file.

Configuring the DB2 instance 'SICILY1'...
```

```
Enabling the DB2 instance 'SICILY1', this might take a while...
Done.

Silent configuration successfully finished.


Press any key to continue . . . _
```

When the configuration was successfully completed, the instance is registered and enabled for monitoring.

Note: For Performance Expert Server V2.2, monitoring the instance would start when the PE server is restarted. For V2.2 Fix Pack 1, monitoring will start immediately, if the PE server is active while we run the silent configuration.

3.1.5 New PE server configuration features of V2.2 Fix Pack 1

In the following section, we describe the new configuration features and the differences from the examples shown in 3.1.4, “Server configuration examples” on page 61.

In general, with V2.2 Fix Pack 1, the configuration tools:

- Show advanced status information.
  - The current status of the PE server.
  - The monitoring status of the instances.
  - Pending configuration changes.
- Allow dynamic monitoring.
  - Stop monitoring of a registered instance.
  - Start monitoring of a registered instance.
  - Change an instance’s configuration and activate the changes.
  - Drop an instance registration and terminate monitoring.
  - The dynamic monitoring means that the tasks listed can be performed while the PE Server is running.

Advanced status information and dynamic monitoring is supported in both the command-line interactive mode and the GUI interactive mode. Silent configuration mode now also supports dynamically starting monitoring.

Windows GUI Server configuration example

The following example describes the scenario where Performance Expert Server has been started and is monitoring the instance WISLA_DB2, which we configured as described in the section above.

When you start the configuration GUI in V2.2 Fix Pack 1, the windows shown in Figure 3-13 will be present.

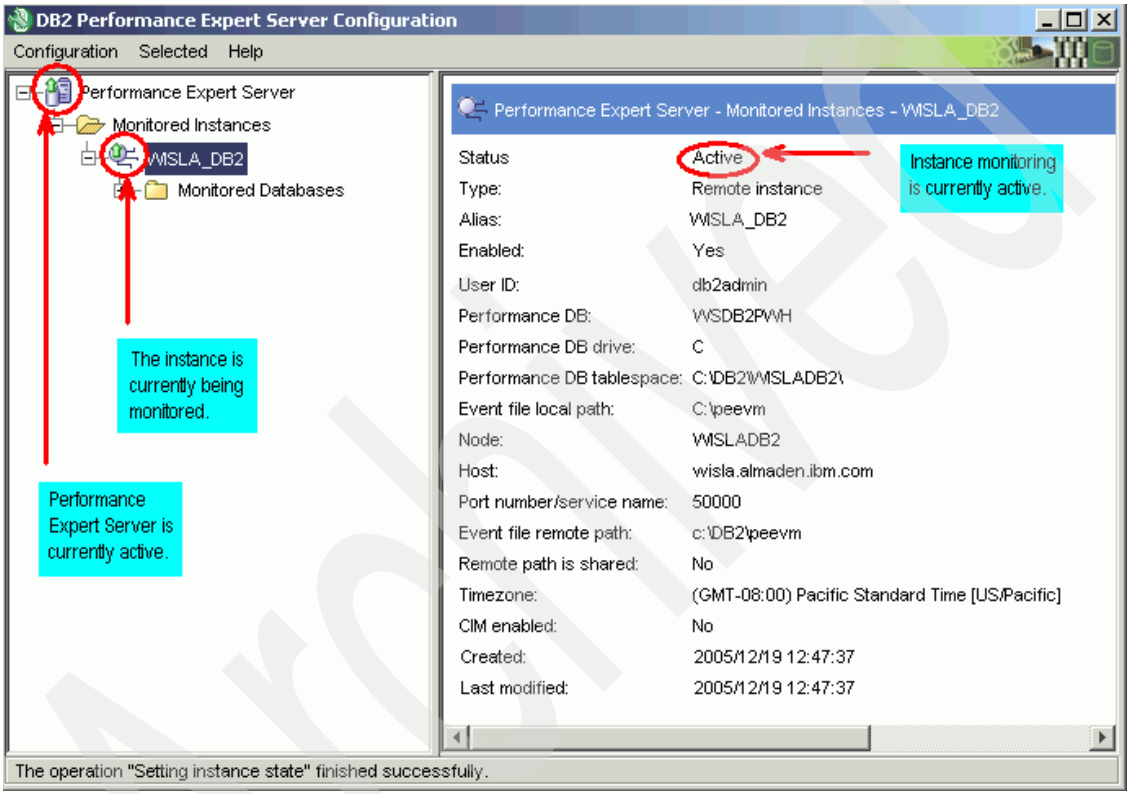


Figure 3-13 Performance Expert Server Configuration - all systems active

Advanced status information

There are several new graphical items.

To the left of the window, you can see the current status of both the Performance Expert Server and the monitoring status of the configured instances. The green arrow in the icon to the left of the “Performance Expert Server” tree item shows that the PE server is currently active. The green arrow in the icon to the left of the instance name WISLA_DB2 means that the instance is currently being monitored

by the PE server. If instance monitoring stops for any reason or the PE server terminates, the color of the arrows changes automatically to red.

In the details pane of the configured instance to the right of the window, there is a new field called “Status”. This field shows “active” if the selected instance is currently being monitored by Performance Expert Server; otherwise, it shows “inactive”.

You may stop and start Performance Expert Server to see the icons change from green to red and back to green when the server is up and running again.

Dynamic monitoring

Now we discuss the effect of disabling and enabling an instance for monitoring.

- Stop monitoring of a registered instance.

  Assume we have the scenario described above, Performance Expert Server is actively monitoring DB2 instance WISLA_DB2. For some reason, it becomes necessary to stop monitoring of instance WISLA_DB2 (but we do not want to interrupt monitoring of other instances we had potentially configured for monitoring). We can achieve this by either:

  – Selecting **WISLA_DB2**, open the **Selected** menu, and select **Disable monitoring**.

  – Right-clicking **WISLA_DB2** and select **Disable monitoring**.

The icon to the left of the instance name WISLA_DB2 changes and now shows a red arrow indicating that the instance is currently no longer monitored (see Figure 3-14). Furthermore, the Status field to the right switches to inactive. Performance Expert Server is still active.

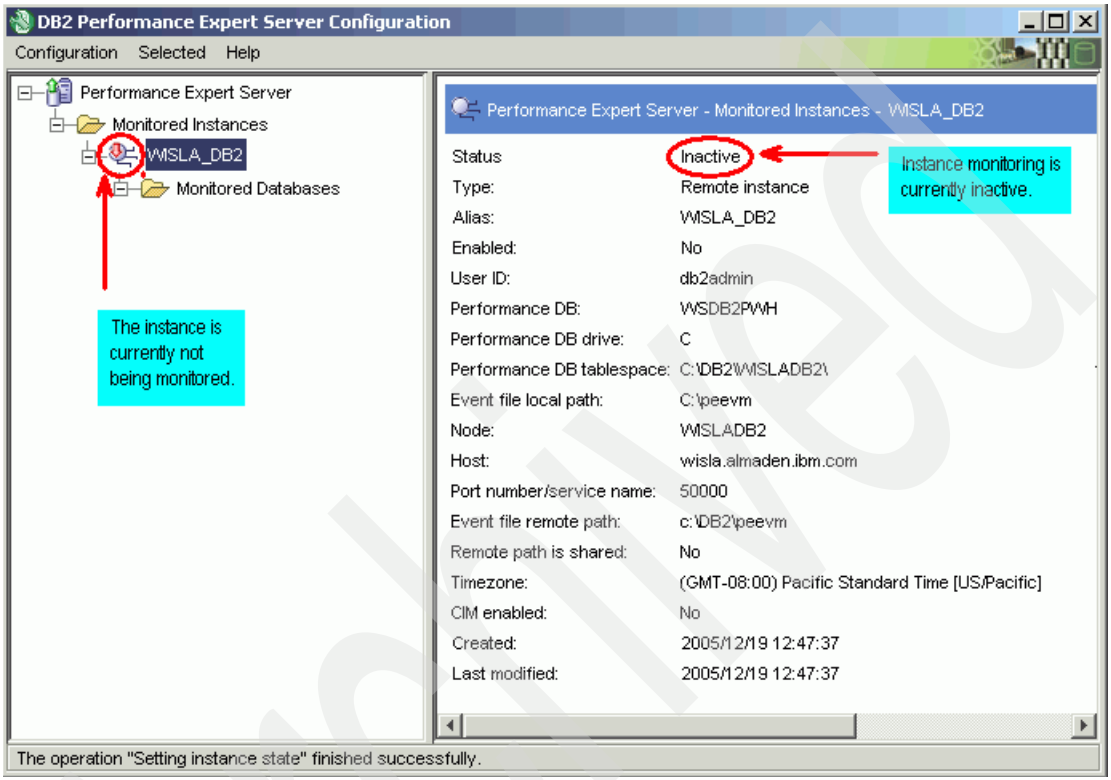


Figure 3-14 Performance Expert Server Configuration - instance monitoring inactive

See also the Performance Expert Server console log, which shows that the server's monitoring components, like Performance Warehouse (PWH), snapshot processing (SNAP), exception processing (EXCP), and so on, have been terminated for that instance.

- Start monitoring of a registered instance.

  You may decide at any time to start monitoring of the selected instance again. Just do either of the following:

  - Select instance **WISLA_DB2**, open the **Selected** menu, and select **Enable monitoring**.
  - Right-click **WISLA_DB2** and select **Enable monitoring**.

On the Performance Expert Server console, you will recognize that the monitoring components are started again for that instance. The status field on the configuration tool window switches back to Active and the icon arrow next to the instance name becomes green.

- Change an instance's configuration and activate the changes.

  Changing the configuration of an instance may become necessary for several reasons. For example, you might be forced to change the password of the user that was configured for accessing the instance, you may want to add or remove some databases to or from the list of monitored databases, you may need to change the event monitor settings of one or more databases, you decide to use a shared file system now for exchanging event monitor data instead of database objects, and so on.

  Once you changed the configuration, you will see new status information in the configuration tool window. This new information is shown if the PE server is actively monitoring the instance that was changed. It shows that configuration changes are pending that have not been activated yet (see Figure 3-15).

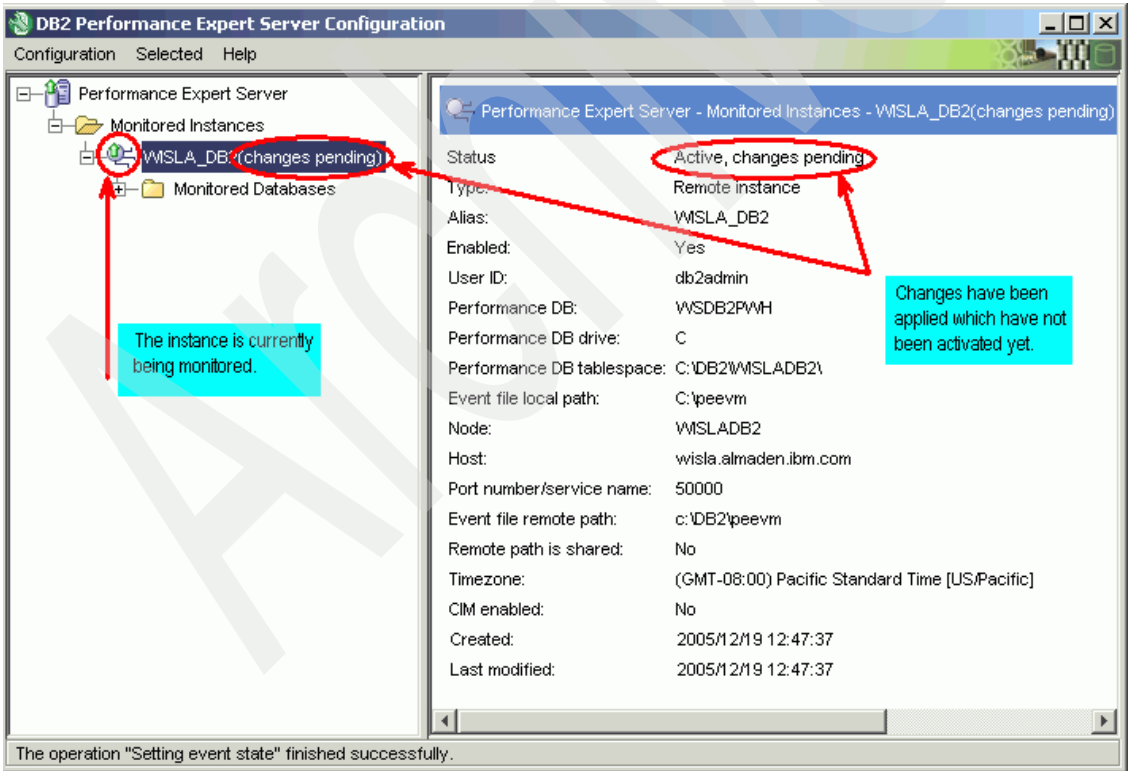


Figure 3-15 Performance Expert Server Configuration - changes pending

Before PE server V2.2 Fix Pack 1, you had to shut down the PE server and start it again to activate the changes. This stopped monitoring for all configured instances, not just the changed one.

To activate the changes for Fix Pack 1, do either of the following:

- Select instance **WISLA_DB2**, open the **Selected** menu, and select **Restart monitoring**'.
- Right-click instance **WISLA_DB2** and select **Restart monitoring**.

The status information "changes pending" disappears. If you see the PE server console log, you will see that monitoring of the instance is stopped and immediately started again. Thus, restart monitoring means disable and subsequently enable monitoring in one step.

- Drop an instance registration and terminate monitoring.

  You can now drop an instance registration while it is currently being monitored. It is no longer necessary to stop Performance Expert Server to stop monitoring the instance and then restart Performance Expert Server in order to proceed monitoring the other instances (see Figure 3-26 on page 101).

.

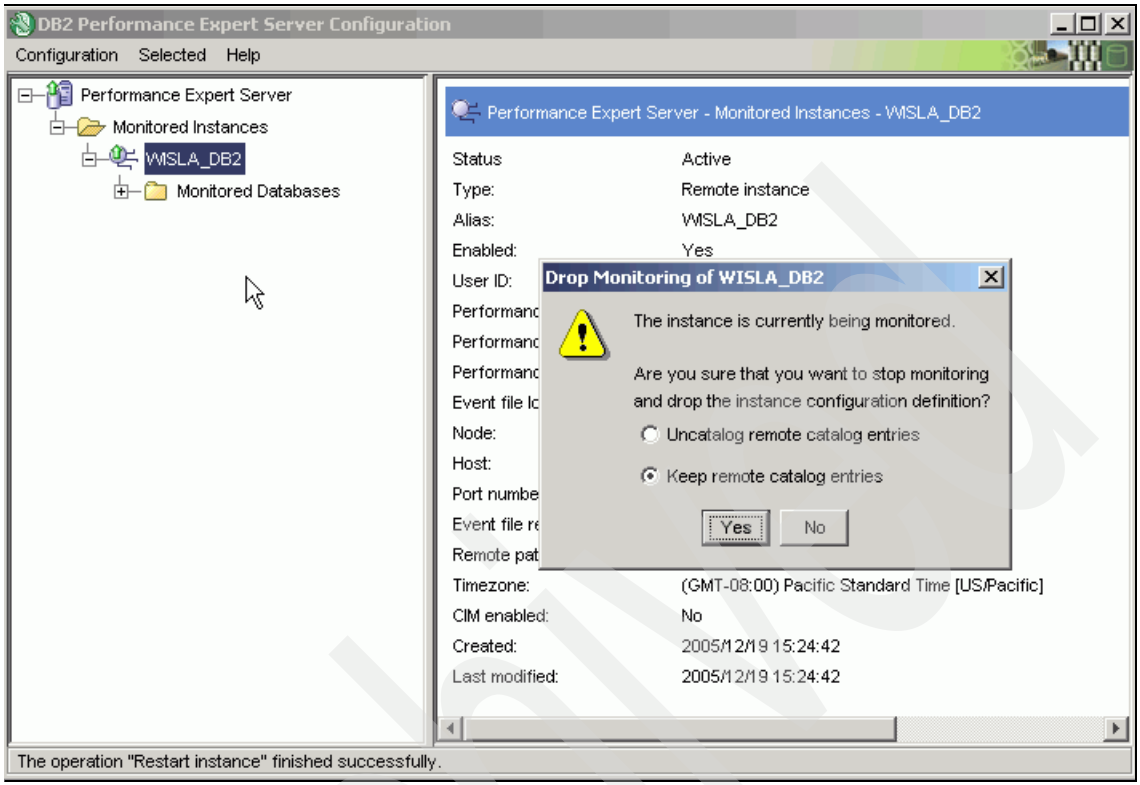


Figure 3-16 Performance Expert Server Configuration - drop registration of an actively monitored instance

To drop the instance registration, do either of the following:

- Select instance **WISLA_DB2**, open the **Selected** menu, and select **Drop monitoring**.
- Right-click instance **WISLA_DB2** and select **Drop monitoring**.

Since the instance is currently being monitored, a message box pops up that reminds you that monitoring is active and asks you for confirmation to proceed. Furthermore, it asks you what to do with the catalog entries if you confirm to proceed; whether to drop these entries from the catalog to or to keep them.

If you confirm to proceed, monitoring is stopped immediately for this instance, and the instance registration and the performance database are dropped. See also the Performance Expert Server console log, which shows that the server's monitoring components are terminated for that instance.

AIX Server command-line configuration example

The following paragraphs describe the differences of V2.2 Fix Pack 1 from the configuration of an instance via the command-line tool; see the full example described in 3.1.4, “Server configuration examples” on page 61.

To run in interactive command mode, run **peconfig -console** from the bin subdirectory of the installation directory. Note that changes are marked with bold highlighting, as shown in Figure 3-17.

```
-
------------------------------------------------------------------------------------------------
--

peconfig => list

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

Instance ID = 1
-----------------
    Enabled = No
    Status = Inactive
    CIM Object Manager enabled = No
    Monitored Instance Alias = SICILY1
    Node, Host, Port/Service name = SICILY1, sicily.almaden.ibm.com, 60000
     Database, remote alias, local alias, event monitoring = TRADE3DB, TRADE3DB, SICTRD3, No
 ------------------------------------------------------------------------------------------------
```

Figure 3-17 LIST command of V2.2 Fix Pack 1 configuration utility

The LIST command now offers additional information for each registered instance:

- The Status field shows whether the instance is currently monitored by Performance Expert Server (active) or not (inactive).

In our example, instance SICILY1 is currently not being monitored and still not enabled for monitoring. To enable it, enter the ENABLE command followed by the instance ID, as shown in Example 3-6.

Example 3-6 ENABLE command of V2.2 Fix Pack 1 configuration utility

```
peconfig => enable 1

Performance Expert Server now creates the performance database for the 'SICILY1' DB2 instance
and sets configuration parameters.

Creating the 'PWHINST1' database in the DB2 instance 'db2in8pe' on which DB2 Performance
Expert Server runs.
This might take a while. [    > ]
Updating parameters for 'PWHINST1' database... [    > ]
Creating buffer pools...
Creating tablespaces...
Starting monitoring of 'SICILY1' DB2 instance...
The DB2 instance is successfully enabled for monitoring.
```

```
peconfig => list 1

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

DB2 instance ID = 1
   Enabled DB2 instance = Yes
   Status = Active
   Monitored instance alias = SICILY1
   User login name = db2inst1
   Node name = SICILY1
   Host name = sicily.almaden.ibm.com
   Port number/Service name = 60000
   Performance database = PWHINST1
   Performance database path = /db2pe
   Performance db tablespace = /db2pe/db2in8pe/db2inst1/
   Time zone = America/Los Angeles
   Shared event monitoring path = Yes
   Event monitoring local path = /db2pe/peevm
   Event monitoring remote path = /jampeevm
   CIM Object Manager enabled = No
   Creation time = 2005-12-20 13:30:38.27251
   Last modification time = 2005-12-20 13:33:22.586755
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
Registered databases:
   Local alias=SICTRD3; Name =TRADE3DB; Remote alias=TRADE3DB; EVM=Yes

------------------------------------------------------------------------------
```

In PE V2.2, you are no longer required by the configuration tool to restart the instance db2in8pe. You can enable instances without interrupting the PE server. Furthermore, when enabling the instance, the configuration tool automatically initiates a start of monitoring the instance by the PE server.

After enabling, you might want to see the current status of the monitoring of the instance using the LIST command. Enter the LIST command to see a summary of all instance definitions or the LIST command followed by the instance ID to see the detailed configuration information.

Note that you might see a status of "'Inactive" while "Enabled DB2 instance" shows "Yes". Remember that with the first enabling of an instance the performance database is created. This takes a while. When this setup has completed (see the "alive" message on the Performance Expert Server console log), then the status changes to "Active".

Now change the configuration. For example, switch on event monitoring for a database and see the new status information by using the LIST command. Then use the RESTART command to immediately activate the changes (see Example 3-7).

Example 3-7 Change configuration using the command in the V2.2 Fix Pack 1 configuration utility

```
peconfig => evmon 1 samberg

The database is successfully updated.

If Performance Expert Server is started, use 'RESTART' command to submit changes to Performance
Expert Server and start to collect performance data for the monitored DB2 instance and
registered databases.

peconfig => list

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

Instance ID = 1
-----------------
   Enabled = Yes
   Status = Active, changes pending
   CIM Object Manager enabled = No
   Monitored Instance Alias = SICILY1
   Node, Host, Port/Service name = SICILY1, sicily.almaden.ibm.com, 60000
   Database, remote alias, local alias, EVM = TRADE3DB, TRADE3DB, SICTRD3, Yes

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

peconfig => restart 1

Restarting monitoring of 'SICILY1' DB2 instance...
The DB2 instance is successfully restarted.
```

Check also the PE server console log to see that monitoring is stopped for the instance and immediately started again with the configuration changes done.

When you no longer require the instance to be monitored and you want to unregister the instance that is currently being monitored, you can do this now without stopping the PE server. Enter the DROPINST command followed by the instance ID. Monitoring is automatically stopped before the instance is unregistered, as shown in Example 3-8 on page 89.

Example 3-8 DROPINST command of V2.2 Fix Pack 1 configuration utility

```
peconfig => dropinst 1

Performance Expert Server now deregisters the 'SICILIY1' monitored instance.

DROPINST - Are you sure that you want to stop monitoring and deregister the instance? [Yes] =>
The default value is used.

DROPINST - Do you want to keep the DB2 catalog entries (Y/N) ? [Yes] => no
Stopping monitoring of 'SICILY1' DB2 instance...
Deregistering remote 'SICILY1' DB2 instance...
Deregistering monitored databases for the 'SICILY1' instance...
Database 'SICTRD3' is successfully deregistered.
Deregistering the node 'SICILY1', with 'sicily.almaden.ibm.com' host name and '60000' port
number/service name.
Dropping the 'PWHINST1' database on the 'db2in8pe' DB2 instance on which Performance Expert
server runs...
The DB2 instance is successfully deregistered.
```

3.1.6 Client setup

After the PE Client installation, you must define the DB2 instances to be monitored from the client before you can view performance data about them. This section discusses starting PE Client and adding DB2 instances to monitor, and it also lists some possible error messages you may receive while adding instances to monitor.

Starting PE Client

To start PE Client:

- In Windows, use one of the following methods:
  - Double-click the IBM DB2 Performance Expert V2 icon on the desktop.
  - Click **Start** → **Programs** → **IBM DB2 Performance Expert V2** → **IBM DB2 Performance Expert V2**.
  - Run **db2pe.bat** from the bin subdirectory of the installation directory (this is the script run by each of the two methods above).
- In UNIX, run **db2pe** from the bin subdirectory of the installation directory.

After it has been configured, PE Client can also be accessed through the DB2 Control Center using one of the methods below. It will be launched if it is not already open:

- Click the DB2 Performance Expert icon on the DB2 Control Center toolbar (see Figure 3-18).

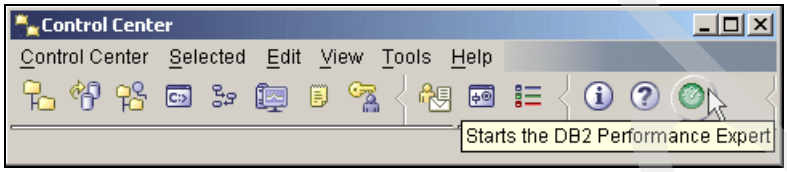


Figure 3-18 Launching PE Client from the DB2 Control Center

- Right-click a registered instance in the objects pane of the DB2 Control Center, select **DB2 Performance Expert**, and select the monitoring function you wish to open (see Figure 3-19 on page 91). You will be prompted to log on if you are not already connected to the selected instance.

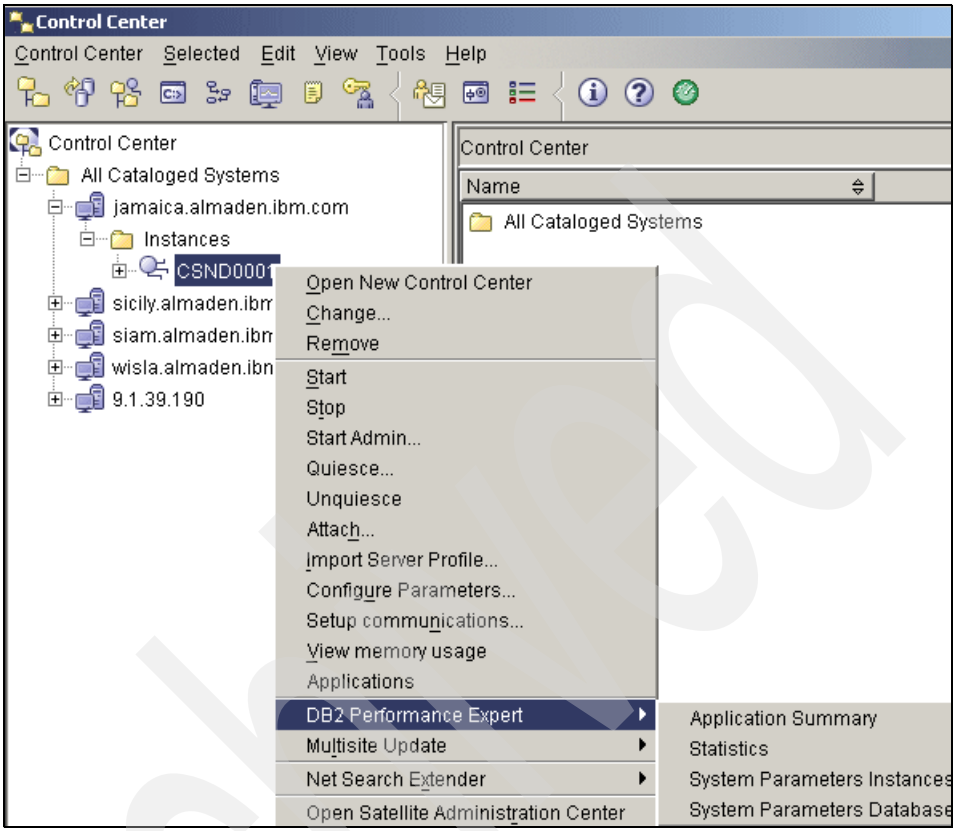


Figure 3-19 Launching a specific monitoring function from the DB2 Control Center

Note: If only the DB2 Runtime Client is installed, then these last two methods of opening PE Client will not be available, but all other PE Client functionality that does not require the use of DB2 GUI tools will be available.

When launching PE Client for the first time, you can choose whether to set the Extended design or the Classic design as the default format for the System Overview window, which is the main window in PE Client (see Figure 3-20). You can change this selection later by selecting **Extended design** or **Classic design** from the View menu of the System Overview window. The design that is selected each time PE Client is closed will be the design used the next time it is opened.

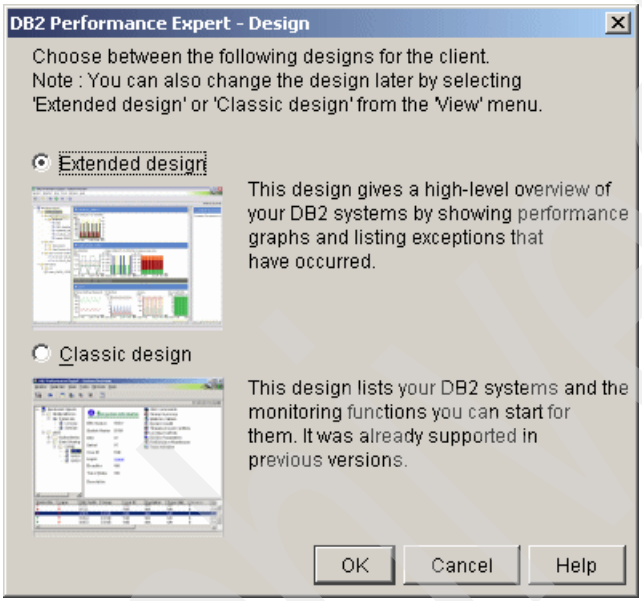


Figure 3-20 Design options for the System Overview window

The *extended* design (see Figure 3-42 on page 116 for an example) shows more detail about the instances being monitored. It shows graphs that provide a performance overview of the monitored instances to which you are logged on. These graphs result from data views that have been defined in the System Health window (see Chapter 4, “Features and functions - online and short-term monitoring” on page 143 for more details), and the two designs look similar until these data views have been defined. This design also has helpful tips and links for tasks within PE Client.

The *classic* design has a more simplified format in which the middle pane functions as a contents pane for the objects pane on the left. It includes instance information for the highlighted instance.

In both designs, the right pane contains a list of the 100 most recent event exceptions (if exception processing has been enabled). The bottom pane, which can be shown or hidden from the View menu, contains DB2 system status information for all monitored instances.

Defining DB2 instances to monitor

Each PE Server may monitor one or multiple DB2 instances, and each PE Client may connect to one or multiple PE Servers to access the performance data for these instances. You may add instances to monitor to PE Client either manually or by exporting them from another PE Client and then importing them. You do not need to configure monitoring in PE Client for individual databases; you will automatically have access to performance information for each database registered for monitoring on the instances you select.

Manually adding new DB2 instances to monitor

The DB2 instance on which PE Server runs must be started in order to successfully add DB2 instances for monitoring in PE Client. However, in order to log on to monitored instances in PE Client after adding them, the PE Server and the monitored DB2 instance must also be started. To add instances to monitor, you will need the information in Table 3-4. Later in this section, we will use the example values shown, which correspond to the Windows example values shown in the PE Server configuration worksheets (see Table 3-2 on page 55 and Table 3-3 on page 58) to demonstrate adding instances to monitor in PE Client.

Table 3-4 PE Client configuration worksheet

Parameter	Description	Example value
PE Server host name	The host name or IP address by which the PE Server may be accessed from the PE Client.	siam.almaden.ibm.com
PE Server port	The port number of the DB2 instance on which the PE Server runs. If Fix Pack 2 or later is installed, the port name may also be used.	50000
User ID / password	The ID must be a member of the group that was given access to PE Server when **`pecentralize`** was run on UNIX or when PE Server installation was performed on Windows. It may be an ID with administrative privileges on the DB2 instance on which the PE Server runs, or it may be a different ID.	PEuser

Parameter	Description	Example value
Monitored instance alias	This value defaults to the alias that was input when the PE Server was configured for monitoring. You may use the default or change it. Once this field has been changed, it will default to the new values when this screen is accessed again.This value will identify the monitored instance in the System Overview window. This alias will not appear in the local DB2 node directory.	WISLA_DB2
DB2 connection alias	A value of your choice to represent both the performance database and Performance Warehouse database of the monitored instance, which comprise tables in the same database. PE Client will use this value as the alias in a **catalog database** command. This value will be displayed in the Performance Warehouse window. This alias can be up to eight characters long and must not already be in use in the database directory. If the instance is already registered on this PE Client, then this field will already be completed.	WSDB2PWH

To find the port name, obtain the value of the SVCENAME parameter in the database manager configuration of the DB2 instance on which the PE Server runs. For example, execute the command **db2 get dbm cfg** and find the following parameter in the output:

```
TCP/IP Service name                          (SVCENAME) = DB2c_db2inst1
```

To find the port number corresponding to this service name, find the entry for this port name in the services file on the server on which that DB2 instance runs (for example, in /etc/services on AIX, or in C:\WINNT\system32\drivers\etc\services on Windows), as shown below:

```
DB2c_db2inst1 50000/tcp
```

Manually defining DB2 instances to monitor will create local node and database catalog entries in a DB2 instance on the PE Client workstation. If you have multiple DB2 instances installed on this workstation, you can specify the DB2 instance to be used (before opening PE Client) by adding the command **set DB2INSTANCE=*yourinstance*** in the file db2pe.bat (Windows) or .db2pe (UNIX) anywhere before the start call, as shown in bold in Example 3-9 on page 95.

Example 3-9 Sample db2pe.bat showing change to DB2 instance used by PE Client

```
@echo OFF
rem ****************************************************************************
rem *  Licensed Materials - Property of IBM                                     *
rem *  5724-F89 5724-F90                                                       *
rem *  (c) Copyright IBM Corporation 2002, 2004.                               *
rem *  All rights reserved.                                                    *
rem *                                                                          *
rem * US Government Users Restricted Rights - Use, duplication or             *
rem * disclosure restricted by GSA ADP Schedule Contract with                 *
rem * IBM Corporation.                                                        *
rem ****************************************************************************
rem
rem  Usage: db2pe
rem
rem ****************************************************************************
rem * The following line was added to change PE Client's default DB2 instance*
rem ****************************************************************************

set DB2INSTANCE=db2inst2

rem ****************************************************************************
rem * The following content has been customized by Setup                       *
rem ****************************************************************************
:custom
C:
cd C:\Program Files\IBM\IBM DB2 Performance Expert V2\bin
start "" "C:\Program Files\IBM\IBM DB2 Performance Expert V2\ibm
jre\bin\javaw.exe" -cp "C:\Program Files\IBM\IBM DB2 Performance Expert V2\ibm
jre;C:\Program
Files\IBM\SQLLIB\java\db2java.zip";db2pe.jar;images.jar;orgxml.jar;pehelp_en.zi
p;pehelp_ja.zip;pehelp_ko.zip;pehelp_tw.zip;..;.
com.ibm.db2pm.sysovw.main.SystemOverview

if errorlevel 0 goto end
echo.
pause

:end
```

PE Client manual configuration example

The following example uses the example values from the PE Client configuration worksheet in Table 3-4 on page 93. We perform the following steps on a Windows workstation to manually add DB2 instances to monitor:

1. First, we click **Monitor** → **New DB2 System** in the System Overview window.

2. On the Object page (see Figure 3-21), we select **DB2 on Multiplatforms** and then click **Next**.

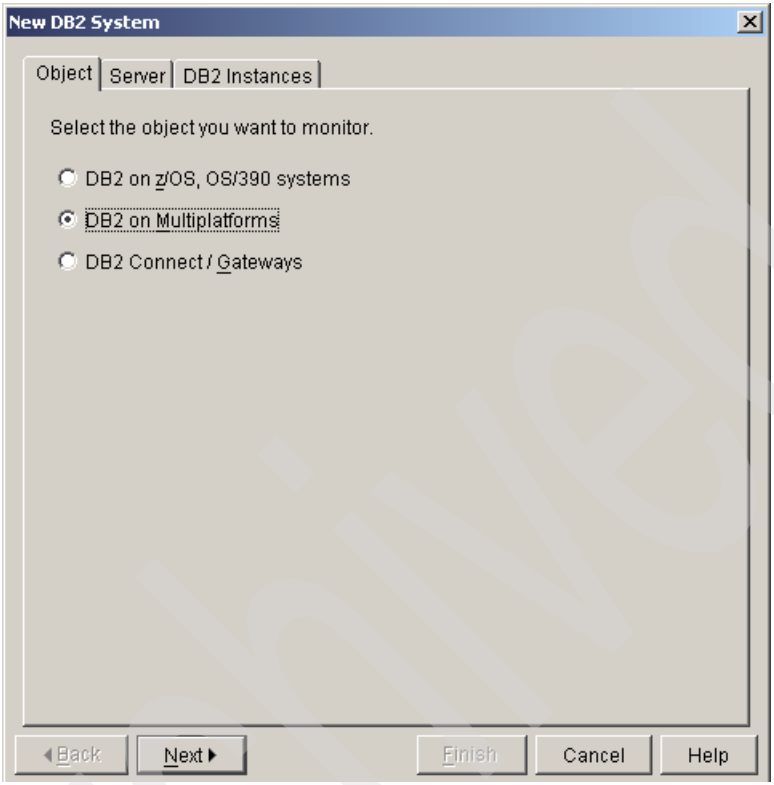


Figure 3-21 New DB2 System window - Object tab

Note: The options to monitor “DB2 on z/OS, OS/390® systems” or “DB2 Connect / Gateways” are not covered in this redbook.

3. On the Server page (see Figure 3-22 on page 97), we enter the host name and port of the PE Server and click **Next**.

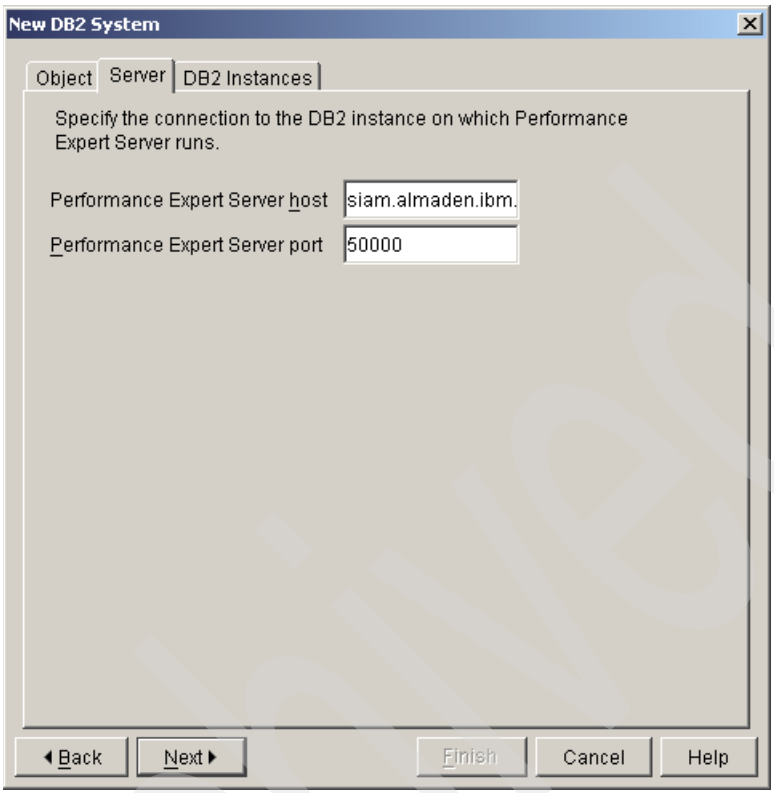


Figure 3-22 New DB2 System window - Server tab

4. On the DB2 Instances page, click the **Retrieve Monitored DB2 Instances** button.

> **Note:** If a node catalog entry already exists with a host name and service name that exactly match the host and port entered in the New DB2 System window, regardless of whether it was created by PE, then a new catalog entry will not be created. Also, each time you click the **Retrieve Monitored DB2 Instances** button after entering a valid or invalid new combination of host name and port that have not already been locally cataloged, a new entry is created in the DB2 node catalog of the DB2 instance on which the PE Client is running. You may wish to uncatalog any unused entries once you are sure they are not needed (by using the Configuration Assistant or the **`uncatalog node`** command).

5. When prompted (see Figure 3-23), we enter a user ID that belongs to the group with access to PE Server, enter the password, and click **OK**.

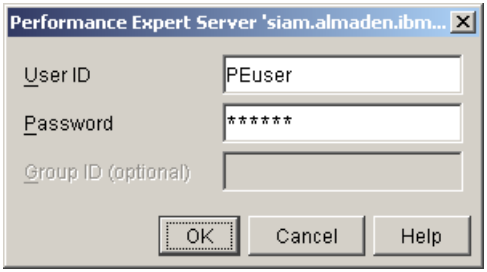


Figure 3-23 New DB2 System window - prompt for User ID and password

6. The list of instances in Figure 3-24 on page 99 that have been configured for monitoring on the PE Server appears. We want to register the instance listed in the configuration worksheet and also three other instances that have been configured for monitoring. We perform the following steps for each row of the list:

   a. We double-click the **DB2 Connection Alias** field for each row and enter a database alias for the performance database of the monitored instance. We choose to use the performance database name as the DB2 connection alias for each instance.

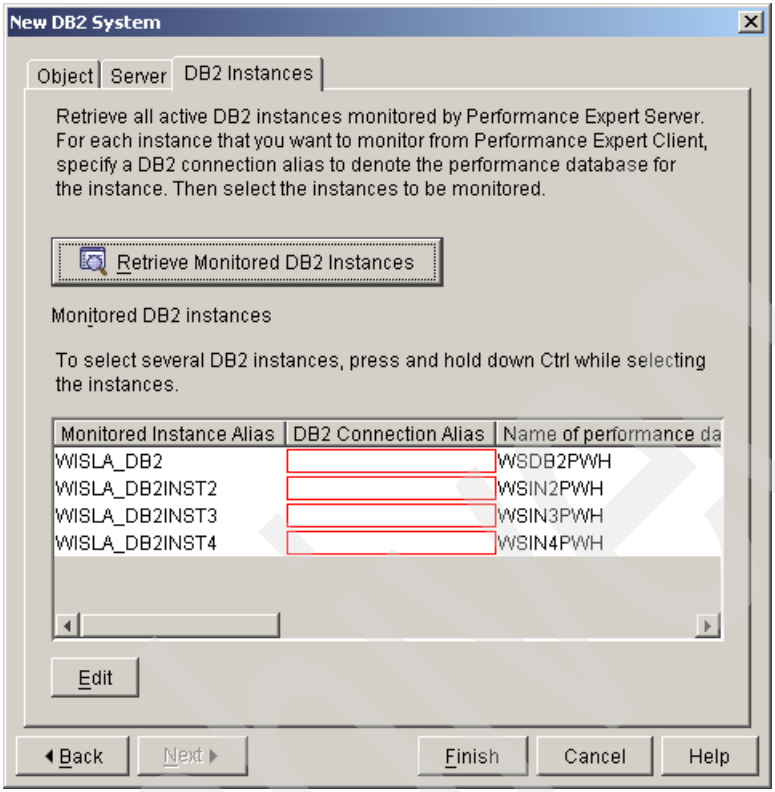


Figure 3-24 New DB2 System window - list of monitored instances

b. We decide not to change the Monitored Instance Alias field for any of the instances, because we still find the naming convention that we used when we registered the instance for monitoring on the PE Server to be useful.

   Clicking the **Edit** button would provide the same functionality to edit these fields but would also display the monitored databases for each instance and the event exception monitoring status for each database.

c. We select the row of each instance to monitor, pressing Ctrl while selecting each one in order to select multiple instances (see Figure 3-25).

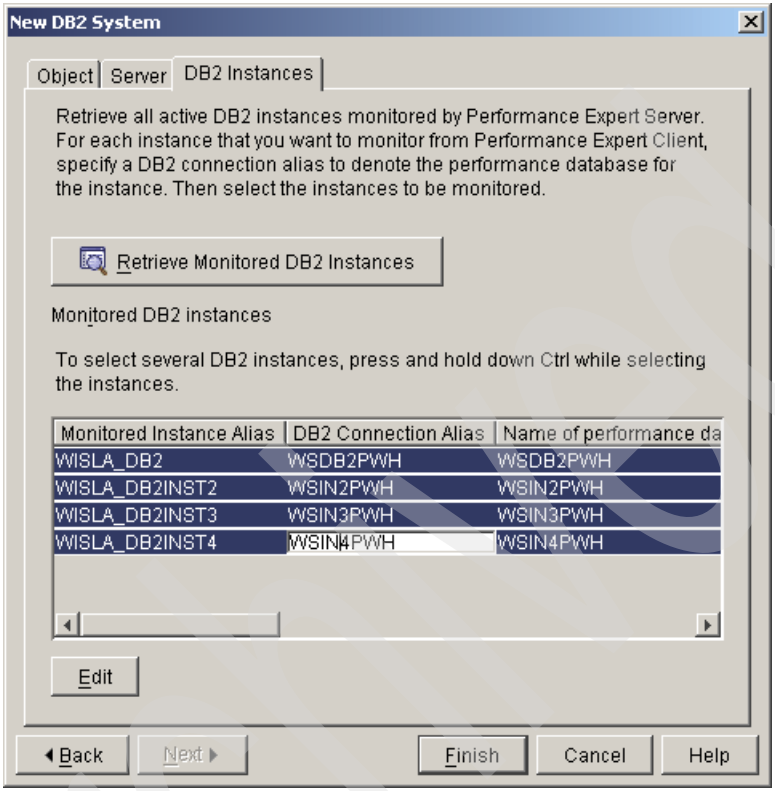


Figure 3-25 New DB2 System window - selecting instances to monitor

7. Click **Finish**. A window appears with a successful log entry for each instance added (see Figure 3-26 on page 101). The instances we specified for monitoring now appear in the System Overview window in the objects pane and in the DB2 Server Status pane (see Figure 3-27 on page 101).

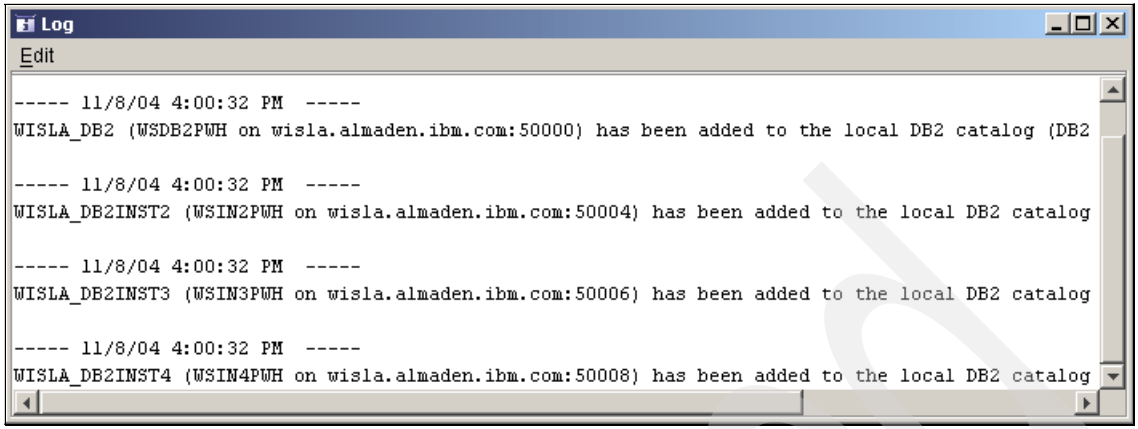


Figure 3-26 Successful log messages after registering instances

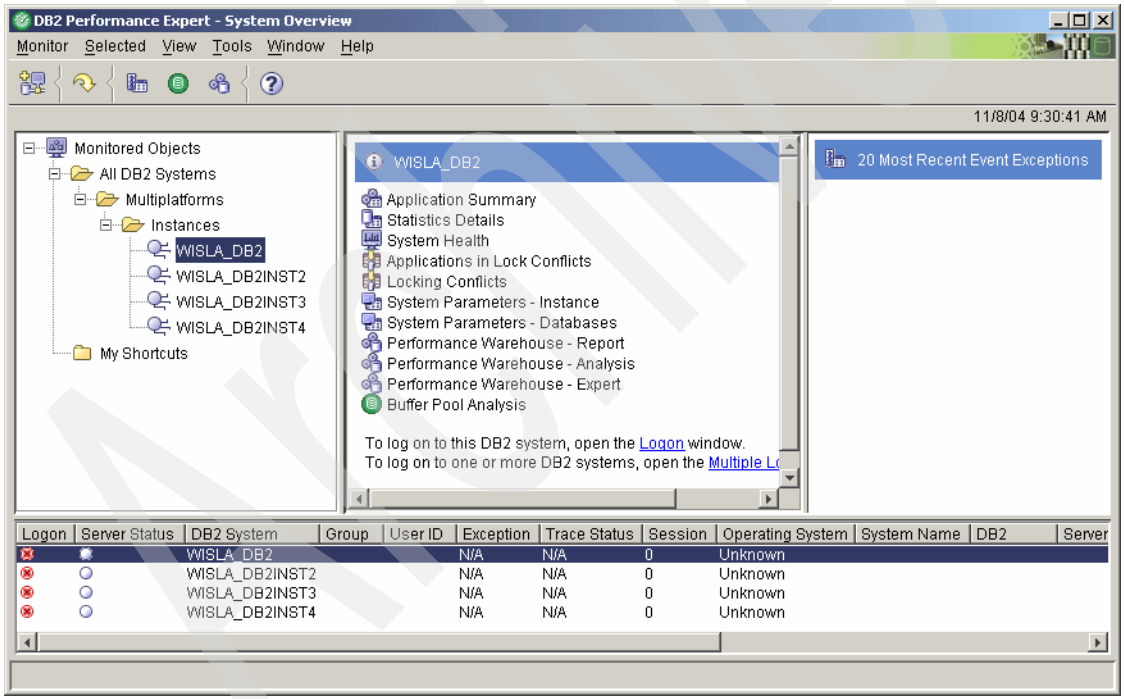


Figure 3-27 Viewing newly registered instances in the System Overview

We check and see that PE Client has created the node and database directory entries in Example 3-10 in the active local DB2 instance.

Example 3-10 DB2 catalog entries created during PE Client configuration

```
Node 1 entry:

 Node name                      = CSND0001
 Comment                        = PE Server: siam.almaden.ibm.co
 Directory entry type           = LOCAL
 PRoTocol                       = TCPIP
 Hostname                       = siam.almaden.ibm.com
 Service name                   = 50000

Database 1 entry:

 Database alias                       = WSIN2PWH
 Database name                        = WSIN2PWH
 Node name                            = CSND0020
 Database release level               = a.00
 Comment                              = PEClient: WSIN2PWH on wisla
 Directory entry type                 = Remote
 Authentication                       = SERVER
 Catalog database partition number    = -1

Database 2 entry:

 Database alias                       = WSIN3PWH
 Database name                        = WSIN3PWH
 Node name                            = CSND0020
 Database release level               = a.00
 Comment                              = PEClient: WSIN3PWH on wisla
 Directory entry type                 = Remote
 Authentication                       = SERVER
 Catalog database partition number    = -1

Database 3 entry:

 Database alias                       = WSIN4PWH
 Database name                        = WSIN4PWH
 Node name                            = CSND0020
 Database release level               = a.00
 Comment                              = PEClient: WSIN4PWH on wisla
 Directory entry type                 = Remote
 Authentication                       = SERVER
 Catalog database partition number    = -1

Database 4 entry:
```

```
Database alias                       = WSDB2PWH
Database name                        = WSDB2PWH
Node name                            = CSND002O
Database release level               = a.OO
Comment                              = PEClient: WSDB2PWH on wisla
Directory entry type                 = Remote
Authentication                       = SERVER
Catalog database partition number    = -1
```

PE Client manual configuration messages

Below is a partial listing of messages that may be received if problems are encountered while adding instances to monitor in PE Client along with possible causes. For a complete list of messages for Performance Expert, please see the publication *IBM DB2 Performance Expert for Multiplatforms, Workgroups, and z/OS IBM DB2 Performance Monitor for z/OS IBM DB2 Buffer Pool Analyzer for z/OS Messages*, SC18-7974.

The system name cannot be resolved

Figure 3-28 on page 104 shows the error. This error can be caused by the following items:

- The host name entered is incorrect.
- The host is currently unavailable in the network. Begin troubleshooting connectivity by pinging it from a command prompt.
- The host name entered is incomplete (try using the fully qualified host name). For example, a host named jamaica may only be accessible by using jamaica.almaden.ibm.com or the IP address.

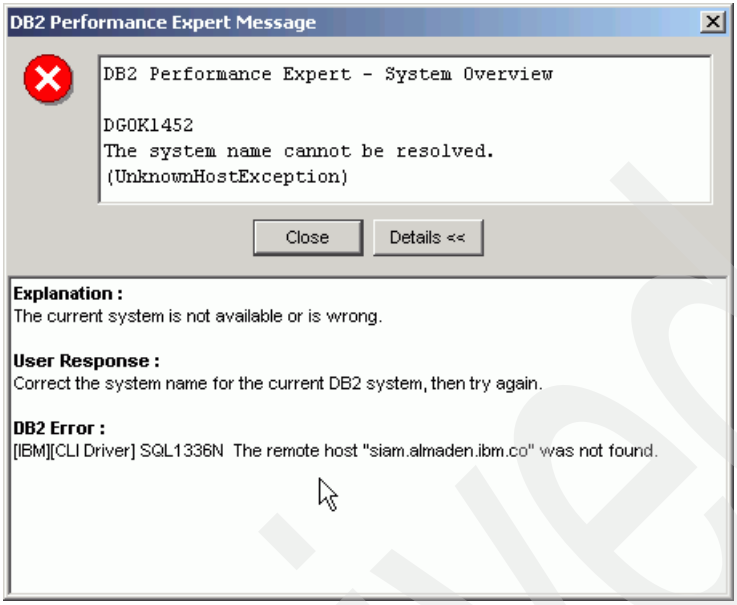


Figure 3-28 PE Client configuration message - system name cannot be resolved

Wrong port number specified

Figure 3-29 on page 105 shows the error. This error can be caused by the following items:

- You specified a port number other than the one in use for the DB2 instance on which PE Server runs.

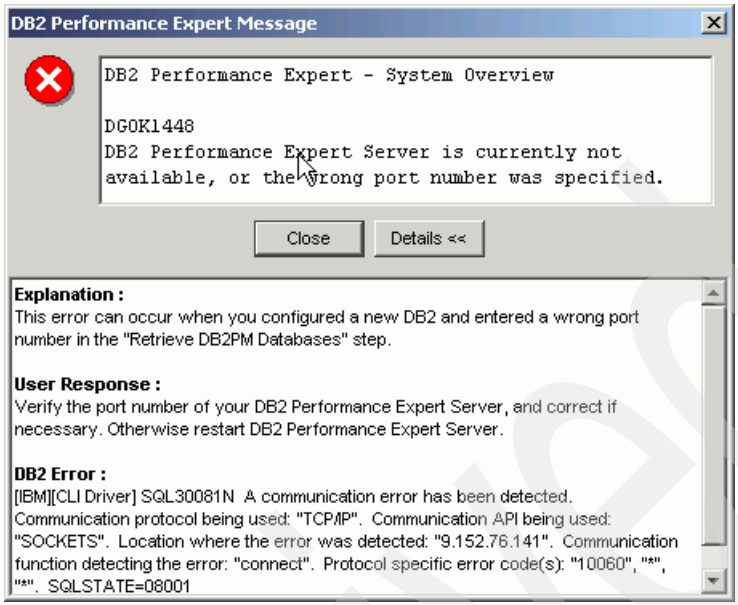


Figure 3-29 PE Client configuration message - wrong port number specified

Authorization verification failed

Figure 3-30 on page 106 shows the error. This error can be caused by:

- The user ID supplied does not exist on the host system. In our testing, the user ID entered in the PE Client was not case sensitive on UNIX or Windows systems, even though the user ID was case sensitive when logging on directly to the systems involved.
- The password supplied was incorrect.
- The user ID supplied is a member of the defined PE Server group, but the connect authority has been revoked from the PE group in the master database for the PE Server.
- A correct host name was supplied along with a port number that exists on the host but is not the port number defined for the DB2 instance on which PE Server runs (for example, the port number of a different DB2 instance on the same host), regardless of whether the user ID and password supplied have access to either DB2 instance.

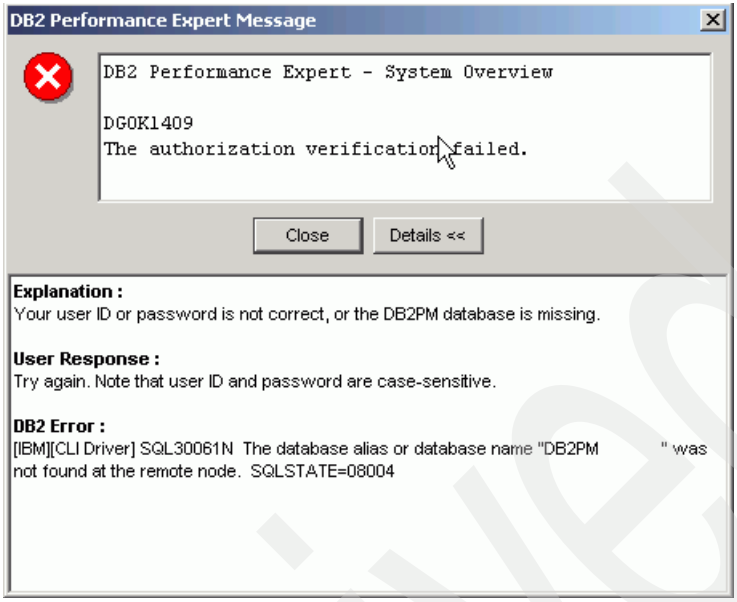


Figure 3-30 PE Client configuration message - authorization verification failed

User ID does not have the correct privileges

Figure 3-31 on page 107 shows the error. This error can be caused by the following items:

- The user ID supplied exists on the host system but does not belong to the group that was given access to PE Server when **`pecentralize`** was run on UNIX or when PE Server installation was performed on Windows.
- Object privileges in the master database for the PE Server have been revoked from the defined PE group (revoked privileges other than those shown in the example may result in a similar message). The necessary privileges are granted by PE Server and should not be manually altered.

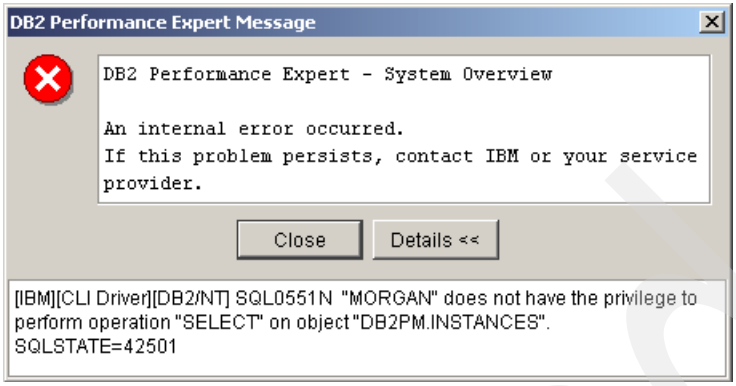


Figure 3-31 PE Client configuration message - ID privileges problem

DB2 connection alias not correct

Figure 3-32 shows the error. This error can be caused by the following items:

- The alias entered is longer than eight characters, begins with a number, or contains one or more invalid characters (@, #, $, the underscore character, and so on).

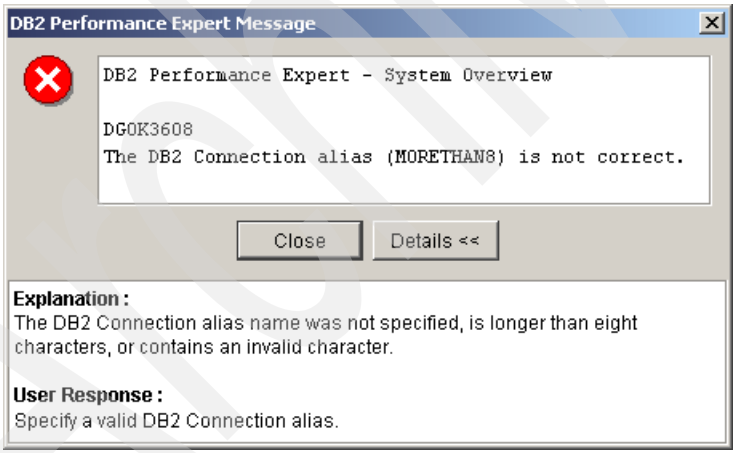


Figure 3-32 PE Client configuration message - alias is not correct

No monitored DB2 instance is selected

Figure 3-33 shows the error. This error can be caused by the following item:

- A row was not highlighted in the Monitored DB2 instances table when the **Finish** button was clicked.

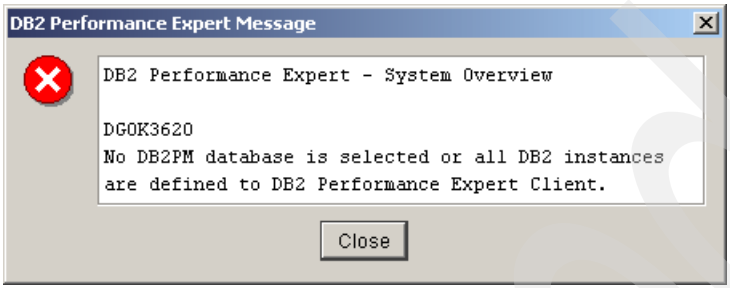


Figure 3-33 PE Client configuration message - monitored instance not selected

Monitored instance alias already in use

Figure 3-34 shows the error. This error can be caused by the following item:

- You attempted to catalog a new DB2 instance with a monitored instance alias already in use.

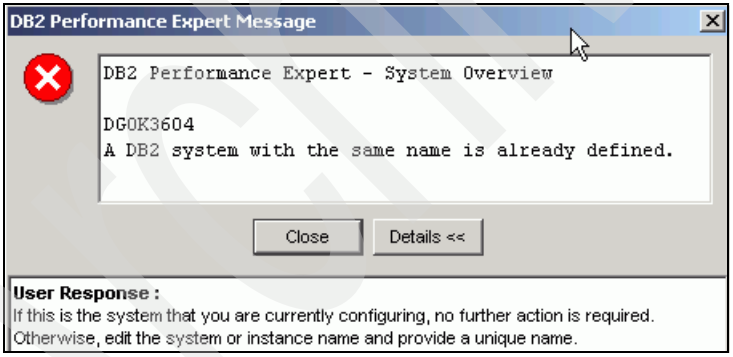


Figure 3-34 PE Client configuration message - monitored alias in use

Database alias already in use

Figure 3-35 on page 109 shows the error. This error can be caused by the following items:

- You attempted to register a monitored instance that has already been registered but used a different form of the host name than was used the first time (host name or IP address), and you specified the same database alias.

- You attempted to catalog the performance database for a new monitored DB2 instance with an alias that is already in use in the local database directory.

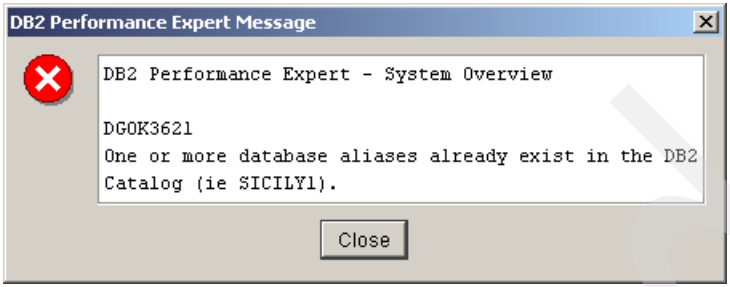


Figure 3-35 PE Client configuration message - database alias in use

Monitored instance already defined

Figure 3-36 shows the error. This error can be caused by the following items:

- You attempted to register a monitored instance that was already registered on the same PE Client, regardless of the monitored instance alias or DB2 connection alias specified, and you used the same form of the host name (name or IP address).

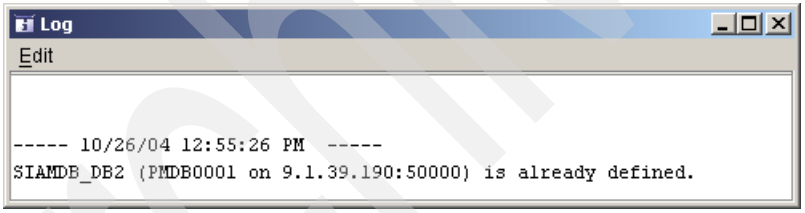


Figure 3-36 PE Client configuration message - monitored instance already defined

Database alias not found

Figure 3-37 shows the error. This error can be caused by the following items:

- You attempted to log on to an instance that has not been enabled yet, hence the appropriate performance database has not been created. Run **`peconfig`** on the PE Server, executing **`list`**, which gives an overview about all configured instances, and enable subject instance. Remember that the performance database for the monitored instance is created when the instance is enabled for the first time.

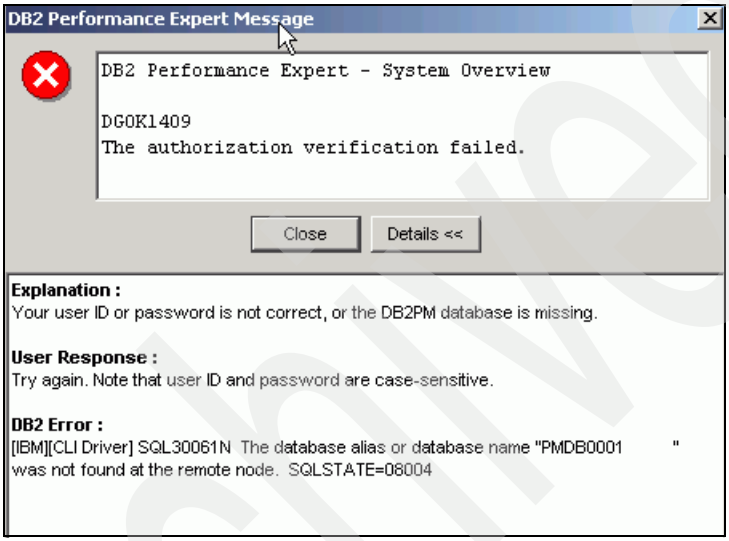


Figure 3-37 PE Client configuration message - database alias not found

Exporting/importing DB2 instances

Exporting and importing information about registered DB2 instances can be useful if you wish to use several PE Clients to connect to the same PE Server(s). You only need to define the DB2 instances at one client and then use the export and import features to transfer these properties to other PE Clients.

To export DB2 instances, select **Monitor** → **Export DB2 Systems**. Input the path and file name to which you want to save the information and click **OK**. A window opens containing the output of the export operation (see Figure 3-38 on page 111).

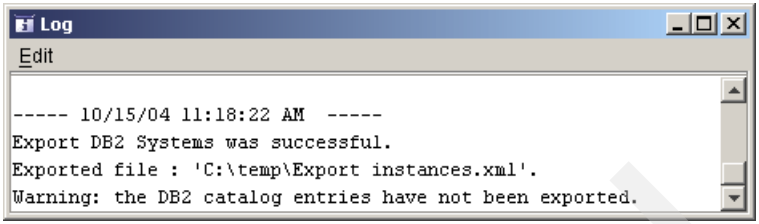


Figure 3-38 Log for exporting DB2 Systems

To import DB2 instances to another PE Client, select **Monitor** → **Import DB2 Systems**. Input the path and file name for the file from which you want to import the information and click **OK**. A window opens containing the output of the import operation. The imported instances should now appear in the System Overview window. You must be logged off from all monitored instances in order to import instances.

Manually adding DB2 instances will automatically create the appropriate node and database catalog entries in the DB2 instance used by the PE Client, but importing the instances will not create these entries. You may receive a message such as the one shown in Figure 3-39 if you attempt to log on to instances that are not locally cataloged, or are cataloged on a different local DB2 instance than the one PE Client is currently using.

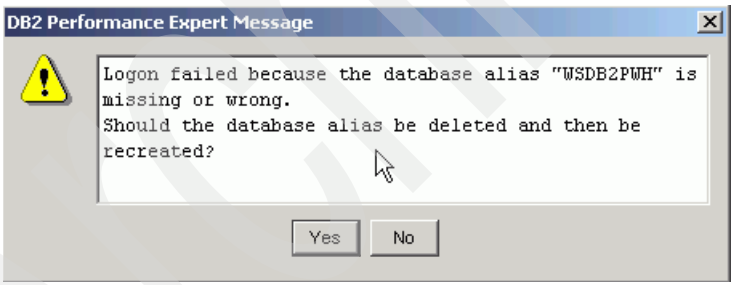


Figure 3-39 Error: attempt to log on to an instance not cataloged at the PE Client

Click **Yes** to get the database alias catalogued by the PE Client.

If cataloging does not work for some reason, you may also create the appropriate catalog entries manually in one of the following ways:

- Manually create them using the DB2 Command Line Processor or Configuration Assistant.

  If you do not have the necessary information, the local database alias, host name, and port number that were previously used can be found in the PE Server Properties window in the PE Client by selecting the imported instance

in the System Overview window and selecting **Selected → Properties** (or right-click it and choose **Properties**). The name of the performance database on the PE Server for the instance you want to monitor can be found by running **peconfig** on the PE Server, executing **list**, which gives an overview about all configured instances, then executing **list [instanceid]**, which lists detailed information, and then looking at the line for “Performance database”.

Note: If you do not have access to the PE Server configuration, the name of the performance database can alternately be found by opening the exported XML file containing the exported instance information in a text editor and searching for the centralsrvinst parameter.

In the database catalog entry, the database name and alias must match those that were used on the other system. In the node catalog entry, the host name must point to the correct host, but it may be in a different form (IP address, fully qualified host name, and so on). The node name may change, but it must match between the new database and node entries. For example, the following statements would recreate the catalog entries necessary at the PE Client to monitor an imported instance for the PE Server properties depicted in Figure 3-40 on page 113, and for which the name of the performance database on the PE Server is PWHINST1:

```
db2 catalog tcpip node jamaica remote jamaica.almaden.ibm.com server 60004
db2 catalog db pwhinst1 as jampwh1 at node jamaica
```

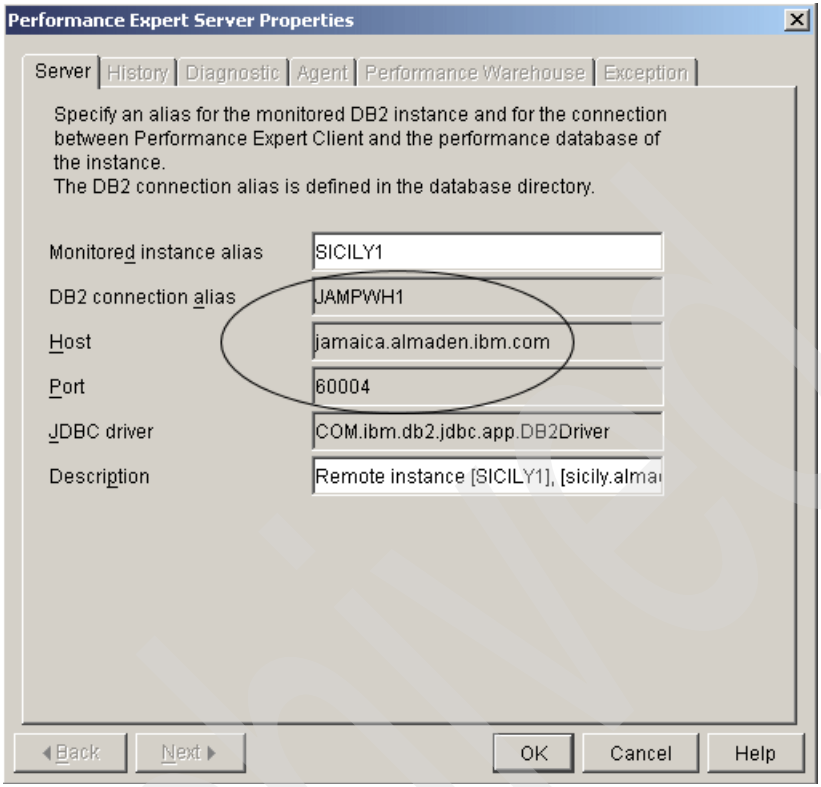


Figure 3-40 PE Server information needed for manually creating catalog entries

- Export the catalog entries from the system from which the DB2 instances were also exported from PE Client. This can be done using the Export Profile feature in the DB2 Configuration Assistant. Then import them using the Import Profile feature of the Configuration Assistant on the PE Client system.

Note: You do not need to create catalog entries on the PE Client workstation for the actual DB2 instance or databases being monitored. These are cataloged in the DB2 instance used by the PE Server.

3.1.7 Getting started

In this section, we describe some preliminary steps, including logging on to monitored instances and changing PE Server and Client settings. We also discuss which PE features are available from the System Overview window both with and without further configuration.

Logging on to monitored instances

To view performance data for a monitored DB2 instance, you must first log on through the PE Client to the PE Server that is configured to monitor that instance. When you use the logon feature, the ID and password you enter are authenticated by the PE Server at which the monitored DB2 instance has been defined and enabled for monitoring. Two DB2 connections are initially made to the performance database that has been created in the DB2 instance on which the PE Server runs for the monitored instance, and additional connections are made as necessary when you use additional features in the PE Client. You must log on separately for each DB2 instance monitored by the same PE Server. We will often refer to this process as logging on to a monitored instance, even though no authentication takes place at the monitored DB2 instance for the ID and password you use in the PE Client to log on.

The PE Server, the DB2 instance on which it runs, and the monitored DB2 instance must all be started before you log on. To log on, you need access to a user ID that belongs to the user group that was given access rights to PE Server (for UNIX systems, this group was specified when **pecentralize** was executed; for Windows systems, this group was specified during PE Server installation).

To log on to a single monitored instance, click the instance in the System Overview window and select **Selected** → **Logon** (or right-click the instance and select **Logon**). Enter the user ID and password and click **OK**.

To log on to multiple monitored instances, your user ID and password must have been given access to all of the instances to which you wish to log on in one step. In the System Overview window, select **Monitor** → **Multiple Logon**, select the DB2 instances to which you want to log on (optionally, select the components you wish to open), and click **OK**. In the Logon window, enter the User ID and password and click **OK** (see Figure 3-41 on page 115).

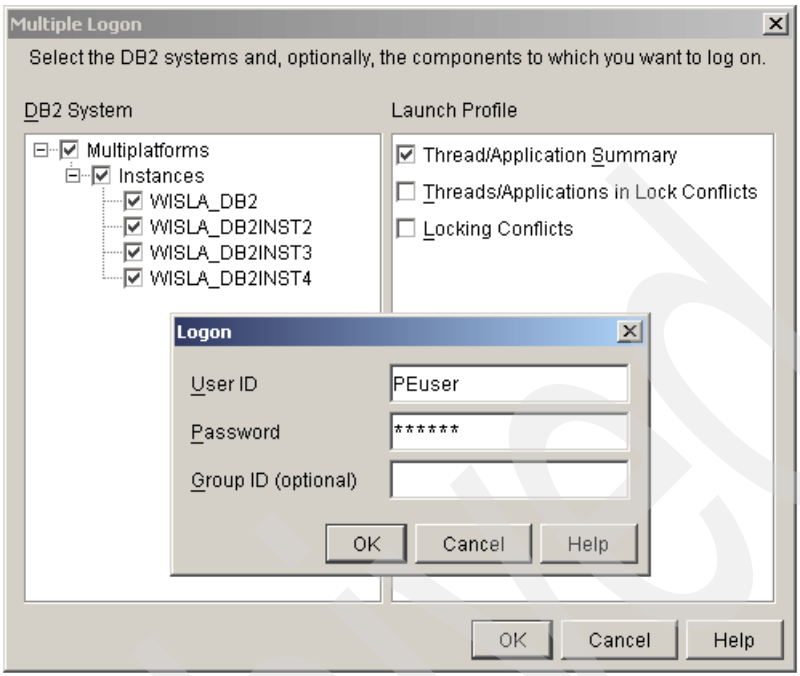


Figure 3-41 Multiple Logon in PE Client

After you have logged on to monitored instances individually or by using the multiple logon feature, a green check mark in the Logon column of the bottom pane of the System Overview window indicates to which instances you have logged on.

If you have added a large number of monitored instances and want to organize them, you can create a new folder under the My Shortcuts folder in the System Overview and then copy and paste defined instances to these folders. For example, you could create a folder for each PE Server that contains shortcuts for each DB2 instance it monitors, or you could create separate folders for development, test, and production systems.

Figure 3-42 shows several features of the System Overview window, including shortcuts, the icon indicating users are logged on, data views defined in the System Health window, and event exceptions reported.

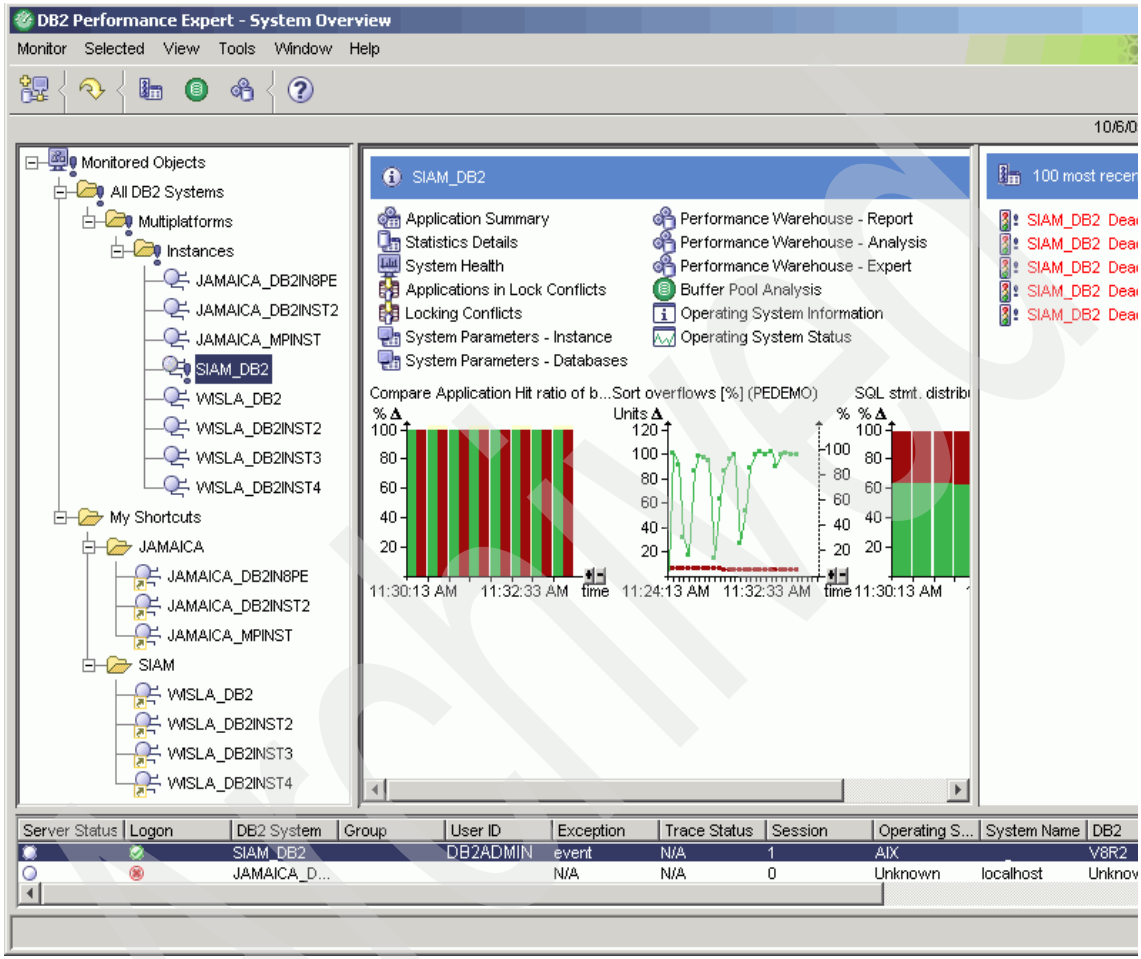


Figure 3-42 System Overview window

Configuring PE Client settings

To configure global settings for PE Client, select **Monitor** → **Configuration**.

- On the Preferences tab (see Figure 3-43 on page 117), you can specify your default Web browser, permanently save your logon passwords for all DB2 instances, and reactivate the display of message DGOK3836 (monitor switches are not set). Furthermore, with V2.2, you can set the minimum period of time that, in the folders pane of the System Overview window, an

exception icon is shown next to the DB2 instance for which an exception was recorded and next to the folder containing this instance. If you specify a duration of 0 minutes, the exception icon is never shown. By clicking the **Advanced** button, you can view or change the port number used by PE Client.

- On the Diagnostics tab, you can specify e-mail settings to send diagnostic data to IBM support. Note that these e-mail settings may be different than those you can specify for e-mail notification for exception monitoring (see 4.2, “Alerts and exceptions” on page 193).
- On the Accessibility tab, you can specify an appearance scheme to alter the level of display contrast and the font size.

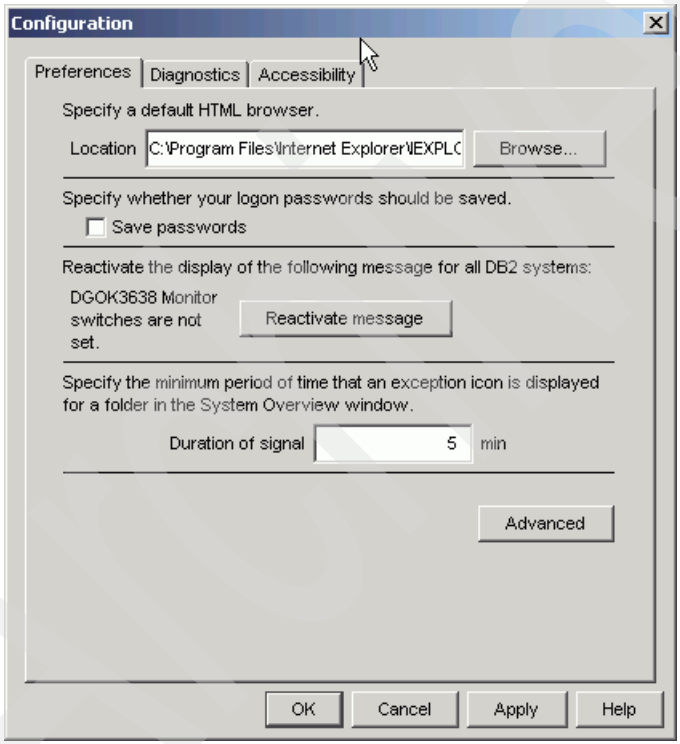


Figure 3-43 PE Client configuration

Changing PE Server properties

When you are *logged off* from a PE Server instance and you select **Selected** → **Properties** while its entry is selected (or right-click it and select **Properties**), then only the Server tab is available in the Performance Expert Server Properties window. In this window, you can change the following fields:

- Monitored instance alias

  This alias is only used in PE Client to identify the monitored instance in the System Overview window. An entry is not made for it in the local DB2 node directory.

- Description

  This field is for informational purposes only. By default, it contains information about the monitored instance, such as the server name and port number, but you may wish to change it or add additional information to it. For example, you might choose to add the name of the performance database on the PE Server for the monitored instance. Adding this detail would be useful if you exported the registration information for an instance and imported it to another client on which the performance database was not already cataloged.

When you are *logged on* to a PE Server that monitors an instance and you select **Selected** → **Properties** while its entry is selected (or right-click it and select Properties), then all tabs except for the Server tab can be edited.

Tip: As the window title indicates, all changes made to editable fields in this window while logged on to the PE Server will be made at the PE Server, and they are specific to one monitored DB2 instance. These changes will be visible to and effective for all PE Clients that connect to the PE Server to monitor the same DB2 instance.

- On the History tab (see Figure 3-44 on page 119), you can enable and alter history settings, including a recording interval, the multiple of that interval to be used for each component, and a time frame for history data retention for all components. Once enabled, the PE Server takes DB2 snapshots of databases in each enabled monitored instance at these intervals, regardless of how many PE Clients are currently open.

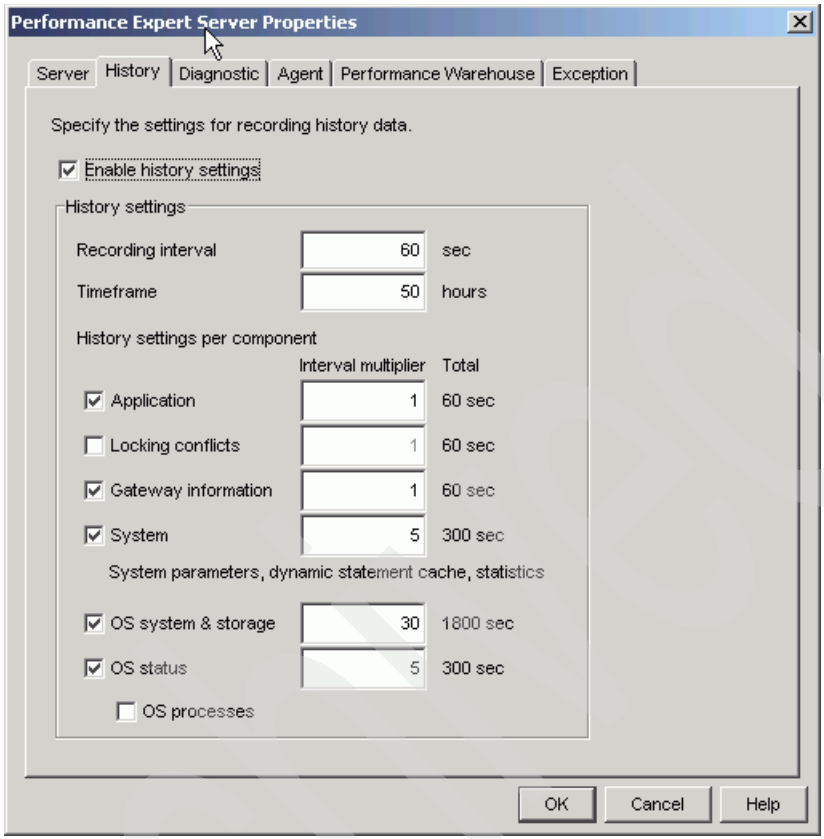


Figure 3-44 Altering PE Server Properties

Tip: By default, history recording is disabled for *locking conflicts* and enabled for all other *components*. You may want to enable collection of historical locking data or disable historical data collection for other components, depending on your needs and desired level of monitoring *overhead*.

- On the Diagnostic tab, you can specify trace utility settings for running different traces to analyze problems with PE Server. See 3.2.3, "Troubleshooting the PE Server" on page 133 for details.
- On the Agent tab, you can enable automatic updates to the PE Agent.
- On the Performance Warehouse tab, you can enable the storage of history data in the performance database and specify the data storage settings. If the history settings are enabled on the History tab, then data for the components checked there can be stored.

- On the Exception tab, you can enable and set retrieval intervals for event and periodic exception processing. You can also specify e-mail settings for exception notifications. See 4.2, “Alerts and exceptions” on page 193 for details about setting up and using exception processing.With V2.2 FP1, user exit settings can be specified as well.

Monitoring DB2 instances

After you have logged on to a monitored instance, many types of performance information will be available without further configuration by selecting the following choices from the System Overview window, as shown in Figure 3-45:

- Application Summary
- Statistics Details
- Applications in Lock Conflicts
- Locking Conflicts
- System Parameters - Instance
- System Parameters - Database
- Operating System Information
- Operating System Status

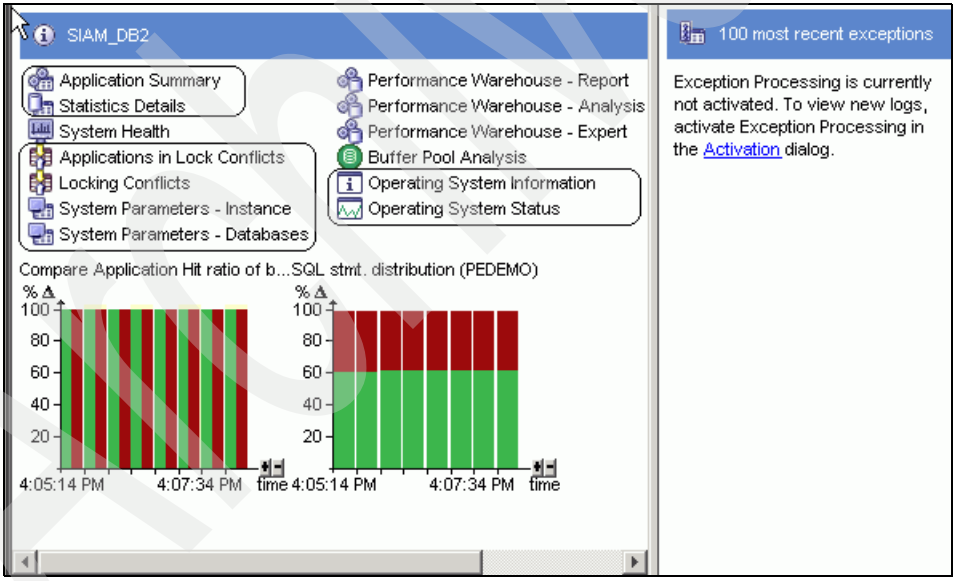


Figure 3-45 Performance Expert features available without customization

Viewing performance information in the following main features of Performance Expert will require further configuration to customize the information you wish to see, as shown in Figure 3-46 (the necessary steps are discussed in Chapter 4, “Features and functions - online and short-term monitoring” on page 143):

- System Health
- Performance Warehouse
- Buffer Pool Analysis
- Exception Processing (from the Tools menu)
- SQL Activity Tracing (in Application Summary or Performance Warehouse)

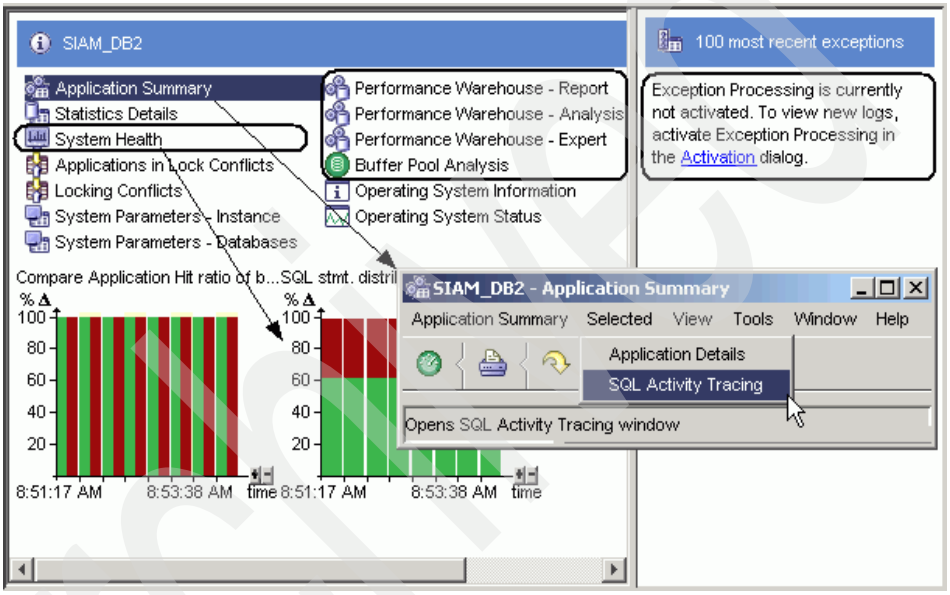


Figure 3-46 Performance Expert features available after customization

In all features, the fields that contain data will reflect the monitoring switches that are enabled on the monitored DB2 instance and the components for which you have enabled history data in the PE Server properties. By default, you will be prompted with a message when you open PE Client if one or more monitor switches are not enabled on the monitored DB2 instance.

Multiple windows can be opened in one step by selecting **Monitor** → **Multiple Launch** and choosing the desired DB2 instances and PE components. For example, you could use this feature to quickly check for locking conflicts on multiple monitored instances, as shown in Figure 3-47. Instances will appear on the Multiple Launch screen only after you have logged on to them.

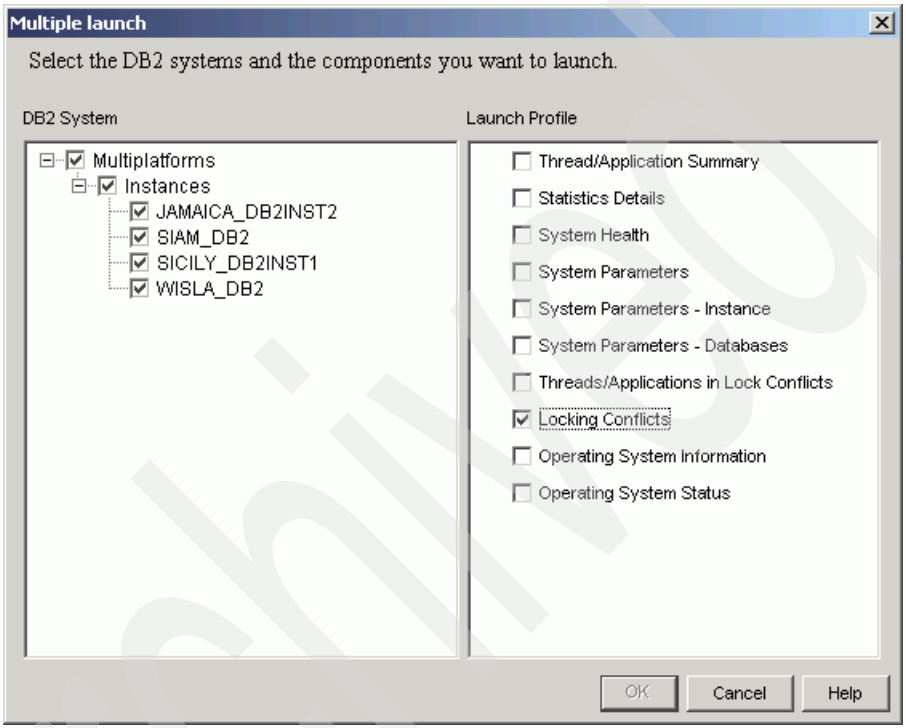


Figure 3-47 Using the "Multiple launch" feature for locking conflicts

3.2 Installation verification and support

After installation, or whenever you add a new monitored instance to PE, you should check PE Server log files to confirm the final stages of setup have completed successfully. The first time you start the PE Server after adding a new monitored instance, the PE Server populates its control tables in the new performance database and creates some other database objects. Clients cannot successfully connect to the PE Server until these steps are completed. Generally the first-time initialization tasks can take three to five minutes, but may vary from system to system.

This section describes the files to look at after installation to confirm success, and suggests some troubleshooting steps in case of problems in the PE Server.

3.2.1 What log files does PE generate?

The PE Server keeps one active log file (db2pesrv.log) and one active trace file (db2pesrv.trc, db2pemaster.trc with V2.2 FP1 or later) at the PE Server level, and one active trace file (db2pesrv.trc) for each monitored instance.

There is also a trace file (peconfig.trc) for the peconfig program, which tracks every invocation of the peconfig program. This can be useful for keeping track of what changes were made to the server configuration.

If you enable the snapshot trace, a second companion snapshot trace file (fpesnap.trc) is generated.

Each active log and trace file can grow to a maximum size of four megabytes, then a new active file will be created and the full file is renamed to <filename>.logbak or <filename>.trcbak. When the next active file is full, the old bak file is removed. The contents of the log and trace files are controlled by diagnostic trace settings in the PE Client. This is discussed in “Enabling server traces” on page 137.

Generally you do not need to enable additional tracing, and the log files will stay small. By default, only severe errors are logged, and this is usually enough for day-to-day troubleshooting. Enhanced tracing is performed at the request of IBM support as necessary.

Note: For the rest of this discussion on logging, we use the term “log file” interchangeably with “trace file.”

Where are the log files?

The AIX environment used in this example matches the diagram in Figure 3-49 on page 127. In this example, the PE Server is installed in /opt/IBM/db2pesrv/V2.2/bin. The db2pesrv.cfg file is always stored in /var/db2pe/v22 and has a pointer to the PE working directory. In these examples, the working directory is /db2pe. A directory under the working directory is always added for the PE Server, and this is where the server working files go.

- PE install directory: /opt/IBM/db2pesrv/V2.2/bin
- PE working directory: /db2pe

Note: On UNIX platforms, during **pecentralize** you are asked to supply the PE working directory. This value is stored in db2pesrv.cfg as "db2pe_homedir". To avoid confusion with a UNIX user home directory, we refer to this as "working directory" in this redbook.

All the working files specific to the PE Server are kept in the working directory, and all the working files related to each monitored instance are kept together in subdirectories underneath.

Server-level logs

For example, on /db2pe/db2in8pe we find the following PE Server log files:

- PE Server log and trace files
  - db2pesrv.log
    - PE start/stop messages, significant errors.
  - db2pesrv.trc (respectively db2pemaster.trc with V2.2 FP1)
    - One entry for PE Server start/stop.
    - Contents controlled by server properties in GUI.
- peconfig trace files
  - peconfig.trc
    - Logs all output from **peconfig**.
    - Level of detail controlled only by db2pesrv.prop file, not from GUI.
  - pesilent.trc
    - If you run peconfig in silent mode, the output is logged here.

Note: The db2pesrv.prop file, in the PE working directory, also contains some keywords that influence tracing levels. It is simpler to control diagnostic tracing from the PE Client GUI, and the results are the same. The only reasons you would use the db2pesrv.prop file are:

- You are asked to do so by IBM support.
- You are unable to use the PE Client and need to research some other problem.
- You need to trace peconfig in more detail.

Logs per monitored instance

For each monitored instance, there are also log files. In our AIX example, we are monitoring two instances, db2inst2 on jamaica, and db2inst1 on sicily. When a

new monitored instance is added, a new subdirectory under the main working directory is created; the name matches the PE node name given in the peconfig.

For example, for the monitored instance on jamaica, the following log files are kept in /db2pe/db2in8pe/JAMINST2:

- db2pesrv.trc
  - One entry for PE Server start/stop.
  - Contents controlled by server properties in GUI (see Figure 3-48 on page 126).
- fpesnap.trc
  - Detailed logging for snapshot processing, controlled by GUI as above or db2pesrv.prop file settings.
- fpecim.trc
  - Detailed logging for CIM data processing, controlled by GUI as above or db2pesrv.prop file settings.

We would find the same structure of files in /db2pe/db2in8pe/SICILY1, for example.

Windows differences

The PE Server directory structure is a bit different on Windows, but the log files are the same. In Windows, there is no db2pesrv.cfg file, and you cannot specify a working directory, so all logs are created in directories under the install path.

Figure 3-48 is an example for a PE Server on Windows. A non-default installation path was chosen, but the structure is the same. The PE Server instance name is PEINST, and the server is monitoring several instances.

- Server-level logs are in <install_path>\instances\<PE Server instance name>, for example, C:\PEServerV2\instances\PEINST.
- Monitored instance logs are in <install_path>\instances\<PE Server instance name>\<monitored instance node name>, for example, C:\PEServerV2\instances\PEINST\PE_DB2 and C:\PEServerV2\instances\PEINST\PE_INST1.

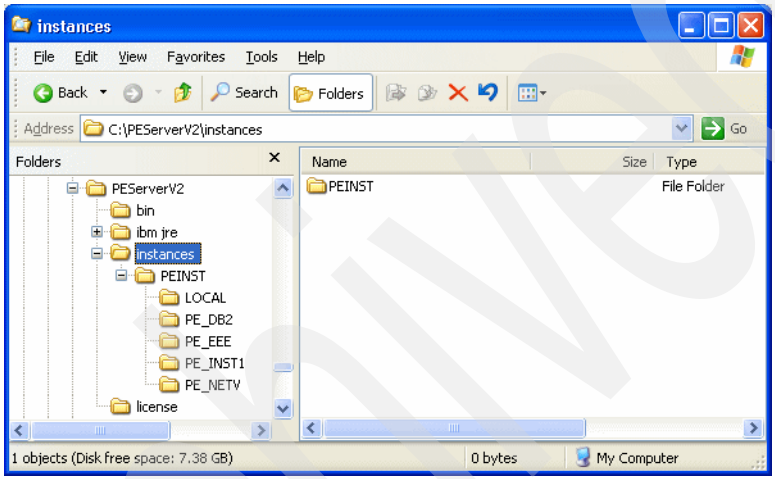


Figure 3-48 PE log location on Windows

3.2.2 Interpreting and using the log files

We have described the kinds of log files PE keeps, and where they are. Now we can discuss when and how to use them.

The examples in this section correspond to the environment in Figure 3-49 on page 127.

.

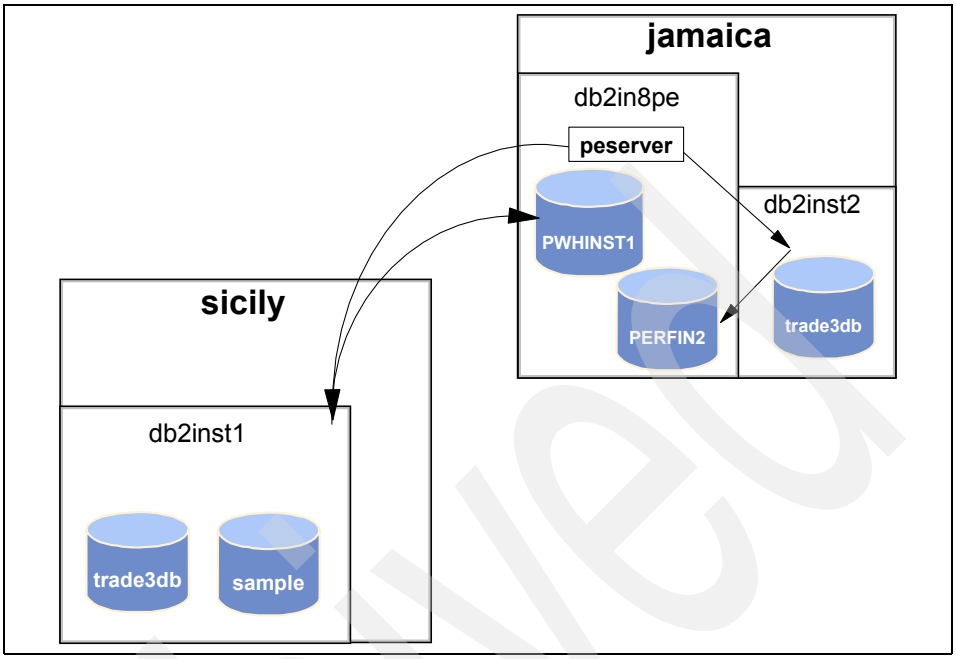


Figure 3-49 Sample environment

Scenario: Verifying PE Server is up and running

You should not try to connect to the PE Server from the PE Client until the server is completely started and has displayed the `alive` message in the log. If you do try, you will get an error, as shown in Figure 3-50.

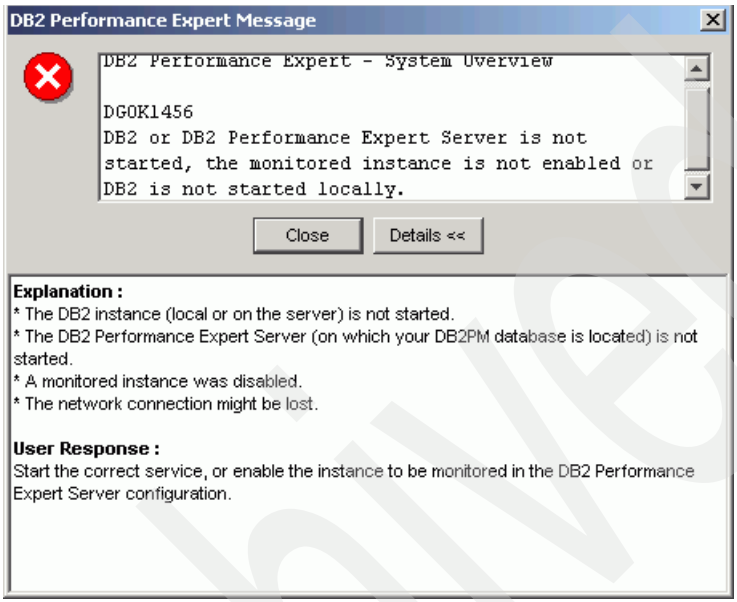


Figure 3-50 Client tried to log on before server was initialized

The snippet from **peconfig** in Example 3-11 shows that the PE Server on jamaica is configured to monitor two instances: one on sicily and one on jamaica. The node name shown in the **peconfig** output (JAMINST2), corresponds to the value seen in the db2pesrv.log shown in Example 3-12 on page 129.

Example 3-11 Description of jamaica PE Server configuration

```
...
peconfig => list

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

Instance ID : 1
-----------------
   Enabled : Yes
   Status = Inactive
   CIM Object Manager enabled = No
   Monitored Instance Alias : SICILY1
   Node, Host, Port/Service name : SICILY1, sicily.almaden.ibm.com, 60000
   Database, remote alias, local alias, EVM : TRADE3DB, TRADE3DB, SICTRD3, Yes
```

```
   Database, remote alias, local alias, EVM : SAMPLE, SAMPLE, SICSMPL, Yes

Instance ID : 2
-----------------
   Enabled : Yes
   Status = Inactive
   CIM Object Manager enabled = No
   Monitored Instance Alias : JAMAICA_DB2INST2
   Node, Host, Port/Service name : JAMINST2, localhost, 60020
   Database, remote alias, local alias, EVM : TRADE3DB, TRADE3DB, JAMTRADE, Yes
...
```

The db2pesrv.log sample shown in Example 3-12 shows a successful startup of the AIX PE Server environment as described in Figure 3-49 on page 127, and in the **peconfig** output in Example 3-11 on page 128.

Tip: The instance ID used in peconfig does **not** correspond to the monitored instance identifier number used in the db2pesrv.log file.

When you see the line showing the instance status is alive, this means all the threads are initialized and server is ready to accept connections from the PE Client. You should always wait for this message to be sure the server is up and running. In UNIX environments, if you enter the **pestart** command at the command line, these log messages are displayed at the console. In the Windows environment, you should look at the log file to be sure the server is started.

Example 3-12 Successful PE Server startup show in db2pesrv.log

```
-------------- Start at Wed Oct 13 19:47:34 PDT 2004
--------------------------------------------------------------------------------
[19:47:34.484] IBM(c) DB2 Performance Expert Server for Multiplatforms
[19:47:34.687] IBM(c) DB2 Performance Expert Server for Workgroups
[19:47:34.727] Version V2.2, Code Level "U220_GAHOT-U347" [R220_GAHOT-L1217,N220_GAHOT-E134],
Build Level "pmbuildV22-B048"
[19:47:34.754] Home direcory = "/db2pe/db2in8pe".
[19:47:34.754] Trace file    = "/db2pe/db2in8pe/db2pesrv.trc"
[19:47:34.755] Log file      = "/db2pe/db2in8pe/db2pesrv.log"
[19:47:35.207] Starting PE instance [3,SICILY1] using performance database 'PWHINST1'.
[19:47:35.235][3] PEServer: PE instance initializing ...
[19:47:35.237][3] PEServer: Home directory = /db2pe/db2in8pe/SICILY1
[19:47:35.237][3] PEServer: Trace file     = /db2pe/db2in8pe/SICILY1/db2pesrv.trc
[19:47:35.237][3] PEServer: SP Trace file  = /db2pe/db2in8pe/SICILY1/fpesnap.trc
[19:47:35.237][3] PEServer: CIM Trace file = /db2pe/db2in8pe/SICILY1/fpecim.trc
[19:47:35.29 ] Starting PE instance [6,JAMINST2] using performance database 'PERFIN2'.
[19:47:35.292][6] PEServer: PE instance initializing ...
[19:47:35.292][6] PEServer: Home directory = /db2pe/db2in8pe/JAMINST2
[19:47:35.293][6] PEServer: Trace file     = /db2pe/db2in8pe/JAMINST2/db2pesrv.trc
```

```
[19:47:35.293][6] PEServer: SP Trace file  = /db2pe/db2in8pe/JAMINST2/fpesnap.trc
[19:47:35.293][6] PEServer: CIM Trace file  = /db2pe/db2in8pe/JAMINST2/fpecim.trc
[19:47:35.395] PE instance [6,JAMINST2] status: initializing.
[19:47:35.396] PE instance [3,SICILY1] status: initializing.
[19:47:35.554][3] ****** Starting setup for performance database: PWHINST1 ******
[19:47:36.071][3] Default startup. Going to update database tables
[19:47:36.191][3] Database update successful.
[19:47:36.192][6] ****** Starting setup for performance database: PERFIN2 ******
[19:47:36.321][6] Default startup. Going to update database tables
[19:47:36.739][3] PWHLOAD started.
[19:47:36.877][3] PWH started.
[19:47:36.878][6] Database update successful.
[19:47:37.118][3] PEXP started.
[19:47:37.138][3] EXCP started.
[19:47:37.169][3] SNAP started.
[19:47:37.331][6] PWH started.
[19:47:37.331][6] PWHLOAD started.
[19:47:37.332][6] PEXP started.
[19:47:37.541][6] EXCP started.
[19:47:37.586][6] SNAP started.
[19:47:38.795] PE instance [6,JAMINST2] status: alive (1/2).
[19:47:38.795] PE instance [3,SICILY1] status: alive (2/2).
```

Scenario: Failed startup

If the PE Server monitors multiple instances, there will be an `alive` entry for each one. If the PE Server encounters problems during the startup, the `alive` message will not appear. In Example 3-13 on page 131 we show a case, under Windows, where the server could not start monitoring an instance.

The PE Server can continue to operate even if some of the monitored instances do not start normally. The PE Server will continue to try to start monitoring the failed instance by retrying at regular intervals. The attempts are logged in the db2pesrv.log file, so you can see how long the problem may have been happening.

In Example 3-13 on page 131, the reason for failure suggests the monitored instance may be down. We did not notice this error right away, so PE continued to try to start for about 10 minutes. The PE_INST1 server was indeed down, and it was restarted at about 17:52. At the next retry interval, PE Server was able to successfully start monitoring it, started the server, and shows the `alive` message.

Example 3-13 Failed monitored instance startup in db2pesrv.log

```
------------- Start at Sat Oct 30 17:39:01 PDT 2004
--------------------------------------------------------------------------------
[17:39:01.446] IBM(c) DB2 Performance Expert Server for Multiplatforms
[17:39:01.446] IBM(c) DB2 Performance Expert Server for Workgroups
[17:39:01.526] Version V2.2, Code Level "U220_GAHOT-U347"
[R220_GAHOT-L1217,N220_GAHOT-E134], Build Level "pmbuildV22-B048"
[17:39:01.546] Home direcory = "C:\PEServerV2\instances\PEINST".
[17:39:01.546] Trace file    = "C:\PEServerV2\instances\PEINST\db2pesrv.trc"
[17:39:01.546] Log file      = "C:\PEServerV2\instances\PEINST\db2pesrv.log"
[17:39:05.352] Unable to start monitoring for PE instance [12,PE_INST1], a
communication error has been detected, the database manager might not have been
started, SQL code = -30081, waiting to retry.
[17:39:05.462] *** Warning *********************************
[17:39:05.462] * No instance is currently being monitored. *
[17:39:05.462] *********************************************
[17:39:33.502] Restarting PE instance [12,PE_INST1] ...
[17:39:34.544] Unable to start monitoring for PE instance [12,PE_INST1], a
communication error has been detected, the database manager might not have been
started, SQL code = -30081, waiting to retry.
[17:40:03.525] Restarting PE instance [12,PE_INST1] ...
... <lines removed for example>
[17:52:33.694] Restarting PE instance [12,PE_INST1] ...
[17:52:34.726] Unable to start monitoring for PE instance [12,PE_INST1], a
communication error has been detected, the database manager might not have been
started, SQL code = -30081, waiting to retry.
[17:53:03.697] Restarting PE instance [12,PE_INST1] ...
[17:53:06.281] Starting PE instance [12,PE_INST1] using performance database
'PERFTP1'.
[17:53:06.281] PE instance [12,PE_INST1] status: initializing.
[17:53:06.281][12] PEServer: PE instance initializing ...
[17:53:06.311][12] PEServer: Home directory =
C:\PEServerV2\instances\PEINST\PE_INST1
[17:53:06.311][12] PEServer: Trace file     =
C:\PEServerV2\instances\PEINST\PE_INST1\db2pesrv.trc
[17:53:06.311][12] PEServer: SP Trace file  =
C:\PEServerV2\instances\PEINST\PE_INST1\fpesnap.trc
[17:53:06.311][12] PEServer: CIM Trace file  =
C:\PEServerV2\instances\PEINST\PE_INST1\fpecim.trc
[17:53:08.264][12] ****** Starting setup for performance database: PERFTP1
******
[17:53:09.686][12] Default startup. Going to update database tables
[17:53:13.862][12] Database update successful.
[17:53:15.144][12] PWH started.
[17:53:15.164][12] PWHLOAD started.
[17:53:15.454][12] PEXP started.
[17:53:15.584][12] EXCP started.
[17:53:16.395][12] SNAP started.
```

```
[17:53:16.398][12] PECIM started.
[17:53:20.291] PE instance [12,PE_INST1] status: alive.
```

Recommendation

You should devise some naming convention for your PE Server monitored instances and node names. The names are used throughout the PE Server logs and the PE Client, so it helps to be able to always know which is which. We did not do this in the examples shown in this section.

Interpreting the db2pesrv.log file

The PE Server starts writing to its log files as soon as it begins the startup. The first few lines identify the date and time of the startup, some information about the level of code running on the server, and some information about the location of the other log files. There is a separator line between each invocation of PE Server so you can easily find the date and time of the startup.

Next in the log is the startup of the individual monitoring threads, one for each monitored instance. Each monitored instance is assigned a number that will be used for all entries in the log file so you can distinguish between the monitored instances. In this example, Example 3-12 on page 129, we are monitoring two instances, so we see two monitoring threads. The monitored instance identifier number will always be included in the log entry so you know which monitored instance is referenced. In this example, the thread ID for JAMINST2 is [6].

When the server completes the startup and logs the `alive` marker in the log, the log will only be updated if there are errors or other important messages.

Tip: A new date and time separator line is added to the log file only as necessary when a new log entry is written, not when the date changes to a new day.

Log subsections

The PE Server has several tasks to do while it monitors your databases. The primary task, or thread, is sometimes seen in log files as MASTER. The other main components have a different heading in the logs. These do not usually log anything other than the startup. If there is a problem starting these threads, the server will not get to `alive` status.

The log shows information about starting the main components of the PE Server:

- PWH: Performance warehouse process executions
- PWHLOAD: Performance warehouse aggregation
- PEXP: Periodic exception processing
- EXCP: Event exception processing

- SNAP: History collection (snapshots)
- PECIM: Operating system data collection

The only other time log entries will be made for these components is if you enable diagnostic tracing.

3.2.3 Troubleshooting the PE Server

As with any product or application, sometimes problems occur. This section describes how to perform basic troubleshooting for PE and suggests some steps to take prior to calling IBM support.

PE Server logs startup information for the server itself, and if it encounters DB2 errors, it will capture them and display in the log as well and indicate what impact that error has on PE. Sometimes a DB2 error might be bad enough for PE Server to stop, but usually it is some sort of communication error between PE Server and monitored instance. One example of this is shown in “Scenario: Failed startup” on page 130. In that scenario, the resolution of the problem was to start the instance, and PE recovered automatically.

Common problem scenarios

While we cannot suggest every possible problem situation you might encounter, we describe several common issues and resolution steps that we encountered during writing this book.

Monitored instance stops while PE is monitoring it

If the monitored instance becomes unavailable for any reason (for example, force applications all, db2stop, a crash, and so on):

- PE Server reaction:
  - PE Server will display the DB2 errors in the db2pesrv.log.
- DBA/PE admin response:
  - Restart the monitored instance. PE Server will automatically reconnect.

Password expired on monitored instance

If the password expires on the monitored instance:

- PE Server reaction:
  - PE Server will get a DB2 error, and display in the db2pesrv.log, db2pesrv.trc, and monitored instance db2pesrv.trc files.
  - PE Server will terminate processing for the monitored instance.

- DBA/PE admin response (these steps also apply if you just need to change the password before it expires):
  - Change the password on the monitored instance.
  - The password is also stored (encrypted) in the PE Server database so the server can connect to the monitored instance. To change it there, you need to run **peconfig** and use the CHANGE command to update it.
  - Restart the PE Server so it can start using the new password. With V2.2 Fix Pack 1, you do not need to restart the whole PE server; instead, you may just restart monitoring of the instance. See 3.1.5, “New PE server configuration features of V2.2 Fix Pack 1” on page 79.

Important: Because PE Server stores the password for its own use and even though it is stored encrypted, you should ensure that you have the appropriate security assigned for the PE Server databases, and use an appropriate user ID for PE Server to connect.

We also recommend that you create a new user ID just for the PE Server, and assign it the correct OS and DB2 privileges. Do not use the instance owner user ID, or your own user ID.

Capturing detailed server traces

If there is some problem that you still cannot figure out from the logs, PE provides a facility to generate more detailed trace information. Generally, these are enabled only as directed by IBM support, but you can still run them yourself.

Tracing is available for the PE Server, the peconfig program, and the PE Client. We describe each procedure.

Tracing peconfig

The peconfig program keeps a trace file called peconfig.trc. To capture more detail in this file, you must edit the db2pesrv.prop file in the PE Server working directory (not the one in the bin directory). Uncomment the trace lines:

```
tracerouter.level=5
```

and add the CONFIG component to the component line:

```
tracerouter.component=config
```

The next time you start **peconfig**, the detailed trace is generated. This is also true if you are using the silent config, but the trace file is pesilent.trc.

A sample db2pesrv.prop file is shown in Example 3-14 on page 135.

Example 3-14 db2pesrv.prop file showing tracing options

```
################################################################
#
#  Licensed Materials - Property of IBM
#
#  5724-F89 5724-F90
#  (C) Copyright IBM Corporation 2002, 2005
#
#  All rights reserved.
#  US Government Users Restricted Rights -
#  Use, duplication or disclosure restricted by GSA ADP
#  Schedule Contract with IBM Corporation.
#
################################################################
#---------------------------------------------------------------
# The following properties determine tracing options.
#
#    tracerouter.level=[0..5]
#     Specifies one of the levels of debug information.
#     0 does not produce any debug output and
#     5 the maximum detailed information.
#
#    tracerouter.component=[all,master,snap,pwh,pexp,excp,config]
#     Specifies one or more of the program components
#     activated for tracing, components must be separated
#     by comma.
#     all       all components
#     master    the PE master component
#     snap      the snapshot history component
#     pwh       the performance warehouse server component
#     pexp      the periodic exception component
#     excp      the exception event component
#     config    the PE configuration component
#
#---------------------------------------------------------------
#
tracerouter.level=5
tracerouter.component=config
#
#
#
#---------------------------------------------------------------
# The following properties are only eligible for deadlock event
# monitoring, if non-shared file system support was configured
# for an instance to be monitored.
#
# Property "excp.pctdeactivate":
#    It defines the percentage how full the tablespace must be
```

```
#    before the event monitor writing into table automatically
#    deactivates. The value can range from 0 to 100. The default
#    value is 90 (percent). This is for DMS tablespaces only.
#    Property "excp.pctdeactivate" can be used to define the value
#    for all instances being monitored, property
#    "excp.pctdeactivate.<database alias>" can be used to define
#    the value for a particular database alias as defined in the
#    Performance Expert Server configuration.
#
# Property "excp.tablespace":
#    It defines the tablespace to be used for tables created by
#    the event monitor in the monitored databases. By default
#    tablespace USERSPACE1 is used.
#    Property "excp.tablespace" can be used to define the value
#    for all instances being monitored, property
#    "excp.tablespace.<database alias>" can be used to define
#    the value for a particular database alias as defined in the
#    Performance Expert Server configuration.
#
#-----------------------------------------------------------------
#
#excp.pctdeactivate=90
#excp.pctdeactivate.<database alias>=90
#
#excp.tablespace=USERSPACE1
#excp.tablespace.<database alias>=USERSPACE1
#
#
#
#-----------------------------------------------------------------
# The following property defines the maximum number of user exits
# which can be invoked concurrently for exceptions detected by
# the PE server. Exceptions are denied as long as the maximum
# number of parallel user exit threads are active.
# Min=1, max=100, default=10.
#-----------------------------------------------------------------
#
#excp.max_num_userexits=10
#
#
#
#-----------------------------------------------------------------
# The following property is used by the Performance Expert Server
# configuration. It determines whether the defined shared folder
# for event monitoring is to be checked for accessibility before
# a remote DB2 instance is enabled for monitoring.
#
#    eventmonitor.checkfolder  [yes, no]
#     yesdefault value, verification is performed
```

```
#      no no verification of the shared folder
#
#-----------------------------------------------------------------
#
#eventmonitor.checkfolder=no
#
#
#
```

Enabling server traces

To capture detailed trace for the PE Server, use the PE Client. In Figure 3-51, we show the interaction between the client diagnostic check boxes and the resulting output in the PE Server db2pesrv.trc file in the monitored instance working directory path. Each trace component prefaces the log entry with its components identifier. For example, if you trace the Event exception processing, the relevant lines in the db2pesrv.trc file will start with EXCP.

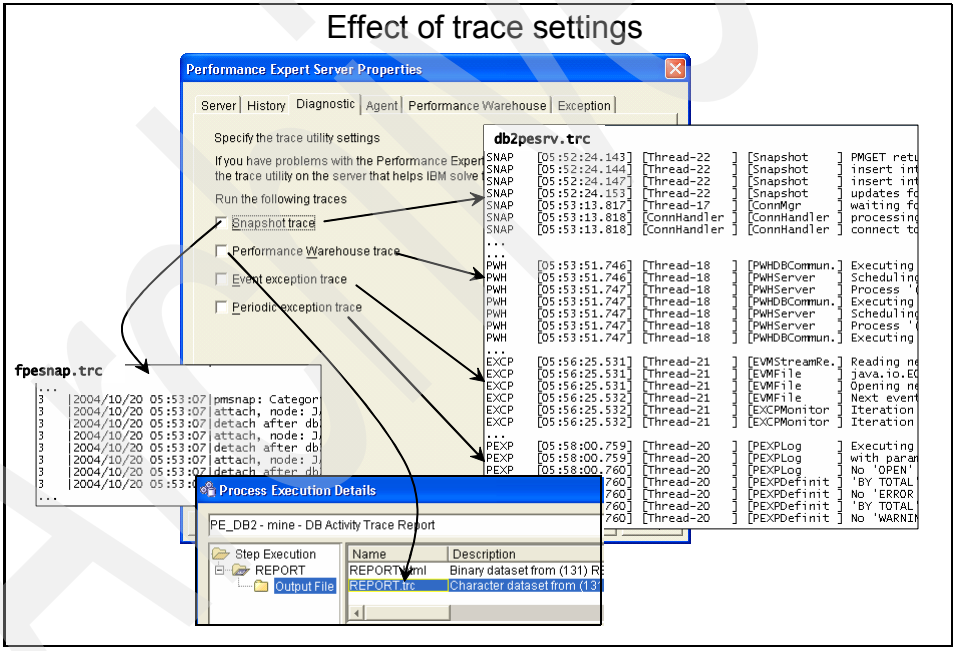


Figure 3-51 Effect of trace settings

Be aware that enabling these traces can generate a lot of log data, and can add overhead to the PE Server processing. You should not enable the trace unless you need it, and you should disable it when you have finished analyzing the problem. You should also only enable the trace for the component you are analyzing - this makes the logs much easier to read.

The trace data is configured per monitored instance, so if your PE Server monitors many instances, you can enable the trace on a single instance. If you tried to use the db2pserv.prop file to control tracing, this would enable tracing for ALL the monitored instances - another reason to use the GUI to control tracing.

Note: If you enable the Performance Warehouse trace, a separate trace file is generated with every PWH report that executes. This trace is actually stored in the performance database, so be very careful when using PWH trace. This is shown in Figure 3-51.

3.2.4 Troubleshooting the PE Client

This section describes the problems you may encounter after installing or using the PE Client. We provide some basic troubleshooting steps for these problems and general steps to produce a diagnostic report that may be requested by IBM support when you contact your IBM representative.

When you encounter a problem, such as when a function does not work, which cannot be resolved by yourself, you can contact IBM support. In some cases, IBM support may need more information and will ask you to create a diagnostic report or trace. The general steps for creating diagnostic report are:

1. From the Client system System Overview panel, select **Help** → **Diagnostics** (Figure 3-52).

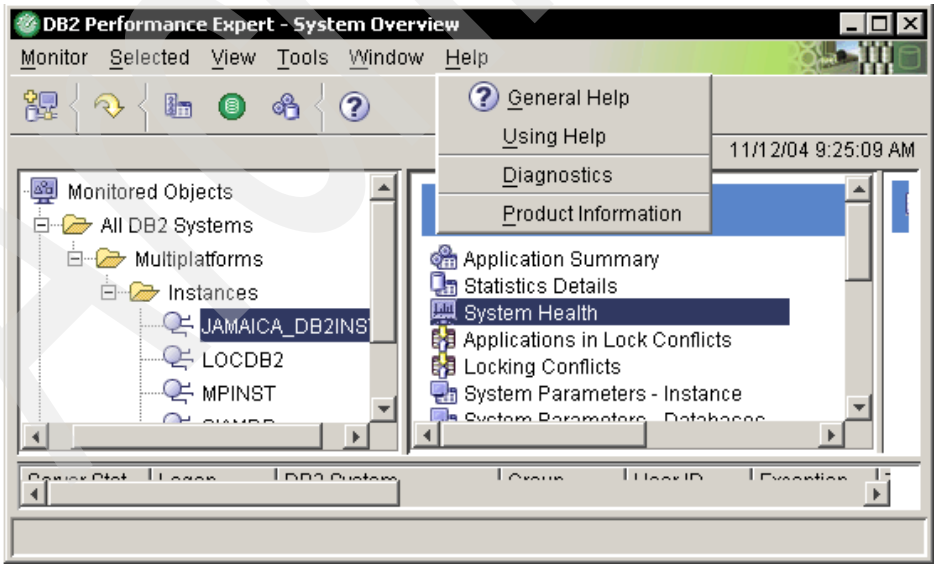


Figure 3-52 Starting PE Diagnostics

2. The Diagnostics windows pops up (see Figure 3-53). Turn the trace on by clicking the Play button ( ).

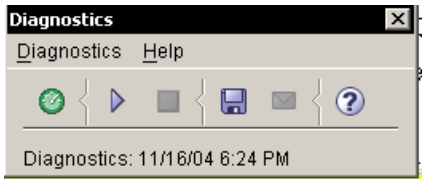


Figure 3-53 PE Diagnostics

3. Perform the steps you wish to debug.
4. Once the problem recreation steps have been completed, push the STOP button ( ) to stop the trace.
5. You can now save the diagnostics report by clicking on the Save button ( ). The save diagnostics window is displayed. In the Scenario field, provide a detailed description of the error and click OK and save the file to a location of your choice on your system. The trace file is stored in the .db2pev2 path on the user's machine. In a Windows environment, it is stored under C:\Documents and Settings\<your id>\.db2pev2.

   If you use a UNIX PE Client, the trace is stored in the .db2pev2 path under your user home directory. Note the path name starts with a dot.

   The client trace file is always overwritten at each start, but if one already exists, you are prompted before overwriting.

   This diagnostics trace file can then be provided to the IBM Support Center for further analysis.

Common problems

We listed here a number of common problems that you might encounter in using PE Client and suggest some problem resolving steps.

You cannot start Performance Expert Client

If Performance Expert Client does not open after you have installed the client code, you may wish to create a trace to investigate the problem. You can then evaluate this trace to determine the cause of the problem or provide the trace to the IBM Support Center for further analysis.

Since PE Client cannot be started, we cannot use the Diagnostic report from the PE Client. The following steps will create a trace report. During the trace, the installed files will also be exhumed and copied again. If the problem is caused by the corrupted files, the trace steps can fix the problem.

1. Open the db2pm.prop file that is located in the bin subdirectory of the installation directory of Performance Expert Client.
2. Change the following attributes in the property file:
   – tracerouter.level:

     This attribute specifies the detail level of trace. The default value is 0; change to 5 as follows:

     ```
     tracerouter.level=5
     ```

   – tracerouter.target:

     The attribute specifies the path and file name in which the trace is to be stored. The default value is stdout. Change to the location of your desire, for example, c:\\tmp\\trace.txt on Windows or $HOME/tmp/trace.txt on a UNIX-based operating system.

   – tracerouter.component:

     This attribute specifies the PE component to be traced. The default value is SYSOVW|GLOBAL. Change to ALL as follows:

     ```
     tracerouter.componenet=ALL
     ```

3. Try to start Performance Expert Client again.
4. Once the trace stops, try to start the PE Client. If the PE Client cannot be started, reinstall PE Client or contact your IBM representative.
5. If the trace is stopped while copying XML files, one or more XML files are in read/write mode. In this case, continue with the following steps.
   a. Open the C:\Documents and Settings\userid\.db2peV2\dgodata directory on Windows or the $HOME/.db2pev2/dgokdata directory on a UNIX-based operating system.
   b. Ensure that all XML files are copied to the dgokdata directory the next time you start Performance Expert Client. To verify that all the XML files are copied successfully, you can compare the install directory c:\Program Files\IBM\IBM DB2 performance Expert V2\dogkdata with the C:\Documents and Settings\userid\.db2peV2\dgodata directory on Windows or the $HOME/.db2pev2/dgokdata directory on a UNIX-based operating system with the one at the PE installed directory.
   c. Open the db2pe.prop file that is located in the bin subdirectory of the installation directory of Performance Expert Client.

      Append the value of db2pm.olm.level entry with _b, for example:

      ```
      db2pm.olm.level=L1194_b
      ```

   d. Try to start Performance Expert Client again.

e. If PE Client cannot be started, reinstall PE Client or contact your IBM representative.

Before you reinstall, please remove the client first:

- For Windows, use the add/remove feature. For details, see Chapter 5, “Installing Performance Expert Client on Windows systems”, in *Installation and Configuration,* SC18-9191.
- For UNIX, see Chapter 11, “Installing Performance Expert Client on UNIX and Linux systems“, in *Installation and Configuration,* SC18-9191 for the detail steps.

If the reinstallation does not solve the problem, contact your IBM representative.

Starting PE Client after changing the DB2 instance

Performance Expert Server uses a DB2 database to store the performance information for each monitored DB2 instance. These performance databases are under the PE Server instance. PE Client can access these PE performance databases via DB2 client/server accessing facility, that is, the PE Server instance and the performance databases are cataloged under a DB2 instance on the PE Client machine.

After the PE Client is started, when you log on to a monitored instance to access the performance database, PE Client, by default, uses the default instance specified in the environment variable DB2INSTANCE. If you have more then one instance in your Client system, and the default instance is changed, PE Client will not be able to find the catalog entries.

To solve this problem, insert the following command in the db2pe.bat file on Windows or the db2pe script on a UNIX-based operating system, before the Java command:

```
set DB2INSTANCE=new_name
```

where new_name is the name of the DB2 instance containing catalog entries of the performance databases.

Running SQL activity traces concurrently

From the PE Client, you can start several SQL activity traces concurrently. These trace activities are performed at the DB2 server where the SQL statements are run. If the monitored DB2 server does not allocate enough memory for the monitoring, the PE Client side can have a slow response. The memory required for monitoring DB2 is allocated from the monitor heap and controlled by the MON_HEAP_SZ configuration parameter of the database manager. You may need to adjust this parameter if you have started several SQL activity traces concurrently.

The memory required for monitoring the SQL activity varies. Please refer to DB2 UDB manual *IBM DB2 UDB System Monitor Guide and Reference V8.2,* SC09-4847 to evaluate the required size.

Responding to dialog boxes

Performance Expert supports modal dialog boxes, such as message boxes, to which you must respond before you can proceed with any other action. In some cases, the dialog box will be in the background when you have several windows open on your system.

For example, you may be in the Exception Processing window, and you now wish to switch to another dialog window that allows you to do other tasks, such as HELP, or you can be accessing the Application Summary window and you select **Customize Columns**. If you attempt to go back to the Exception Processing window, the system issues an “error/warning audio sound” because there is a hidden modal window that is still waiting on a task. To locate the hidden window, the user needs to do the following:

1. Press Alt+Tab. A dialog box is displayed. The hidden modal window is displayed as a Java cup logo in this dialog box
2. Locate the Java cup logo by pressing the Tab key while holding the Alt key until the Java cup logo is selected.
3. Release the Alt-key.
4. Respond to the dialog box.

Performance Expert Client does not work correctly after startup

If you have a firewall installed on your workstation, ensure that Performance Expert Client has the correct access permissions. Otherwise, the Performance Expert Client will not function. Users may experience a message such as:

```
DGOK3601, logon is not possible, the selected performance expert server is not
running.
```

When viewing the diagnostic traces, there will be messages issued indicating the opening of a socket and pinging of the server.

A user can also ping the server from the command prompt window. If it fails within the command prompt, users will need to verify their permissions for authentication.

4

Features and functions - online and short-term monitoring

DB2 Performance Expert is functionally rich. PE provides you with the capability of online, short-term, and long-term monitoring. In this chapter and the next chapter, we guide you through the options that will help you get the most out of the product, and therefore out of your application.

In this chapter, we focus on PE features and functions that are used to perform online and short-term monitoring. Please note that these features and functions are not limited to only online and short-term monitoring. Long-term monitoring also can utilize these features and functions. Functions that are introduced include:

- System Monitoring
  - Application
  - Statistics details
  - Locks
  - System Health
- Alerts
  - Event Exception Processing
  - Periodic Exception Processing
- Operating System Monitoring

- Operating System Information
- Operating System Status

4.1 Online monitoring

Online monitoring, as the name suggests, is a DB2 Performance Expert feature that provides statistics and data related to the DB2 database and application processes using the DB2 database. It provides the in-depth, detailed data required to observe, supervise, or analyze the different aspects of information about your database. It provides continuous data to examine the systems, instances, and applications currently running, as well as the data needed to investigate past events, past performance problems, and database behavior irregularities.

Online Monitoring is very useful feature, as it helps you:

- Determine if your database server and applications are working efficiently or not; this way, you can determine whether you can improve the performance of your database or application using the database. It provides you all the necessary data to analyze the database performance, and also provides you leads in the right direction.
- Calibrate the instance and system configuration for any change in environment, such as increased workload, additional applications, and so on.
- Analyze user and application activities. Monitoring user and application activities helps in managing the resources in efficient way.
- Monitor the exception conditions proactively.
- Monitor SQL statements, using the information like execution time, read or write activities involved etc.
- Troubleshoot and debug any problems caused by the database or applications using database by checking certain thread activities, application activities, and specific event monitoring.
- Analyze various type of bottlenecks like memory usage, CPU utilization, excessive I/O blocking, number of connections, number of statements etc.
- Monitor hardware and system related problems.

DB2 Performance Expert presents analytical data in an organized fashion. It provides you details in different views that are easy to access and understand. It provides a single view to monitor DB2 system performance at the current time as well as for past times. It not only provides statistical data in detailed form but also provides the graphical representation of continuously updated data for better and easier understanding.

DB2 Performance Expert monitoring is an extension to DB2 snapshot, as it displays DB2 Snapshot data in enhanced views. DB2 Performance Expert monitors different kinds of monitor elements; a few very important types are:

- Counters

  These types of monitors checks for certain activities and give a count of their occurrences, for example, number of times query was executed, number of rows read, and so on.

- Gauges

  These type of monitor elements are used to view the current state of a certain access parameter, for example, size of logs.

- Watermark

  Indicates the highest (maximum) or lowest (minimum) value an element has reached since monitoring was started.

- Information

  Gives detailed information about a monitored object, for example, the user ID of a user run application.

- Timestamp

  It indicates the time stamp of activity occurrence. This can be very important while you are trying to back track some activities, for example, database startup time, shutdown time, and so on.

- Time

  These elements give the amount of time a certain activity took, for example, query execution time, index reading time, and so on.

DB2 Performance Expert monitoring provides the user an option to view monitoring information in multiple time frames. It provides a features to analyze:

- Real time snapshot monitoring

  Real time snapshot, run-time, or online monitor is a picture of database activity at a certain point in time. It shows current activities on your system, which can be quite useful for analyzing and debugging the problems occurring currently on your system. Through real-time monitoring, you can determine whether the current status of a DB2 instance or system is capable of handling the increased workload, or whether the increased load will have significant effect on performance.

  Online monitoring helps you make a real-time diagnosis of your database by providing extensive drill down information about your database. Online monitoring views of DB2 Performance Expert renders the actual data about DB2 UDB databases at run time, which makes it very easy for DBA to pinpoint

and address the actual problem as soon as it occurs. You can use online monitoring features for

- Monitoring database for application information
- Tracing SQL activity
- Gathering statistics details
- Monitoring system health
- Analyzing lock conflicts
- Analyzing deadlocks
- Analyzing system and instance parameters

All the above features will be discussed in detail in this section.

- History snapshot data monitoring

  History snapshot data can be displayed for any particular historical instance of time, which means DB2 Performance Expert has the ability to retain historical data about DB2 database events. This feature provides DBAs a technique to go back in time and determine what happened at a particular time or during a particular period of time. This data can further be used to identify trends and peak workloads and to capture representative database activity during periods of normal and abnormal database activity.

  DB2 Performance Expert is capable of showing history up to the user specified time. DB2 Performance Expert can provide historical data for following elements:

  - Application Summary
  - Instance Parameters
  - System Parameters
  - System Health
  - Locking Conflicts
  - Statistics Details at all sub-element levels, such as database, table, tablespace, buffer pool, memory pool, and SQL statements

In history mode, DB2 Performance Expert provides a slider to help you quickly glance at the snapshot of a particular time frame. The slider is divided into multiple segments, where each segment represents a particular time in history. Figure 4-1 shows the mode and slider.

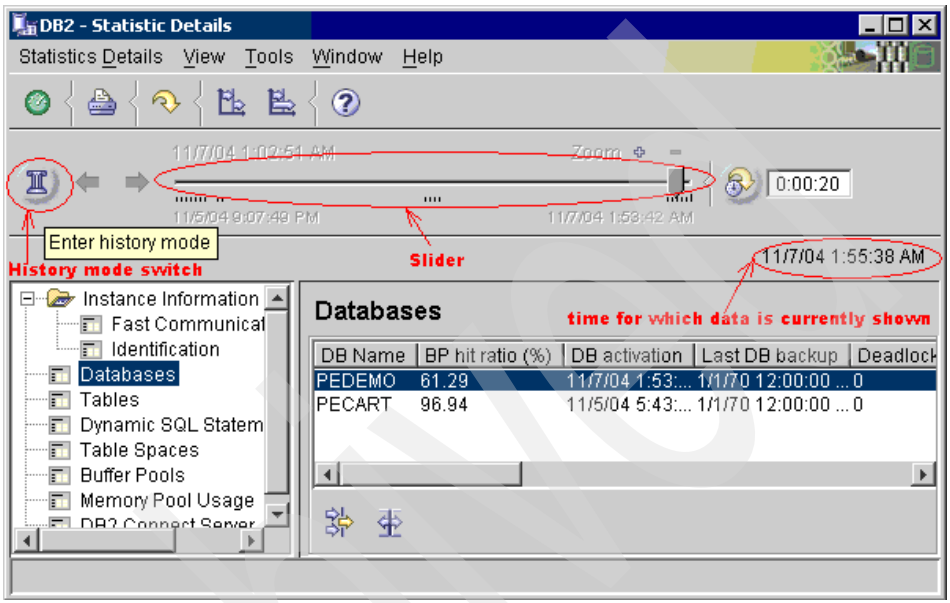


Figure 4-1 History mode and slider usage

New in V2.2 - Operating System monitoring

Introduced in V2.2 of DB2 Performance Expert is the ability to monitor certain operating system statistical information, such as space usage at the file system level, CPU, and memory usage. This feature is available for the AIX and Solaris operating systems as of V2.2.0, with Linux support added in V2.2.0.1. It is implemented via the Common Information Model (CIM) interface. All the same capabilities you have with DB2 snapshot data you also have with the OS data, such as viewing in real time or history mode, using exception processing, System Health charts, and the Performance Warehouse.

4.1.1 Application details

DB2 Performance Expert provides a feature that gives you detailed data from an application's perspective. This helps you keep track of the vital statistics of your database and applications. This data is really useful for debugging applications, benchmarking, finding design flaws and application optimization. Small degradation at the database level or with the database access code can have a high impact on scalability and performance of an application.

The application details feature of DB2 Performance Expert can be used to view the current and recent activity of all active applications connected to a DB2 instance. You can get an overview of all connected applications and detailed information about a specific application.

Viewing application summary

The Application Summary view shows the values of key monitoring elements for all applications connected to the database of the monitored DB2 instances. As shown in Figure 4-2, the elements are in tabular form for all the applications using the databases monitored. A more detailed view for the particular application can be launched by double-clicking the application row.

The Application Summary view can be launched either by double-clicking on **Application Summary** in System Overview window, by selecting **Selected** → **Monitor** → **Application Summary**, or via the context menu.

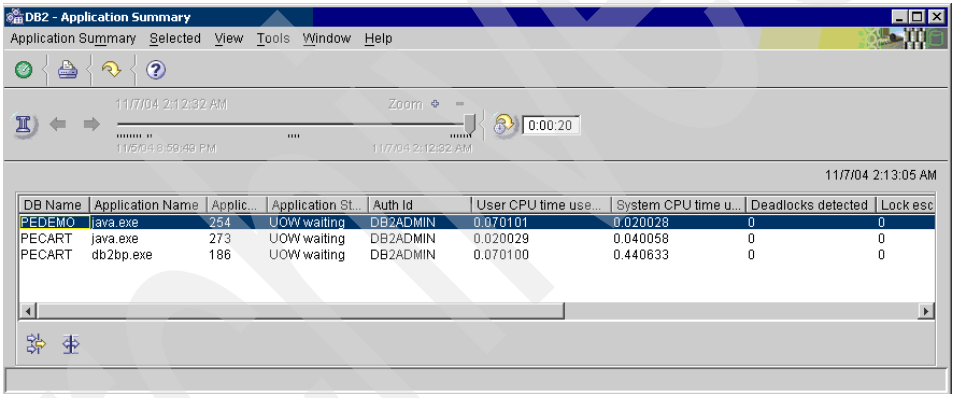


Figure 4-2 Application Summary

The Application Summary view can be used to view important monitor elements. The default list of elements shown in the Summary view are shown here:

- Application Handle

  It is a system wide unique ID for an application.

- Application ID:

  It is network wide unique ID allocated to an application. Its format varies with the communication protocol used.

- Application Name

  It is the name of the application running in a client (as known to the database manager).

- Application Status

  It shows the status of an application at certain point in time.

- Auth ID

  This element shows the authorization ID of the user who invoked the application that is being monitored.

- Auth level

  Shows the highest authority level granted to an application.

- Commit

  This counter shows the total number of SQL COMMIT statements attempted by this application since the connection opened.

- Coordinating Node Number

  In the case of a multi-node system, this monitor represents the node number of the node where the application connected.

- Coordinator pid/thread

  Shows the process ID (UNIX systems) or thread ID (Windows systems) of the coordinator agent for the application.

- Deadlocks detected

  This counter shows the number of deadlocks that occurred for the selected application.

- DB Name

  This is the real database name to which the application is connected (not the alias name.)

- DB Path

  Shows the full path of the location where the database is stored on the monitored system.

- Execution ID

  Shows the ID that the user specified when logging on to the operating system. This ID differs from the authorization ID that the user specifies when connecting to the database.

- Failed operations

  Shows the number of SQL statements that were attempted, but failed.

- Hit ratio (%)

  It is calculated as ((Physical reads - data) + (Physical read - index))*100/((Logical reads - data) + (Logical reads - index)).

- Host execution elapsed time

  Gives the execution time for SQL statement at server side.

- Locks held by appl.

  Represents the number of locks currently held by this application.

- Node

  This monitor shows the node that is running the application.

- Number of Agents

  This counter represents the number of subagents associated with the application.

- PID

  Shows the process ID of the client application that made the connection to the database.

- Seq#

  This counter represents the number of transactions that run, irrespective of whether they are committed or rolled back.

- SQL Statement Text

  Shows the statement that was executing at the time the snapshot was taken. N/P is displayed for statements like COMMIT for example.

- Status Change Time

  Represents the time at which the current status was achieved.

- Successful SQL statements

  Shows the difference between the number of dynamic and static SQL statements that were attempted and the number of SQL statements that were attempted but failed, that is, (Number of dynamic + static SQL statements attempted) - failed statements.

- System CPU time used by agent

  Shows the total system CPU time used b the y database manager agent process.

- User CPU time used by agent

  Shows the total CPU time used by the database manager agent process.

You can use the online help to view the descriptions for any of the values shown on the screen. Press F1 to invoke help.

There are other columns that are not shown in the default view. You should look at the **Customize Columns** window to select or hide other columns, or to rearrange the columns (see Figure 4-3).

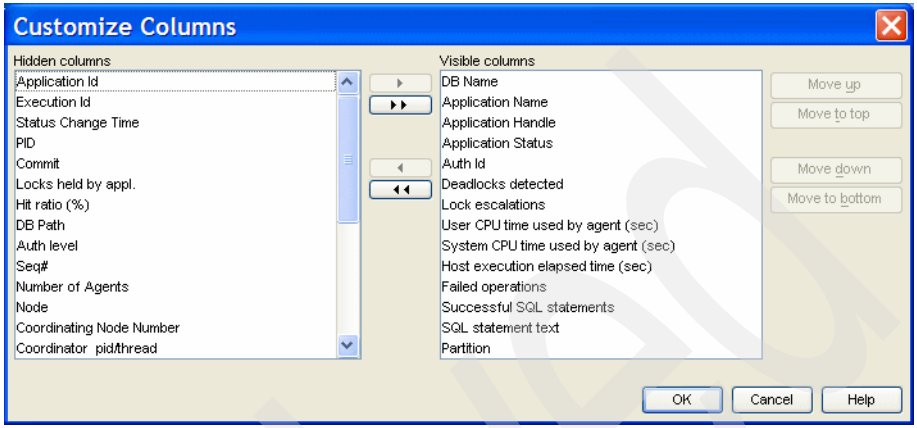


Figure 4-3 Application Summary - Customize Columns

In the case of a multipartition system, you can view the application summary for a partition, group, or global level by selecting it in the **Show Data for** selection box. Figure 4-4 shows the usage of Application Summary in a multipartition environment.

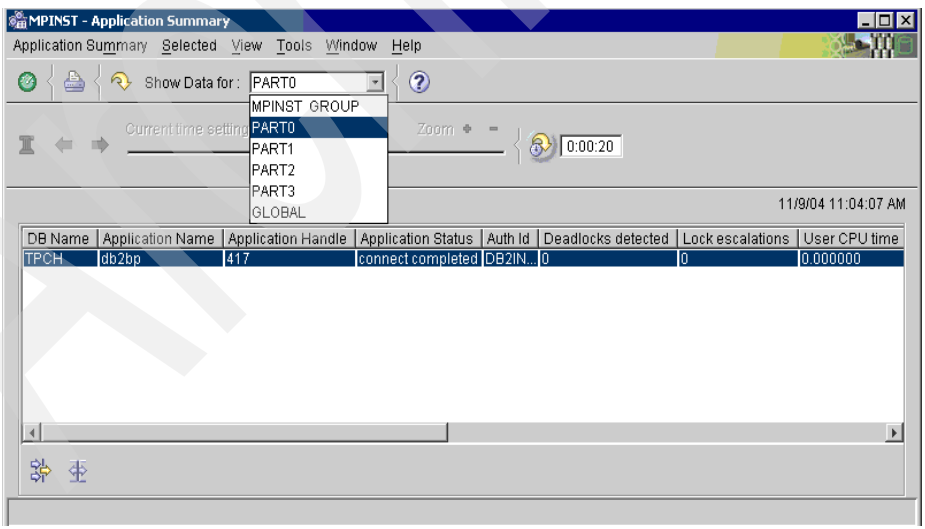


Figure 4-4 Application Summary in a multipartition environment

Forcing an application

In V2.2.0.1, the capability to force off an application was added. This is equivalent to using the FORCE command. You can only force the application from the Application Summary screen. To initiate the force, right-click the application, which will display the context menu. Select **Force Application**, then you are prompted for user ID and password. When you press **OK**, the application displays on the Application Summary screen in italics, with its status changed to “forced”. When you refresh the screen, the application is gone if the force is completed.

Note: The user ID will be pre-filled with the user ID of the application being forced. Most likely this is not the user ID that has the proper authority to force applications. You must enter a user ID that has SYSADM, SYSCTRL, or SYSMAINT authority, and its correct password. If the user ID entered is the application owner ID, SYS* authorities is not required.

See Figure 4-5 for a scenario showing how to force an application.

You can only force one application at a time using the PE interface.

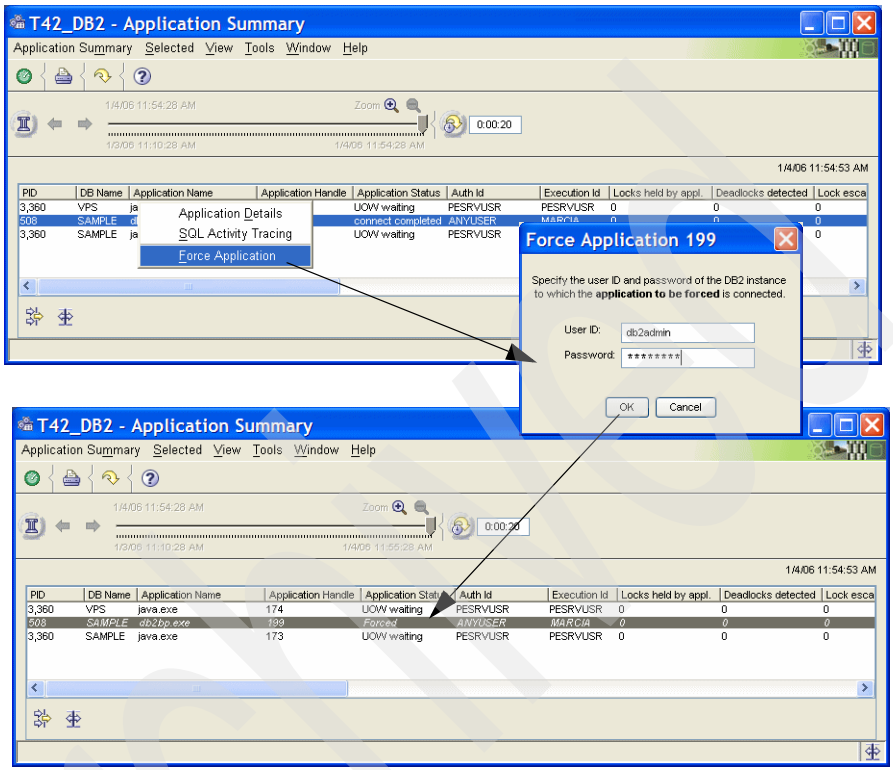


Figure 4-5 Forcing an application

Viewing application details

The Application Detail view provides further more information about an application. This view gives all the information required to analyze applications using a database. It provides many key monitor elements in different forms, which can intrigue DBAs as well as an application developer.

You can launch the Application Details view by double-clicking on the application row in Application Summary view or by selecting **Selected** → **Application Details** in the Application Summary menu or through a context menu.

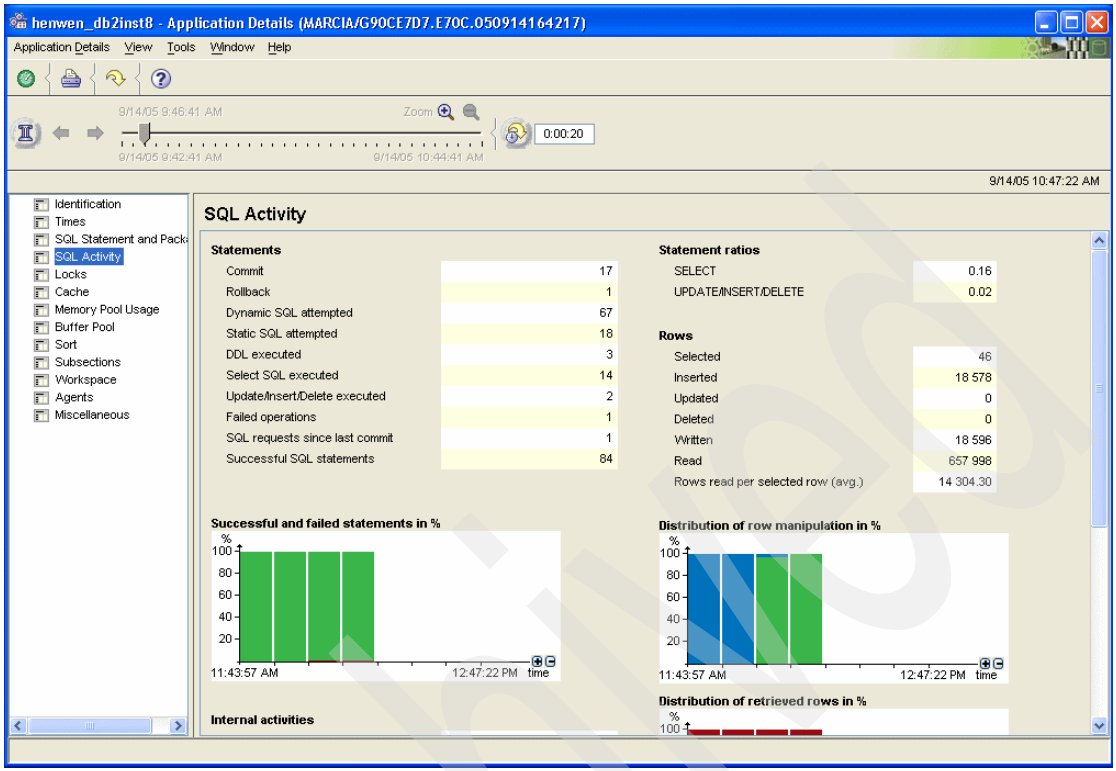


Figure 4-6 Application Details

This view provides a great deal of information in different perspectives or panes. As shown in Figure 4-6, the content pane shows:

- Identification

  This pane provides complete information related to the application; it gives all the names, ports, process IDs, and so on, to identify the application.

- Times

  Provides most of the time monitor counters associated with an application.

- SQL Statement and Package

  This view gives details of the statements executed by the application.

- SQL Activity

  This pane provides details on the SQL statement activity performed by the application.

- Locks

  Provides information about all the locks held and deadlocks detected for the selected application.

- Cache

  This view shows all the cache data for the selected application. It provides information needed to judge if the cache is being used effectively or not.

- Memory Pool Usage

  Provides key data regarding the memory pool used by different application heaps.

- Buffer Pool

  This pane shows information about buffer pools associated with the application. It gives details about buffer pool usage and effectiveness.

- Sort

  This pane can be used to view the sort work performed by the selected application and details on the hash joins.

- Subsections

  This pane provides information about different subsections of the application.

- Workspace

  This pane can be used to view data related to the private and shared work space of the selected application.

- Agents

  This pane provides information about the different coordinator agents, subordinator agent or subagents used by the selected application.

- Miscellaneous

  This provides the rest of the information regarding the selected application, such as inbound communication address, bind/precompile, priority type, and so on.

Note: In some cases, the panes show information in tabular form; to view the details of each row in a table, double-click the row, and the new tab will be added to show you further details associated with the selected item.

4.1.2 SQL activity tracing

SQL Activity Tracing is an activity that is critical to a DBA in order to accurately detect and diagnose all sorts of database issues, such as poor SQL statements, wrong indexing, too many wait events, and so on.

Using this feature, you can collect SQL activity information about selected applications by selecting **Selected** → **SQL Activity Tracing** in the Application Summary view or **Selected** → **Application Details** → **Application Details** → **SQL Activity Tracing** in the Application Details view. This will launch the SQL Activity Tracing - Stop Condition window shown in Figure 4-7.

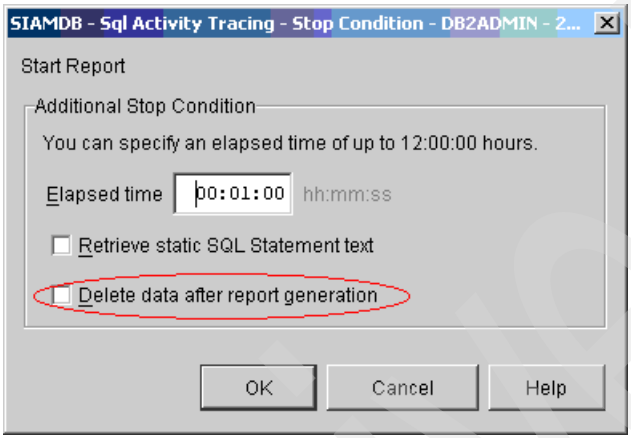


Figure 4-7 Stop Condition window

Internally, DB2 Performance Expert will create a file event monitor in the DB2 database that traces the statement events of the application, such as the start and stop time of the statement, the CPU used, the text of dynamic SQL, and the return code of the SQL statement. This file event monitor is deleted after the trace completes successfully and the trace data is loaded into the Performance Warehouse (PWH) data tables for report generation. The collected trace data can be preserved for later analysis in the PWH data tables or can be deleted by checking/unchecking the **Delete data after report generation** field, as shown in Figure 4-7.

In the same window, you may set the elapsed time for tracing, and the flag to retrieve static SQL statements. This window triggers the SQL activity tracing data collection and open a progress window, as shown in Figure 4-8. Once data collection, loading tables with data, and report generation are completed, you will get a well formatted HTML page with an SQL Activity trace report.

Figure 4-8 SQL activity tracing progress

Note: If you are tracing an application that uses static SQL, you can retrieve the static statement text only by checking the box on the setup screen (see Figure 4-7.) The static statement text is not captured in the event monitor itself, but rather must be looked up in the system catalog. This is a costly activity. The DB2 documentation says “Exercise caution in obtaining static statement text, because this query against the system catalog table could cause lock contentions. Whenever possible, only use this query when there is little other activity against the database.”

The SQL Activity Trace report is generated in HTML format and PE will launch a web browser to display it. In Internet Explorer, this report looks like Figure 4-10 on page 160.

You can configure which browser PE uses by specifying it in the PE Client configuration window. On the System Overview window, select **Monitor** → **Configuration**, then you can set the browser, as shown in Figure 4-9, where we set it to use Firefox. By default, PE will automatically use your system default browser. If you want to use a different browser, you can change it for PE.

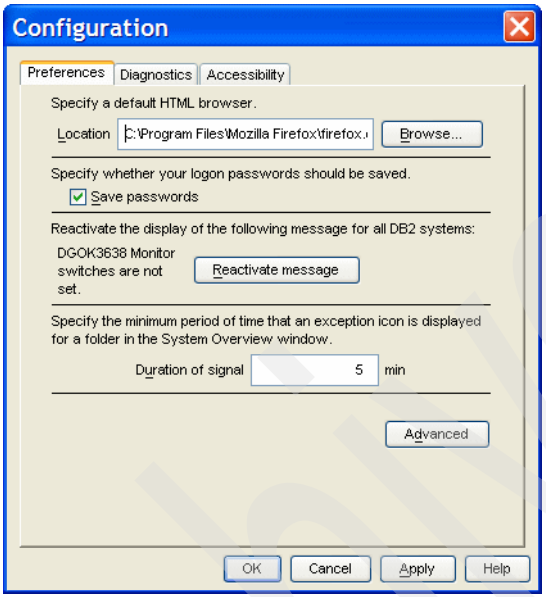


Figure 4-9 Configure which browser to use

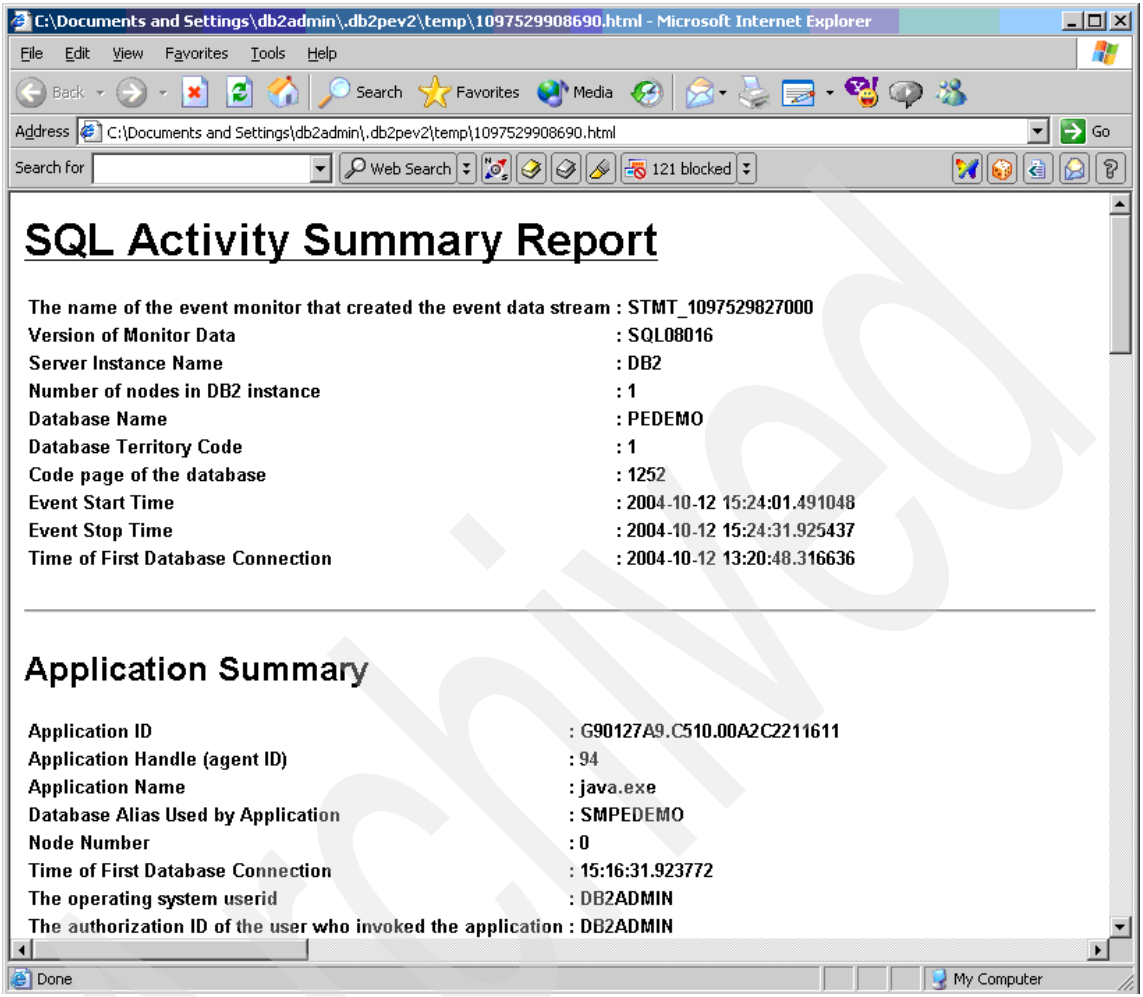


Figure 4-10 SQL Activity Summary Report

The SQL Activity Summary Report reports all the static and dynamic statements of the application and their activities. The data in this report is divided into two sections:

- Application Summary

  This section provides information related to the application on which the report is generated, and lists the statements executed by this application. This list provides a quick tabular glance of statistics for each SQL statement with its data, which includes:

  – Unique statement identifier

- Unique internal identifier for an SQL statement text
- Statement Type
- Type of statement (SQL type)
- Total Execution Time (sec.ms)
- Total number of executions
- Number of SQL errors
- Number of SQL warnings
- Total number of successful fetches performed on a specific cursor
- Total number of rows read from table
- Total number of rows changed in table
- Total Preparation Time (sec.ms)
- Total number of statement preparations
- Total number of errors during statement preparation
- Total number of rows read during statement preparation
- Total Sort Time (sec.ms)

► Statement Operations

The Statement Operation section shows details of each run of the SQL statement, with all the data required to analyze it. The data is in tabular form showing:

- Unique statement identifier
- Unique internal identifier for an SQL statement text
- Statement operation
- Event Start Time
- Total Operation Time (sec.ms)
- Number of Agents Created
- Sequence Number
- Blocking Cursor
- Number of Successful Fetches
- Internal Rows Deleted
- Internal Rows Inserted
- Internal Rows Updated
- Rows Read

- Rows Written
- SQL code
- SQL state
- System CPU Time (sec.ms)
- User CPU Time (sec.ms)
- Sort Overflows
- Total Sort Time (sec.ms)
- Total Sorts
- Buffer Pool Data Physical Reads
- Buffer Pool Data Logical Reads
- Buffer Pool Index Physical Reads
- Buffer Pool Index Logical Reads
- Physical read requests into the temporary tablespace
- Logical read requests into the temporary tablespace
- Physical read requests into the temporary tablespace
- Logical read requests into the temporary tablespace

4.1.3 Statistics Details

Information pertaining to the entire DB2 instance is recorded and viewed in the Statistics Details view. This information is particularly useful for diagnosing the DB2 database activities. Information on the utilization and status of the buffer pools, DB2 locking, DB2 logging, and DB2 storage is shown by this view.

The Statistics Details view provides a comprehensive view of DB2 performance over time. It can be used to monitor various monitor elements or counters at different levels. This helps you watch server loading for performance, availability, and capacity planning.

Gathering and showing of Statistics Details data can be done in three different modes:

- Regular processing

  This is the default whenever you open a statistics window. The values displayed are accumulated from the start of DB2. This mode is selected by default, and is active if **View** → **Delta** is not selected.

- Interval processing

  The interval mode allows you to isolate DB2 statistics activity beginning at a specific time. In this mode, the delta values accumulated since the start of interval processing are shown for some columns. When you stop interval processing, all the subsequent snapshots are shown in regular mode again, that is, as accumulated since the start of DB2. This mode can be selected by selecting **View** → **Interval**.

- Delta processing

  The delta mode allows you to isolate DB2 statistics activity between two snapshot refreshes. It can help you discover a problem when running exception processing. In this mode, all statistics windows show the delta between the values of two consecutive snapshots. This means accumulation starts at zero after each snapshot. When you stop delta processing, all the subsequent snapshots are shown in regular mode again, that is, as accumulated since the start of DB2.This mode can be selected by selecting **View** → **Delta**.

In all modes, statistics values are updated each time a refresh occurs.

As shown in Figure 4-11, Statistics Details is divided into different categories by using panes; each pane shows data related to object associated with it.

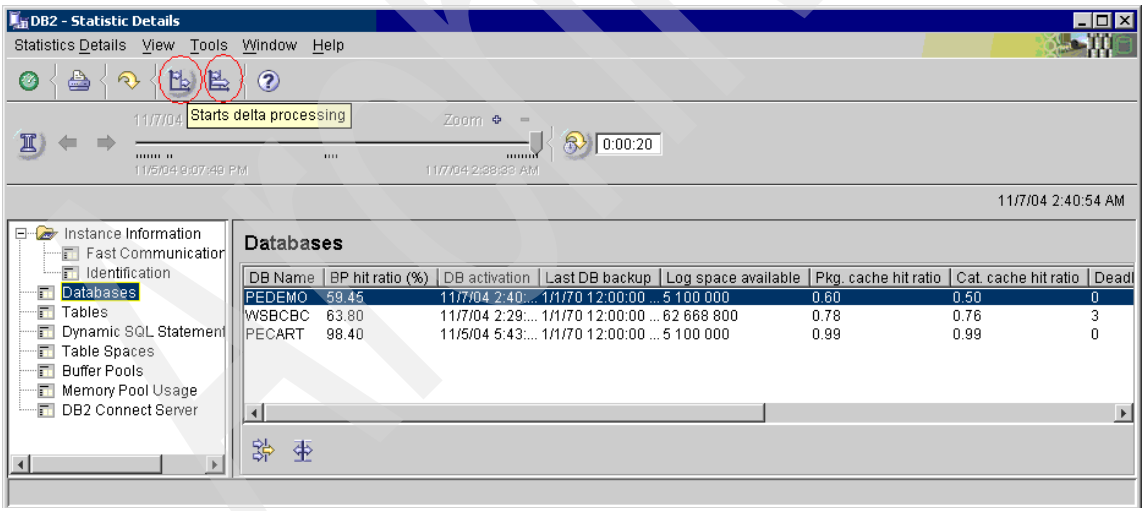


Figure 4-11 Statistics Details

- Instance Information

  This pane can be used to view information about the sort work performed by the DB2 instance being monitored and about its connections and agents. It shows the following information regarding the instance.

  - Connections
    - Current connections
    - Attempted connections for DB2 Connect
    - Current connections for DB2 Connect
    - Connection waiting for host reply
    - Connection waiting for client to send request
    - Remote connections to DBM
    - Remote connections executing in the DBM
    - Local connections
    - Local connections executing in the DBM
    - Local databases with current connects
  - Sorts
    - Total heap allocated
    - Hash join threshold
    - Post threshold sorts
    - Piped sorts requested
    - Piped sorts accepted
    - Sort private heap high water mark
    - Piped sorts (%)
  - Agents
    - Agents registered
    - Agents waiting for token
    - Maximum agents registered
    - Maximum agents waiting
    - Committed private memory
    - Agents assigned from pool
    - Agents created due to empty pool
    - Stolen agents

- Agents created (%)
- Maximum coordinating agents
- Connection switches
- Total inactive DRDA® agents
- Idle agents
- Maximum agent overflows

Under Instance Information, there are two sub categories:

– Fast Communication Manager

The Fast Communication Manager pane of the Statistics Details can be used to view statistics on Fast Communication Manager (FCM). You can get FCM information about each partition of a database manager instance by double-clicking the instance. It shows the following data about FCM in tabular form:

- Free buffers
- Free buffers low water mark
- Free connection entries
- Free connection entries low water mark
- Free message anchors
- Free message anchors low water mark
- Free request blocks
- Free request blocks low water mark
- Instance
- Partition

– Identification

The Identification section of the Statistics Details shows identification and status information about the DB2 server. It shows the following monitor elements:

- General, which includes:

  Database manager type at monitored (server) partition

  Server version

  Time zone displacement

  Snapshot time

  Partition number

Server product/version ID

Configuration name at monitoring (server) partition

Server instance name

Start database manager time stamp

- Instance information, which includes:

  Last reset time stamp

  Partitions in instance

  Status of database (textual)

► Databases

This pane shows statistics information about different databases being monitored by the configured server. It shows data in tabular form corresponding to each database in this instance.

You can double-click each row and get further statistics information about the selected database. This information is further divided into different panes categorizing information according to different components:

– Buffer Pool and Details

  This view provides statistics information about the buffer pool activity for the selected database.

– Application

  This view gives statistics information about the applications connected to the selected database.

– Locks

  This view provides information about the locks held by and deadlocks detected for the selected database.

– Logs and Log Details

  This Statistics Details window provides information about logging for the selected database. The Log Details window shows the first/current active/archive log file numbers.

– SQL Activity

  This view provides statistics related to SQL statements activities for this database.

– SQL Workspaces

  This view provides SQL private and shared work spaces information, including maximum size, overflow, section lookup, and section insert.

- Cache

  This view provides the statistics about the package and catalog caches of the selected database.

- Space Management and Storage Paths

  This view provides information about the amount of allocated and free space for all tablespaces.

  If your database is using Automatic Storage (available after UDB V8.1 FP9), the information for the storage paths is shown in the sub-pane of Space Management.

- Sort/Page cleaners

  This pane provides the information related to the sort work performed by the selected database, information about the hash joins and page cleaners, and database status information.

- Rollforward

  This view shows data to monitor the progress of the recovery of the selected database.

- High Availability Disaster Recovery (HADR)

  This view provides details on the high availability disaster recoveries performed for the selected database. This only has data if you have HADR set up and configured.

- Tables

  The Table statistics pane shows the information about all the active tables the selected database contains. It provides statistics like Overflowed Rows, Page Reorg, Partition, Rows Written, Rows Read, Table File ID, Table Name, Table Schema Name, and Table Type in a tabular form. You can get this information in a page form for a table by double-clicking the row.

- Dynamic SQL Statements

  The Dynamic SQL Statements pane of the Statistics Details window shows statistics on the SQL statements stored in the DB2 statement cache. The DB2 statement cache stores the SQL statements that are frequently used. The information in this pane helps you identify the dynamic SQL statements that are executed most frequently and the queries that consume most of the resource. In addition, it helps you determine if SQL tuning can improve database performance.

It shows the information about each dynamic SQL statement in tabular form, which can be viewed in page form by double-clicking any row. The table in this view shows the following information:

- DB name
- Statement
- Executions
- Elapsed exec. time
- Avg. time per exec.
- Worst prep. time
- Best prep. time
- Sorts
- CPU per stmt.
- CPU per user stmt.
- Rows read
- Rows written
- Int. deleted rows
- Int. inserted rows
- Int. updated row

- Tablespaces

  This view provides the list of tablespaces used by the databases under the monitored instance. It provides information on each tablespace, such as page size, type, and content type. It also shows space usage information. By double-clicking the tablespace, it shows further details about the selected tablespace in the following panes:

  - Access

    It provides statistics of the activities of the selected tablespace. This information includes read and write counters, average synchronous and asynchronous read/write monitor values, buffer pool utilization, disk access, and so on.

  - Configuration

    This pane provides configuration information related to the selected tablespace, including extent size, prefetch size, page size, and so on.

– Space Management

  Use this pane to view the amount of allocated and free space for the selected tablespace. Details about automatic storage are also shown on this pane.

– Containers

  Use this pane to get an overview of the containers in the tablespace, including actual space allocated and used in storage device. You can see more information about a particular container by double-clicking the row.

– Partitions

  This pane provides a list of partitions on which the selected tablespace is active and an overview of the activity of this tablespace on each partition. Double-clicking a partition shows further details about a partition in different panes, which includes:

  • Information per Partition Details

    This pane shows details on the activity of a tablespace on the selected partition.

  • Space Management

    This pane shows the amount of allocated and free space for the tablespace on the selected partition.

  • Containers

    This pane can be used to get an overview of the containers in the tablespace. You can see more information about a particular container by double-clicking the selected row.

  • Quiescer Activity

    This pane shows tablespace quiescer activity on the selected tablespace in the selected partition.

  • Range Status

    This pane shows ranges or stripe sets used in the tablespace map on the selected partition. This map is used to map logical tablespace page numbers to physical disk locations. It is made up of a series of ranges.

- Buffer Pools

  This pane provides statistics details about the buffer pool used by the selected database. This information table contains data like buffer pool name, hit ratio, data hit ration, index hit ratio, average physical write time, and average physical read time. On double-clicking the selected buffer pool, you will see the following panes containing the following information:

  - Access

    Provides statistics on the activities of the selected buffer pool. This information includes buffer pool hit ratios, logical and physical read and write counters for this buffer pool, average synchronous and physical read/write time values, buffer pool utilization, IO access, and so on.

  - Configuration

    This pane provides configuration information related to the selected buffer pool, including current size, new size, tablespace mapped, pages left to remove, and so on, in tabular form. Double-clicking the buffer pool information for a particular partition shows the same information in page format.

- Memory Pool Usage

  This pane provides statistics on the different memory pools used for the monitored instance. In tabular form, it provides such information as the number of fragments, maximum fragment size, and current size of memory pool for each memory pool. It provides information for each of the following memory pools:

  - Backup/ Restore/Util heap
  - Catalog cache heap
  - Package cache heap
  - Buffer pool heap
  - Database monitor heap
  - Database heap
  - Lock manager heap
  - Other memory

  You can see further details of each memory pool by double-clicking on the row. In the details page, you will see the details about the selected memory pool and each fragment of the selected memory pool.

- Utility Information

  Use this pane to view details on selected utilities, such as Backup, Rebalance, Runstats, Reorg, and so on.

- DB2 Connect Server

  If you are sending DB2 Connect monitoring information to a multiplatform PE server, you can view information about the DB2 Connect server here. (This would be a very rare case. Normally, you send DB2 Connect monitoring information to a PE server running on Z/OS.)

4.1.4 System Health

A picture is worth a thousand words. This is especially true in the DBA world, where the size of performance data and reports is huge. The analysis of this monitoring data is not easy, as information gathered by 24x7 systems is always increasing in a linear fashion. The same data, if represented in graphical form, can show values that the DBA can analyze more easily.

The System Health feature of DB2 Performance Expert does the same thing by providing data in graphical form. It is a diagnostic and visualization feature that allows DBAs to isolate and resolve database performance issues quickly. DBAs can find problems with the database or system at a glance in the data view pane and fix it.

Using System Health, you can monitor performance counters over time instead of looking at individual monitor values. You can define a threshold for each counter you are interested in and you can specify the format in which you are interested to view the data. This is done by creating customized data views for different monitors of interest. Each data view consists of a chart showing the values of the selected counters, a legend, and a table showing the value over time. Each data group can contain a different selection and arrangement of data views. Figure 4-12 on page 172 is a sample System Health view. You can define your own data views or use predefined data views as templates.

You can combine related charts into a named Data Group. The Data Group can be expanded or collapsed in the System Health window for easier viewing. The folder hierarchy has been simplified in V2.2.0.1, with the removal of the extra Data Views folder. Now you see the individual data views directly under the Data Group, as shown in Figure 4-12.

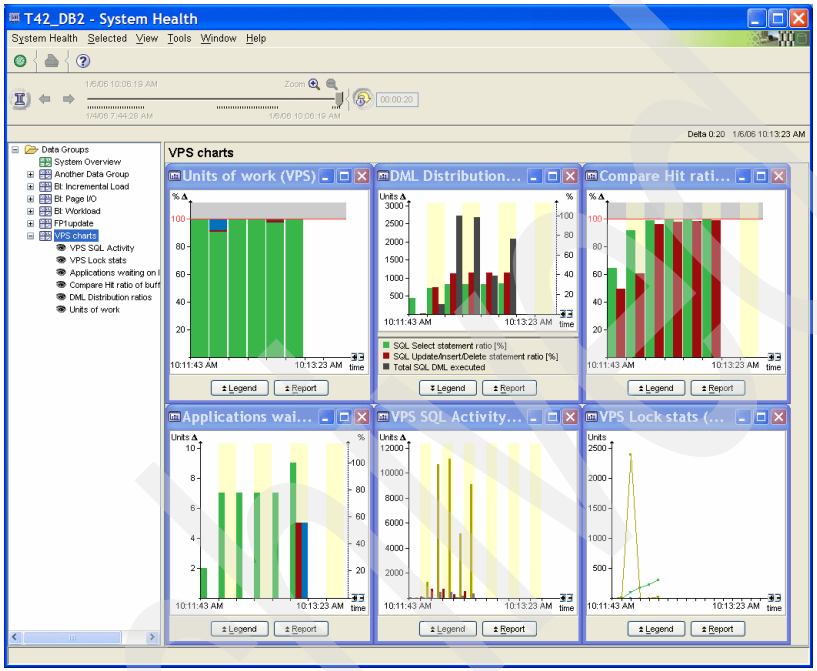


Figure 4-12 System Health

Using predefined data views

DB2 Performance Expert provides predefined data views designed to cover common or complex monitoring methods and can, therefore, only be changed to a limited degree.

You can use these predefined data views directly by opening a predefined view by selecting **Selected** → **Open Predefined Data View**, which will open an input window, as shown in Figure 4-13 on page 173. In this window, select the data view you want and click **OK**.

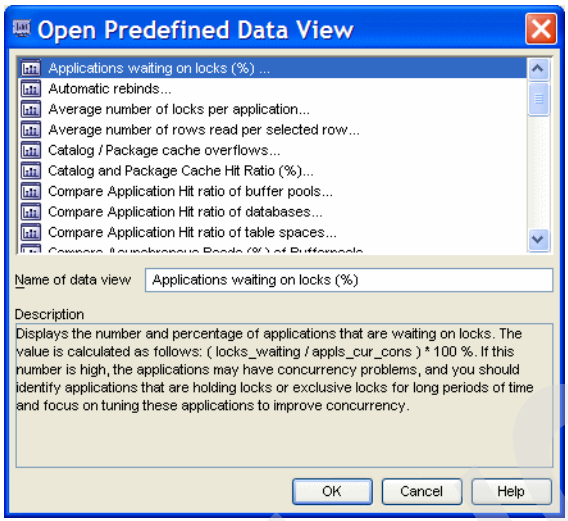


Figure 4-13 Using predefined data views

You can also use the predefined data view template to create a new data view. This can be done by creating a new chart based on a predefined data view, then renaming the data view by right-clicking it. You can also change the properties, such as the chart type or threshold values.

Creating new data views

Sometimes a DBA is required to keep a track of certain high alert monitor values. In those cases, he can create data views for that particular monitor value. This can be easily done by performing the following steps:

1. Select **Data Views** under the data group in which you want to create the new data view.
2. Right-click it and select **New** or select **Selected** → **New**.

3. Select the **Data view category**, which will define the level you want to monitor. There are several categories to choose from, but the most-used category will be “Statistics Details, Databases”. If your monitored system is CIM-enabled, you can also choose the OS-level elements for your chart, such as space used for a particular file system (see Figure 4-14).

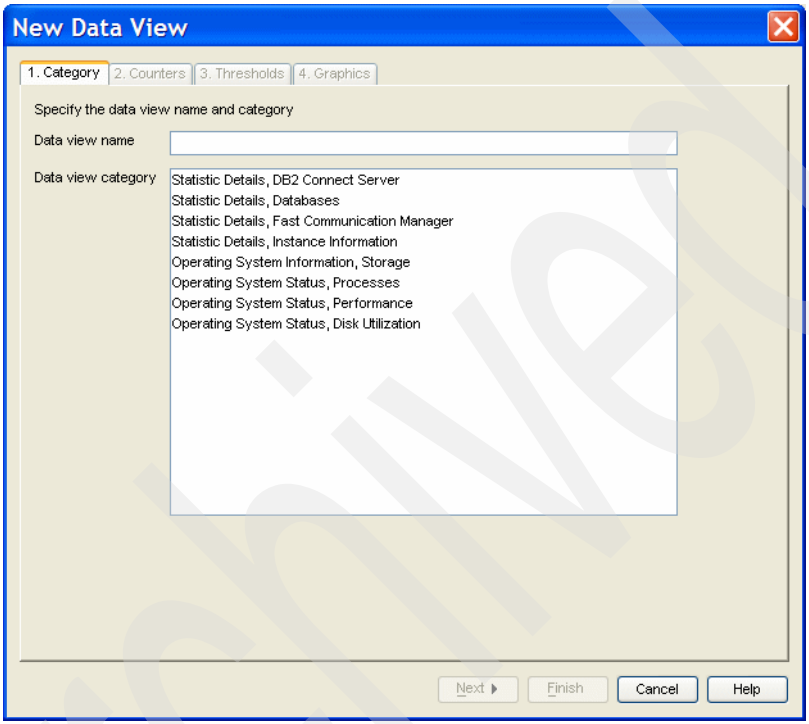


Figure 4-14 New Data View dialog

4. Select the counters and database to be monitored.
5. Provide the threshold values.
6. Finally, provide the information about what type of chart you want your data to be presented in.

Once you click **Finish**, you will see your data view in your folder pane, and a graph corresponding to it in the data pane.

You can now start receiving data for the data view by clicking the **Start and stop receiving data** button, as shown in Figure 4-15 on page 175.

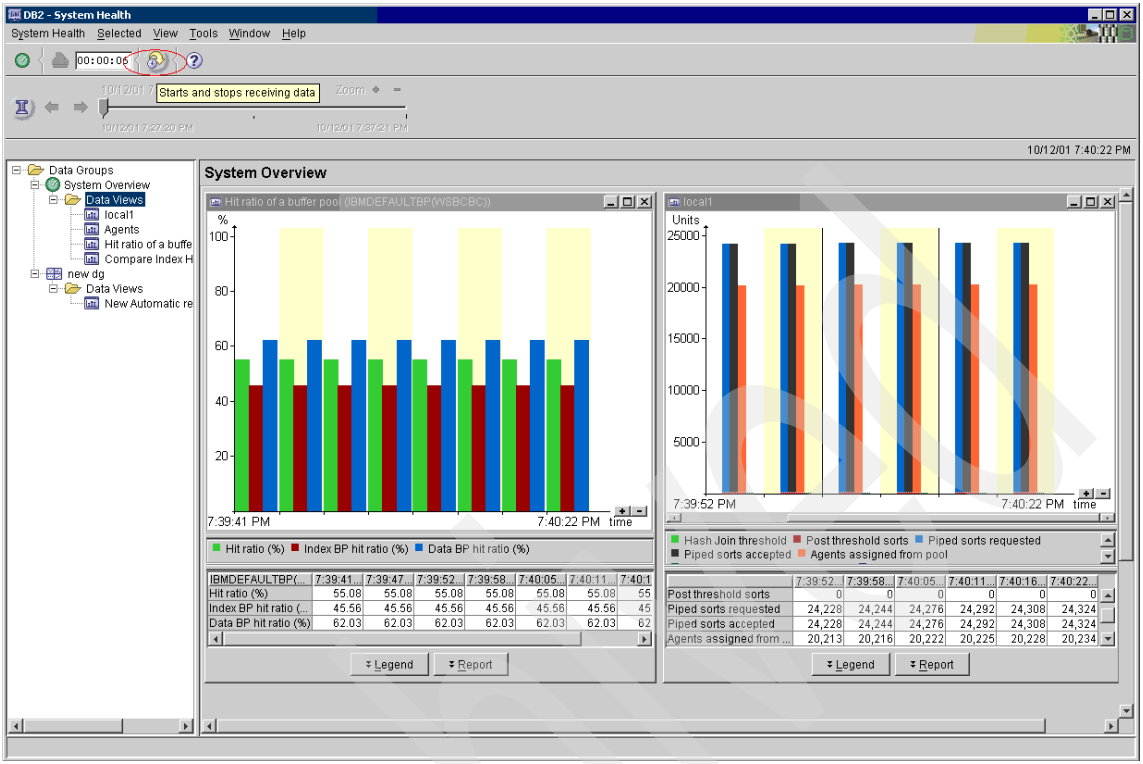


Figure 4-15 System Health view

Select this button ( ) to start automatically collecting monitor data at the specified interval, and the data will be presented in graphical form, as shown in Figure 4-15.

Displaying charts on System Overview

In V2.2.0.1, the method for displaying charts on System Overview window is revised. Previously, you had to make a copy of the chart and paste it onto the System Overview data group. In V2.2.0.1, you only have to set a property on the chart to display it. There is no copy and paste involved, and the original chart is preserved in the original data group.

To display a chart on System Overview, first create the chart in another Data Group. Then right-click the chart, and select **Display in System Overview** from the context menu (see Figure 4-16).

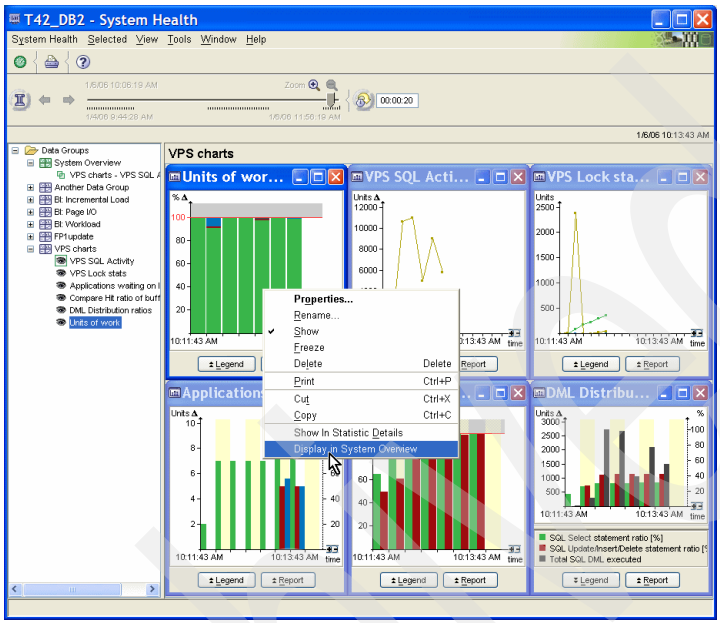


Figure 4-16 Display chart on System Overview

Refresh rate

An enhancement in V2.2.0.1 is the ability to control the refresh rate of the charts on the System Overview window. In earlier releases, the refresh rate was always 20 seconds. Now you can configure this by changing the value on the System Health window. The same value applies to all charts per monitored instance. If you monitor multiple instances, you may have different refresh rates for each instance.

The default refresh rate is six seconds, which is very frequent. It can be very expensive to collect snapshots at that frequency and calculate the chart data. We recommend that you change the refresh rate to at least 20 seconds.

Display charts in History mode

You can also view the charts in history mode by clicking the History Mode button. You can create a chart and view history data for it, even if the chart was not defined when the data was collected.

In Figure 4-17, we highlight several of the features of System Health, particularly in history mode. In this case, we entered history mode, and moved the slider some days back in the past. The data will be available for viewing for the amount of time configured in the history snapshot retention period. We resized the Units of Work chart and clicked the **Report** button so we could see the details behind each plot point. We notice the Delta value is shown as five minutes, which is the history snapshot frequency value configured for the PE server. For more information about the history configuration settings, see 3.1.1, “PE Server configuration and related tasks” on page 47.

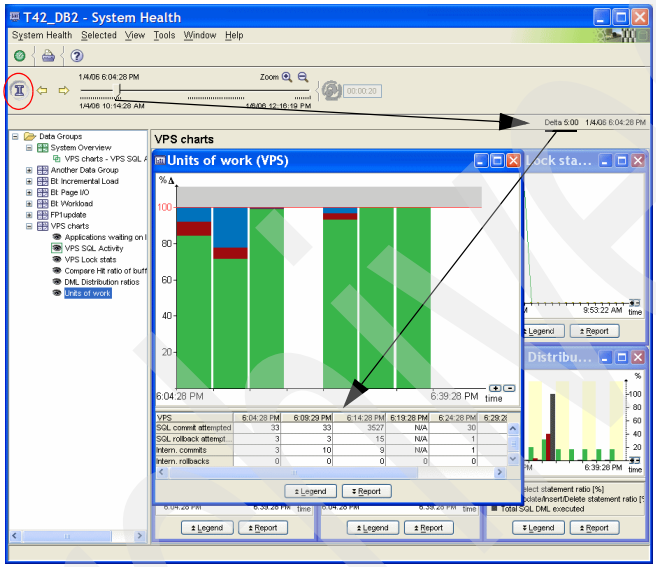


Figure 4-17 System Health chart in history mode

Viewing charts in history mode can be very useful, because you do not have to catch a problem exactly when it occurs; you can look back in time to see what happened in the past.

Consult the *IBM DB2 UDB System Monitor Guide and Reference V8,* SC09-4847 and online help for additional details on how to use System Health charts.

System Health charts for partitioned instances

If you are monitoring a partitioned DB2 instance, this window contains a “Show Data for” drop-down list of partitions (see Figure 4-18). To view statistical information about another partition or the entire DB2 instance, click the **Show Data for** arrow and then click the partition or the instance.

We could choose to view only one partition, view all partitions, or view aggregated data over all partitions, by using the drop-down box at the top of the window.

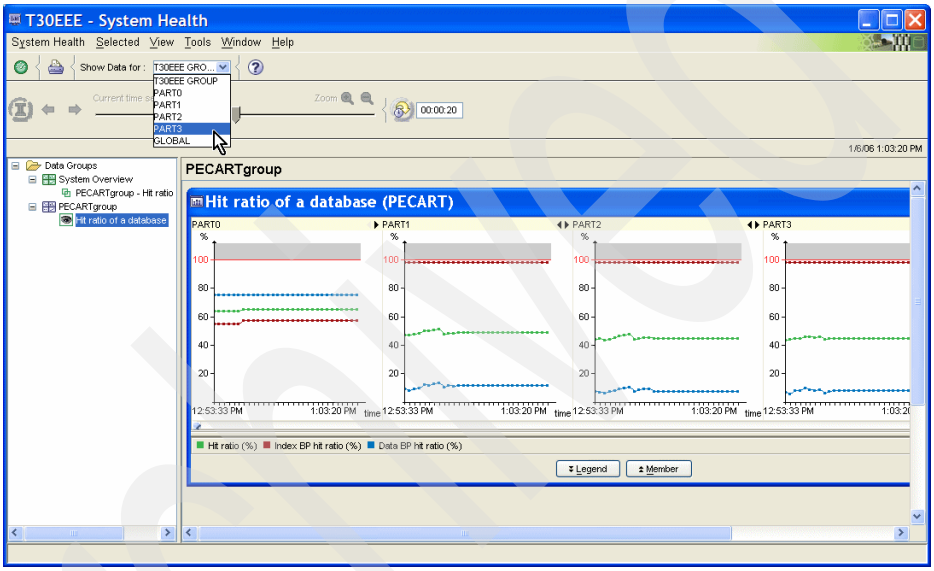


Figure 4-18 View system health statistics for a partition

When you select the GROUP option, PE will attempt to display a chart for each individual partition. If you have a very large number of partitions, this may not be practical. In Figure 4-19 on page 179, we show an example of how you can view the counters across each partition, in our case, we only have four partitions. The graph for partition 0 looks very different from the other partitions. If you see such a deviation, you might want to investigate why the work is not being distributed evenly. In our case, it is because partition 0 is the catalog partition and does not have any data on it. To eliminate partition 0 from this view, click the **Member** button and de-select partition 0.

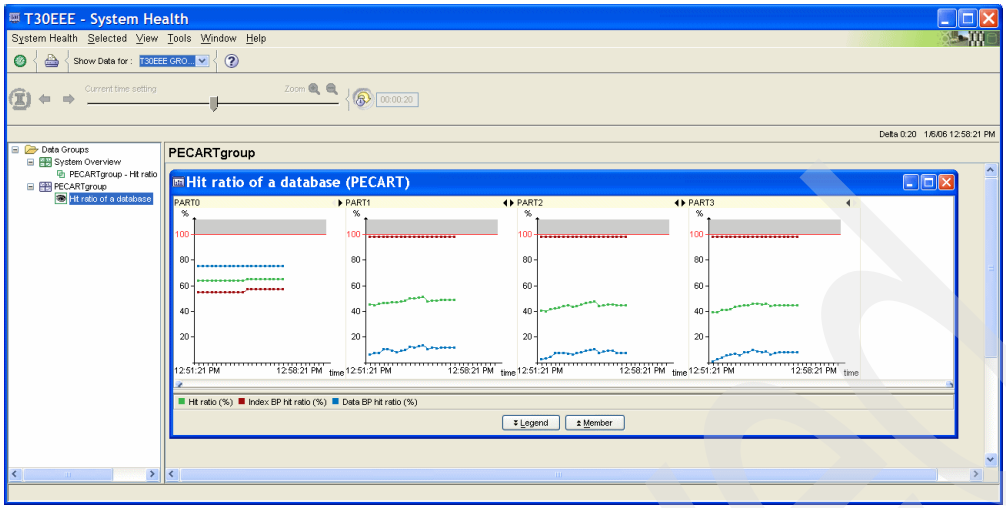


Figure 4-19 Viewing chart for each partition

Importing charts for BI and Content Manager environments

In PE V2.2.0.1, there are sets of predefined data views containing multiple charts for those who are monitoring Business Intelligence (BI) or Content Manager (CM) environments.

The charts were customized by BI and CM experts to identify common performance counters for those environments. If you do not have one of these environments, you may still find the charts useful as you learn PE.

The charts are stored as XML files in the SAMPLES directory under PE Client install location. To use them, you must import them and answer some simple prompts (database name, for example).

Select the **Data Groups** folder, then right-click and select **Import**. This will bring up the dialog showing the predefined data groups. Select one of them and press **OK**. For each chart that needs a specific object (for example, database name or buffer pool), you will be prompted to select the object from a list.

4.1.5 Locks monitoring

Integrity of data is one of the most important aspect of RDBMS. DB2 guarantees this integrity of data by using the locking technique. Locking is a technique that enables you to access data from different applications without interfering in the other's functionality and data changes. It does so by updating particular data one transaction at time. The efficiency of these locking mechanisms can have an important effect on the performance of DB2 applications.

While analyzing the locks, you may be concerned about many factors associated with them. A few of the important ones are:

- Lock Suspension

  This is a typical locking performance issue that occurs when a transaction has acquired a lock and another transaction requests a lock on the same resource. In this case, the requesting transaction must wait. The application is suspended and will stop running until the lock can be acquired. Since the transaction and application is in the wait state, lock suspensions can have a significant effect on application duration and performance. In this section, we discuss how we can use DB2 Performance Expert to look for such locking conflicts and also about applications holding the locks.

- Deadlocks

  Deadlock is a condition in which cyclic dependency for locks occurs between multiple transactions. The deadlock occurs when two or more transactions connected to the same database wait indefinitely for a resource and this waiting is never resolved because each transaction is holding a resource that the other needs to continue. The deadlock problem has a great performance impact for any concurrent scenarios, as the only recovery from this problem is to roll back one of the transactions. We can use the Locks view under Application Details (see 4.1.1, “Application details” on page 148) and the Locks view under the Statistic Details, Database view (see 4.1.3, “Statistics Details” on page 162) to get information about the number of deadlocks that occurred for the selected application or database.

- Timeouts

  When an application has been suspended beyond a predetermined period of time, DB2 terminates and rolls back the transaction. The application will receive an error code. This unexpected termination is termed a *timeout*. The wait time is defined using the DB2 Lock timeout configuration parameter (locktimeout) at the database level. Its value can be anything in seconds or 0 (no waiting) or -1 (wait forever). The value for locktimeout should be set to less enough to quickly detect waits that are occurring because of an abnormal situation, such as a transaction that is stalled. On the other hand, it should be high enough so valid lock requests do not timeout because of peak workloads, during which time there is more waiting for locks.

  From the Locks view under Application Details and the Locks view under Statistic Details, Database view, we can get information about the number of deadlocks that occurred for the selected application or database.

- Lock escalations

  A lock escalation occurs when the number of locks held on rows and tables in the database equals the percentage of the locklist specified by the maxlocks database configuration parameter. In this situation, the database manager

begins converting many small row locks to table locks for all active tables. As lock escalations reduce concurrency, conditions that might cause lock escalations should be avoided. We discuss lock escalation in greater detail in 6.1.2, “Scenario: Connection and workload management” on page 340.

DB2 Performance Expert provides locks information at different levels. In this section we look into details of lock conflicts or lock suspensions. DB2 Performance Expert provides two different views for analyzing lock conflicts: Locking Conflicts and Applications in Lock Conflicts.

Locking Conflicts

Locking Conflicts monitor views show all the locking conflicts currently in occurrence for the selected DB2 instance. It could includes the following items:

- Count
- Database Name
- Lock Attributes
- Lock Count
- Lock Hold Count
- Lock Mode
- Lock Name
- Lock Object Type
- Lock Release Flags
- Lock Type
- Partition
- Table Name
- Table Schema Name
- Tablespace Name

The items to be displayed and the display order can be customized to your needs.

Figure 4-20 shows the Locking Conflict view with one locking conflict. Double-clicking the row in the Locking Conflict table or selecting **Selected** → **Applications in Locking Conflicts** will launch the Application in Locking Conflicts view.

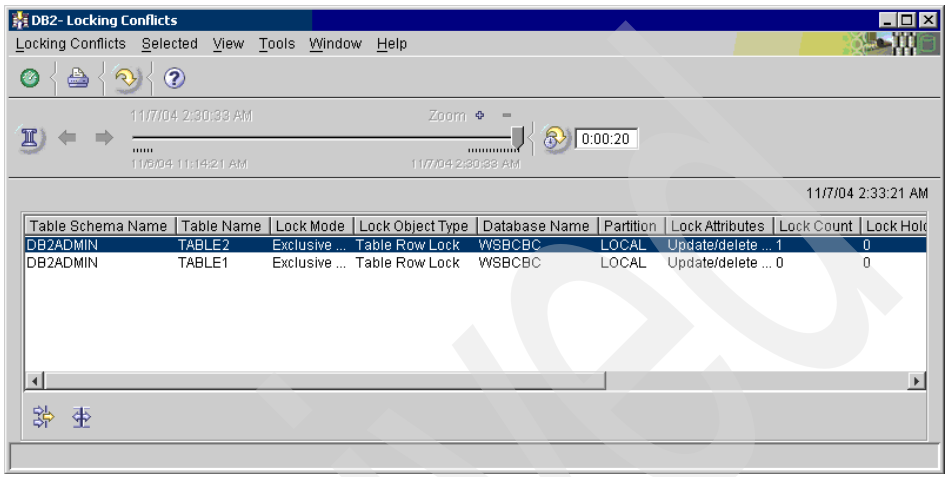


Figure 4-20 Locking conflicts

Applications in Locking Conflicts

The Applications in Locking Conflicts section of DB2 Performance Expert can be used to view all applications that are involved in a locking conflict situation in the selected DB2 database. This view shows both holding and waiting applications until the time conflict is resolved. It shows the following items in tabular form:

- Database Alias
- Mode
- Application Name
- Application ID
- Application Handle
- Auth. ID
- Application Status
- Table Name
- Table Schema Name
- Tablespace Name
- Lock Mode
- Lock Type
- Lock Object Type
- Lock Wait Time (ms)
- Partition
- Sequence Number
- Lock Name

- Lock Mode Requested
- Lock Type Requested

You can view further details about the application in a locking conflict by double-clicking the selected row or by selecting **Selected** → **Application Details**. Figure 4-21 shows two applications in lock conflicts. One of them is waiting and another on is in the holding stage. Each row will remain visible in this view until the conflict is resolved.

You cannot force off an application from this window. You can only force applications from the Application Summary window.

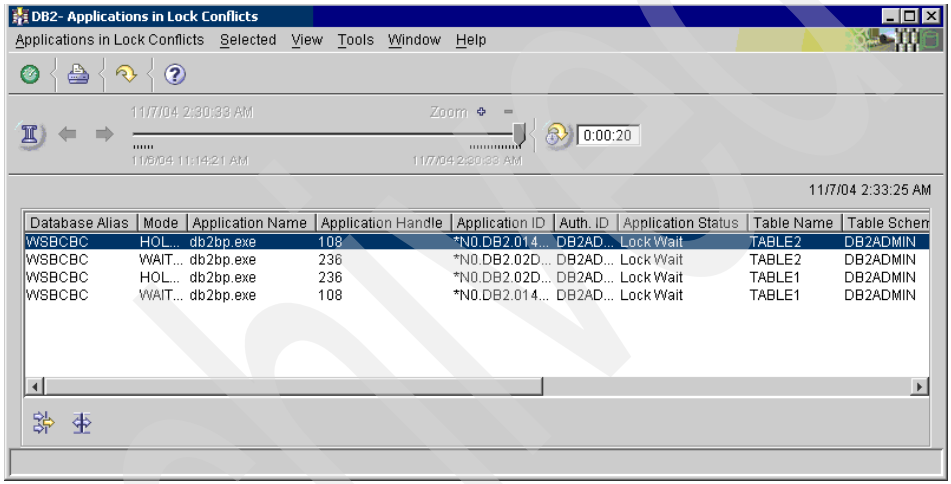


Figure 4-21 Applications in lock conflict

4.1.6 System parameters

Under the System Overview, DB2 Performance Expert provides two views for DB2 system parameters, one for instance-level (Database Manager configuration) and one for database-level (Database configuration).

The parameter values shown in both views are the CURRENT values, not any delayed values. You should be aware of this when doing analysis. There is no indication that some value may have a delayed setting waiting for the next instance restart.

System Parameters - Instance

Whenever an instance is created, a corresponding DB2 database manager configuration file is also created and initialized. Each DB2 database manager configuration file comprises different parameters. Some of these parameters can have a significant effect on performance. So keeping track of these parameters is really very important. DB2 Performance Expert provides the *System Parameters - Instance* feature to view these parameters for your instance in a very organized way. It categorizes in a self-explanatory fashion. As shown in Figure 4-22 on page 185, the database manager configuration data for an instance is divided into four categories according to their purpose and function:

- Instance Management

  This pane shows the database manager configuration parameters that help you manage your DB2 instance. In this pane, data is organized in two groups. They are System management and System monitor parameters.

  - Instance Administration

    The parameters in this group are shown the Instance Administration, Diagnostic, and Authentication sections.

- Capacity Management

  This pane shows the configuration parameters at the DB2 instance level that can affect the throughput on your system. Further in this pane, the data is divided into different sections showing parameters related to agents, agent private memory, agent/application communication memory, database application remote interface (DARI), database manager instance memory, and database shared memory.

- Communications

  This pane shows the database manager configuration information required to efficiently set up client/server environment. These parameters are categorized in four groups: Communication protocol setup, Distributed services, Capacity management, and DB2 discovery.

- Logging/Recovery/Parallel

  This pane shows the database manager configuration parameters used to efficiently recover your data or transactions and provide information about parallel processing. It shows data in three different sections, displaying parameters related to recovery, distributed unit of work recovery, and parallel processing.

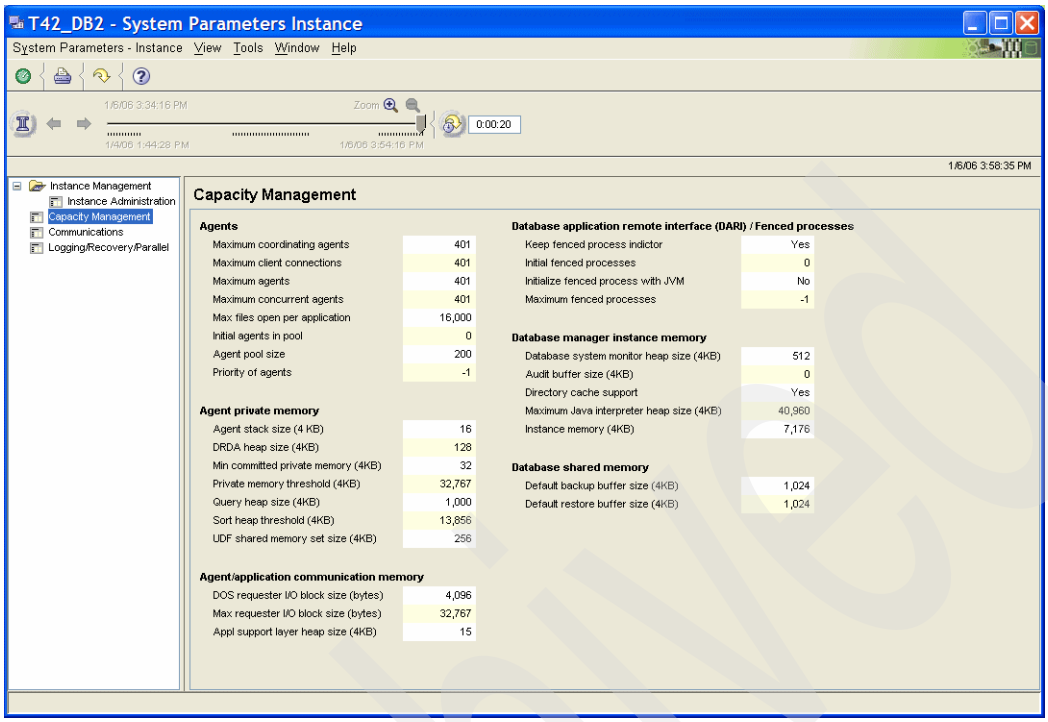


Figure 4-22 System parameters- Instance

System Parameters - Databases

Just as a DB2 database manager configuration file is created and initialized whenever an instance is created, a database configuration file is created and initialized each time a new database is created. Each database configuration file comprises over eighty different parameters.

The System Parameter - Databases view (Figure 4-23 on page 186) shows database manager configuration parameters in tabular form for all the databases in the instance. Double-clicking the database row will show you the System Parameters in a page format with the parameters divided into four different sections according to its usage:

- Capacity Management

  This pane shows the configuration parameters at the DB2 database level that can affect the throughput on your system. The data is divided into different sections showing parameters related to application and database shared memory, I/O storage, locks, agents, and the agent private memory.

- Communications

  This pane shows the object name in the DCE namespace and the type of discovery mode that is started when the DB2 administration server is started.

- Logging and Recovery

  This pane shows the configuration parameters that help you recover your data or transactions. These parameters are categorized in the following groups: Database log activity, Database log files, Backup and Recovery, and Recovery (Tivoli® Storage Manager).

- Database Management

  This pane provides information about the selected database configuration parameters that can influence the management of this database. These parameters are categorized in the following groups: Compiler settings, DB2 data links manager, Attributes, and Query enabler.

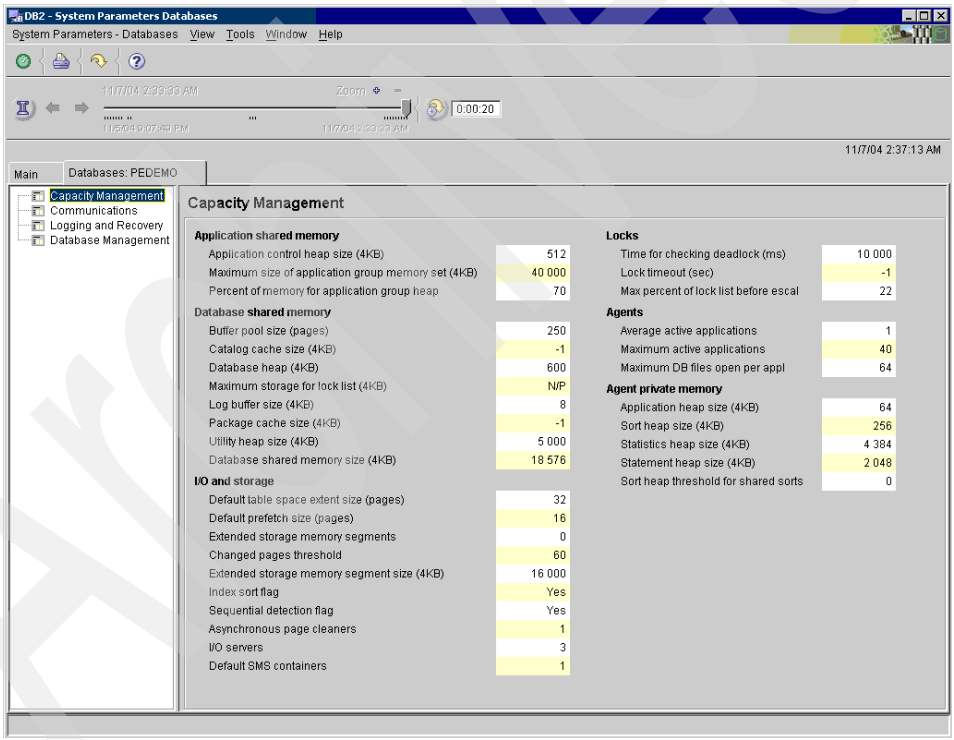


Figure 4-23 System Parameter - Database

4.1.7 Operating System monitoring

As a DBA, you often want to know more about the underlying operating system performance, not just your DB2 engine statistics. Performance Expert V2.2.0 now provides this capability for the AIX and Solaris operating systems, and with V2.2.0.1, Linux is also supported. The capture of OS-level statistics is handled by the CIM interface, and must be configured and enabled for the monitored instance in PECONFIG.

Performance Expert shows operating system information in two main windows, each with sub-panes. The type of information is described here:

- Operating System Information
  - System: Basic system information, such as memory and number of CPUs
  - Storage: File system information, similar to the output from the **df** command
    - Disk Information: (Solaris) Additional disk statistics per disk
- Operating System Status
  - Performance: CPU, memory, active processes, and paging
  - Processes: Running process information, similar to the output from the **ps** command

There are some differences between the AIX, Solaris, and Linux implementations of the CIM collection, so there are slight differences in the amount or format of data shown on the Performance Expert screens. If/when the CIM implementations change, PE will be enhanced to accommodate the changes where appropriate.

For example, in Figure 4-24, we see an OS Status - Performance window from AIX, and in Figure 4-25 on page 189, we see one from Solaris.

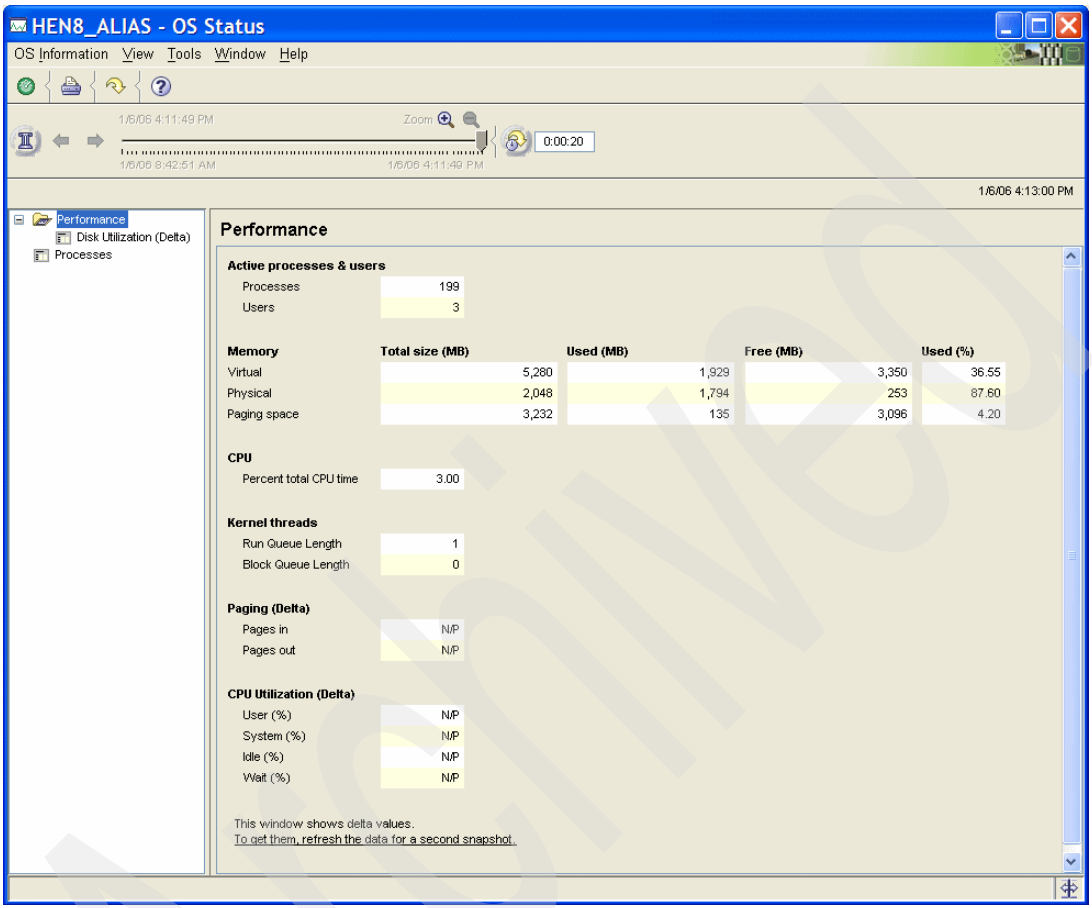


Figure 4-24 Operating System Status - Performance window for AIX

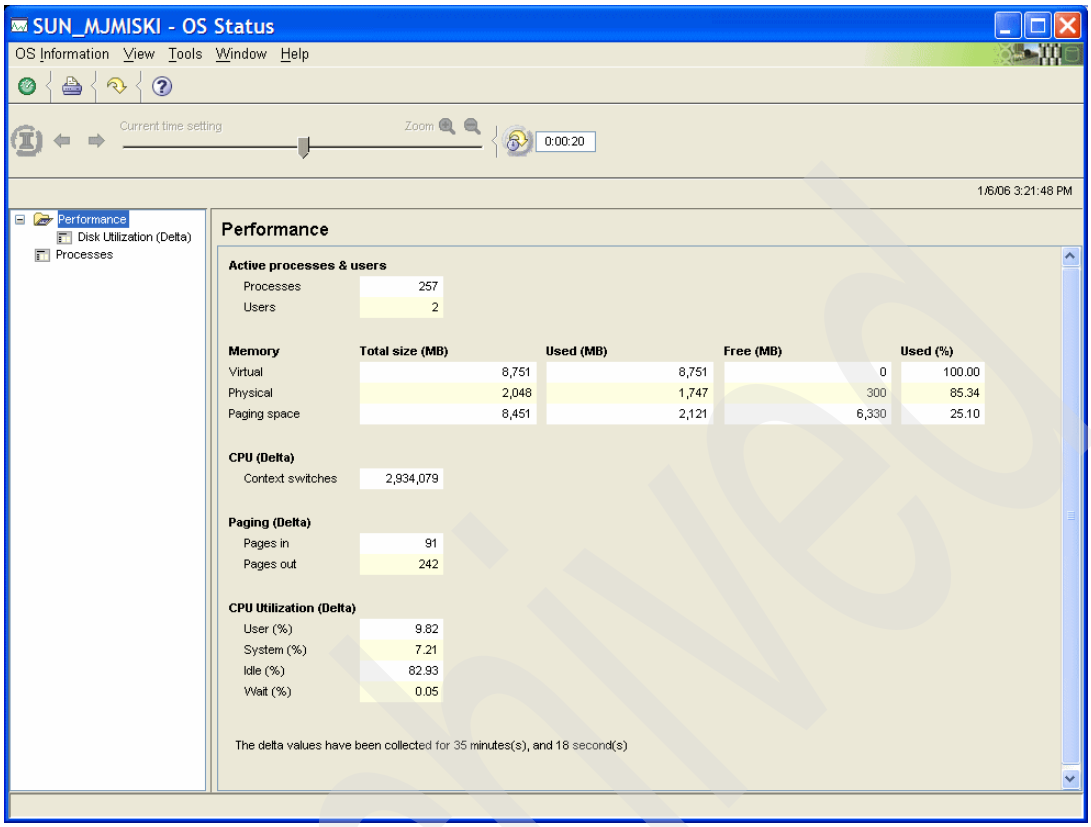


Figure 4-25 Operating System Status - Performance window for Solaris

As with the other PE windows, you can use History mode to look at the different values from past snapshots. Remember that OS data does not come from DB2 snapshots, but from CIM. They can be configured to be collected at different intervals than the DB2 monitor snapshots. We recommend that you use less frequent OS collection intervals than DB2 snapshots. The type of information collected is not so volatile that very frequent collection is necessary. Intervals of 15 or 30 minutes are probably sufficient.

Space management

File system usage information is especially interesting to a DBA, especially the storage used in tablespaces. This information is not only shown on the OS-level screens launched from the System Overview window, but also embedded within the other Statistics Details screens, relevant to the individual tablespaces and databases themselves. Some space-related counters are also available in System Health charts and Periodic Exception processing.

In Figure 4-26, we see the top-level Tablespaces window that now includes space-usage statistics.

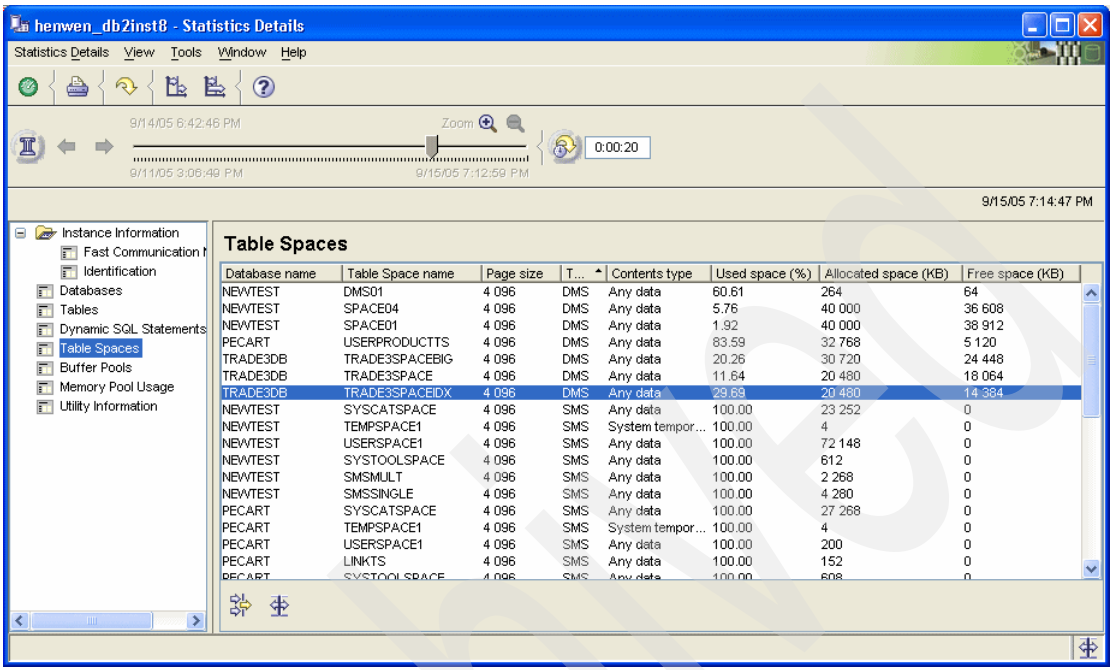


Figure 4-26 Table Spaces window showing space usage (DMS)

We drill down into the DMS tablespace USERPRODUCTTS, where we can see other space information, as shown in Figure 4-27 on page 191.

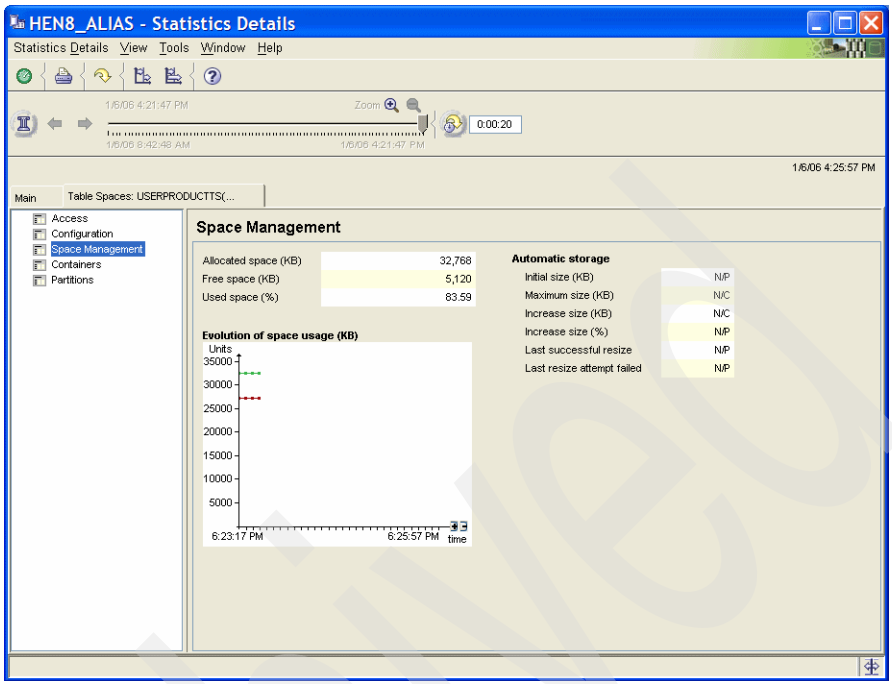


Figure 4-27 Table Spaces - Space Management detail

We can see yet another view of the space used by looking at the Containers pane, where we can drill into an individual container, as shown in Figure 4-28.

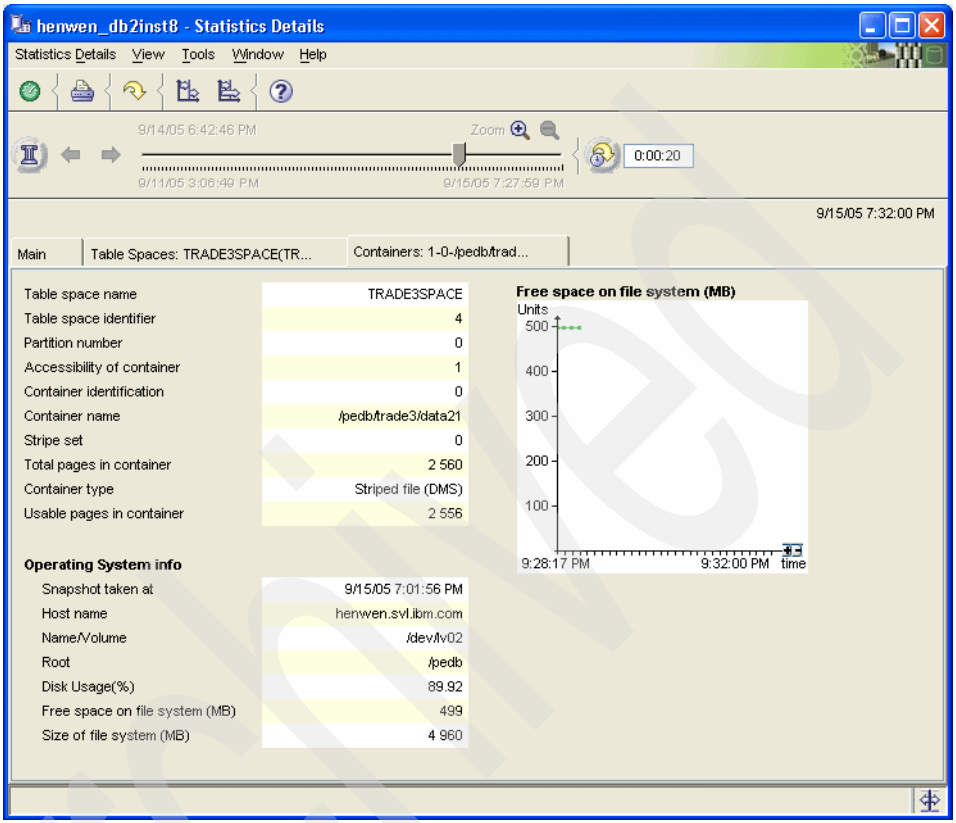


Figure 4-28 Table Space Containers - Space usage detail

The examples shown above have been for a single tablespace. Performance Expert also has a Space Management window at the Database level, so you can see at a glance how much space is being consumed across all tablespaces. This is shown in Figure 4-29 on page 193. As with all other graphs or charts within PE, if you press the left mouse button and hold it over the graph, the values for the plot points will be shown in a pop-up next to the mouse pointer.

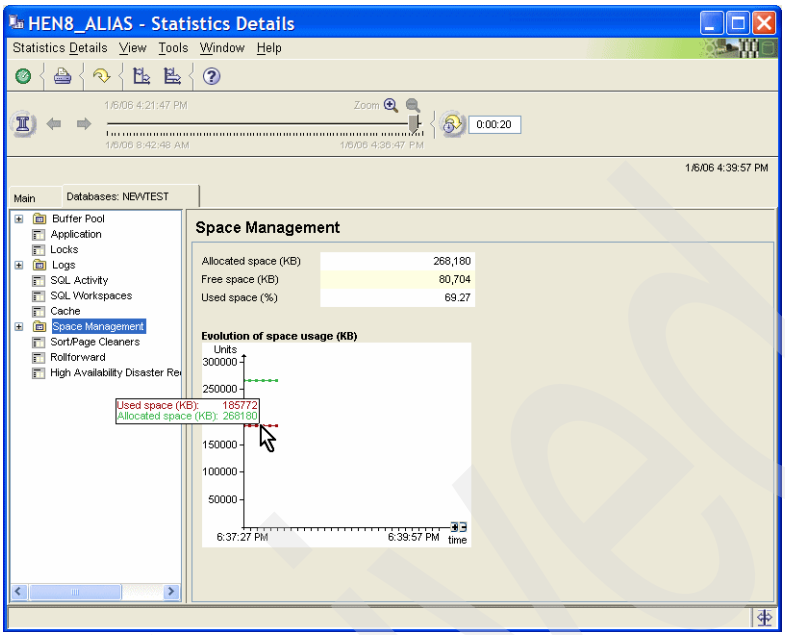


Figure 4-29 Space Management at Database level

4.2 Alerts and exceptions

As a DBA, you need to be concerned with the health of your databases, but you do not have time to spend hunting for problems. If you had a tool that would watch for problems in areas that you specify, this would be a help. DB2 Performance Expert provides this function with Exception Processing.

There are currently two components of exception processing:

- Periodic exceptions: User-configurable thresholds for a variety of DB2 snapshot monitor elements.
- Deadlock event exceptions: A deadlock event monitor.

In Figure 4-30, we show the differences between the two types of exception processing. While they can both result in alerts to the DBA, the configuration and use of them are different. We describe this topic in detail in this section.

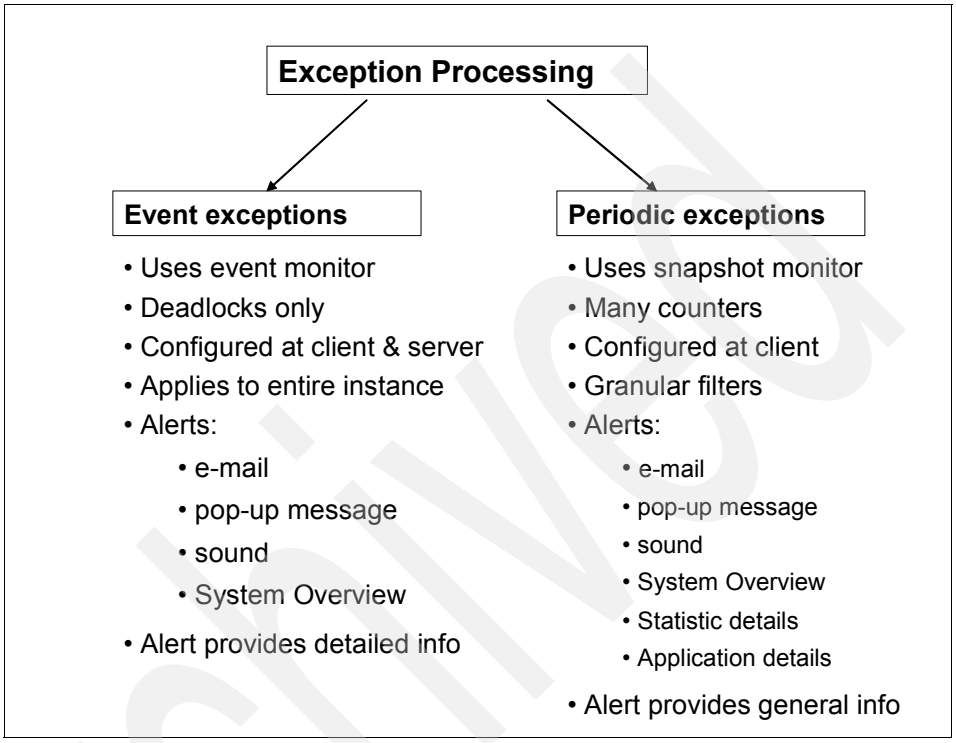


Figure 4-30 Exception processing overview

The *periodic exception* component of exception processing relies on DB2 snapshot data, just like what you find in the Application Summary or Statistics Details in PE. You can create “threshold sets” that define to PE which snapshot elements (counters) to watch, and what critical values should trigger an exception occurrence. All thresholds offer additional filtering capabilities so you can, for example, raise alerts only related to a particular database, user, or application.

The *deadlock exception* component uses a DB2 deadlock event monitor as the notifying source, and will report an alert for all occurrences of a deadlock anywhere in the monitored instance. To use deadlock exception processing, you must first enable event monitoring on a monitored database by issuing the EVMON command in peconfig. This step is performed during the configuration of the PE Server and described in 3.1, “Configuration” on page 46. For purposes of

this discussion, we assume the event monitoring is already configured and enabled.

This section shows you how to enable these two features, how to set up and respond to alerts, and makes some suggestions for creating threshold sets. The information in this section complements the information in Chapter 10, “Working with exception processing“, in *Monitoring Performance from the Workstation,* SC18-7976.

What’s new in V2.2?

Significant enhancements were made to exception processing in the V2.2 release of DB2 Performance Expert. These new features will be described in this section, and identified by the change bars on the left of the page. The key enhancements made are:

- Display of exception status using graphical indicators
- Individual threshold violation fields are highlighted
- Predefined threshold set templates for OLTP and BI environments
- Different indicators for warning and problem exceptions
- Many more counters available for threshold evaluation

4.2.1 Periodic exception processing

Periodic exception processing enables you to check the DB2 application activity or statistics fields against thresholds that you set, at intervals defined by you. When the value of a threshold exceeds these limits, an exception is logged.

Periodic exception processing is based on data returned in snapshots. The snapshot data depends on the settings of your default monitor switches, so before you create new threshold sets, you should ensure the counter values you want to compare against will be returned in the snapshot. For example, if DFT_MON_BUFPOOL is OFF, a buffer pool snapshot will return zeros in most of the data elements. If you define a threshold set that watches for a low buffer hit ratio, you will get unexpected results when the ratio calculation returns a zero as the hit ratio! This example applies to many of the counters, so you should be aware of how they are used when you set thresholds.

There are several steps you need to follow to use periodic exception processing:

1. Create or modify a threshold set.
2. Configure properties for periodic exception processing
3. Activate periodic exception processing
4. Examine threshold exceptions
5. Stop periodic exception processing.

Each of these steps will be described in this section.

Step 1: Create or modify a threshold set

The first step in enabling alerts for periodic exceptions is to create a threshold set. The threshold set contains one or more thresholds. Some key points to understand about threshold sets are:

- Threshold sets are activated at the monitored instance level.

  Granularity for monitoring exceptions at the database, tablespace, or buffer pool level is controlled within the individual threshold by setting qualifier values (filters).

- Only one threshold set per user can be active for one monitored instance.

  This means you might want a larger number of thresholds in one set, if you monitor many different databases on one instance.

- The same threshold set can be activated for multiple monitored instances.

  If your architecture is such that you have an application that contains one database per instance, but you monitor many instances, you could define one threshold set and activate it for all your monitored instances.

We provide examples of these variations in the following discussion.

To open Exception Processing, select **Tools** → **Exception Processing**, or use the tool bar icon (Figure 4-31.)

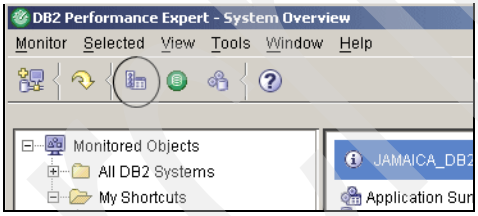


Figure 4-31 System Overview tool bar - Exception Processing

The Exception Processing window shows the Event and Periodic exceptions as separate folders. Each type of exception produces its own log, which you can view individually or together. The first time you open this screen, there will not be any exceptions or threshold sets (see Figure 4-32 on page 197). We first show you how to create a new threshold set.

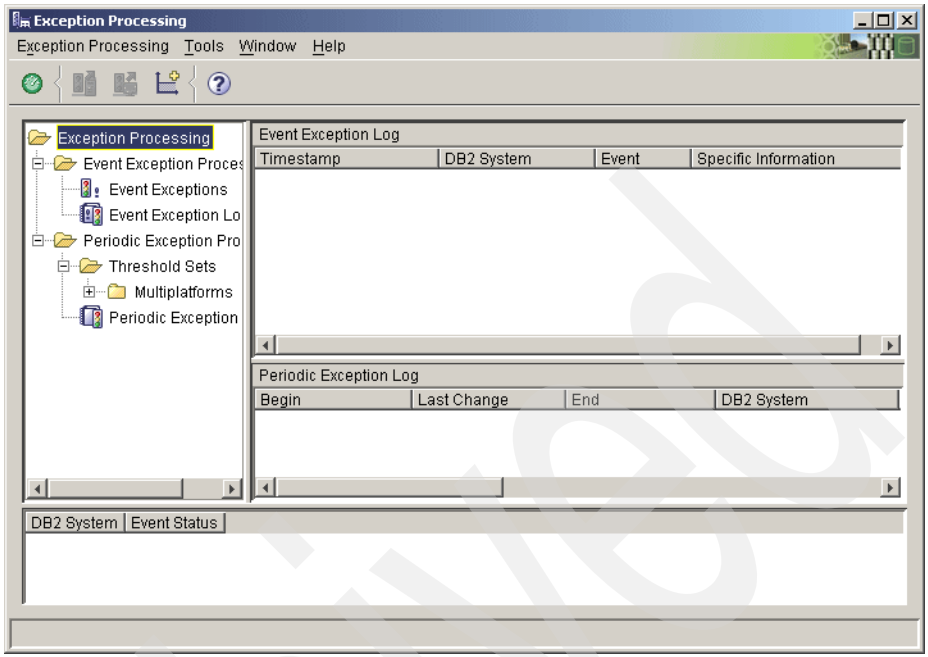


Figure 4-32 Exception Processing main window

To create a new threshold set, select **Exception Processing** → **New Threshold Set**, or use the icon on the tool bar. You can also use the context menu on the Threshold Sets folder, by right-clicking on the Threshold Sets folder and selecting **New Threshold Set**.

The New Threshold Set dialog is shown in Figure 4-33 on page 198. Be sure to select the **Multiplatforms** radio button, or the counters will be for Z/OS and of no use to you. The Name and Author fields are required. Select the **New** radio button and click **OK** to continue.

New in Performance Expert V2.2 is the ability to create a new threshold set based on a predefined template set. The steps for creating a new threshold set from scratch are still the same.

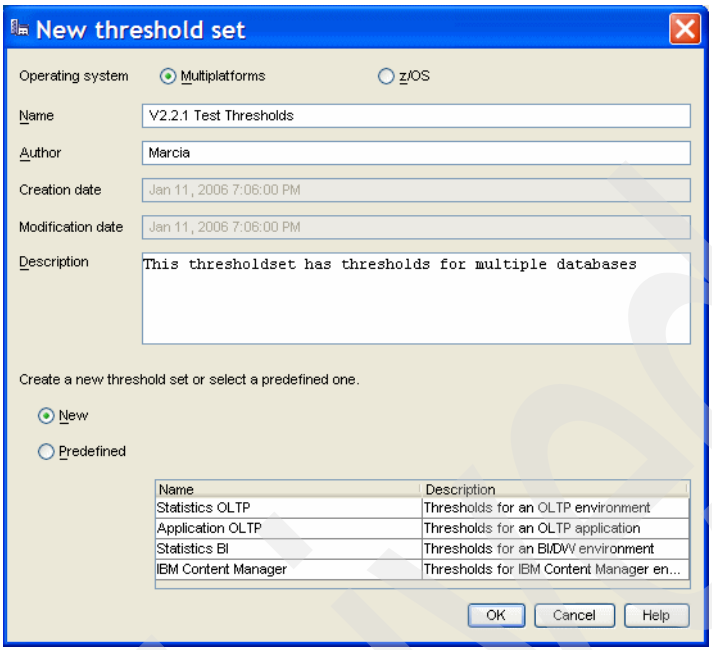


Figure 4-33 Creating a new threshold set

Figure 4-34 shows the Threshold Set Editor dialog, where you create one or more thresholds. In the redbook test environment used in this example, we are monitoring one instance, db2inst2 on server jamaica, and two databases within that instance, TRADE3DB and SAMPLE.

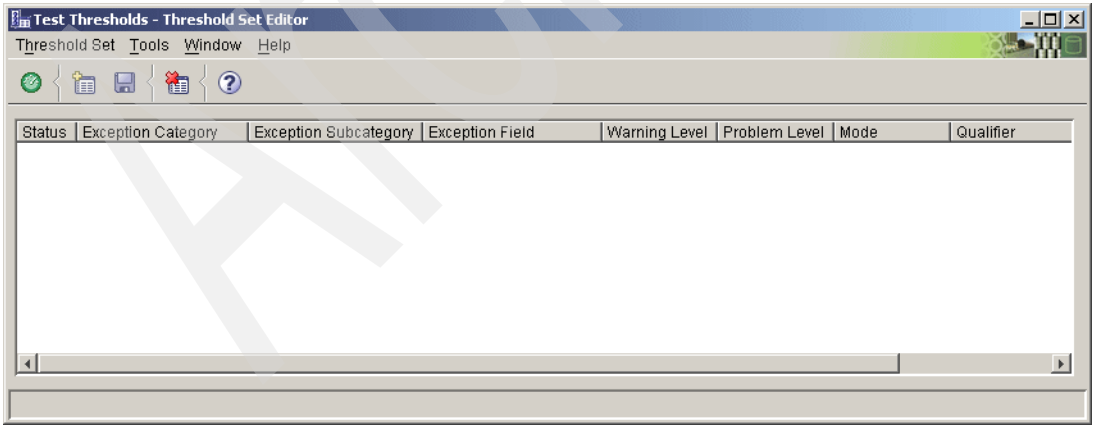


Figure 4-34 Threshold set editor - new thresholds

The quickest way to open the threshold editor is to use the tool bar icon ( ). Figure 4-35 shows different ways you can open the New Threshold dialog.

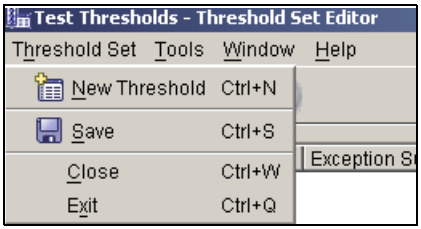


Figure 4-35 Threshold Set Editor - shortcuts

Tip: Most screens in PE Client offer multiple ways to accomplish the same task - menu selection, tool bar icons, or keyboard shortcuts. We will not describe each of these methods for every screen; you can find and use the method that suits you.

Figure 4-36 on page 200 shows the New Threshold window where you define the evaluation criteria for the exception. There are over 350 counters available to you as thresholds and we do not presume to discuss them all here.

The counters available here all correspond to the snapshot counters shown on the Application Details, Statistics Details, System Parameter and Operating System windows. The selectable counters are grouped into sections that also correspond to the screens you see in those areas. For further information about specific DB2 snapshot monitor elements, you should consult the following resources:

- Field help within PE Application Details and Statistics Details.
- DB2 System Monitor Guide; details can be found in the DB2 UDB manual *IBM DB2 UDB System Monitor Guide and Reference V8*, SC09-4847.
- The online InfoCenter for DB2 at:

  `http://publib.boulder.ibm.com/infocenter/db2help/index.jsp`

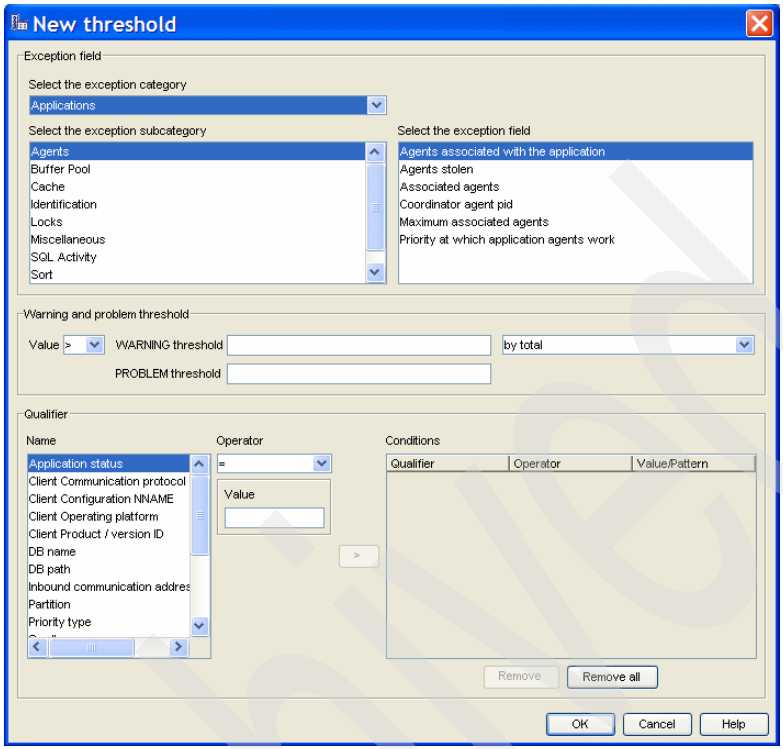


Figure 4-36 New Threshold window

The highest level group is the exception category, and each category available corresponds to one of the items from the System Overview contents pane. This is shown in Figure 4-37 on page 201.

Note: While all threshold counters correspond to fields you see on the snapshot screens elsewhere in PE, not every snapshot element is made available for use as a threshold. All counters that presents time stamps, textual information, and counters for some DB other objects are not available. More elements will be available in future releases of Performance Expert.

Threshold set exception categories and subcategories are:

- Applications (see Figure 4-38 on page 202.)
  - Agents
  - Buffer Pool
  - Cache
  - Identification
  - Locks

  - Miscellaneous
  - SQL Activity
  - Sort
  - Workspace
- Operating System Information
  - Storage
- Operating System Status
  - Performance
- Statistics (see Figure 4-39 on page 202.)
  - Buffer Pools
  - DB2 Connect Server
  - Databases
  - Instance Information
  - Tablespaces
  - Tablespaces - Partitions
  - Tablespaces - Partitions - Containers
- System Parameters - Instance (see Figure 4-40 on page 203.)
  - Capacity Management
  - Communications
  - Instance Management
  - Logging/Recovery/Parallel

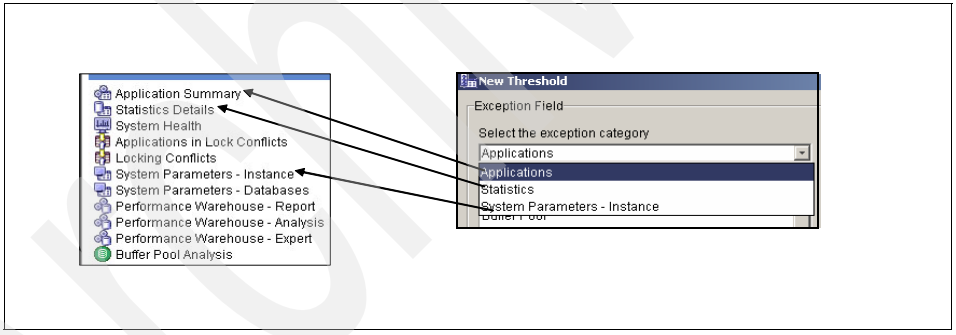


Figure 4-37 Threshold Set categories mapped to System Overview

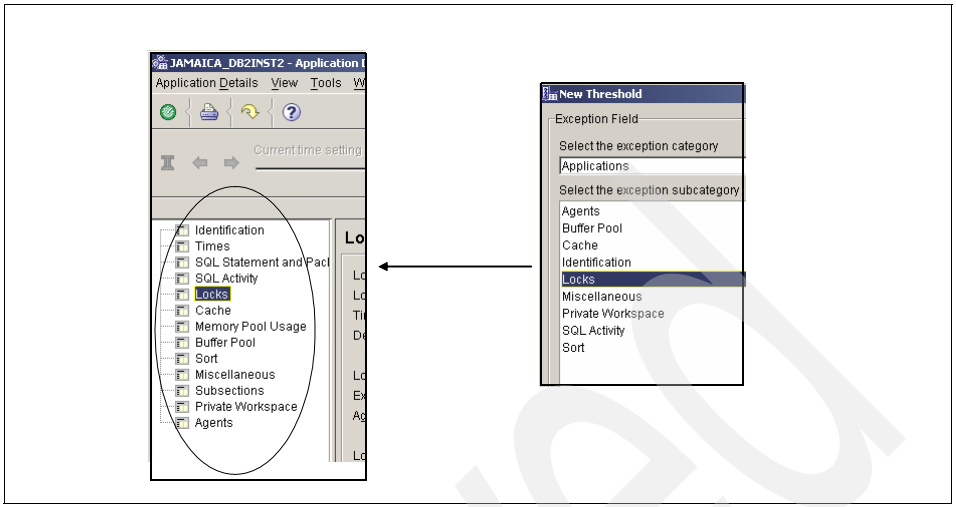


Figure 4-38 Applications threshold set subcategory mapped to Application Details

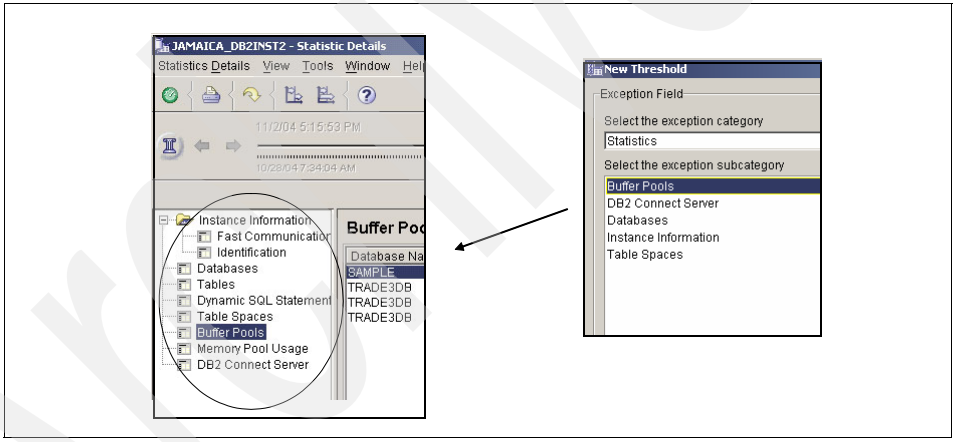


Figure 4-39 Statistics threshold set subcategory mapped to Statistics Details

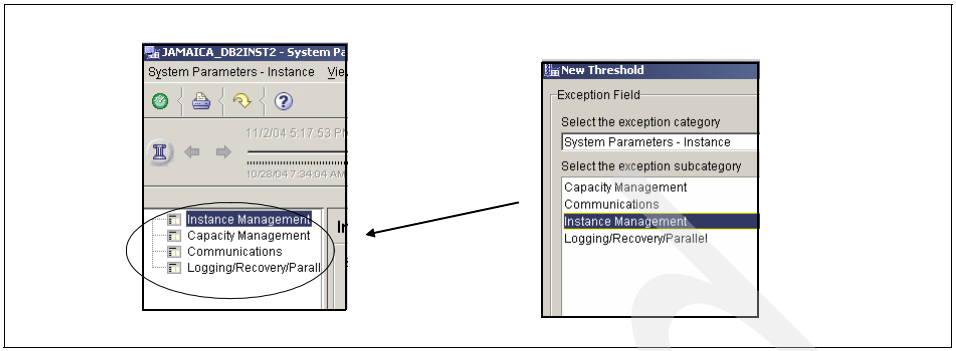


Figure 4-40 System Parameters subcategory mapped to System Parameters

In this example, we create one threshold set that will contain the following criteria:

- DBM CFG DIAGLEVEL parameter falls below 3.

  This will point out if someone changed the parameter.

- Application using more than 5000 bytes of log space per second in one UOW.

  We have had problems in the past with logs filling up, so we are interested in learning more about that topic.

- Buffer pool data hit ratio for SAMPLE database falls below 85%.

  Just an example. PE provides many places to find BP hit ratios.

- Any application that gets more than two lock timeouts.

  We recently lowered the LOCKTIMEOUT parameters, so we want to know if our changes have any bad effects.

Note: These are rather contrived thresholds, and should not be taken as recommendations, but are used here to demonstrate the function of periodic exception processing.

On the New Threshold screen, the Exception Field area is where you tell PE what counter to evaluate. The center section is the definition of the comparison values. The bottom Qualifier area is where you can further limit the evaluation.

We show in Figure 4-41 how to create threshold example 3, where we look at a buffer pool hit ratio for a specific database. The Category is Statistics, Subcategory is Databases, Exception Field is Data BP hit ratio %. We selected an arbitrary hit ratio floor of 85% as the alert warning threshold, and 80% as the problem threshold.

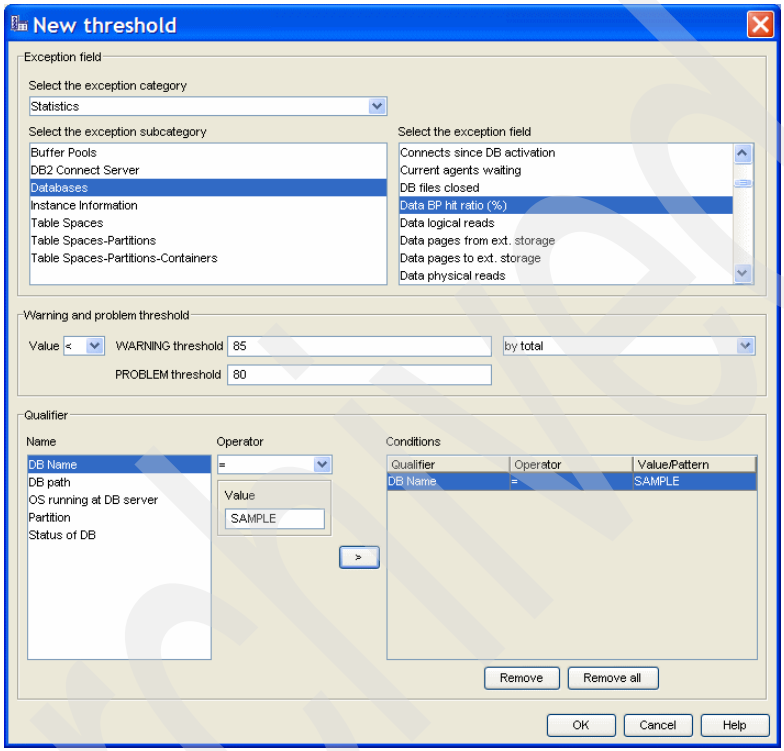


Figure 4-41 Buffer pool threshold

New in Performance Expert V2.2 is that a different image will appear next to warning and problem exceptions. You should take care to decide what the actual warning and problem thresholds are for your particular counter. This may take some trial and error, or you can use the predefined Threshold Sets as a starting point.

Next, we add a qualifier to limit the alert to only include the SAMPLE database. We do not want to consider the BP ratio on the other database being monitored.

Notice we are using the BP hit ratio from the Database subcategory and not the Buffer Pool subcategory. This means that the 85% limit will be evaluated for all buffer pools in the SAMPLE database - we are not filtering a particular buffer pool. In our case, this is acceptable, because we only have one buffer pool. If we

wanted to filter by buffer pool, we could use the Buffer Pool subcategory, which provides a filtering qualifier by buffer pool name. The mechanism for adding qualifiers is the same for all categories and counters.

Tip: You can use multiple qualifiers in one threshold. Each is evaluated as an "AND" with the other qualifier values. However, you cannot use the *same* qualifier more than once in the same threshold.

To enter the filter (qualifier) value (see Figure 4-42):

1. Highlight the qualifier.
2. Type the comparison value in the Value entry field; this will enable the arrow button.
3. Press the arrow button to move it to the right-side box.

Important: The qualifier value is *case sensitive*! Check how the value is displayed on the snapshot screens to be sure of how it will be evaluated. Hint: Database object names such as database name and buffer pool name are always uppercase.

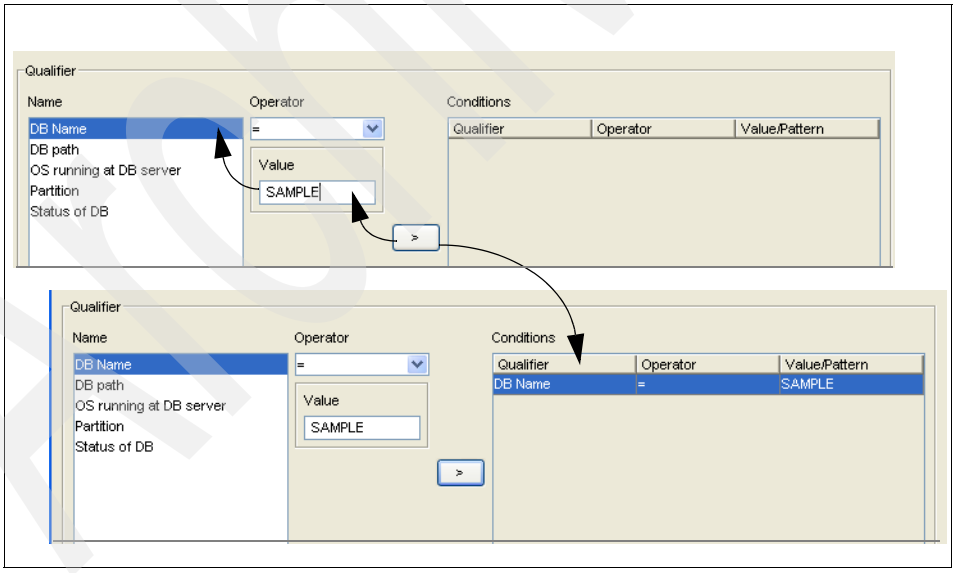


Figure 4-42 Adding a threshold qualifier

You can add multiple filters, but only one of each kind will be allowed.

For the Statistics and System Parameters category counters, you should only use the “by total” evaluation mode. This is because of the type of counter and data available to evaluate. For the Applications category counters, the other evaluation modes are available because there is a way to do the math for each particular application by evaluating it against the duration of the application or the UOW, for example.

The evaluation modes available are:

- By total

  The current counter value is used, and there is no average. Depending on the selected operator, the counter value must be equal to, greater than, or less than the specified threshold value.

- Per commit

  This is the average per number of commits in an application. Depending on the selected operator, the counter value must be equal to, greater than, or less than the result of dividing the specified threshold value by the number of application commits.

- Per second

  This is the average per elapsed time of the application in seconds. Depending on the selected operator, the counter value must be equal to, greater than, or less than the result of dividing the specified threshold value by the application elapsed time in seconds.

- Per minute

  This is the average per elapsed time of the application in minutes. Depending on the selected operator, the counter value must be equal to, greater than, or less than the result of dividing the specified threshold value by the application elapsed time in minutes.

- By percentage

  This is the percentage of a single application in relation to the sum of all qualified applications. Depending on the selected operator, the counter value must be equal to, greater than, or less than the result of *sum of matching thread values/sum of all thread values * 100*.

Tip: When using a counter that is already defined as a percentage, such as a buffer pool hit ratio, you should use the *total* evaluation mode, *not* the *percentage* mode. The hit ratio counter is already calculated as a percentage, and this has nothing to do with the threshold percentage mode.

We do not show here the steps for creating the other example thresholds, but the steps are the same. Figure 4-43 on page 207 shows the completed threshold set

for this example. You must save the threshold set before you can exit the screen, or you will be prompted to save it. Notice we have defined four individual thresholds in the set. Each threshold is evaluated separately by the PE Server.

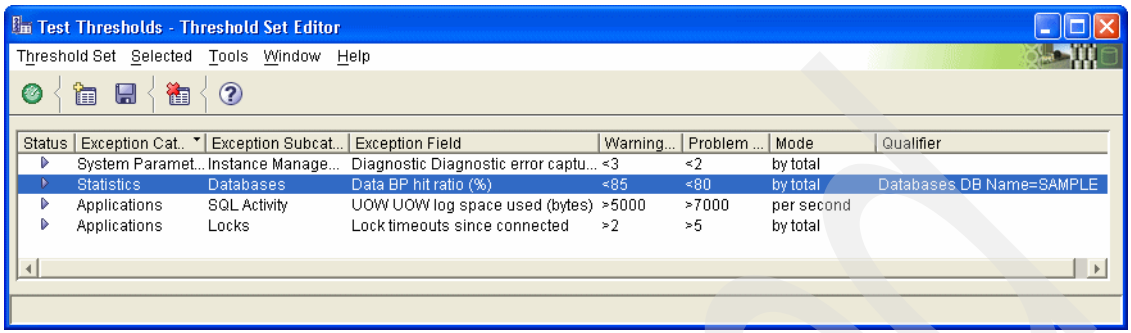


Figure 4-43 Definition of example threshold sets complete

How to use wild cards

In V2.2.0.1, PE introduced the capability to use wildcards in the qualifier. Prior to this, you could only evaluate one thing at a time. You now can evaluate values that are equal, not equal, like, or not like. This is shown in Figure 4-44.

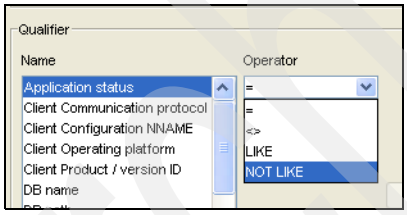


Figure 4-44 Wildcard options on threshold qualifier

In Figure 4-45, we show an example of a threshold that will evaluate Lock Escalations for all applications that are *not* coming in via the loopback address.

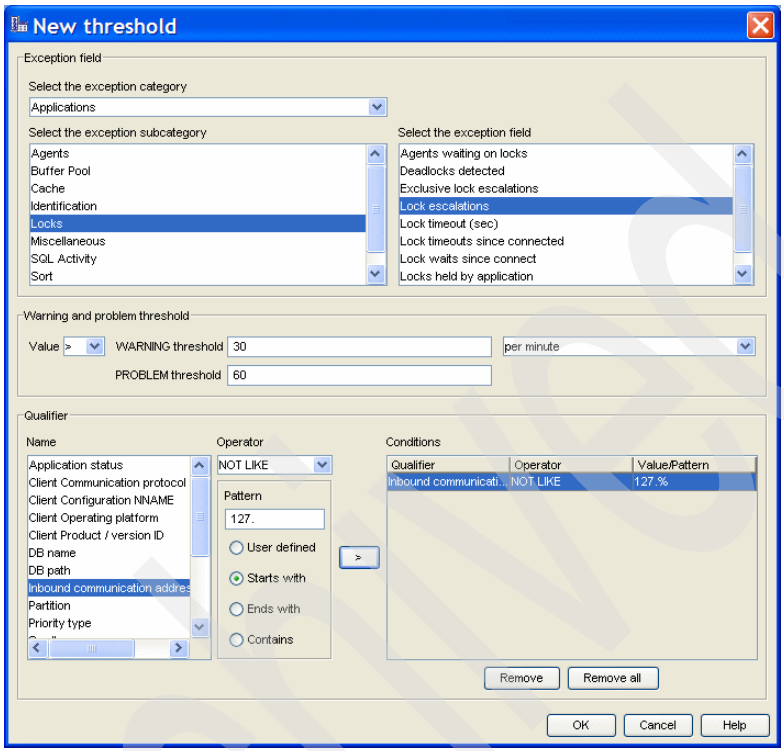


Figure 4-45 Threshold wildcard example

Modifying an existing threshold set

To modify an existing threshold set, select the threshold set in the tree pane of the Exception Processing window, then right-click and choose **Edit** from the context menu to open the Threshold Set Editor for that threshold (see Figure 4-46 on page 209). You can also double-click the Threshold Set to enter edit mode.

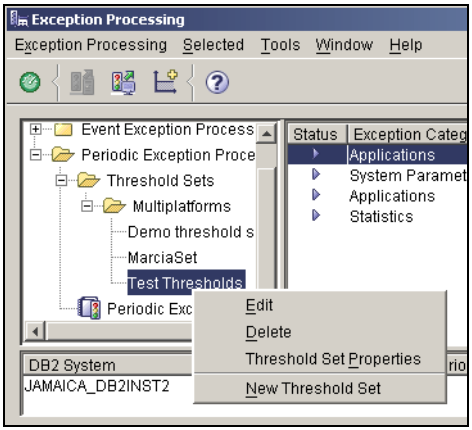


Figure 4-46 Modify an existing threshold set - open the editor

Once you are in the Threshold Set Editor, you can open any threshold by double-clicking and make your changes, as described previously for creating a new threshold set. You can add new thresholds to an existing set the same way.

To change a qualifier value in a threshold, select the qualifier in the right side (see Figure 4-45 on page 208), type in the new comparison value, and press the arrow button. This will override the old value in the right side box.

Turning off selected thresholds within a threshold set

You can selectively disable particular thresholds within a threshold set. Maybe you created a large set, but you do not want to monitor some counters for a while. Rather than deleting the threshold completely, you can simply disable it so the PE Server will not evaluate it.

Open the Threshold Set Editor, as above, highlight the threshold, right-click and de-select the **Active** menu item, as shown in Figure 4-47.

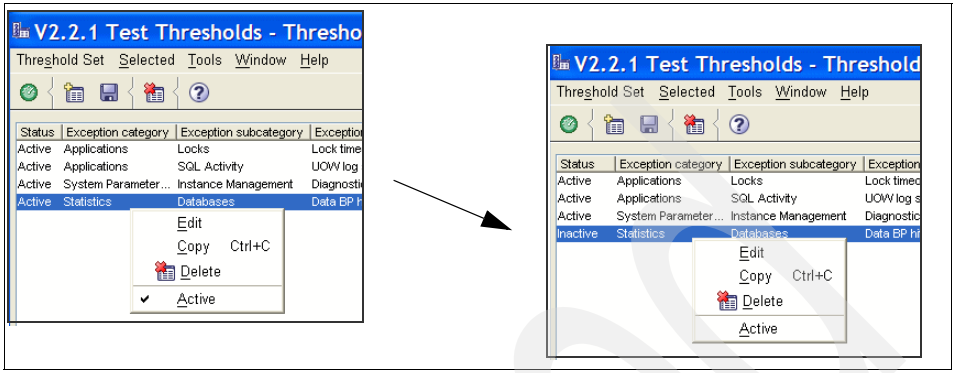


Figure 4-47 Threshold status activate/deactivate

The Status column will display “Active” or “Inactive” for each threshold (see Figure 4-47). Any threshold in the “Inactive” state will not be evaluated and you will not see exception alerts for it.

To reactivate later, select the **Active** menu item.

Using a predefined threshold set (V2.2)

When you create a new threshold set, you can create one from scratch, as described above, or you can use a predefined set. To use a predefined set, you should still perform the “Add threshold set” steps, then, as shown in Figure 4-33 on page 198, you should select the **Predefined** radio button, then select one of the predefined sets. An example is shown in Figure 4-48 on page 211.

There are sets for the following types of environments:

- OLTP: Statistics-related counters
- OLTP: Application-related counters
- Business Intelligence (BI)/Data Warehouse: Statistics-related counters
- IBM Content Manager: Statistics and OS-related counters

These threshold sets have been developed by experts in each environment. The threshold exception fields, and the warning and problem levels, are the most typical values for most customers. They are an excellent starting point, and can be used *as-is*, or you can modify them. The threshold set for IBM Content Manager (CM) is very specific to CM, and includes, for example, specific CM buffer pool names, so you would only want to use these if you are really using CM.

You can look in the “Introduction to CM database monitoring” on page 472 and Appendix C, “DB2 Performance Expert For Business Intelligence” on page 525 for some additional information about using DB2 Performance Expert with BI and CM environments.

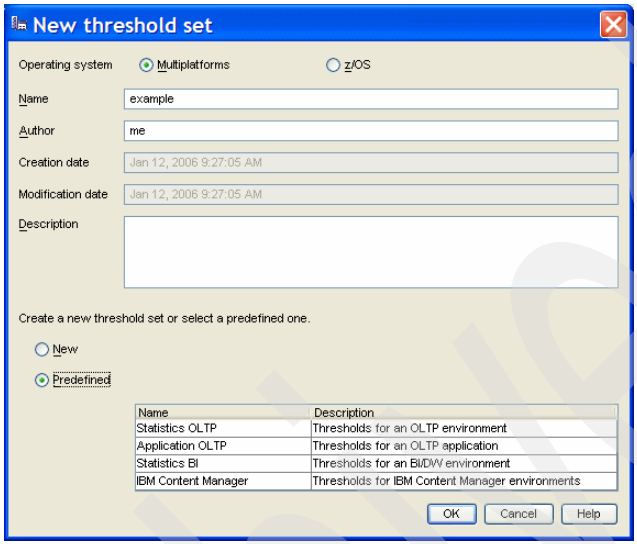


Figure 4-48 Creating a new threshold set using predefined set

Click the **OK** button and you will go to the Threshold Set Editor dialog. This dialog will be populated with some thresholds that would be appropriate for the type of environment you selected (OLTP, BI, or CM.) At this point, you can add, delete, or modify any of the thresholds described in this section.

Important: Most of the predefined threshold sets do not have any qualifiers defined. For example, if you are monitoring multiple databases and you only want to set thresholds for one of them, you should edit each threshold and add a qualifier for the desired database. Otherwise, you will get alerts for every database. For just getting started with exception processing, you can use these *as-is*, but you will want to fine-tune these as you use them over time.

Step 2: Configure properties and settings for exception Server properties

Several server settings must be configured before using periodic exception processing. These settings are controlled at the monitored instance level, so you must open the PE Server Properties for your monitored instance from the System Overview window. Select the instance, then right-click and select **Properties**, and move to the Exception page. The PE Server Properties window is shown in Figure 4-49 on page 213.

- Configuring threshold interval

  The check box “Enable periodic exception processing” controls only whether you can use exception processing at all. If you deselect this box, you will not be able to enable any threshold sets. This is enabled by default.

  The default threshold checking interval is 60 seconds. This value is not necessarily the actual interval used, but rather the base multiplier used at activation time. This is discussed in “Step 3: Activate periodic exception processing” on page 217.

- Configuring e-mail notification

  If you want to receive e-mail alerts, you must supply PE with the address of your SMTP mail server. PE uses SMTP services to send the mail, but does not act as a mail server itself - so you must supply the SMTP server name and port. On the Properties screen, you can also test the setup to ensure the mail alerts will arrive properly.

  There is no default e-mail information, so if you do not fill in these fields, you will not be able to use e-mail alerts.

- Configuring user exit

  In PE V2.2.0.1, you can now invoke a user exit when an alert exception occurs. You must enable the user exit here, and provide the program/script name that will be executed. See “How to use the exception alert user exit” on page 213.

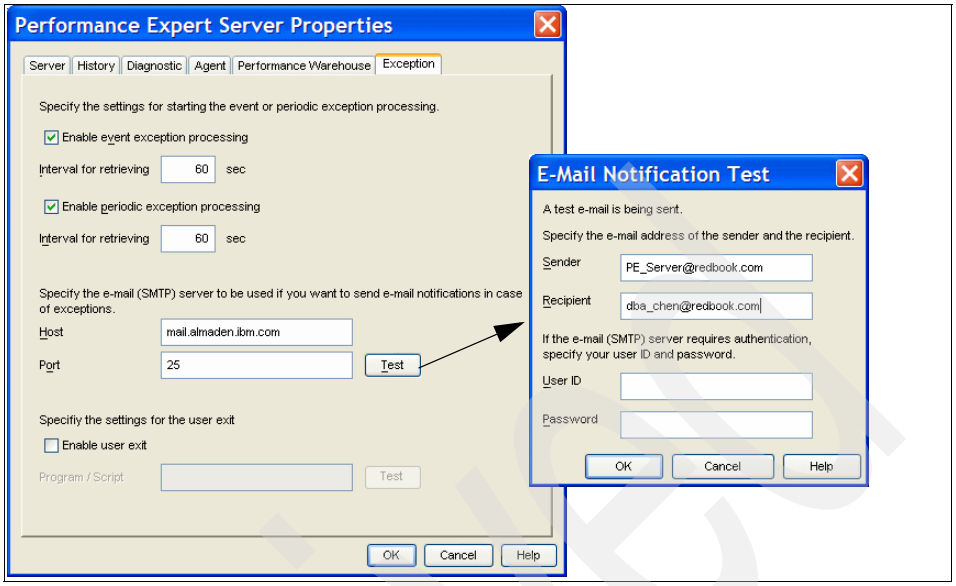


Figure 4-49 PE Server Exception properties

How to use the exception alert user exit

PE V2.2.0.1 now provides the option of invoking a user exit when an exception occurs, either periodic or deadlock. You must enable the user exit before you can use it. You can have a different user exit for each monitored instance. There is a sample user exit program, written in C, that is in the samples directory under the PE Server install path. The *IBM DB2 Performance Expert for Multiplatforms IBM DB2 Performance Expert for Workgroups Installation and Configuration*, SC18-9191 has a section that describes in detail how to use the user exit.

The data passed to the user exit is in XML format, so your exit must be able to parse the XML. There are two sample XML files also provided in the samples directory, one for threshold exceptions, and one for deadlock exceptions.

We do not show a user exit example here.

Exception Processing notification and log settings

To configure what types of alerts are given, how many alerts are shown in the log, and who should receive e-mails, you need to go to the Exception Settings dialog.

On the System Overview window, open Exception Processing from the menu by selecting **Tools** → **Exceptions** → **Exception Processing**.

From the Exception Processing window, open the Settings window by selecting **Exception Processing** → **Settings**. The settings window is shown in Figure 4-50.

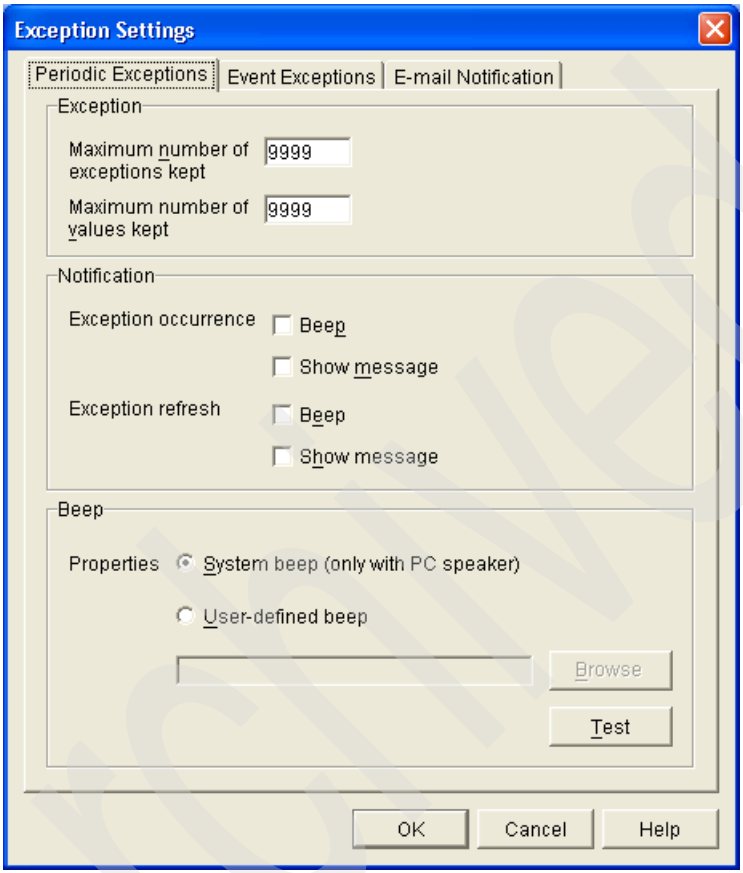


Figure 4-50 Exception Settings dialog

There is a settings page for both periodic and event exceptions, and one page for e-mail notification.

Periodic and Event exceptions pages contains the following sections:

- Exception
  - Maximum number of exceptions kept

    Controls only how many are shown on the screen log, not how many are kept in the database. This value does not affect how many log entries are actually kept in the performance database exception log table. So if you

lower the number one day, you can increase it the next and see older log items again.

- Maximum number of values kept

  This does not apply to multiplatforms and should be ignored.

- Notification
  - Exception occurrence

    Choose to receive pop-up message or sound. We do not recommend the pop-up message, as it does not present any useful information other than to point you to the log.

  - Exception refresh

    You can have a sound or pop-up occur when the log screen is refreshed. Since the default interval is 60 seconds, we do not recommend using this option unless you have a very long interval. It does not provide much additional value.

- Beep

  Here you can choose just what sound you want to hear when an exception is added to the log, if you select the Beep box as above. It can be useful if you set the periodic and event exceptions to use a different sound. You can leave the Exception Processing window closed and just listen for the deadlock sound, for example. Otherwise, we recommend a normal system beep, if you use beep at all.

- E-mail notification page

  In Figure 4-49 on page 213, we show how to configure the e-mail settings for the PE Server. That configuration does not include who should receive the e-mail alerts. This is set up in the E-mail Notification tab, as shown in Figure 4-51.

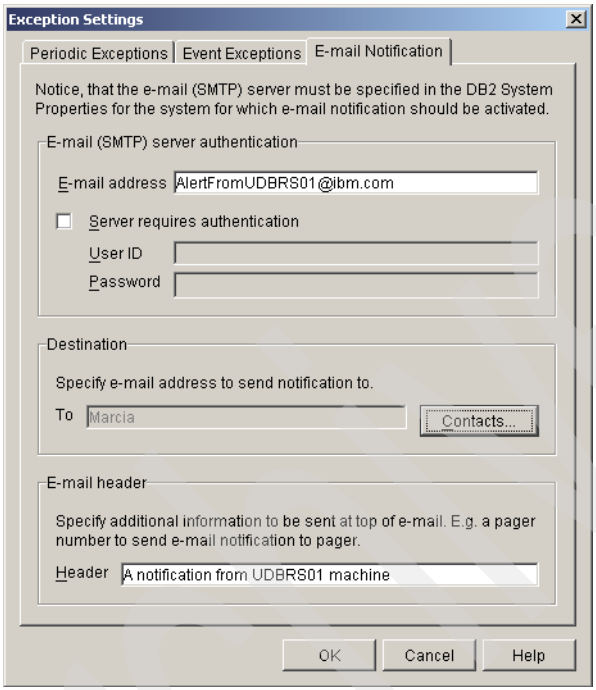


Figure 4-51 Exception E-mail Notification window

  - E-mail address

    This is the address that will be used in the FROM field the e-mail. This should be appropriate for your company.

  - Destination

    This field is for the recipient(s) who will receive the e-mail alerts. To create the list, use the Contacts button, which opens a simple dialog to add names; it is not described here.

  - E-mail Header

    This field will contain the additional text to appear at the top of the e-mail. It is *not* the subject line. The subject line will always be “IBM (C) DB2 Performance Expert” and cannot be changed.

Step 3: Activate periodic exception processing

Now you have everything in place to successfully activate the periodic exception processing. To open the activation window, you can either open it from the Exception Processing window, or use the context menu on the monitored instance in the System Overview window or the Exception Processing window. (see Figure 4-52.) You can use the tool bar icon, or use the menu, by selecting **Exception Processing** → **Activation - Multiplatforms**.

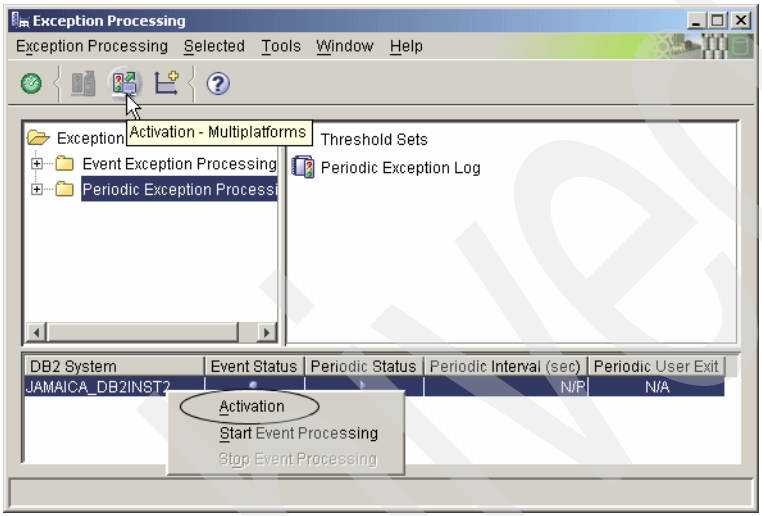


Figure 4-52 Opening the exception processing Activation window

In our example, we want to activate one threshold set on one server. Figure 4-53 shows the exception processing window, with a before and after look at activation, and shows the relationship between the monitored instance properties setting for exception interval and the exception processing multiplier. This window is used for starting and stopping exception monitoring for both periodic and event exceptions.

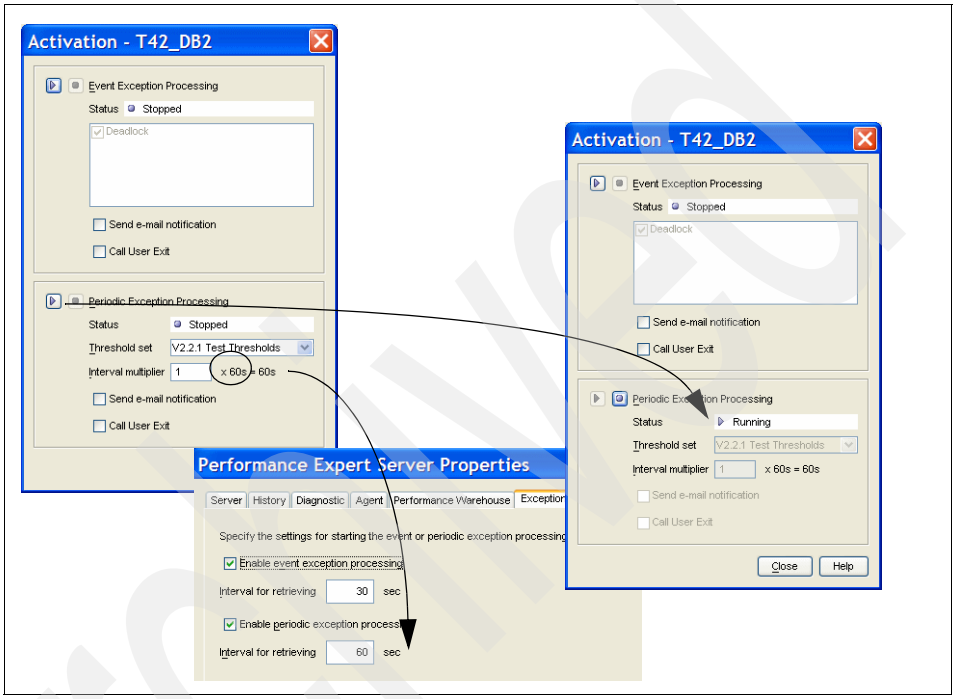


Figure 4-53 Periodic Exception Processing Activation window

There are four steps to activate the threshold set:

1. Choose the threshold set

   When status is `stopped`, you can then select which threshold set to activate.

   If you have previously activated exception processing for one threshold set, you must stop it before you can switch to a different threshold set. The current status is shown on the Activation window.

   In our example, we are using a threshold set named Test Thresholds.

2. Set the threshold evaluation interval

   The Interval multiplier field tells the PE Server how often to check for violations of the thresholds. This is called a “multiplier” because it is derived from the retrieval interval specified on the Exception page of the PE Server

Properties window. Each monitored instance can have a different retrieval interval, so if you activate a threshold for multiple monitored instances, you can have them evaluate the condition at the frequency appropriate for each instance. The default value for the multiplier is 1; if you type a different multiplier, the actual frequency will be displayed in the area next to this field.

To set another interval, change the Interval field on the Exception page of the Performance Expert Server Properties window.

In our example, our server interval is set to 60 seconds, and we selected a multiplier of 1, so it will check every minute.

3. Enable E-mail (if configured)

   The e-mail check box will be shown only if you previously configured e-mail for the PE Server. This is discussed in “Step 2: Configure properties and settings for exception Server” on page 212.

   In our example, we did not ask for e-mail alerts.

4. Enable user exit (if configured)

   The user exit check box will be available only if you previously configured user exit for the instance. This is discussed in “Step 2: Configure properties and settings for exception Server” on page 212.

   In our example, we did not invoke a user exit.

5. Click Start button

   Finally, click the **Start** button () to activate. The status will change to Running. Close the activation dialog.

   Once the exception processing has been activated on the PE Server, you can close the Exception Processing window if you like. If a violation occurs, you will be notified via whatever method you chose when you did the configuration.

Activate threshold set for multiple instances

You can activate periodic exception processing on more than one monitored instance. You can activate them individually or as a group. The Activation window is shown in Figure 4-54. The usage is the same as described earlier, but your selection will apply to whichever system you have checked in the left side of the screen.

Different threshold sets can be activated on different systems.

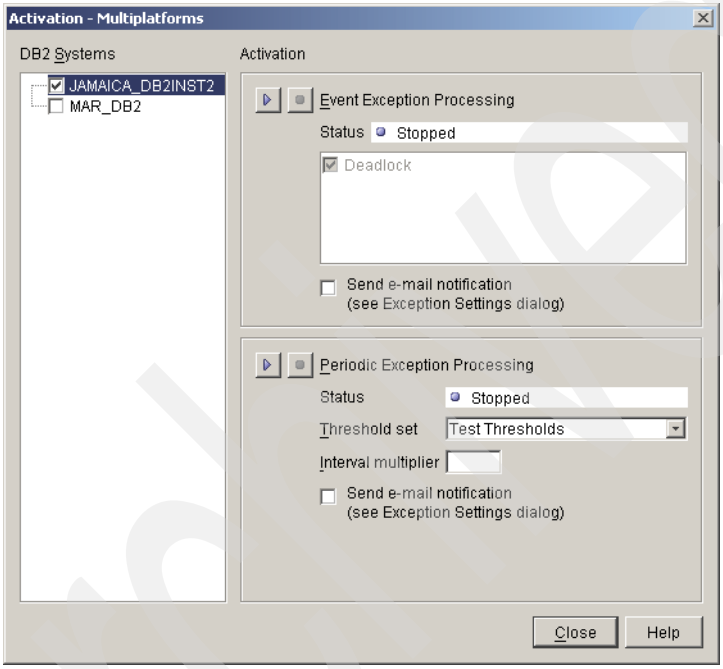


Figure 4-54 Exception Processing Activation for multiple monitored instances

Step 4: Examine threshold exceptions

As the PE Server evaluates the thresholds you set, the PE Client will communicate with the server and refresh the Exception Processing log at the interval you specified. In our example, it is every minute. In V2.2, the PE Client will now also show a graphical indicator on the System Overview window when a threshold has been violated. These indicators are called “signal lights.” When you are looking at Statistics or Application details, you will also see individual counters highlighted if it has exceeded the warning or problem threshold. This is described in “Viewing periodic exception signal lights” on page 227.

Viewing periodic exceptions in the exception log

If the server finds no violation, no notification occurs. If the server finds a violation, it will make an entry into the periodic exception log, which will display in the Periodic Exception Processing Log screen. This type of notification always happens. If you have also configured the PE Client for pop-up messages and audible beeps, the client will handle that notification. If you set up e-mail notification, the PE Server will send the e-mail. When notification occurs, you need to open the Periodic Exception Processing Log to see the details of the violation.

To open the Exception Processing log, press the Ctl+Alt+E keys while inside PE, and select the Periodic Exception Log in the tree pane. You can also use any of the several other ways to open the Exception Processing dialog.

In our contrived example, we ran some small queries against SAMPLE just to prime the buffer pool. It settled at a spot just above the threshold, thus ending the violation state. Then it fell below again and created a new violation.

- For the Data BP hit ratio on the SAMPLE threshold, the following sequence of events occurred:
  a. The first violation occurred at 6:24:20 PM (see the Begin column in Figure 4-55).
  b. It was still in violation at 6:26:33 PM with a 80.48% hit ratio (see the Last Change column in Figure 4-55).

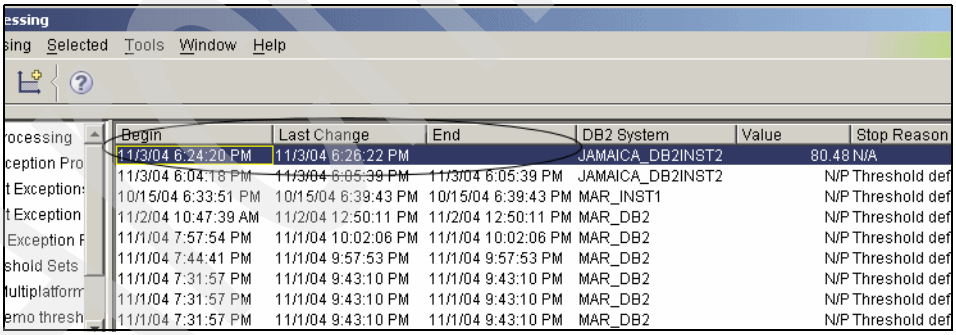


Figure 4-55 Violation #1

c. It was still in violation at 6:35:29 PM with a 49.55% hit ratio (see Figure 4-56).

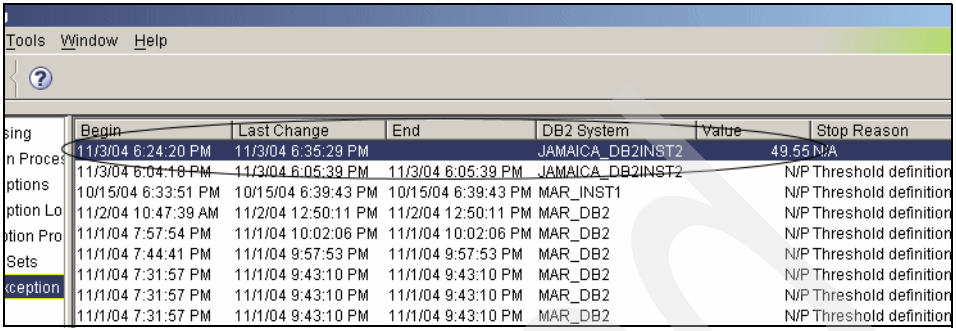


Figure 4-56 Violation #1 still active

d. At the 6:41:31 PM evaluation time, the BP ratio was at 86.31%, above the threshold and no longer in violation (see the End and Stop Reason columns in Figure 4-57).

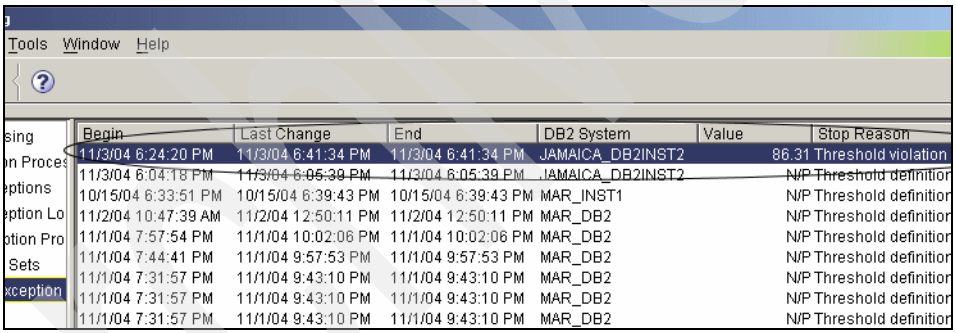


Figure 4-57 Violation #1 ended

e. At 7:01:50 PM, the hit ratio dropped below the threshold again (See the Begin column in Figure 4-58 on page 223).

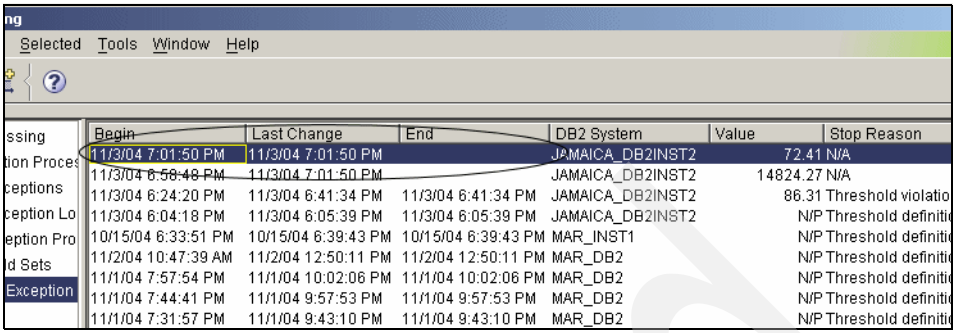


Figure 4-58 New violation for Data BP Hit Ratio on SAMPLE

This scenario demonstrates how a threshold can be exceeded, and remain exceeded, and how PE will show this in the log. If a violation remains active or open for a long time, you probably have a more critical problem than if it happens once or twice over a day. Understanding how PE displays the violations will help you in assessing the performance of your system.

Understanding the Periodic Exception Log

The columns of the Periodic Exception Log are:

- Begin: The time stamp for when the violation was first detected.
- Last Change: The most recent refresh when the violation was still active.
- End: The time the violation ended based on Stop Reason.
- DB2 System: The monitored instance where the violation occurred.
- Exception Category: As defined in the threshold.
- Exception Subcategory: As defined in the threshold.
- Exception field: The counter name to evaluate as defined in the threshold.
- Value: The actual counter value that exceeded the threshold setting.
- Warning: Threshold comparison value.
- Problem: Threshold comparison value.
- Stop Reason: Why the threshold is no longer in violation. Possible values are:
  - n/a: Violation is still active.
  - Monitored object does not exist any more: For an application threshold, this means the application process is gone, so it can no longer be evaluated
  - Threshold violation finished: A threshold was violated, but is no longer in violation.

– Threshold definition deactivated or removed: If you stop periodic exception processing completely, disable one threshold within an active set; or delete a threshold from an active set, any active violations are marked with this reason, and no further evaluation is done.

Note: You can rearrange the columns on the Log screen by dragging them by the column heading. The column width is resizable by dragging the column heading separator. The default column order has the exception value too far to the right, so it was moved for the screen images here. We suggest you rearrange them to your satisfaction.

Viewing periodic exception details

For any violation shown in the log, you can double-click the log entry to view the details for the exception (see Figure 4-59 on page 225). The Periodic Exception Details window will show additional information about the exception. The type of additional data will vary based on the type of counter used for the exception. For Application exceptions, the Application ID is included, so you can look for that application in the Application Summary window.

If you have an e-mail alert configured, the e-mail will contain the same information that you see on the details screen, but in text format.

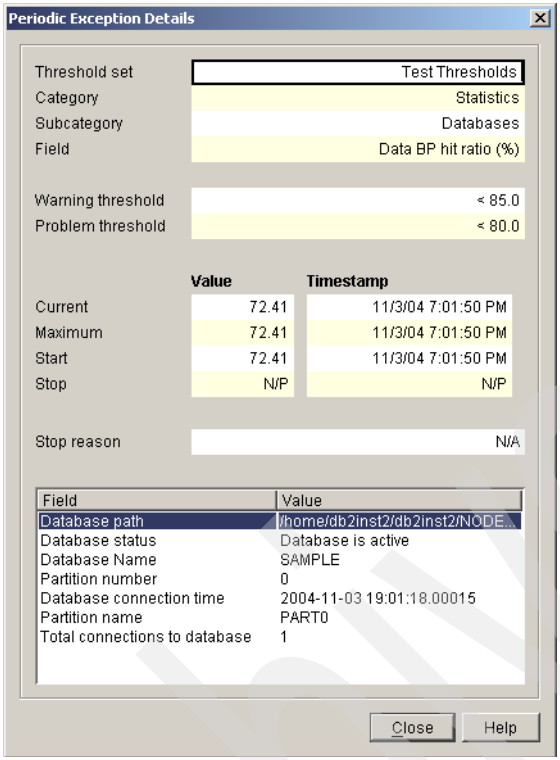


Figure 4-59 Exception details for reappearance of threshold violation

Comparing Periodic Exception Log with System Health chart

Figure 4-60 shows a system health graph from the same time period as our example, showing the same kind of threshold violations but in graphical format. We show this as an example for deeper understanding of the different elements in PE and how information can be presented in different ways. Health charts can depict current or historical exceptions, but do not provide alerts, nor do they show detail for a particular point in time. Threshold sets can report on individual exceptions with detail about an incident, with notification of the exception available. These two approaches complement each other.

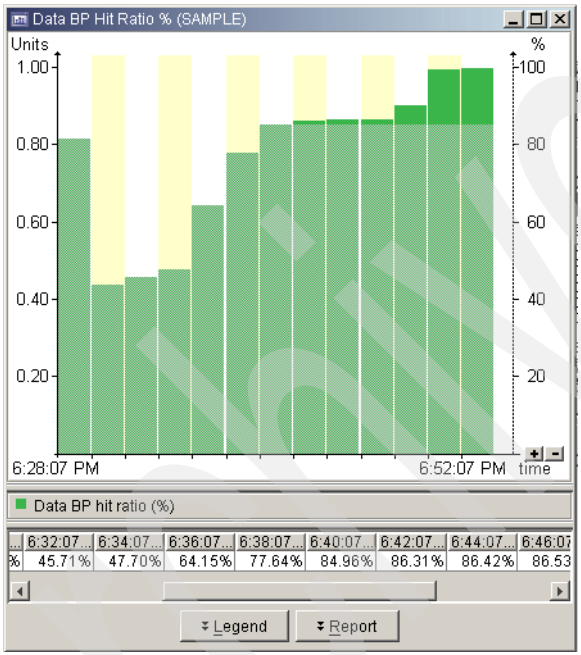


Figure 4-60 System Health chart showing hit ratio threshold exceeded

Performance impact of using periodic exception processing

The chart shown in Figure 4-60 was displayed using History mode in System Health. You may notice the times on the report do not match the times on the Periodic Exception Log. Periodic Exception processing uses its own snapshot monitoring interval, which you set up when you activate the threshold set. This interval is totally separate from the History snapshot collection interval configured in PE Server Properties for the monitored instance. You should understand this and the impact this might have on your monitored instance, if you configure very frequent snapshots. You should determine the monitoring frequency that is meaningful for your environment. The overhead incurred for any

snapshot is dependent on the volume of activity on your system, the type of snapshot being taken, and the settings of your default monitor switches.

There is no reason not to use both history collection and periodic exception processing. Each is useful in its own right, and they can complement each other. We only mention this here as a point to consider as you implement DB2 Performance Expert in your environment.

Viewing periodic exception signal lights

Performance Expert V2.2 has enhanced the exception processing features. In the System Overview window on the right-side pane, you will now see not only deadlock event information, but also warning and problem threshold violations when they occur. Warning violations are indicated by a yellow triangle symbol, while problem threshold violations are shown with a red X symbol. Deadlock events are shown as an exclamation mark. In Figure 4-61, we see an example of several thresholds in violation, as well as a deadlock. Deadlock events are discussed in 4.2.2, “Deadlock event exceptions” on page 233.

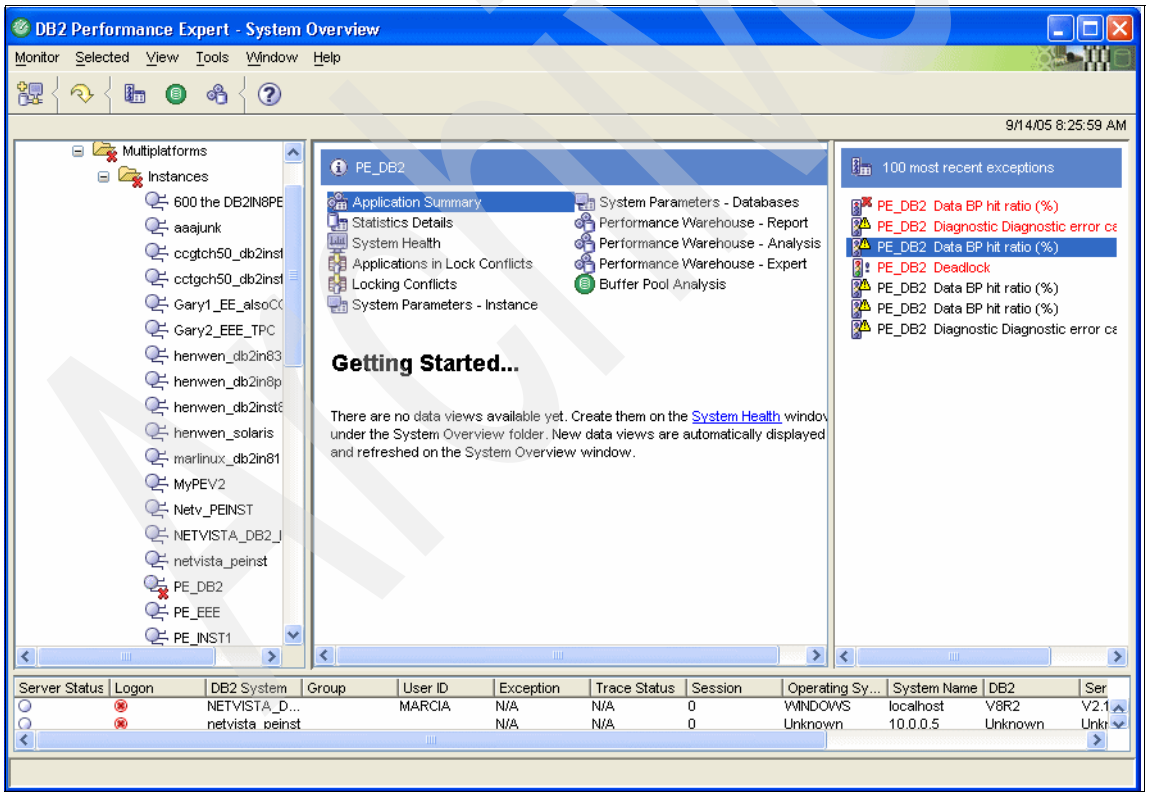


Figure 4-61 System Overview showing signal light for periodic and deadlock exceptions

In the individual Statistics or Application details screens, if a counter has a threshold defined and it has been violated, the field itself will also be highlighted with a yellow triangle (warning) or a red X (problem) indicator. When you hold the mouse (hover) over the field, the details of the violation are shown in a pop-up window.

For example, we look at an example of a BP data hit ratio violation, as shown in Figure 4-61 on page 227. We right-click the exception to show the context menu. You can choose to view the details of the violation, just like you see when looking at the Periodic Exception Log and described previously. You can also choose to view the Statistics Details (or Application Details, if the exception is in the Application category). We choose to see the Statistics Details. The Performance Expert client will navigate to History mode to show the closest history snapshot to the first time the threshold was violated. This is shown in Figure 4-62. In this case, the violation time was close to 9:22 AM.

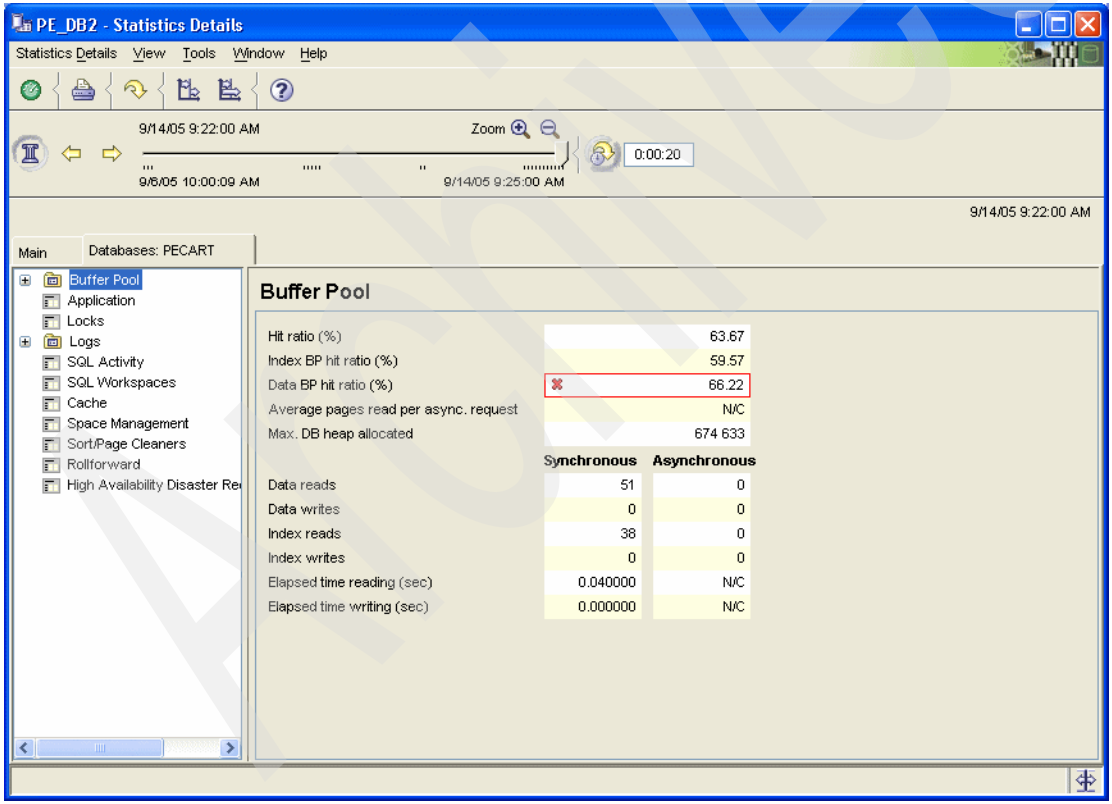


Figure 4-62 Periodic exception field is highlighted in history mode

If you want to see details of the violation, move the mouse pointer to the violation icon (red x and yellow triangle) and leave it there for a moment, then a pop-up “hover help” will appear, as shown in Figure 4-63.

Tip: The pop-up detail will only appear if you point to the icon, not if you point to the value.

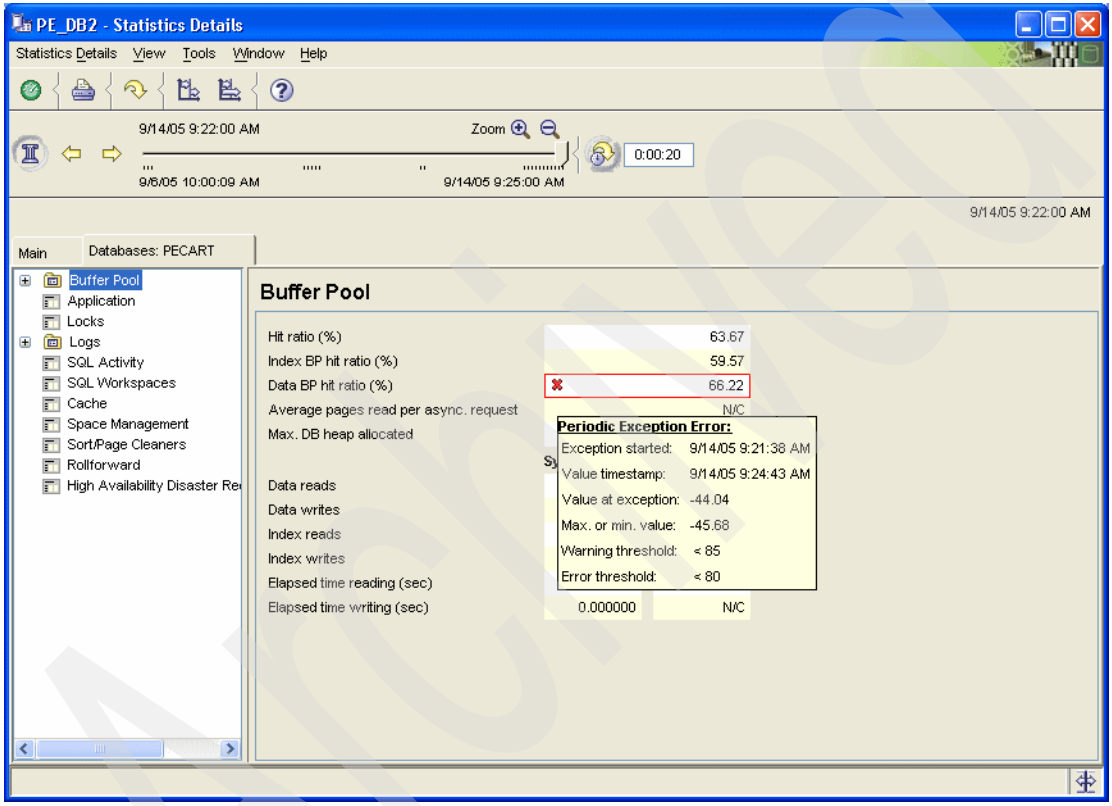


Figure 4-63 Periodic exception detail in hover pop-up window

You might notice in the figure that the values shown in the pop-up do not match what is on the Statistics Details window. This is because the history snapshot (which is what you are looking at in Figure 4-63 and is from 9:22:00 AM) was not the exact moment of the violation. In this case, the first violation was detected at 9:21:38 AM. The PE client will try to come as near to that time as it can when showing you the history screen. The periodic exception evaluations are done on a different thread than the history snapshots, so the two may not exactly match. So you may see different values when you compare the exact exception violation

to the history screen. This should not be a problem, because you do not need to know precise detail about these violations; you are looking at general health of your system.

If you are just looking at the snapshot screens, and you do not notice a violation on the System Overview window, the field will still be highlighted when a violation occurs. You do not need to be in history mode. An example of this is shown in Figure 4-64, where we see the DIAGLEVEL configuration parameter has been set to 2, which is defined as a warning. We do not want this to be 2, so we can go investigate what happened.

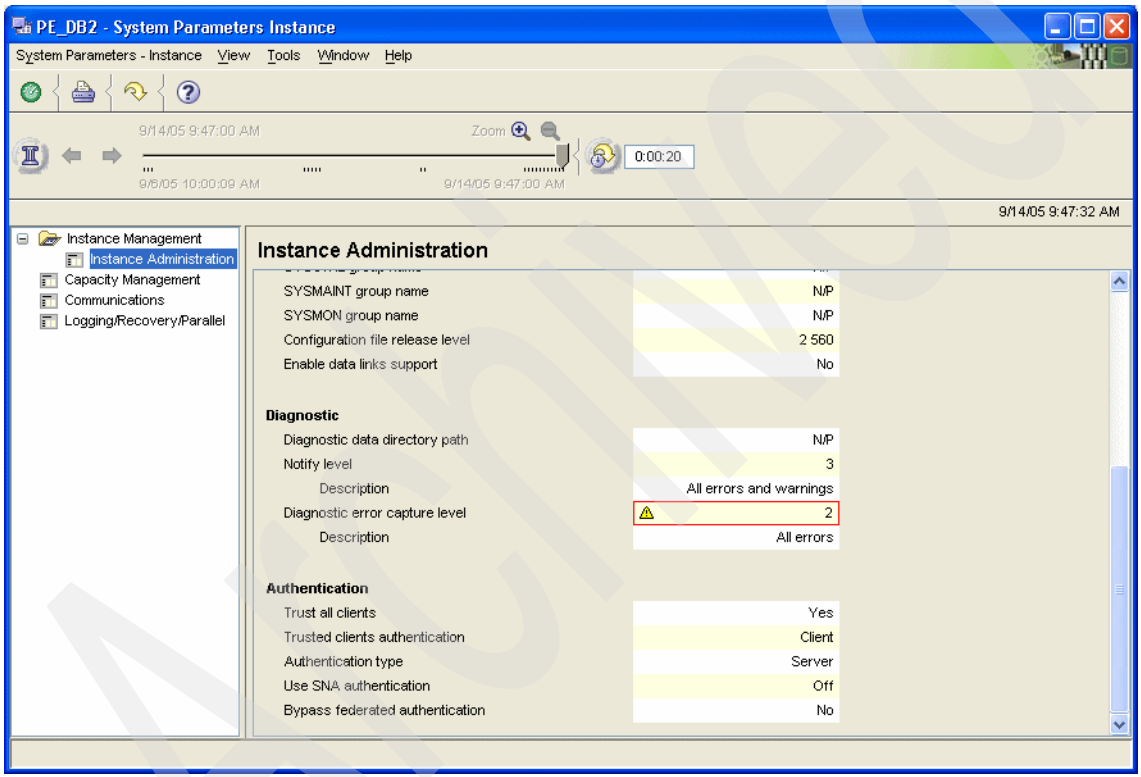


Figure 4-64 Periodic Exception shown in current snapshot

When a threshold is no longer in violation, the warning or problem icons will disappear. For example, we changed the DIAGLEVEL back to 3, so now the window looks like Figure 4-65 on page 231.

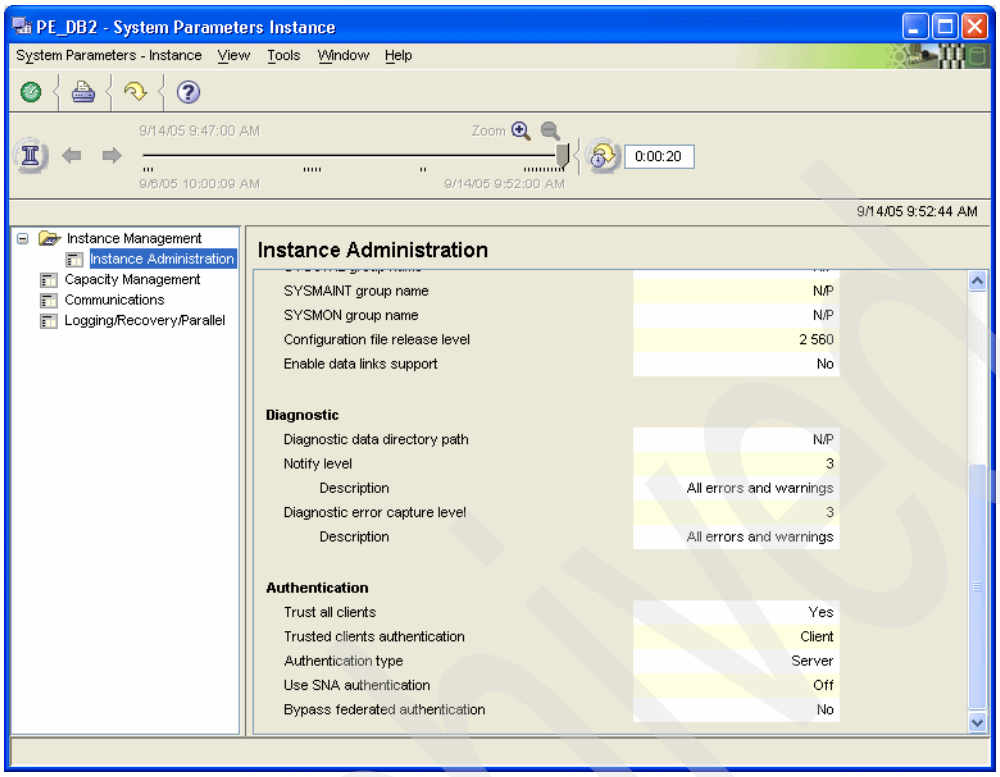


Figure 4-65 Periodic exception no longer shown, violation has ended

Step 5: Stop periodic exception processing

To stop a running threshold set, return to the Activation window (see Figure 4-53 on page 218) and press the stop button ( ). The PE Server will terminate the monitoring threads and mark any active violation as complete, as described above.

Another way to start or stop the exception processing is to use the monitored instance context menu on the System Overview screen. Right-click, select **Exceptions**, and you are offered the option to start or stop the event exception processing, periodic exception processing or both.

Keeping periodic exception processing active while not logged on

Once you activate periodic exception processing, it runs on the PE Server until you explicitly stop it. The activation of a threshold set must occur from the PE Client, but once activated, the PE Client is no longer required, and you can log off and close the PE Client.

If you log off from the monitored instance while a threshold set is active for the instance, you are prompted with the question shown in Figure 4-66.

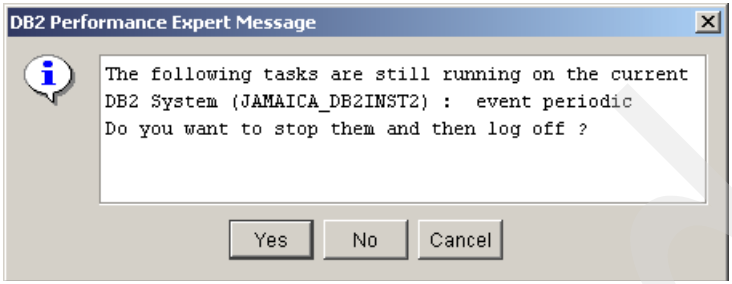


Figure 4-66 Logoff prompt when exception processing is active

This questions is really just asking if you want to turn off exception processing at the PE Server. The results of the answers are:

- Yes: Exception processing is stopped. You are logged off.
- No: Exception processing stays active on server. You are logged off.
- Cancel: Exception processing stays active on server. You are not logged off.

If you keep exception processing active, it will continue to monitor for violations and log them to the exception log. If you have configured e-mail notification, the PE Server will send e-mail for any violation that occurs when it occurs.

Note: If more than one threshold is violated in one interval, the PE Server will include all of the violations in one e-mail, rather than sending multiple notes.

The next time you log on in the PE Client, any violations that occurred while you were out will be displayed in the Periodic Exception Log.

Other Periodic Exception Tips

Here are couple of tips for using Periodic Exception.

- **Where are the threshold set definitions stored?**

  Threshold sets are created and maintained at the client level, and stored as XML files on your PC. There is not currently a supported way to copy or share threshold sets between different PE users. When a threshold set is activated from the client, the definition is stored temporarily at the PE Server so the server can evaluate the threshold condition. When you deactivate (stop) the periodic exception processing, the threshold set definition is removed from the server. This approach makes it easy for you to activate threshold sets for all the monitored instances you support, instead of having to create them for each one.

Because threshold sets are created and saved locally, you do not have to be logged on to a PE Server to create them, but you will need to log on before you can activate the threshold set.

- **Multiple PE Client users looking at exceptions**

  When you activate a threshold set, the PE Server connects the processing of that set to your user ID. If you log off the PE Client, and later log in with a different user ID, you will not see the exceptions that occurred for the set activated by the first user ID. Likewise, if you go to another PC and log on to the PE Client using your original user ID, you *will* see the exceptions in the log, but since the threshold set itself is stored on the original PC, you would not be able to make changes to the threshold set from the second PC.

  Each user on a PE Client can activate threshold sets. Each user will only see the exceptions associated with their active set. If every user creates and activates the same thresholds, the PE Server will still treat those as separate thresholds and will issue separate snapshots. We recommend that if you have multiple DBAs using Performance Expert, that you coordinate the use of Periodic Exception Processing. One way to accomplish this is to use e-mail alerts that are sent to multiple users.

4.2.2 Deadlock event exceptions

If event monitoring is enabled for your monitored database, the PE Server will create the deadlock event monitor as part of its startup routine. This activity will be active and collecting any event exceptions into the exception log until you disable monitoring of the entire instance or you disable event monitoring for the monitored database. To enable/disable deadlock event monitoring for the monitored database, you must use the **`peconfig EVMON`** command. See 3.1.1, “PE Server configuration and related tasks” on page 47 for a description of using **`peconfig`**.

Compare this behavior to the Periodic Exception Processing that is controlled at the PE Client. They both monitor for exceptional conditions, and present the exception to you in the same way, but the underlying monitoring mechanism is different. It is not important for you to understand this difference as you work with the two types of exception processing.

In the PE Client, when you activate Event Exception Processing, you are only controlling how the deadlock events are presented to you in the client. By default, the Event Exception Processing is not turned on in the PE Client, even if the PE Server has EVMON for a monitored database. The PE Server has stored all the event exceptions into the log, so if you want to see them, you have to activate the Event Exception Processing.

The activation steps are the same as for Periodic Exception Processing; they will not be described in detail here, but the Activation windows are shown in Figure 4-67 on page 235. The only optional item here is to have PE Server send you e-mails when a deadlock occurs. The rest is the same as for Periodic Exception Processing activation.

The PE Server System Properties Exception page (see Figure 4-53 on page 218) does have a configuration for setting the interval for retrieving event exceptions. This tells the PE Server how long to wait between checking the event monitor for deadlocks. At that interval, the PE Server collects any deadlock information from the event monitor file, and stores it in the performance database. The PE Client will then show the deadlock information on the screen.

The default event exception retrieval interval is 60 seconds. You should consider how often you need to be informed of deadlocks in your environment. Maybe you cannot tolerate any deadlocks at all, and deadlocks occur infrequently, so it might be reasonable to check every 60 seconds, even at the expense of some overhead in looking often for something that is rarely present.

If you are monitoring a development environment, for example, where deadlocks might be more likely, but also less critical (and the developer or tester probably detects it anyway), you might want to set the interval longer, such as five minutes.

There is no strict rule for how to choose a value.

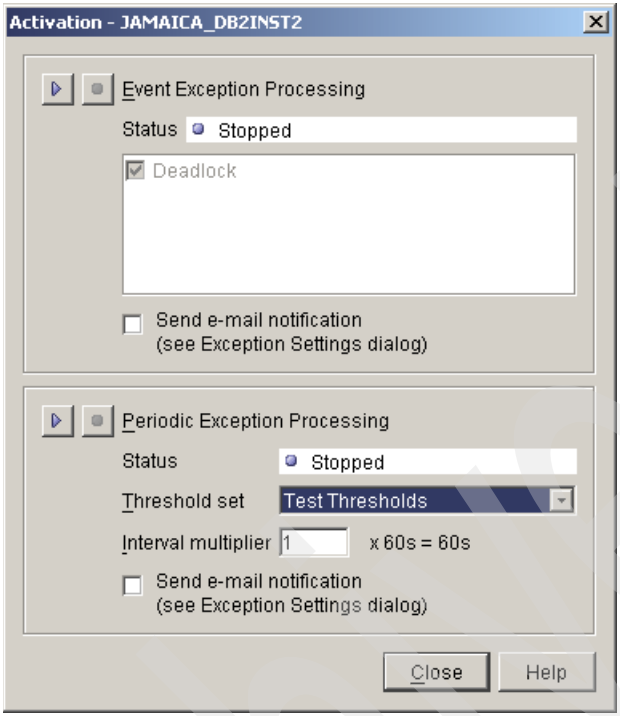


Figure 4-67 Event Exception Processing activation window

After activating Event Exception Processing at the client, when a deadlock event occurs, you can view it in the Exception Processing Log. PE Client will also display the 20 most recent deadlocks in the System Overview window, as shown in Figure 4-68 on page 236.

In PE V2.2, on the System Overview window, the graphical indicator for deadlocks has changed from an asterisk to a traffic light with an exclamation mark. This is shown in Figure 4-61 on page 227. The rest of the behavior for deadlock exceptions has not changed.

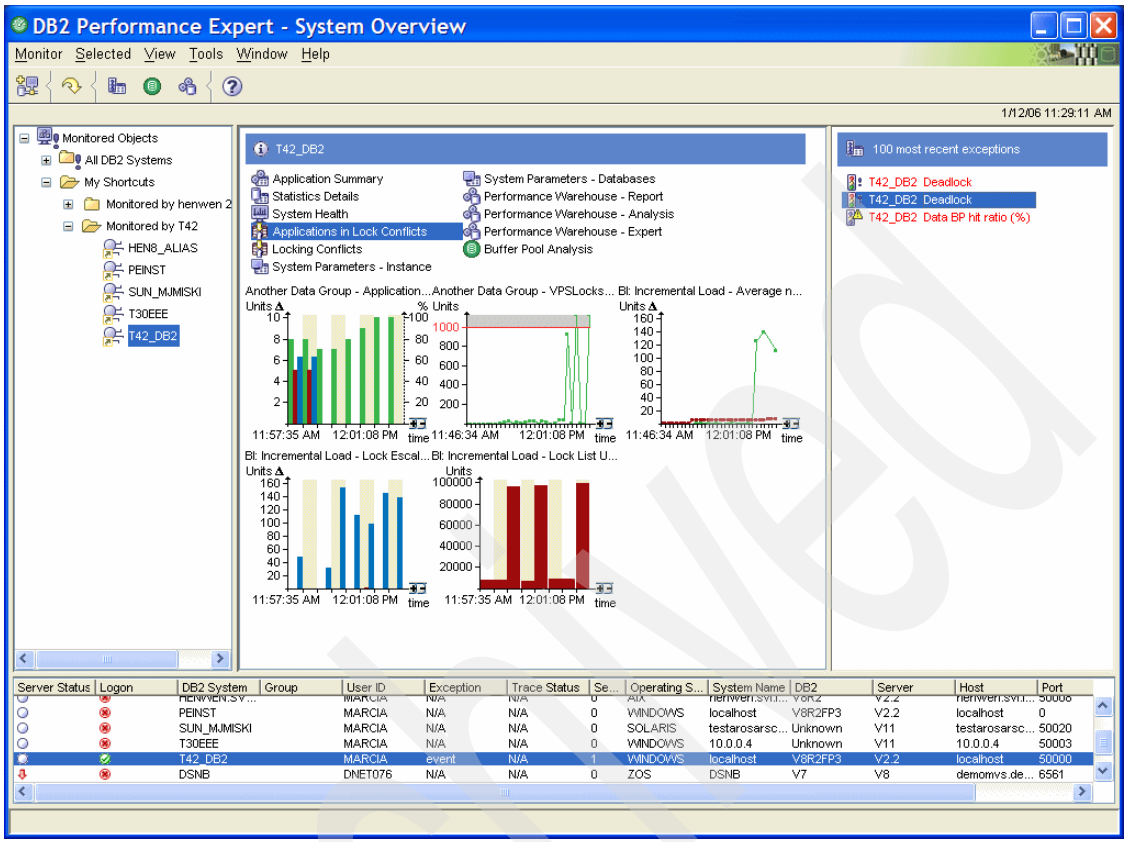


Figure 4-68 System Overview showing deadlock exceptions (V2.2.0.1)

A deadlock event will show in red until you look at it. Deadlocks are also shown in the Event Exception Log area of the Exception Processing window, as shown in Figure 4-69.

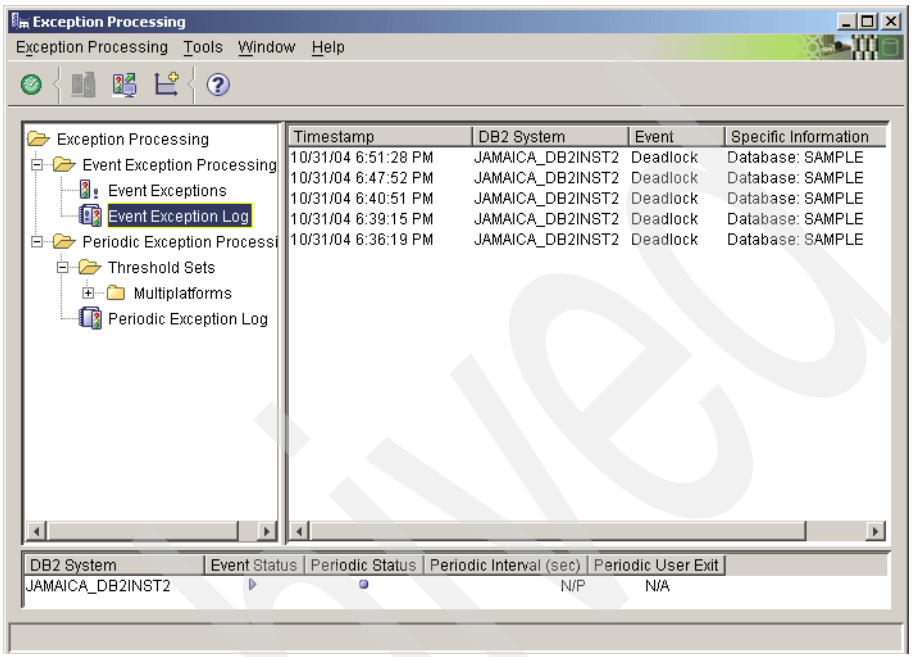


Figure 4-69 Event Exception Log window

To show the details of a deadlock, double-click the deadlock from either the System Overview window, or the Event Exception Log. Both will open the Deadlock Event Details window as shown in Figure 4-70. Details for each participant in the deadlock are shown, including the statements that caused the deadlock. If the application is still active in the system, you could go to the Application Summary screen and drill down to see more information about it.

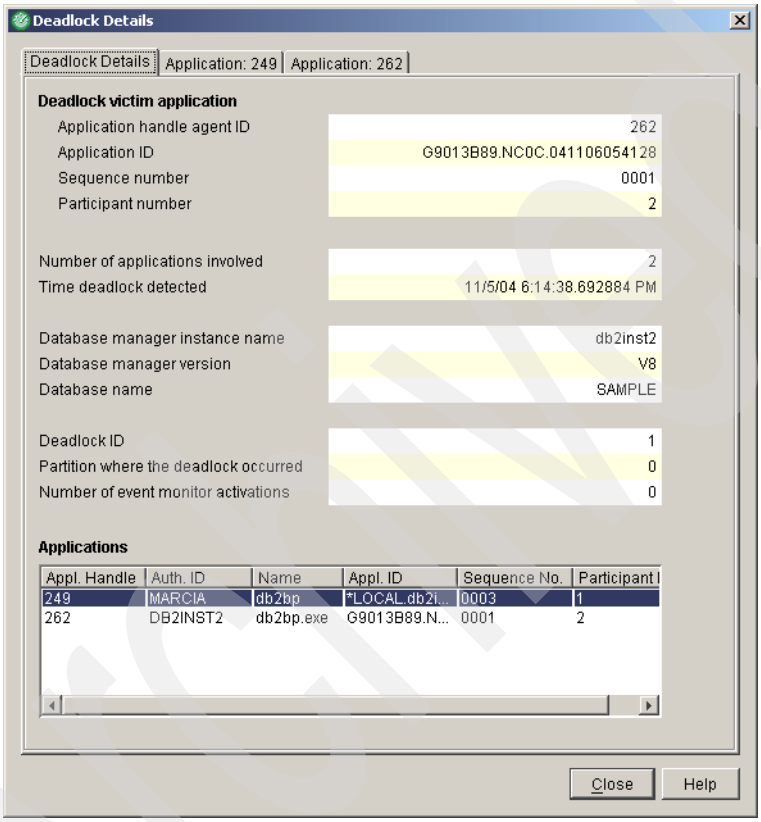


Figure 4-70 Deadlock Details window

Tip: In DB2 UDB V8, when you create a new database a deadlock event monitor called DB2DETAILDEADLOCK is automatically created and enabled. If you plan to use Performance Expert deadlock event monitoring, you should drop, or at least disable, this default event monitor so you do not incur unnecessary overhead.

5

Features and functions - long-term monitoring

This chapter shows you the PE features and functions used for long-term monitoring. In this chapter, we introduce the following functions:

- Performance Warehouse
  - Reports
  - Queries
  - Rules of Thumb
- Buffer Pool Analysis

5.1 Performance Warehouse

When monitoring performance for any system, it is important to look at the performance data over long periods of time. Short-term monitoring is useful, but does not give insight into the long-term effects of system changes or aid in capacity planning for the future. DB2 Performance Expert (PE) addresses this need by capturing the short-term performance snapshot data and saving it into a long-term Performance Warehouse. While the history data of PE is only retained for a short time (some hours/days), once data is in the Performance Warehouse, the data is available for reporting and trend analysis over much longer periods.

In this section, we describe some historical background of Performance Warehouse, its components, and how they interface with the other areas of PE. We discuss in detail the life cycle of performance data and how to use it to determine which components of Performance Warehouse (PWH) will best help you in assessing your DB2 performance over the long-term. We also provide some examples, suggestions, and tips for getting the most out of PWH.

5.1.1 Background of Performance Warehouse

The same Performance Expert client is used for monitoring DB2 databases on both multiplatform (distributed) environments and z/OS. The DB2 Performance expert product is an evolution and merger of the z/OS products DB2 Performance Monitor and DB2 Buffer Pool Analyzer, with support added for multiplatforms. The user interface has remained and been modified somewhat to support all environments, but because it was created to only support z/OS, some of the mechanisms and terminology can seem mainframe oriented. This may not be an issue if you, as a DBA, support databases in both environments, or if you are a multiplatform DBA who has previous experience in a mainframe world. If, however, you are not familiar with z/OS or DB2 on the mainframe, you may find navigating the Performance Warehouse a bit of a challenge. We hope to allay your fears and show you the fastest path through PWH to allow you to accomplish your tasks.

In PE for z/OS, loading data into the PWH is done by running specific load processes. These encompass various traces or other collection processes to gather the performance data. In PE for multiplatforms, the majority of PWH data based on snapshots is captured automatically. To capture PWH data based on applications, you still do that manually with a collection process called a SQL activity trace, which is discussed later.

In the z/OS environment, the Performance Warehouse is a separate entity apart from the history data. In the multiplatform environment, PWH data and the history data are in the same database. In the multiplatform environment, the

Performance Warehouse is simply a set of tables under a different schema within the performance database for the monitored instance. The PWH tables are created under schema PWH, while the other PE history and control tables are under DB2PM.

Tip: One of the z/OS predecessor products to PE was called DB Performance Monitor (DB2PM.) That is why we still see that name in places within Performance Expert.

5.1.2 Life cycle of performance data

It is important to understand where the PWH data comes from so you can choose the appropriate components of PWH to help you analyze your database performance. In this section, we describe how data is collected by the PE Server, how it ends up in the Performance Warehouse, and how you can get it out of the warehouse. This is also shown in Figure 5-1 on page 242.

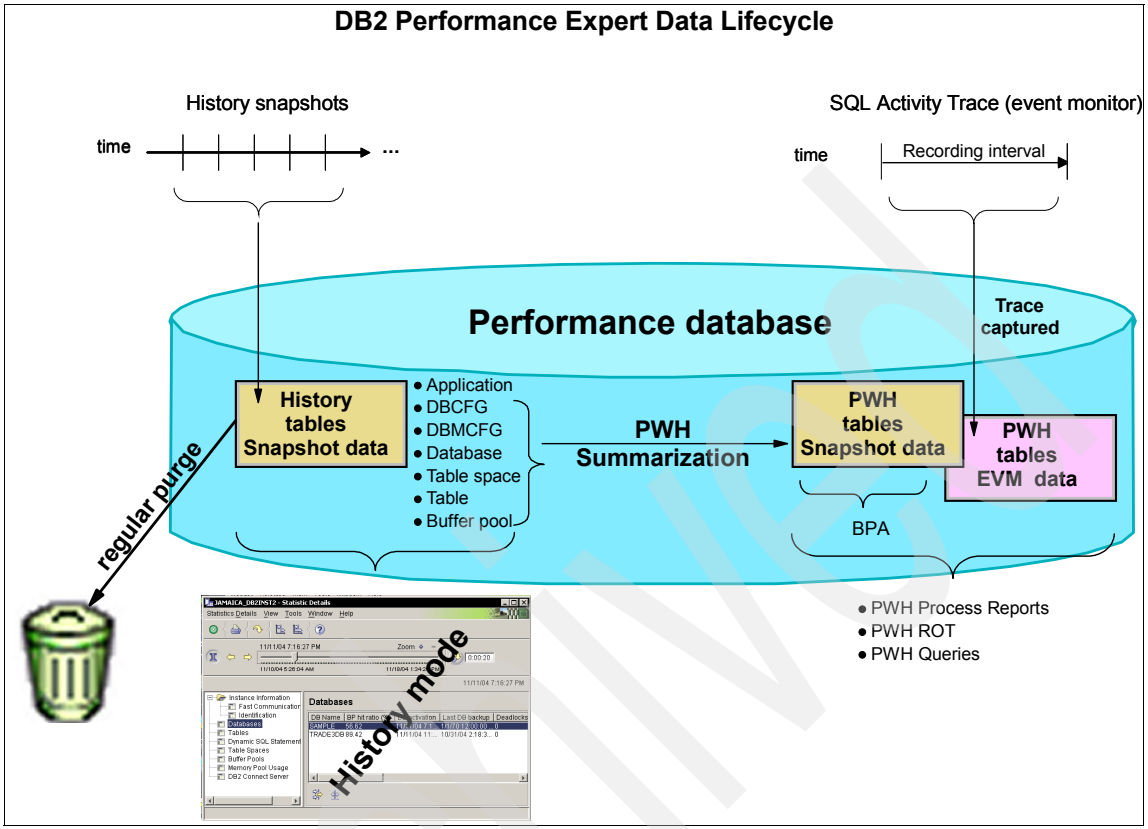


Figure 5-1 DB2 Performance Expert data life cycle

Getting data into the Performance Warehouse

There are two ways performance data gets into the PWH:

- Migration of history snapshot data and operating system (OS) data
- Collection of SQL activity trace event monitor data

These methods are described next. You should refer to Figure 5-1 when reading the descriptions below.

Migration of history snapshot and OS data

The PE Server makes periodic snapshots of system activity (DB2 and OS) at the interval you specify in the PE Server Properties - History page. This interval is called the *Recording interval*. As shown in Figure 5-2 on page 243, the history interval on our server is set to 60 seconds for application and locking, 300 seconds (five minutes) for statistic details, dynamic SQL and OS status, and 1800 seconds (thirty minutes) for OS system and storage information. History

data is stored in the performance database, and the data is used when you display data in history mode on any of the screens.

PE will keep history data in the history tables for as long as you specify in the Timeframe history property. This is usually referred to as the retention period. As shown in Figure 5-2, the history retention on our server is set to 50 hours. The history cleanup process wakes up periodically and deletes any history entries that are older than the retention period. This process runs regularly and is not configurable by the user. History data that has been deleted is not recoverable.

Important: The history tables have large amounts of insert and delete activity, and so are excellent candidates for regular runstats and reorgs. PE provides the runstats and reorgs scripts pereorgV8.ddl and perunstats.ddl. These scripts are in the bin directory of the PE server installation directory.

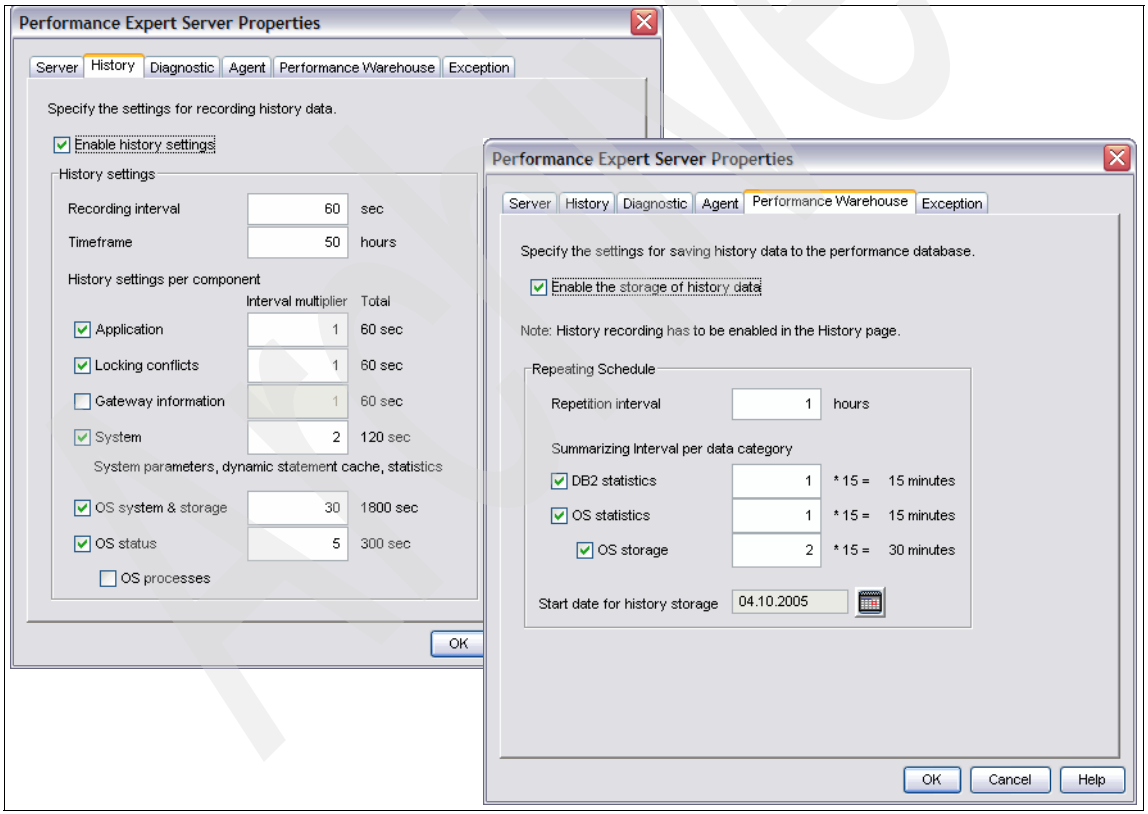


Figure 5-2 PE Server Properties - History and PWH

Note: In the PE Client, if you simply refresh the screen while you are looking at it, or use autorefresh, that snapshot data is *not* stored in history tables. Only the data captured by the PE Server history collection thread is stored in history tables.

At an interval configured by you, the PWHLoad thread will wake up and begin the task of taking the previously captured history snapshot data and migrating it into the PWH tables. This interval is called the *Repetition interval*. As shown in Figure 5-2 on page 243, on the Performance Warehouse page, the summarization interval on our server is set to one hour. The repetition interval must be less than the history retention period, or else you will not have any data to summarize!

A second PWH configuration value, the “Granularity for summarizing the history data”, is also required for setting up the PWH. This value controls how many snapshot rows are collected into one row of PWH data. This is often referred to as the summarization or aggregation setting. We may use these interchangeably in this book. As shown in Figure 5-2 on page 243, the Granularity on our server is set to 15 minutes for DB2 statistics and OS statistics values and 30 minutes for OS storage values.

PE will validate that the values you enter on these two properties pages are compatible with each other. For example if you configure to retain history for eight hours, then try to set the Summarization interval to 24 hours, you will get an error message.

Unlike history data, Performance Warehouse data is kept in the database until you manually delete it. There is currently no automated way to purge or archive the PWH data, but having a good history of performance data is actually very useful for reporting. If you should need to purge hold performance data, you could write a delete script. In Appendix A, “Overview of the Performance Warehouse database“, in *IBM DB2 Performance Expert for Multiplatforms, Workgroups, and z/OS IBM DB2 Performance Monitor for z/OS Monitoring Performance from the Workstation*, SC18-7976, there is some information about the PWH table organization.

Note: The Application and Locking conflicts snapshot data are not migrated into the PWH. Only statistics from database, tablespace, table and buffer pool snapshots, DB and DBM configuration changes, and OS data (except process data) are stored in PWH.

So in our example, we have told the PE Server to collect history snapshots on statistics counters every two minutes, and wake up every hour to summarize the

history data into 15 minute groups. In our case, we will have seven or eight rows (depending on the exact moment of the snapshot) of history snapshot data for every one row of PWH data. This is what we call the aggregation.

The smallest aggregation allowed is 15 minutes.

Every performance book out there always says you should monitor values over time in order to have them be statistically significant for spotting trends. That is what the PWH allows you to do. There are three types of monitor elements that are collected:

- *Watermark* indicates the highest (maximum) or lowest (minimum) value an element has reached since monitoring was started.
- *Gauge* is an indicator for the current value for an item. Gauge values can go up and down.
- *Counter* is a representation of information that is cumulative up until the sample is taken. The counter counts values that increase, such as the number of deadlocks. Counters are reset when you stop and restart an instance or database.

The PE Server handles each monitor element type differently when it summarizes them from history into PWH.

PWH aggregation methods

The PWH aggregation methods used for the three types of monitor elements are:

- For *watermarks*, the highest (maximum) or lowest (minimum) value is used.
- For *gauge* values, average values are calculated because they can increase and decrease depending on the database activity.
- For *counter* values, delta values are calculated over aggregation interval.

In our example, where we have aggregation in 15 minute intervals, we would see the following:

- For a watermark, the highest or lowest value reached, up to the end of the most recent 15 minute interval.
- For a gauge, the value would be averaged over the seven or eight detail history rows within the 15 minute interval.
- For a counter, the difference in the value between the last and the first history row of the seven or eight rows.

These will become clearer in later sections where we describe the reports in more detail.

There are two other groups of snapshot elements that are also stored in the performance database: *Database Manager configuration* (DBM CFG) and *Database configuration* (DB CFG). The aggregation of these values into the PWH is a bit different from the other elements, because these are not of the types watermark, gauge, or counter. These are informational elements, and they are moved into PWH only if any value has changed from one interval to the next. If your configuration parameters rarely change, you will have few rows in the PWH DBCFG and DBMCFG tables. The data is still available for use in PWH queries to correlate current monitor elements back to when the corresponding configuration parameter changed, to show effects of changing them. An example query showing how to do this is in 5.1.6, “Queries” on page 280.

Important: In the documentation and online help for PE, the monitor elements are often referred to generically as “counters.” This can be confusing because not all monitor elements are counters; some are gauges or watermarks. The element type (watermark, gauge, or counter) for any DB2 snapshot value can be found in the DB2 System Monitor Guide or the online InfoCenter. For example, the rows_read element is defined as a counter (see http://publib.boulder.ibm.com/infocenter/db2help/topic/com.ibm.db2.udb.doc/admin/r0001317.htm?)

Collection of SQL activity trace event monitor data

The other type of data stored in the Performance Warehouse is that which is collected by the statement event monitors you can run from within PE. You can run an SQL activity trace from either the Application Summary, where you trace one specific application, or as a process in the PWH, where you can trace all applications or some filtered subset of applications. The SQL activity trace can be scheduled in advance or run on demand, and is described in more detail in 5.1.4, “Performance Warehouse processes” on page 253.

As shown in Figure 5-1 on page 242, we see the SQL activity trace data is stored in the PWH in different tables than the history data. When you run a trace, the PE Server will create an event monitor, let it run for the time you indicated on the setup, then stop it and load the event trace data into the PWH tables. Depending on the type of request, it may also generate a report. In all cases, the data is stored in the warehouse, even if you do not report on it immediately. The data can be used in later queries and Rules of Thumb.

The use of SQL activity trace is described in more detail in 5.1.4, “Performance Warehouse processes” on page 253.

In Table 5-1 on page 247, we summarize the capabilities of the different PWH reporting features. There are some things you can do with one feature and not another. You can use the chart to determine how best PE can meet your

reporting requirements. If PE cannot meet your reporting requirements, you should contact your IBM representative to communicate your needs back to development.

Table 5-1 Performance Warehouse reporting capability summary

	Can you...?	Process	Rule of Thumb	Query	BPA
1	Execute via schedule	✓			
2	Execute on demand	✓	✓	✓	✓
3	Execute SQL Activity trace	✓			
4	Report on SQL activity trace data	✓	✓	✓	
5	Use PWH data	✓	✓	✓	✓
6	Use History data [a]			✓	
7	Copy/save report templates	✓	✓	✓	
8	Change/create report format		✓ (some)	✓	
9	Use calculated values		✓	✓	
10	Automatically save output	✓			✓
11	Manually save/browse output	✓		✓	

a. Not officially supported (see “Query a history table” on page 287.)

5.1.3 Using data from the Performance Warehouse (PWH)

Previously, we discussed what data is stored in the Performance Warehouse and how it gets there. In the next sections, we discuss the ways you can report on that data.

Opening the Performance Warehouse

To get to the Performance Warehouse features of PE, you can launch Performance Warehouse from the System Overview window (see Figure 5-3 on page 249). You can also launch PWH from the Tools menu. There are three Performance Warehouse selections shown on the overview window. Each one will launch the same Performance Warehouse main window, but each activates a different tabbed page, or workspace, on the window (see Figure 5-4 on page 250).

- Performance Warehouse - Report

  Shows only Process Groups

- Performance Warehouse - Analysis

  Shows only Rules of Thumb and Queries

- Performance Warehouse - Expert

  Shows Process Groups, Rules of Thumb, and Queries

For the examples in this book, we use the *Performance Warehouse - Expert* page because it provides access to all three components of Performance Warehouse reporting. You can determine which workspace page is most useful for you.

Note: The Performance Warehouse - Trace page is not used for multiplatform environments. You can safely ignore it.

All the examples in this section will use the Expert page.

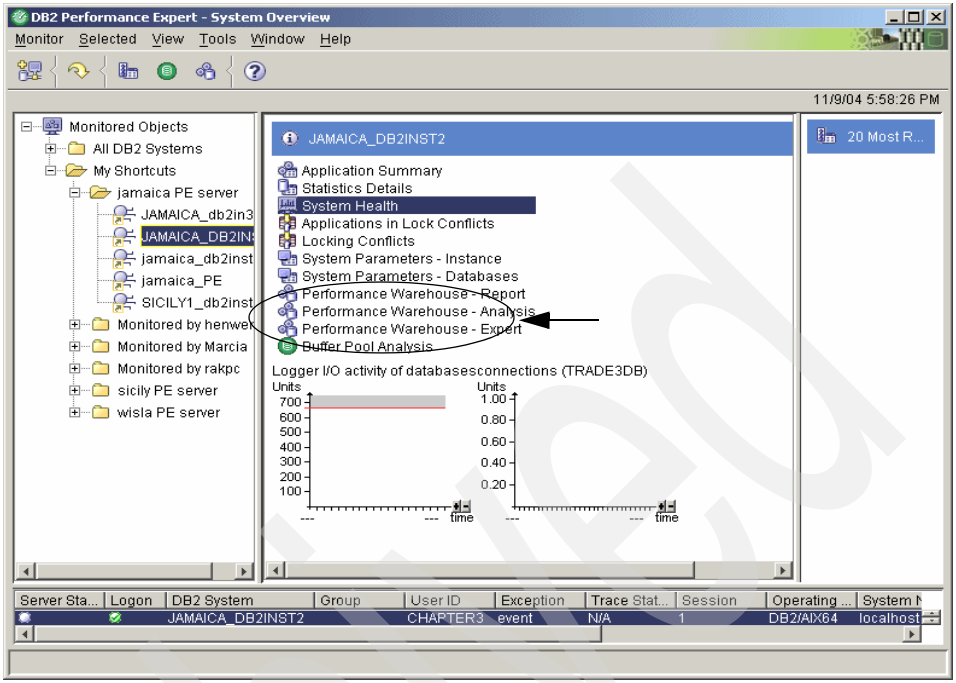


Figure 5-3 Launching Performance Warehouse from System Overview

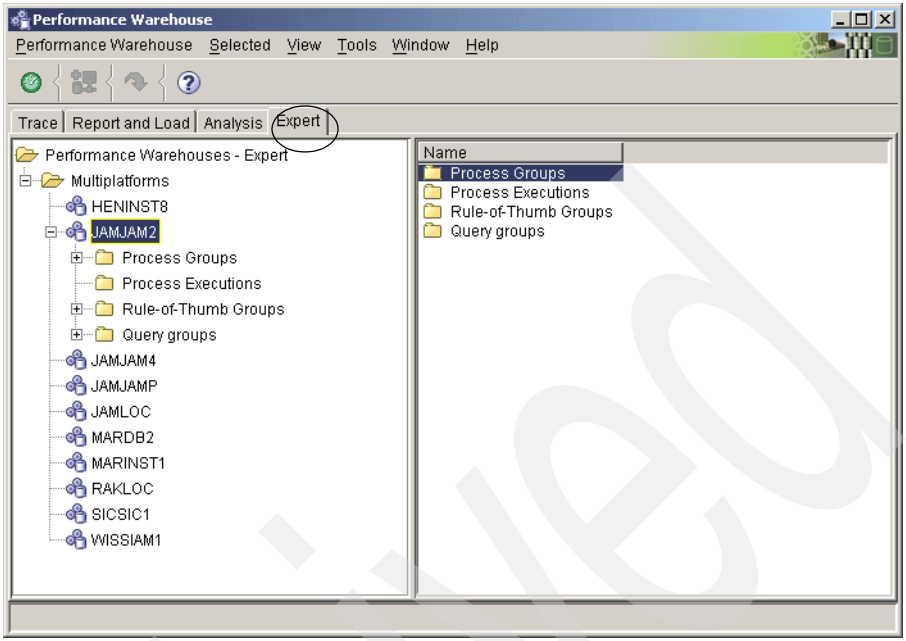


Figure 5-4 Performance Warehouse - Expert page

On the left side of the PWH Expert page, you see the list of Performance Warehouses available for reporting. You will notice these names are not the same as the names on the System Overview window. In Figure 5-5 on page 251, we show the relationship between the two values you set up in configuration, and what you see on the System Overview and PWH screen. In PWH, the connection alias is shown because you can work with the PWH independently from the PE Server. You just connect to the connection alias database. You can then do reports, queries, or Rules of Thumb.The key bit of information is that the Monitored Instance Alias appears on the Overview screen, and the DB2 Connection Alias appears on the PWH screens. For this reason, we suggest you devise some naming convention that will be meaningful to you. See 3.1, “Configuration” on page 46 for additional information about configuring your PE Client.

The reason these aliases are different is because you can also use the PWH even if your PE Server is not actively running. By cataloging the performance database directly on your local PC, you can still use PE Client to connect to the database using DB2; you are not required to go through the PE Server. Note, however, that while you can connect to the PWH when the PE Server is down, you cannot schedule or execute any processes.

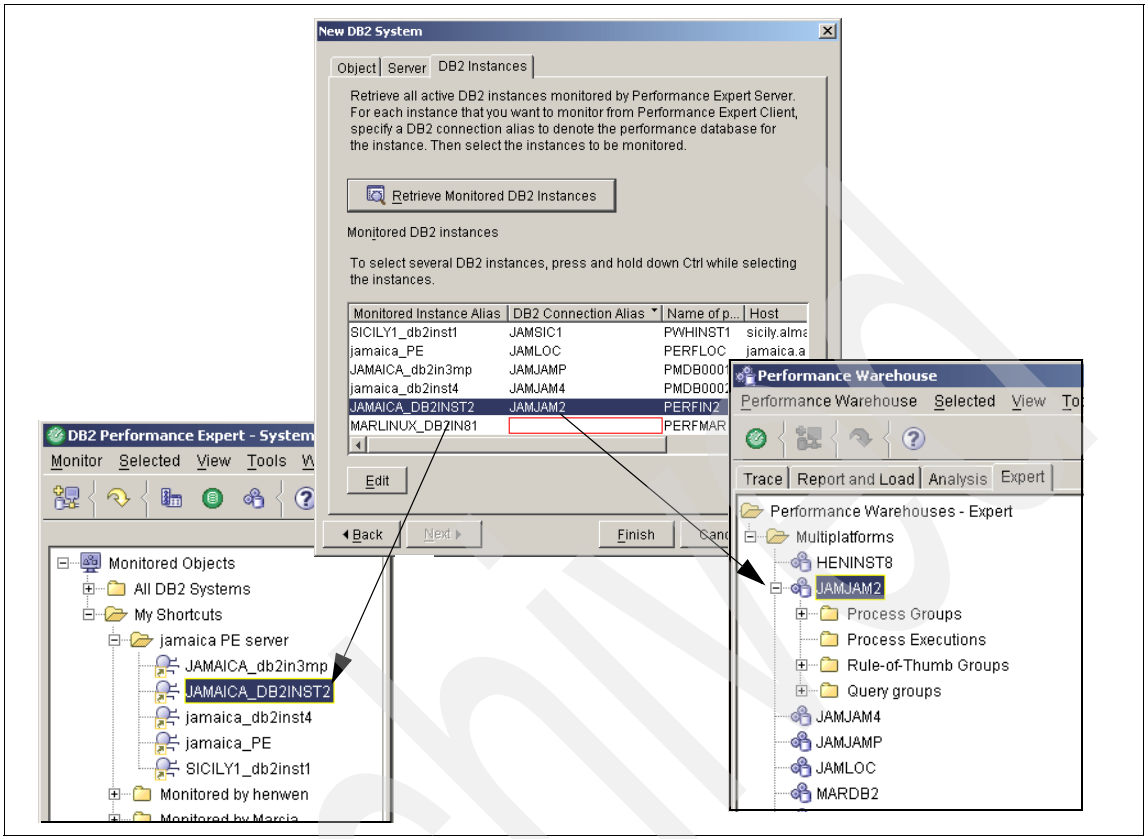


Figure 5-5 How PWH name correlates to client configuration setup

The PE documentation *Monitoring Performance from the Workstation,* SC18-7976 discusses how to add a Performance Warehouse, but this is more a task for z/OS users. In PE for multiplatforms, the Performance Warehouse data is in the same database as all the other performance data for the monitored instance, so the PWH is automatically already added and available on the PWH screen as soon as you configure a new monitored instance in the PE Client. There is no need to perform any additional steps in order to be able to use the Performance Warehouse.

If you delete the PWH from the PWH list, it will reappear the next time you open the window, if you are logged on to the PE Server.

Recommendation: You can safely ignore any steps for adding or deleting a Performance Warehouse.

Performance Warehouse components

In Figure 5-1 on page 242,we see four types of reporting that are available:

- PWH Processes (includes Reports)
- PWH Rules of Thumb
- PWH Queries
- Buffer Pool Analysis (BPA)

The PWH components are presented in the PE Client GUI in the Performance Warehouse windows, while the BPA gets its own interface, but they all use the PWH data as input to the reports.

In the Performance Warehouse screens, there are three components from which to choose:

- Process groups

  The PWH process groups are made up of processes, and each process can have up to three steps - Collect Report Data (CRD), Load, and Report. Not all steps are always performed when you execute a process. In this redbook we sometimes refer to the processes as “process reports”, though that is not an official term. We tried to separate the understanding of how data gets into the PWH from the understanding of how to report on the data. Sometimes you can do both within one process.

  Often referred to as the “canned reports”, the Process Groups provide a set of ready-made reports that you can run on demand or on a schedule. The reports can be in a summary or trace (detail) format, and include data at the database, tablespace, buffer pool, and table levels. You can choose any reporting time interval, so your reports can be as short-term or long-term as you like.

  The Process Groups component is also where you can run or schedule a SQL activity statement event monitor to collect very detailed application-related information over a specific time period. This data is captured and stored in the PWH for immediate or future reporting.

  All reports generated by executing a process are in HTML format, and are retained and available for later viewing via the Process Execution Log.

  You cannot modify the layout of the canned reports.

  We discuss this component in detail in 5.1.4, “Performance Warehouse processes” on page 253.

- Rules of Thumb

  A Rule of Thumb is a broad recommendation for performance tuning in your database. PE provides some ready-made Rules of Thumb, with recommendations for typical environments. The Rules of Thumb (RoT) apply a few simple rules and ratios to key performance indicators. You can also

create your own RoTs or copy one of the ready-made RoTs and modify it. This provides flexibility so you can adjust the RoT comparison values to suit your needs.

Rules of thumb are executed only on demand and output is viewable only online. You cannot save the output from a RoT report.

We discuss this component in detail in 5.1.5, “Rule of Thumb analysis” on page 272.

- Queries

  The query facility within Performance Warehouse offers the most flexibility in reporting on your performance data. PE provides a set of template queries that can be used *as-is*, or you can copy and modify them for your environment. You are also free to write any SQL query you like from scratch, using all the tables in the PWH as input.

  Queries are executed only on demand and output is not automatically saved in a log or on disk, but you can save it manually.

  We discuss this component in detail in 5.1.6, “Queries” on page 280.

A word about Buffer Pool Analysis

As we show in 5.1.2, “Life cycle of performance data” on page 241, the Buffer Pool Analysis (BPA) feature also uses the Performance Warehouse data as its data source for reporting. The other components of PWH can also report on buffer pool performance data, so we discuss those reports in those sections; however, the BPA component itself also provides other features and those are described in more detail in 5.2.1, “Generating Buffer Pool Analysis reports” on page 302.

5.1.4 Performance Warehouse processes

Performance Expert comes with a set of ready-made reports, in the form of public templates, that you can copy, configure, and execute. The reports are collected into sets called “Process Groups”, which are further grouped into Public and Private processes. The templates are of the following types:

- Buffer pool trace reports
  - One process step: Report.
  - Report includes all data elements related to buffer pool snapshot data.
  - Short report includes information about buffer pools and tablespaces.
  - Long report shows information about buffer pools and tablespaces and tables.
  - Discussed in “Buffer pool reports” on page 263.

- Database activity trace report
  - One process step: Report.
  - Report includes data elements related to database-level activity.
  - Summary style will show one row per report block for the time interval and database specified, summarized over the entire interval.
  - Trace style will show each row in PWH for time and database specified.
  - Discussed in “Database activity reports” on page 263.
- OS activity trace report
  - One process step: Report.
  - Report includes data elements related to OS data.
  - Summary style will show one row per report block for the time interval and system specified, summarized over the entire interval.
  - Trace style will show each row in PWH for time and system specified.
  - Discussed in “OS activity reports” on page 265
- SQL Activity processes
  - Two or three steps: Collect Report Data (CRD), Load, and Report.
  - Data is captured and loaded into PWH.
  - Report includes all data elements from the statement event monitor.
  - Discussed in “SQL Activity trace and reports” on page 265.

Note: The Buffer Pool and Database reports are called “Trace” reports, which can be misleading, as there is no tracing involved. The reports simply show all the data elements from the buffer pool, tablespace, and database tables in the Performance Warehouse for the filter criteria you specify. The “trace” name is another existing name from the PE for z/OS. The term “process” is also a z/OS existing term, which has little relevance in the multiplatform world of PE, and these PWH processes are in no way related to UNIX or Windows operating system processes.

General concepts about Process Groups

When you want to run a report-generating process, there are steps you need to follow for any of the report types. In this discussion, we use an example of creating a group for a workload test we were going to execute. The steps for using Process Groups are the same for any of the report types. The details of each report type is discussed later. This example is to introduce you to the navigation and terminology of using Process Groups.

You need to perform these steps:

1. Create at least one new process group.

   You must always have at least one group that you create yourself.

   You might want to create different groups for different databases or applications you want to monitor. The groups are just a handy way to put similar reports together, like using a folder. The group has no special behavior. The report output is not stored with the group folder.

   To create a process group, right-click the top-level Process Groups folder and select **Create** (see Figure 5-6).

   In this example, we create a new group called Workload Group.

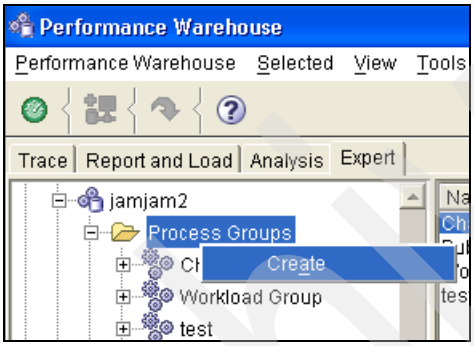


Figure 5-6 PWH - Create a new Process Group

2. Copy a public process template report to your process group.

   The template reports cannot be executed on their own, because they do not yet contain the proper filter criteria, such as which database to report on. The process templates in the process group “Public” cannot be edited; they can only be copied. A process can be edited by you (the owner) only if its status is not “active”.

   To copy a template, right-click it and select **Copy**. You are prompted for the Process group you want to copy it into.

The list of public templates is shown in Figure 5-7.

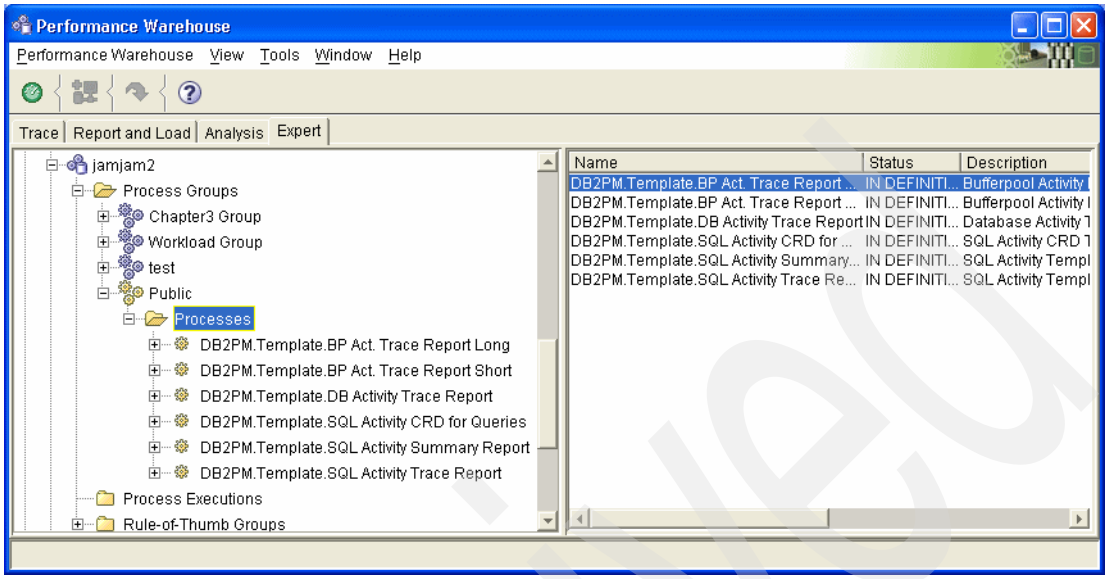


Figure 5-7 PWH - Public report templates

The public templates provided by PE are all named with DB2PM in the first part of the name. If another user identifies one of their reports as Public, it will show up in the public template list as well, but it will be prefixed with their user ID.

Note: You will not see your own public reports in the Public templates list; this is because you are the owner and you can edit the report. You only see templates created by someone other than you. If you are the only user of PE in your shop, there is no particular benefit in making anything public.

In this example, we copy the DB Activity Trace report template to our private group named Workload Group, and also give the copied template a new name of DB Activity Workload01. This is shown in Figure 5-8 on page 257. (We also made another copy of the same report template to Workload Group and named it DB Activity Workload02, for future use.)

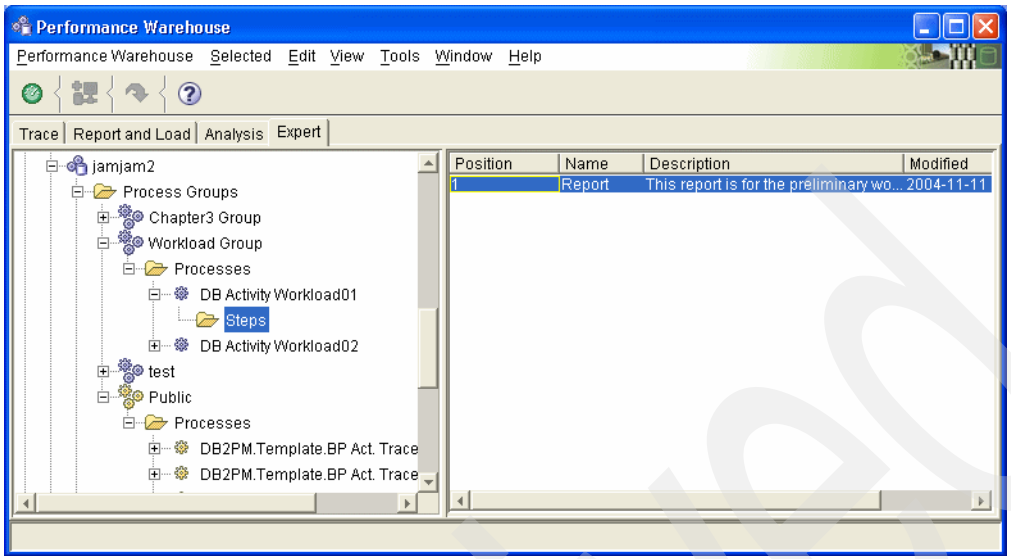


Figure 5-8 PWH - new copied process report

3. Modify the Options on the report.

   When you copy the template into your personal group, then you can edit the settings or options and configure it to run. The canned report templates do not contain anything to tell PE which database to report on. You have to fill in these report options before you run the report.

   We recommend that you first open the Report Step Options, then you can edit the report's Process Properties if necessary. To find the Report Step Options, expand the Process Group and expand the specific process by clicking the "+" symbol in the tree. There will always be a Steps folder within the process. When you highlight the Steps folder, the pane on the right side of the screen will identify which steps are part of the process.

   In our example, we chose a process template that contained a single step: Report. These types of processes are simply predefined queries with predefined output formatting that PE runs against the PWH data tables. Before you can execute the process, you need to tell it some options to use and what time period to report on. These criteria vary slightly by the type of report, so we discuss those in more detail in the report sections to follow.

   To open the Report Step Properties, double-click the **Report** step in the RIGHT-SIDE pane. The General page of the properties window is where you can add a description of the purpose of the report. We do not describe this here. The Options page is where you set the report criteria.

In our example, shown in Figure 5-9, we are testing a new application that will run on the TRADE3DB database. We have set up some long-running tests and we want PE to capture data while the tests are running, and then we will report on it later. For this example, we want to report on the activity in the TRADE3DB database on November 6, 2004 between noon and 8:30pm. We want the Trace type of report because we want to see the data for every interval during the reporting time period. If we chose the Summary type, we would only see a single row of output, with data summarized over the reporting time period. This would not be useful for our experiment.

If you are working in a partitioned (DPF) environment (which we were not, in this example), you should know the Scope and Partition fields are also usable for you. The Scope option can be Global or Detail, and determines how the data is presented in the report - either summarized across all partitions (Global), or summarized within each partition (Detail). You can further refine the amount of data in the report by specifying the partition(s) you are interested in. In our example, our system is not DPF, so these fields are not enabled.

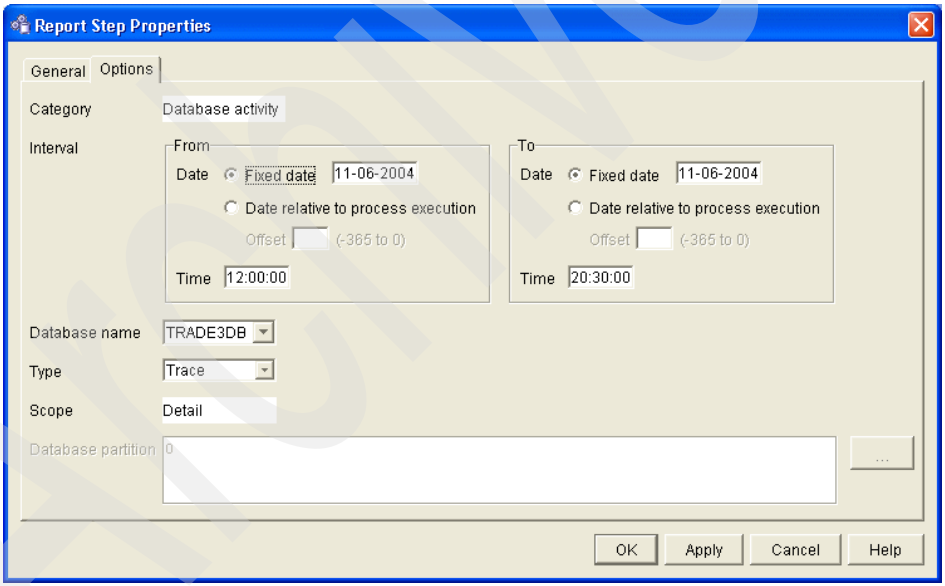


Figure 5-9 PWH - Report Step Properties window

Click **OK** to save the settings.

We could now edit the process properties, but it is not really necessary unless we want to change an existing schedule or set the process back to 'in definition' status to make other changes.

4. Execute the report process.

   If you are using Performance Expert V2.2 to run this report, right-click **Process** and then click **Execute**.

   The process starts running immediately, and you can watch its progress on the Process Execution screen.

   If you schedule a report to run at a future time, or if you run it on a recurring schedule, you do not use the Execute method. The execution is handled automatically by the PE Server when you set up the schedule. If you want to run a process on a certain schedule, you first set the schedule, then you mark the process as “Active” in the Properties of the Process.

   If you are using Performance Expert V2.2.0.1, to run this report, right-click **Process** and then click **Execute**. A scheduler window pops up that allows you to specify whether the process should be executed once immediately, once at a later time, or repeatedly.

   In this example, we choose to execute it once immediately and we can watch its progress on the Process Execution window. An example of using a process in a schedule is described in “Customizing and scheduling a process” on page 314, and shown in Figure 5-47 on page 316.

5. View the report output.

   Select **Process Executions** in the PWH Process tree view to see the log of all reports previously run, or any that are currently running. In our example, the report completed quickly, so it was already finished by the time we opened the view, shown in Figure 5-10.

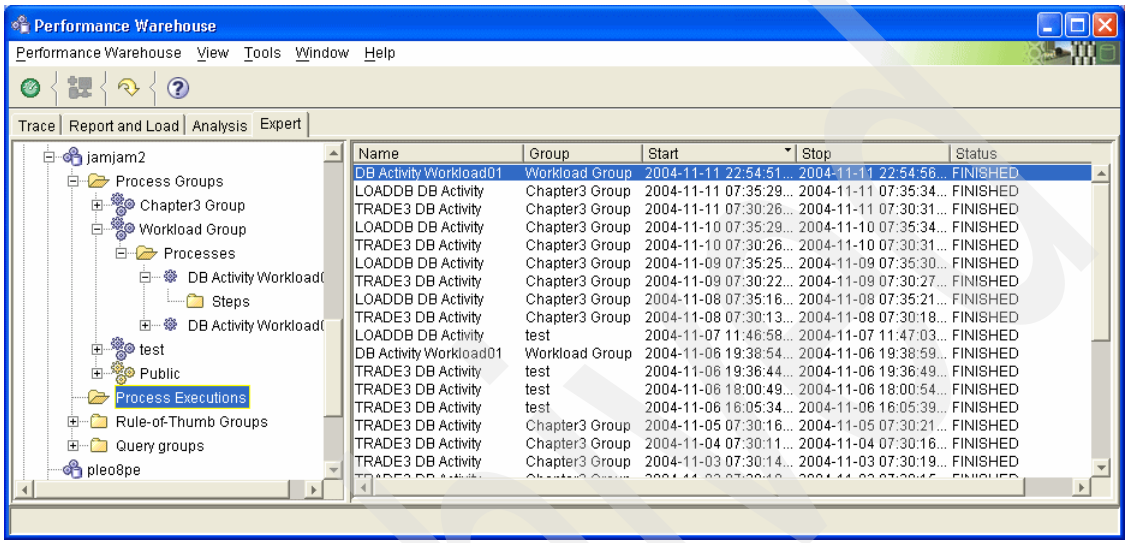


Figure 5-10 PWH - Process Execution log

To see the output, double-click the report in the log to open the Process Execution Details window. You will need to expand several folders, but the report is there (see Figure 5-11 on page 261). Click the **Open** button to launch a browser - all the report output is generated in HTML format. A browser output is shown in Figure 5-12 on page 262.

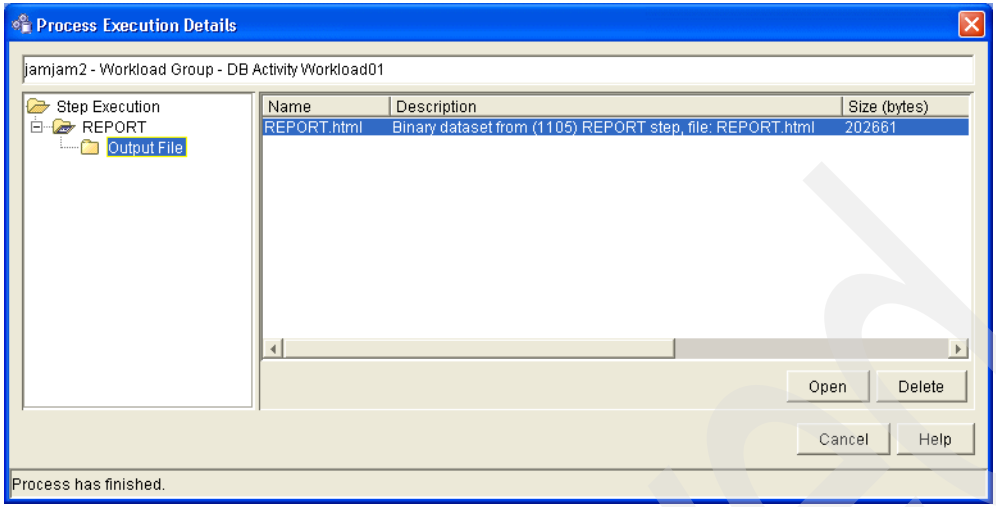


Figure 5-11 PWH - Process Execution Details

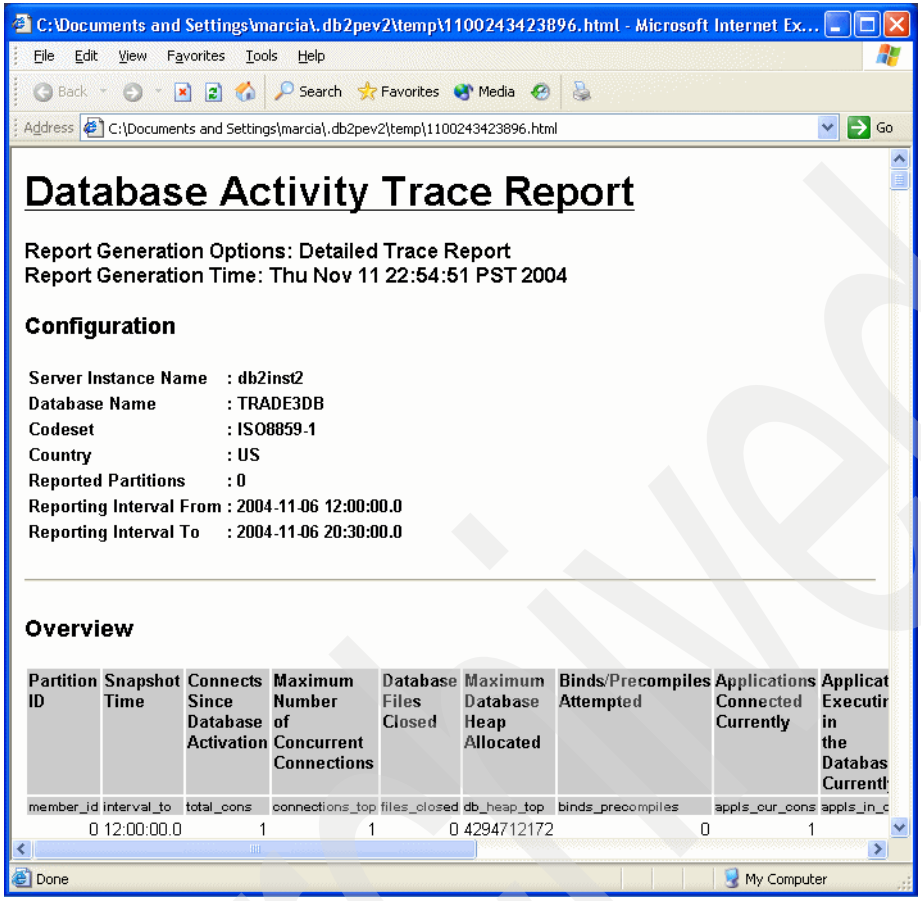


Figure 5-12 PWH - View report output in browser

Now we can view the output and see what was going on in our test database during each PWH aggregation interval. Once the report has been generated, you can always go back and view the output again later by opening it from the Process Execution log. The output is saved in the PWH tables; you do not have to re-run the report to see the output again.

Notice in the HTML output that the process report options as entered in Figure 5-9 on page 258 are shown at the top of the report, also included is the date and time you ran the report.

Buffer pool reports

The buffer pool reports are similar in layout and format to the Database activity reports discussed in the next section, with each section of information in a “block” with data for that particular group of elements. The buffer pool report blocks are:

- Buffer pool
- Tablespace
- Table

The buffer pool report provides additional filters where you can limit the report to only certain buffer pools, tablespace, and tables. We do not go into detail about these here, other than to give you a hint on the filters for TABLESPACE_TYPE and TABLE_TYPE. These values are codes, not text, so you need to know the right code to get the right output.

- TABLESPACE_TYPE
  - 1 = SMS.
  - 2 = DMS.
- TABLE_TYPE
  - 1 = User table.
  - 4 = System catalog table.
  - Table types 2, 3, and 5 are not used in PWH.

Buffer pool analysis is a broad topic, so we have the most discussion about all things buffer pool-related in 5.2.1, “Generating Buffer Pool Analysis reports” on page 302.

Database activity reports

Database activity reports include many of the monitor elements that are displayed in the Statistics Details screens, but they are collected and aggregated over time so you can see the overall, long-term behavior of your system. These types of reports are extremely valuable for trend analysis and for future capacity planning.

Database activity reports can also be useful for profiling a new or changed application in a test environment. For example, you can collect performance data over a series of test cycle runs, keep the data for each run, and compare them later. You can compare the test results with later production results as a way to validate your projections and improve for the future.

If you use a strict test process where you can reset an environment back to a “zero state” before each run, the PWH data can be very valuable in proving reproducible results for a test. We do not discuss performance test modeling here, but we suggest that PE can be a useful tool in your test tool arsenal.

The DB activity reports are laid out in “report blocks” with each section reporting some different grouping of monitor elements. Not surprisingly, these report blocks map back to data elements shown in the Statistics Details - Databases screens, and also directly to the PWH Query column-assist sections. If you study the DB online statistics screens, you will see the same elements as on the activity reports, and vice versa. As you work more with PE and the PWH, you will become more familiar with what monitor counters are most interesting for your environment.

The DB activity report has nineteen report blocks:

- Configuration
- Overview
- SQL statistics
- Row statistics
- Sorts
- Hash joins
- Internal DB Statistics / Counts - Logging
- Internal DB Statistics / Counts - Locking
- Internal DB Statistics / Counts - Catalog Cache
- Internal DB Statistics / Counts - Package Cache
- Internal DB Statistics / Counts - Shared Workspace
- Internal DB Statistics / Counts - Private Workspace
- Bufferpools / IO Statistics - Non-bufferred I/O Activity
- Bufferpools / IO Statistics - Data section
- Bufferpools / IO Statistics - Index section
- Bufferpools / IO Statistics - Times
- Bufferpools / IO Statistics - Extended Storage Section
- Bufferpools / IO Statistics - Page Cleaner Section
- Applications

See “Relationship between Process Groups, RoT, and Queries” on page 293 for some more discussion on the relationship and mapping between the canned DB activity reports and the canned queries.

When you copy the DB activity template, you can then set up the appropriate criteria for your report. For this report, you can specify the reporting date and time range, as for any report, and also the database name, and, if you are using DPF, you can specify whether you want to have a detail or global report. If you choose details, then you can specify a partition. If you specify global, then the data is aggregated over all partitions in the report. There is no other filtering

available. That is, you cannot run the report only for five of the 19 report blocks. If you want that sort of information, you should use PWH Queries.

The DB activity reports would be useful as something you might schedule to run each day or week, for the previous period's activity. The reports could run during the overnight time, and be ready for you to look at in the morning. The detail inside the reports may make it difficult to spot long-term trends, but you could still use the data for assessing the previous day's activity. If you saw something that looked odd, you might then run a Rule of Thumb, which has analysis built in.

OS activity reports

The OS activity reports include OS data collected and aggregated over time for CPU identification and utilization, available and consumed memory, and file systems. These are many of the elements that are displayed in the Operating System Information and Operating System Status windows. Process information is not available in the PWH tables.

The data is arranged in blocks similar to the buffer pool or database activity report.

An OS activity report can either be created for a single system or for all systems that belong to the DB2 instance you are monitoring.

SQL Activity trace and reports

The third set of Performance Warehouse Process groups is the SQL Activity processes. This is where you can run a DB2 statement event monitor to capture detailed information about applications running on your database. If you look again at Figure 5-1 on page 242, you will see that this process is how the data gets into the PWH. The event monitor runs, then PE stops it and transfers the data into the PWH event data tables. These tables are then used for reporting from PWH.

You can also run a SQL Activity trace from the Application Summary window. This will trace one specific application, and can load the data into the PWH in the same way as the process group described here, and is capable of being reported on in the same way too, so we do not discuss it further. An example of how to run this trace is in 6.2, "Problem reported" on page 365.

To initiate a trace in the PWH, you first have to copy the template to your own process group, using the same method as described in “General concepts about Process Groups” on page 254. The setup parameters are also similar - you need to open the Process steps, then configure the values as requested by the steps. The SQL Activity processes always have at least two steps:

- CRD (Collect Report Data)
- Load

If you run the SQL Activity Report Summary or Trace process, there is a third step:

- Report

  This step prepares a report based on data just collected in previous step.

Note: For reporting on DPF partitions, you should use the “SQL Activity CRD for Queries” process, then report on it using the PWH Queries. The canned SQL Activity Report Trace process does not handle DPF.

These steps are described in more detail in the next section.

CRD step

During the CRD step, the PE Server creates and activates the event monitor on the monitored database, for the duration you specified in the setup. When you configure the CRD step, you tell it which database to operate on, how long you want to collect data, and whether you want any additional filtering. The Options page is shown in Figure 5-13 on page 267. In this example, we show how to set up a trace to run for one minute, on the TRADE3DB database. We left the event monitor name and file size as the default values. Please note that If you choose the process template SQL Activity CRD for Queries, you can collect data for more than one partition at the same time. For the other SQL Activity processes, it is only possible to choose one partition.

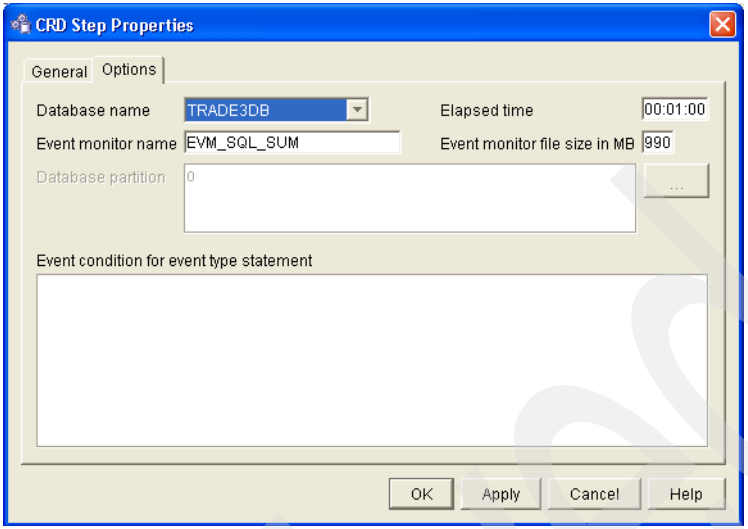


Figure 5-13 PWH - SQL Activity trace CRD options

By default, when you run the SQL activity CRD, you will collect data for ALL applications that are connected to the database. This may generate a very large amount of data, so you might want to limit it. This is done by using the “event condition” box. Unfortunately, there is no dialog to fill this in, you have to compose it manually, but it is not too difficult. The online help describes the syntax for this option, and is shown in Figure 5-14. This is the syntax as described in the DB2 help for the CREATE EVENT MONITOR command.

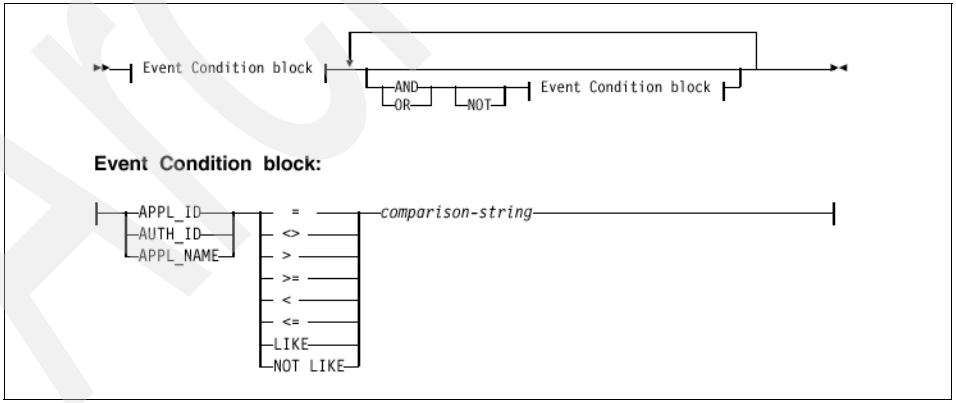


Figure 5-14 PWH - SQL Activity - event condition

Using this filter, you can limit the trace to operate only on a particular application ID (which is the same as running the trace from Application Summary screen!), or a particular user, or some combination of these.

The syntax looks like a standard SQL WHERE clause, but you should *not* include the word WHERE (see Figure 5-15). The WHERE statement is already included by the PE Server when it creates the event monitor. If you include it, you will get an error when you execute it.

This example would trace, for one minute, all applications connected to SAMPLE with user CHAPTER3. A connection made with a different authorization would not be traced. If we ran this trace with applications active, as shown in Figure 5-16 on page 269, only applications 45 and 326 would be traced, because they are the only ones connected with user CHAPTER3. Using this filter is an excellent way to reduce the amount of data being captured and loaded; the DB2 statement event monitor can generate a very large volume of data, so be careful and capture only what you really think you need.

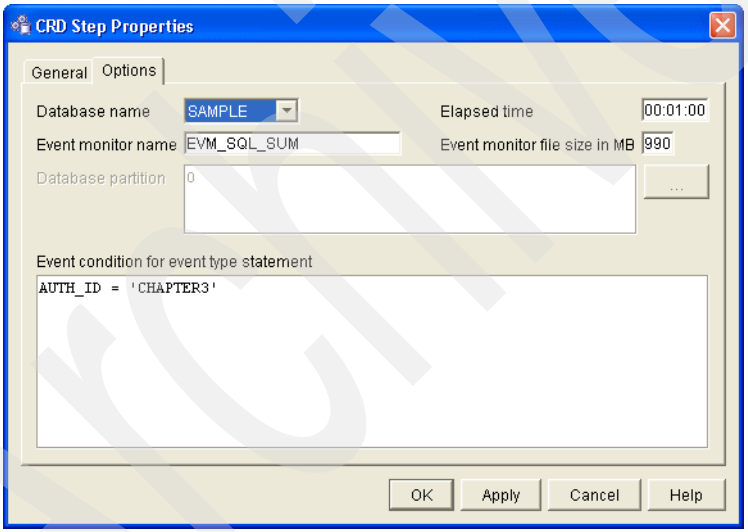


Figure 5-15 PWH - SQL event condition example

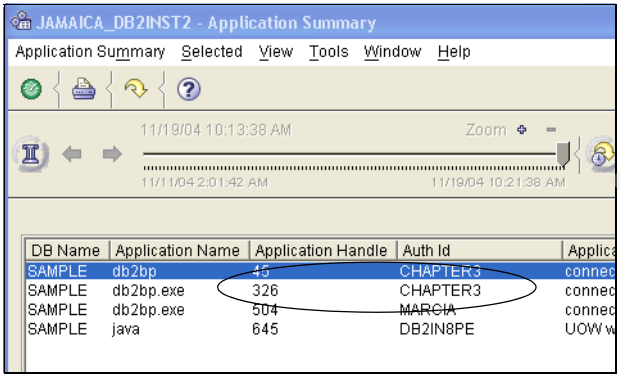


Figure 5-16 PWH - SQL event condition - which apps will be traced

Load step

During the load step, the event monitor data created during the CRD step is read and loaded into the PWH tables. These tables all have names that start with EVM_ and can be referenced in the Rules of Thumb and PWH Queries.

You can also delay the load step to run at some later time than the CRD step. For example, you could collect SQL activity data for 15 minutes in the morning, then set the load step to run some hours later. This is a feature more useful for z/OS systems, but you can also do it on multiplatform, so we mention it here. For most users, allowing the load step to run immediately after the CRD step is most practical.

One important piece of information from the Load step is the time stamp and loadlog_ID of when the load happened. This can be referenced later in the Rules of Thumb and Queries, to limit the amount of data being evaluated.

Appendix A, “Overview of the Performance Warehouse database“, in *Monitoring Performance from the Workstation,* SC18-7976 has a description of how the PWH EVM tables are laid out. The key thing to know is the Load step always generates a Load log ID number (LL_ID) and this is how you can filter which data is in the report. (You can also use the time stamp.) This is described in more detail in the 5.1.6, “Queries” on page 280, and also applies to Rules of Thumb, although it is not described there.

Note: If at any time you are confused about the PWH, you can look at the public queries, which can provide some insight into how the data is retrieved from the PWH tables.

Report step

There are two process templates that will create a report after collecting and loading the SQL activity trace data. That is what happens in the Report step. When complete, you can view the report in HTML format by opening it from the Process Execution log, as described in “General concepts about Process Groups” on page 254.

The SQL Activity Trace Report is quite large, so we show just a piece of it in Figure 5-17 on page 271. This report used the filtering condition described in Figure 5-15 on page 268, where we limited the trace to only applications from user “CHAPTER3.”

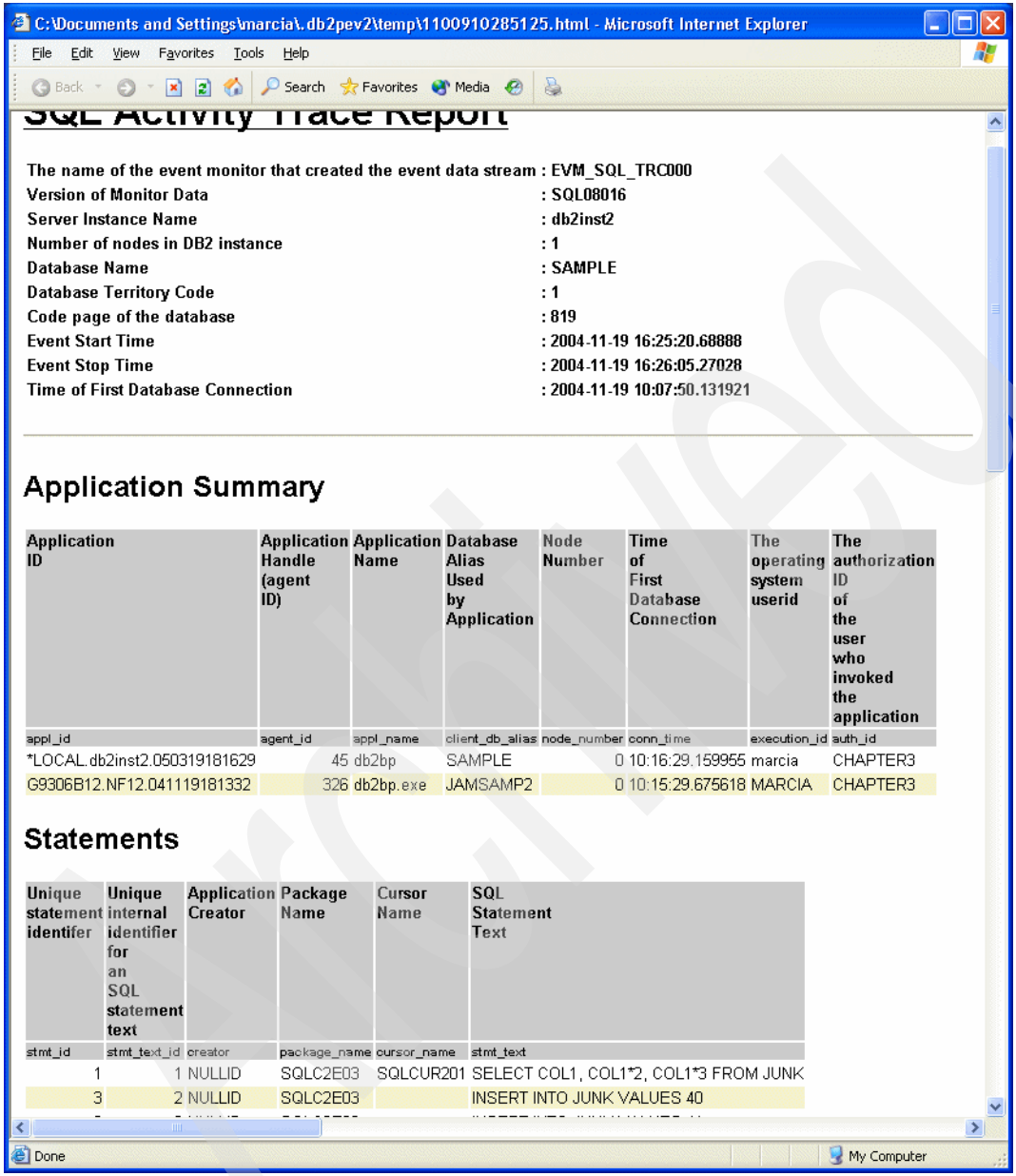


Figure 5-17 PWH - SQL Activity Trace output report

5.1.5 Rule of Thumb analysis

The Rules of Thumb function provides a powerful way to analyze long-term data and to present recommendations to the user on how to tune his system. This function is part of the Performance Warehouse. Once long-term data has been stored into the performance warehouse, users can then use the predefined Rule of Thumb for analysis, based on what has been defined within the SQL queries for violation warnings or problem thresholds.

Rules of Thumb apply a few simple rules and ratios to key performance indicators. Sample Rules of Thumb have been provided for you. The predefined Rule of Thumb cluster definitions for multiplatform and workgroups consist of the following:

- Buffer pool

  The buffer-pool cluster contains Rules of Thumb that evaluate counters related to buffer pools.

- Database

  The database cluster contains Rules of Thumb that evaluate counters related to databases.

- SQL Activity

  The SQL activity cluster contains Rules of Thumb that evaluate simple counters. These counters are compared with values that are a means to reduce the size of the result of the underlying query.

- Tablespace

  The tablespace cluster contains Rules of Thumb that evaluate counters related to tablespaces.

Predefined and adaptable Rules of Thumb (RoT)

Rules of Thumb are created within the Performance Warehouse GUI of the PE Client. Similar to the Process definition in the Performance Warehouse, Rules of Thumb are grouped together into Rule of Thumb clusters, which can be grouped together in Rule of Thumb Groups (see Figure 5-18 on page 273). To create, delete, or modify a Rule of Thumb group, Rule of Thumb cluster, or Rule of Thumb, the corresponding folder has to be selected in the left tree and the requested operation chosen from the shown context menu.

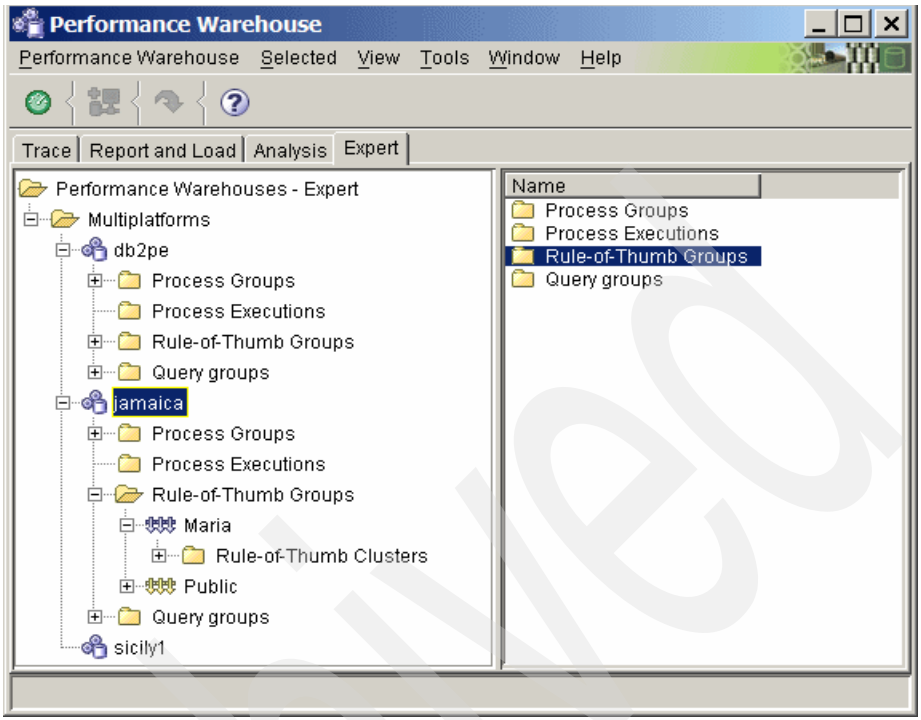


Figure 5-18 Performance Warehouse RoT

The Rule of Thumb cluster defines the type of data in the Performance Warehouse (DB activity, SQL activity, Tablespace, or Buffer Pool) against which the Rule of Thumbs in the cluster should be executed. The type of a Rule of Thumb Cluster is undefined till the first Rule of Thumb in the cluster is created. The first Rule of Thumb defines its type; this type will become the type of the Rule of Thumb cluster and therefore the type for all Rules of Thumb within this Rule of Thumb cluster. Depending on this type, each Rule of Thumb cluster can limit the number of checked objects in the performance warehouse through filter expressions.

If the first Rule of Thumb is defined, the definition pane of the Rule of Thumb cluster properties can be opened and the available DB2 PE counters combined through an SQL-like filter expression displayed. You can add columns or specify the filler expression (Figure 5-19). The Rule of thumbs in the Rule of Thumb cluster will then only be executed against the objects, which pass through this filter.

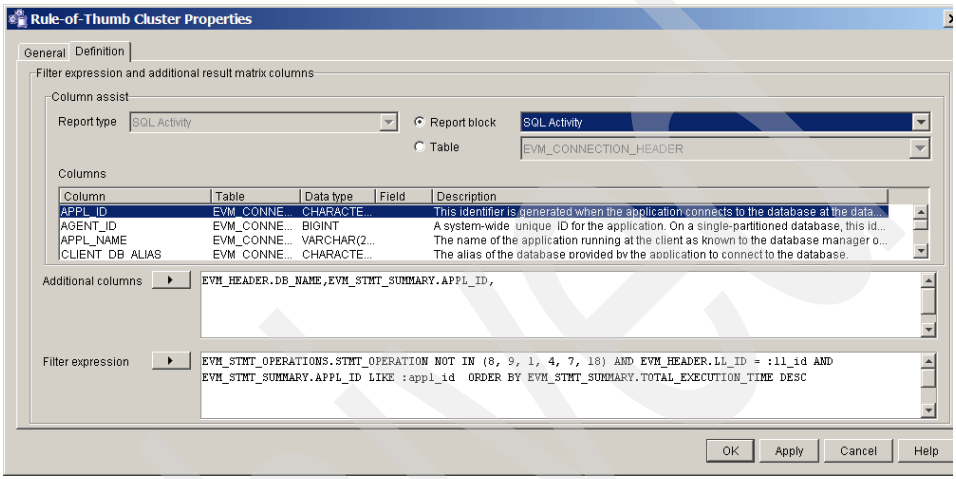


Figure 5-19 RoT cluster filter options for SQL activity RoTs

Similar to the Rule of Thumb cluster filter definition, the Rule of Thumb property panel supports the user in defining a Rule of Thumb expression. The definition pane shows the available DB2 Performance Expert counters and allows the user to create an expression and specify the warning and problem threshold. The expression is then ready to be evaluated and a warning or problem raised if the value violates the specific level. In addition to the thresholds, the user can specify a recommendation in case of a violation (see Figure 5-20 on page 275).

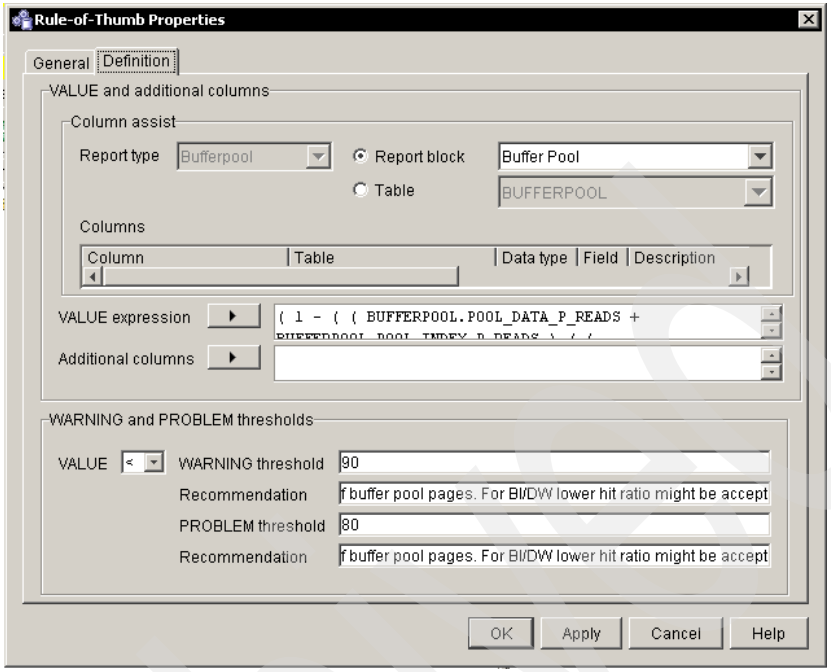


Figure 5-20 Specify RoT violation recommendation

To analyze the data in the performance warehouse, the user has to choose the “analyze” item from the context menu of the RoT cluster in the Options pane of the notebook (see Figure 5-21). Choosing the filter option in the tree lets the user specify an SQL WHERE clause to filter out the not requested data in the analysis. Choosing the sort option lets the user specify the order sequence in which the found data records should be displayed. Any changes to the filter and sort option will be immediately reflected in the displayed result matrix. By default, all found data records will be shown ordered by time stamp of the first column in the result matrix.

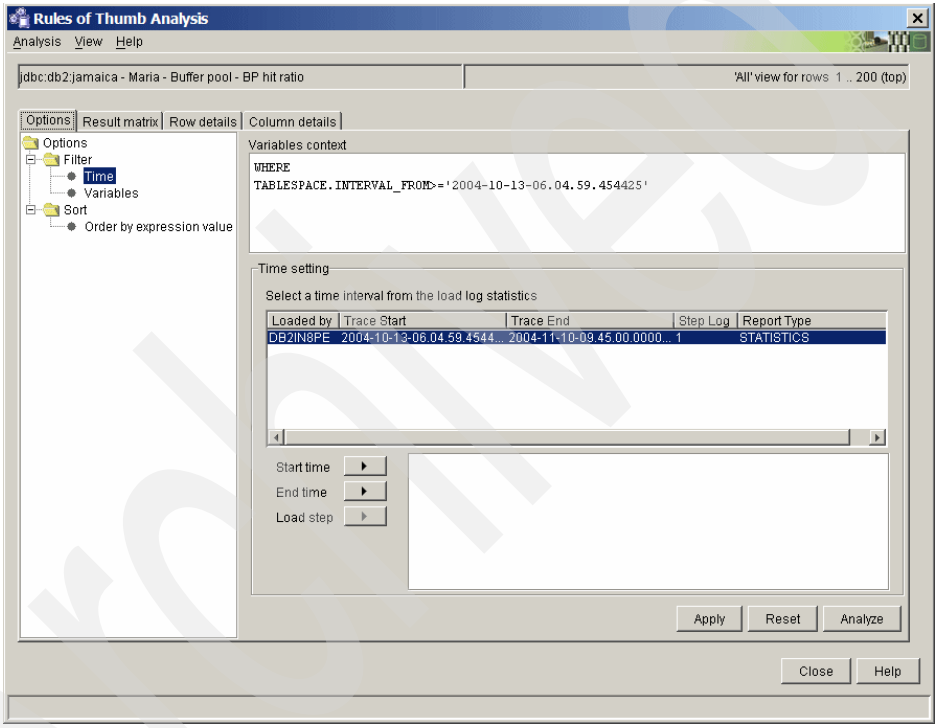


Figure 5-21 RoT Analysis options

Once the analysis options are specified, press **Analyze.** The results of the RoT cluster execution will then be shown in the Rules of Thumb Analysis result matrix (see Figure 5-22 on page 277). The result matrix displays the evaluation of the arithmetic expression (that represents the RoT) over time, indicating a problem or warning situation.

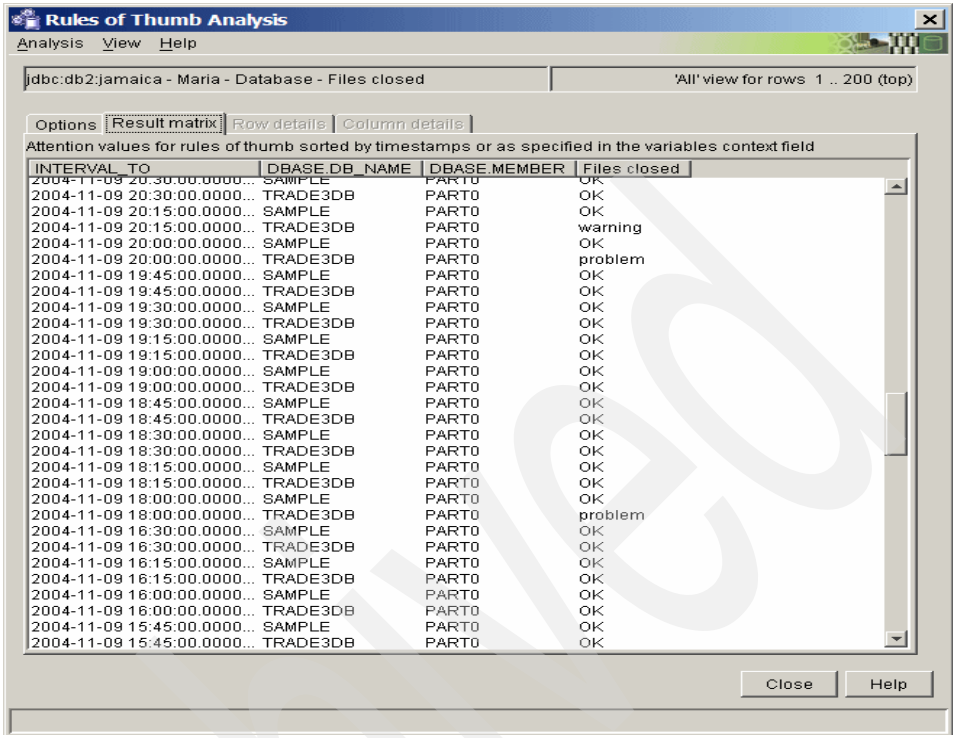


Figure 5-22 RoT Analysis result matrix

If a row or column in this matrix is select, the details of the selected item can be viewed in the notebook pane Row details and Column details. The Row detail pane shows the results of all Rule of Thumbs for the selected point in time (see Figure 5-23).

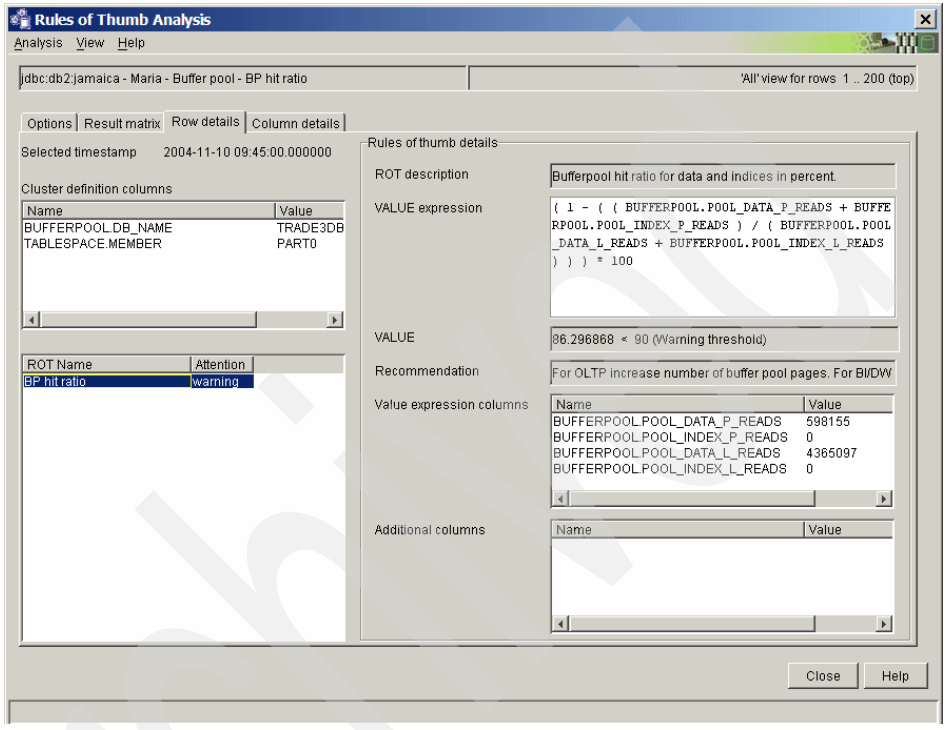


Figure 5-23 RoT Analysis Row Detail

If a Rule of Thumb itself is selected, details about this Rule of Thumb for all data records (over time) are displayed (for example, values of the expression, recommendation, and so on; see Figure 5-24 on page 279).

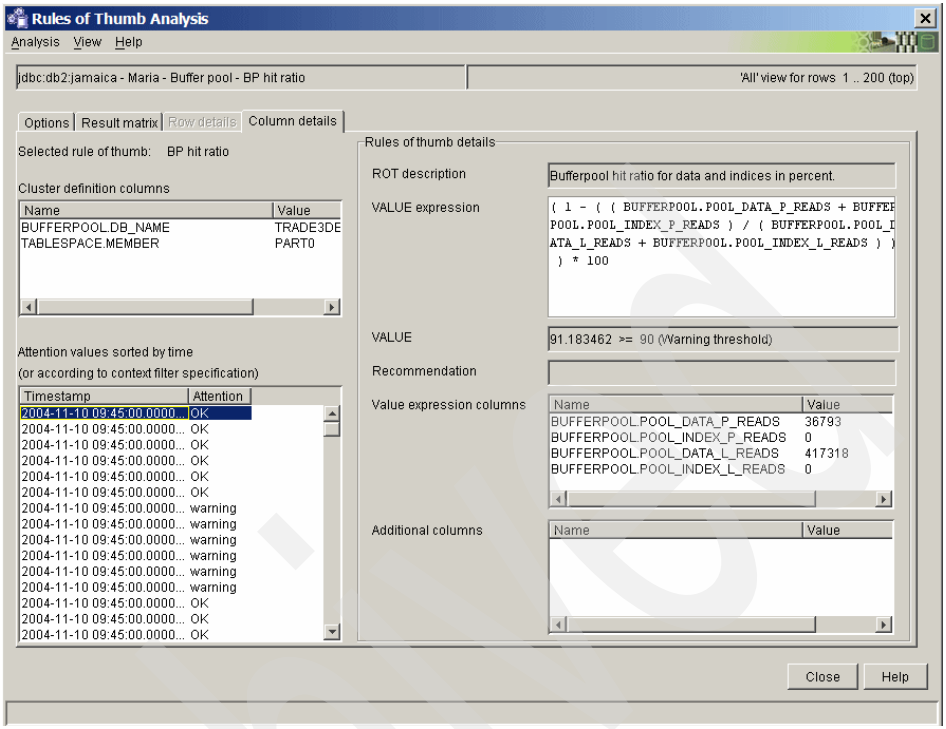


Figure 5-24 RoT Analysis Column Detail

The predefined Rules of Thumb are shipped with the product and can be found under the Public Folder in the Rule of Thumb Group folder in the tree (see Figure 5-25).

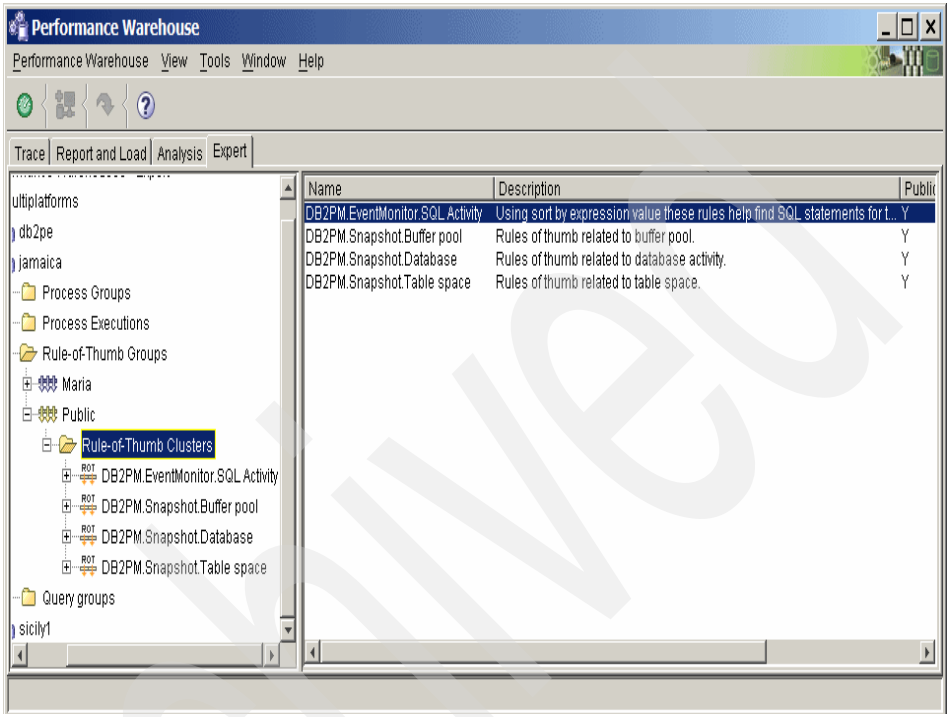


Figure 5-25 RoT predefined Public folder

For more details on Rules of Thumb, see the online Help in the following windows:

- Definition page of the Rules of Thumb Cluster Properties window
- Definition page of the Rule of Thumb Properties window
- Rule of Thumb Analysis window.

Also see “Buffer-pool cluster definitions” on page 458.

5.1.6 Queries

The third major component of the PWH is the Query feature. Queries are standard SQL select statements that operate against the PWH tables. As we describe in earlier sections, and show in Figure 5-1 on page 242, the PWH tables contain both aggregated history snapshot data and event monitor data captured

in the SQL CRD step. In this section, we provide some additional information about how to use queries, some tips, and several sample queries.

With queries, you can either use predefined queries or create your own, using standard SQL. Performance Expert provides a column-assist feature in the query builder, so you can pick tables and columns from a list. This is handy if you are not familiar with all the PE tables, but you do not have to use the column assist. If you are comfortable with SQL you can just write the query freehand.

A really nice feature of the PE queries is the ability to insert a variable into the query, which you can later fill in at run time. We show examples of using variables in this section. Variables can also be used in the Rules of Thumb as well.

The predefined queries (see Figure 5-26 on page 282) can show you a lot of information; some of them map roughly to the predefined reports and Rules of Thumb. You can also use the predefined queries to gain a richer understanding of the PE data model and techniques for retrieving data. The predefined queries can be classified into the following groups:

- Problem type queries

  These queries only return data containing certain expressions, for example, buffer pool hit ratio, that breach a predefined warning or problem threshold. These queries use the same expressions and thresholds as the Rules of Thumb.

- Report type queries

  These queries return parts of the data contained in the PWH, as is similar to the processes that create reports. You can use these queries as alternative to the processes or adapt them to customize your own reports.

- Configuration queries

  These queries correlate the behavior of particular monitor elements, for example, sort-related elements, with a corresponding database or database manager configuration value, for example, SORTHEAP. These queries are very valuable in evaluating trends and looking for the effect of changing a configuration value. There are also queries available that report on database or database manager configuration changes over time.

  Important: For DB and DBM CFG values, the PWH only inserts a row if a value has changed. Each of these tables has the config value as a column in the table. The INTERVAL_TO column in the DBCFG and DBMCFG tables does not contain the same time as the INTERVAL_TO columns in the other PWH tables. That is why the *left outer join* joining method is used in the configuration queries. You should always include this in a query comparing config values to statistics values.

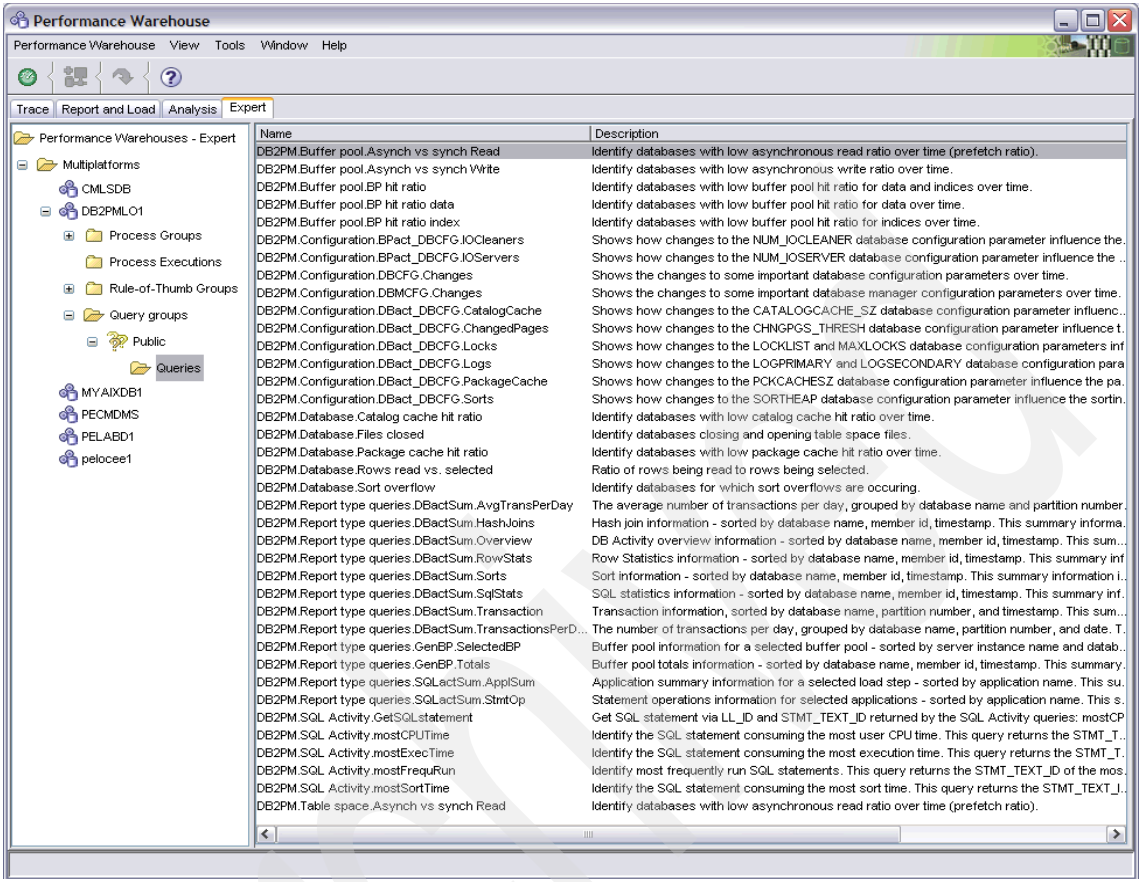


Figure 5-26 List of predefined PWH Queries

There are some differences between the queries for SQL activity and those for snapshot history data. When you execute a query, you are presented with a dialog where you can provide run-time variable values or select time ranges. There is often much confusion about the time ranges; we will try to clear that up here. Refer to Figure 5-27 on page 284 in the following discussion about times.

For history snapshot data queries (and Rules of Thumb), you can safely ignore the Time page, as shown in Figure 5-27 on page 284. The time values used here apply only to data loaded into PWH from SQL activity CRD process step. There *is* a time entry for the history data - it is always shown as Report Type = Statistics, and the StepLog (LL_ID) is always 1. However, this is not really useful in queries, and we show other examples of limiting query results by time in the next sections.

For queries that operate on the SQL CRD data, the time interval can be important, but for most of your queries, it is likely that you only want to query *one* time interval, and that interval can be referenced by the LL_ID variable. In Figure 5-27 on page 284, we see the column labeled Step Log is the number used in the queries as LL_ID. Each time you execute a CRD and Load step, the LL_ID is entered and gets an ID, with report type = SQL Activity. If you can determine which SQL Activity step log is the one you want, all you need to do is enter that LL_ID value in the variable page (also shown in Figure 5-27 on page 284) and you can ignore the Start Time, End Time, and Load Step areas on the Time page. We strongly recommend this approach, as it will save you pain in the long run.

When would you want to use the start/end time/load step values? These could be useful if you ran a series of short traces, and wanted to report on them over all intervals, not just one. Another reason might be if your CRD was for a very long period of time, but you only wanted to report on a subset of that time. In our estimation, it would still be simpler to modify the query and use date functions on the start/end time stamp columns than it would to use this screen. That is only our opinion; you may find other good reasons to need to use these values on the Time page.

It is not necessary to specify anything in the Start/Stop/Load values on the Time page, which is why we show it as empty.

Note: The Time page is also used in the Rules of Thumb, and the same recommendations apply there.

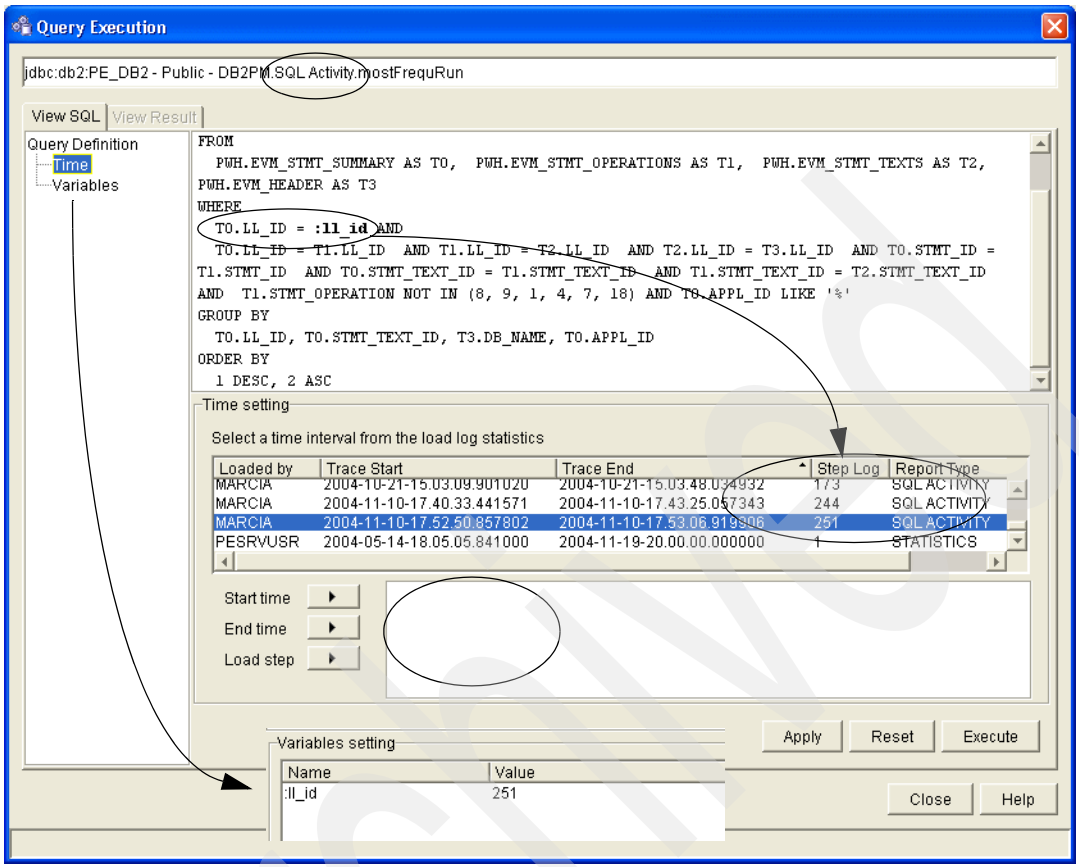


Figure 5-27 PWH Query - Time versus Variables revealed

Query hints and tips

- All interval_to and interval_from fields (history snapshot data only) are data type TIMESTAMP so you can easily use the date and time functions, such as DATE(), HOUR(), and so on.
- SQL Activity data also uses TIMESTAMP columns on START_TIME and STOP_TIME, but these apply to a specific statement, not the range of all trace activity. To query all activity within the CRD, use the LL_ID variable.
- Use the variables freely - it saves you from having to make new queries for each variation. Variables are any string prefixed with a colon, for example, `:db_name`. A variable name cannot include blanks or dashes. We show many examples of variables in this section.
- When using variables, build them with the accommodation of data type already built in, so all the variable entries are consistent. For example, if the

variable is a character field, set it up as `':db_name'` with the quotes already in place. Most of the predefined queries to not do this, so you will have to know the data type of the variable on the predefined queries and add quotes for character columns. This is why we recommend making your own versions of all queries and establishing a standard convention for variables.

- When using variables, the comparison will be case sensitive, so be aware as you write the query and enter the value. For example, if your query includes a predicate `WHERE DB_NAME = ':db_name'`, the value you enter should be in uppercase, because that is how the database name is stored in the PE tables.
- If you are using a LIKE predicate, you can either put the '%' character in the where clause - such as:

  ```
  where db_name like ':db_name%'
  ```

  or you can fill it in when you enter the variable value at execution time (see Figure 5-28):

  ```
  where db_name like ':db_name'
  ```

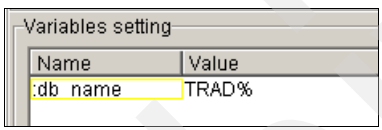


Figure 5-28 Including LIKE character in variable at execution time

- If you want to put comments in your query, you need to put them at the end of the query, as shown in Example 5-1 on page 286. The PE query parser does not seem to like them at the top, or embedded within the query itself. You can also document your query in the query properties and skip the comments completely.
- The MEMBER_ID column is included in all the sample queries. This column refers to the partition number in a DPF environment. In a non-DPF environment, the column value is always 0. The column is, however, included in each table's index, so you may want to consider that when building your query.

Like other components within the Performance Warehouse, there are some predefined queries that come with Performance Expert. The predefined queries are found in the Public group. Unlike Process Groups, you can execute the predefined queries *as-is*; however, most of the PE-provided queries do not have much filtering in them. For example, if you monitor more than one database in a monitored instance, you would probably want to add an additional WHERE clause in your query to only show results for one database. You can use the Copy function to copy into your own Query Group and then modify the query to add the filter.

We already described the basic navigation techniques for creating groups and copying templates in steps 1 and 2 of “General concepts about Process Groups” on page 254, so we do not go into detail about that here. The basic steps are the same throughout the PWH.

Perhaps more useful than using the predefined queries is the ability to create your own queries. It does require that you know some SQL, but PE provides predefined queries you can use as a model.

Sample queries

Here we provide some additional queries that you might find useful. You can copy and paste the text into your own query in the PE Client. These are working samples and can be executed *as-is*, or you can adjust them to suit your needs and style.

Show which tablespaces use which buffer pools

This query (Example 5-1) shows which tablespaces use which buffer pools. You can get this out of the system catalog of course, but we offer this as a demonstration of the different types of information you can get from the PE tables.

Example 5-1 Show which tablespaces use which buffer pools

```
select distinct
  t1.db_name
, t1.bp_name
, t2.tablespace_name

from pwh.bufferpool T1
   , pwh.tablespace T2

where t1.bufferpool_id = t2.tablespace_cur_pool_id
  and t1.db_name       = t2.db_name
  and t1.member_id     = t2.member_id
  and t1.interval_to   = t2.interval_to
  and t1.interval_from = t2.interval_from

order by t1.db_name

-- example 1: Shows which tablespaces use which bufferpools
```

Show which tables reside in which tablespaces

This query (Example 5-2 on page 287) shows the mapping between table and tablespace. This query, as written, does not take into account if a table previously existed in a different tablespace, so if you moved it, you might see it twice in the

report. You could add some date columns or filters to the query if this is important for you.

Example 5-2 Show which tables reside in which tablespaces

```
select distinct
  t1.db_name
, t1.tablespace_name
, t2.table_name

from pwh.tablespace T1
   , pwh.table      T2

where t1.tablespace_id = t2.tablespace_id
  and t1.db_name       = t2.db_name
  and t1.member_id     = t2.member_id
  and t1.interval_to   = t2.interval_to
  and t1.interval_from = t2.interval_from

order by t1.db_name, t1.tablespace_name

-- example 2: Shows which tables reside in which tablespace
--  note - this does not show if you changed/moved the table
```

Query a history table

PE can report on applications in locking conflict, and this data can also be captured into history (it must be enabled). However, locking conflicts may not last very long, so if you look at the Locking Conflicts screen, it could be empty, so you use history mode and go back in time snapshot by snapshot, looking for a conflict. That method works, but is tedious.

The locking conflict snapshot data is stored in the performance database history tables, but not in the Performance Warehouse. We show a query in Example 5-3 on page 288 that pulls data from the history tables (schema DB2PM) instead of the PWH tables. The query will look for any locking conflicts and show what time they occurred. Knowing the time and date, you can use history mode and easily go directly to the conflict.

Attention: This type of query is not officially supported. The PWH Queries are intended as a way to report on long-term data, and the history mode on online screens is used to show short-term data.

Example 5-3 Find locking conflicts in history tables

```
select
  interval_to
, db_name
, table_name
, lock_object_type_st
, lock_mode_st
, lock_attributes

from db2pm.lockedres

where date(interval_to) >= ':lock_date'

-- the date filter is arbitrary, you can change to suit you
-- Use the date and time on the report output to find the spot in history mode
-- on Locking Conflicts screen
```

As an example, we ran this query on our system using October 10 as the “from date”. This is not quite realistic because there is no history data available back that far. The query still works, but you should remember that history data is deleted after a period of time you specify. If you do query history tables, you can only ever see short-term data.

When we ran the query, we found the results shown in Figure 5-29.

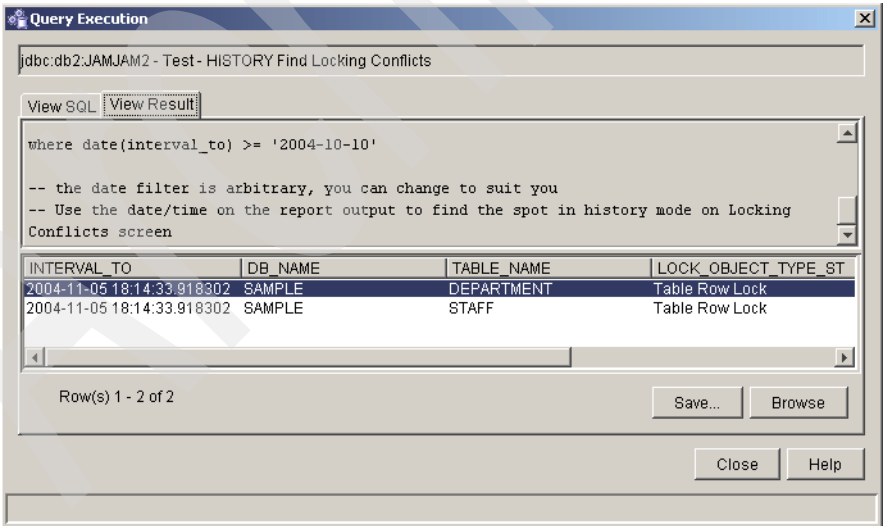


Figure 5-29 Lock conflict query results

We see there was only one conflict since October 10, and it occurred on November 5 at 6:14 PM. We opened the Locking Conflicts screen and moved the history slider back to the 6:14 PM on November 5, and we saw the conflict, as shown in Figure 5-30.

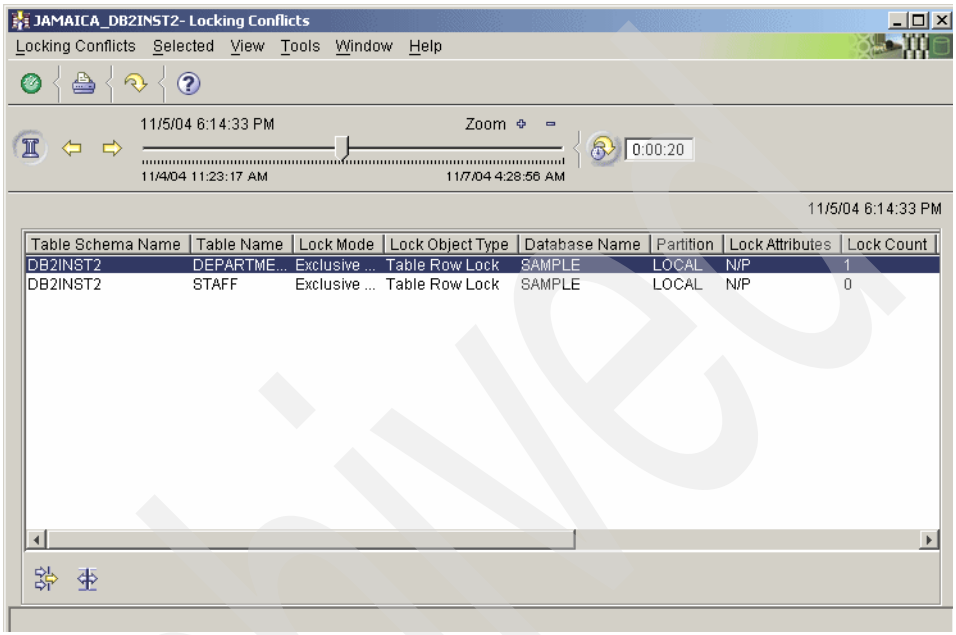


Figure 5-30 Locking Conflicts screen in history mode

Simple trend query

You can write queries that will help identify trends in certain counters. For example, you can write queries for the following:

- Are secondary logs are being allocated, and are they growing?
- What is the average maximum number of connections to a database?
- What DB or DBM CFG values have changed, and when?

We show examples of these in the next pages.

Trends for log usage

This sample query, shown in Example 5-4, reports on some of the log and log space usage monitor elements and allows you to enter a date range. You might be able to track log growth or look for an increase in secondary log allocations, which might indicate some impending space issues in the database. If you collect OS data as well, then you can extend the query with file system information, like Size and Used space of the file system, that is used for the log files. This query is shown in Example 5-5. With this query, you can track log growth together with the implications on the file system.

With any of the trend examples shown in this section, if you see something that looks unusual, you can always run additional reports or queries to try to drill down to more information. If the data is still recent enough to exist also in history, you could also use the online PE screens to examine activity that was happening during the time period where you saw unusual numbers.

Example 5-4 Query for log usage

```
SELECT
interval_to
, db_name

, TOTAL_LOG_USED
, TOTAL_LOG_AVAILABLE
, TOT_LOG_USED_TOP
, SEC_LOGS_ALLOCATED
, SEC_LOG_USED_TOP
, connections_top

FROM pwh.dbase

WHERE db_name = ':db_name'
and date(interval_to)  between ':from_date' and ':to_date'
```

Example 5-5 Query for log space combined with file system information

```
SELECT
db.interval_to
, db.db_name

, DB.TOTAL_LOG_USED
, DB.TOTAL_LOG_AVAILABLE
, DB.TOT_LOG_USED_TOP
, DB.SEC_LOGS_ALLOCATED
, DB.SEC_LOG_USED_TOP
, DB.CONNECTIONS_TOP

, OS.ROOT
```

```
, OS.FILE_SYSTEM_SIZE
, OS.USED_SPACE

FROM pwh.dbase DB, pwh.filesystem OS

WHERE db.db_name = ':db_name' and os.root = ':root'
and date(db.interval_to)  between ':from_date' and ':to_date'
and db.interval_to = os.interval_to
```

What is average, min, and max connections_top?

This query is shown in Example 5-6 is meant to give an example of working with a watermark monitor element. The element in this example is CONNECTIONS_TOP, which is the highest number of simultaneous connections to the database since the database was activated.

Since this is a high watermark, we might be interested to know how high, on average, it is. We might also like to know how low the highest value is, and what is the simple average. Knowing this might help us spot upward or downward trends and help with capacity planning.

Tip: Note the way we used variable names in the example and how they were resolved in the output screen shot.

Example 5-6 Query the connections_top watermark

```
SELECT
  db_name
, DATE(interval_to)
, AVG(CONNECTIONS_TOP) as avg_conn_top
, MIN(connections_top) as min_conn_top
, MAX(connections_top) as max_conn_top

FROM PWH.DBASE

where date(interval_to) between ':from_date'  and ':to_date'
 and  db_name in (':db_name1',':db_name2',':db_name3')

GROUP BY db_name, date(interval_to)
```

We executed this query on our test system and found the results shown in Figure 5-31.

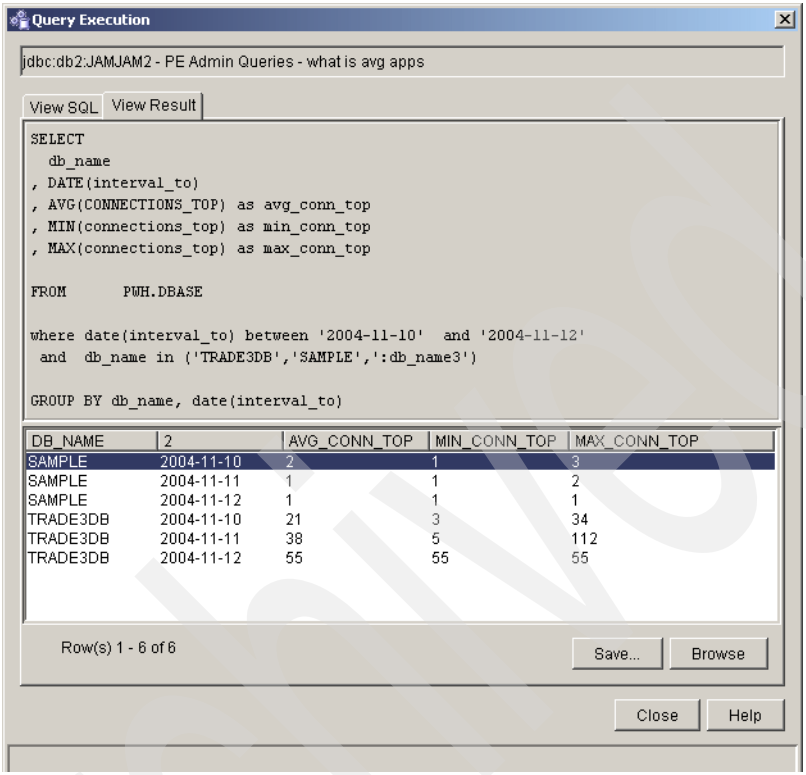


Figure 5-31 Query output for connections_top

What process execution logs are taking up the most space?

Whenever you run a Process Report, the output is stored in the Performance Warehouse. You can delete the reports from the Process Execution log. It could be useful, however, to know if there are some reports that are taking up a large amount of space, or perhaps are orphan reports that got created, but the process failed for some reason and the log record is no longer available (this could happen in the case of Buffer Pool Analysis reports).

We provide a query here that can look at the internal PE table that lists the reports and their sizes. The reports reside in the OUTPUT tablespace, so if you notice that tablespace has grown especially large, you might want to try this query to see if you can determine which reports might be taking up the most space, and then you can take action to delete reports as necessary.

Attention: This query (Example 5-7) is *not* officially supported. If you use the query, you should still use the PE Client to delete the reports; do not attempt to delete the rows directly.

Note the tables in this query have different privileges than the others, so be aware of this situation.

Example 5-7 Query to show space used in process execution log

```
select
  p_creator  as Owner
, p_name  as ProcessName
, di_size as Size_in_bytes
, di_creationts as RunDate
, di_dsname
, di_description
, di_id
, di_sl_id

from db2pm.dsindex , db2pm.steplog, db2pm.process
where di_sl_id = sl_id and sl_p_id = p_id
order by di_size desc, di_creationts
```

Relationship between Process Groups, RoT, and Queries

The predefined processes, queries, and Rules of Thumb all operate on the same data from the PWH. They also have some similarities, so you can compare the output of the different features and also look at the input, which will help you determine which feature might best suit your needs.

In Figure 5-32 and Figure 5-33 on page 295, we show which predefined queries map roughly to the Processes and RoT. If you examine the SQL in the query, for example, you can see how the data in the process was selected. These are not the exact SQL used in the report, but close enough for you to study and learn. This is how we learned to create the queries for this redbook.

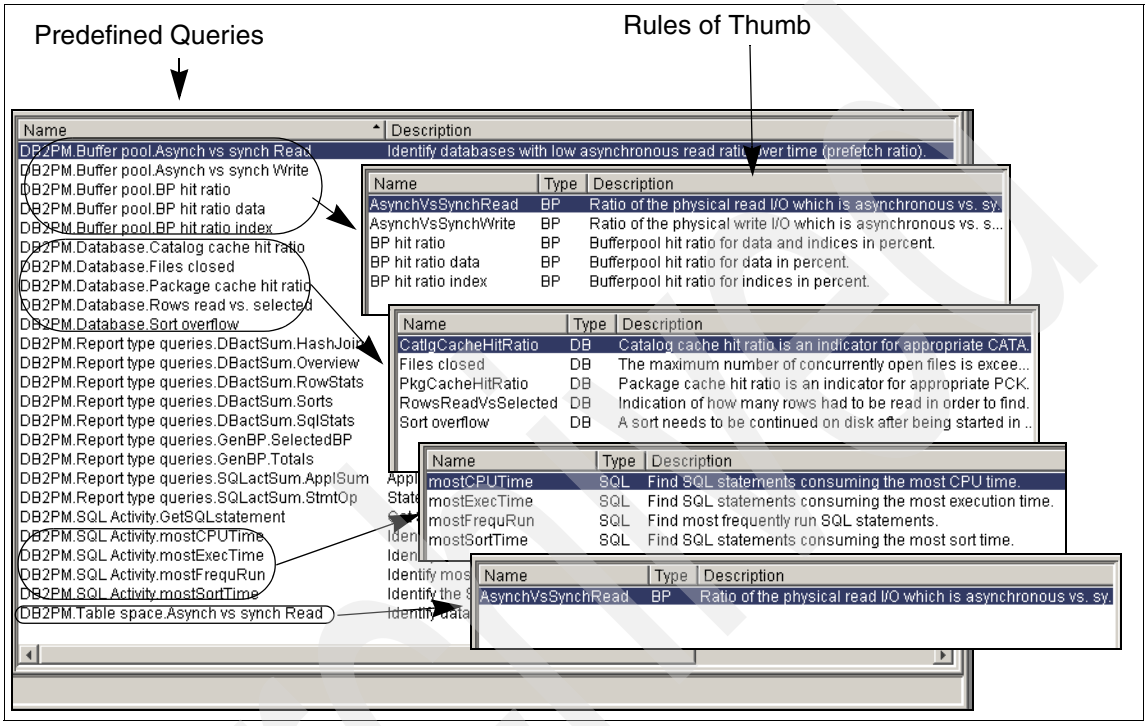


Figure 5-32 Predefined queries and Rules of Thumb

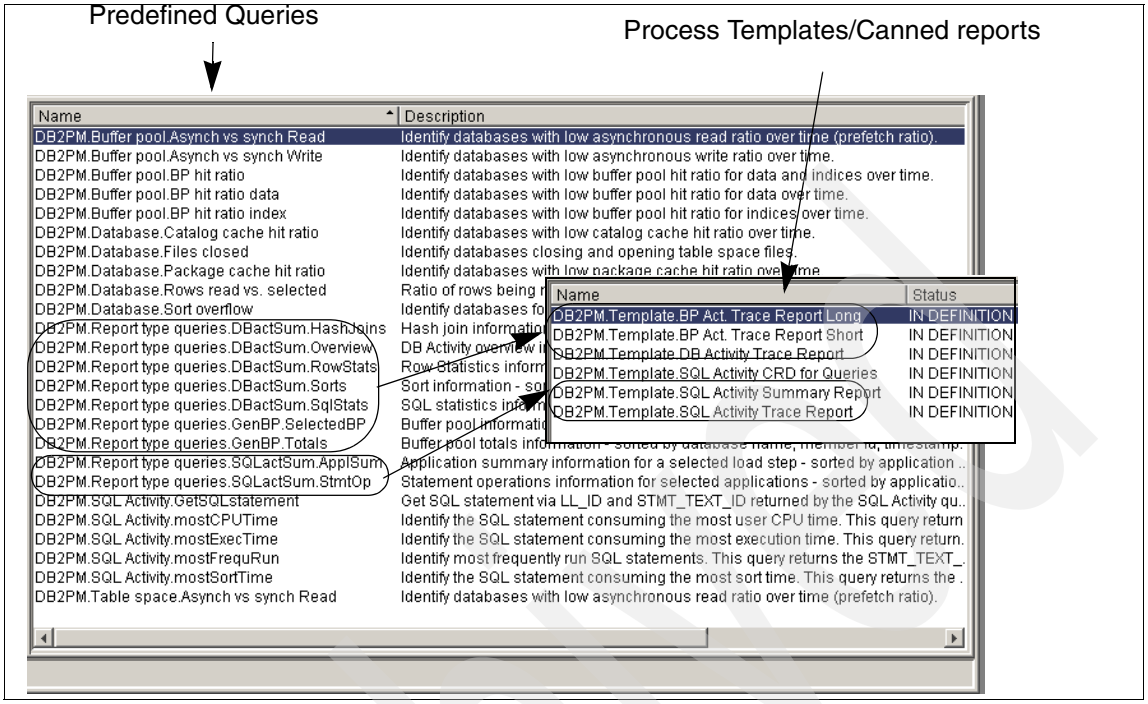


Figure 5-33 Predefined queries and Process template reports

5.2 Buffer Pool Analysis

The three DB2 UDB database objects that can have a significant impact on database performance are buffer pools, tablespaces, and indexes. In this section, we discuss how DB2 Performance Expert reports and diagnoses buffer pool effectiveness.

At the risk of patronizing readers familiar with DB2, we briefly explain what is the function of buffer pools. Fundamentally, they are an area of global memory dedicated to DB2 for caching data. This is either for new data added to the database or for data retrieved from the database. It is the “scratchpad” area for DB2, enabling fast access in memory as opposed to significantly slower access from disk.

In the *Advanced DBA Certification Guide and Reference: for DB2 Universal Database v8.1 for Linux, UNIX, and Windows*, by Snow, et al, it states “Because most data manipulation takes place in the database buffer pools, configuring the buffer pools is *the single most important tuning area for DB2 UDB databases*.” It also recommends multiple buffer pools over a single buffer pool, and that “buffer

pools will require constant monitoring and tuning in order to keep them performing optimally."

This is one of the most convincing reasons to use DB2 Performance Expert, as it records comprehensive data on buffer pools to a) view immediately and b) to generate reports for future trend analysis.

Each database has at least one buffer pool, namely IBMDEFAULTBP, which is created when the database is created. The memory for the buffer pool is allocated on the same machine as the database.

There is some good general advice about buffer pools that is accessible online in DB2 Performance Expert by clicking on the **Help** button from the Rule-of-Thumb Properties screen. Access to, and using Rules of Thumb, are discussed in 5.1.5, "Rule of Thumb analysis" on page 272. The full help text is reproduced in Appendix A, "Rules of Thumb definition samples" on page 457. There are five buffer pool Rules of Thumb. The first considers the ratio of asynchronous (prefetched) to synchronous (random) physical reads. The second discusses the ratio of asynchronous to synchronous writes. The other three rules concern hit ratios, or ratios of physical to logical reads. We consider each of the three categories:

- Ratio of asynchronous to synchronous physical reads

  The more data that is prefetched, the more efficient the buffer pool is behaving. The suggested thresholds for warning and problem are 97 and 90, respectively. Data Warehouse (DW) environments can expect lower thresholds.

  The number of I/O servers set by the database configuration parameter NUM_IOSERVERS affects the amount of prefetch I/O for the database. The default value is three, and is also the recommended minimum. Generally, for DW, it should be set to the number of disks. For OLTP, it should be set to the number of CPUs.

  The parameter values set for PREFETCHSIZE and EXTENTSIZE, when creating a tablespace, has a direct effect on the prefetch to random read ratio. The prefetch size should be a multiple of the number of tablespace containers and the tablespace extent size. The default for both parameters is 32 pages. Generally, the prefetch size should tend to be reduced, while the extent size should tend to be increased. However, because these values cannot be changed after the tablespace has been created, they are inconvenient to experiment with, because the tablespace has to be re-created each time.

  Lastly a decreasing prefetch ratio can indicate a need to reorganize tables if they have become unclustered.

- Ratio of asynchronous to synchronous physical writes

  Asynchronous write I/Os are generally twice as fast as synchronous write I/Os. Therefore, the recommended warning and problem thresholds are recommended to be set high at 95 and 90 percent, respectively.

  The database parameter NUM_IOCLEANERS specifies the number of asynchronous page cleaners for a database. These page cleaners write changed pages from the buffer pool to disk before the space in the buffer pool is required by DB2. The default value is one. This should be increased to the number of CPUs.

  Note: Even with a single CPU, the DB2 Configuration Advisor can recommend a value of two for NUM_IOCLEANERS.

  Another database parameter that can be adjusted to improve the asynchronous write percentage is CHNGPGS_THRESH. This parameter specifies the percentage of changed pages at which the asynchronous page cleaners will be started. The default value is 60 percent. When 60 percent of the pages in the buffer pools become dirty, the page cleaners asynchronously write the changed pages out to disk. For a DW, this is reasonable, as there are few updates. However, for OLTP, the resultant surge in write I/Os to disk can cause a temporary blip in transaction elapsed times. The recommendation is to set CHNGPGS_THRESH to a minimum of 30 percent of the database parameter SOFTMAX, which itself has a default of 100 percent.

- Hit ratios of physical to logical reads

  The buffer pool hit ratios are ratios of physical data reads to logical data reads. This is presented in three ways:

  – Data reads only.
  – Index reads only.
  – Combination of data and index reads; this value is between the other two values.

  In all three circumstances, the advice is generally the same. The warning and problem thresholds are 90 and 80 percent, respectively.

  – Increase the number of buffer pool pages for OLTP

    This can be relaxed a little for DW; less than 80 percent is acceptable if the prefetch ratio is high.

- Consider the actually physical space of the buffer pool to the tablespaces that use the buffer pool.

  If the size of the buffer pool is not much smaller than the tablespace, then a high hit ratio is inevitable. However, if the tablespace is thirty times larger than the buffer pool, a high hit ratio will be less likely.
- Note that when a database manager has just started, the hit ratio will be much lower.
- If the hit ratio decreases over time, this could indicate data volumes are growing or, due to insert and delete activity, the table needs reorganizing.

Due to the buffer pools' importance to database performance, their diagnosis appears in most functional areas of DB2 Performance Expert. Buffer pool information can be obtained from the following menu items in the System Overview screen:

- Application Summary
- Statistics Details
- System Health
- System Parameters - Databases
- Performance Warehouse - Report
- Performance Warehouse - Analysis
- Performance Warehouse - Expert
- Buffer Pool Analysis

Periodic Exception Processing also incorporates buffer pool information by creating alerts for key user defined buffer pool thresholds. This is covered in 4.2, "Alerts and exceptions" on page 193.

Note: Although there are three separate menu items for Performance Warehouse, the functions of "Report" and "Analysis" can both be accessed directly from the "Performance Warehouse - Expert" menu item.

The following describes where to access buffer pool information in DB2 Performance Expert:

- Application Summary

  From the Application Summary screen, double-click an application to open:
  - Application Details

    Select **Memory Pool Usage**, which displays three buffer pool heap values:
    - Current Size of Memory Pool
    - Max Fragment Size - high water mark
    - Fragments

Select **Buffer Pool** from the left pane:

- Buffer Pool

  Some of the details found here are Hit Ratio, Index Hit Ratio, and Data Page Hit Ratio, as well as Logical Reads, Physical Reads and Writes, and Sector Read and Write timings.

- Statistic Details

  From the left pane in Statistics Details, click **Databases**.

  Double-click one of the databases listed in the right pane, and then click **Buffer Pool** back in the left pane. This will display hit ratios and Read and Write information. These details are aggregated numbers for all the buffer pools in the database selected.

  Further details like physical read and write times and prefetch information can be displayed by expanding the tree under Buffer Pool, and clicking **Details**.

  Details per individual buffer pool can be found lower in the tree in the left pane in Statistic Details; close the current databases page and click **Buffer Pools**.

  The right pane then displays a list of all the buffer pools defined in the instance being monitored (see Figure 5-34).

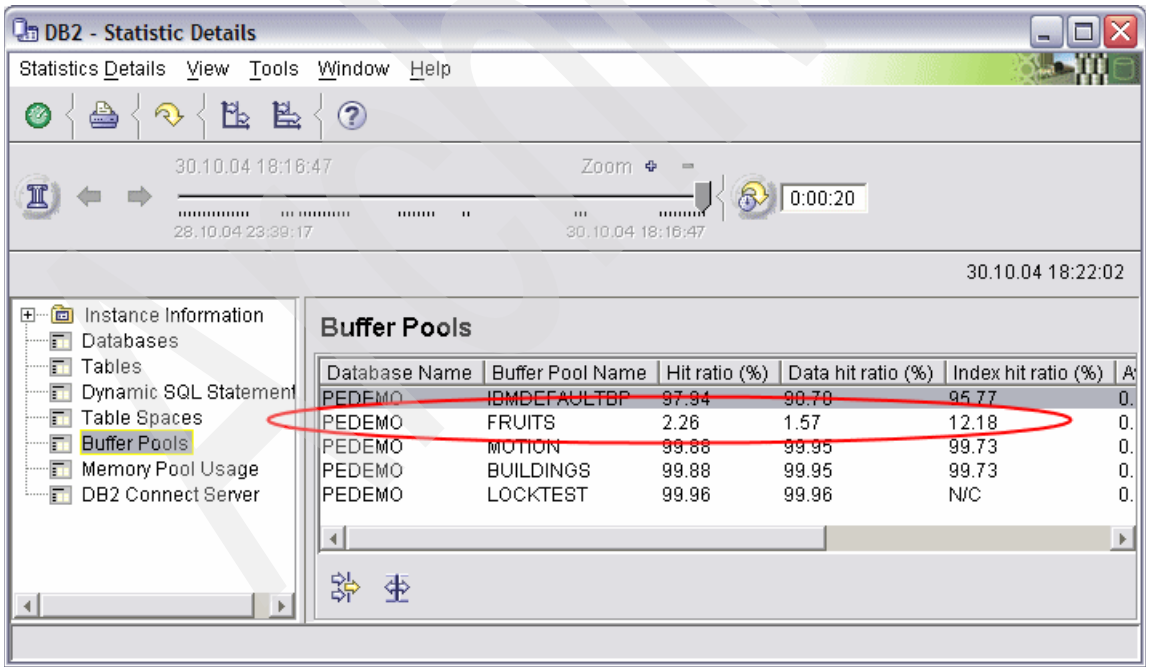


Figure 5-34 Accessing important buffer pool information from Statistics Details

Drilling down for more details by double-clicking on a particular buffer pool opens up another tabbed page with size information. We select the FRUITS buffer pool, since it has a low hit ratio. In this case, the hit ratio problem is due to the buffer pool having been allocated a small amount of space. This can be seen in Figure 5-35.

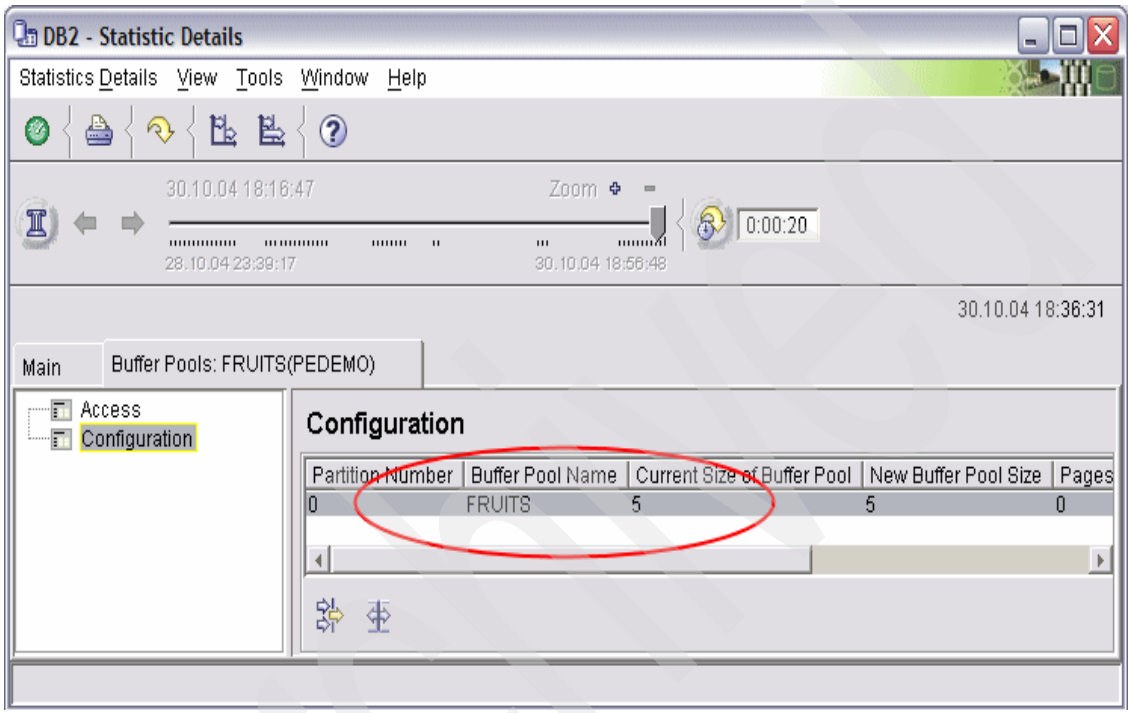


Figure 5-35 Current size of problem buffer pool

Finally, in Statistics Details, from the Main tabbed page, click **Memory Pool Usage** to show the buffer pools' contribution to Current Size of Memory Pool, Maximum Fragment Size, and number of Fragments.

- System Health

  Several predefined data views exist that give buffer pool information, for example:

  - Compare Application Hit ratio of buffer pools
  - Compare Data Hit ratio of buffer pools
  - Compare Hit ratio of buffer pools
  - Compare Index Hit ratio of buffer pools
  - Hit ratio of a buffer pool

In this example (Figure 5-36), three Data Views have been created for buffer pool, data, and index hit ratios. They clearly illustrate decreasing effectiveness for the FRUITS buffer pool.

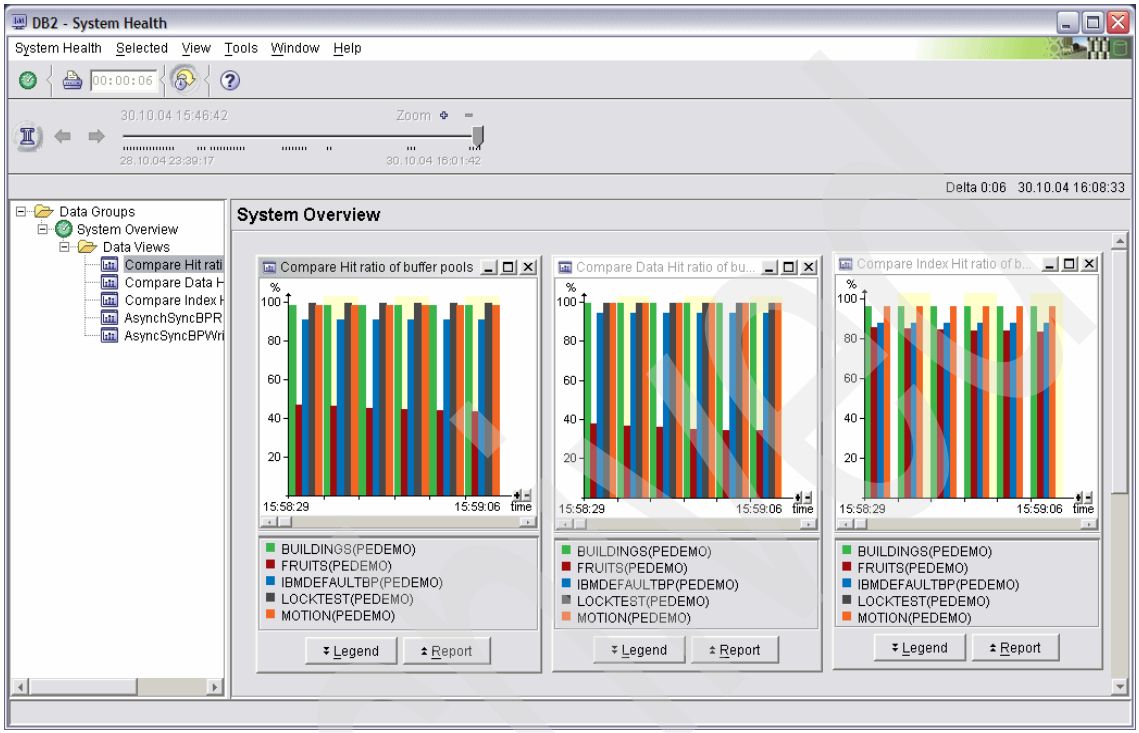


Figure 5-36 Data views illustrating declining buffer pool efficiency

To increase the FRUITS buffer pool size, run the following command in a DB2 command window:

```
ALTER BUFFERPOOL FRUITS IMMEDIATE SIZE 250
```

- System Parameters - Databases

From the System Parameters Databases screen, double-click a database. The right pane then displays the Capacity Management details. The Database shared memory section includes the counter Buffer pool size (pages).

- Performance Warehouse - Expert

  The Performance Warehouse is a large component of Performance Expert. It is split into three areas:

  - Processes: Which can be scheduled.
  - Rules of Thumb: Which can be tailored or new ones added.
  - Warehouse queries: 25 predefined queries and option to create your own.

  The areas of Performance Warehouse that specifically relate to buffer pools is covered in 5.2.4, “Performance Warehouse for buffer pool analysis” on page 312.

- Buffer Pool Analysis

  This is the menu item dedicated to investigating buffer pools. This function is split into two main areas, which are generated simultaneously each time a buffer pool report is created: Text Reports and Interactive Reports. The report generation and content of these reports are described in 5.2.1, “Generating Buffer Pool Analysis reports” on page 302.

5.2.1 Generating Buffer Pool Analysis reports

In DB2 Performance Expert, all buffer pool information is derived from DB2 snapshots. The snapshots are taken periodically as set in the Recording interval field, in the Performance Expert Server Properties panel under the History tab. In these examples, the snapshots are taken every 60 seconds. The information used to populate the Buffer Pool Analysis reports is stored in the Performance Warehouse. This data is summarized based on the Granularity for summarizing the history data field in the Performance Expert Server Properties panel under the Performance Warehouse tab in the client (see Figure 5-37 on page 303).

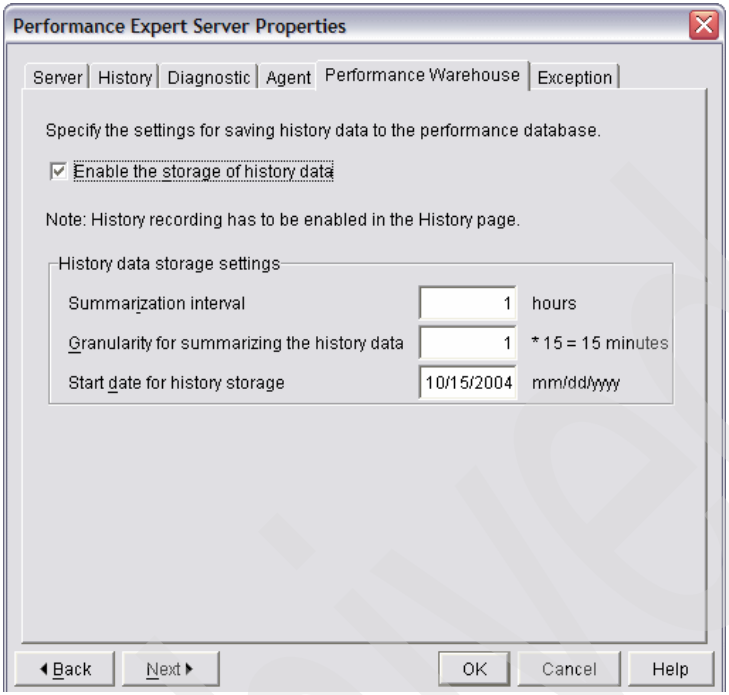


Figure 5-37 Performance Expert Server Properties - PWH

For example, if the granularity is set to one, the history snapshots that are taken every 60 seconds will be summarized over 15-minute periods, resulting in four rows per hour in the Performance Warehouse tables. It is this summarization data that is used to populate the buffer pool reports.

The first time you access the **Buffer Pool Analysis** screen, no databases or reports are displayed, as shown in Figure 5-38.

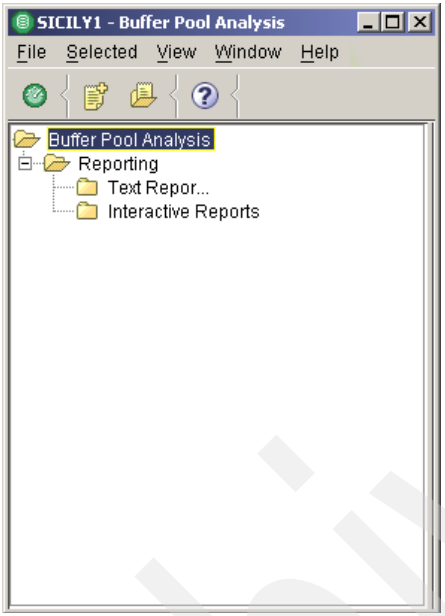


Figure 5-38 New BPA screen

The next step is to generate reports for the databases and time periods in which you are interested. On the Buffer Pool Analysis menu bar, select **File** → **Generate New Report** or use the toolbar button ( ).

The Report Generation dialog box is displayed (see Figure 5-39 on page 305). From here you select the database on which you are reporting from the drop down list, and enter the start and end times for the reporting period.

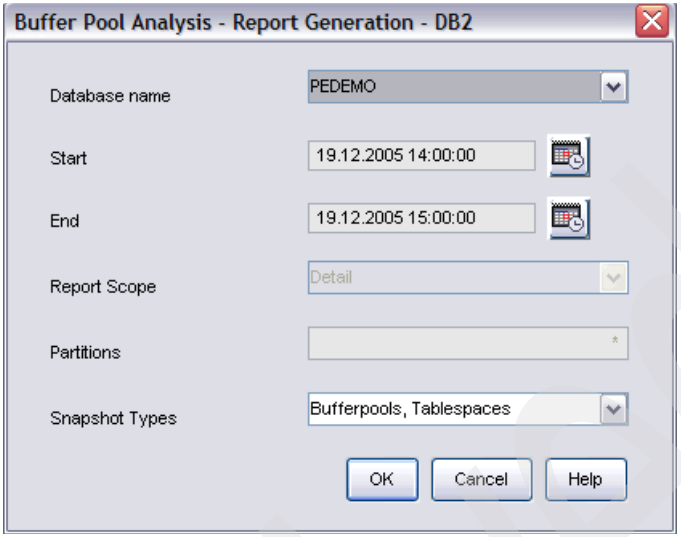


Figure 5-39 Report Generation parameter dialog box

The End time defaults to the last time stamp of available data in PWH, and the Start time defaults to the End time less the previously monitored period. But be aware that the period you specify to report on may not exactly match the period for which snapshots were taken.

Note: Although the Report Generation pop-up allows you to specify very specific start and end times, the reports will only include information from the summarized history data in the Performance Warehouse, at best, every 15 minutes.

Therefore, it is prudent to only generate reports that are coincident with the summarization of your warehouse reporting.

In the example shown, we are reporting on a non-partitioned database, namely PEDEMO. The Snapshot Types field determines whether we include tables in the report, or just buffer pools and tablespaces.

Note: The dates in the dialog box are displayed in the format of the machine where the PE Client is installed.

Click **OK** to continue to the next pop-up screen, which displays the progress generating the Text Reports and Interactive Reports (seen in Figure 5-40 on page 306).

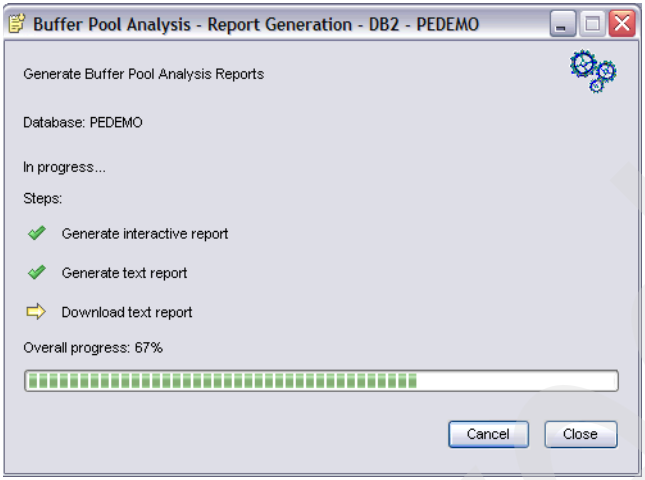


Figure 5-40 Buffer Pool Analysis Report Generation

After you have generated several reports, the Buffer Pool Analysis screen will look similar to Figure 5-41 on page 307. The reports are grouped into Text Reports or Interactive Reports, and further grouped by database.

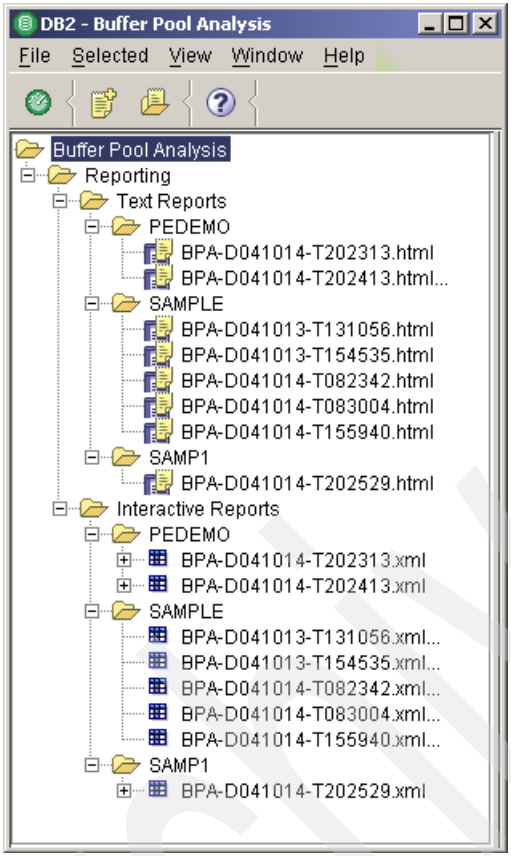


Figure 5-41 Buffer Pool Analysis screen populated with reports

Each time a buffer pool report is generated, two files are created: a HTML file for the text report and an XML file for the interactive report.

Both types of file are stored locally at the same location as the PE Client. The HTML reports have the format BPA-Dyymmdd-Thhmmss.html and are stored in the following text folder C:\Documents and Settings\Userid\.db2pev2\bpa-luw-reports\hostname-port-DB2PM database\text\monitored database.

The XML reports are similarly named, but with an XML file extension and are stored in the folder C:\Documents and Settings\Userid\.db2pev2\bpa-luw-reports\hostname-port-DB2PM database\xml\monitored database.

Note: The XML file itself only contains metadata about the report. The actual data about the buffer pools, tablespaces, and tables is selected from the PWH tables when you open the XML report.

The reports are split into two major types, which are discussed separately: Text Reports and Interactive Reports.

5.2.2 Buffer Pool Analysis Text Report

The Buffer Pool Analysis Text Reports are viewed from a Web browser. They display details collected from DB2 snapshot reports stored in the Performance Warehouse tables. The report is over 50 columns wide. This can appear to be an overwhelming amount of information on first sight. There are several columns that can be useful to check first, especially significant changes in values between subsequent runs.

Here is a small extract from the buffer pool analysis Text Report of the IBMDEFAULTBP buffer pool in Figure 5-42.

Buffer Pool Name: IBMDEFAULTBP

Buffer Pool Characteristics

Buffer Pool for Database : IBMDEFAULTBP(PEDEMO)
Buffer Pool ID : 1

Partition ID	Snapshot Time	Hit Ratio (%)	Data Hit Ratio (%)	Index Hit Ratio (%)	Average Asynchronous Read Time (sec.ms)	Average Asynchronous Write Time (sec.ms)	Average Data Pages Read per Asynchronous Request	Average Sectors Direct Reads	Average Sectors Direct Read Time (sec.ms)
member_id	interval_to	pool_hit_ratio	data_page_hit_ratio	ipool_hit_ratio	avg_async_read_time	avg_async_write_time	avg_data_rpa_sync_req	avg_direct_sec_reads	avg_direct
0	08:00:00.0	63	74	54	0	0	0	3	
0	08:15:00.0	100	100	100	0	0	0	2	
0	08:30:00.0	95	98	92	0	0	0	2	
0	08:45:00.0	81	89	74	0	0	0	2	
0	15:00:00.0	64	75	56	0	0	0	3	
0	15:15:00.0	100	100	100	0	0	0	2	
0	15:30:00.0	90	94	85	0	0	0	2	
0	15:45:00.0	81	89	74	0	0	0	2	
0	16:00:00.0	93	95	91	0	0	0	2	
0	16:15:00.0	93	96	91	0	0	0	2	
0	16:30:00.0	94	96	92	0	0	0	2	
0	16:45:00.0	95	96	93	0	0	0	2	
0	17:00:00.0	98	98	95	0	0	2	2	
0	17:15:00.0	97	98	95	0	0.012286	4	2	

Figure 5-42 Sample BPA Text Report

There is currently a restriction when printing the HTML buffer pool reports. The print is restricted to the width of the screen; therefore the information on the right can only be viewed online.

The DB2 buffer pool snapshot command returns a number of different direct buffer pool data items. The Buffer Pool Analysis Report holds a subset of these, plus some other derived values. Derived values are averages and ratios that are based on the raw data.

For example, the buffer pool hit ratio is derived from the formula:

```
( 1 - ( ( BUFFERPOOL.POOL_DATA_P_READS + BUFFERPOOL.POOL_INDEX_P_READS ) / (
BUFFERPOOL.POOL_DATA_L_READS + BUFFERPOOL.POOL_INDEX_L_READS ) ) ) * 100
```

The *Advanced DBA Certification Guide and Reference: for DB2 Universal Database v8.1 for Linux, UNIX, and Windows*, by Snow, et al, lists 16 of frequently used snapshot values in tuning DB2 UDB performance, of which the first 15 are included in the buffer pool report:

1. Data Physical Reads
2. Data Logical Reads
3. Asynchronous Data Reads
4. Index Physical Reads
5. Index Logical Reads
6. Asynchronous Index Reads
7. Data Writes
8. Asynchronous Data Writes
9. Index Writes
10. Asynchronous Index Writes
11. Total Buffer Pool Physical read Time
12. Asynchronous Read Time
13. Total Buffer Pool Physical Write Time
14. Asynchronous Write Time
15. Asynchronous Read Requests
16. Time Waited for Prefetch

5.2.3 Buffer Pool Analysis Interactive Report

The Interactive Reports are not viewed from a browser, but are viewed from within the Buffer Pool Analysis screen. The details are displayed graphically in the right hand pane for each variable.

Each Interactive report (Figure 5-43 on page 310) is split into two parts:

- By Snapshot Type
- Hierarchical View

All the useful information is held under the first section: Snapshot Type.

The Hierarchical View simply lists, in the left pane, all the buffer pools, tablespaces, and, optionally, tables that are detailed in the Snapshot section. For the lowest level in the hierarchy, which is either tablespaces or tables, the snapshot data is also included. The Hierarchical View makes it simple to identify which tables were active in which tablespaces and which tablespaces were active in which buffer pool.

The rest of this section focuses on the Snapshot Type section of the report.

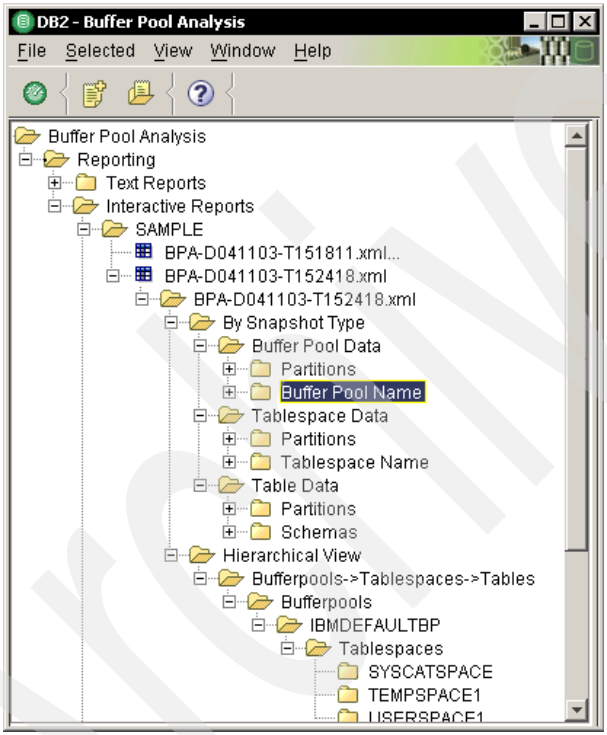


Figure 5-43 BPA Interactive Report tree structure

The Snapshot Type section is split into the following sub-sections:

- Buffer Pool Data
  - Partitions
  - Buffer Pool Name
- Tablespace Data
  - Partitions

  - Tablespace Name
- Table Data
  - Partitions
  - Schemas

Note: When the Buffer Pool Analysis screen is first displayed, the Interactive Reports do not have the “plus” or “minus” expansion boxes on the tree elbows. Therefore, you have to double-click the report item to access the sub-folders.

In a non-partitioned database, the Partitions folder holds the total buffer pool values for all buffer pools. Within the Buffer Pool Name folder, the data is divided per separate buffer pool.

Expanding the default node folder beneath the Partitions folder displays the Total Values folder. Double-click this folder to list 32 buffer pool variables in the right hand pane. Within this Partitions folder, listed underneath the Total Values folder, are 47 buffer pool measured variables. Double-clicking on **Buffer Pool Data Logical Reads** for example, displays a chart in the right hand pane, with one line graph per buffer pool in the database being reported on. The values begin at the second snapshot in the reporting period.

Note that some values are calculated per minute, and not per second, as they are in the Text Reports.

Viewing buffer pool information in the form of the interactive reports can make it easier to notice changes in buffer pool usage. There are 48 individual interactive reports you can view. See Figure 5-44 for an example of a Data Hit Ratio interactive XML chart for all buffer pools in the PEDEMO database.

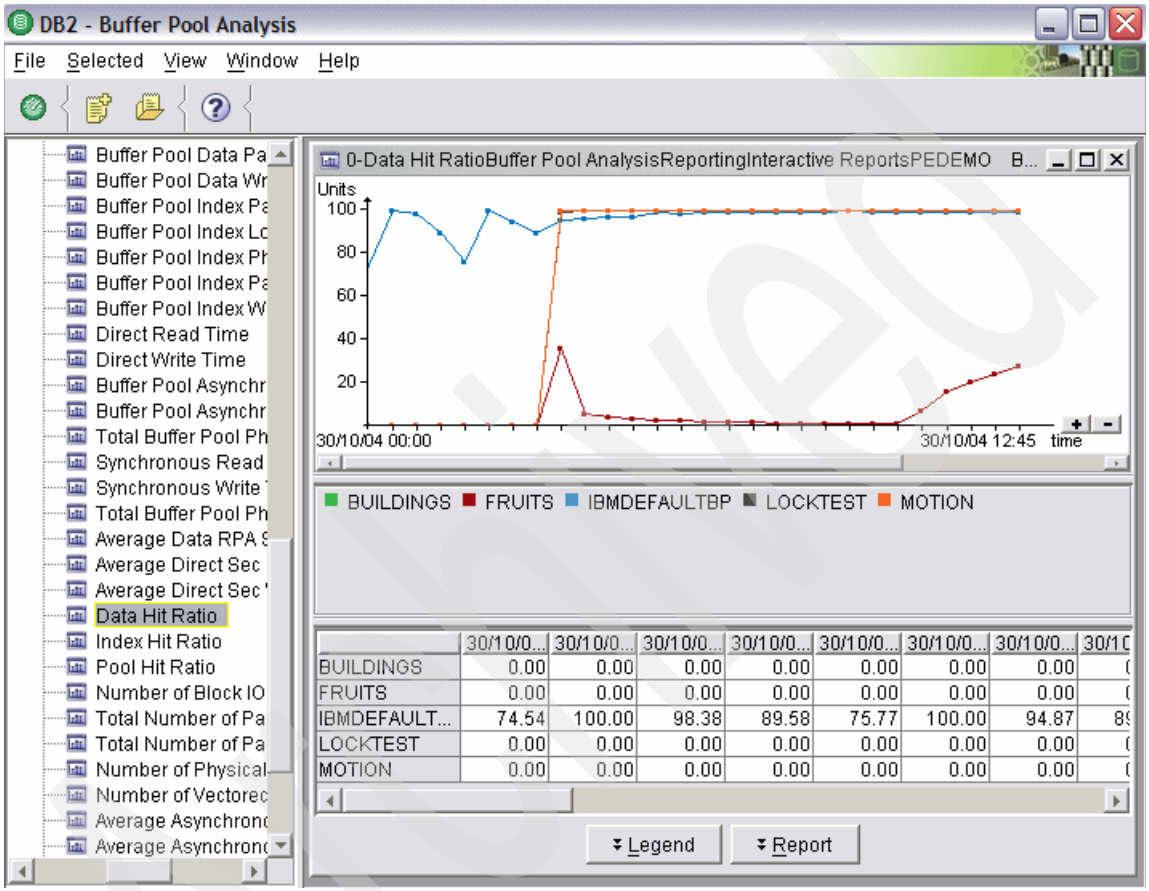


Figure 5-44 Example of BPA Interactive Report

5.2.4 Performance Warehouse for buffer pool analysis

The information derived from the Buffer Pool Analysis section of DB2 Performance Expert is obtained from the data stored in the Performance Warehouse. Therefore, it is not surprising that much of the functionality of the Performance Warehouse involves buffer pool reporting.

We consider the three major areas of the Performance Warehouse:

- Process Groups: Schedules generation of reports.
- Rules-of-Thumb: Runs pre-formed or modified reports.
- Queries: Investigates problem data determined by performance indicators.

Process Groups

Processes enable you to execute and schedule predefined reports. There are two Public Processes, or templates, for specifically reporting on buffer pool activity. These are long and short buffer pool activity reports. The short report includes buffer pools and tablespaces, while the long report includes tables.

Note: Although both buffer pool public reports are called “activity traces”, after the templates have been copied to your own process group, they can also both be run as summary reports.

In order to change or even run the Public Processes, they must be copied into your own Process Group. Create your own process group as follows:

Select **System Overview** → **Performance Warehouse - Expert**, right-click **Process Groups** and select **Create**. Enter **Name,** for example, PEDEMO, select **Description**, and click **OK**.

Navigate back to the Public Processes, right-click **DB2PM.Template.BP Act. Trace Report Long** and select **Copy**. Copy to your newly created PEDEMO process group.

You can customize the process name at this point or rename it later by right-clicking on the process and selecting **Rename**. This enables you to create as many processes as you require to generate “buffer pool long and short” reports covering different periods or with different filters.

Repeat the method to also add your own DB2PM.Template.BP Act. Trace Report Short process.

The resultant window should look like Figure 5-45.

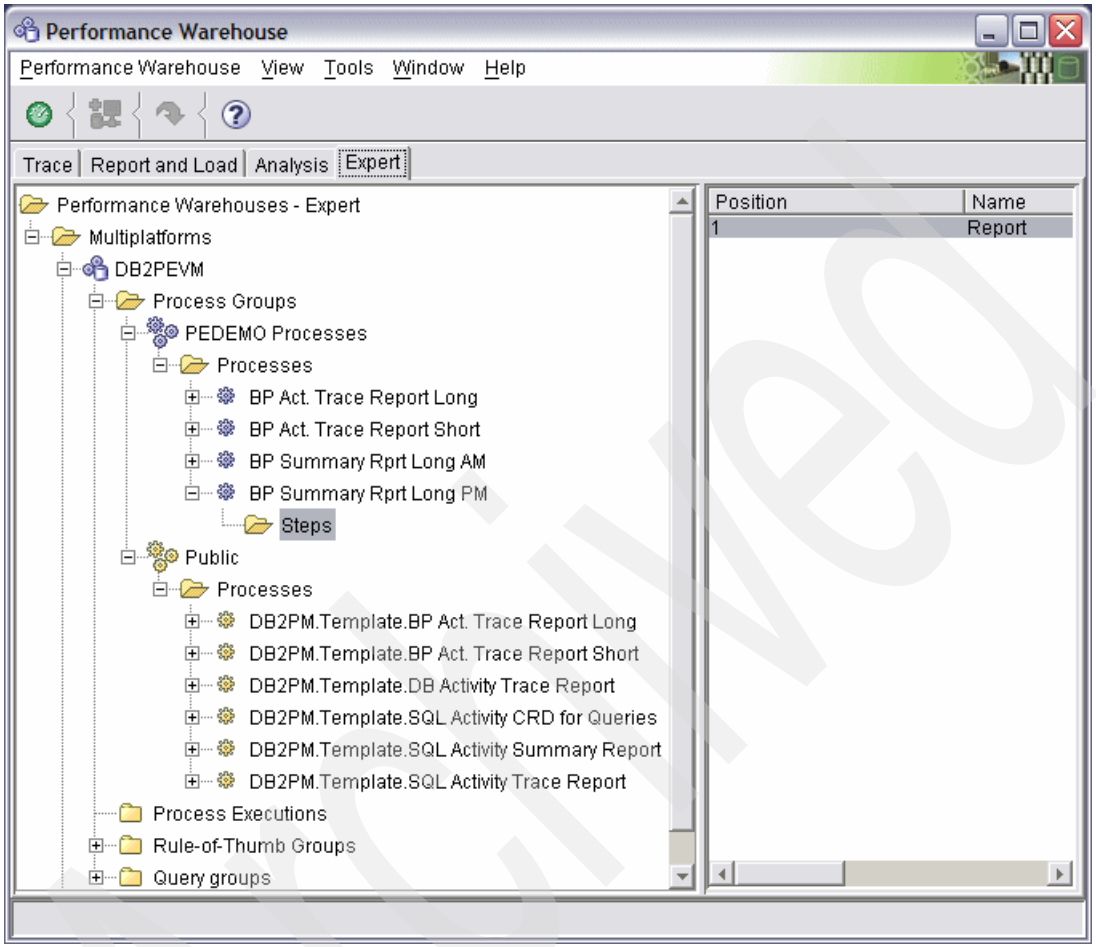


Figure 5-45 PEDEMO Process Group

Customizing and scheduling a process

The report can be customized from the Report Step Properties page. This enables you to specify the interval, the database and the type of report, and filter the objects that are in the report. The scheduling function of the warehouse enables you to create buffer pool reports regularly from the processes copied or created in the PEDEMO Process Group.

Double-click one of your process reports to show the process step in the left pane. Click the process **Steps**, right-click **Step Detail** in the right pane, select **Properties**, and select the **Options** tab.

Check the radio buttons for **Date relative to process execution** and enter the value “-1” in the From and To Offset fields to signify the report will be run for *yesterday's* data. Set the **Time** variables from 12:00:00 to 18:00:00, because we are setting up the PM report. Make sure the **Database name** is set to PEDEMO. Set the **Type** of report to Summary. This summarizes the snapshot data from the warehouse over the reporting interval into a single row.

Finally, click the button with the three dots at the bottom of the panel to set the filters. Here you can set the criteria to filter the report on buffer pool, tablespace, and table. You can specifically exclude objects or include objects in the report. The left pane displays the objects in the report query predicates, which you can filter. Click one of the buffer pools, tablespaces, or tables to highlight the row. Select **Add**, select a Boolean operator, enter the **Value** of the object, and click **OK** twice when the addition of the filters is complete.

The resulting process, Report Step Properties, is shown in Figure 5-46.

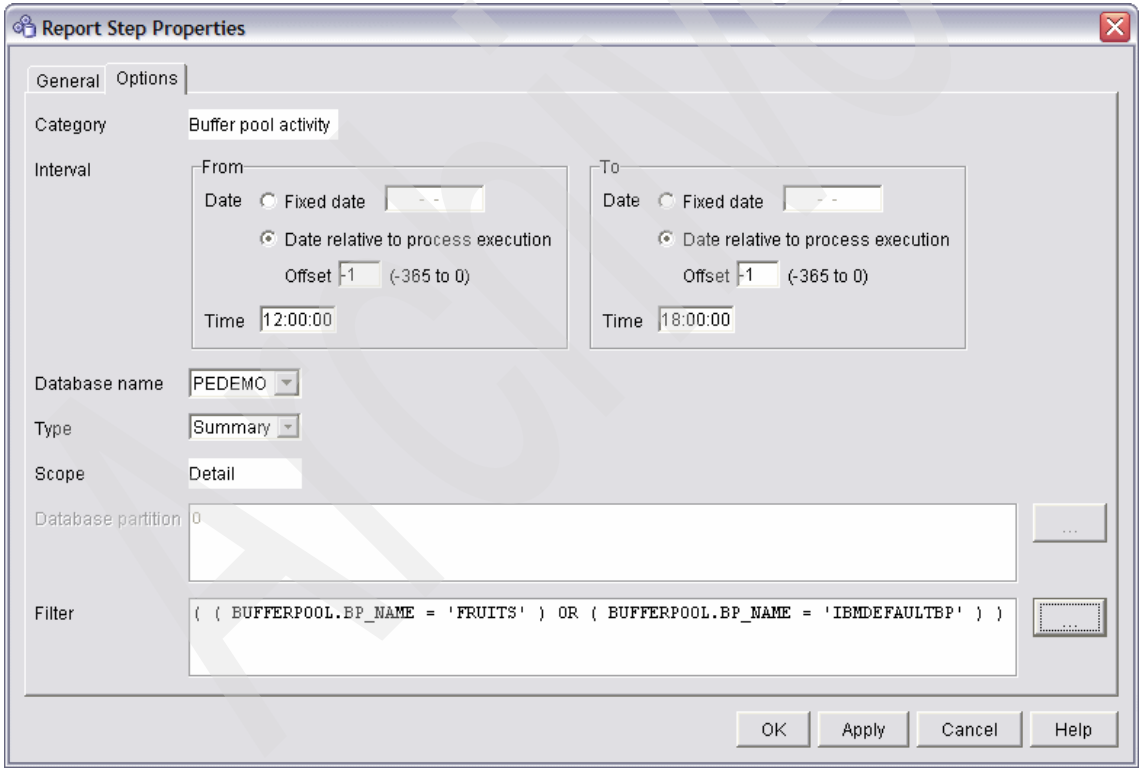


Figure 5-46 Sample values for summary report step

The next step is to schedule the execution of the report.

Right-click **BP Summary Rprt Long PM** and select **Properties** → **Modify Schedule of process**. The Schedule Process window is displayed. After you have entered the schedule information, click **Finish**. Back on the Process Properties window, change the **Status** from “in definition” to “active”, and click **OK**.

In Figure 5-47, the schedule has been set up to produce the report each Monday at 08:00 AM. The *content* of the report is yesterday afternoon’s buffer pool summary report, as determined by the Step Properties.

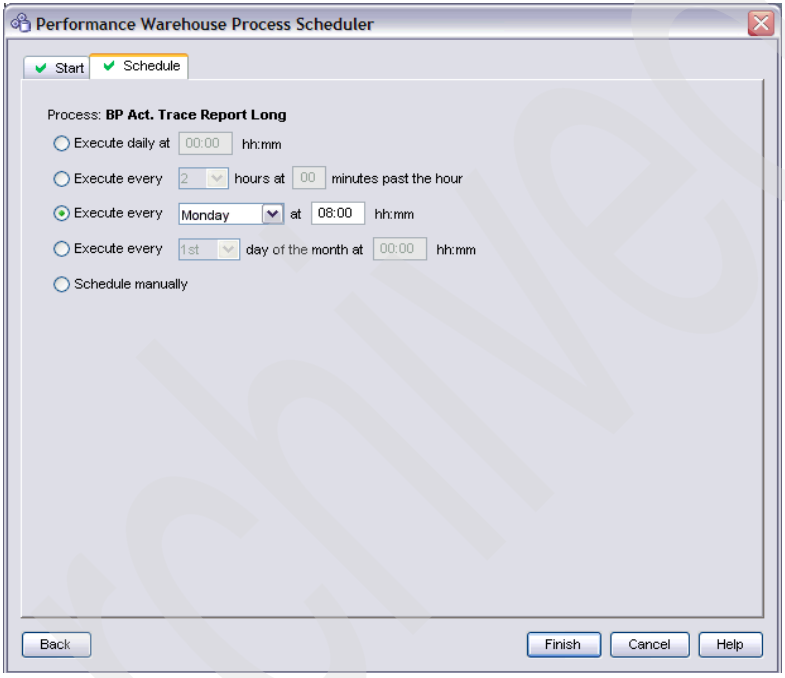


Figure 5-47 Process schedule details

Access the generated reports as follows:

Select **Performance Warehouse - Expert**, click **Process Executions**, double-click **BP Summary Rprt Long PM** in the right pane, click **Open** in Process Execution Details**,** select the report, and click **Open**.

The report is in HTML format and is displayed in your browser. The file is saved in C:\Documents and Settings\Administrator\.db2pev2\temp\1099681357922.html.

The report heading details the type of report, when it was generated, the time interval reported on, and if filters were used (see Figure 5-48 on page 317).

Buffer Pool Summary Report

Report Generation Options: Detailed Summary Report
Report Generation Time: Fri Nov 05 18:32:15 GMT 2004
Report Filters are used.

Configuration

Server Instance Name	: DB2
Database Name	: PEDEMO
Codeset	: IBM-1252
Country	: US
Reported Partitions	: 0
Reporting Interval From	: 2004-10-30 12:00:00.0
Reporting Interval To	: 2004-10-30 18:00:00.0

Buffer Pool Name: FRUITS

Buffer Pool Characteristics

Buffer Pool for Database : FRUITS(PEDEMO)
Buffer Pool ID : 2

Partition ID	Hit Ratio (%)	Data Hit Ratio (%)	Index Hit Ratio (%)	Average Asynchronous Read Time (sec.ms)	Average Asynchronous Write Time (sec.ms)	Average Data Pages Read per Asynchronous Request	Average Sectors Direct Reads
member_id	pool_hit_ratio	data_page_hit_ratio	ipool_hit_ratio	avg_async_read_time	avg_async_write_time	avg_data_rpa_sync_req	avg_direct_
0	2	1	12	0	0	0	

Figure 5-48 PWH process buffer pool output report

Note: If any problems were encountered generating the report, an additional text file named REPORT.log is created detailing the error.

Using Rules of Thumb

There are four public Rule of Thumb (RoT) clusters provided with the warehouse. The one we are interested in here is the DB2PM.Snapshot.Buffer pool cluster. This cluster has five RoTs:

- Percentage ratio of asynchronous physical read I/O to synchronous I/O
- Percentage ratio of asynchronous physical write I/O to synchronous I/O
- Percentage buffer pool hit ratio for data and indexes
- Percentage buffer pool hit ratio for data
- Percentage buffer pool hit ratio for indexes

Similar to creating and executing the report Public Processes, the RoTs need to be copied to your own group before you can edit them. The RoTs can be executed as is or after editing. Do the following:

Select **System Overview** → **Performance Warehouse - Expert**, right-click **Rule-of-Thumb Groups** and select **Create,** enter the **Name** "PEDEMO RoT Group" and a **Description**, and click **OK**.

Under the public **Rule-of-Thumb Clusters**, right-click **DB2PM.Snapshot.Buffer pool** and select **Copy**. Accept the default of the newly created PEDEMO RoT Group and change the RoT cluster name to PEDEMO RoT Cluster. Click the RoT under PEDEMO RoT Cluster; the five buffer pool Rules of Thumb are shown in the right pane.

Review the existing PEDEMO RoT Cluster properties by right-clicking **PEDEMO RoT Cluster** and selecting **Properties** → **Definition**. The basis or core of the SQL that is issued against the PWH is the arithmetic expressions that are defined by the RoT in the cluster. In the Definition tab, you can add columns that will be displayed in the result matrix (see Figure 5-49 on page 319).

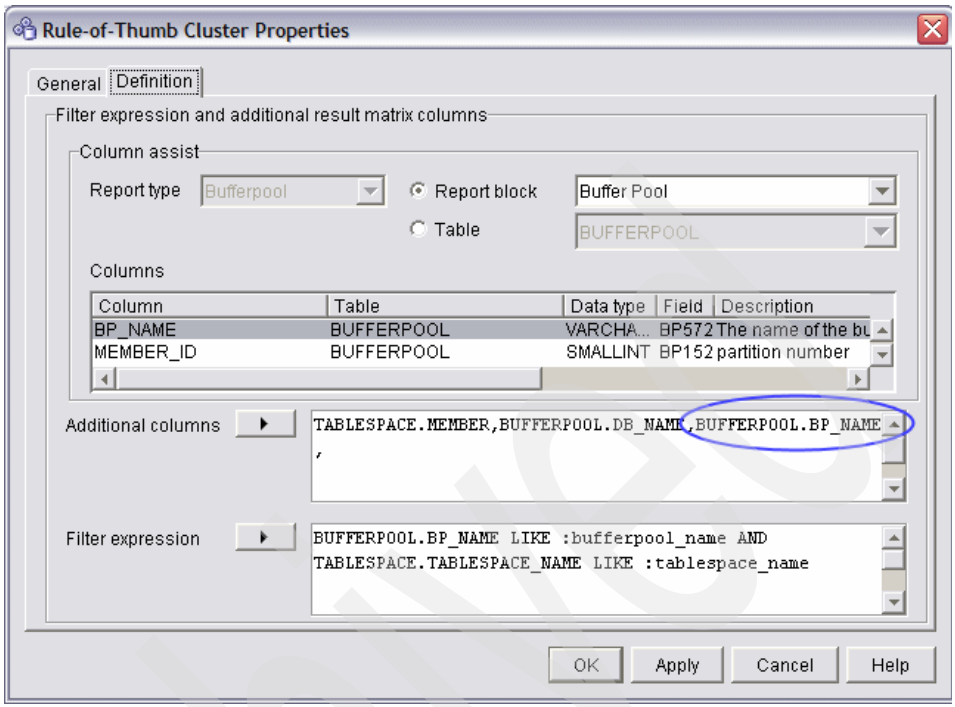


Figure 5-49 Rule-of-Thumb cluster properties

The next step is to prepare and run a single Rule of Thumb. This example runs a analysis for the buffer pool hit ratio for data. In the left pane, expand **PEDEMO RoT Cluster** and select **Rules of Thumb**. In the right pane, right-click **BP hit ratio data** and select **Properties** → **Definition**. Here you can see more of the SQL, including the thresholds for Warnings and Problems. The SQL and the thresholds can be changed in this panel (see Figure 5-50).

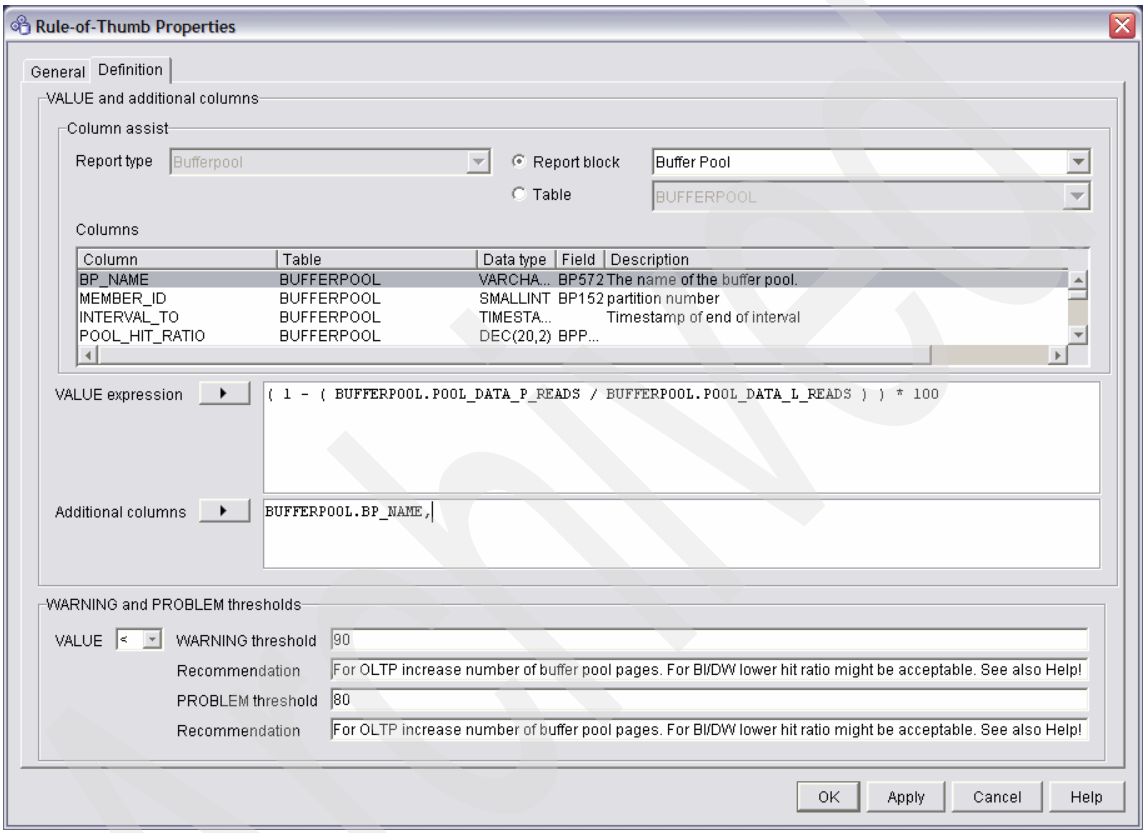


Figure 5-50 Rule of Thumb properties

Click **OK**, right-click **BP hit ratio data**, and select **Analyze** → **Options**.

On this screen shown in Figure 5-51 on page 321, you enter the variables that are passed to the predicate “host variables” at execution time. The date range for the report is entered. Care needs to be taken here, as the user has control over the SQL syntax of the query. In Figure 5-52 on page 322, the Start time and End time are added to the query. This is done in several steps:

1. Click **Start time** and edit the inserted date to become the required start date and time.

2. Click **End time** and edit the inserted date to become the required start date and time.
3. Click **Apply**.
4. Update the SQL by adding two “and” conjunctions between the predicates, as shown in Figure 5-52 on page 322.
5. Click **Analyze**.

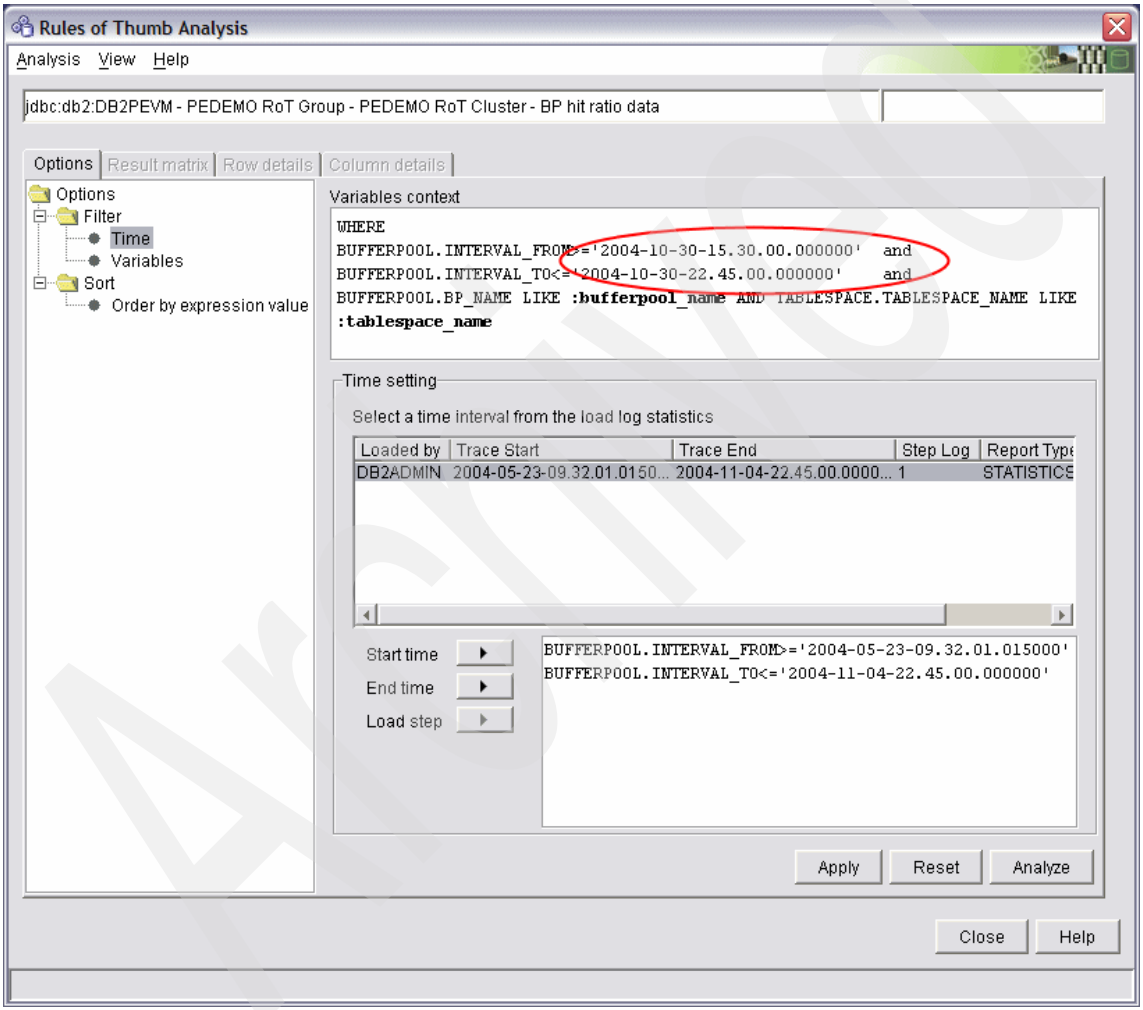


Figure 5-51 Set date ranges for PWH RoT reports

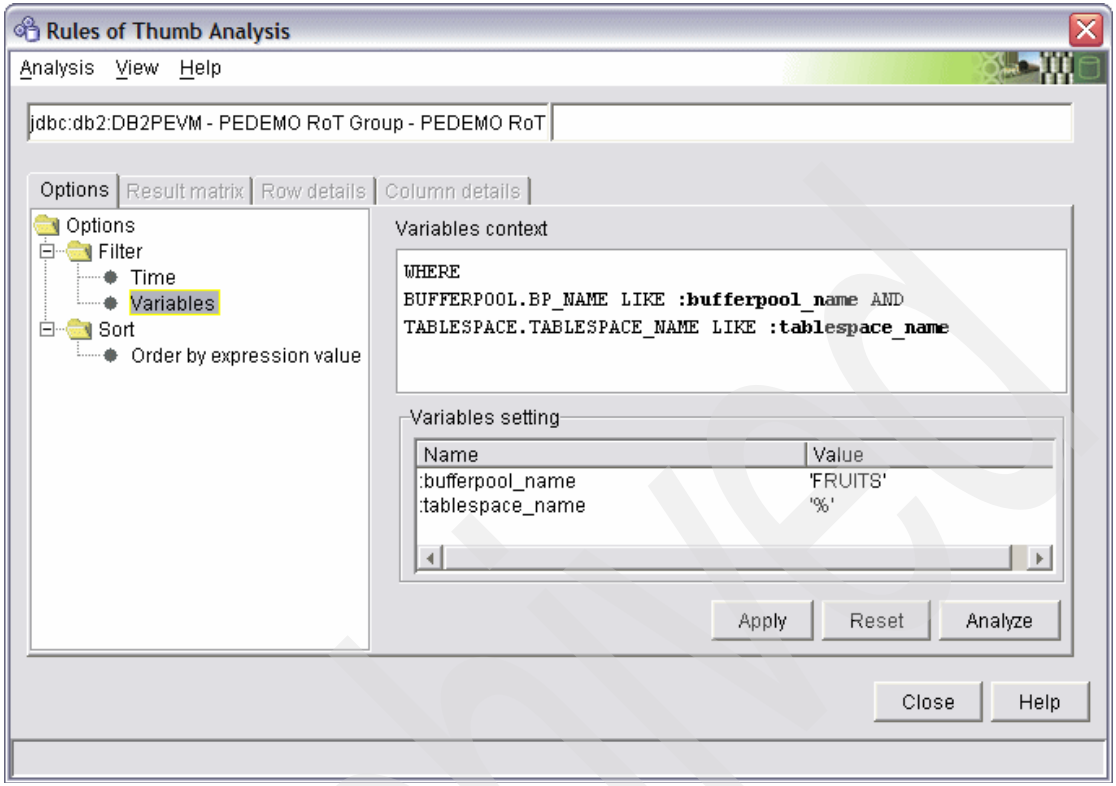


Figure 5-52 Set variables for BPA RoT Analysis

Finally, you are ready to run the report; click **Analyze** to generate the RoT results. These are displayed in a tabbed page (see Figure 5-53 on page 323).

Each RoT can be executed individually, or the cluster can be executed, which will run all the member RoTs using the same time and variable values.

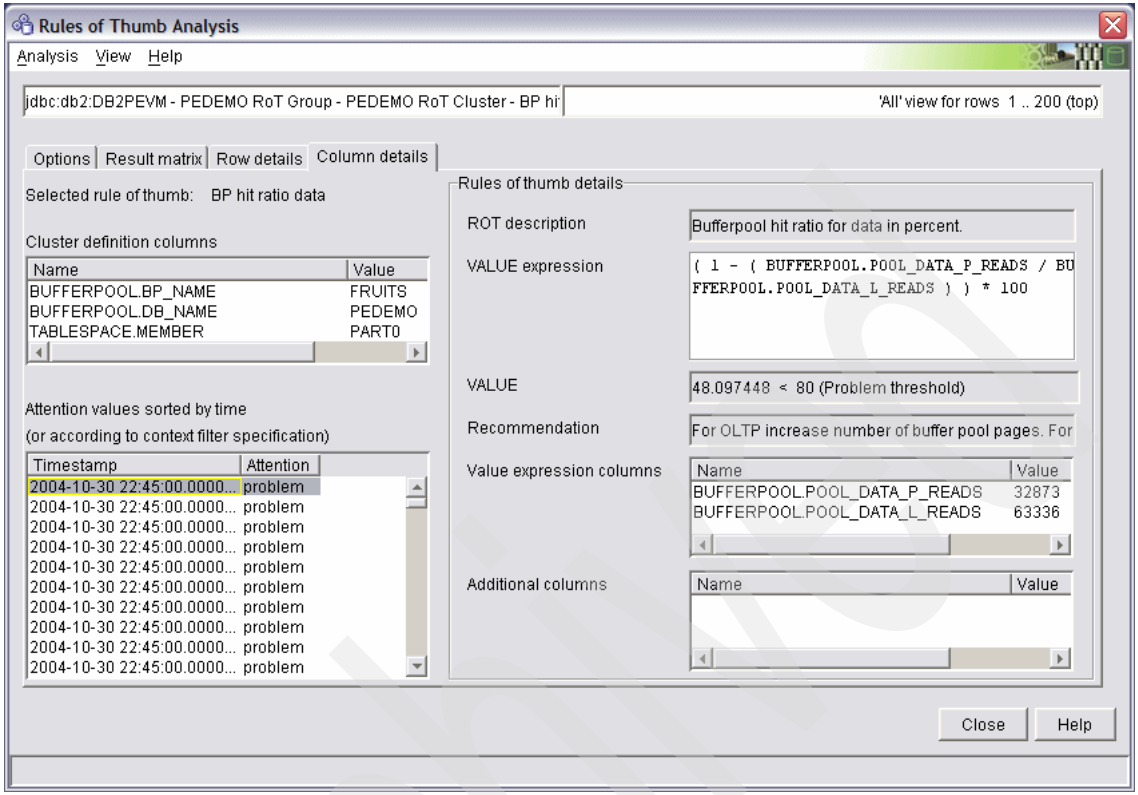


Figure 5-53 RoT analysis results page

Using queries

There are 38 predefined queries provided in the warehouse; of these, there are five that we are interested in for buffer pool analysis. Again, the public queries have to be copied into your own Query group in order to edit them. The first task is to copy the public queries for buffer pool analysis to your own group:

Select **System Overview** → **Performance Warehouse - Expert**, right-click **Query groups** and select **Create**, enter the **Name** “PEDEMO Query Group” and a **Description**.

Next, copy across the predefined query templates. Expand **Public** queries and click **Queries** to list all the predefined queries in the right pane. In turn, right-click each of the five buffer pool queries in the right pane, which are the first five in the list. Click **Copy** and accept the default (the New query group name that you just created). Give the query a more meaningful name if required, and click **OK**. This adds all the buffer pool queries to your own group and enables them to be edited

prior to execution. The public queries cannot be edited, although you can supply time and object variables before executing.

You create your own queries by either copying and using the supplied queries as templates or right-click **Queries** and click **Create**.

Customizing and executing the queries are similar to the RoTs. However, there is no concept of the “cluster”, so the Properties and Analysis functions are accessed and customized directly by right-clicking the specific query.

The user has far more autonomy over the warehouse queries, and therefore responsibility for the SQL syntax. For example, when the Start time and End time are “Applied” in the Query Execution screen, the time predicates are inserted at the cursor position in the SQL. The user must ensure the cursor is at the appropriate position in the predicate clause, and then insert the “and” conjunctions.

Some sample output from a customized query is shown in Figure 5-54 on page 325.

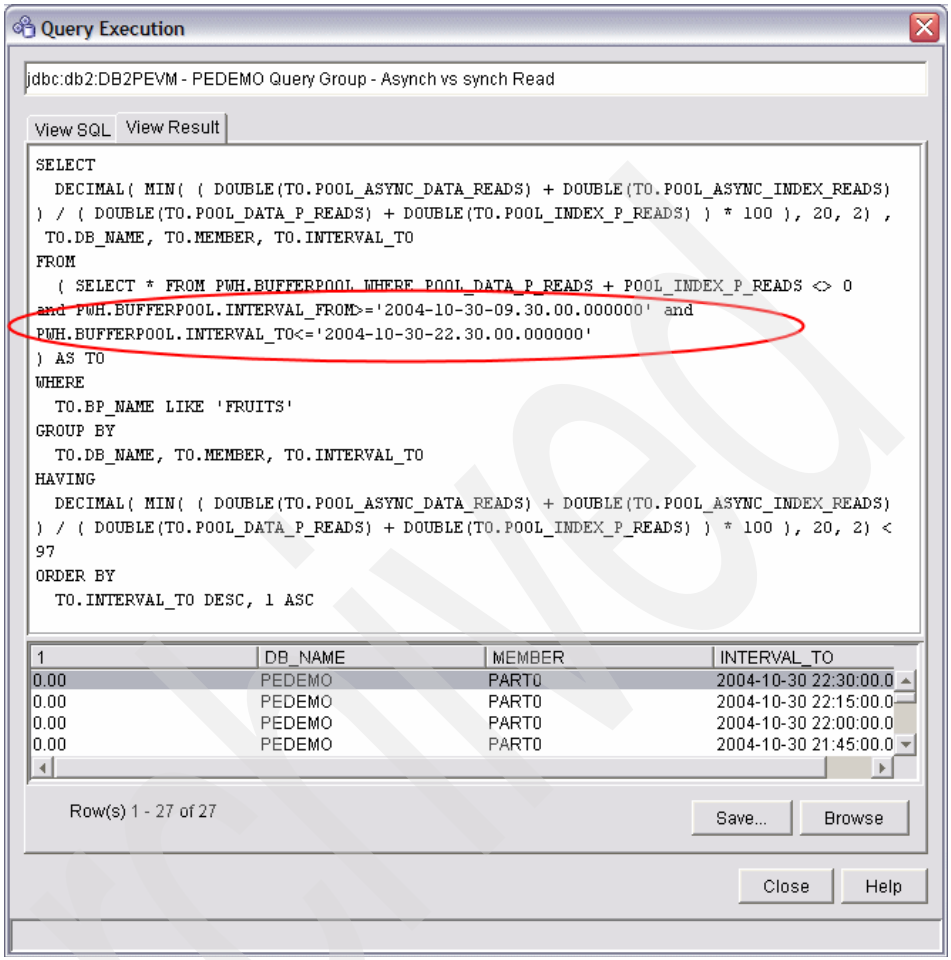


Figure 5-54 Sample output from a tailored query in PWH

Note: Rules of Thumb and Queries cannot be scheduled from within Performance Warehouse. Only Processes can be scheduled.

6

Scenarios

In this chapter, we provide you with some examples of using PE to do database or application performance trend analysis, identify performance problems with various PE functions, and tuning databases and applications using PE.

The scenarios included in this chapter are:

- Buffer pool utilization
- Connection and workload management
- Evaluating the effects of a configuration change
- Identify and fix slow SQL statements
- Responding to Performance Expert alerts
- Transaction log full
- Tuning DB2 for OLTP application with PE

Note: The scenarios in this chapter are all based on PE V2.1 FP1 or FP2, since they were executed before PE V2.2 was available. But for our purposes, the scenarios are still just as valid and have not been changed.

6.1 Trend analysis

In this section, we illustrate the ability of DB2 Performance Expert to help solve and prevent performance problems through trend analysis. This is achieved by using the tool saving key performance data in the Performance Warehouse component and running reports against this collected data and displaying it graphically when required. The functionality and usability of the Performance Warehouse is discussed in detail in 5.1, “Performance Warehouse” on page 240.

6.1.1 Buffer pool utilization

Continuous, efficient use of buffer pools by your application is crucial to maintaining acceptable levels of performance. One of DB2 Performance Expert’s strengths is its ability to record, store, and present a long-term history of buffer pool efficiency. This is discussed in depth in 5.1, “Performance Warehouse” on page 240.

Here we describe how regularly checking the reports available from DB2 Performance Expert can alert you to a possibly deteriorating situation.

A decline in buffer pool efficiency can occur for many reasons. To best illustrate the diagnostic process, a single realistic cause is chosen in order to focus on a specific area.

Description of the application

The main application used for this scenario is the one supplied with DB2 Performance Expert, the PEDEMO application. When this application is used in isolation, it does not provide an obvious trend in buffer pool usage. Therefore, an additional and separate workload is generated within the same database.

Environment configuration

The environment used for this scenario is depicted in Figure 6-1 on page 329.

The machine used for the scenario is a ThinkPad T40 with 2 MB of memory. VMware Workstation V4.5.1 is installed, which acts as a virtual machine running Windows 2000. The DB2 Performance Expert Client is installed on the main machine, or “host”. The DB2 Performance Expert Server and the PEDEMO application are installed within the VMware “guest”. VMware is a useful tool for version testing and regression testing in the development projects. It is also a good environment for a DBA to train themselves using PE. Even though the VMware will not be used in the production environment, the steps show in this scenario are applicable to an environment without VMware.

Contrary to the optimum topology, both the PE Server and the monitored database are in the same DB2 instance.

Both the host and guest are at the level of DB2 UDB ESE V8.2 (Fix Pack level 7a). A smaller DB2 footprint could have been used on the host, for example, DB2 Runtime Client. DB2 Performance Expert has Fix Pack 1 installed for both the client and server.

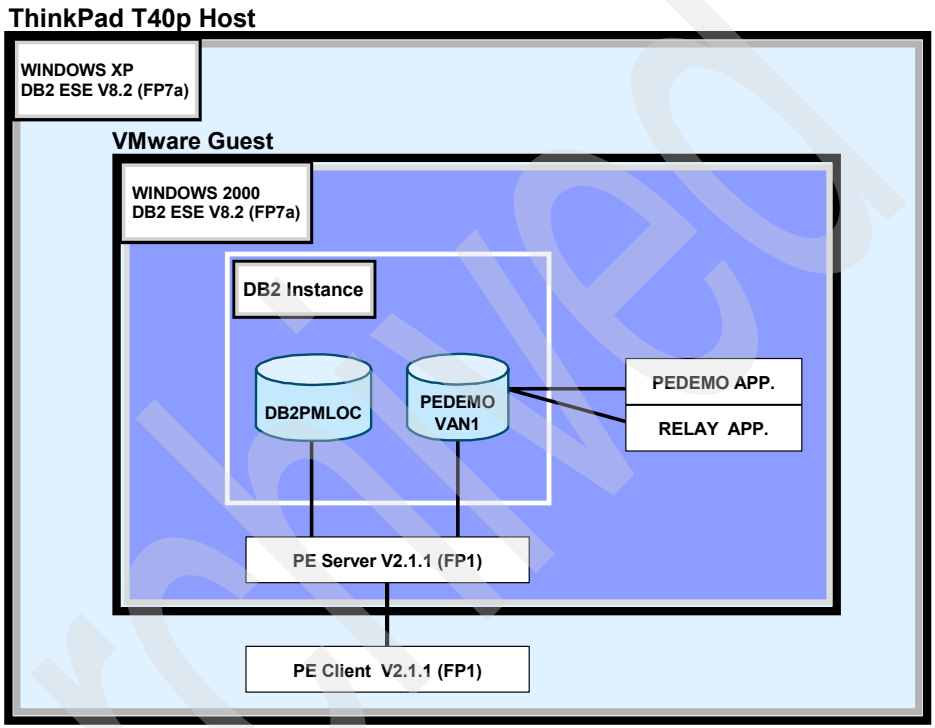


Figure 6-1 ThinkPad configuration for DB2 Performance Expert

Note: DB2 Performance Expert Client and Server and the monitored application can all be installed in a VMware guest. This can enable you to become familiar with all DB2 Performance Expert's features and functions in a separate environment, independent of any other workload or application.

DB2 Performance Expert settings

This scenario is mainly concerned with the Buffer Pool Analysis component of DB2 Performance Expert. Because the buffer pool component derives all its data from the performance warehouse, we need to set the warehouse parameters.

Note: Although we are considering trend analysis which would normally be days or weeks, here we are taking the same measurements at much shorter intervals to compress the scenario time frame.

The granularity for summarizing the history data is set to the minimum of every 15 minutes. The summarization interval is also set to the minimum of one hour. These can be seen in Figure 6-2.

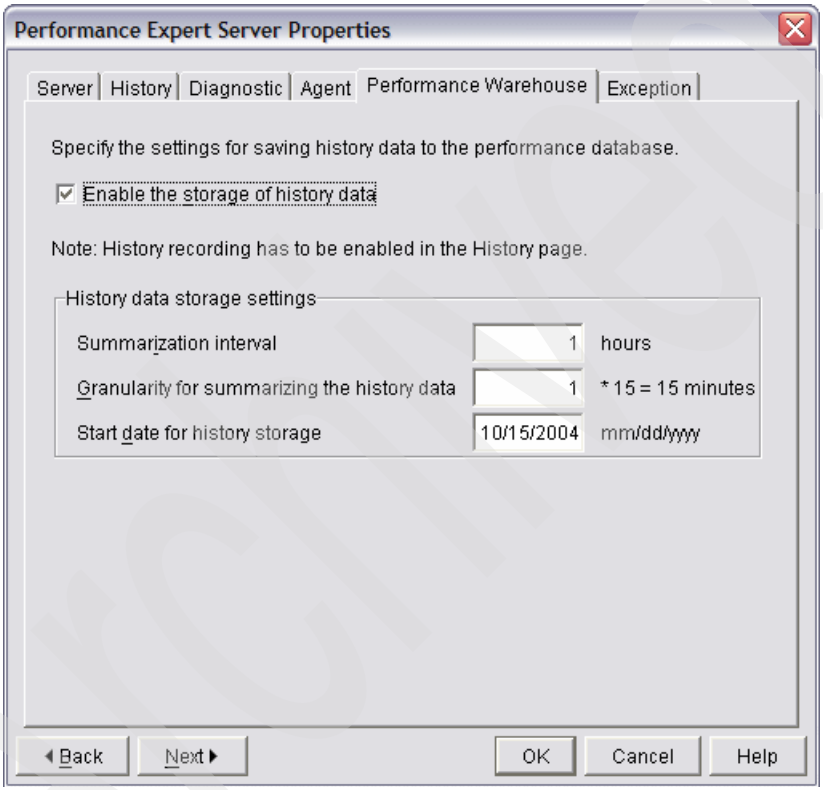
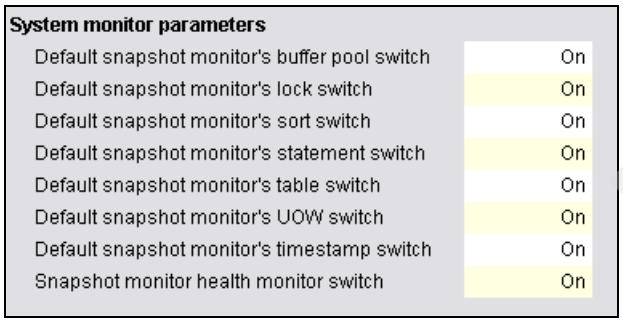


Figure 6-2 PE Server properties - Performance Warehouse page

All eight of the DB2 Monitor settings are switched ON, as seen from System Parameters - Instance, in Figure 6-3 on page 331.



System monitor parameters	
Default snapshot monitor's buffer pool switch	On
Default snapshot monitor's lock switch	On
Default snapshot monitor's sort switch	On
Default snapshot monitor's statement switch	On
Default snapshot monitor's table switch	On
Default snapshot monitor's UOW switch	On
Default snapshot monitor's timestamp switch	On
Snapshot monitor health monitor switch	On

Figure 6-3 Snapshot monitor status from System Parameters - Instance

Workload used

The PEDEMO application simulates an active workload of updates and queries, and is designed to simulate several aspects of performance problems. In the area of buffer pools, it simulates the adverse effect of having a very small buffer pool relative to the volume of data being processed by DB2. This problem becomes apparent very quickly. We have set the FRUITS buffer pool to a more realistic size of 250 4 KB pages to enable it behave efficiently. However, in this scenario, we are aiming to show a trend of deterioration of buffer pool efficiency over time.

This is achieved by adding the following extra objects and components to the PEDEMO database; we refer to this additional load as Relay:

- Buffer pool - MARATHON: Set to a fairly small 25 4 KB pages.
- Tablespace - RELAY: Holds table and indexes.
- Table - VAN1 (see Figure 6-4).

Key	Name	Data type	Length	Nullable
	GENID	SMALLINT	2	No
	ID	SMALLINT	2	No
	"NAME"	CHARACTER	15	No
	ADDRESS	CHARACTER	250	No

Figure 6-4 Van1 table structure

- Indexes:
  - VAN1IX1U: Unique on column GENID asc
  - VAN1IX2C: Clustering on column “NAME” asc

- Small Java application: Listed in Example 6-1.

Example 6-1 Java application

```
//db2 CREATE TABLE PEDEMO.VAN1
//   ( GENID SMALLINT  NOT NULL  GENERATED ALWAYS AS IDENTITY
//       (START WITH 1, INCREMENT BY 1, NO CACHE ),
//     ID SMALLINT  NOT NULL ,
//     NAME CHARACTER (15)  NOT NULL,
//     ADDRESS CHARACTER (250)  NOT NULL   ) IN RELAY ;
//path/java Van1ins n db ID pwd
import java.sql.*;
public class Van1ins
{
   public static void main(String[] args)  throws Exception
   {
   System.out.println("Hello World!");
   Class.forName("COM.ibm.db2.jdbc.app.DB2Driver");
   Connection
con=DriverManager.getConnection("jdbc:db2:"+args[1],args[2],args[3]);
   PreparedStatement prep=con.prepareStatement("insert into pedemo.van1
(ID,NAME,ADDRESS) values(?,?,?)");
   con.setAutoCommit(true);
   for(int i=0;i<Integer.parseInt(args[0]);i++){
      prep.setInt(1,i);
           prep.setString(2,"name"+i);
      prep.setString(3,"address"+i);
      prep.execute();
      System.out.println("inserted product numbered "+(i+1)+" ");
        }
    System.out.println("Press enter to exit and close all connections");
    System.in.read();
    con.commit();
    }
}
```

The Java program Van1ins inserts blocks of rows into table PEDEMO.VAN1. This enables the simulation of the growth in size of the VAN1 table. One of the parameters used when invoking the program is the number of rows to be inserted. Initially, the table VAN1 is populated with 2000 rows.

Each time period, in this case 15 minutes, a further 2000 rows is inserted. The GENID column remains unique, as it is defined as a “Generated Always” identity column.

After the initial insert block, the table was reorganized and runstats was run using DB2 commands in DB2 Command window. The REORGCHK output in

Figure 6-5 shows the table data is perfectly clustered according to the clustering index.

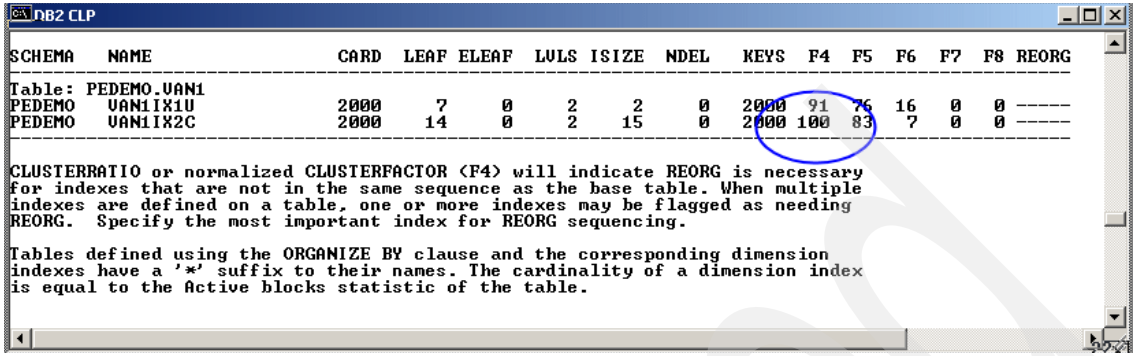


Figure 6-5 Initially VAN1 table is clustered

Also, we can see in Figure 6-6 that the clustering index VAN1IX2C is used for the queries run against VAN1.

A typical query, explained here, that is run against the VAN1 table is:

```
select id, name, address from pedemo.van1 where name like 'name22%';
```

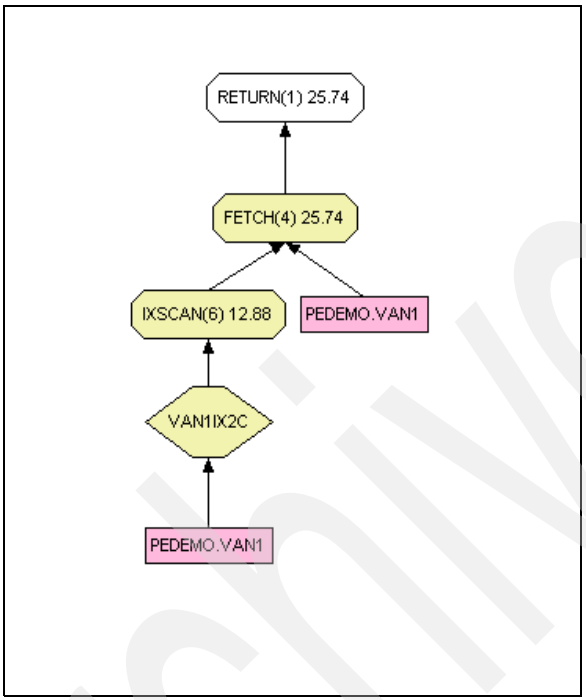


Figure 6-6 Visual explain output of typical Van1 query

Investigating the problem

The nature of this scenario is not that a problem has occurred, but that you are able to spot a downward trend that may eventually result in a problem, or have significant deterioration of performance that is apparent to the end user.

When setting up this scenario, it was easier to view the progress using a Data View to get near-instant feedback instead of running reports. The early stage of the buffer pools reaching a “steady-state” is graphed in the Data View in Figure 6-7 on page 335.

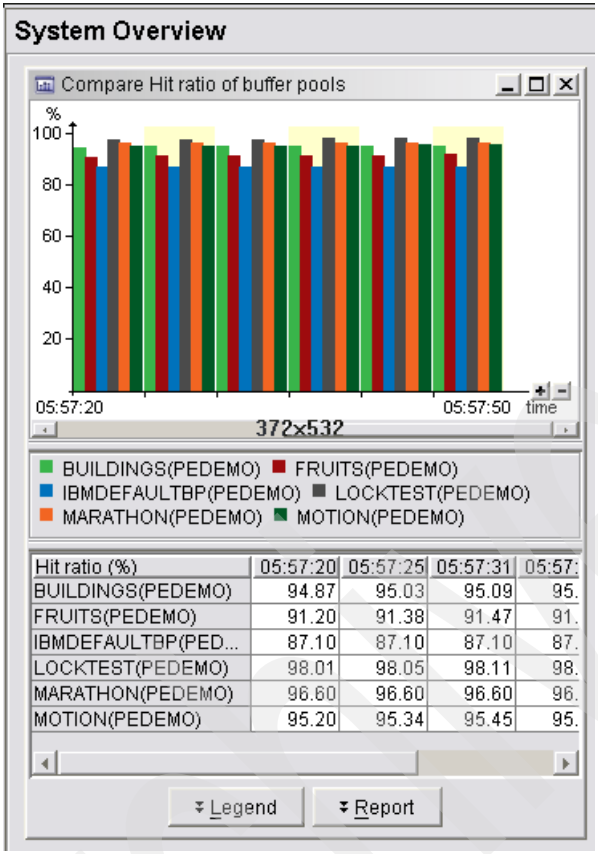


Figure 6-7 Buffer pools reaching equilibrium

Identifying a trend in data is clearly easier to identify when presented graphically rather than in raw tabular format. The Interactive Reports that are written as XML and can be viewed graphically from the Buffer Pool Analyzer is the method used. Subsequent analysis to uncover more detail can be achieved by running processes and queries from the performance warehouse.

Root cause of the problem

The SELECTs against the VAN1 table are stable and consistent and initially show an efficient use of its MARATHON buffer pool. As data is inserted into the VAN1 table using the Java insert program and the table grows, the data becomes unclustered.

Buffer pool hit ratio percentage is reduced in the Data View. Eventually, as the data becomes unclustered, more data needs to be accessed per select statement and will reduce the efficiency in terms of increased disk I/O at the expense of reads from the buffer pool, also known as logical reads. In Figure 6-8, the MARATHON buffer pool is shown as the shortest bar in the chart and is getting shorter with each set of queries run. A typical query that is run against the VAN1 table is:

```
select id, name, address from pedemo.van1 where name like 'name22%';
```

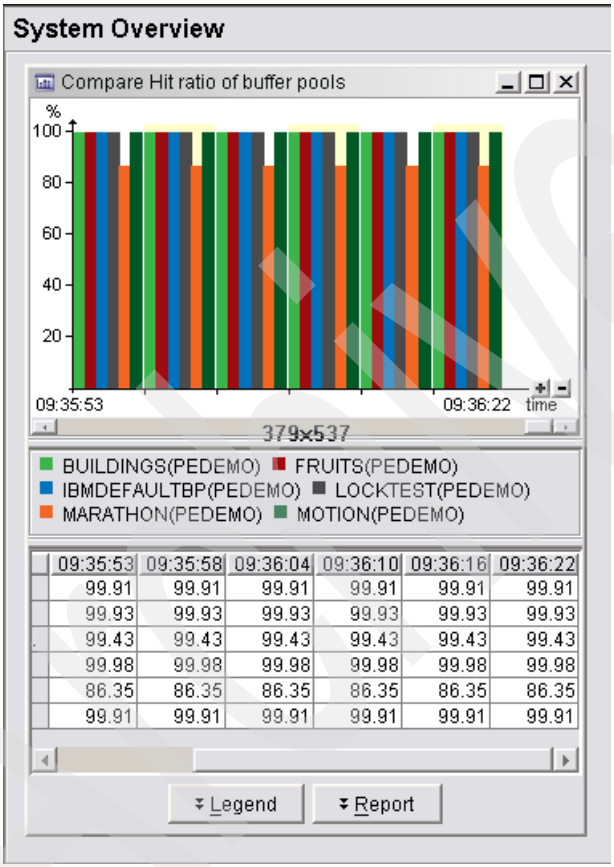


Figure 6-8 Deteriorating hit ratio of MARATHON buffer pool

Additional verification is illustrated using REORGCHK. The final state of the Van1 table is shown in Figure 6-9 on page 337. The F4 cluster factor is reduced from 100 to 7 and the * in the REORG column indicates that the data needs reorganizing.

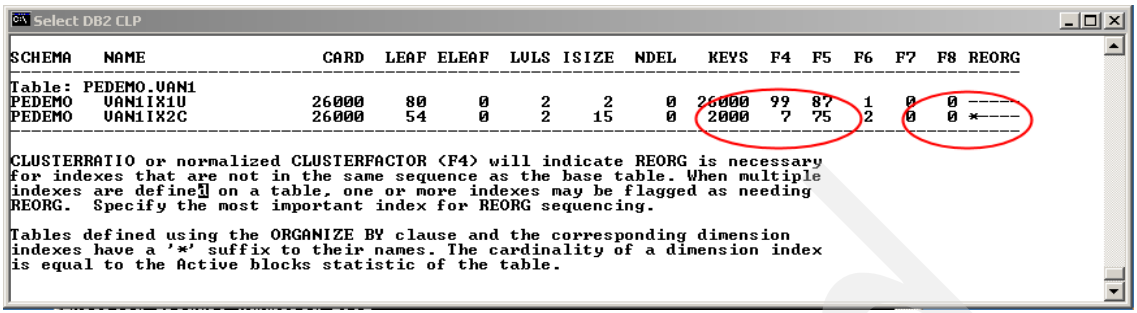


Figure 6-9 Final REORGCHK showing unclustered data

In spite of the unclustered data, looking at the output from Visual Explain in Figure 6-10 shows that DB2 is still using the clustering index for access.

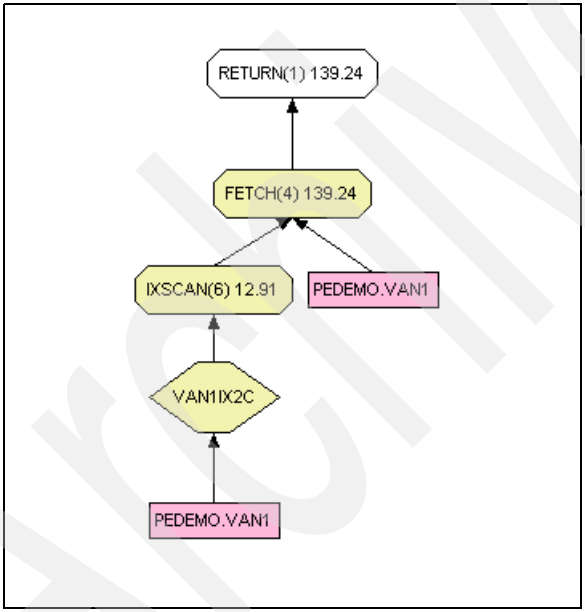


Figure 6-10 Visual explain output after mass inserts

Trend analysis

So much for the theory of clustering and access paths; we need to look at the reports from the Buffer Pool Analyzer to spot trends. The BPA Text Report in Figure 6-11 shows the decline in Hit Ratio over the active period.

Buffer Pool Name: MARATHON

Buffer Pool Characteristics

Buffer Pool for Database : MARATHON(PEDEMO)
Buffer Pool ID : 6

Partition ID	Snapshot Time	Hit Ratio (%)	Data Hit Ratio (%)	Index Hit Ratio (%)	Average Asynchronous Read Time (sec.ms)	Average Asynchronous Write Time (sec.ms)
member_id	interval_to	pool_hit_ratio	data_page_hit_ratio	ipool_hit_ratio	avg_async_read_time	avg_async_write_time
0	05:45:00.0	0	0	0	0	0
0	06:00:00.0	99	98	99	0	0.002345
0	06:15:00.0	99	99	99	0	0.001435
0	06:30:00.0	97	95	99	0	0.00191
0	06:45:00.0	96	91	98	0	0.000382
0	07:00:00.0	95	88	98	0	0.000884
0	07:15:00.0	89	74	97	0	0.000212
0	07:30:00.0	93	83	98	0	0.00595
0	07:45:00.0	93	83	98	0	0.001008
0	08:00:00.0	92	78	98	0	0.000427
0	08:15:00.0	92	80	98	0	0.000001
0	08:30:00.0	90	74	98	0	0.000611

Figure 6-11 BPA text report for MARATHON

On inspection of the Buffer Pool Analysis report in the Text format, it is apparent that the hit ratio is decreasing. However, it is easier to see the degradation in graphical format. The next three figures show the XML interactive version of the Buffer Pool Analysis reports.

Although this is graphed over a short time period of a few hours, the declining trend is apparent. The MARATHON buffer pool is mapped as the lowest line on the chart (Figure 6-12 on page 339). To investigate further, it is worth splitting this down into Data and Index hit ratios (see Figure 6-13 on page 340).

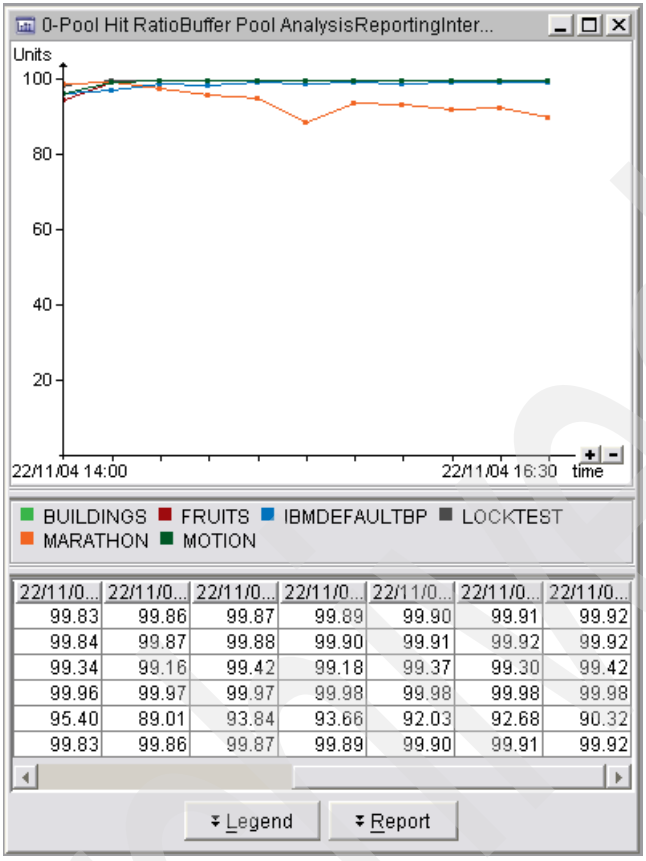


Figure 6-12 BPA XML report for BP hit ratios

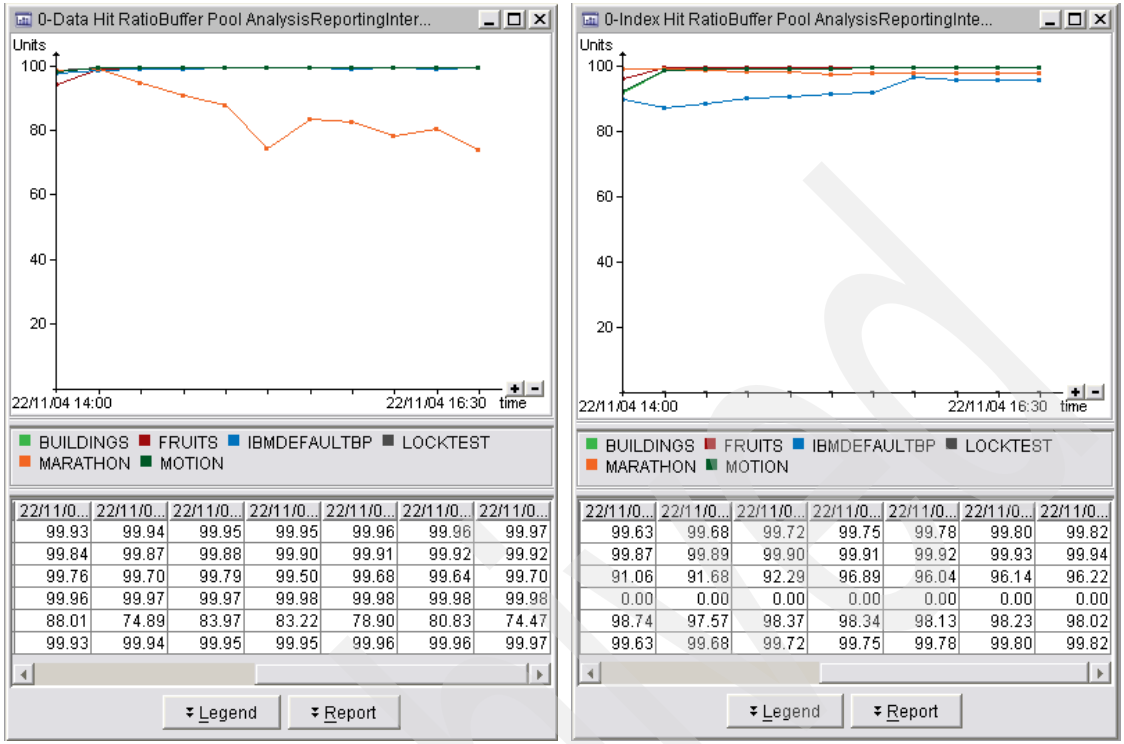


Figure 6-13 Comparison of Data hit ratio and Index hit ratio

In both graphs, the data points in the Report for the MARATHON buffer pool is the penultimate row of data. So it would appear that the clustering index has remained in good shape while the data clustering has deteriorated significantly. As the REORGCHK confirmed, a reorganization and runstats is required.

The significance of DB2 Performance Expert's role in this scenario is that even if an alert had been set up, the Interactive Reports can indicate a trend that requires corrective action.

6.1.2 Scenario: Connection and workload management

Trade3 Creating connections in RDBMS is one of the most time and CPU consuming activities and it can really degrade RDBMS performance. In DB2 UDB, every time a new connection is created, the agent is created; this agent can be quite CPU consuming, especially in environments in which applications are continuously connecting and disconnecting from the server.

Configuring the connection constraints properly is one of the key performance aspects. DB2 Performance Expert helps DBA in this aspect by providing a quick

trend analysis for DB performance statistics and helps you configure connection related parameters to maintain a healthy performance.

Background

Database connections should always be carefully managed. The connection related instance and database variables should only be set after a thorough check and planning as it may impact your system performance very badly. In many current day applications, especially Web applications, the connections are acquired for a very limited amount of time. Each user/application thread acquires a large number of connections and releases them after a short period. In such applications, the cost of creating and deleting connections and agents can be high. To streamline the overhead caused by these costly operations, the DB2 UDB supports agent pooling, and most of the current day applications use or implement connection pooling at their end.

By utilizing a specified set of connections and pooling them between multiple sessions or database clients, developers can increase the scalability and extensibility of their applications. If many users are connected to the DBMS, but only a few of them are actually active, then limits on the maximum number of connections, coordinating agents, and pool agents can reduce memory utilization.

DB2 agents

DB2 agent (db2agent) is thread or process created to service all connection requests from a client, irrespective of whether the client is remote or local. This coordinating agent assigns the work to connection coordinates on behalf of the connection and communicates with other agents. In a multiple partitioned database environment, the data may be distributed over multiple nodes. The coordinator agent distributes the database requests to subagents (db2agntp), which is assigned for each node the data is distributed on.

Agent pooling and connection pooling

The process of establishing and tearing down a connection is very costly. To solve the problem of a large volume of small transactional activities and high connection creation and deletion, a resource utilization technique called pooling is used, which allows reuse of an established connection infrastructure for subsequent connections.

In the case of DB2 agents, an initial pool of coordinating agents is created when an instance is started. When a connection request comes, an agent is assigned to the connection. This agent then connects to the database. When a disconnect request for this connection is issued, the agent is disassociated, but the connection to server is not released. If another connection request comes, this agent can be assigned.

On the other hand, application connection pooling is a connection pooling feature supported or used by most current day applications and application servers. The main difference between agent pooling and connection pooling is that with application pooling, all the connections use the same user ID, password, and isolation levels. The agent pooling and agent pooled connections is completely application, machine, and user independent.

In this scenario, we will not discuss connection pooling, but will concentrate on agent pooling only.

Scenario description

In this scenario, we cover DB2 connection related issues and setups, with the help of a Web based application. Now, considering this application, we can use DB2 Performance Expert with respect to connection constants at various levels:

- Configuring DB2 connection

  In this section, we will configure database and instance configuration parameters for optimized connections and agents usage. In this section, we will run our application with DB2 Performance Expert, analyzing the application for optimizing the agent's usage.

- Ongoing overload problem analysis or bottleneck tracking

  Once your application is deployed and used by the users, continuous monitoring and analysis of the database for the connection data is suggested. The objective of this monitoring is to collect the information about connections on the instance during the normal, slow down, or peak periods to analyze and tune systems to keep the connect performance goal. In this section, we look into a continuously changing workload scenario and analyze it.

- Applications run-time problem analysis

  In this section, we will analyze the applications using DB2 Performance Expert for finding out application problems. In this section, we will see how Performance Expert can help us find application problems and fix the application source.

Scenario prerequisites and setup

The Trade3 database and application is used to recreate this problem scenario. The Trade3 setup should be completed prior to starting this analysis. To create a scenario of multiple connections, we use Web Performance Tool (WPTool), configured for a minimum run time but for maximum connections. This means we do not want to analyze Web Application performance, but want to create as many connections as we want.

So in our scenario, two clones of Trade3 sample application are installed on WebSphere Application Server on two Win XP machines. Both are configured to

use same Trade3 database on AIX server. The topology for this scenario is shown in Figure 6-14.

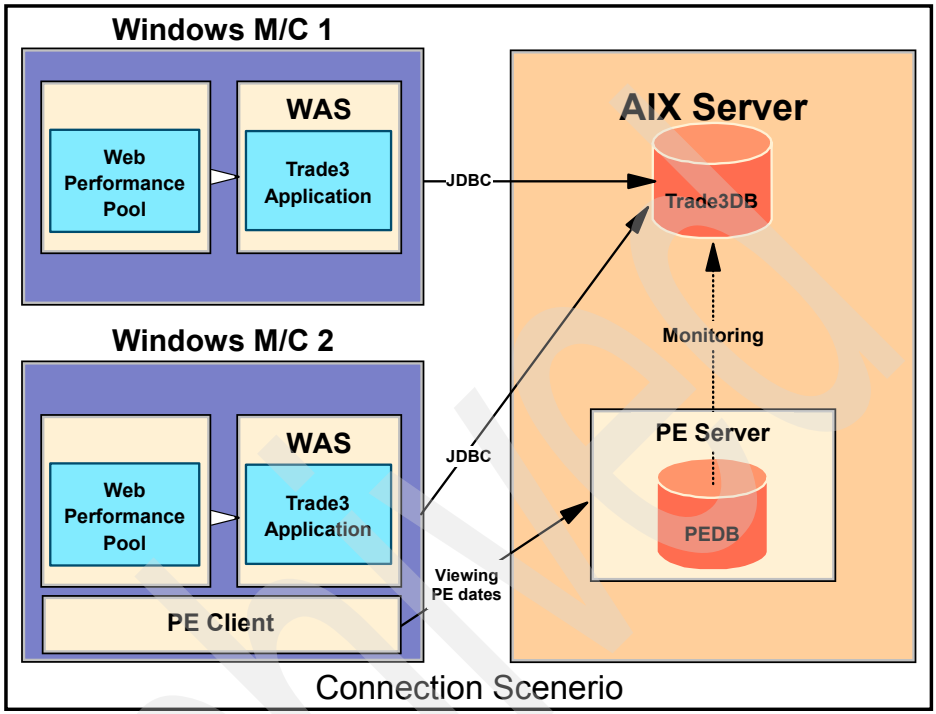


Figure 6-14 Connection scenario topology

The WebSphere Application Server data source settings are:

- Connection timeout: 1800 seconds
- Max Connections: 300
- Min Connection: 1
- Reaptime: 180 Sec
- Unused Timeout: 1800 Sec

The Web Performance Tool settings are:

- Clients: 100
- ThreadsPerClient: 1
- Connection: keep-alive
- KeepAlive: 300

Configuring DB2 connections

A few connection specific DB2 database manager and database configuration settings affects DB2 database performance tremendously. These configuration parameters can lead to excessive memory requirements that your server may not be able to support. They may also lead to slow connection or reduce the number of users that can access your application concurrently. DB2 Performance Expert can help you monitor the effects of these configuration parameters.

- Database manager configuration parameters

  The DBM configuration parameters that influence the connection are:

  - Maximum number of client connections (max_connections)

    This parameter indicates the maximum number of client connections allowed per partition. So this is the parameter that restricts the number of applications that can use the database. If the value of this parameter is equal to max_coordagents, each agent gets its own private memory and share resources, such as buff pool and so on, with other agents. If you do not intend to use the connection concentrator, you should keep the value of max_connections equal to max_coordagents.

  - Maximum number of agents (maxagents)

    This parameter indicates the maximum number of database agents that can be a coordinating agent (defined by max_coordagents) or subagent, and are currently available to cater the connection request. This parameter is very critical, as it impacts the memory usage on your server and the maximum number of connections a database can provide at certain time. This number should be provided on the basis of the target database load you are expecting for your database and the target user workload you are expecting on your application server.

  - Maximum number of concurrent agents (maxcagents)

    This parameter is critical during peak hour database activities, as it restricts the number of agents that can be executed simultaneously. You can restrict your memory usage at peak times using this parameter.

  - Maximum number of coordinating agents (max_coordagents)

    As we know, one coordinating agent is acquired for each local or remote application that connects to a database or attaches to an instance. An application is in an active state only if there is a coordinator agent servicing it. Otherwise, the application is in an inactive state. Requests from an active application will be serviced by the database coordinator agent. Requests from an inactive application will be queued until a database coordinator agent is assigned to service the application, when the application becomes active. As a result, this parameter can be used to control the load on the system in a same way as maxagents.

- Database configuration parameters

  Database configuration parameters which influence the connection are:

  - Maximum number of active applications (maxappls)

    This parameter specifies the maximum number of local and remote applications that can concurrently access the database. It is recommended that the maxappls should be defined on the basis of potential number of users and number of nodes for your system. Also, while changing maxappls, the locklist should be kept in mind.

DB2 Performance Expert can be used to effectively manage the database connections and all the above configuration parameters. In this scenario, we started a database with very high parameter values and then lowered values for all the above parameters. This is a very bad configuration, as this can potentially cause memory overload, which is very bad for a production server, but here we are using these values to obtain the optimum values for the above parameters.

After setting the above discussed parameters to “large values”, the application was run for expected workload. For example:

```
max_connections=400
maxagents=400
maxcagents=400
maxappls=400
```

Both Web Performance Tool opens 50 Web applications for each Application Server, which means a total of 100 connections will be opened concurrently.

Now open Performance Expert Client and open Statistics Detail's Instance information pane, as shown in Figure 6-15.

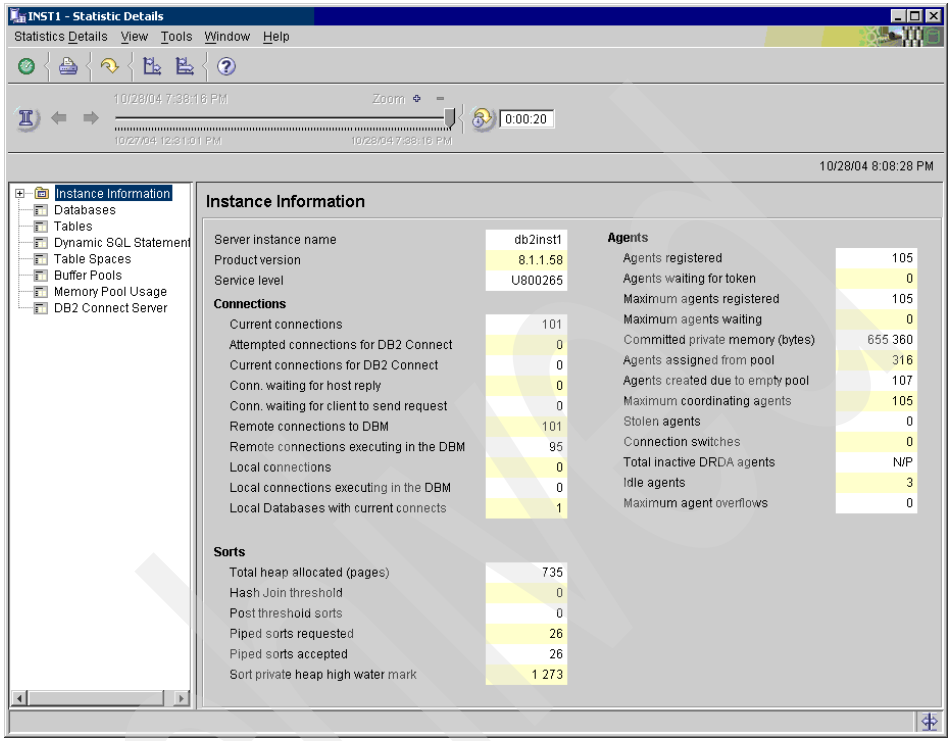


Figure 6-15 DB2 PE Connection details

Watch out for the following fields during application (Trade3) execution and after completion of WPTool execution:

- Current Connections

  In a resource constrained environment, this variable can be quite useful for setting the required values for the maxappls database manager configuration. Monitor the value for this gauge in your snapshot history to find the maximum range for your application user's number.

  Also, this parameter should be used to set the max_coordagents value in the database manager configuration. max_coordagents should be set greater than max(Current Connection) + max(local connections). Make sure that maxappls is not set higher than the values your system can handle, because a very high value for this can lead to over flooded agents and connections.

  In our scenario, we see that the current connections value reaches 101, which includes our 100 application users connections. So in this case, setting

maxappls to 120 and max_coordagents to 120 should be more then sufficient. We will execute Run 2 using these values.

- Remote Connection to DBM

  This parameter shows the current number of connections initiated from remote clients on the database manager that is being monitored. This parameter should be used to set the value for the max_connections parameter; max_connections should be set greater than sum(max(Remote Connection to DBM)+max(Local Connection)).

  In our scenario, we see this value reaches 101, as all our connections are remote connections. We change the value of max_connections to 120 for Run 2.

- Local Connection

  It shows the number of local applications that are currently connected to a database within the database manager instance being monitored. This parameter should be used to set a value for the max_connections parameter; max_connections should be set greater than sum(max(Remote Connection to DBM)+max(Local Connection)).

  As we do not have any Local Connections, we are not really concerned with this parameter, but in your case, you may have to consider this value for setting max_connection.

- Agents registered

  This monitor is the most important monitor to watch for. It shows the number of agents registered in the monitored instance. This value can be used to set maxcagents and maxagents in database manager configuration. Agents registered should be less than maxcagents and maxagents, even in a resource constrained environment. A higher Agents registered value is an indication of possible connection failures.

  This value is 105 in our case, which includes all the coordagents and subagents. Considering this value, we set maxcagents and maxagents to 120 for our Run 2.

- Agents waiting for token

  Each application has its own dedicated coordinating agent. To execute a transaction, the application need to get a token, which represents access to a concurrent agent. This gauge shows the number of agents waiting for a token so that they can perform a transaction on the database. This parameter can be used with the Agents registered counter to determine the percentage of sleeping agents. If the percentage is too high, it shows poor concurrency, so you should increase the value for maxcagents.

As we have a sufficient agent pool defined for our database manager value for agents, waiting is always zero. In case you get more frequent non-zero values for this parameter, increase maxcagents.

- Maximum agents waiting

  This indicates high water mark agents waiting for token reached, which represents the number of concurrent connections your applications are seeking. If your resource permits, raise the value for *maxcagents* if the Maximum agents waiting value is high. In our scenario this values stay at zero so no action is required.

- Agents assigned from pool and Agents created due to empty pool

  Agents assigned from pool indicates the number of agents assigned from the agent pool and *Agents created due to empty pool* shows the number of agents created because agent pool does not have any agents to cater to this request. These two counters can be used to determine how often an agent must be created because the pool is empty. If the ratio of these two counters is high, it might indicate that the maxagents database manager configuration parameter should be increased. A low ratio suggests that maxagents is set too high, and that some of the agents in the pool are rarely used and are wasting system resources.

- Maximum coordinating agents

  This value specifies the highest number of coordinating agents that existed on a DB2 instance. One coordinating agent is required for each local or remote application that connects to the database or is attached to the instance. This counter can be used to watch the load on the system. It can be directly used to set max_coordagents.

  If the value of Maximum coordinating agents seems to be quite consistent to you, you may consider changing num_initagents to a value equal to the consistent value.

  This value is 101 in our case. Considering this value, we set maxcagents to less than 105 for our Run 2.

- Stolen agents

  Stolen agents show the number of times that agents are stolen from an application. Agents are stolen when an idle agent associated with an application is reassigned to work on a different application. If this counter frequently shows non-zero values, you need to increase maxcagents, as this indicates forceful access to tokens.

- Idle agents

  Idle agents is a gauge of the number of agents that are currently unassigned to an application and are therefore “idle”. If the Idle agents field is consistently zero or very low, increase the value of maxagents to minimize the cost of

agent creation and destruction. If the Idle agents field is consistently high, decrease the value of maxagents to avoid wasting system resources.

Run 2

From Run 1, we have derived the proper value of database parameters for our application to have optimal performance and resource usage. We run the same application again to verify the new setting. The following are the values of database configuration:

```
max_connections=120
maxagents=120
maxcagents=120
max_coordagents = 120
maxappls=120
NUM_INITAGENTS=0
NUM_POOLAGENTS=60
```

Running the Web Performance Tool, opening 50 Web applications per application server for users, each using separate connection, will result in the statistics details shown in Figure 6-16.

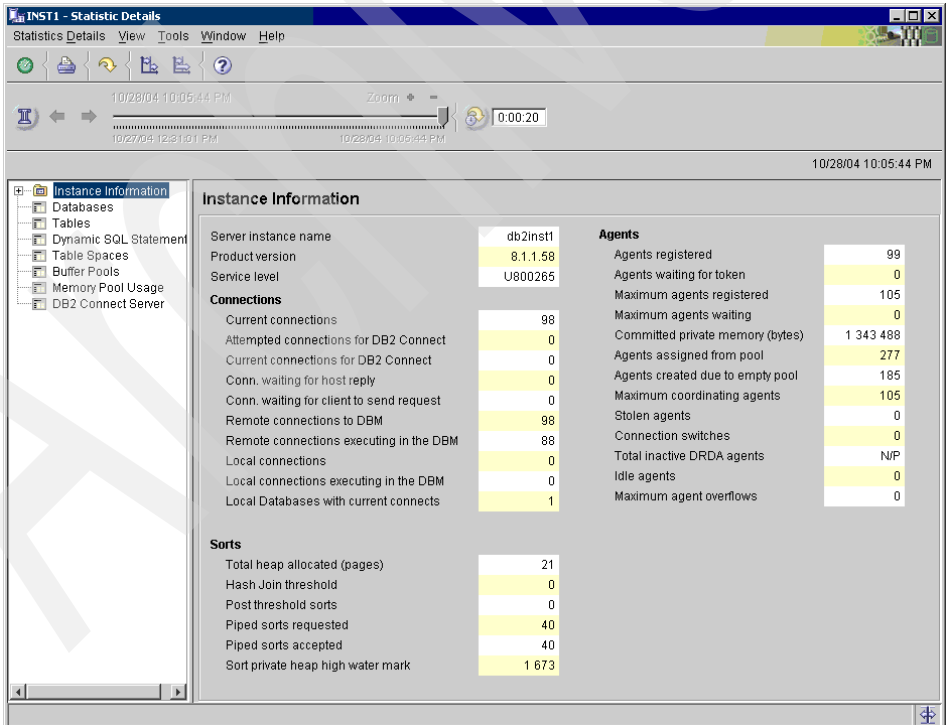


Figure 6-16 Statistics details to view connection constraints

These statistics show that for the workload required here the configuration defined is sufficient to satisfy the application's need.

Overload problem analysis and resolving

DB2 Performance Expert provides monitoring feature to look out for specific events that identify a performance problem. It can also be used to diagnose a database's condition 24x7, which can identify continuous throughput changes or portend connection problems in the near to immediate feature, allowing you to take corrective action. The number of times this kind of analysis is required is not because of any problem, but just to keep up with the increase in demand for your Web application. So the conditions where continuous monitoring can be quite helpful would be in one of the following conditions:

- Increase in server workload from some external sources, for example, the continuous or abrupt changes in the number of application users.
- Increase in server workload because of the failure of one of the servers in a multiserver environment.
- Increase in time consuming tasks, with the result that the connections are returned to the pool after a longer period of time and a report is generated.
- Resource crunch because of other applications on the system.

In this scenario, we analyze our Trade3 applications using the DB2 performance Expert tool. We have the Trade3 application running with the configuration discussed in the previous section. We need to continuously analyze and look for conditions that can halt our system or give us any kind of connection related performance issues.

To create this scenario, we follow the steps defined below:

1. Set up and configure our database for the expected workload.
2. Set up the periodic exception monitor to get feedback on critical issues.
3. Start monitoring connections using DB2 Performance Expert.
4. Continuously change the workload using Web Performance Tool.
5. Analyze the collected data.
6. Make changes on the server before it is too late.

Set up and configure our database for expected workload

The Trade3 database is configured with the required database manager and database configuration parameter. We are expecting up to 500 users to concurrently access our application. The values set for the database and database manager configuration is as follows:

```
max_connections=300
maxagents=500
maxcagents=300
max_coordagents =500
maxappls=300
num_initagents=0
num_poolagents=300
```

Set up periodic exception monitor to get feedback on critical issues

Keeping track of the agents and connections manually at all points of time is not a feasible option. Instead of having the DBA check for the status in statistics information, in other views, or from a snapshot, he expects to be informed when a critical stage is reached. DB2 Performance Expert provides this feature of periodic exception monitor, where you can set up personalized check points on certain gauges for warnings and problems.

In this step, we set up the periodic exception monitor on gauges that we are interested in to monitor connection and agent events. Figure 6-17 show the periodic events set, where we set the following thresholds:

- Maximum assigned agents: Warning Level > 450 and Problem Level > 470
- Agents associated: Warning Level > 450 and Problem Level > 470
- Associated agents: Warning Level > 450 and Problem Level > 470
- Agents stolen: Warning Level > 10 and Problem Level > 20
- Agents created: Warning Level > 270 and Problem Level > 280
- Database Application connection: Warning Level > 250 and Problem Level > 270
- Max. Concurrent: Warning Level > 250 and Problem Level > 270

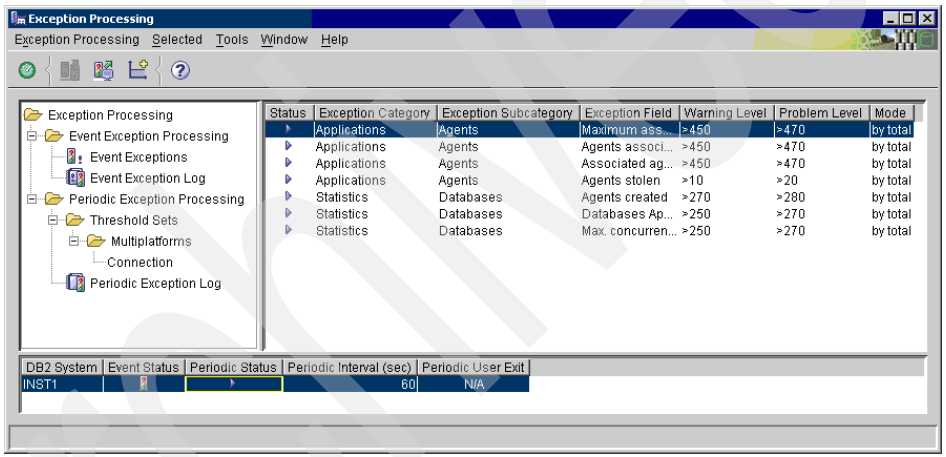


Figure 6-17 Connection Periodic events

For further details about setting periodic events, see 4.2.1, “Periodic exception processing” on page 195.

Start monitoring connections using DB2 performance expert

In order to continuously monitor the system, one of the essential requirements is the ability to take a quick glance at a large amount of data. DB2 Performance Expert helps you in this aspect by providing a graphical view of monitors. The System Health view of PE allows you to set up data views of your interest and helps you monitor large amounts of data quickly. This feature can help you analyze the trend of your database connections and help you define the actual workload on the database system.

For a trend analysis of connection constraints, we created two data views:

- Agent dataview, which monitors:
  - Agents assigned from pool (in green)
  - Agents created due to empty pool (in brown)
  - Stolen agents (in blue)
- Connection dataview, which monitors:
  - Connects to database

For more information about the setup of dataviews view 4.1.4, “System Health” on page 171.

Continuously change the workload using Web Performance Tool

Once our DB2 Performance Expert monitoring is enabled, it will start collecting and providing graphical data and events information. In addition to the continuous workload generated by two application servers, we started a multithreaded Java application that randomly connects, uses, and disconnects from the database using JDBC. This random load creates or uses the agents from the agents pool or creates a new one if it is not available.

Analyze the collected data

As we reached a threshold for connections and agents, DB2 Performance Expert sent an e-mail to the DBA and generated an event message, informing the DBA about the critical gauge values. Figure 6-18 shows the event message sent by DB2PE when the periodic exception situation was reached.

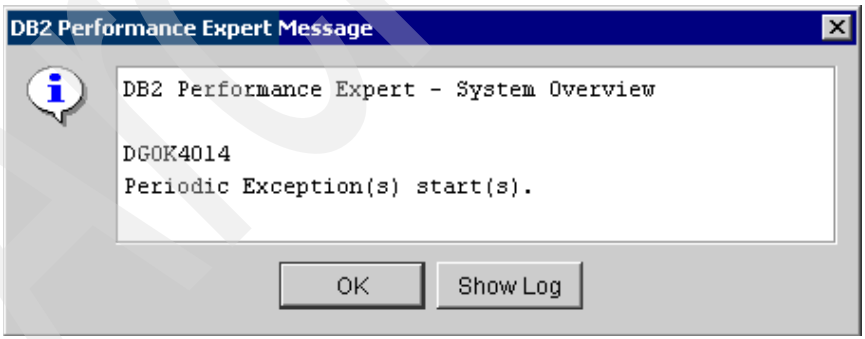


Figure 6-18 Event message

The DBA can now view the periodic logs to see further details of the periodic exceptions generated by system. Figure 6-19 shows the periodic logs generated by our workload.

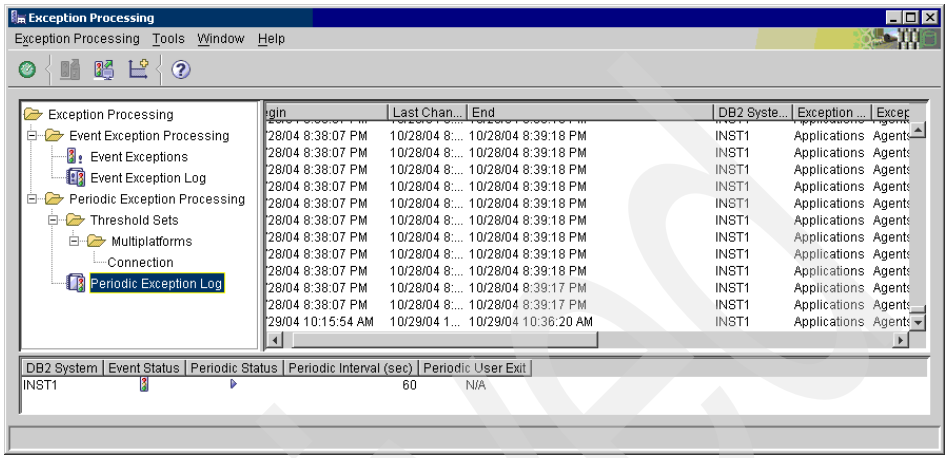


Figure 6-19 Periodic event logs

The trend analysis is another important aspect of watching the database. The graphical view in Figure 6-20 on page 355 shows the agent creation and usage trend as well as connection trends. The first graph in dataview shows the increase in already existing agents, which is good for system performance, and randomly generated workload, which is not causing any additional creation of agents at this point in time. Also, it shows the initial heavy agent creation load on the system.

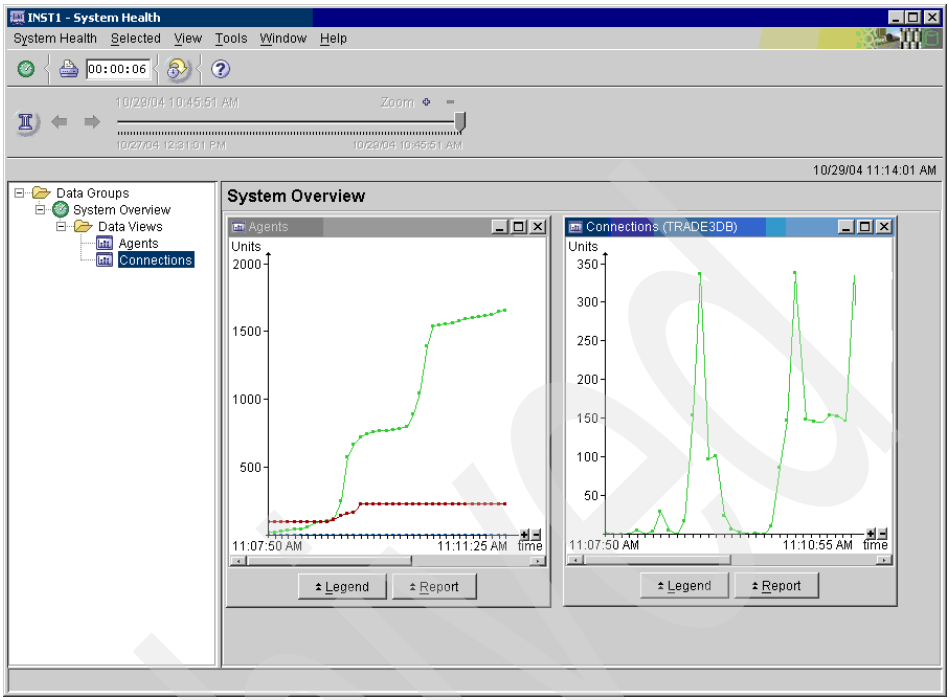


Figure 6-20 Connection and agents data view

Application problem analysis

Another set of problems where DB2 Performance Expert can help is application problems, for example, the number of times the application source code causes problems, such as continuously increasing deadlocks because connections are not closed, coarse transactions causing more load on system, under or overutilization of resources, and so on. All these issues are because of bad application design. DB2 Performance Expert cannot fix these problems, but it can help you figure out the problems and aid you in resolving them.

In this subsection, we look into one such problem. We created a Java program that was slowly but continuously creating a connection without releasing it. The DB2 PE System Health dataview shows us the trend shown in Figure 6-21.

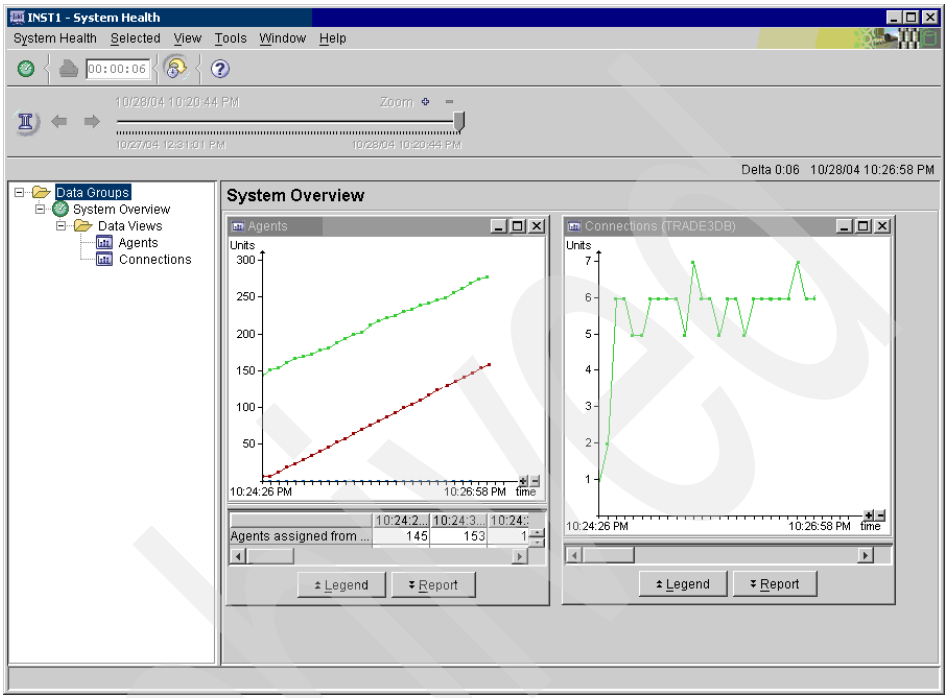


Figure 6-21 Continuously increasing workload

This trend for agents shows the underutilization of each opened agent. This trend can easily produce the conclusion that any agents assigned to an application is never returned back. We looked back at the source code and easily found that the application was using a connection and not releasing it.

6.1.3 Evaluating effects of a configuration change

When working with an application, whether the application is brand new or undergoing some upgrades, the DBA needs to understand the impact the application will have on the system and how to modify the system or database configurations to accommodate the application. It may not always be clear what parameter (if any!) needs to change, or that changing a parameter may actually cause degraded performance.

In this scenario, we examine how you can use Performance Expert to evaluate the effects of a database configuration change. This type of activity might

typically be performed during a system test or performance test cycle. We will highlight the use of several key PE features, including PWH queries and BPA.

Description of the application

We are using the TRADE3 application, also known as WebSphere Performance Benchmark Sample. It does some simple transactions against a database. We use this application not to test WebSphere, but rather as a ready-made, publicly available sample application only. We used a simple load driver to simulate a number of users going to the Web site.

Environment configuration

The lab environment used for this scenario is:

- Database server: jamaica, AIX 64-bit
- Instance: db2inst2, 64-bit UDB V8.1.6
- Database: TRADE3DB
- DB2 Performance Expert server: jamaica, AIX 64-bit, PE V2.1 (FP1 and FP2)
- PE instance: db2in8pe 32-bit UDB V8.1.6
- WebSphere server: SIAM, Windows 2000, WebSphere Application Sphere 4.x.

DB2 monitored instance settings

The monitored instance default monitor switch settings are shown in Example 6-2.

Example 6-2 jamaica db2inst2 default monitor switches

```
db2 get dbm cfg | grep DFT_MON
   Buffer pool                          (DFT_MON_BUFPOOL) = ON
   Lock                                    (DFT_MON_LOCK) = ON
   Sort                                    (DFT_MON_SORT) = ON
   Statement                               (DFT_MON_STMT) = ON
   Table                                  (DFT_MON_TABLE) = ON
   Timestamp                            (DFT_MON_TIMESTAMP) = ON
   Unit of work                               (DFT_MON_UOW) = ON
```

This scenario shows how you might use PE to examine the effect of database configuration changes. We decided (somewhat arbitrarily) to look at what might happen if we changed the AVG_APPLS database configuration parameter.

Note: The steps in this scenario are not scientific and we skipped things a good DBA might do, such as rebind the package. The intent is to show the *approach* and the *process* for using PE for this type of analysis, not to show you how to be a good DBA.

The *IBM DB2 Universal Database Administration Guide: Performance V8.2*, SC09-4821 manual describes AVG_APPLS this way:

"Average Number of Active Applications (avg_appls)

The SQL optimizer uses the avg_appls parameter to help estimate how much of the buffer pool might be available at run-time for the access plan chosen. Higher values for this parameter can influence the optimizer to choose access plans that are more conservative in buffer pool usage. If you specify a value of 1, the optimizer considers that the entire buffer pool will be available to the application."

We decided to try a AVG_APPLS value of 17, based on some quick analysis of previous load on our database. We ran a PWH query, not shown here, to look at the values of APPLS_IN_DB2 to help us come up with 17.

DB2 Performance Expert settings

The PE settings relevant to this scenario are:

- History snapshot interval: 60 seconds
- PWH granularity: 2 (30 minutes)

Workload used

We ran the load driver several times for about four hours for each run, starting at about 11:30 AM on November 11, 2004, with a number of users between 20 and 90.

We changed the AVG_APPLS value from 1 to 17 at about 11 AM on November 11, 2004.

Capturing the results

PE automatically captured the history data, at 60 second intervals, and the PWH tables were also loaded automatically, summarizing the data over 30 minute periods. We could run the tests unattended and then look at the data later. This is an important feature of PE that can be especially useful for when you do a controlled or long-running test in your environment - all you have to do is look at the results.

Using Buffer Pool Analysis

We opened the BPA feature, and ran a simple report for the period of November 11, 2004 at 6 AM through November 12, 2004 at 6 AM. This setup is shown in Figure 6-22.

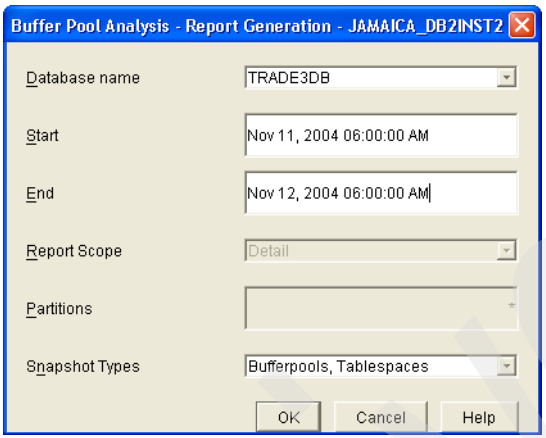


Figure 6-22 BPA Report setup for AVG_APPLS scenario

When this report completed, we looked at the graph for the Data Hit Ratio for one of the buffer pools, TRADEBP. This graph is shown in Figure 6-23. The graph shows a very interesting reduction in the hit ratio after making the configuration change! So maybe changing the AVG_APPLS was not such a good idea. We decided to look at a few more things.

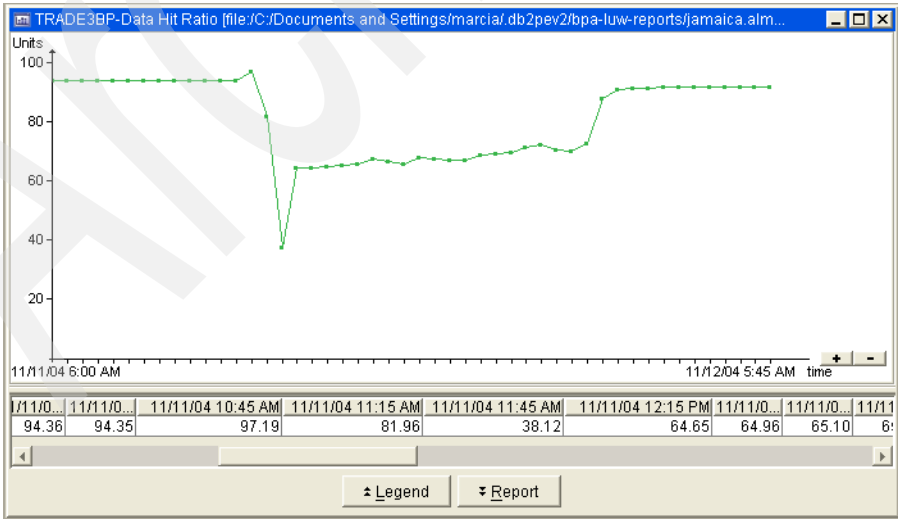


Figure 6-23 BPA Data Hit Ratio graph for AVG_APPLS scenario

Using PWH Queries

The BPA report showed that the hit ratio seemed to be negatively affected by changing the AVG_APPLS parameter. We also saw in the chart legend that the hit ratio was around 94%, up to about 11:00 AM, then it went way down and slowly improved, but only to about 73%.

We are pretty sure that we made the change to the parameter around 11 AM, and we know that we restarted the application at about 11:30 AM, so the slow increase could indicate that the buffer pool is getting primed (we restarted the database instance at this time as well.)

To verify this hypothesis, we prepared a PWH query to correlate the AVG_APPLS DB CFG parameter value to the BP data hit ratio. This query is shown in Example 6-3.

Example 6-3 PWH Query comparing AVG_APPLS to Buffer Pool data hit ratio

```
SELECT
  T1.INTERVAL_FROM AS DB_INTERVAL_FROM
, T1.INTERVAL_TO as DB_INTERVAL_TO
, T2.INTERVAL_TO as DBCFG_INTERVAL_TO
, T1.DB_NAME
, T1.BP_NAME
, T1.data_page_hit_ratio
, T2.avg_appls as DBCFG_avg_appls

from PWH.bufferpool T1

left outer join PWH.DBCFG T2
 ON  T2.INTERVAL_TO < T1.INTERVAL_TO   and T2.INTERVAL_TO >=  T1.INTERVAL_FROM
AND T1.DB_NAME   = T2.DB_NAME
AND T1.MEMBER_ID = T2.MEMBER_ID
where t2.db_name = ':db_name'
  and t1.bp_name = ':bp_name'
  and date(t1.interval_to) between ':from_date' and ':to_date'

ORDER BY T1.DB_NAME
```

Next, we executed the query, filling in the variables, as shown in Figure 6-24 on page 361.

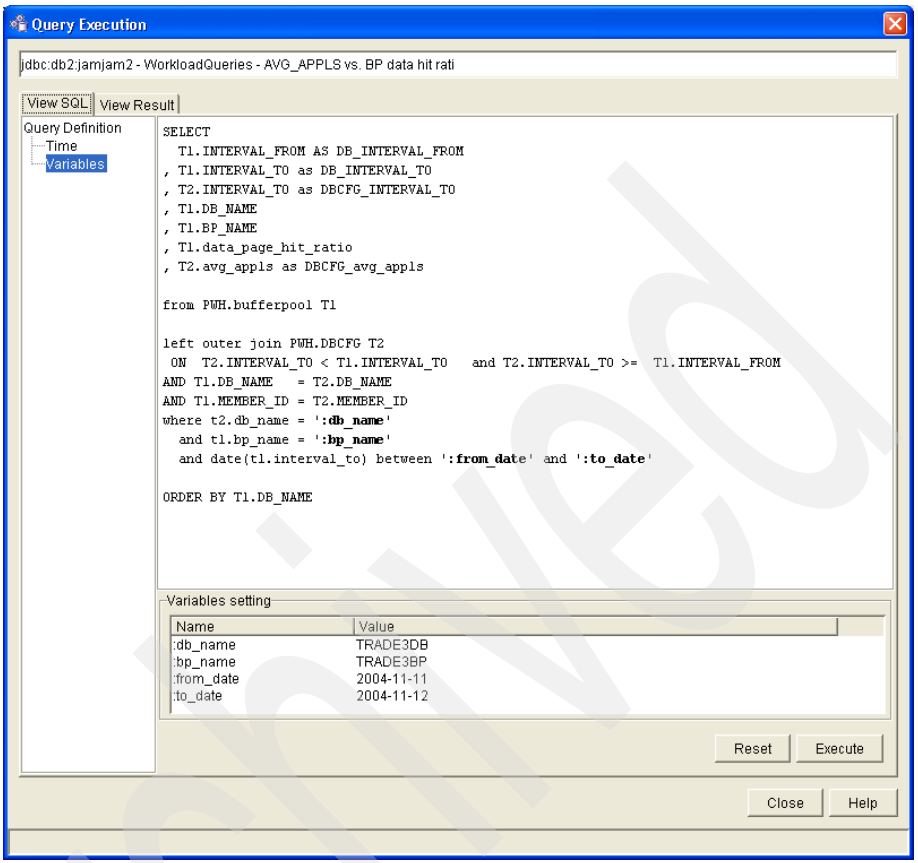


Figure 6-24 PWH Query for AVG_APPLS showing variables

The query results are shown in Figure 6-25 and Figure 6-26 on page 363. In the first figure, we see the AVG_APPLS started at 1 (the default) and changed to 17 somewhere between 10:49 AM and 11:42 AM. The time range is due to the way PE captures the DB configuration values, but this is perfectly acceptable for our purposes, and it validates our recollection that we changed the value around 11 AM. (While PE captures DB and DBM configuration values, it should never be considered an audit trail.)

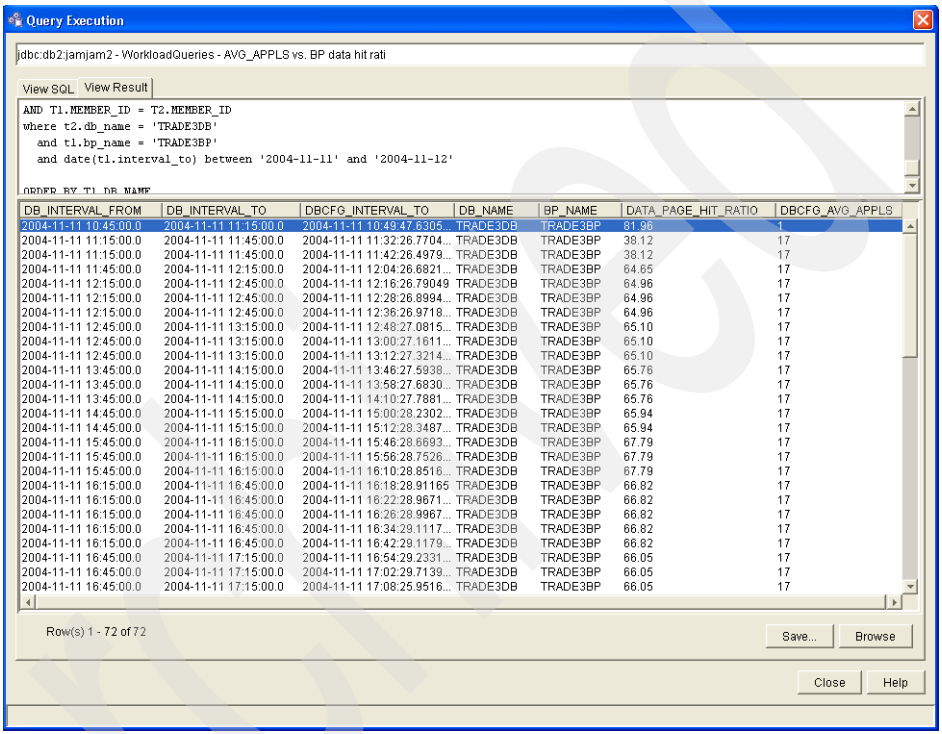


Figure 6-25 PQH Query results for AVG_APPLS, part 1

If we scroll down in the query results, we see, in Figure 6-26 on page 363, that the AVG_APPLS value changed again around 10 PM, from 17 to 8. We notice again that this change did not seem to make any improvement in the hit ratio. At roughly 11:30 PM, we changed the AVG_APPLS value back to its default and decided to go home for the night, knowing our database was back to its original state. (In a real world test, you would be more precise in resetting your environment, of course.)

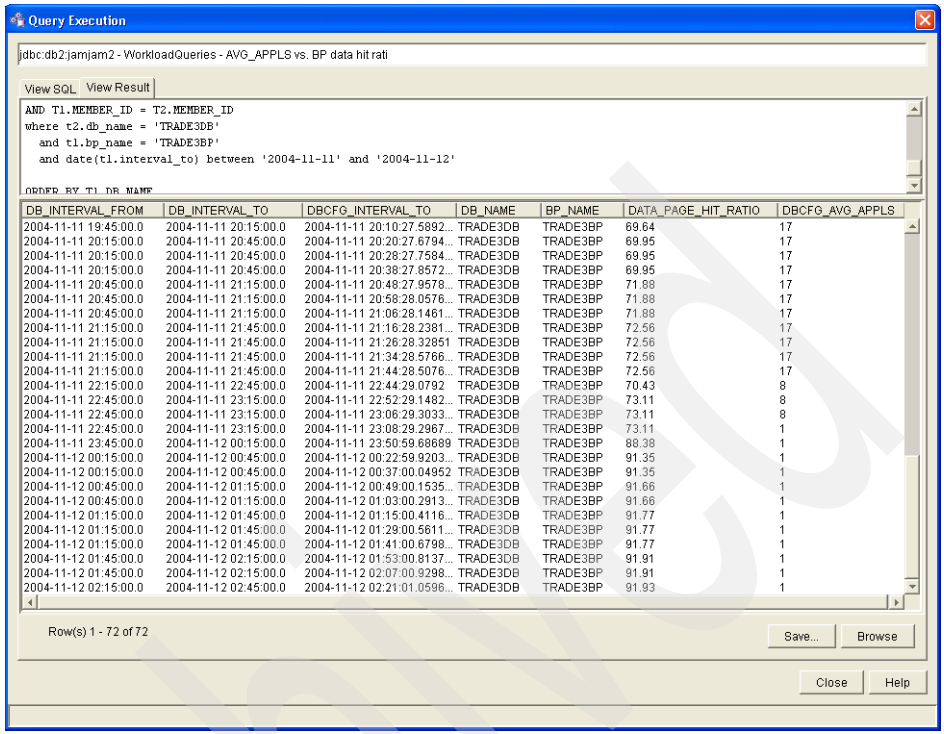


Figure 6-26 PQH Query results for AVG_APPLS, part 2

So, looking at the BPA graph and the PWH query, we see further evidence that changing the AVG_APPLS value had a negative impact on performance.

One other “trick” you can do with PE is take the query output, show it in a spreadsheet, and use the spreadsheet program’s graphing capabilities to create charts that PE does not provide.

On the query output, if you press the **Browse** button, the output is converted to HTML and shown in a browser. This is shown in Figure 6-27. If you have Microsoft® Excel®, and use Internet Explorer, you can just drag the URL from the browser into an Excel spreadsheet. You could also save the file and bring it into a spreadsheet manually.

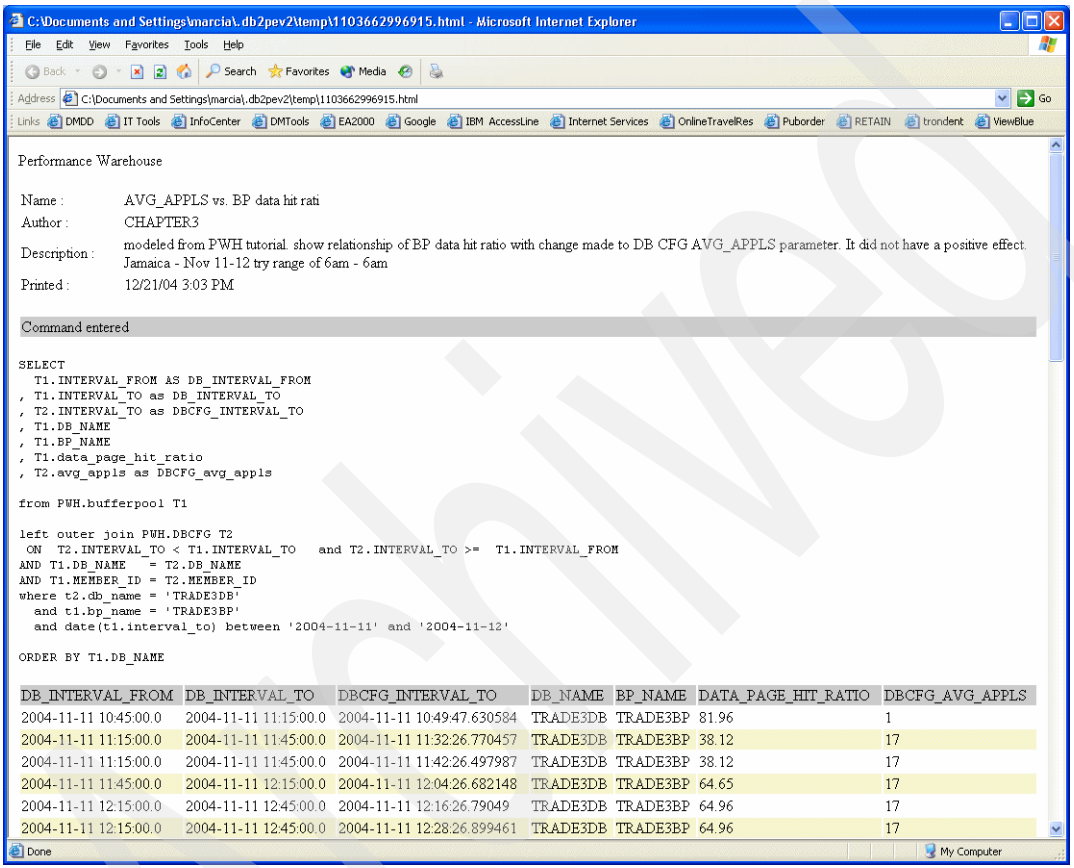


Figure 6-27 PWH Query output in browser

Once the data is displayed in the spreadsheet, you can select the appropriate column and row data to make a chart. We do not show those steps here, but we show the resulting “quick-and-dirty” (no nice formatting) chart in Figure 6-28 on page 365. This is a very interesting chart because now we can see both the Buffer Pool hit ratio *and* the AVG_APPLS value on the same graph, which highlights even more visually the correlation between the change in the configuration value and the degraded performance.

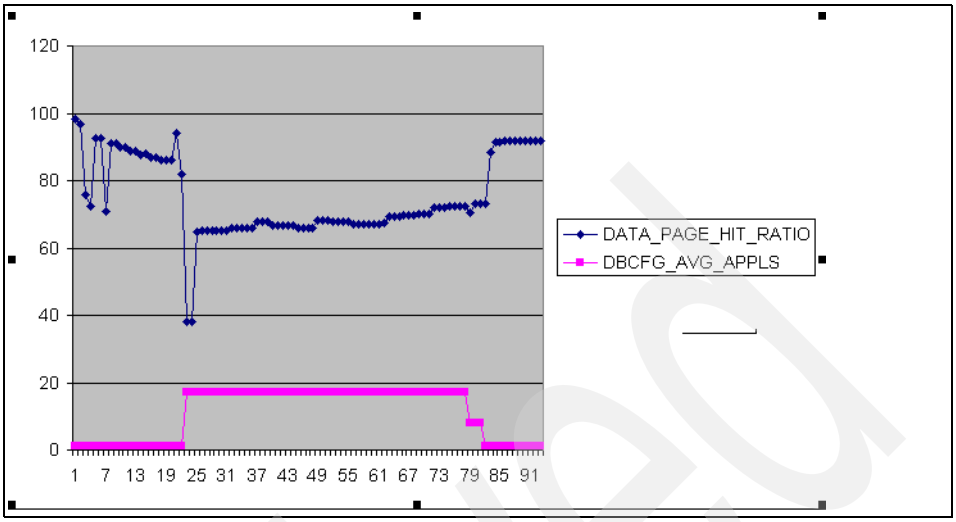


Figure 6-28 Correlated graph of AVG_APPLS and BP Hit Ratio

We leave the validity of this scenario to your imagination, but the intent was to show how PE can be used as part of your performance management tool set, and that you can use other tools as well to complement what PE does not yet provide.

6.2 Problem reported

So far we have seen the scenarios in which we did not have any reported problems, but we were doing regular analysis for keeping up the health of our DB2 database and system. Another important segment of DB2 system maintenance is handling problems when they occur. In this section, we cover scenarios on handling situations where the problem is shown by the database in the form of a return code or error, slow performance, or diaglog showing the database problem.

Scenarios covered in this section are as following:

- Identify and fix slow queries
- Alert and warnings setup and usage
- DB2 diaglog reported problems

6.2.1 Identify and fix slow SQL statements

Optimizing the database performance is one of the biggest challenges for a DBA, and optimizing the SQL statements is a challenge for both a DBA and application developer. Without any doubt, the performance of a database system, database, and SQL queries are vital elements for an optimal application. The poor query response time causes dissatisfaction in users and could lead to loss of business. It is a DBA's and application providers' responsibility to ensure the performance of business-critical applications by proactively identifying performance problems and making recommendations to correct poorly performing SQL statements.

DB2 Performance Expert provides you with all the necessary weapons to tackle the problem of diagnosing and analyzing your database and SQL queries for bad performance. The objective of this scenario is to demonstrate how to use PE to diagnosis and optimize the SQL queries for optimized resource usage and efficient response time. Also in this scenario, we look into other aspects of the database that may cause poor transactional time for user activities.

Background

Most of the DBAs will agree that the most difficult to tackle and influential reason for poor database application is inefficient SQLs. It is a widely accepted industrial fact that 80% of response time service level agreement (SLA) failures are caused by poor SQL.

Consider the following scenario: You developed an application whose functionality is perfect and the users want to use it, but its response time is slow. Users must wait for each activity. You look at your application source code and it is optimized well. The problem could be a poorly designed database, bad SQL queries, or an improper database server configuration.

Another common scenario is that your application is running smoothly and suddenly it slows down. The application response time grows and the transactions are delayed. Because of gradually degrading SQL queries, the growth in database size requires tuning the database configuration.

In this section, we try to cover problems caused by inefficient query or related database design. The performance of an SQL query is measured in terms of:

- Response time
- CPU usage
- I/O time
- Execution and compilation time

As most of the Web applications are recursive in nature, tuning SQL queries is the most effective way to improve the application performance. Badly written queries is one of the major factors for inefficient running queries, but it is not the

only one. A variety of factors can cause significant performance hitches in SQL queries execution. The reason can be:

- Badly written SQL queries

  SQL's syntax is easy to learn, but it is hard to be proficient at writing efficient queries. Writing efficient queries requires an understanding of how the database system is working. We frequently see SQL queries that are simple but have problems like fetching undesired data, inefficient search, and sort criterion. Badly written queries are those have one or multiple inefficient SELECT, WHERE, or GROUP BY sections.

- Inefficient access path

  An access path is an algorithm used by DB2 to execute SQL statements. When an SQL statement is launched, it is broken down, parsed, and turned into executable code. These paths can be a simple series of sequential reads to much more complicated strategies, such as using multiple indexes to access data. In order to create optimal SQL statements, it is very essential to understand the access path for your query and optimize it.

- Inefficient database design

  Much of the inefficiency of SQL queries is caused by bad database design. The bad design can be because of inappropriate column sizes or types, over-indulged columns, over or under normalization, inappropriate concurrents, and so on.

- Inefficient index

  A DB2 index is a modified B-tree structure that orders data values for rapid retrieval. The values being indexed are stored in an inverted tree structure. Badly defined or non-existent indexes are one of the primary reasons for poor performance, and fixing these can often lead to phenomenal improvements.

- Inappropriate database configuration

  After index and database design, database configuration is another major reason for bad performance for SQL statements. This factor can impact database applications very badly. The configuration includes proper setup of tablespace, buffer pools, partitions, and so on.

- Locking constraints

  Another reason for slow query response is locking problems. This delay can be cause by lock waiting, lock upgrading, deadlocks, or inappropriate locking or isolation levels.

- Bad sorting

  Sorting is one of the most time and CPU consuming activities there is, and it should always be avoided. The SQL query can be improved several times over if efficient sorting is used. The performance problems caused by sorting can be inappropriate sort heap size, inappropriate indexes causing excessive sorting, over use of sorting, and so on.

- Inefficient cache usage

  DB2 uses cache at many levels, such as statement cache, table or index cache, and so on. One of the most important performance elements is buffer pool. A buffer pool is memory used to cache table and index data pages as they are being read from disk or being modified. The buffer pool improves database system performance by allowing data to be accessed from memory instead of from disk. A proper setting for buffer pool is essential to SQL performance.

- Inefficient user defined functions and procedures

  Using stored procedures and user defined functions for recursive or frequently used SQL activities is one of the most important database improvements as it saves response time by saving on compilation time and reutilizing the execution plan. Also, it improves the network if the set of SQL queries are required to be run. On the other hand, inefficient user defined functions, and stored procedures can cost a lot, as they are run on the server side.

- Network bottleneck

  Many times the SQL response time is bad because of network involved in the query. This bad database application performance can be avoided by reducing the number of remote requests. This can be done by using batch, stored procedures, and so on.

- Memory constants

  As discussed in cache usage, memory is one resource that can diminish continuously and quickly when inefficiently used. On the other hand, insufficient memory can cause higher disk I/O, which is not good for database performance, and hence slower response time. As much as possible, database parameters should be set in such a way that disk I/O is minimized.

Scenario description

This scenario is aimed to minimize the poor SQL performance using DB2 Performance Expert. DB2 Performance Expert helps us at various levels of query improvement. We show the usage of DB2 Performance Expert for SQL improvement at three levels:

- Identifying slow SQL statements

  One inefficient SQL query can cause a huge performance impact, because most of the time it is used repetitively. We understand that the most important facet of SQL queries improvement is finding the queries that are not working efficiently. No doubt this is a difficult task, because in a typical complex system, the number of SQL queries executing is very large and a number of applications are simultaneously run.

- Fixing slow SQL statement problems

  Once inefficient SQL statements are identified, the next step is to do the necessary steps to improve their performance. This process can include improvement of database design, addition or removal of indexes, improvement in the SQL statement itself, or database setups. The DB2 system where SQL statements are running can also be the reason why the SQL statements run slowly. We also give examples of problem identification using PE.

To show the usage of DB2 Performance Expert, we implemented an application and database to set up the scenario. The application used is a simple Java shopping cart application, which randomly creates a workload for a database server. The shopping cart application is an application that frequently but randomly does the following activities:

- Creates a shopping cart for a user

  This is an activity in which a shopping cart is created randomly. The frequency of its occurrence is set to once in four activities. The SQL statements involved in this activity is shown in Example 6-4.

Example 6-4 Create shopping cart SQL activities

```
select userid,name,balance,info from user where name=? group by
userid,name,balance,info order by balance

select productid,productname,type,amount,description from product1 where
productid= ? group by type,amount,productname,productid,description order
by amount

select userid,productid,quantity from cartproductlink where productid= ?
and userid=?

delete from cartproductlink where productid= ? and userid=?

insert into cartproductlink values(?,?,?)

update user set balance=? where userid=?
```

- Delete shopping cart for a user

  This activity involves deletion of shopping cart for a randomly selected user. The frequency of its occurrence is set to once in four activities. The SQL activities involved in deletion of cart are shown in Example 6-5.

Example 6-5 Shopping cart deletion SQL activities

```
select userid,name,balance,info from user where name=? group by
userid,name,balance,info order by balance

select productid,productname,type,amount,description from product1 where
productid= ? group by type,amount,productname,productid,description order
by amount

select userid,productid,quantity from cartproductlink where productid= ?
and userid=?

delete from cartproductlink where productid= ? and userid=?

update user set balance=? where userid=?
```

- View products from shopping cart for a user

  This activity fetches the products in a shopping cart for a randomly selected user. The frequency of this activity is set to once in 10 activities. The SQL statements involved in this activity are shown in Example 6-6.

Example 6-6 Viewing shopping cart products SQL activities

```
select userid,name,balance,info from user where name=? group by
userid,name,balance,info order by balance

select userid,productid,quantity from cartproductlink where userid=?
```

- Add product to shopping cart for a user

  In this activity, we randomly add a product to a shopping cart for a randomly chosen user. The frequency of this activity is once in 10 activities. The SQL queries involved in this activities are shown in Example 6-7.

Example 6-7 Update shopping cart SQL activities

```
select userid,name,balance,info from user where name=? group by
userid,name, balance, info order by balance

select productid,productname,type,amount,description from product1 where
productid= ? group by type, amount, productname, productid, description
order by amount
```

```
select userid,productid,quantity from cartproductlink where productid= ?
and userid=?

update cartproductlink set productid= ? ,userid=?,quantity=? where
productid=? and userid=?

update user set balance=? where userid=?
```

- Updating product for change in price

  This activity is not run as frequently as all the activities above (once in 100 activities). In this activity, all the products are picked and their prices are changed. The importance of this activity is that it is huge activity; it hit our performance in the form of long running transaction. The SQL statements involved in this activity are shown in Example 6-8.

Example 6-8 Update product SQL activity

```
select productid,productname,type,amount,description from product1  group
by type,amount,productname,productid,description order by amount

update product1 set amount=? where productid=?
```

- Discounting, sale, and promotion implementation for product

  This activity is similar to the above activity; the only difference is that it uses UDF for discounting the product price. The frequency of this activity is set to once in 50 activities. The SQL statements involved in this activity are shown in Example 6-9.

Example 6-9 Discount product price SQL activity

```
select productid,productname,type,amount,description,discounted(amount) as
discounted from product1 group by
type,amount,productname,productid,description order by amount

update product1 set amount=? where productid=?
```

- Adding a user and product to database

  This activity is another less frequent activity (once in 50 activities) created to increase the size of database by adding more users and products in the database. This activity involves the SQL statements shown in Example 6-10.

Example 6-10 Adding user and product SQL activities

```
select max(userid) from user

select max(productid) from product1

insert into user values(?,?,?,?,?,?,?)

insert into product1 values(?,?,?,?,?,?)
```

All the above SQL statements are written with the aim of showing the improvements that can be done to SQL queries.

Scenario prerequisite and setup

After installing DB2 UDB and DB2 Performance Expert, the database is created and the required database and instance configurations are set. The DiscountRate.class is copied to the SQLlib\Function directory. Example 6-11 shows the contents of Startup.bat, which contains the database creation and configuration setting statements.

Example 6-11 Database setup and data loading

```
create db pecart
connect to pecart
create table user
  (userid int,
   name varchar(100),
   address varchar(500) ,
   city varchar(100),
   balance double,
   history blob,
   info varchar(3000) )

create table product1
  (productid int,
   productname varchar(100),
   description varchar(3000),
   amount double,
   type varchar(100),
   manufacturer varchar(500))

create table cartproductlink
```

```
  (userid int,
   productid int,
   quantity int)

CREATE FUNCTION discounted(DOUBLE) RETURNS DOUBLE EXTERNAL NAME
'DiscountRate!discount' SCRATCHPAD 10 FINAL CALL VARIANT NO SQL PARAMETER STYLE
DB2GENERAL LANGUAGE JAVA NO EXTERNAL ACTION

update dbm cfg using DFT_MON_BUFPOOL ON
update dbm cfg using DFT_MON_LOCK ON
update dbm cfg using DFT_MON_SORT ON
update dbm cfg using DFT_MON_STMT ON
update dbm cfg using DFT_MON_TABLE ON
update dbm cfg using DFT_MON_UOW ON
update db cfg for pecart using locklist 500
db2set DB2_FMP_COMM_HEAPSZ=15000
disconnect pecart

db2stop
db2start
set classpath=./db2pe.jar;%classpath%
java com.ibm.db2pe.sample.DBLoadMT -dbname pecart -load 1000
```

The DBLoadMT Java class actually inserts the required number of users and products into the database.

Important: This scenario is executed on DB2 UDB 8.2 and DB2 Performance Expert Version 2 Fix Pack 2.

Now DB2 Performance Expert server is started and begins monitoring the PECART database.

1. Set up history settings.

   Select the database instance and select **Selected** → **Properties**. In the History tab, set Recording interval to 60 seconds and enable Locking conflict with Interval multiplier 1, as shown in Figure 6-29.

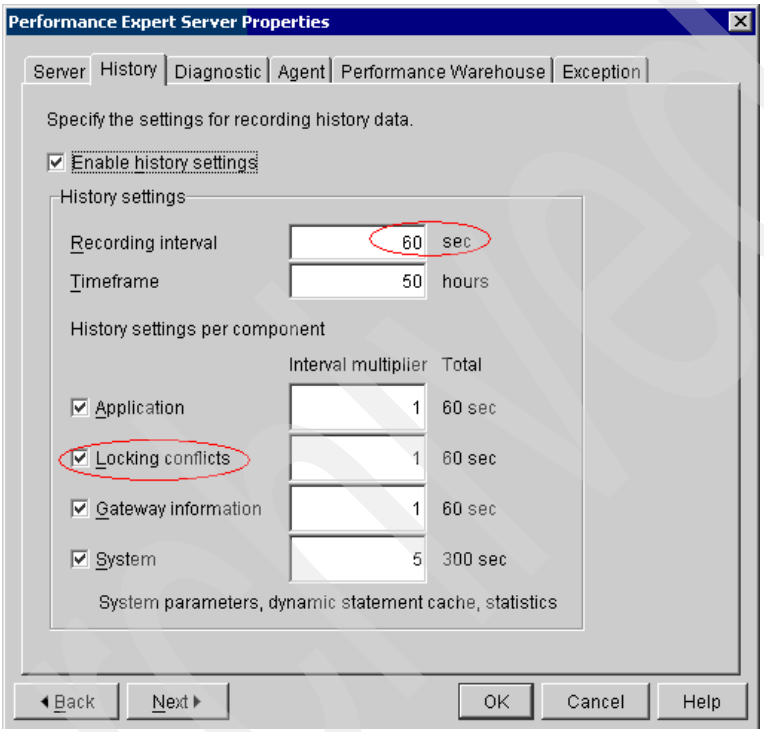


Figure 6-29 History settings

2. Start exception monitoring for deadlock.

   Select the database instance and select **Selected** → **Exception** → **Start** → **Event**. In the lower frame of System Overview window, you should see "event" in Exception column for the database instance.

3. Set up System Health for quick graphical views.

   In the System Health view, follow the steps below:

   a. Create a new data view group by selecting **Selected** → **New**.

b. Open the following predefined data views by selecting **Data Views** and selecting **Selected** → **Open Predefined Data View**. Select the following predefined data views:

- Logger I/O activity of databases
- Number of lock escalations
- Hash join data exceeded available sort heap space
- SQL stmt throughput
- Prefetchers and page cleaner
- SQL stmt. distribution
- Shows the data reads per asynchronous request
- Sort overflows [%]

4. Set up Performance Warehouse.

   Open Performance Warehouse - Expert, and follow the steps below to configure Performance Warehouse to start collecting warehouse data, which will be further used to identify slow queries and configure your database optimally.

   a. Create a process group by selecting **Process Groups** and selecting **Selected** → **Create**; name it *pecard*.

   b. Select the process **Groups** and select **Public** → **Processes** → **DB2PM.Template.SQL.Activity Trace Report**, and select **Selected** → **Copy**.

   c. Select **pecart** → **Processes** → **SQL Activity Trace Report** → **Steps** in the newly copied Process.

   d. Select **CRD** in the right-hand pane and select **Selected** → **Properties**; this will open the CRD step Properties, as shown in Figure 6-30 on page 376.

e. Select the database name PECART in the drop-down box, and enter 10 minutes as the elapse time, as shown in Figure 6-30.

f. Select **pecart** → **Processes** → **SQL Activity Trace Report** and select **Selected** → **Execute**.

g. Check if the performance warehouse process is started in Process Executions.

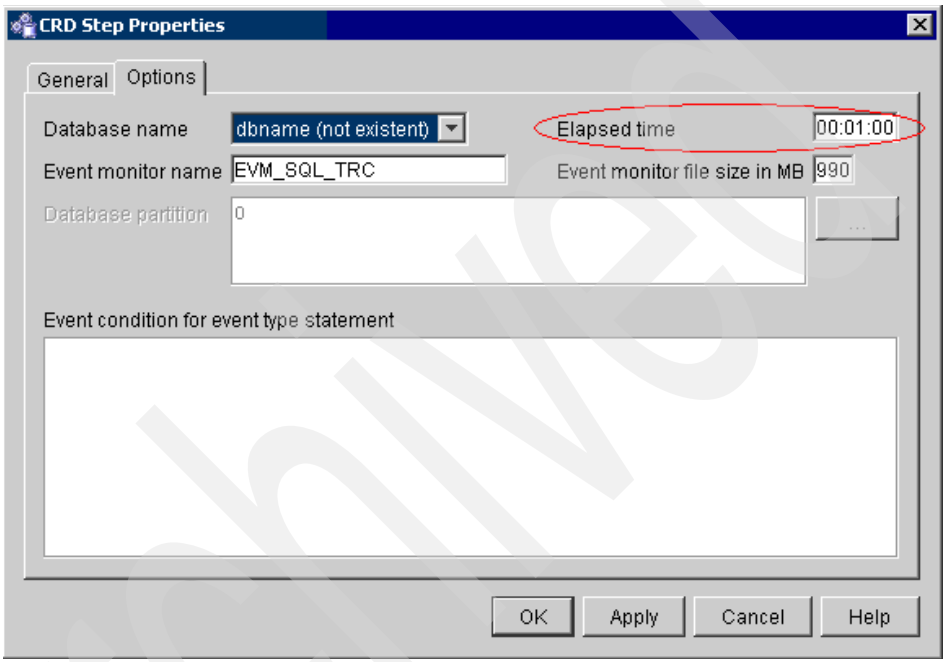


Figure 6-30 CRD step properties

The necessary setup for DB2 Performance Expert is done now; it is time to start the shopping cart application by launching **`run.bat`** or **`java com.ibm.db2pe.sample.DBRunMT`**, as shown in Example 6-12 on page 377.

This will start the application for 5 minutes using 100 connections (this is the default; these values can be changed using the -time and -connection parameters).

Example 6-12 Shopping cart application

```
C:\pesample>run.bat
C:\pesample>set
classpath=.\pecart.jar;.\pecart.jar;.\pecart.jar;.\pecart.jar;.;C:\IBM\SQLLIB\j
ava\db2java.zip;C:\IBM\SQLLIB\java\db2jcc.jar;C:\IBM\SQLLIB\java\sqlj.zip;C:\IB
M\SQLLIB\java\db2jcc_license_cu.jar;C:\IBM\SQLLIB\bin;C:\IBM\SQLLIB\tools\db2XT
rigger.jar;C:\IBM\SQLLIB\java\common.jar

C:\pesample>java com.ibm.db2pe.sample.DBRunMT -help
Usage: java com.ibm.db2pe.sample.DBLoadMT [-help][-dbname <database name>]
[-user <user>] [-passwd <passwd>] [-schema <schema name>] [-conn <Number of
concurent  connections>] [-time <Runtime in minutes>]

C:\pesample>java com.ibm.db2pe.sample.DBRunMT -dbname pecart -conn 100 -time 5
-showdetails
created cartproductlink for userid=303 and productid=4
Deleted cartproductlink for userid=428 and productid=303
created cartproductlink for userid=821 and productid=804
Could not update cartproductlink for userid=312 and productid=863
created cartproductlink for userid=915 and productid=154
Could not update cartproductlink for userid=84 and productid=560
Could not update cartproductlink for userid=461 and productid=372
created cartproductlink for userid=120 and productid=217
created cartproductlink for userid=374 and productid=398
Deleted cartproductlink for userid=671 and productid=867
Deleted cartproductlink for userid=688 and productid=129
Deleted cartproductlink for userid=243 and productid=126
Could not update cartproductlink for userid=66 and productid=95
created cartproductlink for userid=471 and productid=89
.
.
.
.
connection was
DB2Connection
{
    connectionHandle = 6
    SPConnected = false
    source = pecart
    user =
    conArgs =
    closed = false
    describeCached = false
    describeParam = true
    isReadOnly = false
    autoClose = false
    LONGDATA compat = false
}
```

```
COM.ibm.db2.jdbc.DB2Exception: [IBM][CLI Driver][DB2/NT] SQL0911N  The current
transaction has been rolled back because of a deadlock or timeout.  Reason code
"2".  SQLSTATE=40001

        at
COM.ibm.db2.jdbc.app.SQLExceptionGenerator.throw_SQLException(Unknown
 Source)
        at
COM.ibm.db2.jdbc.app.SQLExceptionGenerator.throw_SQLException(Unknown
 Source)
        at COM.ibm.db2.jdbc.app.SQLExceptionGenerator.check_return_code(Unknown
Source)
        at COM.ibm.db2.jdbc.app.DB2PreparedStatement.execute2(Unknown Source)
        at COM.ibm.db2.jdbc.app.DB2PreparedStatement.executeUpdate(Unknown
Sourc
e)
        at com.ibm.db2pe.sample.DBRunMT.updateproduct(DBRunMT.java(Compiled
Code
))
        at com.ibm.db2pe.sample.DBRunMT.run(DBRunMT.java(Compiled Code))
.
.
.
.
.
Total create time for 211 activities is 7306sec Average=34sec
Total update time for 238 activities is 8309sec Average=34sec
Total delete time for 236 activities is 7051sec Average=29sec
Total view time for 99 activities is 97sec Average=0sec
Total updateproduct time for 21 activities is 2423sec Average=115sec
Total discountproduct time for 17 activities is 1655 Average=97sec
Workload completed press any key with enter to close all connections and exit
.
.
.
```

We can see that we got some deadlocks while running our user shopping cart application. Also, we can see that the average creation time is 34 seconds, the average time for update is 34 seconds, the average delete activity time is 29 seconds, the view time is less then 1 second, the updateproduct time average is 115 seconds, and the discount product average is 97 seconds. We will consider these values in performance measurement of the shopping cart application.

Identifying slow SQL statements

Identifying slow SQL statements is one of the most difficult task in application tuning. It is like finding a needle in a haystack, especially when a number of applications are accessing the database server. In such a scenario, the number of queries and concurrent connections are huge and analyzing them separately is not an easy task. DB2 Performance Expert can help you in this respect, as it provides all the information you need to monitor your queries performance.

DB2 Performance Expert provide a huge amount of data; some of this information directly provides information regarding the performance, while some of the information has to be used with other data to deduce the information. So the data provided by DB2 Performance Expert can be interpreted in many ways. DB2 Performance Expert provides information about SQL statements running or ran on your server in many forms.

Online monitoring information shows the statements' information while they are being executed. The information is provided at the DB2 server level for each application. These views can be used very easily and directly to watch for information about statements. The snapshot history is maintained for a short period of time for analysis after the application is executed.

To view the online monitoring information, open the Statistics Details view and select the **Dynamic SQL Statements** pane. Check the **Receive statement cache information** check box. This will show the information about the DB2 statement cache that stores frequently running queries.

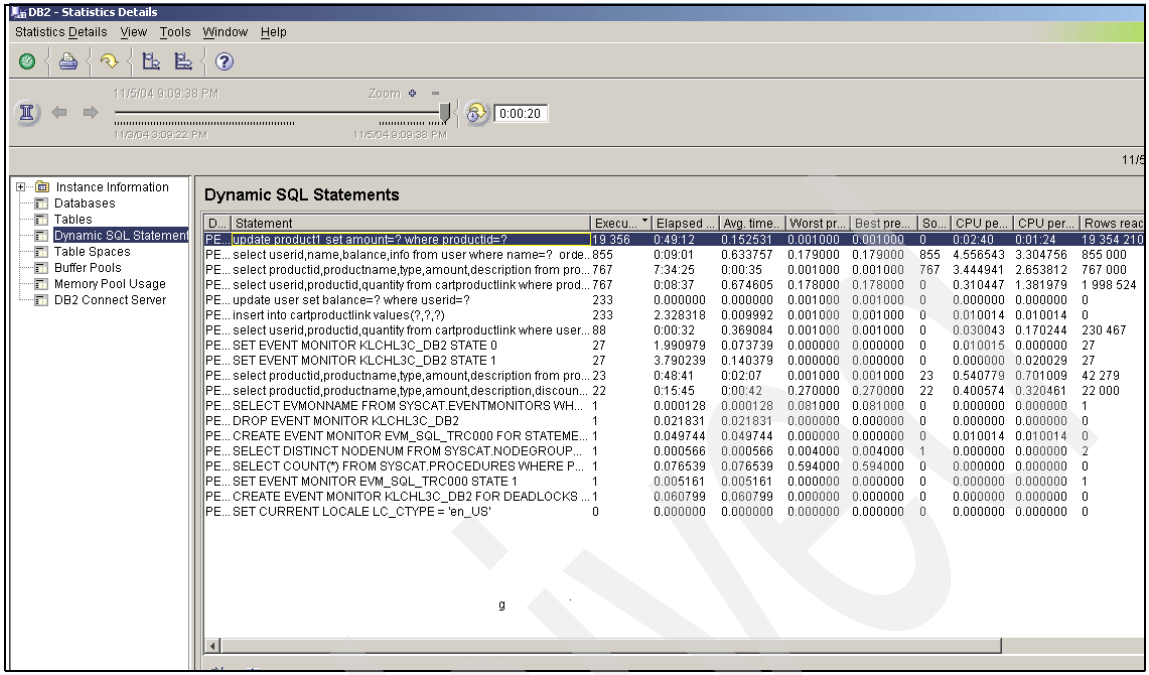


Figure 6-31 Dynamic SQL queries

This view shows you information that can be used to show the statements and identify the most frequently run statement, query which is slowest and killing your application performance, or the statement that is using most of your CPU. As we can see in Figure 6-31, `Update product1 set amount=? where productid=?` is the most frequently run statement, while `select productid,productname,type,amount,description from product1 where productid= ? group by type,amount,productname,productid,description order by amount` is most costly statement. The performance tuning of both these statements is important, as both of them can influence the application performance a great deal.

Selecting and double-clicking on the statement will show you the details of the statement. These details are shown in Figure 6-32 on page 381, and includes the following information about the statements No. of Executions, No. of times it was Completed, Worst preparation time, Best preparation time, Sorts, Elapsed execution time, Avg. time per execution, CPU per statement, CPU per user statement, Rows read, Rows written, Internally deleted rows, Internally inserted rows, and Internally updated rows.

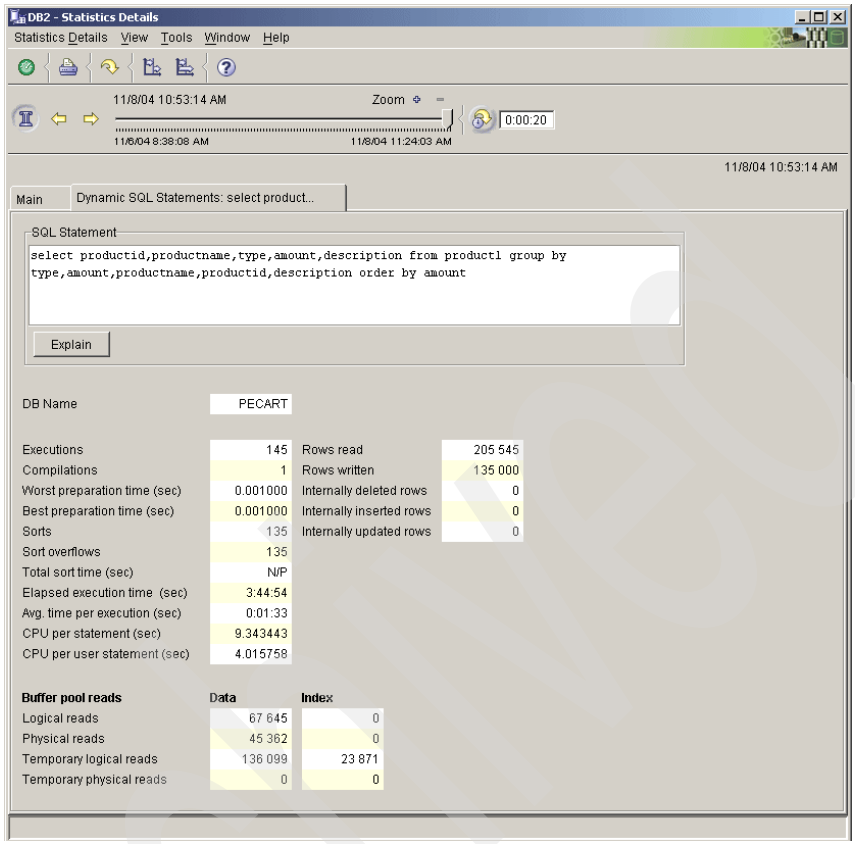
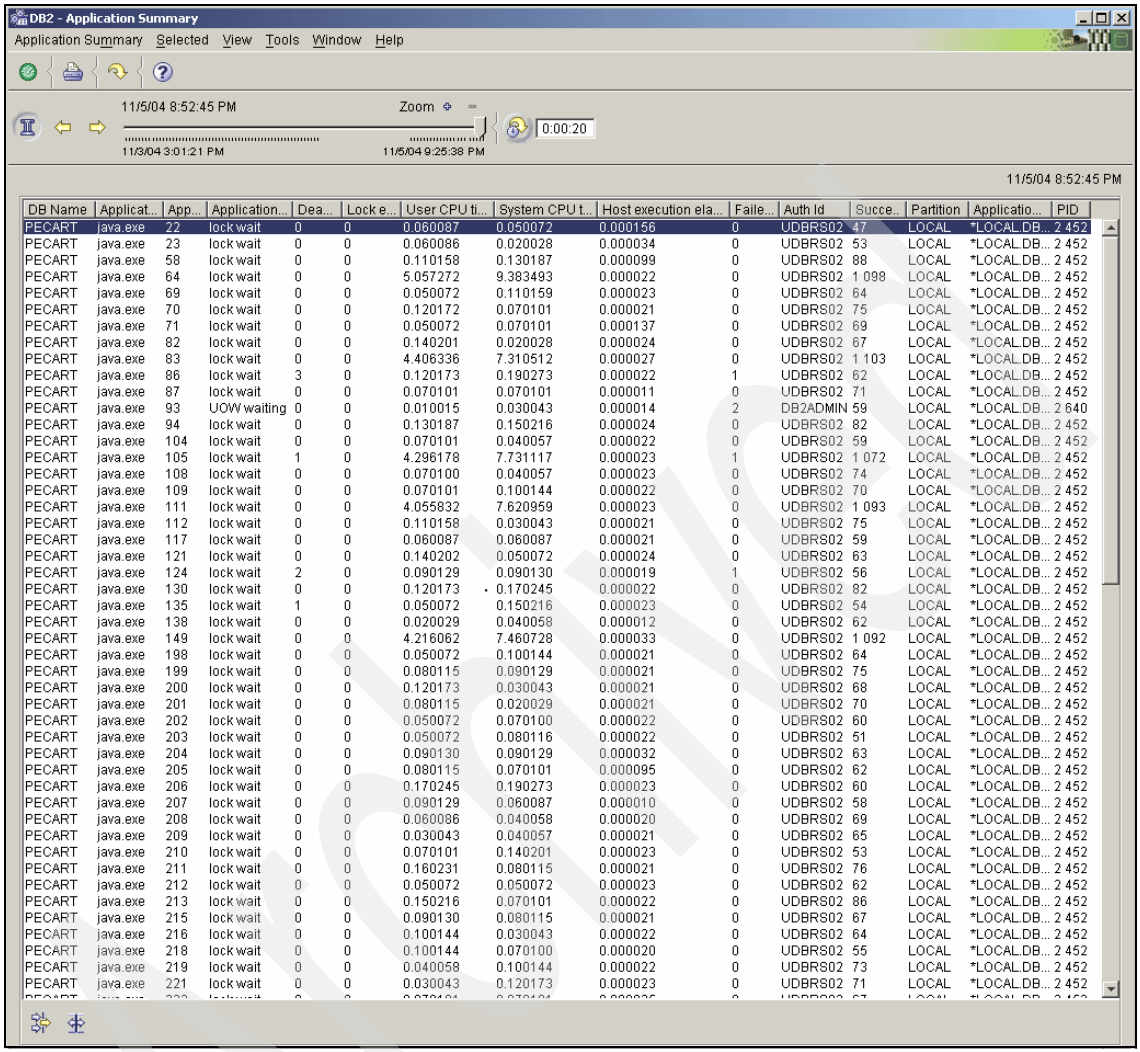


Figure 6-32 Statement details

We use some of this information in the coming sections for improving our system as well as the SQL statements. Clearly, the first SELECT statement in the list is the slowest of all the statements run in the PECART application. Although in the PECART application case most of the statements are executed quite frequently and all of them are quite time consuming, for demonstration purposes, we only analyze the query that has the worst average execution time. It is a nice idea to analyze all the queries that are frequently run, especially ones that are taking up a large segment of CPU processing.

Another section worth watching out for is the Application Summary view; in this view, you can look for all the applications currently connected to the database. If your application is using multiple connections, each one of them is shown separately. In this view, you can see the connection that is executing most of the SQL statements, the one that is using most of the CPU time, and a lot of other useful information about the application. Figure 6-33 on page 383 shows the connections used by the PECART application. The information you see here is the short-term snapshot history for this application.

In this view, all the applications showing deadlocks, locks escalation, or high CPU utilization is of special interest to performance monitoring.



DB Name	Applicat...	App...	Application...	Dea...	Lock e...	User CPU ti...	System CPU t...	Host execution ela...	Faile...	Auth Id	Succe..	Partition	Applicatio...	PID
PECART	java.exe	22	lock wait	0	0	0.060087	0.050072	0.000156	0	UDBRS02	47	LOCAL	*LOCAL.DB...	2 452
PECART	java.exe	23	lock wait	0	0	0.060086	0.020028	0.000034	0	UDBRS02	53	LOCAL	*LOCAL.DB...	2 452
PECART	java.exe	58	lock wait	0	0	0.110158	0.130187	0.000099	0	UDBRS02	88	LOCAL	*LOCAL.DB...	2 452
PECART	java.exe	64	lock wait	0	0	5.057272	9.383493	0.000022	0	UDBRS02	1 098	LOCAL	*LOCAL.DB...	2 452
PECART	java.exe	69	lock wait	0	0	0.050072	0.110159	0.000023	0	UDBRS02	64	LOCAL	*LOCAL.DB...	2 452
PECART	java.exe	70	lock wait	0	0	0.120172	0.070101	0.000021	0	UDBRS02	75	LOCAL	*LOCAL.DB...	2 452
PECART	java.exe	71	lock wait	0	0	0.050072	0.070101	0.000137	0	UDBRS02	69	LOCAL	*LOCAL.DB...	2 452
PECART	java.exe	82	lock wait	0	0	0.140201	0.020028	0.000024	0	UDBRS02	67	LOCAL	*LOCAL.DB...	2 452
PECART	java.exe	83	lock wait	0	0	4.406336	7.310512	0.000027	0	UDBRS02	1 103	LOCAL	*LOCAL.DB...	2 452
PECART	java.exe	86	lock wait	3	0	0.120173	0.190273	0.000022	1	UDBRS02	62	LOCAL	*LOCAL.DB...	2 452
PECART	java.exe	87	lock wait	0	0	0.070101	0.070101	0.000011	0	UDBRS02	71	LOCAL	*LOCAL.DB...	2 452
PECART	java.exe	93	UOW waiting	0	0	0.010015	0.030043	0.000014	2	DB2ADMIN	59	LOCAL	*LOCAL.DB...	2 640
PECART	java.exe	94	lock wait	0	0	0.130187	0.150216	0.000024	0	UDBRS02	82	LOCAL	*LOCAL.DB...	2 452
PECART	java.exe	104	lock wait	0	0	0.070101	0.040057	0.000022	0	UDBRS02	59	LOCAL	*LOCAL.DB...	2 452
PECART	java.exe	105	lock wait	1	0	4.296178	7.731117	0.000023	1	UDBRS02	1 072	LOCAL	*LOCAL.DB...	2 452
PECART	java.exe	108	lock wait	0	0	0.070100	0.040057	0.000023	0	UDBRS02	74	LOCAL	*LOCAL.DB...	2 452
PECART	java.exe	109	lock wait	0	0	0.070101	0.100144	0.000022	0	UDBRS02	70	LOCAL	*LOCAL.DB...	2 452
PECART	java.exe	111	lock wait	0	0	4.055832	7.620959	0.000023	0	UDBRS02	1 093	LOCAL	*LOCAL.DB...	2 452
PECART	java.exe	112	lock wait	0	0	0.110158	0.030043	0.000021	0	UDBRS02	75	LOCAL	*LOCAL.DB...	2 452
PECART	java.exe	117	lock wait	0	0	0.060087	0.060087	0.000021	0	UDBRS02	59	LOCAL	*LOCAL.DB...	2 452
PECART	java.exe	121	lock wait	0	0	0.140202	0.050072	0.000024	0	UDBRS02	63	LOCAL	*LOCAL.DB...	2 452
PECART	java.exe	124	lock wait	2	0	0.090129	0.090130	0.000019	1	UDBRS02	56	LOCAL	*LOCAL.DB...	2 452
PECART	java.exe	130	lock wait	0	0	0.120173	0.170245	0.000022	0	UDBRS02	82	LOCAL	*LOCAL.DB...	2 452
PECART	java.exe	135	lock wait	1	0	0.050072	0.150216	0.000023	0	UDBRS02	54	LOCAL	*LOCAL.DB...	2 452
PECART	java.exe	138	lock wait	0	0	0.020029	0.040058	0.000012	0	UDBRS02	62	LOCAL	*LOCAL.DB...	2 452
PECART	java.exe	149	lock wait	0	0	4.216062	7.460728	0.000033	0	UDBRS02	1 092	LOCAL	*LOCAL.DB...	2 452
PECART	java.exe	198	lock wait	0	0	0.050072	0.100144	0.000021	0	UDBRS02	64	LOCAL	*LOCAL.DB...	2 452
PECART	java.exe	199	lock wait	0	0	0.080115	0.090129	0.000021	0	UDBRS02	75	LOCAL	*LOCAL.DB...	2 452
PECART	java.exe	200	lock wait	0	0	0.120173	0.030043	0.000021	0	UDBRS02	68	LOCAL	*LOCAL.DB...	2 452
PECART	java.exe	201	lock wait	0	0	0.080115	0.020029	0.000021	0	UDBRS02	70	LOCAL	*LOCAL.DB...	2 452
PECART	java.exe	202	lock wait	0	0	0.050072	0.070100	0.000022	0	UDBRS02	60	LOCAL	*LOCAL.DB...	2 452
PECART	java.exe	203	lock wait	0	0	0.050072	0.080116	0.000022	0	UDBRS02	51	LOCAL	*LOCAL.DB...	2 452
PECART	java.exe	204	lock wait	0	0	0.090130	0.090129	0.000032	0	UDBRS02	63	LOCAL	*LOCAL.DB...	2 452
PECART	java.exe	205	lock wait	0	0	0.080115	0.070101	0.000095	0	UDBRS02	62	LOCAL	*LOCAL.DB...	2 452
PECART	java.exe	206	lock wait	0	0	0.170245	0.190273	0.000023	0	UDBRS02	60	LOCAL	*LOCAL.DB...	2 452
PECART	java.exe	207	lock wait	0	0	0.090129	0.060087	0.000010	0	UDBRS02	58	LOCAL	*LOCAL.DB...	2 452
PECART	java.exe	208	lock wait	0	0	0.060086	0.040058	0.000020	0	UDBRS02	69	LOCAL	*LOCAL.DB...	2 452
PECART	java.exe	209	lock wait	0	0	0.030043	0.040057	0.000021	0	UDBRS02	65	LOCAL	*LOCAL.DB...	2 452
PECART	java.exe	210	lock wait	0	0	0.070101	0.140201	0.000023	0	UDBRS02	53	LOCAL	*LOCAL.DB...	2 452
PECART	java.exe	211	lock wait	0	0	0.160231	0.080115	0.000021	0	UDBRS02	76	LOCAL	*LOCAL.DB...	2 452
PECART	java.exe	212	lock wait	0	0	0.050072	0.050072	0.000023	0	UDBRS02	62	LOCAL	*LOCAL.DB...	2 452
PECART	java.exe	213	lock wait	0	0	0.150216	0.070101	0.000022	0	UDBRS02	86	LOCAL	*LOCAL.DB...	2 452
PECART	java.exe	215	lock wait	0	0	0.090130	0.080115	0.000021	0	UDBRS02	67	LOCAL	*LOCAL.DB...	2 452
PECART	java.exe	216	lock wait	0	0	0.100144	0.030043	0.000022	0	UDBRS02	64	LOCAL	*LOCAL.DB...	2 452
PECART	java.exe	218	lock wait	0	0	0.100144	0.070100	0.000020	0	UDBRS02	55	LOCAL	*LOCAL.DB...	2 452
PECART	java.exe	219	lock wait	0	0	0.040058	0.100144	0.000022	0	UDBRS02	73	LOCAL	*LOCAL.DB...	2 452
PECART	java.exe	221	lock wait	0	0	0.030043	0.120173	0.000023	0	UDBRS02	71	LOCAL	*LOCAL.DB...	2 452

Figure 6-33 Applications running

You can double-click the application to see further information about an application. In our particular case, we have an application that is in a deadlock, and many applications are showing lock wait. You should open one of the application details and analyze the lock wait scenario.

Figure 6-34 shows further application details of one of the application lists in lock wait in Figure 6-33 on page 383. As shown in Figure 6-34, we can see that there are more views selections, where `SQL statement and package`, and `SQL Activity` are of special interest to us. We will discuss this topic further when we demonstrate fixing our SQL queries and database configuration.

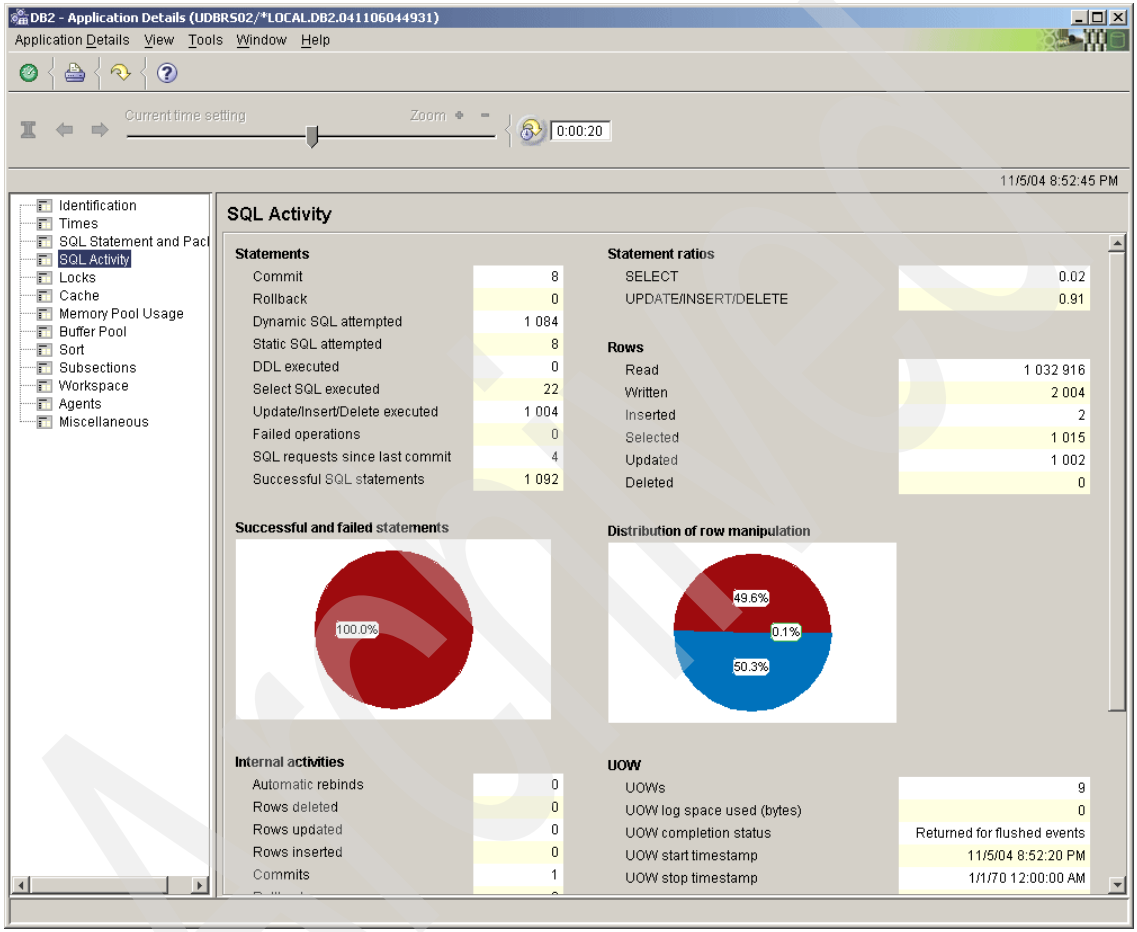
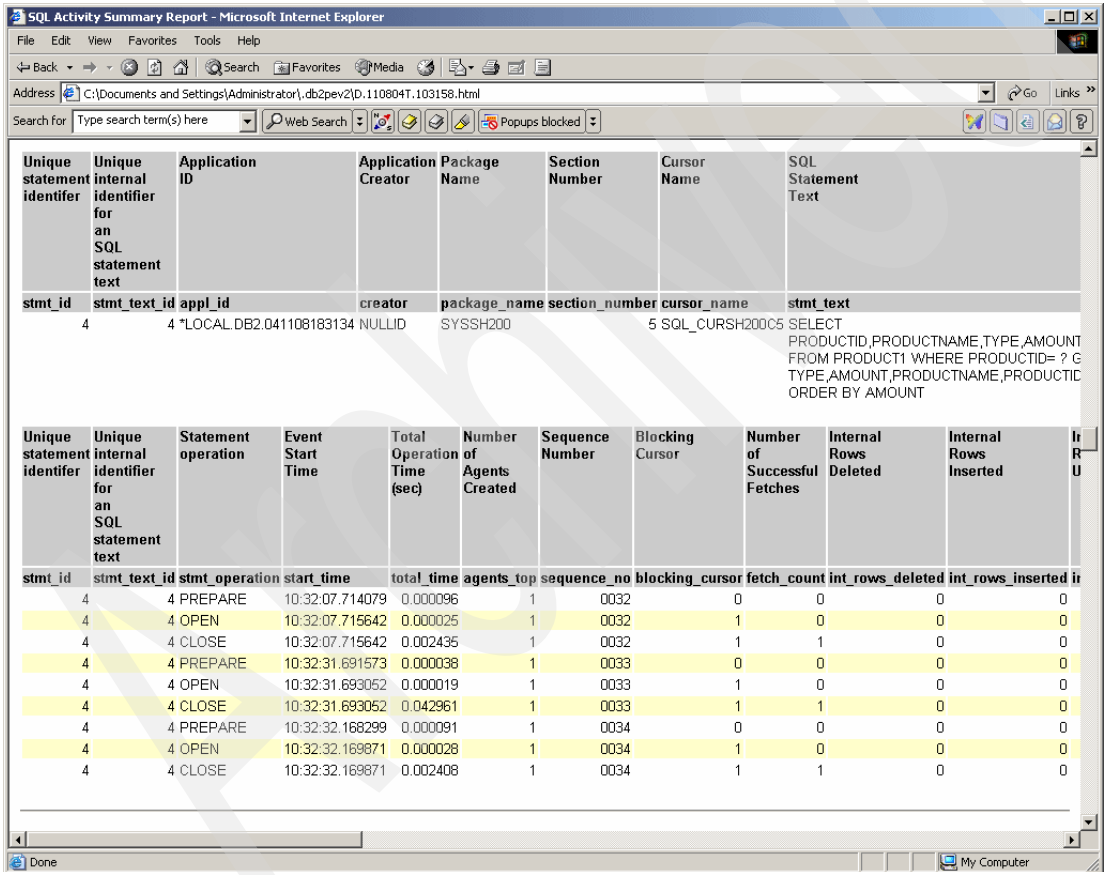


Figure 6-34 Application statement details

Another piece of data we may be interested in is *SQL Activity Tracing*, which provides a tracking information report about ongoing SQL activities. Select the application that is using maximum CPU time or executing maximum statements and select **Selected → SQL Activity Tracing**. This will launch the SQL Activity Tracing - Stop Condition window, in which you can set the trace time to 5 minutes. Clicking **OK** will start data collection from the ongoing activities.

The SQL report for our activities is shown in Figure 6-35. From the information in this report, we can see the various SQL statements our application launched and their details. This information will also be used in the SQL and database optimization step.

As shown in Figure 6-35, the SQL Report provides a run-time activity report of all the SQL statements ran for the selected application. This figure shows the part of the report for the query we are discussing here. The report can be used to see data, such as execution time for each statement, error or warnings, time for searching and fetching data, and so on. All this information can be used to search for statements that are working out slowly.


SQL Activity Summary Report - Microsoft Internet Explorer

File Edit View Favorites Tools Help

Back Search Favorites Media

Address C:\Documents and Settings\Administrator\.db2pev2\D.110804T.103158.html

Search for Type search term(s) here Web Search Popups blocked

Unique statement identifer	Unique internal identifier for an SQL statement text	Application ID	Application Creator	Package Name	Section Number	Cursor Name	SQL Statement Text
stmt_id	stmt_text_id	appl_id	creator	package_name	section_number	cursor_name	stmt_text
4	4	*LOCAL.DB2.041108183134	NULLID	SYSSH200	5	SQL_CURSH200C5	SELECT PRODUCTID,PRODUCTNAME,TYPE,AMOUNT FROM PRODUCT1 WHERE PRODUCTID= ? G TYPE,AMOUNT,PRODUCTNAME,PRODUCTID ORDER BY AMOUNT

Unique statement identifer	Unique internal identifier for an SQL statement text	Statement operation	Event Start Time	Total Operation Time (sec)	Number of Agents Created	Sequence Number	Blocking Cursor	Number of Successful Fetches	Internal Rows Deleted	Internal Rows Inserted	I R U
stmt_id	stmt_text_id	stmt_operation	start_time	total_time	agents_top	sequence_no	blocking_cursor	fetch_count	int_rows_deleted	int_rows_inserted	ir
4	4	PREPARE	10:32:07.714079	0.000096	1	0032	0	0	0	0	
4	4	OPEN	10:32:07.715642	0.000025	1	0032	1	0	0	0	
4	4	CLOSE	10:32:07.715642	0.002435	1	0032	1	1	0	0	
4	4	PREPARE	10:32:31.691573	0.000038	1	0033	0	0	0	0	
4	4	OPEN	10:32:31.693052	0.000019	1	0033	1	0	0	0	
4	4	CLOSE	10:32:31.693052	0.042961	1	0033	1	1	0	0	
4	4	PREPARE	10:32:32.168299	0.000091	1	0034	0	0	0	0	
4	4	OPEN	10:32:32.169871	0.000028	1	0034	1	0	0	0	
4	4	CLOSE	10:32:32.169871	0.002408	1	0034	1	1	0	0	

Done My Computer

Figure 6-35 SQL Report

If you are interested in long-term trend analysis, then you may be interested in the Performance Warehouse SQL analysis report. The Performance Warehouse provides long-term storage for SQL, buffer pool, and database activity data.

There are 25 predefined queries that may be run against the information stored in the Performance Warehouse. In our setup, we started the SQL Activity Trace Report execution and it has finished. You can check the status of the activity by selecting **Performance Warehouse - Expert** → **Mulitplatforms**, selecting the monitored instance, and clicking on **Process Execution**. The right hand pane should show a status of FINISHED. Click the activity and open the details by selecting **Selected** → **Details**, select **Report** row, and open it. It should show you the report.html as an Output file; open this file by clicking **Open** once again.

This will open the report as shown in Figure 6-36, which shows the statements part of the warehouse data collected report. It shows the details of each execution of the statements in the time frame in which performance warehouse tracing was executed.

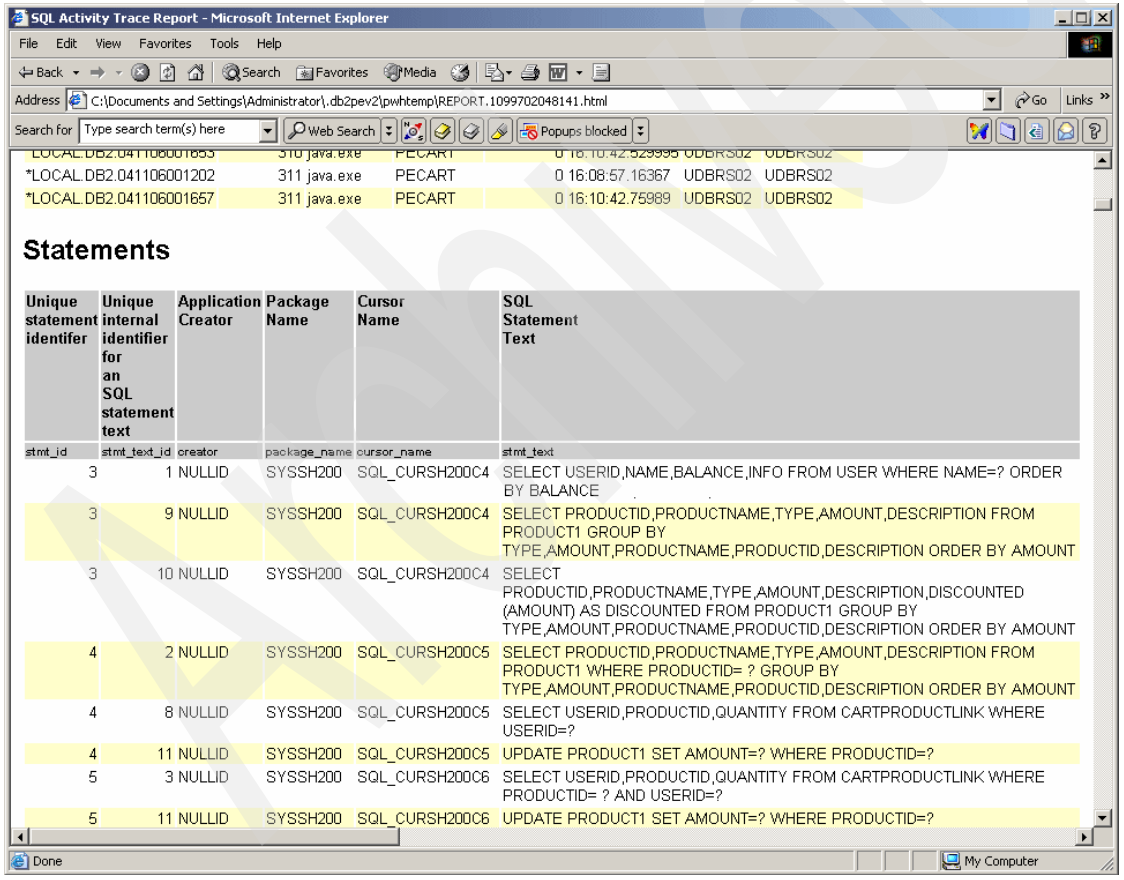


Figure 6-36 Performance warehouse SQL trace report

You can run further queries and Rules of Thumb on the collected data.

You can use the System Health data view configured in the last section for analyzing and quick views of system trends.

Fixing slow statement problems

In typical applications, at least 60% of execution time is spent on retrieving data from the database. This fact itself shows how important it is to tune the SQL statements to have an efficient application. In the previous section, we collected all the necessary data about the SQL statements running in our application. In this section, we use this data to tune our SQL statements, tables, and database.

In Figure 6-31 on page 380, we saw all our SQL statements sorted according to their execution times. This view is the most convenient view to find slow running queries, though all the above discussed methods can be used to find slow queries.

A query may be named a bad query because of a number of reasons. The queries most frequently executed without caching, queries with the longest average execution time, or queries showing maximum sorting are the best candidates for index analysis. Sometimes the queries run for long time, the long running queries are candidates for locking analysis. We will analyze all these queries one by one and try to figure out a solution for them.

Statements with high execution time and frequency

This can help you identify important statements that should likely be well-tuned. For statements that are executed frequently, a single execution may not place large demands on the system, but the cumulative result may degrade performance significantly. Analyzing these statements is really very important. As shown in Figure 6-31 on page 380, `update product1 set amount=? where productid=?` is the most frequently used statement, while `select productid,productname,type,amount,description from product1 where productid= ? group by type,amount,productname,productid,description order by amount` is most time consuming statement. We will analyze this slow statement here.

For this query, the SQL Activity details shown in Figure 6-34 on page 384 shows high read rows and low select, update rows.

The number of read rows for this query is 1032916, which is very high. This counter directly shows that this query has heavy data access. This is clearly very harmful, considering the fact that this is not the number of rows that had to be read in order to return the result set; instead, it is the number of rows read to find the result set elements. If we see selected rows, it is just 1015, which means that if we select every single result, the query reads 1000 rows. This clearly indicates that this query would be more efficient if we create additional indexes and remove the indexes that are not efficiently used.

In our scenario, this value shows that if query is used without any efficient index scans, it may also indicate that the index scan is very high, but the selectivity is very little or there is no selectivity.

To effectively maintain the indexes and analyze the access plan, DB2 Performance Expert uses the DB2 Visual Explain feature. Starting from DB2 Performance Expert Fix Pack 2, the statement details shown in Figure 6-32 on page 381 and the Application details SQL statement and package view shown in Figure 6-37 provides the DB2 Explain feature from within DB2 PE.

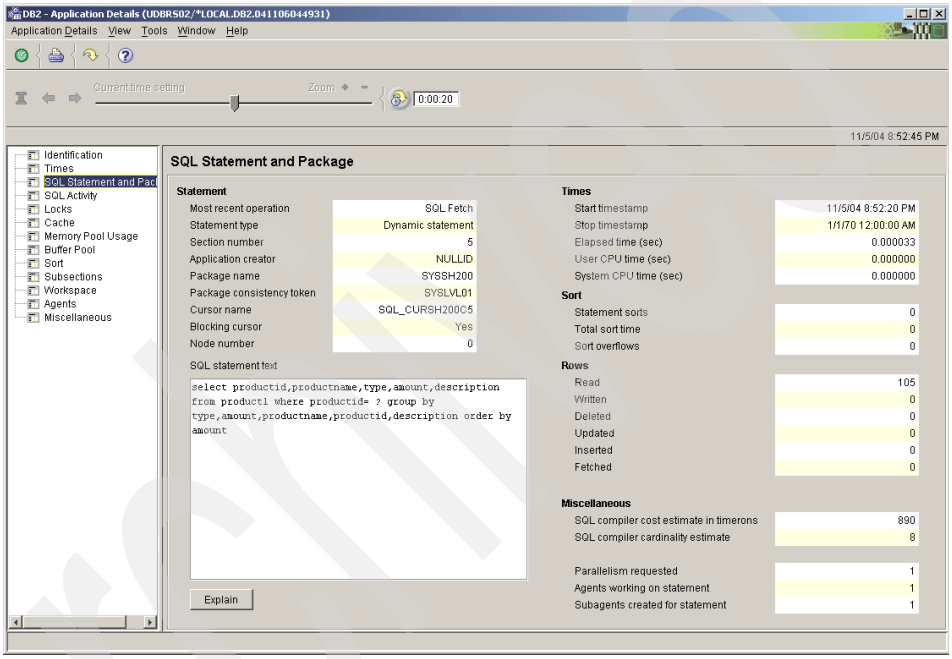


Figure 6-37 Statement and package details

Using the DB2 Explain feature

The DB2 Explain feature provided in DB2 Performance Expert is a versatile graphical feature to examine static and dynamic SQL statements. This features detailed information about the access plan that the optimizer chooses for a SQL statement. The DB2 optimizer can choose from a variety of different techniques as it creates optimal access paths for each SQL statement. These techniques range from a simple series of sequential reads to much more complicated strategies, such as using multiple indexes to access data.

To analyze the access path, follow the steps given below:

1. Launch DB2 Explain by clicking on the **Explain** button (Explain).

2. Input the database alias and click **OK**. In our case, the database name is PECART.
3. This will start fetching data from the explain table, as shown in Figure 6-38.

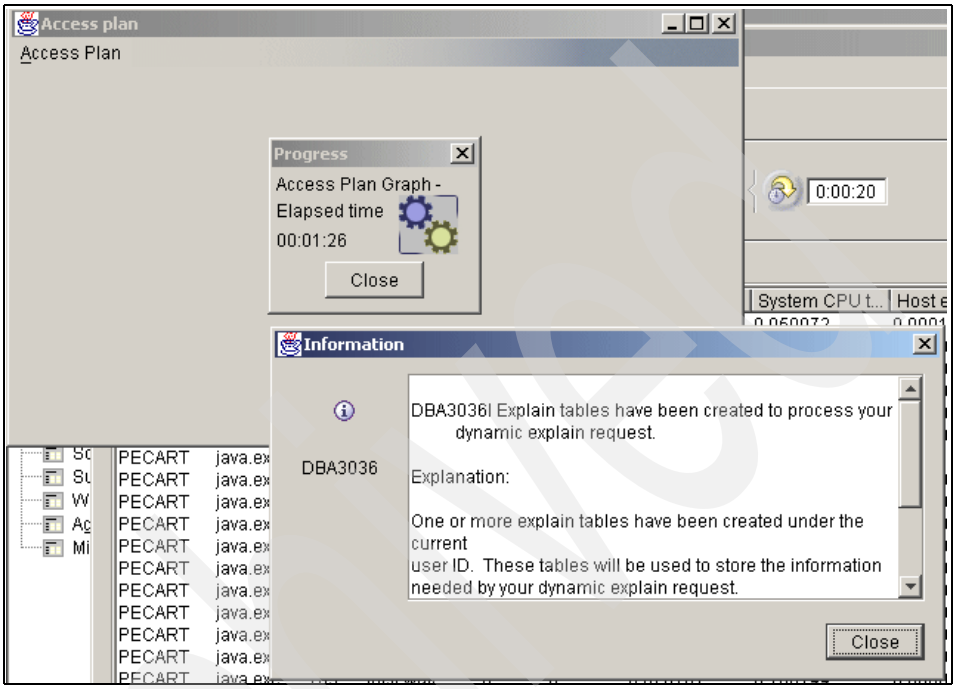


Figure 6-38 Explain execution

4. Finally, it will provide you with an access plan for your statement, as shown in Figure 6-39.

   Each access path operation is placed into a color-coded node in a tree structure. Each node may represent different data; typically, the nodes are:

   – Table
   – TBSCAN
   – IXSCAN, EISCAN, RIDSCN
   – RIDSCN
   – FETCH
   – Joins - NLJOIN, MSJOIN, HSJOIN, UNION, IXAND
   – GRPBY
   – FILTER
   – TEMP
   – RETURN

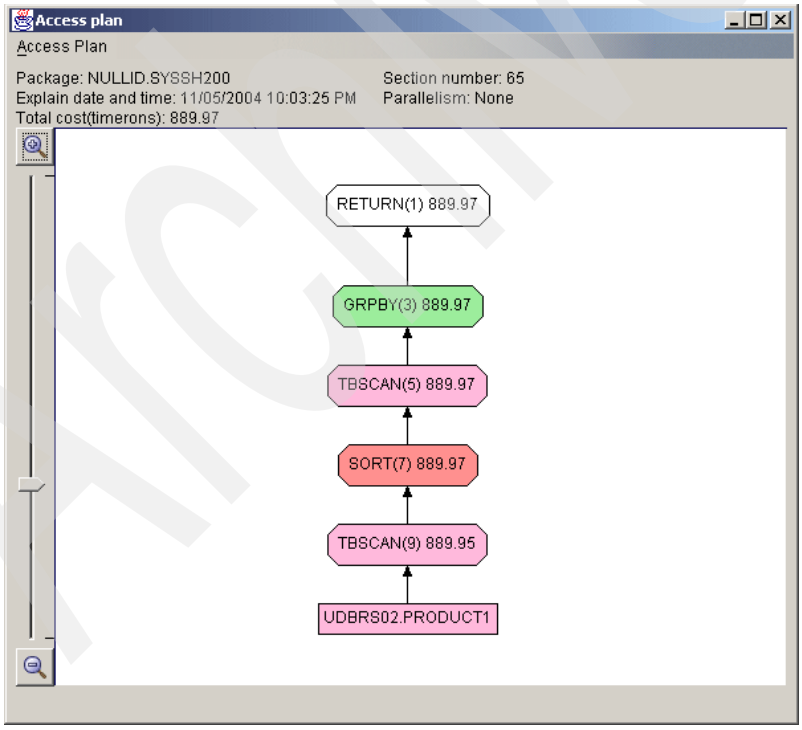


Figure 6-39 Access plan

5. Working from the bottom up, examining the entire plan may help you find specific inefficiencies. Clicking each node allows you to display the arguments to the operator, the estimated cost of the node, and the cumulative cost of all operations up to this point.

   In our particular case, no indexes are used. We can then use DB2 Design Adviser to determine the index for the query. If your application query has used indexes, but still has high rows read, you can still use DB2 Design Adviser to get the effective indexes for your application.

6. DB2 has considerable query rewrite capability. This puts the code into a more Optimizer friendly format. To view the optimized query used by DB2, select **Access Plan** → **Statement** → **Show Optimized SQL Text**. Figure 6-40 shows the optimized SQL Text for our query. This clearly indicates that our SQL query is far from optimized and our tables requires some indexes.

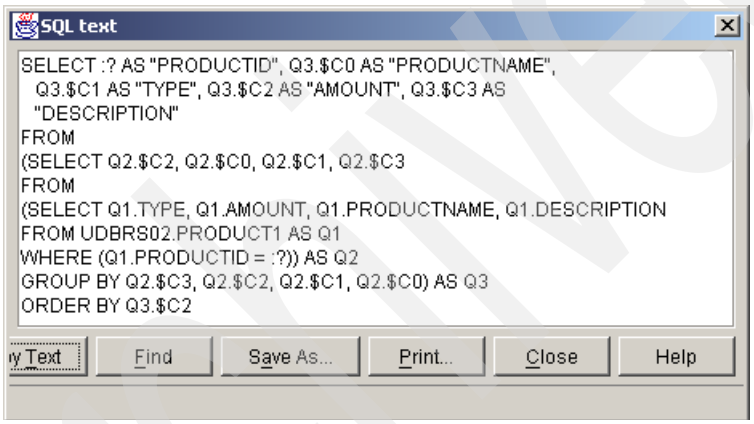


Figure 6-40 Optimized SQL

7. Another important thing our access plan shows two levels of table scan. As we know, index scan is always better then table scan. Open the details of table scan by double-clicking on the tbscan node. This will open the Operation details. Figure 6-41 shows the table scan detail of tbscan(9) node in our example. In this view, look out for the selectivity of predicates. It is 0.04, which means that the probability of getting a hit for each row you want to select in table scan is just four out of a hundred, which shows that if the index is created on the table, it will be highly successful.

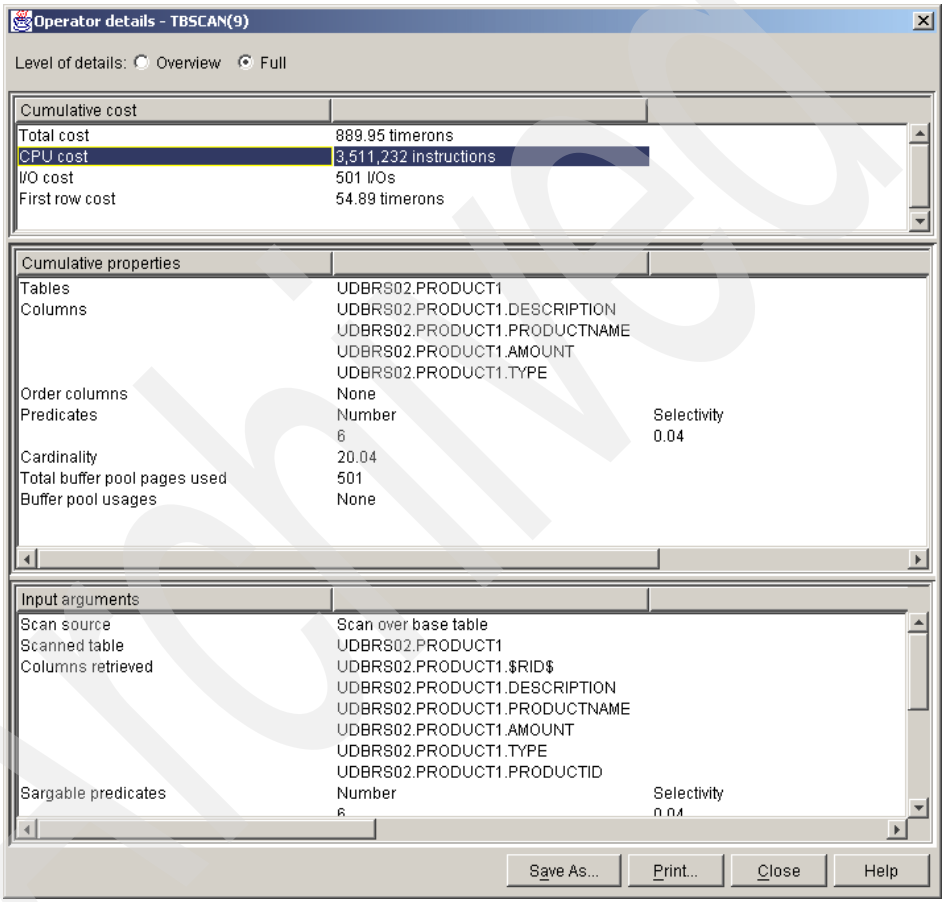


Figure 6-41 Operation details table scan

The high value of rows read in the SQL activities details also indicates table scans are occurring for this select statement.

As indicated by all the analysis above, we definitely need to create indexes on the PRODUCT1 table for effective usage. To decide what index would work

efficiently for this table, we use DB2 Design Advisor. The DB2 Design Advisor is a tool to assist you in choosing an optimal set of indices for your table. It can be used to evaluate current indexes or to get an index recommendation for a particular set of SQL statements.

Example 6-13 shows the usage of db2advis for getting recommendations for our query. As predicted by DB2 Performance Expert, db2advis also indicated that creating an index on PRODUCT1 table will give 80% improvement. So we create an index on productid for the PRODUCT1 table.

Example 6-13 DB2 Design Advisor

```
C:\pesample>db2advis -d pecart -t 5 -s "select
productid,productname,type,amount
,description from product1 where productid=?  group by
type,amount,productname,p
roductid,description order by amount"

Using user id as default schema name. Use -n option to specify schema
execution started at time stamp 2004-11-08-17.34.08.060000
Recommending indexes...
total disk space needed for initial set [   0.017] MB
total disk space constrained to         [   5.516] MB
Trying variations of the solution set.
Optimization finished.
  1  indexes in current solution
 [890.0000] timerons  (without recommendations)
 [178.0000] timerons  (with current solution)
 [80.00%] improvement

--
-- LIST OF RECOMMENDED INDEXES
-- ===========================
-- index[1],    0.017MB
   CREATE INDEX "UDBRSO2 "."IDX411090134100000" ON "UDBRSO2 "."PRODUCT1"
("PRODU
CTID" ASC) ALLOW REVERSE SCANS ;
   COMMIT WORK ;
   RUNSTATS ON TABLE "UDBRSO2 "."PRODUCT1" FOR INDEX "UDBRSO2
"."IDX411090134100
000" ;
   COMMIT WORK ;

--
-- RECOMMENDED EXISTING INDEXES
-- ============================

--
-- UNUSED EXISTING INDEXES
```

```
-- ============================
-- ============================
--

7 solutions were evaluated by the advisor
DB2 Workload Performance Advisor tool is finished.
```

Very slow or halted SQL statements

Halted SQL statements is another important set of queries to look for. These are the queries where an application is showing a status of executing or lock wait for a extensive period of time. In the Application Summary view, you can traverse through history to check if an application is waiting for lock or has been executing for a long time. Figure 6-42 shows one such scenario (application ID 64) in which an application is waiting for a lock for quite some time.

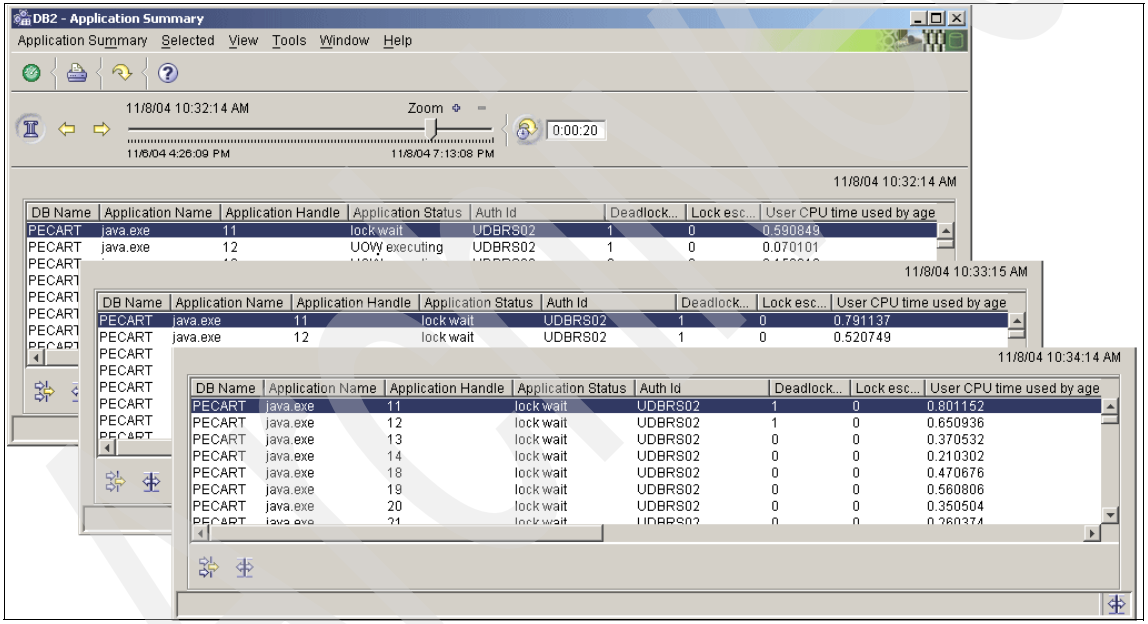


Figure 6-42 Lockwait halted application

The Application Details showed that it is waiting for the lock to execute `update product1 set amount=? where productid=?`. To view the details of the lock, we looked into the applications in lock conflict. Figure 6-43 on page 395 shows the step by step analysis of the applications in lock conflict. It shows that this application is waiting for an exclusive row lock on the PRODUCT1 table, and this lock is locked by another application executing the `update product1 set amount=? where productid=?` statement.

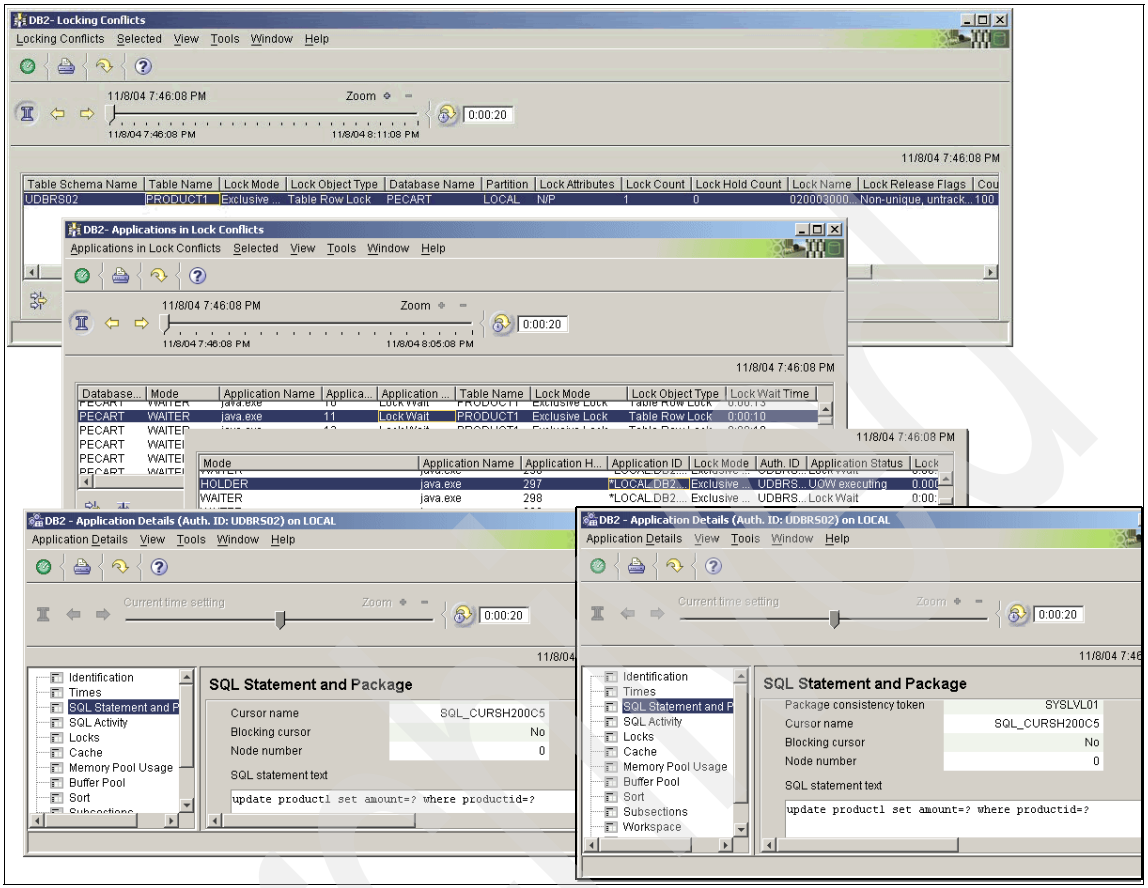


Figure 6-43 Applications in lock conflicts

The application may need to be modified to lock rows for lesser amount of time. We looked into two methods, updateproduct() and discountproduct(), which use this statement. The original code for these two methods is shown in Example 6-14.

Example 6-14 updateproduct and discountproduct source code

```
   public static void updateproduct()throws Exception{

         Connection con=null;
         try{
         con=getConnection();
         con.setAutoCommit(false);

//       PreparedStatement stat2=con.prepareStatement("select
productid,productname,type,amount from product1 group by
type,amount,productname,productid order by amount");
         PreparedStatement
stat2=con.prepareStatement(resource.getString("query15"));
//       PreparedStatement stat3=con.prepareStatement("update product1 set
amount=? where productid=?");
         PreparedStatement
stat3=con.prepareStatement(resource.getString("query8"));
         ResultSet res2=stat2.executeQuery();
         while(res2.next()){
            int amount=res2.getInt("amount");
            int prodid=res2.getInt("productid");
            stat3.setDouble(1,amount+1);
            stat3.setInt(2,prodid);
            stat3.executeUpdate();
         }
         if(show)System.out.println("Products updated");
         con.commit();
         stat2.close();
         stat3.close();

         }catch(Exception ex){System.out.println("connection
was"+con);System.out.println("connection was"+con);ex.printStackTrace();}
         returnConnection(con);
   }



   public static void discountproduct()throws Exception{

         Connection con=null;
         try{
         con=getConnection();
         con.setAutoCommit(false);
```

```
//        PreparedStatement stat2=con.prepareStatement("select
productid,productname,type,amount from product1 group by
type,amount,productname,productid order by amount");
          PreparedStatement
stat2=con.prepareStatement(resource.getString("query14"));
//        PreparedStatement stat3=con.prepareStatement("update product1 set
amount=? where productid=?");
          PreparedStatement
stat3=con.prepareStatement(resource.getString("query8"));
          ResultSet res2=stat2.executeQuery();
          while(res2.next()){
             double amount=res2.getDouble("discounted");
             int prodid=res2.getInt("productid");
             stat3.setDouble(1,amount);
             stat3.setInt(2,prodid);
             stat3.executeUpdate();
          }
          if(show)System.out.println("Product update started wait for some
time\n\n\n\n\n");
          Thread.sleep(100);
          con.commit();
          stat2.close();
          stat3.close();
          }catch(Exception ex){System.out.println("connection
was"+con);ex.printStackTrace();}
          returnConnection(con);
    }
```

This clearly shows that until all the update statements are not run, the transaction is not committed. The number of commits for this SQL activity is just eight, while this SQL statement has been attempted 1084 times so far. This small rate of commits shows that the application is not frequently committed and this might lead to data concurrency or logging problems. This is causing lot of lock waits. We will check the application to see if we can commit the transaction more frequently. As in our application, setting the product discount price and increasing the product price does not need to be done in one transaction. As all the products are independent of each other, we moved the transaction commits to inside the loop, which means the transactions will not lock the row forever and more concurrency is achieved.

But in many cases, the transaction cannot be committed until all the products are changed, in which case, the only optimization we can do is instead of the two queries `select productid,productname,type,amount,description from product1 group by type,amount,productname,productid,description order by amount` and `update product1 set amount=? where productid=?`, we can use `update product1 p1 set amount=(select amount from product1 where productid=p1.productid)+1`, which will minimize the time the application spends between the update and reduce query calls.

Deadlocks

A deadlock is a situation where two or more competing transactions are waiting for the other to finish, so neither one ever does. This lock contention problem has no solution, except one transaction has be terminated and rolled back.

These problems could be caused by the following situations:

- Lock escalation is occurring in the database.
- An application may be locking tables explicitly when system-generated row locks may be sufficient.
- An application may be using an inappropriate isolation level when binding.
- Catalog tables are locked for repeatable read.
- Applications are getting the same locks in different orders, resulting in deadlocks.

As we can see in Figure 6-44 on page 399, we have a few deadlock events while running this application. This also explains the few `SQLExceptions SQL0911N` in Example 6-12 on page 377, which shows that at least few deadlocks had occurred.

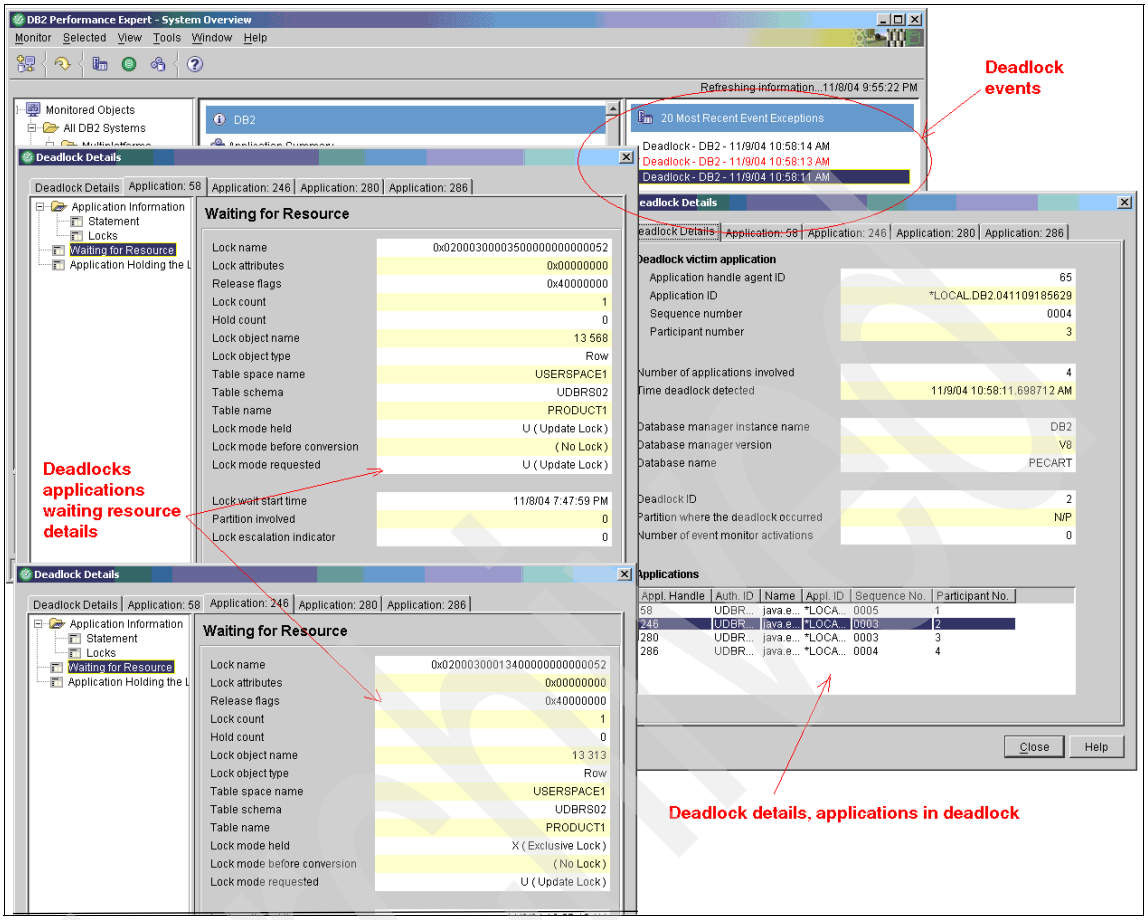


Figure 6-44 Deadlock events

Double-click these events to open the deadlock details, which will show you the applications in this deadlock and other details. If we look into each application, we found that all these applications are waiting for the running statement `update product1 set amount=? where productid=?` and that running this statement requires a row update lock on the PRODUCT1 table.

As we know that this activity is required and runs frequently, we cannot change the application code. We look at the database setting to see if we can tune the database for our application. We check the following database configuration parameters:

- DLCHKTIME: Time interval for checking deadlock dlchktime

  The time interval for checking the deadlock dlchktime DLCHKTIME database configuration parameter is set to lower then required. In our scenario, we have DLCHKTIME set to 10000, which is the default, and for the current application code, we may have to increase this. We increased it to 20000.

- LOCKLIST: Maximum storage for lock list

  Maximum storage for lock list is another parameter that may require tuning here. It indicates the amount of storage that is allocated to the lock list. There is one lock list per database, and it contains the locks held by all applications concurrently connected to the database. If the lock list is full, performance can degrade, since lock escalation will generate more table locks and fewer row locks, thus reducing concurrency on shared objects in the database. This means more applications can go to deadlock. In our scenario, this value is set to 500. It is recommended that this value should be set equal to (512 * *y* * maxappls)/4096, where y is the number of bytes required for the first lock against an object (72 bytes on 32-bit platforms, 112 bytes on 64-bit platforms). For our scenario, running 100 connections, it should be set to (512*72*100)/4096= 900. So we set this value to 500 now.

- MAXLOCKS: Maximum percent of lock list before escalation

  Lock escalation is the process of replacing row locks with table locks, reducing the number of locks in the list. This parameter defines a percentage of the lock list held by an application that must be filled before the database manager performs escalation. When the number of locks held by any one application reaches this percentage of the total lock list size, lock escalation will occur for the locks held by that application. Lock escalation also occurs if the lock list runs out of space. If MAXLOCKS is set too high, a few applications can consume most of the lock space, and other applications will have to perform lock escalation, which means more chance of a deadlock. For our scenario, this value is set to 22. The recommended value for maxlocks is 2 * 100 / maxappls, which means our value should be 2.

User defined function and stored procedures

Another query that was showing a high average time for execution, was `select productid,productname,type,amount,description,discounted(amount) as discounted from product1 where productid>100 and productid<110 group by type,amount,productname,productid,description order by amount`. As we can see in Figure 6-31 on page 380, the average time for execution of this query

(third from top) is 35 seconds. This is a huge amount of time, considering the environment and the fact that similar queries take less time.

While analyzing this query, it also showed the same index problems. But this query still had a really bad execution time and high CPU usage, even after fixing the indexes for the table. The only difference between this query and the previous query we analyzed was the user defined functions.

Analysis of this user defined function found that it was doing many activities. The key point is that if your query is showing very high execution time and it used UDF, make sure that the UDF does only the minimum required activities, as these activities can load the DB2 server heavily. For example, our UDF code was creating thousands of String objects in it and its thread was sleeping at the server end. This UDF was written just to show that how UDFs can impact performance.

High CPU usage and sorting

Another thing to notice in Figure 6-31 on page 380 is the query `select userid,name,balance,info from user where name=? order by balance`. If we look closely the data, it shows very high CPU usage. This clearly shows that something is really wrong. Analyzing the statistics details for this SQL statements shows that the number of sorts is very high. Figure 6-45 on page 402 shows the dynamic SQL statement statistics details, which shows that for 855 executions, it does 855 sorts, which means every execution requires sorting. This explains why the CPU usage for this statement is so high. Also, the rows read for this statement is very high. Both these conditions makes it a nice candidate for analyzing the access path.

Analyzing the access path using DB2 Design Adviser shows that this query will do 80% better if we create an index on columns NAME and BALANCE for table USER.

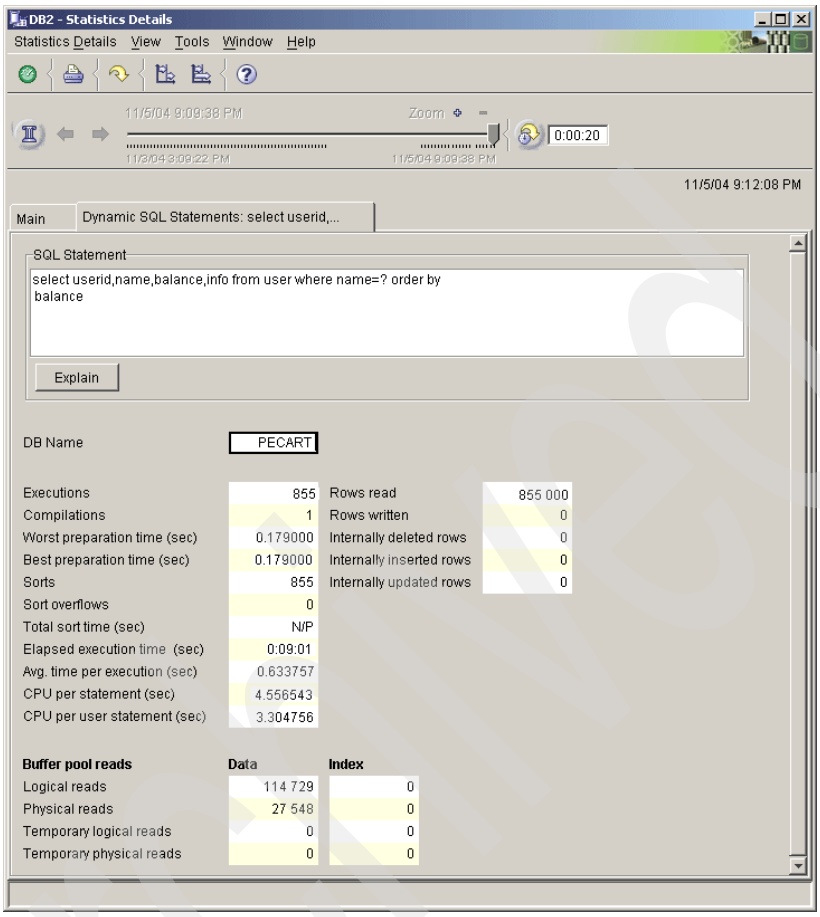


Figure 6-45 Sorting Statistics Details

Now, finally, after fixing all the problems discussed above, we rerun the same shopping cart application with fixed indexes, database parameters, and source code. Example 6-15 show the steps for running the fixed application and results for the application.

Example 6-15 Shopping cart application again

```
1. copy DiscountRateFix.class to your db2/sqllib/function directory
2. run startupfix.bat which includes following commands

   db2 connect to pecart

   db2 CREATE FUNCTION discountedfix(DOUBLE) RETURNS DOUBLE EXTERNAL NAME
   'DiscountRateFix!discount' SCRATCHPAD 10 FINAL CALL VARIANT NO SQL PARAMETER
   STYLE DB2GENERAL LANGUAGE JAVA NO EXTERNAL ACTION
```

```
   db2 update db cfg for pecart using LOCKLIST 1000
   db2 update db cfg for pecart using dlchktime 20000
   db2 update db cfg for pecart using maxlocks 2

   db2 CREATE INDEX UDBRS02.IDX411092216040000 ON UDBRS02.USER ("NAME" ASC,
   "BALANCE" ASC) ALLOW REVERSE SCANS

   db2 RUNSTATS ON TABLE UDBRS02.USER FOR UDBRS02.INDEX IDX411092216040000

   db2 CREATE INDEX UDBRS02.IDX411092217010000 ON UDBRS02.PRODUCT1
   ("PRODUCTID" ASC) ALLOW REVERSE SCANS

   db2 RUNSTATS ON TABLE UDBRS02.PRODUCT1 FOR INDEX UDBRS02.IDX411092217010000
   db2 CREATE INDEX UDBRS02.IDX411092218140000 ON
   UDBRS02.CARTPRODUCTLINK("USERID" ASC, "PRODUCTID" ASC, "QUANTITY" ASC) ALLOW
   REVERSE SCANS

   db2 RUNSTATS ON TABLE UDBRS02.CARTPRODUCTLINK FOR INDEX
   UDBRS02.IDX411092218140000

   db2 commit work
   db2 disconnect pecart

   db2stop
   db2start

3. set classpath=.\pecart.jar;%classpath%

4. Now execute run.bat which includes following commands
   set classpath=.\pecart.jar;%classpath%
   java com.ibm.db2pe.sample.DBRunMTFix -dbname pecart -conn 100 -time 5
   -showdetails

5. This will show the results as
   Total create time for 2144 activities is 16913sec Average=7sec
   Total update time for 2241 activities is 494sec Average=0sec
   Total delete time for 2215 activities is 355sec Average=0sec
   Total view time for 843 activities is 205sec Average=0sec
   Total updateproduct time for 148 activities is 8791sec Average=59sec
   Total discountproduct time for 177 activities is 465 Average=2sec
   Workload completed press any key with enter to close all connections and
exit
```

Example 6-15 on page 402 clearly shows that the activities that were taking 34 seconds initially are now taking less then 1 second. (Except for update and create queries, which are showing improvement but can be still improved.)

6.2.2 Responding to Performance Expert alerts

DB2 Performance Expert can be configured to create periodic exception alerts in the form of a log, a pop-up message, a sound, and an e-mail. In this scenario, we present a case where an e-mail alert has been received by a DBA and we show how PE can be used to investigate. Configuring periodic exception alerts is described in 4.2.1, “Periodic exception processing” on page 195.

There are a huge number of options in creating periodic exceptions, but we will not cover all of them. This scenario shows one of each type: Application, Statistics, and System Parameters.

Description of the application

We contrived a small Java program to perform some inefficient SQL inserts to a small table.

Environment configuration

The lab environment for this scenario is:

- Database server: jamaica, AIX 64-bit
- Instance: db2inst2, 64-bit UDB V8.1.6
- Database: LOADDB
- DB2 Performance Expert server: jamaica, AIX 64-bit PE V2.1.1
- PE instance: db2in8pe, 32-bit UDB V8.1.6

DB2 monitor settings

The instance default monitor switch settings are shown in Example 6-16.

Example 6-16 jamaica db2inst2 default monitor switches

```
$ db2 get dbm cfg | grep DFT_MON
   Buffer pool                          (DFT_MON_BUFPOOL) = ON
   Lock                                    (DFT_MON_LOCK) = ON
   Sort                                    (DFT_MON_SORT) = ON
   Statement                               (DFT_MON_STMT) = ON
   Table                                  (DFT_MON_TABLE) = ON
   Timestamp                            (DFT_MON_TIMESTAMP) = ON
   Unit of work                               (DFT_MON_UOW) = ON
$
```

DB2 Performance Expert settings

The PE settings relevant to this scenario are:

- Periodic exception retrieval interval: 60 seconds (see Figure 6-46 on page 405).

- Periodic exception activation interval multiplier: 1 (see Figure 6-47 on page 406).
- Threshold set:

  We created a threshold set with a single application threshold, which looks for more than 100 commits per second in the LOADDB database. This is a very contrived value and should not be taken as any kind of recommended threshold. We used it here as an easy way to induce the state we needed for this scenario. The name of the Threshold Set is “Too Many Commits (see Figure 6-48 on page 406).

  Note that the number of commits counter we chose was “Statements Commit”, not “Internal activities commits.” These are two different snapshot counter values and do not report the same information. When you choose your thresholds, you should be sure you understand what each counter will return. See 4.2.1, “Periodic exception processing” on page 195 for a discussion about PE periodic exception processing.

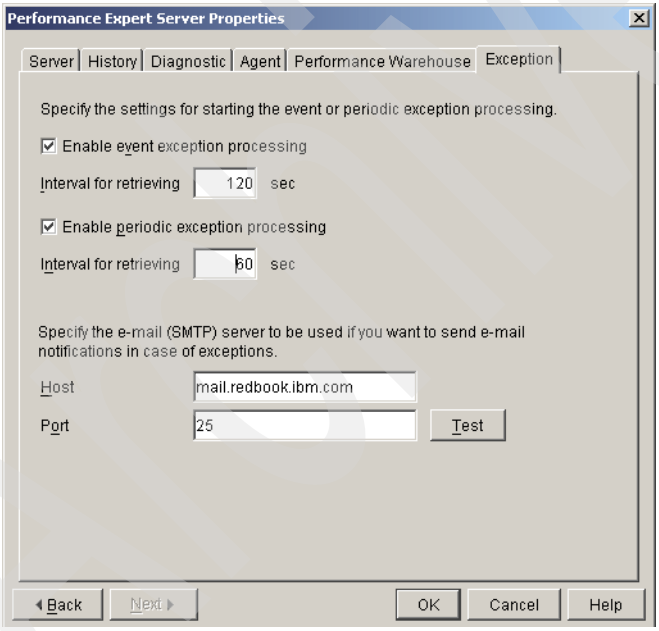


Figure 6-46 Alert scenario - PE Server settings

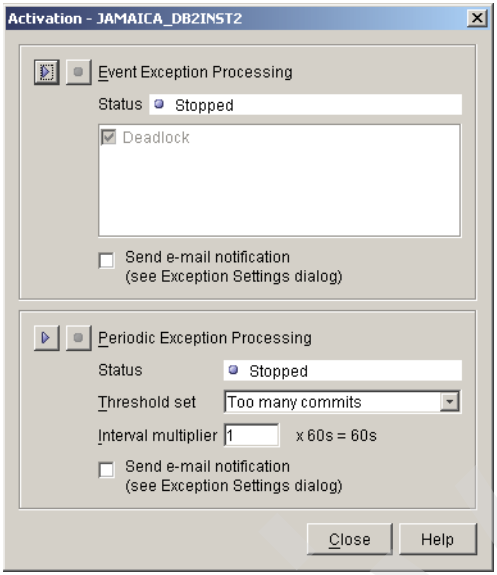


Figure 6-47 Alert scenario - Exception interval multiplier

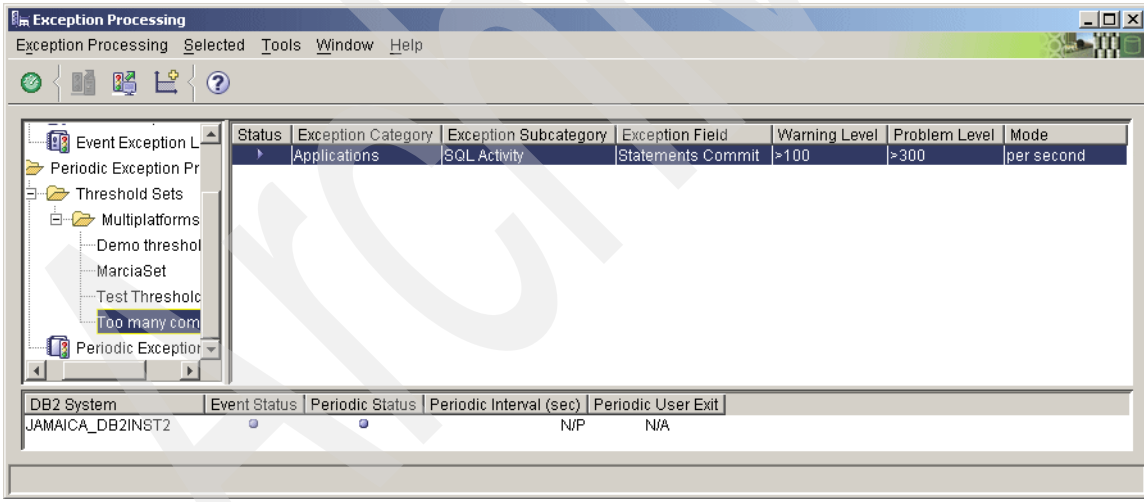


Figure 6-48 Alert scenario - Threshold set for commits

Workload used

We modified the sample Java program found in the article “Putting your DB2 Database to the Test: Measuring Performance”, which can be found at http://www.ibm.com/developerworks/db2/library/techarticle/0303bhogal/0303bhogal.html to accept input parameters, and renamed it ExceptionPopulator, as shown in Example 6-17.

Example 6-17 ExceptionPopulator.java

```
//*  package com.ibm.jmetertest;
import java.sql.*;

public class ExceptionPopulator
{
  public static void main(String[] args)
  {
    try
    {
      Class.forName("COM.ibm.db2.jdbc.app.DB2Driver");
   String dbName = "jamload";
   String url = "jdbc:db2:" + dbName;
   String userName = "db2inst2";
// pw masked for print
   String password = "********";
   Connection con
          = DriverManager.getConnection(url,userName, password);
   int numOfTestRecords = 50000;
   PreparedStatement statement
         = con.prepareStatement("INSERT INTO MARCIA.LOADTABLE VALUES(?,?,?)");
   for (int i =0; i<numOfTestRecords; i++)
   {
     statement.setString(1,"Col1Test-" + i);
     statement.setString(2,"Col2Test-" + i);
     statement.setString(3,"Col3Test-" + i);
     statement.executeUpdate();
     // System.out.println(i);
   }
    }
    catch(Exception e)
    {
       System.out.println("An error has occurred: " + e.toString());
       e.printStackTrace();
    }
  }
}
```

Triggering event

The DBA arrived Monday morning and found an e-mail from Performance Expert indicating a threshold had been exceeded. The e-mail is shown in Example 6-18.

Example 6-18 E-mail alert from DB2 Performance Expert

```
AlertFromUDBRS01@ibm.com 11/07/2004 01:36 PM


Subject
IBM (C) DB2 Performance Expert.




A notification from UDBRS01 machine

IBM (C) DB2 Performance Expert.

DB2 threshold violation was detected at [Nov 7, 2004, 1:35:53 PM PST] on
DB2 system 'JAMAICA_DB2INST2'.

Details :

     Category    : Applications
     Subcategory : SQL Activity
     Counter     : Statements Commit
     Severity    : WARNING
     Owner       : MARCIA

     Current value :  145.781
     Warning level :  100.0
     Problem level :  300.0

     Violator :
     Application ID                : G9013B89.IA12.041107213401
     Agent ID                      : 306
     Application connection time   : 2004-11-07 13:35:21.000367
     Authentification ID           : DB2INST2
     Database name                 : LOADDB
     Database path                 :
/home/db2inst2/db2inst2/NODE0000/SQL00003/
     Application status            : commit active
     Partition number              : 0
     Partition name                : PART0
```

Investigation without recreating the problem

The e-mail tells us the key information for investigating the threshold violation. We know that the violation was detected on jamaica instance db2inst2 at 1:35 PM on Sunday, November 7, 2004, where PE found an application had done 145 commits per second. The rate of 145 commits only exceeds the warning threshold, but we decide to investigate anyway.

Note: The value of commits per second is shown in the e-mail as 145.781. This is because the value is calculated as a *rate*, based on the application start time and commits done up to the point of the snapshot.

Since this is an Application threshold, the Application ID is provided in the e-mail, as well as when the application started. This should be enough for us to find the application in the PE history.

Important: The time of violation detection will not be exactly the moment when the threshold was exceeded. Performance Expert uses data returned in snapshots to evaluate the threshold, and the frequency of the snapshot is configured when you activate it. If you choose a less frequent snapshot, say five minutes, the actual violation may have occurred four minutes earlier than the snapshot and thus before the violation detection time. Be aware of your exception processing interval as you investigate any alerts; you may need to move back through the history screens to find more information about the offending incident.

Armed with the key pieces of information, we open the PE Client and log on to the PE Server monitoring jamaica db2inst2, then bring up the Application Summary window. We move into history mode and position it to around 1:35 PM on November 7. We know the application started at 1:35 PM, but we wonder if the application stayed alive very long, so we move forward a bit more in history, and see that it was still there at 1:40 PM. We drill down into the Application Details for the application and go to the SQL Activity page shown in Figure 6-49.

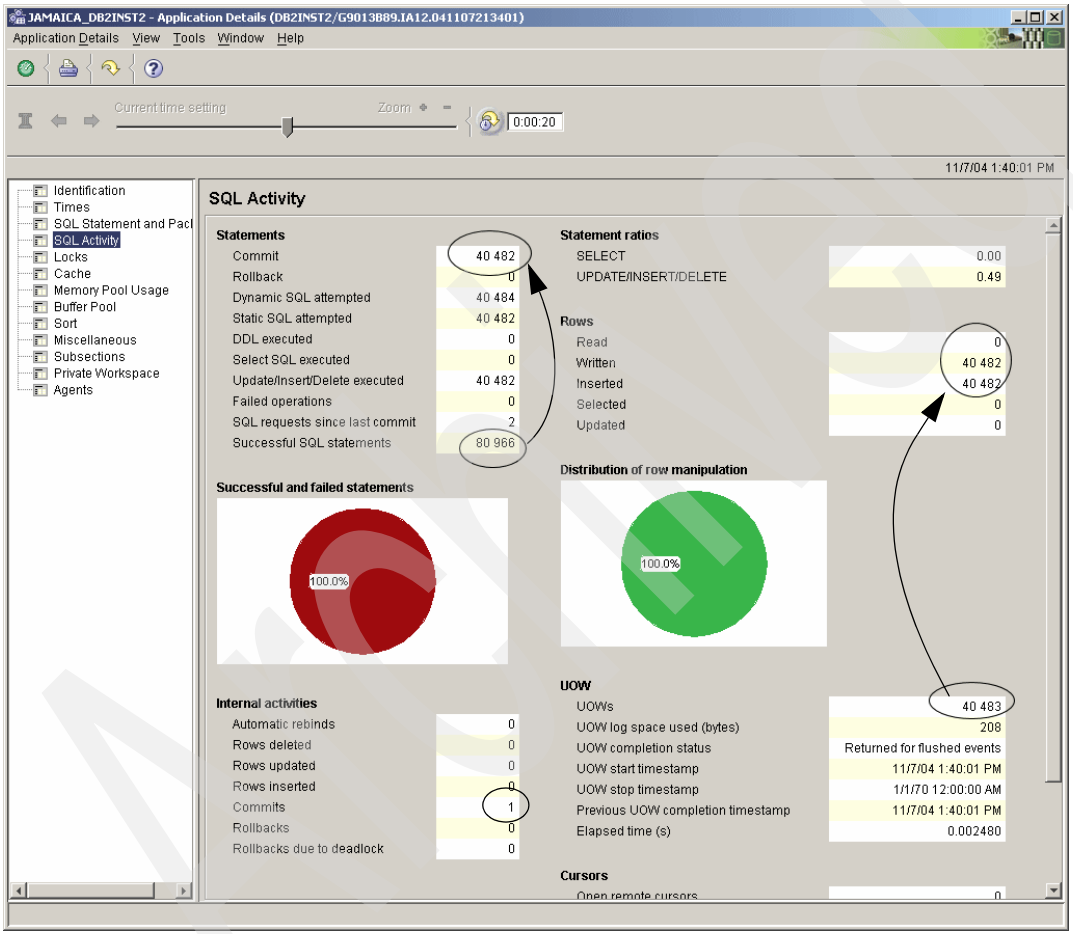


Figure 6-49 Alert scenario - Application Details - SQL Activity

The SQL Activity page shows quite a lot of information, so that we can already deduce what is going on even without seeing the code or the statement. The fact that we see 40,482 commits and 40,482 rows inserted probably means someone is issuing inserts and committing after each one. This is further validated by

noticing the Successful SQL statements value is 80,966, which is very close to the Commits + Inserted.

When we looked at the SQL Statement page, the statement text field was empty, and “Most recent operation” was a commit, so that page does not give us much useful information.

We have a theory now that someone is doing an inefficient load, so we would like to know who, so we can talk with them. The e-mail said the ID was db2inst2, but that is the instance owner name, so we would like to know a bit more. We open the Identification page, shown in Figure 6-50. In this window, we see the Client operating platform was NT (this value is returned for any Windows platform), and the user logon ID was MARCIA. Now we know this program came from a Windows platform and we can talk to Marcia to discover what she was doing.

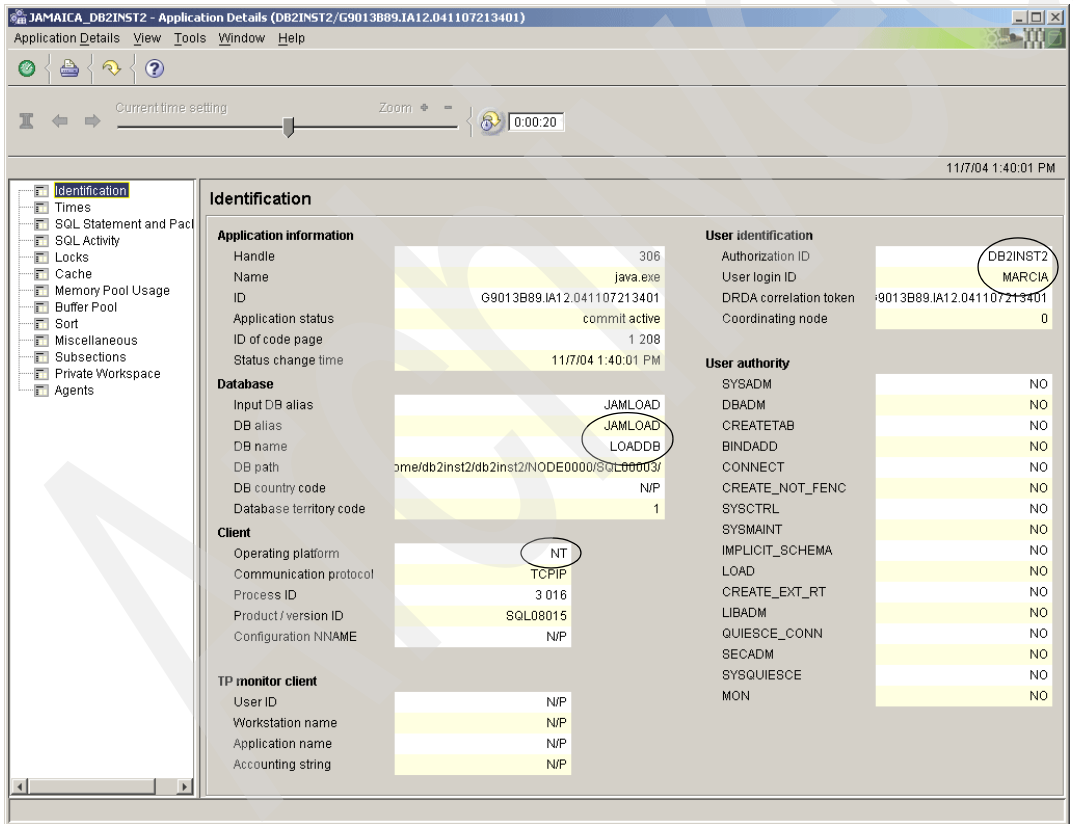


Figure 6-50 Alert scenario - Application Details - Identification

If we did not recognize the user or identification information, we could also look at the Miscellaneous page, shown in Figure 6-51, which shows the IP address of the client. This could help narrow down where the application came from.

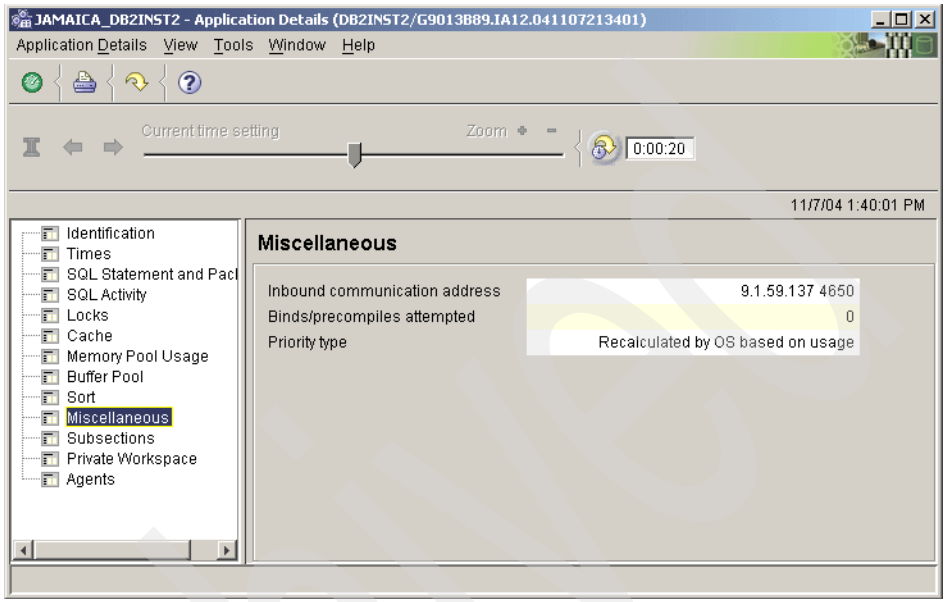


Figure 6-51 Alert scenario - Application Details - MIscellaneous

Root cause of the problem

We talked to Marcia and found she was experimenting with a Java program she found on the Web. She meant to run it against her test database, but mistakenly used the wrong alias name, so that is why the alert triggered on our LOADDB database. She was already making corrections to the program to be more efficient and had fixed it to use the correct database. We were satisfied that the situation was under control.

Compare this scenario to the one in 6.2.4, “Transaction log full” on page 421, where a user’s update statement caused a log full condition. These are two problems with a potentially similar cause - incorrect commit frequency - but different results. The ExceptionPopulator program did not cause any log full conditions, and really did not cause any real problems other than triggering the threshold.

Best practices

Determining the optimum commit frequency for most applications can be an art. In this particular scenario, if the user wanted to load a large amount of data into a table, using individual inserts was about the worst way to go about it. They should have used LOAD or IMPORT. If it had to be done in a program, then the program should be written so it issues less frequent commits. This, of course, would have to be balanced with the log sizes so a log full condition does not occur, as in 6.2.4, “Transaction log full” on page 421.

6.2.3 Another exception scenario

In the IBM Redbook *DB2 UDB Performance Expert for Multiplatform: A Usage Guide*, SG24-6436, there was a scenario that dealt with sort overflows and using PE to investigate the problem. We present a revised and abbreviated version of this scenario to highlight the newer features in Performance Expert V2 that will make the investigation easier. In the PE V1 book, the sort overflow was detected by noticing that the System Health chart graph had spiked, and then a SQL Activity Trace report was run while the bad queries were executed again. After collecting the SQL statement, we opened the DB2 Control Center to invoke a Visual Explain for more analysis. In this scenario, we examine how to use PE V2 to investigate a similar situation without having to rerun the query and how to directly invoke the Explain function.

Description of the application

This scenario uses the TRADE2 application (also known as WebSphere Performance Benchmark Sample) and a collection of several “uncommitted” batch jobs with SQL INSERTs, UPDATEs, and SELECTs to cause the problem situations to occur.

Environment configuration

The lab environment for this scenario is:

- Database server: henwen, AIX 64-bit
- Instance: db2inst8, 64-bit UDB V8.1.7
- Database: TRADEDB
- DB2 Performance Expert server: henwen, AIX 64-bit PE V2.1.2
- PE instance: db2in8pe, 64-bit UDB V8.1.7

DB2 monitor settings

The instance default monitor switch settings are the same as in the previous example shown in Example 6-16 on page 404.

DB2 Performance Expert settings

The PE settings relevant to this scenario are:

- Periodic exception retrieval interval: 60 seconds (see Figure 6-46 on page 405).
- Periodic exception activation interval multiplier: 1 (see Figure 6-47 on page 406).
- Threshold set: We created a threshold set with a single application threshold, which looked for a sort overflow of more than 1. Again, this is a contrived example simply so we can generate alerts. The name of the threshold set is Sort Overflows (see Figure 6-52).

Figure 6-52 Alert scenario 2 - threshold set for sort overflow

Workload used

To generate the information we are interested in for the alert, we generated three queries against the tables in the TRADEDB database, and ran each one separately. We also ran a simple, 20-client workload using the TRADE application on WebSphere just to have more traffic on the system.

Triggering event

The PE Periodic Exception log indicated a violation on the sort overflow threshold. We opened the details and saw the information in Figure 6-53 on page 415.

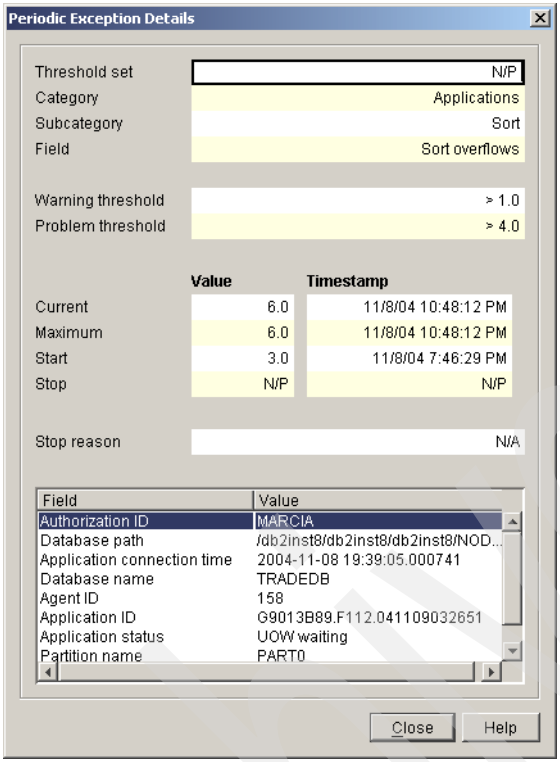


Figure 6-53 Sort overflow exception details

Investigation without recreating the problem

We know from the exception details which application had the sort overflows, and what time the sort overflow was detected. This is enough for us to look at the Application Summary history and see if we can find anything there.

The steps for arriving at the history screen are described elsewhere in this book, so we do not show them here. Figure 6-54 shows the Application Summary list snapshot taken at 7:48:36 PM. This is slightly after the violation occurred and we see that the sort overflow column shows three overflows. We also notice that the total sorts was also three, indicating a 100% sort overflow percentage.

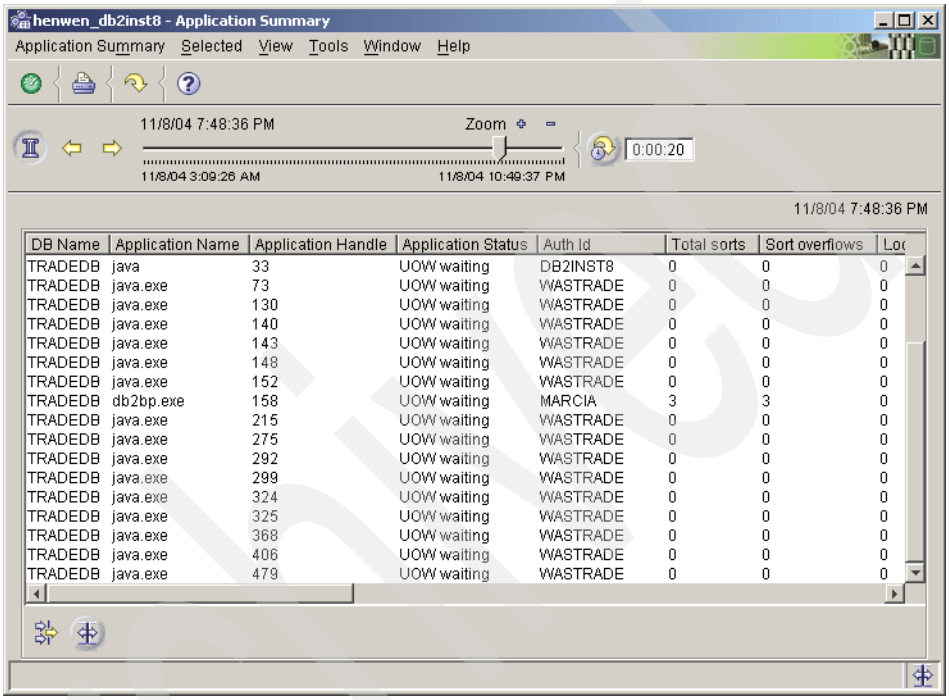


Figure 6-54 Application Summary - sort overflow

We drill down into the application, but were not able to find what statement was being executed; the snapshot did not catch it (see Figure 6-55 on page 417). So, is there still a way to possibly find what statements caused the overflow?

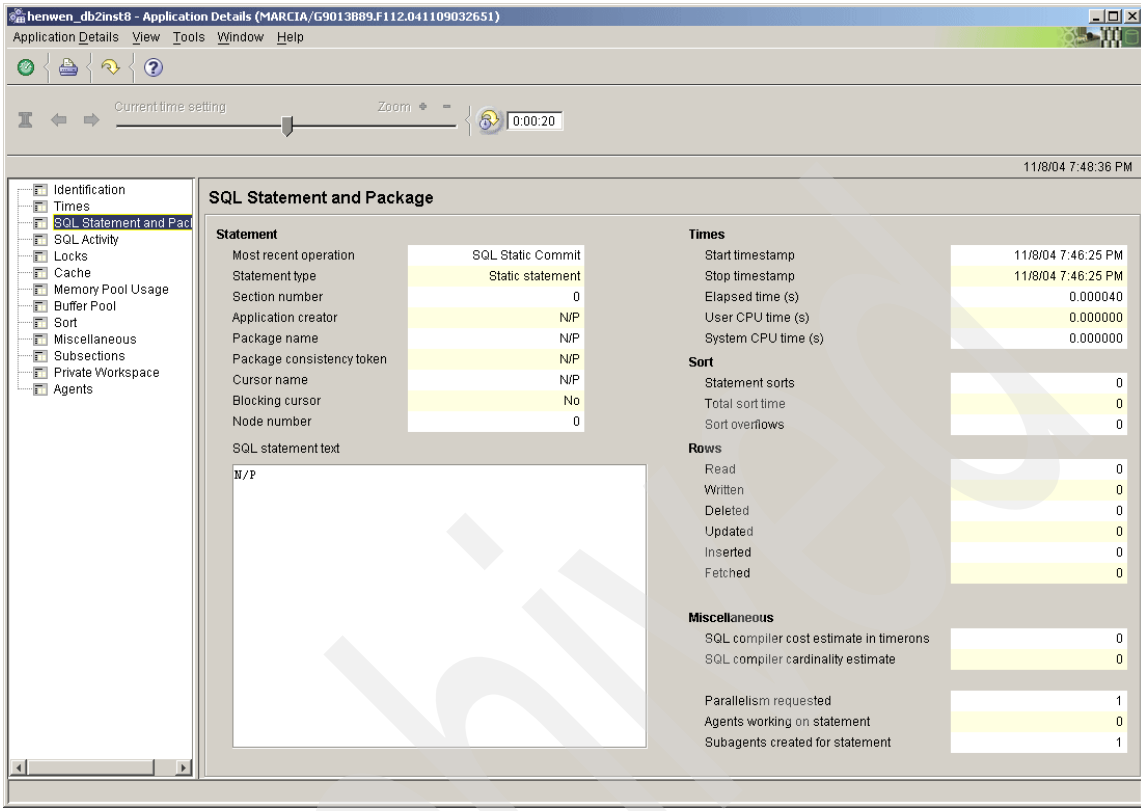


Figure 6-55 Application Details - History - no statement text available

Next, we look in the Dynamic SQL cache in the Statistics Details screen. We open the window and look at the history for a similar time period. This is shown in Figure 6-57 on page 419. We set up a filter for this screen, asking to see only statements for TRADEDB and also only those that had done a sort. The filter is shown in Figure 6-56.

Tip: The Dynamic SQL screen usually has a lot of data, so we suggest always using a filter to restrict the list. In this case, we filtered only database names, but you can also filter statement text, using a LIKE predicate. This can be very handy when you are looking for a particular statement or a statement acting on a specific table.

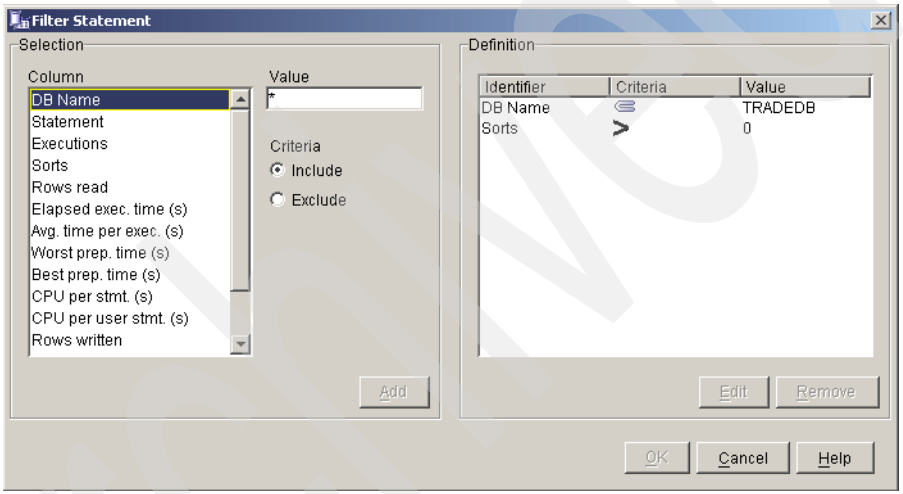


Figure 6-56 Filter for Dynamic SQL window

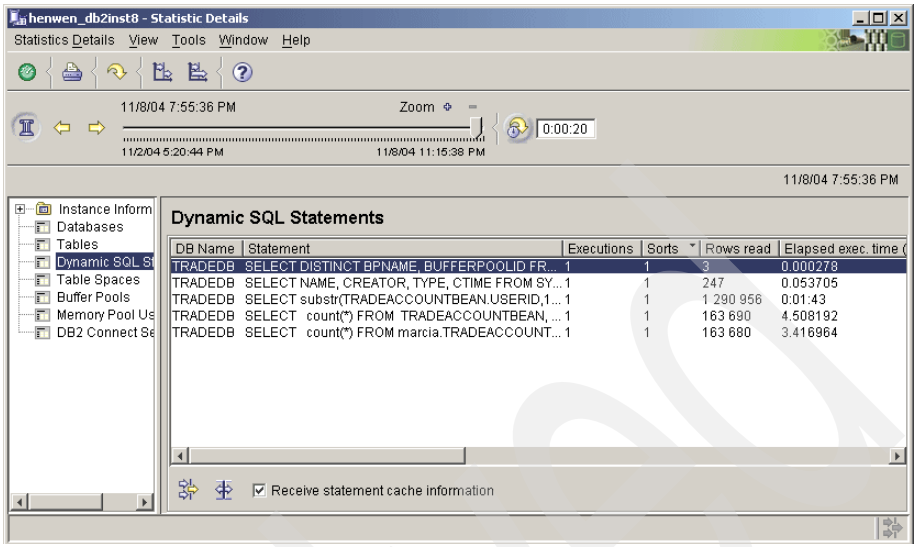


Figure 6-57 Statistics Details - Dynamic SQL cache history

Now we narrowed the view down to five statements in the cache. There is still a possibility that the statement is not there, because the statement cache is not necessarily coordinated with the application snapshot; also, our history capture interval for dynamic SQL is set to five minutes, while the application capture interval is set to one minute. The point is you cannot always guarantee you will find the exact statement, but depending on the situation, you may be able to deduce it.

So we notice the Rows Read values for the first two statements are very low and do not match the large number of rows we noticed in the Application Details history. We also see that the elapsed time for those statements is short, so they are unlikely to be the culprits. Then we notice that the last statement shows a table with schema “MARCIA”, which matches the authorization ID on the Exception Details we saw in Figure 6-53 on page 415. This is most likely the offending statement, or one of them.

We drill into the details for that statement, and see more information, as shown in Figure 6-58. There is a new ability in PE Fix Pack 2 that allows you to launch the Visual Explain directly from PE. We do this to see if we can tune the SQL statement performance by creating new index. This access plan is shown in Figure 6-59 on page 421.

The rest of this scenario is essentially the same as the scenario in the earlier redbook - the DBA determined an index was required, created the index, reran the test, and saw an improvement. We leave that as an exercise for the reader. We wanted to show here some of the newer features in PE Version 2 and how these can help you find problems more quickly.

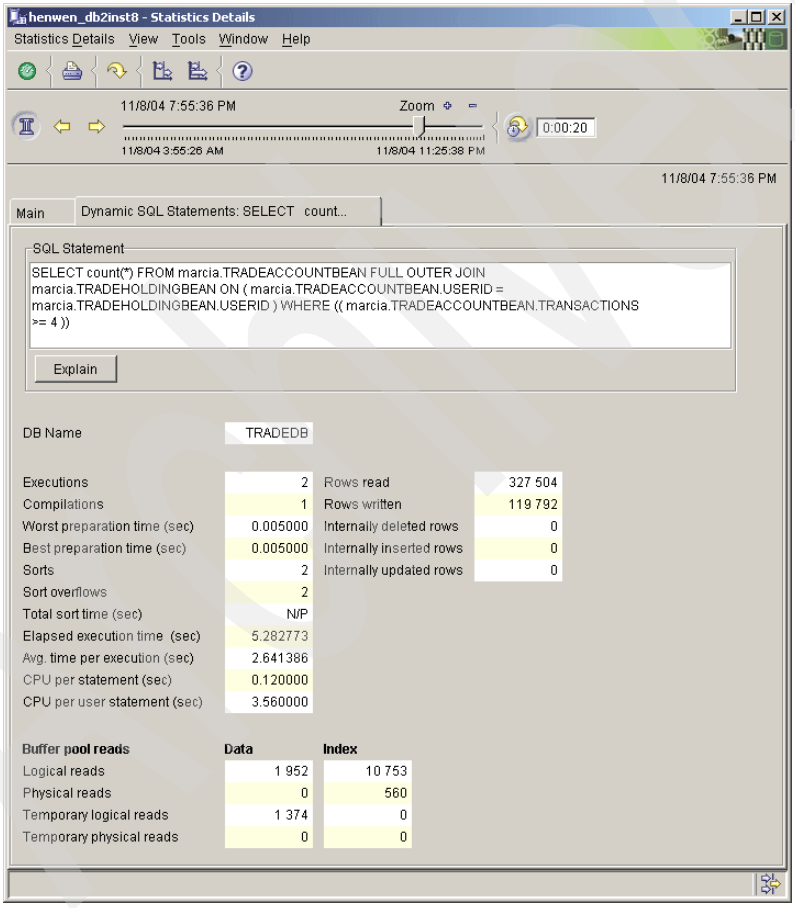


Figure 6-58 Dynamic SQL statement text

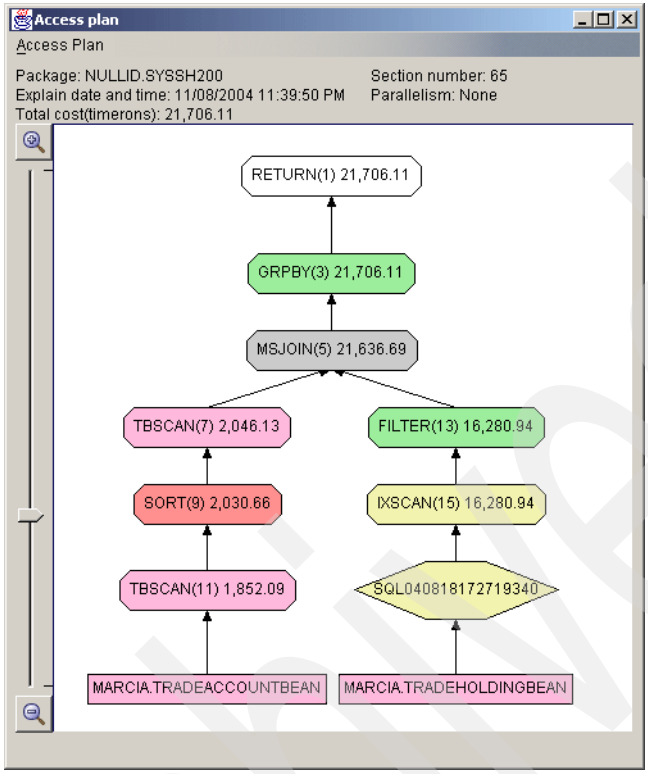


Figure 6-59 Explained query

6.2.4 Transaction log full

In this scenario, we use DB2 Performance Expert as the primary tool to investigate a problem reported by a user about the transaction log filling up. We demonstrate how to use PE to find information about the cause that is not available in the db2diag.log, and we use PE's history data to find information that would generally not be available without reproducing the error while implementing additional monitoring. We also offer insight concerning how to determine if a problem with transaction log space is caused by other performance issues. This problem can often be difficult to investigate and resolve - one database connection may receive an error, another may appear in the db2diag.log file, and yet another connection, or group of them, may be the real cause.

Environment configuration

We used the following environment for this scenario:

- Remote DB2 instance “DB2” on server SIAM (Windows 2000 Server, DB2 v8.1 FP6a)
  - PEDEMO database configuration uses mostly the DB2 V8.1 ESE Windows default values, including the following parameters relevant to this scenario:
    - Log file size (4 KB) (LOGFILSIZ) = 250.
    - Number of primary log files (LOGPRIMARY) = 3.
    - Number of secondary log files (LOGSECOND) = 2.
  - Database manager configuration:
    - Diagnostic error capture level (DIAGLEVEL) = 3.
    - All default monitor switches are enabled.
- PE Server 2.1 FP1 on DB2 instance on server WISLA (Windows 2000 Server, DB2 v8.1 FP6a)
  - PE Server configuration:
    - Recording interval: 30 seconds.
    - Time frame: 50 hours.
    - Components enabled for history collection: Application, Locking conflicts, and System.
    - Interval Multiplier for all components: 1
- PE Client v2.1 FP1 (Windows 2000 Professional workstation)

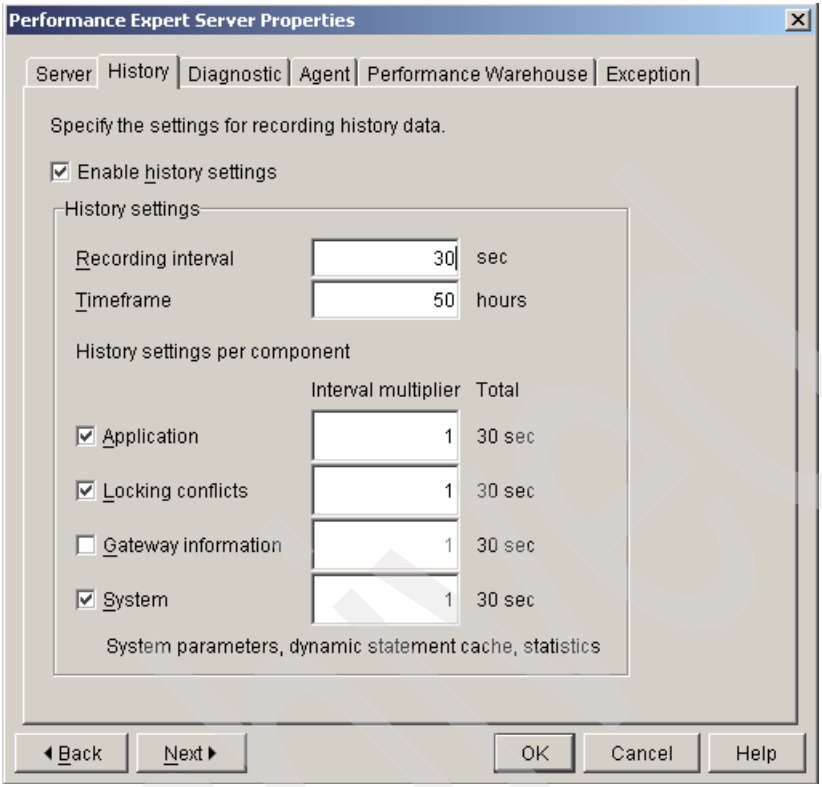


Figure 6-60 PE Server properties for transaction log full scenario

Workload used

The database activity for this scenario is generated by the following connections:

- PEDEMO Java application on server SIAM (various operations)
- loader.exe Java application on server SIAM (large insert operations)
- DB2 command window session on a Windows 2000 workstation with DB2 Administration Client V8.1 FP6a (update statement)

These connections use the PEDEMO database with some additional tables added to it.

Triggering event

A DB2 command window user reported executing the following statement against the PEDEMO database and receiving the message that follows:

```
db2 update db2admin.location_emea set number=number+1
SQL0964C  The transaction log for the database is full.  SQLSTATE=57011
```

The table on which the user executed the statement only had 5130 relatively small rows at the time, so the user did not understand why their statement would be filling the transaction log. They contacted us to investigate further and told us that they executed the statement shortly after 8:30 PM.

Investigation without recreating the problem

Example 6-19 shows the db2diag.log messages that correspond to the message received by the user. These messages identify useful information, such as the application handle of the connection that encountered the error and the exact time at which it occurred, but they do not reveal the root cause of the problem is or how to correct it.

Example 6-19 db2diag.log messages for transaction log full

```
2004-11-11-20.34.56.203000   Instance:DB2   Node:000
PID:3732(db2syscs.exe)   TID:4168   Appid:G901266B.IE0A.009EC2042927
data pRoTection  sqlpgResSpace Probe:2860   Database:PEDEMO

ADM1823E  The active log is full and is held by application handle "54".
Terminate this application by COMMIT, ROLLBACK or FORCE APPLICATION.

2004-11-11-20.34.56.203001   Instance:DB2   Node:000
PID:3732(db2syscs.exe)   TID:4168   Appid:G901266B.IE0A.009EC2042927
data pRoTection  sqlpWriteLR Probe:980   Database:PEDEMO

DIA8309C Log file was full.
ZRC=0x85100009ÿ
```

We have circled some relevant information from the db2diag.log messages. We notice in the db2diag.log file that after a flurry of entries (not shown here), the transaction log full condition seems to have ended. Besides the entries shown for TID 4168, similar entries also appear for TIDs 4256, 4292, and 2680, which also experienced the SQL0946 error, and all entries reference the log being held by application handle 54. We will reference these numbers during our investigation, as well as the other relevant information we have circled in the db2diag.log entries.

To further investigate this problem, we first estimate a reasonable amount of log space we might expect this operation to consume and compare that with the total log space available for the database. We cannot predict the exact amount of log

space that will be used by a transaction. However, the query in this example affects each row in the table, so we estimate that the log space used should not be much larger than the total table size. We check (the Control Center or the **`describe table`** command could be used for this) and see that the table only has 5130 rows and three columns totaling 184 bytes per row. The columns of type varchar only have average lengths of 10 and 22, but even if each row were using the maximum space possible, we would not expect the table size to exceed 1 MB.

Table 6-1 Table structure for db2admin.product1

Column name	Type	Length
Name	Varchar	30
Number	Integer	4
Description	Varchar	150

Next, we check the total log space available. In Performance Expert, we log on to the monitored instance for this database and open the System Parameters - Databases window. We double-click the PEDEMO database and then see all of the log-related parameters by selecting Logging and Recovery, as shown in Figure 6-61 on page 426. We see the Windows default database configuration settings of three primary logs, two secondary logs, and a log size of 250 pages of 4 KB each, corresponding to a total available log space of about 5,100,000 bytes. We also step through the same data in history mode and determine that the values of the parameters have been the same throughout the period in question. We estimate that if this query were the only transaction running against the database, it would not use more than 20% of the log space by itself, and certainly should not be using all the available space.

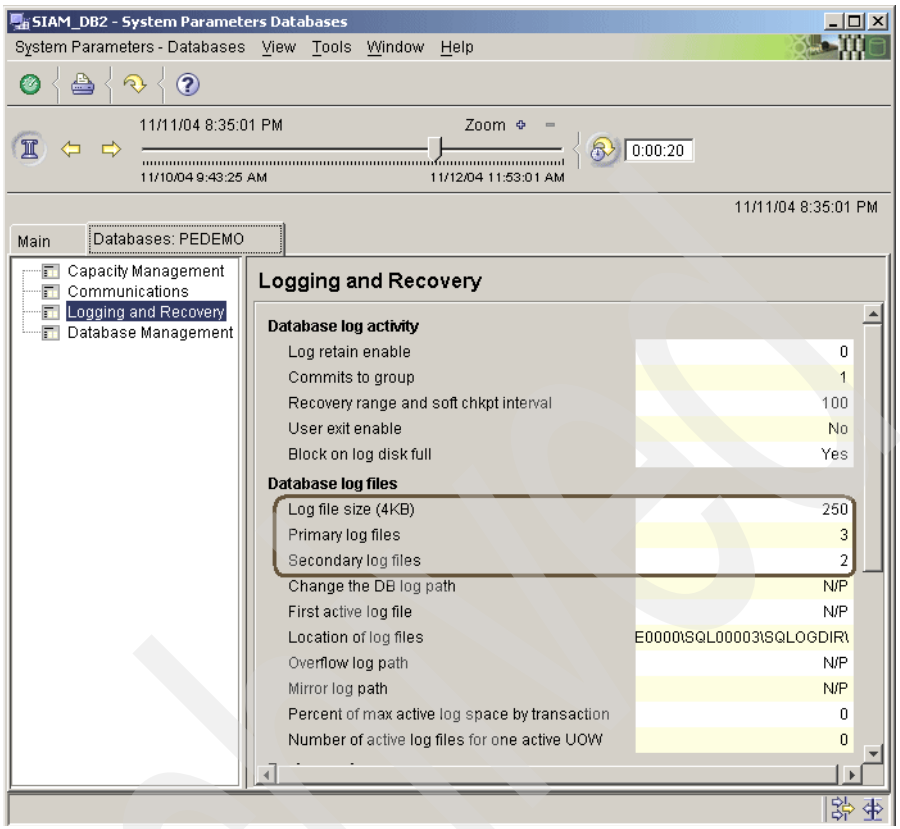


Figure 6-61 Log parameters in the System Parameters Databases window

Next, we open the Statistics Details window to look at the historical log usage information. We highlight **Databases**, add a filter to only include the PEDEMO database, and customize the columns displayed on the right to display the available log-related fields. Figure 6-62 on page 427 shows the first data gathered after the time stamp shown in the db2diag.log file when the log filled up.

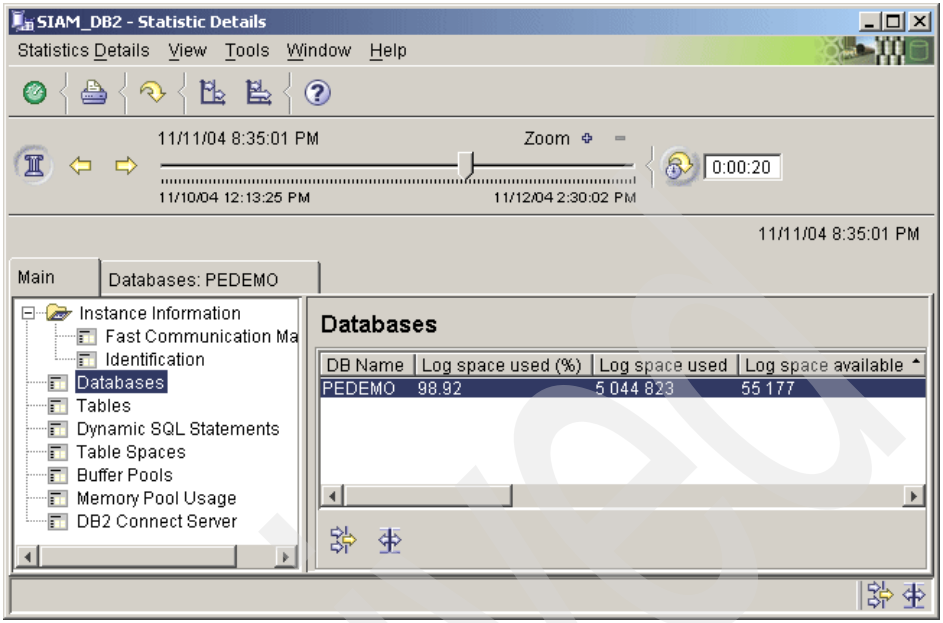


Figure 6-62 Statistics Details window in history mode when log filled up

We double-click the database name and then select **Logs** to see more information. Figure 6-63 on page 428 shows a point in the history data just before the first secondary log was allocated. Note that “Log space used (%)” on the “Main” tab is calculated as the log space used out of the total log space available, including both primary and secondary logs, but “Used space” in the “Distribution of space” graph on the “Databases: PEDEMO” tab is calculated as the log space used divided by just the primary log space.

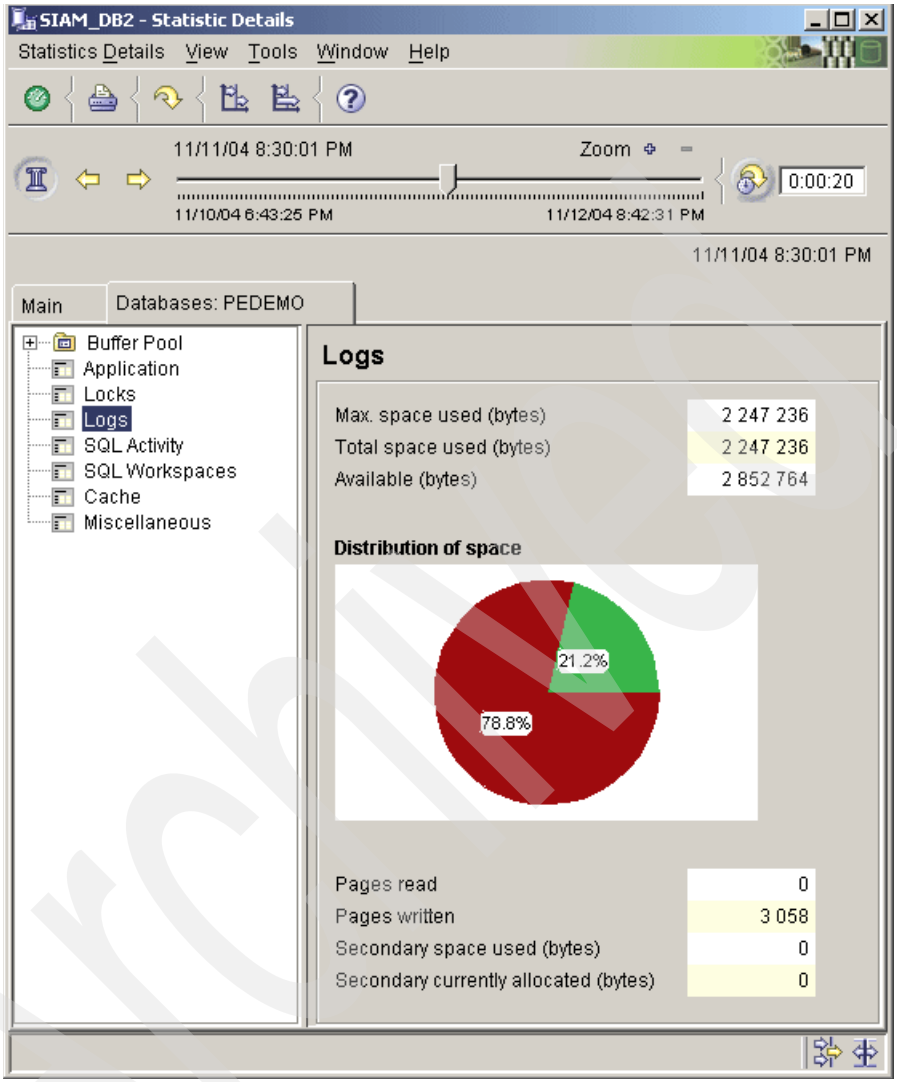


Figure 6-63 Log information in Statistic Details

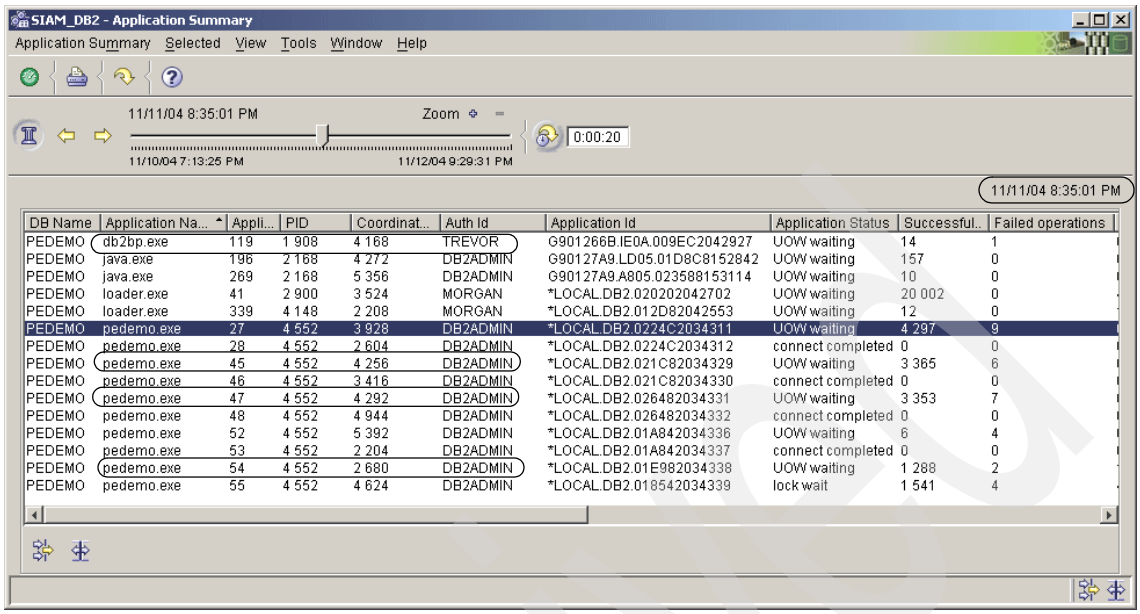
We scroll through the historical data to get a rough idea of the log usage during periods before and after the log filled up. Table 6-2 on page 429 summarizes significant changes in log usage seen in the history data through this period. We also see that the two secondary logs were allocated at 8:31 PM and 8:34 PM.

Table 6-2 Timeline of log space usage

Period	Time	Log space used
1	7:43 PM	0%
2	7:44 PM - 8:27 PM	Gradually increases to 13%
3	8:28 PM - 8:34 PM	Increases more rapidly by about 10% per minute to 78%
4	8:35 PM - 8:36 PM	Suddenly jumps to 99%, where it stays for a couple minutes
5	8:37 PM - 8:43 PM	Drops to 82% and remains fairly constant
6	8:44 PM - 10:14 PM	Drops suddenly to 4% and increases gradually to 20%

Next, we look at the history data in the Application Summary, using the available data nearest to the time stamp of the transaction full error message in the db2diag.log to see if information was captured concerning the user's connection at the time when it received the error. Figure 6-64 on page 430 shows all the applications that were connected to the PEDEMO database at that time. We have circled the connection for which the user reported the error, the connection for application 54 that the db2diag.log message refers to, and the other connections identified in the db2diag.log messages as having received a log full error.

Investigating a problem in the past is possible when the PE Server was configured with a history time frame greater than the amount of time that has elapsed between the problem occurrence and the investigation. Depending on the speed with which entire transactions occur on the monitored database and on the other configured history settings for the PE Server (recording interval, components for which gathering history data is enabled, and the interval multiplier for those components), the history data may or may not contain useful information.



DB Name	Application Na...	Appli...	PID	Coordinat...	Auth Id	Application Id	Application Status	Successful..	Failed operations
PEDEMO	db2bp.exe	119	1 908	4 168	TREVOR	G901266B.IE0A.009EC2042927	UOW waiting	14	1
PEDEMO	java.exe	196	2 168	4 272	DB2ADMIN	G90127A9.LD05.01D8C8152842	UOW waiting	157	0
PEDEMO	java.exe	269	2 168	5 356	DB2ADMIN	G90127A9.A805.023588153114	UOW waiting	10	0
PEDEMO	loader.exe	41	2 900	3 524	MORGAN	*LOCAL.DB2.020202042702	UOW waiting	20 002	0
PEDEMO	loader.exe	339	4 148	2 208	MORGAN	*LOCAL.DB2.012D82042553	UOW waiting	12	0
PEDEMO	pedemo.exe	27	4 552	3 928	DB2ADMIN	*LOCAL.DB2.0224C2034311	UOW waiting	4 297	9
PEDEMO	pedemo.exe	28	4 552	2 604	DB2ADMIN	*LOCAL.DB2.0224C2034312	connect completed	0	0
PEDEMO	pedemo.exe	45	4 552	4 256	DB2ADMIN	*LOCAL.DB2.021C82034329	UOW waiting	3 365	6
PEDEMO	pedemo.exe	46	4 552	3 416	DB2ADMIN	*LOCAL.DB2.021C82034330	connect completed	0	0
PEDEMO	pedemo.exe	47	4 552	4 292	DB2ADMIN	*LOCAL.DB2.026482034331	UOW waiting	3 353	7
PEDEMO	pedemo.exe	48	4 552	4 944	DB2ADMIN	*LOCAL.DB2.026482034332	connect completed	0	0
PEDEMO	pedemo.exe	52	4 552	5 392	DB2ADMIN	*LOCAL.DB2.01A842034336	UOW waiting	6	4
PEDEMO	pedemo.exe	53	4 552	2 204	DB2ADMIN	*LOCAL.DB2.01A842034337	connect completed	0	0
PEDEMO	pedemo.exe	54	4 552	2 680	DB2ADMIN	*LOCAL.DB2.01E982034338	UOW waiting	1 288	2
PEDEMO	pedemo.exe	55	4 552	4 624	DB2ADMIN	*LOCAL.DB2.018542034339	lock wait	1 541	4

Figure 6-64 Application Summary history data from time of error

By drilling down to the application details for each application, we can see more information about what the application was executing and what resources it was consuming. We do this first for the applications we have circled, and then for the other applications that were running. We start at the point in time when the error occurred, but we also look further back in the history data to see if the connections also ran other statements. We also look for what changed at the times when our log space analysis at the database level showed changes (see Table 6-2 on page 429). In the “SQL Statement and Package” window, we can see the SQL statement the application was running when the snapshot was taken. In the “SQL Activity” window, we can see the UOW log space used in the UOW section. This process is quicker if periodic exceptions have already been configured for UOW log space usage.

We show our findings in Table 6-3 on page 431, in which the periods correlate to those in Table 6-2 on page 429.

Table 6-3 Timeline of database activity

Period	Application	Log space (bytes)	Rows inserted/updated	UOW
1	None	N/A	N/A	N/A
2	Various PEDEMO connections	Various small amounts	Varied	Varied
2	339/loader.exe	1,376	10	1
3	41/loader.exe	2,807,702	20,000	1
4	119/db2bp.exe	421,410	30,438	8

The SQL statement reported for the connection of the user who received the error (application 119) matches the statement reported by the user, and the log space used is not significant, as shown in Figure 6-65, which indicates that perhaps another application was the root cause of the problem. After looking at a few other applications, we see that the other applications referred to in the db2diag.log entries were not using significant log space either.

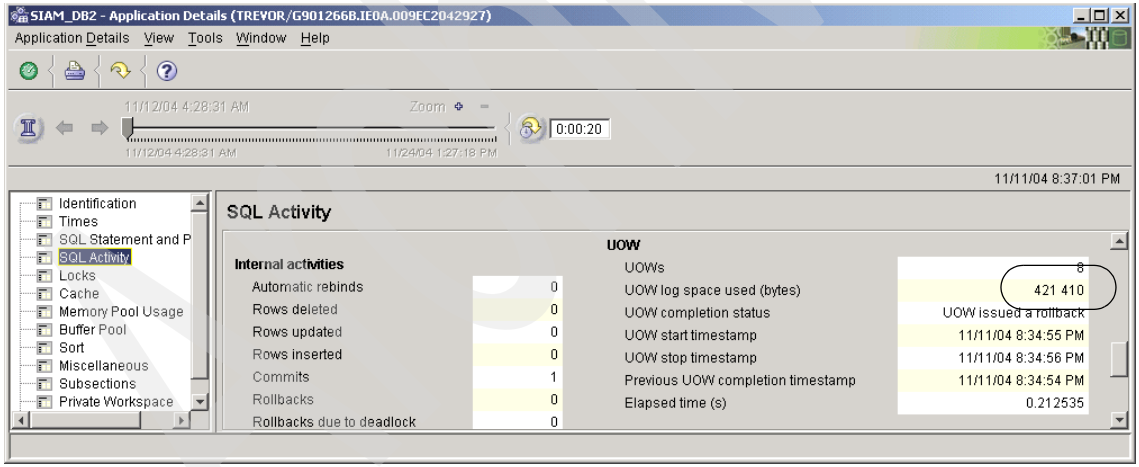


Figure 6-65 Application Details show relatively low transaction log space usage (App 119)

However, one of the two loader.exe applications has a single unit of work that is using a relatively large amount of transaction log space, which is shown in Figure 6-66. The transaction attempted about 20,000 inserts in one unit of work, with no intermediate commits. We also see the SQL statement issued by each one by selecting “SQL Statements and Packages” on the left.

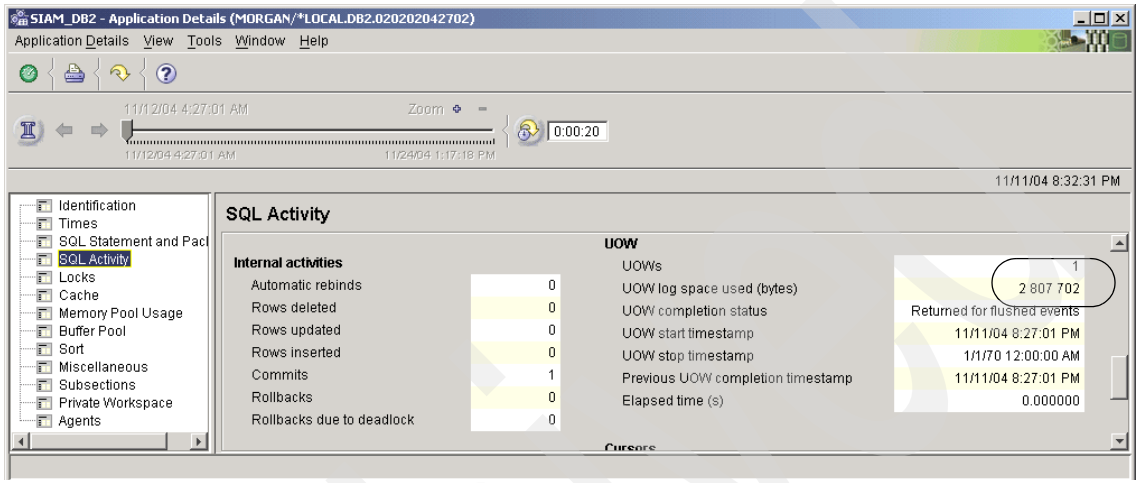


Figure 6-66 Application Details showing relatively high transaction log space usage (App 41)

We also check the listing of all dynamic SQL that has been executed against the database to see if we can identify any statements that may be related to our problem. To do this, we use PE’s Statistics Details window, select **Dynamic SQL Statements**, and check the **Receive statement cache information** box. However, this method does not show what was run during a certain time period, and it will not show you which application executed each statement. See Figure 6-67 on page 433 for a partial listing of results, with the inserts from application 41 circled. Because all of the statements run in this scenario showed up in the history data, this method did not reveal any relevant new information.

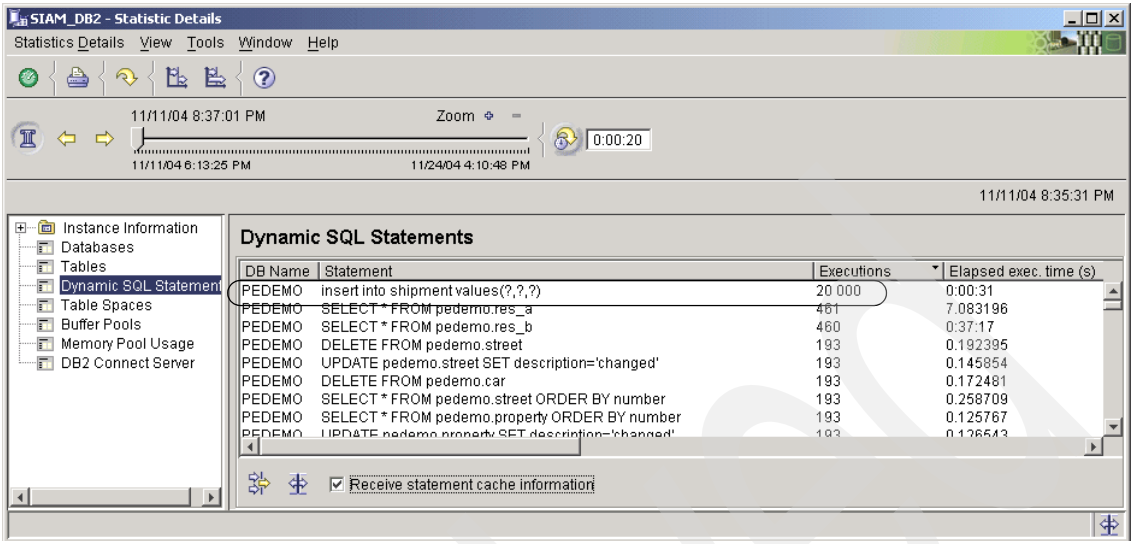


Figure 6-67 Statement cache information from time of error

In the Application Summary window, we also see some deadlocks for the PEDEMO application during this time period, so we must also consider the possibility that because of locking issues, some transactions are holding locks for longer than would be necessary if the locking issues were resolved.

Through this analysis, we make the following observations:

- The update statement run by the user who received the transaction log full error was only a small part of the problem.
- The statement run by application 54, referenced by the db2diag.log entries as holding the active log, was actually using very little log space.
- The insert run by application 41 (loader.exe) attempted to insert too many rows in a single transaction, and was not committed until several minutes after the insert completed. Application 339 also did not commit its work promptly - the loader.exe application appears to execute multiple inserts before issuing a commit.
- The database had relatively low log space usage during other periods of the day, but the log filled up in part because unrelated connections with high log space usage happened to be scheduled during the same period.

Additional investigation - recreating the problem

If insufficient information had been gathered from the available historical snapshot data, we could also have recreated the problem while gathering additional information. Even though snapshot data does contain some counter and high watermark data that represents all activity occurring between the time the monitor switches were enabled or reset (usually when the DB2 instance is started) and when the snapshot is taken, application snapshots only contain application data for applications currently connected when the snapshot is taken. Accordingly, the data viewable through PE's Application Summary and Statistics Details windows only reflects the snapshot data that is available.

Another option is to use a SQL activity trace in PE, which creates a statement event monitor in the monitored database. The resulting SQL Activity Trace Report shows the SQL statements that were run against the database during the time period in which the SQL activity trace was enabled, including application information for each connection that was made. It also shows detailed information about each statement operation of each statement, including the start time and stop time. You can then compare the information returned by the SQL Activity Trace Report with the historical snapshot data in PE's Application Summary and Statistics Details window to gain a more complete picture of the environment.

You can run a SQL activity trace for one application from the Application Summary window for up to 12 hours at a time. This feature is useful when you know which database connection you want to trace and want to minimize monitoring overhead. You can also run a SQL activity trace for all applications, or with a filter, from the Performance Warehouse window for up to 24 hours per execution. This feature is useful when you do not know which connections you want to trace or when the applications you want to trace create other database connections within them.

The data received from SQL activity traces can be very useful, but you should only run them when you have a specific need in order to avoid incurring high monitoring overhead without a need. Accordingly, you will normally not have this data available for problem determination without recreating the problem while running a trace.

We ran the predefined SQL Activity Trace Report process from the Performance Warehouse window for just a few minutes while recreating this problem, and it produced a 25 MB HTML report listing information about the 10 application connections made during the time period, the hundreds of SQL statements executed during this period, and detailed information about the thousands of statement operations performed.This report verified that the key applications we were investigating did not execute any additional statements in between snapshots that were recorded in history mode, and that no other applications

connected that were not present in the Application Summary history data. For example, Figure 6-68 shows a portion of the report listing all applications that were connected to the database while the trace was running, similar to the Application Summary history data shown earlier.

Application Summary

Application ID	Application Handle (agent ID)	Application Name	Database Alias Used by Application	Node Number	Time of First Database Connection	The operating system userid	The authorization ID of the user who invoked the application
appl_id	agent_id	appl_name	client_db_alias	node_number	conn_time	execution_id	auth_id
*LOCAL.DB2.0224C2034311	27	pedemo.exe	PEDEMO	0	19:42:59.102572	DB2ADMIN	DB2ADMIN
*LOCAL.DB2.020202042702	41	loader.exe	PEDEMO	0	20:27:01.629979	DB2ADMIN	MORGAN
*LOCAL.DB2.021C82034329	45	pedemo.exe	PEDEMO	0	19:43:07.698874	DB2ADMIN	DB2ADMIN
*LOCAL.DB2.026482034331	47	pedemo.exe	PEDEMO	0	19:43:17.79889	DB2ADMIN	DB2ADMIN
*LOCAL.DB2.01E982034338	54	pedemo.exe	PEDEMO	0	19:43:29.548321	DB2ADMIN	DB2ADMIN
*LOCAL.DB2.018542034339	55	pedemo.exe	PEDEMO	0	19:43:29.678455	DB2ADMIN	DB2ADMIN
G901266B.IE0A.009EC2042927	119	db2bp.exe	SMPEDEMO	0	20:28:29.846471	UDBRS05	TREVOR
G90127A9.LD05.01D8C8152842	196	java.exe	SMPEDEMO	0	20:22:32.13855	DB2ADMIN	DB2ADMIN
G90127A9.A805.023588153114	269	java.exe	SMPEDEMO	0	20:24:04.348756	DB2ADMIN	DB2ADMIN
*LOCAL.DB2.012D82042553	339	loader.exe	PEDEMO	0	20:25:51.943713	DB2ADMIN	MORGAN

Figure 6-68 Application Summary from SQL activity trace report

Root cause of problem

The DB2 default values for the number and size of primary logs are not adequate for this environment as it is currently configured. This problem is compounded by the poorly written applications that did not commit their work often enough, and which both occurred during the same time period, leaving less transaction log space to be used by other applications.

Application of best practices

Tuning applications and changing log configuration parameters requires application development cooperation and extended monitoring. We should take steps to ensure that enough transaction log space is available for all database transactions to complete successfully, considering the following factors:

- Transactions should commit often enough to avoid using large amounts of transaction log space.

- While still meeting business needs, large jobs should be scheduled in a way that minimizes conflict both between them and with other database connections.
- Improving concurrency may reduce concurrent log space usage in some environments. Consider using type II indexes for improved concurrency. Sometimes just increasing available log space will only mask a concurrency problem.
- The primary log space allocated should be sufficient for most database activity, with secondary log space being used as a safety measure or for infrequent peaks in database activity.
- After all jobs and applications have been tuned and scheduled according to business needs, then we can consider increasing either the number of primary logs or the log size.

Other resources are available that discuss making changes such as these in great detail.

In this scenario, the transaction log filling up can be prevented by any of several possible solutions (more frequent commits during large transactions, staggering the scheduling of large transactions, increasing the log size or number of logs, and so on). Implementing more than one of these solutions will decrease the probability of the transaction log filling up in the future, and continued monitoring will provide input for future changes that may be necessary due to workload changes.

Proactive monitoring

Rather than waiting for a user to notify us of the transaction log filling up in the future, we could have configured periodic exceptions in PE to alert us about different events concerning the transaction logs. For example, we show in Figure 6-69 on page 437 some alerts set up on the database level for:

- The log space currently used (by total, with warning and problem thresholds set at about 70% and 80%)
- The maximum log space used (by total, with the same thresholds)

and also at the application level for:

- UOW log space used (both by total and per commit, with thresholds set at about 20% and 30%)

We have also defined another similar exception for UOW log space used with a qualifier that filters the application name “loader.exe”. We did this to watch for future occasions when this problem application uses more than about 5% of the available log space per commit.

DB2 does not maintain a counter for the number of times the transaction log has filled up, but the counters it does maintain are useful in better understanding your logging needs. We leave the default retrieval interval for periodic exception processing at 60 seconds in the PE Server properties window, and we use a short interval multiplier of 1 in the exception processing activation window when we activate the threshold set.

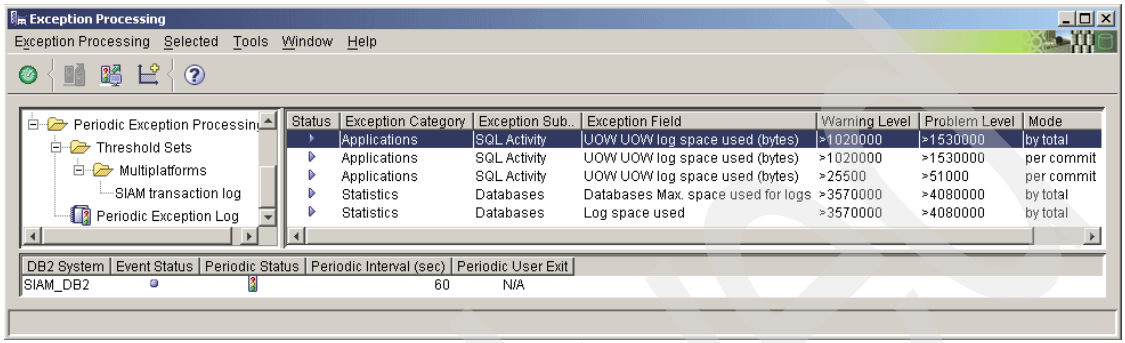


Figure 6-69 A sample periodic exception alert configuration for transaction log conditions

The maximum log space used is useful to track because log space used often can spike in between exception intervals and not trigger an exception. However, once the warning threshold has been triggered for the maximum log space used, or for any other exception monitoring a high water mark, the periodic exception will be triggered once per interval. This will result in the potentially annoying, application modal alert if the “show message” box is checked for exception refresh notification in the Exception settings window, until the next time the monitor switches are reset.

With the monitoring described above in place, the alert in Figure 6-70 for the UOW log space used by the loader.exe application was received several minutes before the log full condition.

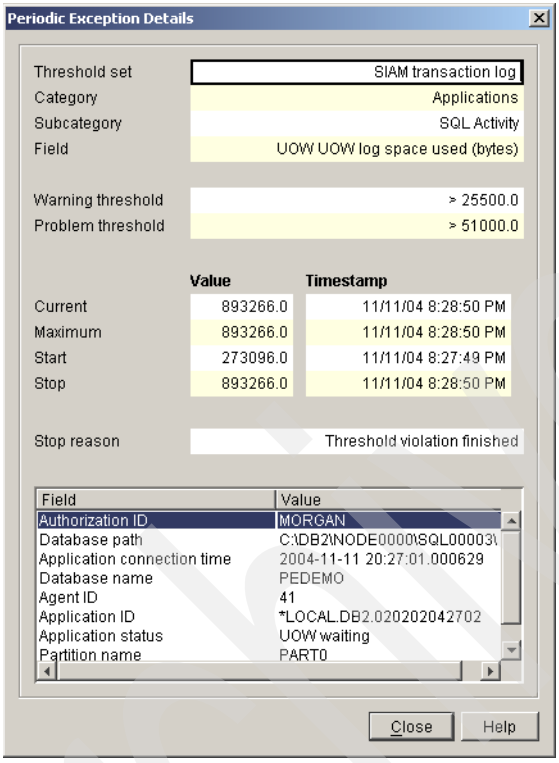


Figure 6-70 A periodic exception received for application before log full condition has occurred

Likewise, the alert in Figure 6-71 on page 439 shows that the total log space used in the database had exceeded the 70% warning threshold several minutes before the log full condition. We also could have configured these alerts to be sent as e-mail notifications.

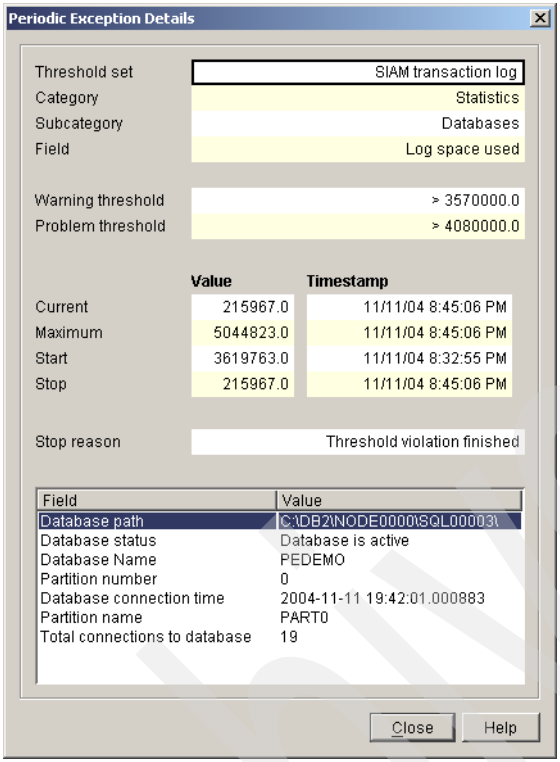


Figure 6-71 A periodic exception received for a database before log full condition has occurred

Trending analysis

Investigating the specific events that occurred during the period in which the transaction log filled up is useful in understanding why the problem occurred under that set of circumstances. Performing a long-term analysis of various performance indicators related to logs, and also other factors that might have an effect on logging space used (commit frequency, concurrency, and so on), provides further insight to understand our logging needs. We also used the Performance Warehouse window to run queries against the warehouse data that has been aggregated over a longer period of time (in the PWH schema) to determine how often the log fills up.

We also used Performance Warehouse to execute custom queries against the actual historical snapshot data (in the DB2PM schema) that we analyzed earlier in this scenario when we looked at various points in history in the Application Summary and Statistics Details windows. Details about using Performance Warehouse to execute queries, and interpreting the data these queries return, are found elsewhere in this redbook.

This trending analysis shows that log space has been a long-term concern in this environment and requires further study. After addressing the specific issues that were identified as contributing causes in this particular incident of the log filling up, we should continue to monitor our logging needs to determine if other causes exist also.

6.3 Tuning DB2 for OLTP application with PE

DB2 PE has rich functions and features for monitoring and analyzing the performance of DB2 database systems and applications. This performance monitoring and analyzing capability not only can be used in production and test environments, but also can be used to verify if the new developed application has met the performance requirements before the application is released. For this scenario, we demonstrate PE usage in performance requirement verification for a newly developed application using an online transaction processing (OLTP) DB2 application. We focus on tuning the DB2 configuration for the application.

OLTP applications support day to day mission critical business activities. OLTP applications are usually used for order entry, banking, stock trading, airline industry, inventory management, and so on. These application support hundreds of thousands of users issuing millions of transactions per day. In this environment, transactions need to be processed in real time and concurrently so the response time needs to usually be in sub seconds and stringent. Tuning your application and the database that your OLTP application will be running against is very critical to maintaining a quick response time to your users.

Database performance can be impacted greatly by how the objects get created and how well those objects interact together. This is why buffer pools, tablespaces, indexes, instances, and database configuration should be taken into consideration when design/architecting and implementing a database application. Below are some common tuning techniques:

- Check the system environment configuration for hardware and software problems
- Change *one* parameter at a time, then re-run the test.
- Use DB2 Configuration Advisor (AUTOCONFIGURE) to get the initial configuration parameters.
- Run RUNSTATS and keep the statistics current.
- SQL
  - Use stored procedures to reduce the data transmission between client and server.
  - Use Visual Explain or db2exfmt to analyze each SQL statement.

- Create and use indexes efficiently.

6.3.1 Scenario description

In this scenario, we used an OLTP application, called HOPPY, to demonstrate the steps of using PE to verify if the database configuration parameters have been properly tuned for the application to achieve the optimal performance. We assume the HOPPY application was designed correctly and our system does not have any serious resource limitation, such as CPU, I/O, DISK, Memory, and so on.

The laboratory environment used for this scenario is:

- An IBM @server® pSeries running AIX 5L V5.3 named TWEEDLEDUM.

  TWEEDLEDUM includes:

  - DB2 UDB V8.2
  - PE Server V2 Fix Pack 2
- The PE Server was installed on its own DB2 instance (db2in8pe).
- The monitored instance was installed on its own DB2 instance (db2inst8).
- This application used a database called RABBIT.

6.3.2 DB2 UDB configuration tuning

Every new application has the performance requirements set. The database instance, database, and application all have to be tuned to meet the performance requirements before production deployment. Here we show how to use PE to verify the database configuration. We demonstrate this PE usage with the following DB2 parameters:

- Buffer pool
- Log buffer
  - Database configuration
  - MINCOMMIT
- Sort heap
- Locks

Buffer pool

As stated in *IBM DB2 UDB Administration Guide: Planning V8.2*, SC09-4822:

"A buffer pool is the amount of main memory allocated to cache table and index data pages as they are being read from disk, or being modified. The purpose of the buffer pool is to improve system performance. Data can be accessed much faster from memory than from disk; therefore, the fewer times the database manager needs to read from or write to a disk (I/O), the better the performance. (You can create more than one buffer pool, although for most situations only one is required.) The configuration of the buffer pool is the single most important tuning area, because you can reduce the delay caused by slow I/O."

Tuning a buffer pool can involve some trade-offs:

- If the buffer pool is too small, then the DB2 SQL request will have to wait until the DB2 disk activity is done.
- If the buffer pool is too large, then memory is wasted, which can cause other problems. For example: if the buffer pool is too large in comparison to physical memory, then operating system paging will occur, which will cause disk activity.

For more information, please refer to the *IBM DB2 UDB Administration Guide: Planning V8.2*, SC09-4822.

Buffer pool tuning

To verify our buffer pools, we use PE Statistic Details and Rules of Thumb (RoT) to analyze our application. Using the PE short-term history mode in Statistic Details for buffer pools, we were able to go back to a particular point in time to gather information about buffer pools. RoT provide recommendations of how to improve the buffer pool usage if required.

The *buffer pool hit ratio* indicates the percentage of the time that the database manager did not need to load a page from disk because it was found in the buffer pool. The greater the buffer pool hit ratio, the more effective the caching in the buffer pool is, and the lower the frequency of disk activity. Ideally, you want to have your buffer pool hit ratio and buffer pool index hit ratio as close as possible to 100%. However, in a real-time live production environment, this may or may not be a realistic goal for your system based on your systems resources.

Our application uses the IBMDEFAULTBP buffer pool. In Figure 6-72 on page 443, you can see our buffer pool hit ratio is at 55% and the buffer pool index hit ratio is at 53% for the IBMDEFAULTBP buffer pool.

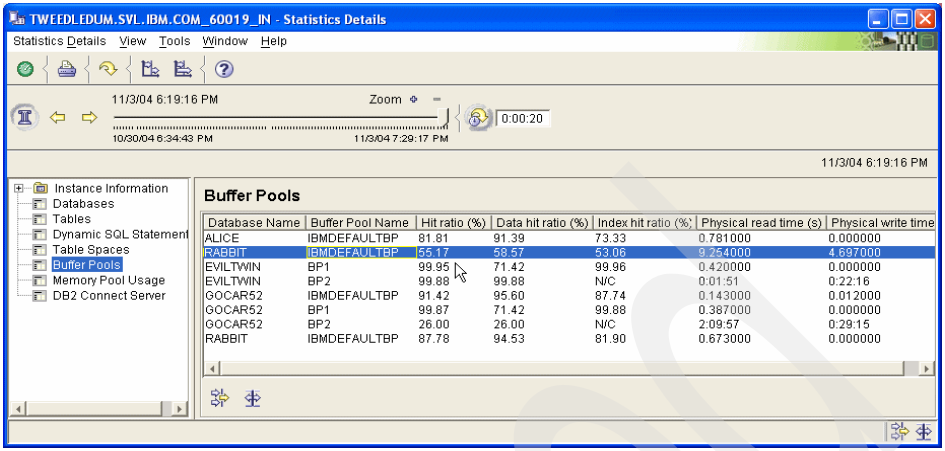


Figure 6-72 Buffer pool hit ratio

Before we make a decision on how we should fix this problem, we gather more information to examine all the facts regarding our buffer pools.

In PE, we can double-click a particular buffer pool to see more detailed information. In Figure 6-73, you can now see the details regarding the IBMDEFAULTBP buffer pool:

- Logical reads for data = 676
- Logical reads for index = 1093
- Physical reads for data = 280
- Physical reads for index = 513
- Data writes = 93
- Index writes = 137

Without PE, you will need these elements to calculate the buffer pool hit ratio and buffer pool index hit ratio manually. You also can use these numbers to do trend analysis.

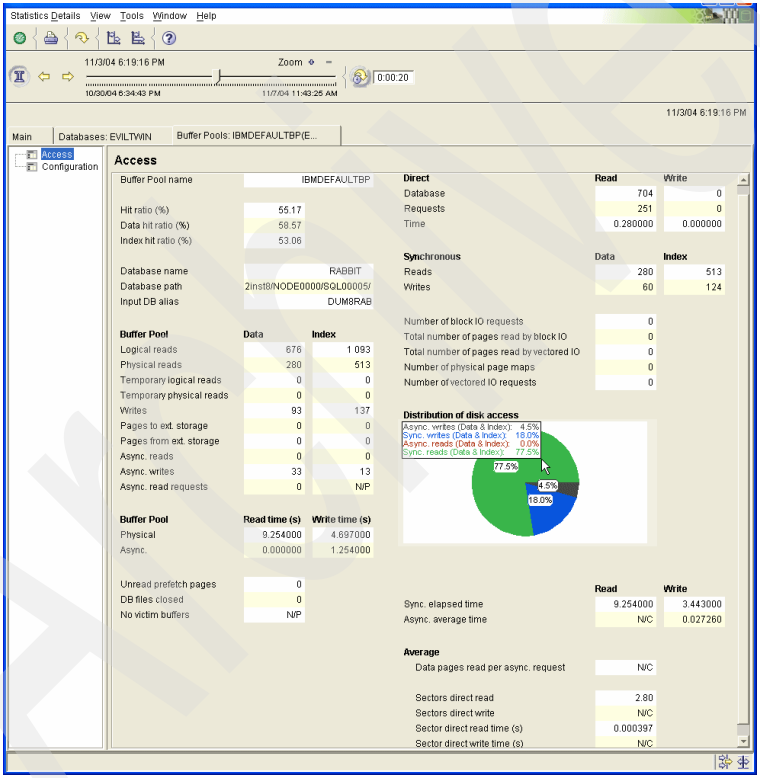


Figure 6-73 Buffer pool details

In our lab environment, we only collected a few hours of application data. We use Statistic Details for online and short-term history monitoring to analyze buffer pool usage.

Once the application is in production or if time permits, you can collect data for an extensive period of time, such as days, weeks, months, or years. You then can use PE Performance Warehouse's buffer pool reports or buffer pool analysis to do trend analysis.

We use a PE Rules of Thumb (RoT) buffer pool cluster to analyze our buffer pools. The RoT detected a problem with our application's buffer pool hit ratio, buffer pool hit ratio data, and buffer pool hit ratio index. AsyncVsSynch write also is a problem. Please see Figure 6-74 for more details.

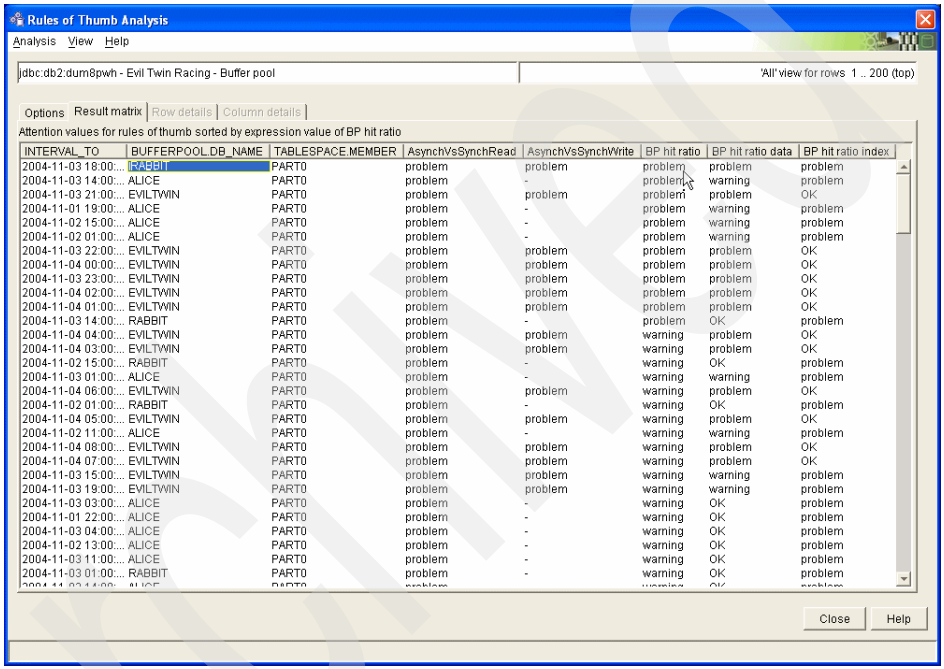


Figure 6-74 RoT buffer pool problems

As you can see in Figure 6-75, PE RoT recommendations is “For an OLTP application to increase the number of buffer pool pages”. In the database server, the number of buffer pool pages is currently 1000. After this is implemented, the PE RoT recommendation is to increase the number of buffer pool pages. Our application has both the buffer pools hit ratio and buffer pool index hit ratio close to 100%.

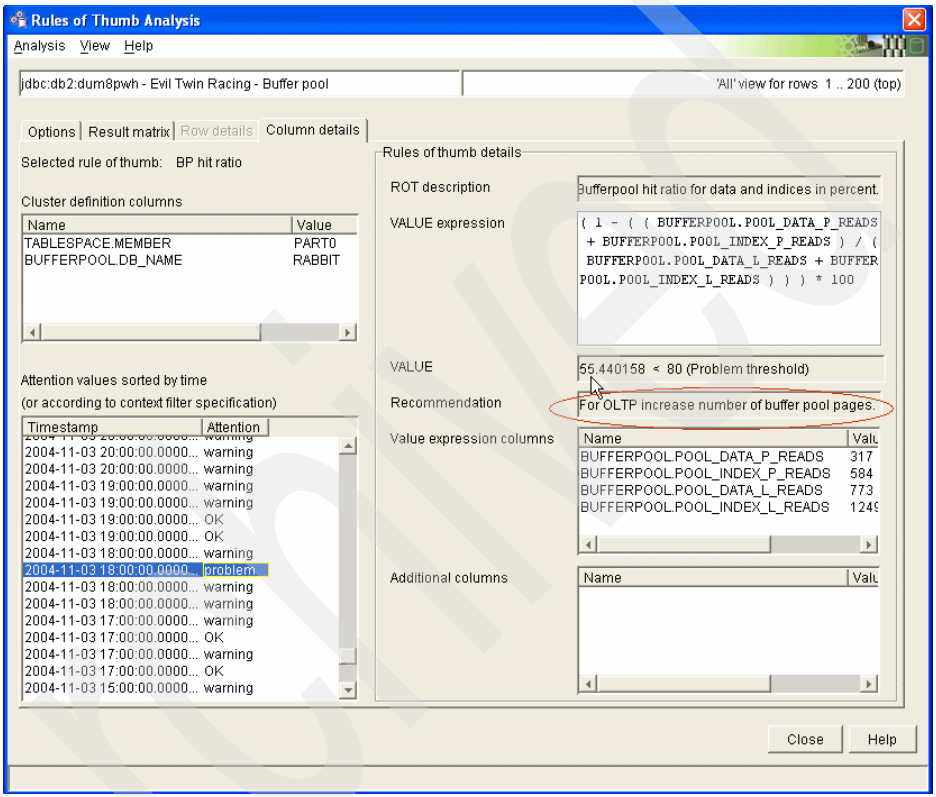


Figure 6-75 RoT recommendation

Log buffer

The DB2 log buffer can help improve performance by buffering the log records to prevent disk I/O activity. All changes to data and index pages are first recorded as log records to the log buffer before being written to disk. The log buffers are written to disk when one of the following occurs:

- A transaction commits or a group of transactions commit, as defined by the MINCOMMIT configuration parameter.
- When the log buffer is full.

- Prior to corresponding, data is written to disk when write ahead logging is used.
- Prior to changes made to metadata that resulted from executing DDL statements.

The log buffer size is controlled by the database configuration parameter LOGBUFSZ. This parameter allows you to specify the amount of the database heap (defined by the DBHEAP parameter) to use as a buffer for log records.

Log buffer tuning

An OLTP environment has a high volume of short transactions, which results in a high volume of COMMITS. The result is frequent physical writes to the log. The DB2 database configuration parameter MINCOMMIT allows you to delay written log records to disk until a minimum number of COMMITS have been performed. That is, each physical write now requires multiple COMMITS. By combining the log writes for multiple COMMITS, the performance is improved.

The trade off is that each individual transaction's COMMIT may be delayed until the threshold number of COMMITS is reached. This trade-off may not be suitable for a DSS database system, but is favorable for an OLTP environment.

We use PE Statistic Details to examine detail log buffer usage; both Log pages written and read should be checked. As shown in Figure 6-76, while we examine our application, the number of Log pages written is 8,217 and Log pages read is zero. The ideal situation is to have a high number of pages written and the log buffer pages read to be 0. For our application, the log buffer usage is OK.

Since log buffer usage will change once transaction volume or data changes, routine monitoring of log buffer usage is recommended. If the number of log records written increases to a high number, analyze and adjust the database parameters LOGBUFSZ, DBHEAP, and MINCOMMIT again. You can use PE online or short-term history mode to view the application or database activities for the past few days. In an OLTP environment, you want to set the log buffer to at least 256 pages.

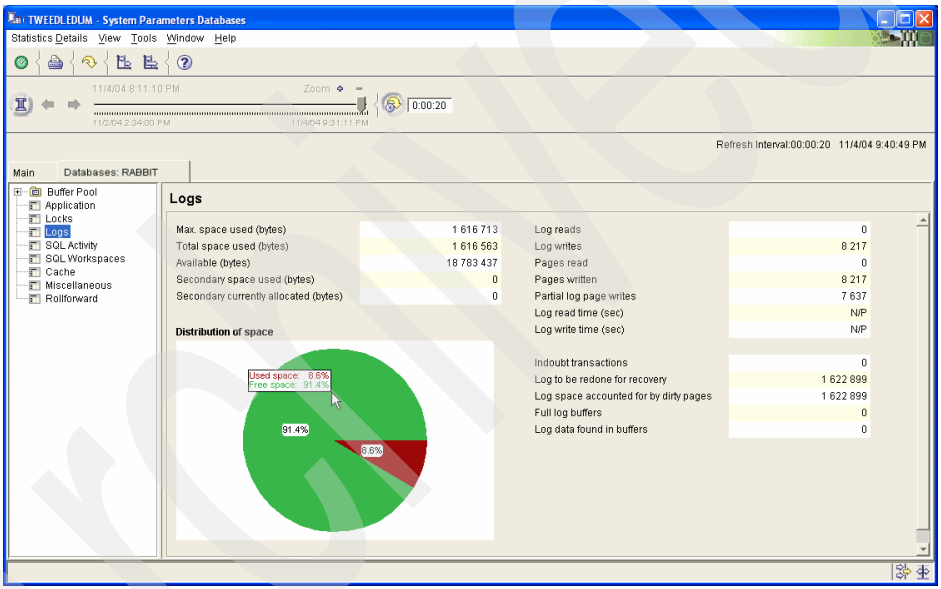


Figure 6-76 Log buffer read

Sort heap

Sort heap defines the maximum number of private memory pages to be used for sorts. Private sorts only affect the agent's private memory. Shared sorts affect the database shared memory. Each sort has a separate sort heap size that is allocated by the database manager when needed.

The DB2 optimizer considers the performance impact of sort as one factor in selecting the access plan. The DB2 optimizer can allocate a smaller sort heap than the one that was originally set by the sort heap parameter in the database configuration.

Considerations for sorting:

- Sorting occurs when no index satisfies the requested ordering (for example a SELECT statement that uses the ORDER BY clause over a non-indexed column.
- Sorting occurs when an index is created and dropped (dropping the index causes the index pages number to be sorted).
- The DB2 optimizer determines that a sort has less of an impact than an index scan. The DB2 optimizer sorts data by:
  - Piping the results of the sort when the query is executed
  - Internally handling of sorts within the database manager
- Hash join buffers and dynamic bitmaps (used for index ANDing and Star Joins) use sort heap memory.
- If many large sorts are done frequently, then you will need to increase the database sort heap parameter.
- If you increase the size of the database sort heap parameter, then you need to consider the following two items:
  - Increasing the size of the sort heap threshold parameter.
  - Rebind applications, because the sort heap size is used by the optimizer to determine the access path.

Sort heap tuning

Sorts can either increase or decrease your system's performance. The DB2 optimizer may decide to help boost performance by using a sort.

In a decision support environment where the queries are more complex, sorting is more common. If you have multiple queries doing sorts, increasing the sort heap value should help improve performance.

In a OLTP environment, when many concurrent transactions occur, you want the number of sorts to be very low. A high number of sorts can be very expensive to your system's CPU and I/O. In OLTP, we recommend that developers try to rewrite their queries or create additional indexes to avoid sorting. These queries are usually short and simple. This is why we recommend initially setting the sort heap size to 256 4 KB pages in an OLTP environment, but you will want to continue to monitor this value. By keeping the sort heap size low, it will reduce the number of sorts by making sorts unattractive to the DB2 optimizer.

We use the PE Performance Warehouse Database report to check the sorting activities that occurred in our application. Figure 6-77 shows the PW HTML report. As you can see, our application has very low sorting activities. The

database parameter for sort heap has already been set to 256 as recommended for OLTP applications; no further tuning is required.

Sorts

Partition ID	Snapshot Time	Active Sorts	Average Sort Time (sec.ms)	Total Sort Heap Allocated	Sort Overflows	Sort Share Heap Currently Allocated	Sort Share Heap High Water Mark	Total Sort Time (sec.ms)	Total Sorts
member_id	interval_to	active_sorts	avg_sort_time	sort_heap_allocated	sort_overflows	sort_shrheap_allocated	sort_shrheap_top	total_sort_time	total_sorts
0	13:00:00.0	0	0.007	0	0	0	0	0.007	1
0	14:00:00.0	0	0	0	0	0	0	0	1
0	15:00:00.0	0	0	0	0	0	0	0	1
0	17:00:00.0	0	0	0	0	0	0	0	1
0	18:00:00.0	0	0	0	0	0	0	0	0
0	19:00:00.0	0	0	0	0	0	0	0	1
0	20:00:00.0	0	0	0	0	0	0	0	0

Figure 6-77 Sort activity report

Locks

DB2 uses locks to help improve performance by allowing as many concurrent applications to access the same table as possible while still guaranteeing work consistency. Locks are normal occurrences and can occur at different levels, rows, tables, and applications. If locks are not used properly, then they can decrease the system's performance. This is why it is very important to investigate locks that occur on the database server.

DB2 provides you three database configuration parameters to help you tune your application/database: LOCKLIST, MAXLOCKS, and LOCKTIMEOUT.

- LOCKLIST

  This indicator tracks the amount of lock list memory that is being used. The database has one lock list, which contains the locks held by applications concurrently connected to the database. Once the lock list memory is used and is unavailable, your database performance decreases.

  If the locklist memory is completely used, then the following occurs:

  - Lock escalation converts from row locks to table locks, which makes concurrent access to tables unavailable, thus serializing access to the table and thereby causing processing delays.
  - Deadlocks are more likely to occur because now the granularity (size of the object locked) is greater going from row to table, making deadlocks between transactions more likely to occur. If a deadlock occurs, then transactions are rolled back.

- MAXLOCKS

This is the percentage of the locks list held by a single application before the database manager performs an escalation. A lock escalation occurs when the number of locks held by one application equal the MAXLOCKS percentage or when locklist runs out of space. If a lock escalation occurs, then it replaces row locks with table locks for the locks held by that application, thereby decreasing the total number of locks the application requires.

If MAXLOCKS is set too low, then lock escalation happens even though there is still enough lock space for other concurrent applications.

If MAXLOCKS is set too high, then a few applications can consume most of the lock space, and other applications will have to perform lock escalation. The need for lock escalation in this case results in poor concurrency.

- LOCKTIME

  This is the number of seconds an application will wait to obtain a lock. Lock requests will wait for a lock to be granted or until a deadlock occurs. If a deadlock occurs, then one of the applications involved in the deadlock will be roll backed. If LOCKTIME out is set to 0, then lock waits do not occur. If LOCKTIME out is set to -1, the lock timeout is disabled.

  In an OLTP environment, we recommend initially setting LOCKTIME to 30 seconds. In a query-only environment, you could start with a high value. In either situation, you will want to continue to monitor this value to prevent performance issues.

The above parameters can help you tune your system to help prevent locking from occurring. Another possible solution to resolving locking is to follow a couple of the guidelines listed below for tuning locks:

- Release locks frequently by issuing COMMIT.
- If you are performing many updates on a single table, we recommend locking the table using the SQL statement LOCK TABLE. When using the LOCK TABLE statement, only the table specified in the LOCK TABLE statement is locked and will not lock the parent and dependent tables.

  To control how locking is done for a particular table, use the LOCKSIZE option in the ALTER TABLE statement.
- Determine the best isolation level.

  Using the Repeatable Read isolation level might result in an automatic table lock.

  Use the Cursor Stability isolation level when possible to decrease the number of share locks held. If application integrity requirements are not compromised, use Uncommitted Read instead of Cursor Stability to further decrease the amount of locking.

Using Repeatable Read, Read Stability, and Cursor Stability, all locks are held until a COMMIT is issued, unless you close your cursor using the WITH RELEASE clause.

Locks tuning

Locks can cause server performance problems. During this testing process, we turn on PE Event Exception monitoring for deadlocks and PE Periodic Exceptions monitoring for locks.

While running our application, we received an e-mail notification from PE event exception monitoring informing us that our database had problems with locks. With our first glance of the PE System Overview windows, we immediately noticed the 20 most recent event exceptions were highlighted in red (see Figure 6-78).

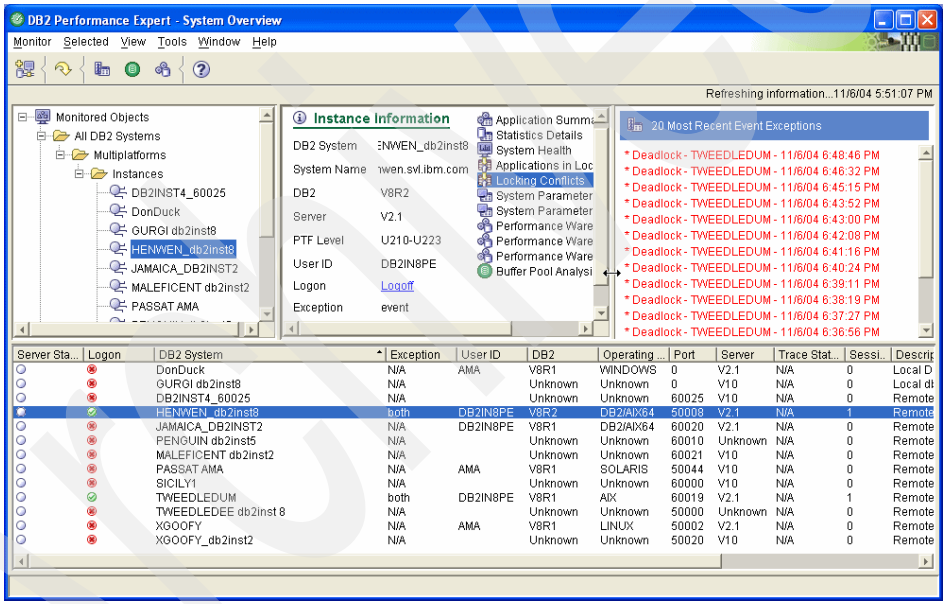


Figure 6-78 Most event excepts

From the System Overview window, we have the option to double-click the items highlighted in red under the 20 Most Recent Event Exceptions or open Applications in Locking Conflicts to find more information. We chose to investigate this problem by using Applications in Locking Conflicts. As you can see in Figure 6-79 on page 453, application ID 318 held an exclusive lock on a table, which causes other applications to wait.

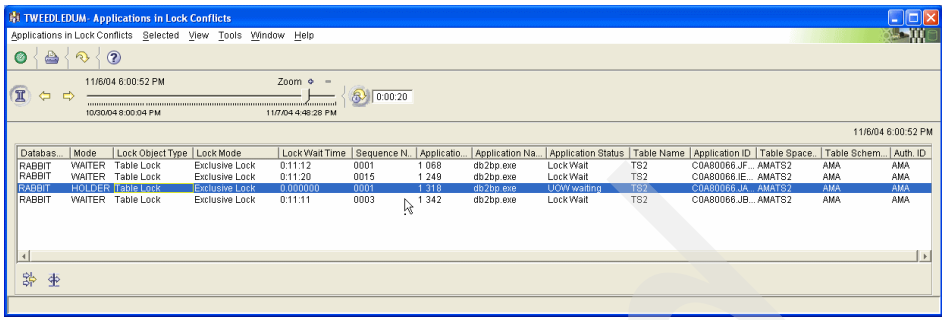


Figure 6-79 Application in lock conflict

We also have the option to drill down to obtain more information regarding this lock by double-clicking on Application Handle ID 318. In Figure 6-80, we see that “Locks held by application” is 6, “Locks waits since the connect” is 1, “Deadlocks detected” is 1, and “Time applications waited on locks” is 8.21 seconds.

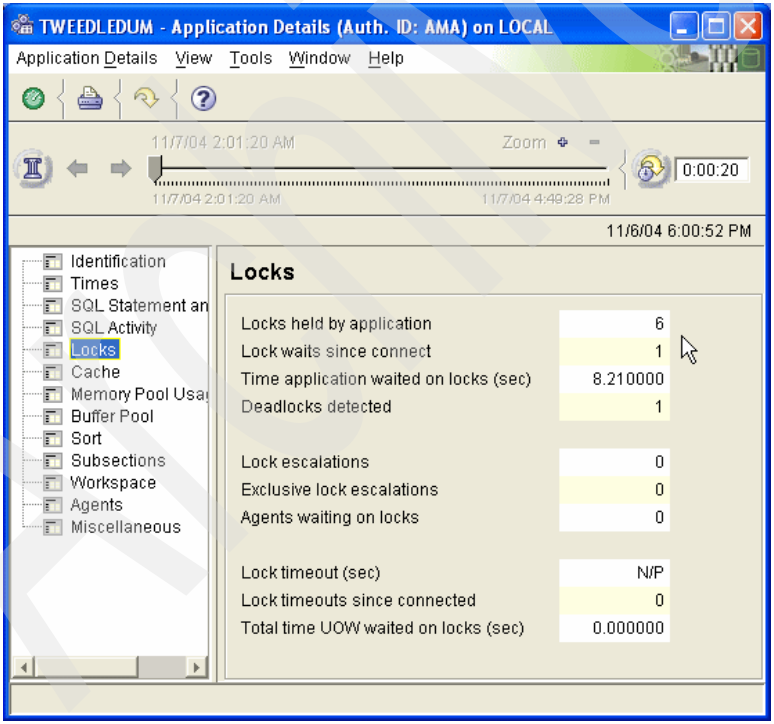


Figure 6-80 Locks details

We checked our database configuration parameters for LOCKLIST, MAXLOCKS, and LOCKTIMEOUT, and they are all at the default level:

```
$db2 get db cfg | grep LOCK
 Max storage for lock list (4KB)             (LOCKLIST) = 100
 Percent. of lock lists per application      (MAXLOCKS) = 10
 Lock timeout (sec)                        (LOCKTIMEOUT) = -1
```

After reviewing the above information, we discovered that our LOCK TABLE statement shown in Figure 6-81 should not be locking the TS2 table. We also discovered that the LOCKTIMEOUT parameter needs to be updated. We first corrected the LOCK TABLE SQL statement and verified the result via PE. We then updated the LOCKTIMEOUT value using the recommended initial value of 30 for an OLTP. Then we re-ran our application. After that, we did not see any applications in locking conflicts. We continue to use PE exception precessions to monitor for deadlocks.

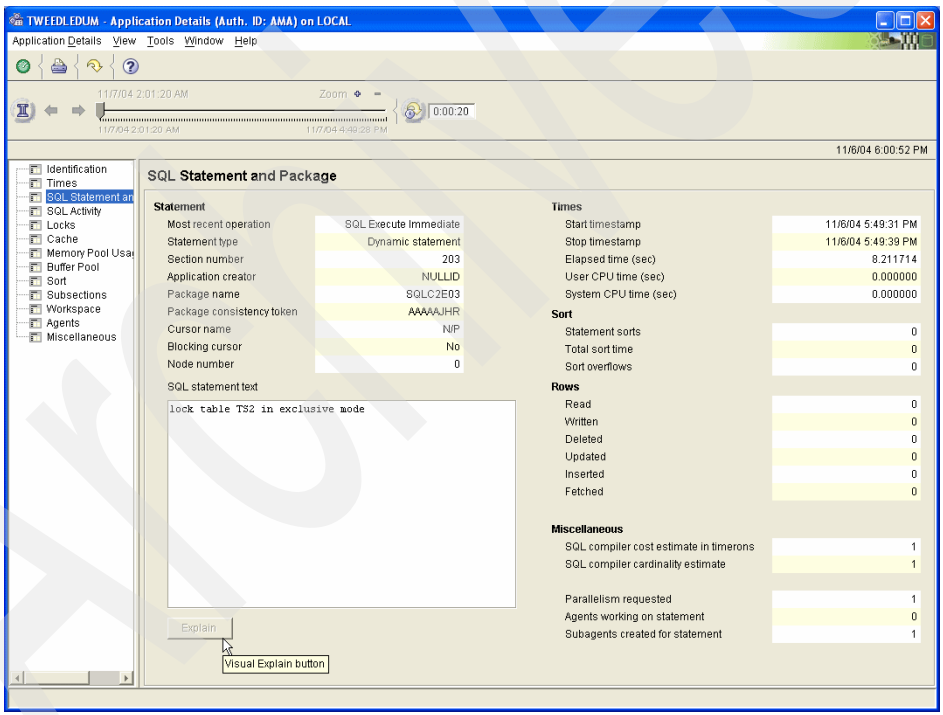


Figure 6-81 SQL statement and package

Summary

Using PE to verify the database design for an application is another way of using PE functions and features. We have shown you the steps using an OLTP application. Other areas, such as the number of asynchronous page cleaners,

application heap, maximum number of active applications, and so on, should also be checked. The more you check and test your design before deploying your application, the less problem you will have in the future.

A

Rules of Thumb definition samples

Rules of Thumb (RoTs) apply a few simple rules and ratios to key performance indicators. DB2 Performance Expert provides sample Rules of Thumbs that are assembled by DB2 experts in the field. These samples are valuable criteria in measuring the performance of applications in a DB2 system.

The predefined Rule of Thumb cluster definitions for Multiplatforms and Workgroups consist of measurements for performance in four areas:

- Buffer pool
- Database
- SQL activity
- Tablespace

This appendix listed the details of sample Rules of thumb definitions. These definitions are also included in the online help of the PE.

Buffer-pool cluster definitions

The buffer pool cluster contains RoTs that evaluate counters related to buffer pools. This cluster includes five measurements:

- AsynchVsSynchRead
- AsynchVsSynchWrite
- BP hit ratio
- BP hit ratio data
- BP hit ratio index

AsynchVsSynchRead

This counter yields the ratio of asynchronous (prefetched) to synchronous (random) physical read I/Os in percent. The formula used in the Value field is:

(POOL_ASYNC_DATA_READS + POOL_ASYNC_INDEX_READS) / (POOL_DATA_P_READS + POOL_INDEX_P_READS) * 1000

Table A-1 on page 459 shows the value of comparison expression, warning threshold, problem threshold, and the content of recommendation for counter AsynchVsSynchRead.

Table A-1 AsynchVsSynchRead counter values

Comparison	Warning threshold	Problem threshold	Recommendation
Expression value <	97	90	These threshold values apply to BI/DW environments. For a mixture of BI/DW and OLTP environments or for OLTP environments only, lower prefetch ratios are expected. To improve the prefetch ratio, try the following: ► Check the value for the number of I/O servers (NUM_IOSERVERS). For BI/DW environments, the value should match the disk configuration, for example, the number of physical disks, but not exceed four to six times the number of CPUs. For OLTP environments, NUM_IO_SERVERS should be equal to the number of CPUs, but not less than the default of three. If your physical tablespace location is RAID5 but your tablespace has only one container, set the registry variable DB2_PARALLEL_IO to ensure parallel I/O. ► Check and adapt your extent and prefetch size values. The prefetch size PREFETCHSIZE of your tablespace should be a multiple of the number of tablespace containers and of the extent size EXTENTSIZE of your tablespace. The tablespace containers should be located on different devices. ► Check whether the tables and indexes must be reorganized (REORGCHK command). A decreasing prefetch ratio over time can indicate a need for reorganization. ► Maintain clustering indexes on the most important columns of a table.

AsynchVsSynchWrite

This counter yields the ratio of asynchronous to synchronous physical read I/Os in percent. The formula used in the Value field is:

(POOL_ASYNC_DATA_WRITES + POOL_ASYNC_INDEX_WRITES) / (POOL_DATA_WRITES + POOL_INDEX_WRITES) * 100

Table A-2 shows the value of comparison expression, warning threshold, problem threshold, and the content of recommendation for counter AsynchVsSynchWrite.

Table A-2 AsynchVsSynchWrite counter values

Comparison	Warning threshold	Problem threshold	Recommendation
Expression value <	95	90	Possible actions: ► Increase the number of I/O cleaners (NUM_IOCLEANERS) to the number of CPUs. ► Reduce the changed-page threshold value (CHNGPGS_THRESH) to a percentage equal to, or higher than, 30% or the SOFTMAX value.

BP hit ratio

This is buffer pool hit ratio for data and indexes in percent. The formula used in the Value field is:

(1 - ((POOL_DATA_P_READS + POOL_INDEX_P_READS) / (POOL_DATA_L_READS + POOL_INDEX_L_READS))) * 100

Table A-3 on page 461 shows the value of comparison expression, warning threshold, problem threshold, and the content of recommendation for counter BP hit ratio.

Table A-3 BP hit ratio counter values

Comparison	Warning threshold	Problem threshold	Recommendation
Expression value <	90	80	If you are running OLTP applications, increase the number of buffer pool pages. For BI/DW applications, a hit ratio that is less than the threshold values is acceptable if the prefetch ratio is high. (See the appropriate Rule of Thumb.) To assess your buffer pool hit ratio correctly, also take the following into account: ► The space used in all tablespaces. For example, a hit ratio of 95% is easy to achieve if a 100 MB buffer pool only serves 150 MB of tablespace data. However, if a 100 MB buffer pool serves 3 GB of tablespace data, a hit ratio of 95% is a very good result. ► Database usage. When the database is first used (that is, the buffer pool is still empty) after a database manager startup or a database activation, the hit ratio is much lower. ► A hit ratio that is decreasing over time indicates that the data volumes are growing or the tables must be reorganized.

BP hit ratio data

This is the buffer pool hit ratio for data in percent. The formula used in the Value field is:

(1 - ((POOL_DATA_P_READS) / (POOL_DATA_L_READS))) * 100

Table A-4 on page 462 shows the value of comparison expression, warning threshold, problem threshold, and the content of recommendation for counter BT hit ratio data.

Table A-4 BP hit ratio data counter values

Comparison	Warning threshold	Problem threshold	Recommendation
Expression value <	90	80	If you are running OLTP applications, increase the number of buffer pool pages. For BI/DW applications, a hit ratio that is less than the threshold values is acceptable if the prefetch ratio is high. (See the appropriate Rule of Thumb.) To assess your buffer pool hit ratio correctly, also take the following into account: ► The space used in all tablespaces. For example, a hit ratio of 95% is easy to achieve if a 100 MB buffer pool only serves 150 MB of tablespace data. However, if a 100 MB buffer pool serves 3 GB of tablespace data, a hit ratio of 95% is a very good result. ► Database usage: When the database is first used (that is, the buffer pool is still empty) after a database manager startup or a database activation, the hit ratio is much lower. ► A hit ratio that is decreasing over time indicates that the data volumes are growing or the tables must be reorganized.

BP hit ratio index

This is the buffer pool hit ratio for indexes in percent. The formula used in the Value field is:

(1 - ((POOL_INDEX_P_READS) / (POOL_INDEX_L_READS))) * 100

Table A-5 on page 463 shows the value of comparison expression, warning threshold, problem threshold, and the content of recommendation for the counter BP hit ratio index.

Table A-5 BP hit ratio index

Comparison	Warning threshold	Problem threshold	Recommendation
Expression value <	90	80	For BI/DW and OLTP applications, increase the number of buffer pool pages. To assess your buffer pool hit ratio correctly, also take the following into account: ► The space used in all tablespaces. For example, a hit ratio of 95% is easy to achieve if a 100 MB buffer pool only serves 150 MB of table-space data. However, if a 100 MB buffer pool serves 3 GB of table-space data, a hit ratio of 95% is a very good result. ► Database usage: When the database is first used (that is, the buffer pool is still empty) after a database manager startup or a database activation, the hit ratio is much lower.

Database cluster definitions

The database cluster contains Rules of Thumb that evaluate counters related to databases.

CatlgCacheHitRatio

This value indicates whether the CATALOGCACHE_SZ configuration parameter is set appropriately. The formula used in the Value field is:

(1 - (CAT_CACHE_INSERTS / CAT_CACHE_LOOKUPS)) * 100

Table A-6 on page 464 shows the value of comparison expression, warning threshold, problem threshold, and the content of recommendation for counter CatlgCacheHitRatio.

Table A-6 CatlgCacheHitRatio counter values

Comparison	Warning threshold	Problem threshold	Recommendation
Expression value <	90	80	Increase the CATALOGCACHE_SZ configuration parameter.

Files closed

This value indicates that the maximum number of files that can be concurrently open is exceeded. DB2 starts closing the files.

The MAXFILOP database configuration parameter defines the maximum number of files that can be concurrently open in DB2. When this number is reached, DB2 starts closing and opening its tablespace files (including raw devices), which decreases the SQL response time and consumes CPU cycles. This query checks whether DB2 is closing files. If this is the case, increase the value of MAXFILOP until the opening and closing of files stops. Use the following command to update the MAXFILOP:

```
db2 "update db cfg for DBNAME using MAXFILOP n"
```

Table A-7 shows the value of comparison expression, warning threshold, problem threshold, and the content of recommendation for counter File closed.

Table A-7 Files closed counter value

Comparison	Warning threshold	Problem threshold	Recommendation
Expression value >	0	1	Increase the value of MAXFILOP.

PkgCacheHitRatio

Indicates whether the PCKCACHESZ configuration parameter is set appropriately. The formula used in the Value field is:

(1 - (PKG_CACHE_INSERTS / PKG_CACHE_LOOKUPS)) * 100

Table A-8 on page 465 shows the value of comparison expression, warning threshold, problem threshold, and the content of recommendation for counter PkgCacheHitRatio.

Table A-8 PkgCacheHitRatio counter value

Comparison	Warning threshold	Problem threshold	Recommendation
Expression value <	95	90	Increase the PCKCACHESZ configuration parameter. In OLTP systems, pay more attention to the package cache hit ratio than in BI/DW systems. In OLTP systems, use parameter markers to make your SQL statement strings generic.

RowsReadVsSelected

Indicates how many rows have to be read until the target rows are found. The formula used in the Value field is:

ROWS_READ/ROWS_SELECTED

Table A-9 shows the value of comparison expression, warning threshold, problem threshold, and the content of recommendation for counter RowsREadVsSelected.

Table A-9 RowREadVsSelected counter value

Comparison	Warning threshold	Problem threshold	Recommendation
Expression value >	5	10	For OLTP applications, improve the indexes. For BI/DW applications, improve the indexes and create MQTs (materialized query tables), where appropriate. For BI/DW applications, these values are difficult to achieve and not as important as for OLTP.

Sort overflow

A sort must be continued on disk after being started in memory. If the sort overflow (SORT_OVERFLOWS / TOTAL_SORTS) * 100 exceeds three percent, serious and unexpected sort problems can occur in the application SQL. Because overflows indicate large sorts, less than one percent sort overflow would be ideal. If excessive sort overflows occur, identify the SQL that is causing the sorts and change the SQL, indexes, or clustering to avoid or reduce the sort cost. (If you increased the size of SORTHEAP you would mask the real performance problem.)

The formula used in the Value field is:

(SORT_OVERFLOWS / TOTAL_SORTS) * 100

Table A-10 shows the value of comparison expression, warning threshold, problem threshold, and the content of recommendation for counter Sort overflow.

Table A-10 Sort overflow counter value

Comparison	Warning threshold	Problem threshold	Recommendation
Expression value >	2	3	For BI/DW applications, it is difficult to eliminate sort overflows. To limit sort overflows, do the following: ► Check whether the statistics data is up to date. If not, run RUNSTATS. ► Improve the appropriate SQL statements and add indexes to the tables used in the SQL statements. ► Increase the size of the SORTHEAP parameter. In addition, check whether the SHEAPTHRES parameter also needs adjustment. It should be at least two times the largest SORTHEAP parameter defined for any database within the DB2 instance.

SQL activity cluster definitions

The SQL activity cluster contains Rules of Thumb that evaluate simple counters. These counters are compared with values that are a means to reduce the size of the result of the underlying query. The filter expression in the cluster definition excludes the following operations:

- Static Commit
- Static Rollback
- Prepare
- Open
- Describe
- Compile

When you are creating a Rule of Thumb (using the Rule-of-Thumb Properties window) and add the column STMT_TEXT from the EVM_STMT_TEXTS table to the Additional columns field, the Additional columns field on the Row Details and

Column Details pages of the Rules of Thumb Analysis window contains the SQL statement executed.

When you sort the analysis result by the arithmetic expression of a Rule of Thumb, the following Rules of Thumb help you find the most time-consuming or most frequently run SQL statements.

mostCPUTime

Find the SQL statements that consume the most CPU time. The formula used in the Value field is:

EVM_STMT_OPERATIONS.USER_CPU_TIME

Table A-11 mostCPUTime counter value

Comparison	Warning threshold	Problem threshold	Recommendation
Expression value >	0.1	0.5	Determine the indexes that can improve the performance of the statements that consume the most time, for example, by using the index advisor.

Determine the indexes that can improve the performance of the statements that consume the most time, for example, by using the index advisor.

mostExecTime

Find the SQL statements that consume the most execution time. The formula used in the Value field is:

EVM_STMT_SUMMARY.TOTAL_EXECUTION_TIME

Table A-12 shows the value of comparison expression, warning threshold, problem threshold, and the content of recommendation for counter mostExecTime.

Table A-12 mostExecTime counter value

Comparison	Warning threshold	Problem threshold	Recommendation
Expression value >	0.01	0.05	Determine the indexes that can improve the performance of the statements that consume the most time, for example, by using the index advisor.

mostFrequRun

Find the SQL statements that were run the most frequently. The formula used in the Value field is:

.EVM_STMT_SUMMARY.TOTAL_NUMBER_OF_EXECUTIONS

Table A-13 shows the value of comparison expression, warning threshold, problem threshold, and the content of recommendation for counter mostFrequRun.

Table A-13 mostFrequRun

Comparison	Warning threshold	Problem threshold	Recommendation
Expression value >	3	5	Take into account how often the SQL statement was run when determining the index.

mostSortTime

Find the SQL statements that consume the most sort time. The formula used in the Value field is:

EVM_STMT_SUMMARY.TOTAL_SORT_TIME

Table A-14 shows the value of comparison expression, warning threshold, problem threshold, and the content of recommendation for counter mostSortTime.

Table A-14 mostSortTime

Comparison	Warning threshold	Problem threshold	Recommendation
expression value >	0.1	0.5	Determine the indexes that can improve the performance of the statements that consume the most sort time, for example, by using the index advisor. Check also whether sort overflows occur by using the appropriate Rule of Thumb and follow its recommendation. Sort overflows can increase the time used for sorts.

Tablespace cluster definitions

The tablespace cluster contains Rules of Thumb that evaluate counters related to tablespaces.

AsynchVsSynchRead

Yields the ratio of asynchronous (prefetched) to synchronous (random) physical read I/Os in percent. The formula used in the Value field is:

(POOL_ASYNC_DATA_READS + POOL_ASYNC_INDEX_READS) / (POOL_DATA_P_READS + POOL_INDEX_P_READS) * 100

Table A-15 shows the value of comparison expression, warning threshold, problem threshold, and the content of recommendation for counterAsnchVsSynchREad.

Table A-15 AsynchVsSynchRead counter value

Comparison	Warning threshold	Problem threshold	Recommendation
Expression value <	10	5	To reach a high buffer pool hit ratio, in particular for OLTP applications, in an environment with several buffer pools, it is recommended to separate tablespaces with a high percentage of asynchronous reads from those with a low percentage of asynchronous reads. An asynchronous read ratio of 10 percent is an optimal threshold for separating the tablespaces into different buffer pools.

.

B

Monitoring CM databases with Performance Expert

This appendix is intended to help Content Management (CM) system administrators use IBM DB2 Performance Expert (PE) for MP V2.2 to monitor and analyze the performance of Content Manager DB2 databases. This appendix provides guidelines for using Performance Expert in different CM performance monitoring and analyzing tasks, for example, determining the reason for bad response times or discovering peak and normal workload times.

IBM DB2 Performance Expert for MP V2.2.0.1 (V2.2 Fix Pack 1) which has been available since January 2006, includes some CM database specific features. These features are explained in the appropriate chapters.

Performance tuning is affected by many factors, including the workload and the system environment. Also, in a CM environment, performance tuning applies to many components, including the CM Library Server, CM Resource Manager databases, applications, and middle tier servers. Performance bottlenecks can occur in all the components as well as in the underlying operating systems. However, Performance Expert currently supports performance monitoring and tuning only on the CM databases and their operating systems.

Introduction to CM database monitoring

This section describes the environment in which Performance Expert (PE) is used to monitor CM Library Server and CM Resource Manager databases. It also lists the performance monitoring and analyzing tasks that you can use Performance Expert to accomplish in order to ensure the best performance of the CM DB2 databases.

Understanding the CM and Performance Expert environments

IBM DB2 Content Manager V8.3 Enterprise Edition is the core of DB2 Content Management software. It manages all types of digitized content, including HTML and XML Web content, document images, electronic office documents, printed output, audio, and video.

IBM DB2 Performance Expert for MP V2.2 is a performance monitoring and analysis tool that simplifies DB2 performance management and tuning. From a single user interface, the Performance Expert client, you can view your CM Library Server and CM Resource Manager databases and monitor applications, system statistics, system parameters, and the operating system in real-time and historical-mode. The DB2 performance metrices that the Performance Expert client provides are based on DB2 snapshot and DB2 event monitor data. Additionally, Performance Expert raises exceptions to anticipate emerging DB2 performance and availability problems and lets you generate reports that show buffer pool, database, or SQL activity over time in order to identify trends or workload peak times.

Figure B-1 on page 473 shows the major CM and Performance Expert components in a environment in which Performance Expert is set up to monitor the CM Library Server and CM Resource Manager databases.

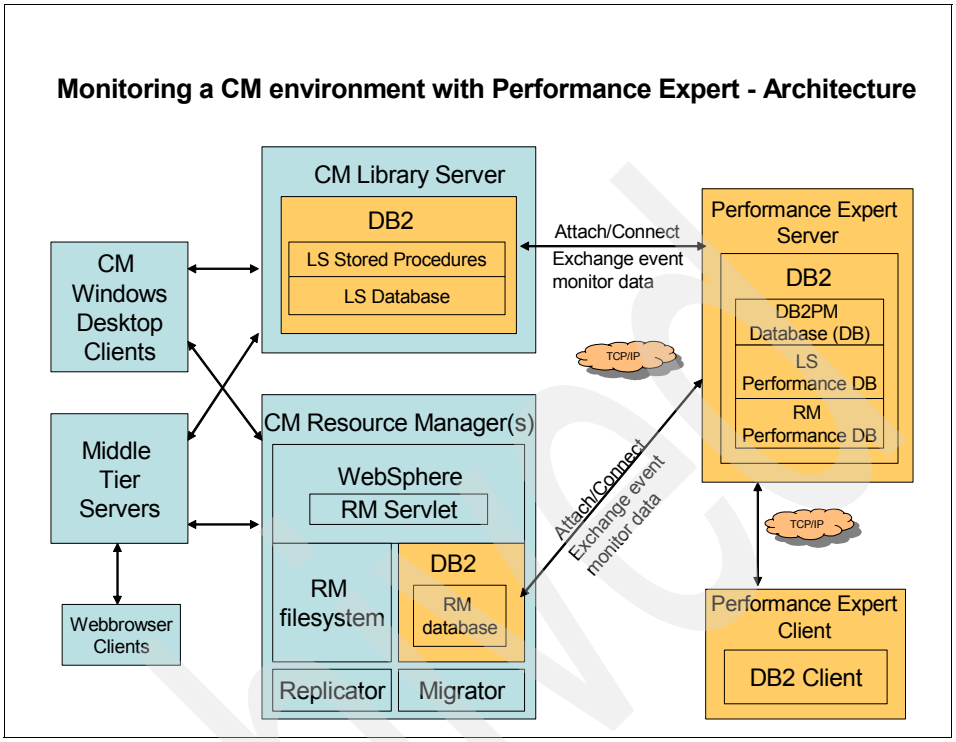


Figure B-1 CM and PE high-level architecture

Performance Expert server runs in a DB2 instance and attaches to the CM Library Server (LS) and CM Resource Manager (RM) DB2 instances to retrieve DB2 snapshot data. All retrieved data is stored in performance databases on the Performance Expert server.

If requested by the Performance Expert user, the Performance Expert server connects to the LS database or the RM databases and starts DB2 event monitors for discovering deadlocks or tracing SQL activity. The event monitor data is written to the CM Library Server and CM Resource Manager machines. The event monitor data is then exchanged with Performance Expert server either via a shared file system or via a DB2 user-defined function that is located on the CM Library Server and CM Resource Manager.

Optionally, you can also use Performance Expert to monitor the operating system of your CM Library Server and CM Resource Manager (this capability is not shown in the Figure B-1 on page 473). This functionality is currently supported only if your CM Library Server and CM Resource Manager run either on AIX 5L V5.2 ML03 or later, or on Solaris 9 or later. Additionally:

- On AIX, an open-source implementation of the Common Information Model Object Manager (CIMOM) called Pegasus Version 2.3.2 or later of AIX Expansion Pack 5.2.6 or AIX Expansion Pack 5.3.2 must be installed and running.
- On Solaris: WBEM Services 2.5 of Solaris 9 or later must be installed and running. CIMOM is part of WBEM Services 2.5.

Recommendations

Here are a couple of recommendations for this environment:

- For maximum scalability, you should separate the different CM components (CM middle-tier servers, CM Library Server, and CM Resource Manager) either by installing them on different machines or by using different DB2 instances for the CM Library Server and CM Resource Manager (if you decide to install both on the same machine). In the latter case, using different DB2 instances instead of a single DB2 instance allows you to fine-tune each instance to better support the different type of workloads that are executed against these CM V8 server components. If you install the CM Library Server and CM Resource Manager on the same machine and you have times with heavy workload or your workload increases over time, you might encounter memory constraints. If this occurs, consider splitting the two components to different machines.
- To avoid resource contention and to keep monitoring completely separated from your CM server workload, you should install the Performance Expert server on a different machine than the CM Library Server and CM Resource Manager.

Out of the box CM database tuning

During the installation of the CM V8.3 Enterprise Edition Library Server and CM V8.3 Enterprise Edition Resource Manager, several database manager and database configuration parameters are changed in order to ensure good performance in general. However, depending on your environment and workload, these settings might not completely apply to your environment. You can use Performance Expert to determine whether some of the configuration parameter values must be adapted.

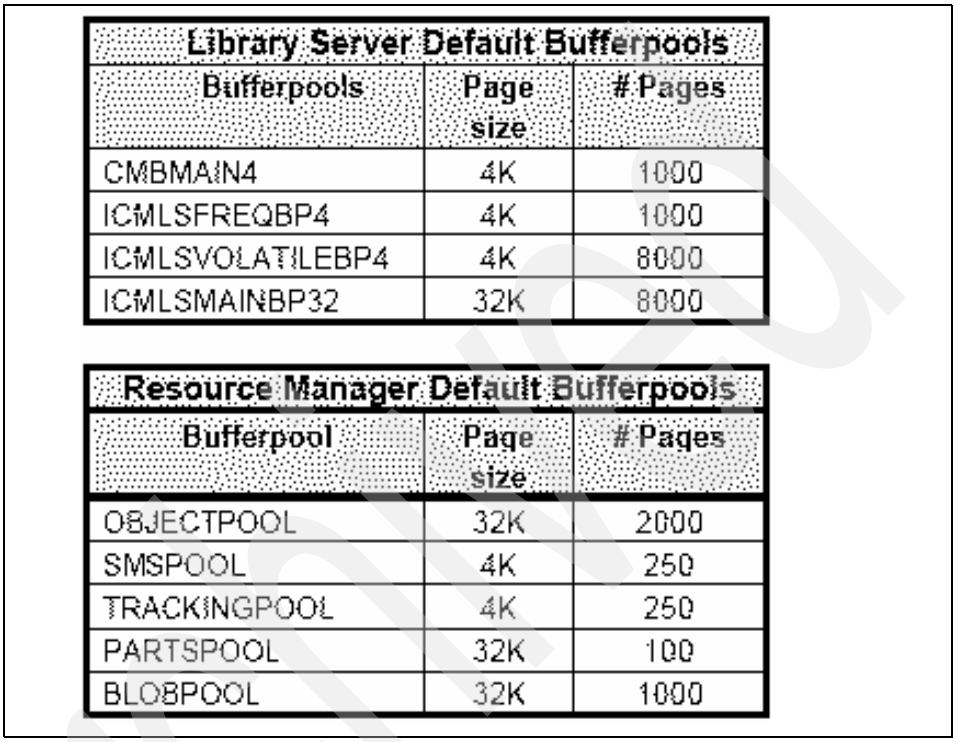
The following configuration parameters (DB2 V8 assumed) are set during the installation of the CM Library Server:

- Database Manager configuration:
  - QUERY_HEAP_SZ 32768
  - UDF_MEM_SZ 7000
  - SHEAPTHRES 10000
  - MAXAGENTS 500
  - AGENT_STACK_SZ 384 (Windows systems only)
- Database configuration:
  - LOCKTIMEOUT 30
  - LOCKLIST 1000
  - STMTHEAP 16384
  - AVG_APPLS 5
  - SORTHEAP 256
  - LOGPRIMARY 10
  - LOGFILSIZ 1000
  - LOGSECOND 20
  - LOGBUFSZ 32
  - MAXAPPLS 200
  - APPLHEAPSZ 1024
  - DFT_QUERYOPT 2
  - DBHEAP 2400
  - APP_CTL_HEAP_SZ 1024

The following configuration parameters (DB2 V8 assumed) are set during the installation of the CM Resource Manager:

- Database Manager configuration:
  - QUERY_HEAP_SZ 32768
- Database configuration:
  - DBHEAP 1000
  - LOCKLIST 1000
  - LOGPRIMARY 25
  - LOGSECOND 50
  - MAXAPPLS 512
  - DFT_QUERYOPT 2
  - APPLHEAPSZ 1024

Buffer pools are created in your CM Library Server database and CM Resource Manager database. The size of the buffer pools are based on the assumption that 2 GB RAM is available on the machine. The buffer pool sizes are shown in Figure B-2.



Library Server Default Bufferpools		
Bufferpools	Page size	# Pages
CMBMAIN4	4K	1000
ICMLSFREQBP4	4K	1000
ICMLSVOLATILEBP4	4K	8000
ICMLSMAINBP32	32K	8000

Resource Manager Default Bufferpools		
Bufferpool	Page size	# Pages
OBJECTPOOL	32K	2000
SMSPOOL	4K	250
TRACKINGPOOL	4K	250
PARTSPOOL	32K	100
BLOBPOOL	32K	1000

Figure B-2 Buffer pools created during CM installation

Monitoring CM databases with Performance Expert

The *IBM DB2 Content Manager V8.3 Enterprise Edition Performance Tuning Guide* describes a CM performance methodology that is based on the following five stages. The *IBM DB2 Content Manager V8.3 Enterprise Edition Performance Tuning Guide* can be found at:

http://www.ibm.com/support/docview.wss?uid=swg27006452

Stages:

1. Planning for performance and scalability

   Document the projected system topology and the projected workload, define measurable performance and scalability objectives, and perform initial capacity planning and sizing estimates.

2. Solution design choices and performance trade-offs

   Understand topology choices (types of client, eClient, Document Manager, Client for Windows, custom client), feature choices (for example, replication, High Availability, text search, versioning, and so on), and their performance implications.

3. Performance tuning

   Plan for an initial tuning period to maximize confidence and reduce risk to meet the objectives that were set in the planning phase. Focus on effective tuning of CM server calls, database and database manager parameters, memory, disk I/O, and network bandwidth.

4. Monitoring and maintaining performance

   Maintain a “performance profile” of the workload metrics and resource utilizations on your CM servers. Monitor over time to observe trends before they become a problem.

5. Performance troubleshooting

   When performance issues arise, use a disciplined and organized approach to solve the problem by using the “performance profile” data and performance tuning guidelines.

Performance Expert can help you to monitor your CM Library Server and CM Resource Manager databases in the following CM Performance Methodology stages:

- Performance Tuning
- Monitoring and maintaining performance
- Performance Troubleshooting

See the following sections for details about how Performance Expert can be used in these stages.

Performance Tuning

During the initial tuning period, the following monitoring tasks using Performance Expert will help you to determine if you need to modify your database or database manager in order to meet the objectives that you set in the planning stage.

- Monitor your overall system health in normal and peak times and adapt configuration parameters if necessary. See the following sections for more information:
  - “Monitoring the health of the CM DB2 databases” on page 480
  - “Monitoring buffer pool effectiveness” on page 502

  - “Discovering peak and normal workload times and monitoring the performance in these times” on page 505

- Monitor the number of connections over time and the memory consumption on your CM Library Server and change the related configuration parameters if necessary. This helps to avoid the memory problems that are described in “Performance Troubleshooting” on page 479. Additionally, it helps you to detect your peak and normal workload times. See the following sections for more information:
  - “Monitoring the connections and agents of your CM library server” on page 491
  - “Discovering peak and normal workload times and monitoring the performance in these times” on page 505

Monitoring and maintaining performance

Do the following monitoring tasks regularly using Performance Expert in order to avoid the problems that are described in “Performance Troubleshooting” on page 479 to make sure that your system is running healthy (well tuned) and to detect trends before they become a problem.

- Monitor the number of connections over time and the memory consumption on your CM Library Server and change the related configuration parameters if necessary. By doing so, you can avoid the memory problems that are described in “Performance Troubleshooting” on page 479. Additionally, it helps you to detect your peak and normal workload times. See the following sections for more information:
  - “Monitoring the connections and agents of your CM library server” on page 491.
  - “Discovering peak and normal workload times and monitoring the performance in these times” on page 505.
- Monitor your overall system health in normal and peak times and change the related configuration parameters if necessary. By doing so, you can avoid most kinds of performance problem. See the following sections for more information:
  - “Monitoring the health of the CM DB2 databases” on page 480.
  - “Discovering peak and normal workload times and monitoring the performance in these times” on page 505.
  - “Monitoring buffer pool effectiveness” on page 502.
  - “Detecting heavy hitter tables” on page 514.
- Monitor whether and how your workload changes over time in terms of workload mix and workload amount. Doing so will help you with capacity

planning and will also help you to proactively detect performance problems before they are experienced by the CM users. Refer to the following sections for more information:

  - "Discovering peak and normal workload times and monitoring the performance in these times" on page 505.
  - "Monitoring stored procedure call throughput" on page 507.
  - For disk space capacity planning, refer to "Monitoring tablespaces and file systems" on page 516

- Running **runstats**, **rebind/db2rbind** and **reorg** is essential to ensure overall good performance; therefore, you should run these tools regularly. Performance Expert can inform you when it is necessary to run them again. For more information, refer to "Detecting the need for running reorg and runstats" on page 501.

Performance Troubleshooting

The following list shows some sample performance troubleshooting scenarios that can occur in a CM environment and that you can diagnose using Performance Expert.

- CM users are complaining about message `Library Server Return Code = 7015, Extended Return Code = -911,` which indicates that the transactions have been rolled back because of a deadlock or timeout. This message affects their work if it occurs regularly. To diagnose this problem, see "Analyzing deadlocks and locking conflicts" on page 497.
- CM users are complaining about the bad response times of their queries. Reasons for bad response times are often table scans that occur because of missing indices on user tables, high sort activity, read and write bottlenecks on the disks, or user queries that return a high amount of data (for example, SELECT *). To diagnose these problems, see the following sections:
  - "Identifying long-running SQL statements" on page 512
  - "Detecting heavy hitter tables" on page 514
  - "Monitoring tablespaces and file systems" on page 516
- You are experiencing memory problems on the CM Library Server machine because most of the memory is consumed by db2fmp and agent processes. Using the DB2 connection concentrator might help here. To diagnose this problem, see the following sections:
  - "Monitoring the connections and agents of your CM library server" on page 491
  - "Monitoring the operating system" on page 520

CM DB2 Database performance monitoring using Performance Expert

This section describes the performance monitoring and analyzing tasks that you can use Performance Expert to accomplish in order to ensure the best performance of the CM DB2 databases in detail.

Monitoring the health of the CM DB2 databases

From the point of view of a CM administrator, a CM system is running healthy if the performance objectives that were defined in the planning for performance and scalability stage are being met over time.

From the point of view of a CM user, a CM system is running healthy if all actions that are executed from a CM client finish without errors and return with a reasonable response time.

From the point of view of a DB2 administrator, a database environment is running healthy if certain performance metrices or calculations based on those metrices have certain values (for example, buffer pool hit ratio) or if certain events do not occur (for example, many locks or deadlocks). If performance metrices are consistently below certain threshold values, these bottlenecks can probably be solved by changing a related configuration parameter. However, missing indices or outdated statistics also can cause bottlenecks.

To meet the CM administrator's performance objectives and to ensure that CM users experience good response times without failures, it is important that your CM Library Server and CM Resource Managers consistently run healthy.

You can use the following Performance Expert functions to monitor the overall health of your CM Library Server and CM Resource Managers databases:

- System Health graphs
- Exception processing

Monitoring the health using System Health graphs

System Health graphs are graphical data views that show how critical performance counters change over time. You can use this information to verify whether your CM Library Servers or CM Resource Managers are running healthy or to detect performance problems immediately. The data views are defined in the System Health component of Performance Expert. Predefined data views are available, or you can define your own data view. More critical data views can be directly placed on the System Overview panel by defining them in the predefined System Overview folder. Doing so allows you to see immediately after you log on

how your system is performing. You can define other data views in a new group, and you can use those data views to fine-tune configuration parameters.

Performance Expert allows exporting data views defined in data groups to an XML file to distribute them to other Performance Expert Clients and import them there. Performance Expert V2.2.0.1 comes with a predefined set of XML files containing data view definitions specific to CM environments. The following described data view definitions are still based on Performance Expert V2.2 and are a subset of the data view definitions contained in the XML files. Refer to “Importing and using CM data views” on page 488 for more information about how to import and use the XML files.

The example System Overview window shown in Figure B-3 contains six data views, all of which are valuable for monitoring your CM Library Servers and Resource Managers.

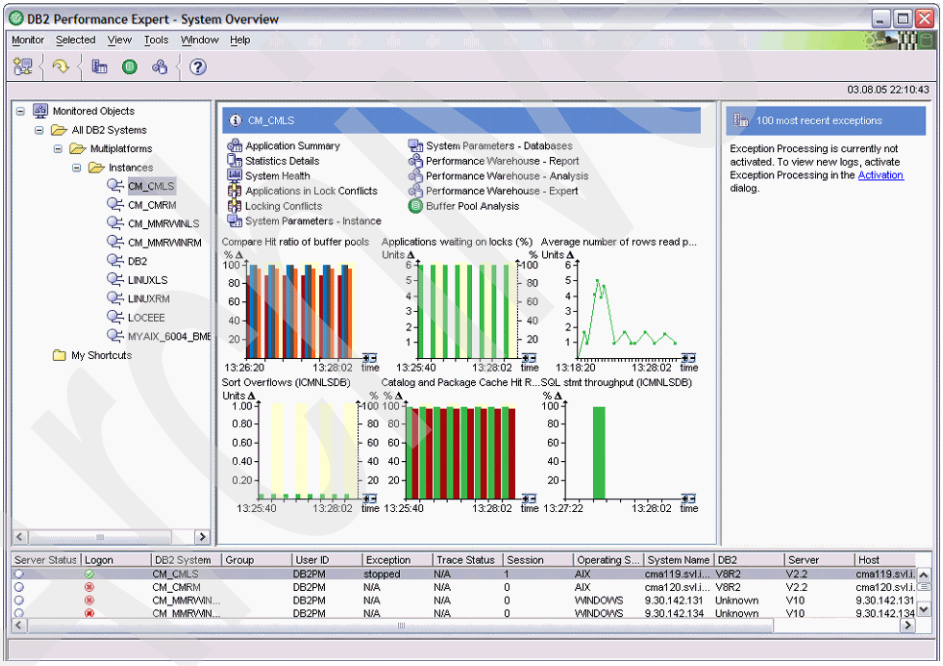


Figure B-3 Performance Expert System Overview panel

The six data views that are shown in the System Overview panel are:

- Compare Hit ratio of buffer pools

  This data view shows the hit ratio percentage for each buffer pool. Buffer pool hit ratios should be between 85% and 100%. If the hit ratio is lower and you have some traffic in the buffer pools, you might consider increasing the buffer pool sizes. See the “Monitoring buffer pool effectiveness” on page 502 for more information.

- Applications waiting on locks (%)

  This data view shows the number of connected applications and the percentage of applications that are waiting on locks. In the System Overview panel that is shown above, no application is waiting for locks, which is good. See “Analyzing deadlocks and locking conflicts” on page 497 if you discover that applications are waiting.

- Average number of rows read per selected row

  This data view shows a ratio that indicates how good your indices are. The ratio should not be higher than 10; optimal values are below 5. Analyze your indices if you see problems here, especially the indices on the ICMUT* tables. See “Identifying long-running SQL statements” on page 512 for more information.

- Sort Overflows

  This data view shows the number of sorts and the percentage of sorts that are overflowed. A sort overflow occurs when the amount of memory that is needed for a sort exceeds the SORTHEAP database configuration value. Also, if your statistics are out of date, an overflow can occur if DB2 requests too small a sort heap and the actual sort operation exceeds the requested amount. Therefore, it is important to keep statistics up to date by regularly running **runstats**/**rbind** on your CM Library Server and CM Resource Manager. Additionally, make sure that the sort is not the result of a missing index. First, update your statistics and check your indices. If you still discover sort overflows that are greater than 3%, increase the SORTHEAP parameter value.

- Catalog and Package Cache Hit Ratio %

  This data view shows the hit ratios of the catalog and package cache in terms of a percentage. Both ratios should be near 100%. The package cache is used for caching sections for static and dynamic SQL statements. Caching packages allows the database manager to reduce its internal overhead by eliminating the need to access the system catalogs when reloading a package, or, in the case of dynamic SQL, eliminating the need for recompilation. The catalog cache is used to cache system catalog information, such as SYSTABLE, authorization, and SYSROUTINES information. Increase the PKGCACHESZ and CATALOGCACHE_SZ

database configuration values if you consistently get hit ratios lower than 90%.

- SQL stmt throughput

  This data view shows the distribution of successful and failed SQL statements running in your databases. Many failed statements might be a reason for poor performance, because failed statements generate some overhead on your system

Figure B-4 shows a System Health panel of the CM data group. The data views in this group can be used to fine-tune some database manager and database parameters of your CM Library Server and CM Resource Managers.

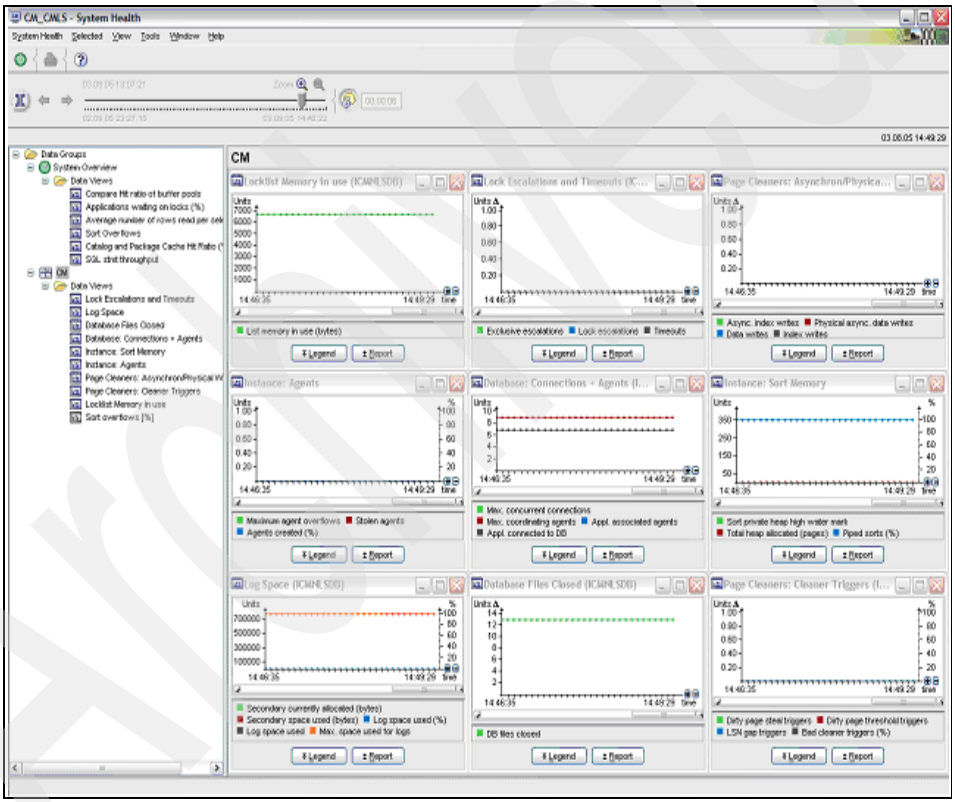


Figure B-4 Performance Expert System Health window

All the data views shown are new data views (none are predefined). During the definition phase, use the Counter compare option, choose dynamic scale, and create them as a line chart.

Two data views can be used to fine-tune the settings for the LOCKLIST, MAXLOCKS, and LOCKTIMEOUT database configuration parameters:

- Locklist Memory in use

  Counter:
  List memory in use (bytes)

- Lock Escalation and Timeouts

  Counters:
  Exclusive escalations

  - Lock escalations
  - Timeouts

There is one lock list per database and it contains the locks held by all applications concurrently connected to the database. On 32-bit platforms, the first lock on an object requires 72 bytes (80 bytes for DB2 V8.2.2 or later), and each additional lock requires 36 bytes (40 bytes for DB2 V8.2.2 or later). On 64-bit platforms, the first lock requires 112 bytes (128 bytes for DB2 V8.2.2 or later), and each additional lock requires 56 bytes (64 bytes for DB2 V8.2.2 or later).

When the percentage of the LOCKLIST that is used by one application reaches MAXLOCKS, the database manager will perform a lock escalation, where row locks for a given table will be traded in for a single table lock. Also, if LOCKLIST is close to being exhausted, the database manager will identify the connection that is holding the most row locks on a table and trade those for a table lock to free up LOCKLIST memory. Locking entire tables can greatly reduce concurrency, and the chance of deadlock is also increased.

If “List memory in use (Bytes)” exceeds 50% of the defined LOCKLIST size, increase the number of 4 KB pages in the LOCKLIST. If there are “Lock escalations” or “Exclusive lock escalations” occurring, increase either LOCKLIST or MAXLOCKS, or both.

The LOCKTIMEOUT value specifies the number of seconds that an application will wait to obtain a lock. This can help avoid global deadlock situations. With -1, the application will appear to freeze if lock-wait is encountered. If “Timeouts” is a high number, it could be caused by the following conditions:

- Too low a LOCKTIMEOUT value
- A transaction holding locks for an extended period
- Lock escalations

Two data views can be used to fine-tune the CHNGPGS_THRESH and NUM_IOCLEANERS database configuration parameters.

- Page Cleaners: Asynchron/Physical Writes

  Counters:

  - Async. Index writes
  - Physical async. data writes
  - Data writes
  - Index writes

- Page Cleaners: Cleaner Triggers

  Counters:

  - Dirty page steal triggers
  - Dirty page threshold triggers
  - LSN gap triggers
  - Bad cleaner triggers (%)

CHNGPGS_THRESH is used to specify the percentage of changed pages in the buffer pool that, when reached, causes the asynchronous page cleaners to start writing the changes to disk in order to make room for new data in the buffer pool. In an OLTP environment, which a CM environment is, a value between 20 and 40 should improve performance (use 20 during periods of very heavy update activity). Lowering the value makes the I/O cleaners more aggressive in their writing out of dirty buffer pool pages, but with less work each time. If there are not a lot of INSERTs or UPDATEs, the default value of 60 should be acceptable.

If “Bad cleaner triggers (%)” is high, try lowering the CHNGPGS_THRESH value. The bad cleaner triggers specify the percentage of the “Dirty page steal triggers” on all cleaner triggers. Also, if "Data writes” or “Index writes" is high and "Async. Index writes” or “Physical async, data writes" is low, try lowering the value as well.

NUM_IOCLEANERS specifies the number of asynchronous page cleaners for a database, which write changed pages from the buffer pool to disk. Start by setting the value for this parameter equal to the number of hard disks used by DB2 on the system. When I/O cleaners are triggered, all of them are started at the same time; therefore, you do not want too many because they can impact performance and block other processing.

Decrease NUM_IOCLEANERS if Asynchronous Write Percentage (AWP) is 90% or higher; increase it if less than 90%. The AWP can be calculated as follows:

```
AWP = (("Physical async. data writes" + "Async. index writes") * 100) / ("Data
writes" + "Index writes")
```

Two data views can be used to monitor and verify your connection and agent settings in the database and database manager configuration. See “Monitoring the connections and agents of your CM library server” on page 491 for a detailed description and usage information.

- Instance: Agents

  Counters:

  - Maximum agent overflows
  - Stolen agents
  - Agents created (%)

- Database: Connections + Agents

  Counters:

  - Max. concurrent connections
  - Max. coordinating agents
  - Appl. associated agents
  - Appl. connected to DB

The *Log Space* data view can be used to fine-tune your LOGPRIMARY, LOGSECOND, and LOGFILSZ database configuration parameter values.

- Log Space

  Counters:

  - Secondary currently allocated (bytes)
  - Secondary space used (bytes)
  - Log space used (%)
  - Log space used
  - Max. space used for logs

LOGPRIMARY specifies the number of primary log files to be pre-allocated and LOGSECOND specifies the number of secondary log files that are allocated on an as-needed basis. LOGFILSIZ defines the size of each log file.

If there is a high number of “Secondary currently allocated”, you should increase LOGFILSIZ or LOGPRIMARY, but make sure that LOGPRIMARY + LOGSECOND does not exceed 256. You can also use “Max. space used for

logs" to help determine your dependency on log file space (primary + secondary logs).

Log file size affects disaster recovery configurations where log shipping is used. A large log file will have better performance, but potentially increases the degree of lost transactions. When the primary system goes down, the last log file and its transactions might never be sent to the secondary system, because the file was not closed before failure. The larger the log file, the greater the potential for lost transactions due to the lost log file.

This *Database Files Closed* data view can be used to fine-tune the MAXFILOP database configuration parameter value.

- Database Files Closed

  Counter:

  - DB files closed

MAXFILOP specifies the maximum number of files that can be open for each database agent. If opening a file causes this value to be exceeded, some files that are in use by this agent are closed. Excessive opening and closing will degrade performance. Both SMS tablespaces and DMS tablespace file containers are treated as files. More files are generally used by SMS.

Increase the MAXFILOP setting until the number of "DB files closed" is 0.

This *Instance: Sort Memory* data view can be used to fine-tune the SHEAPTHRES database manager configuration value.

- Instance: Sort Memory

  Counters:

  - Sort private heap high water mark
  - Total heap allocated (pages)
  - Post threshold sorts
  - Piped sorts (%)

SHEAPTHRES specifies the total amount of memory that concurrent private sorts can consume for all databases in the instance. Additional incoming sorts will be given a smaller amount of memory to use. For OLTP, a good starting point is about 20000.

When “Piped sorts %” is a low percentage, performance can typically be improved by increasing the size of SHEAPTHRES. If ""Post threshold sorts” (sorts that requested heaps after SHEAPTHRES had been exceeded) is a high value (that is, double-digit), try increasing the size of SHEAPTHRES. “Total heap allocated” should be less than SHEAPTHRES. If it is not, increase SHEAPTHRES.

Importing and using CM data views

If you are using Performance Expert V2.2.0.1, then in the samples/SystemHealth directory of your Performance Expert Client installation, a set of XML files is provided that contain CM-specific data view definitions. These XML files can be imported into the System Health component of the Performance Expert Client and help you to set up your System Health view quickly if you monitor either a CM system.

On the Performance Expert Client open the System Health component, right-click **Data Groups**, and select **Import**. Navigate to the samples/SystemHealth directory containing the XML files, select an XML file and press **Open**. Some of the data view definitions in these XML files require parameters to be adapted to your environment, for example, buffer pool names. You can change these during importing the data views. After importing, the data group is displayed in your navigation tree in System Health. Repeat these steps for each XML file you want to import.

The CM data view definitions are provided in these XML files:

- CM_DataViews_CPU_Memory_Disk_Storage.xml
- CM_DataViews_LibraryServer_PageIO.xml
- CM_DataViews_Locks.xml
- CM_DataViews_ResourceManager_PageIO.xml
- CM_DataViews_Workload.xml

Most of these CM XML files can be imported for your Library server system and your Resource Manager system.

Each file, after being imported, results in a new data group with the respective data view definitions. Some of the data views are preselected for being displayed in System Overview.

Figure B-5 on page 489 shows a System Health panel containing all CM data views grouped into different groups that have been imported. Many data views have been defined during the import process giving you a complete picture of what is happening on your CM Library Server and CM Resource Manager. The critical or most important data views are displayed on System Overview as well.

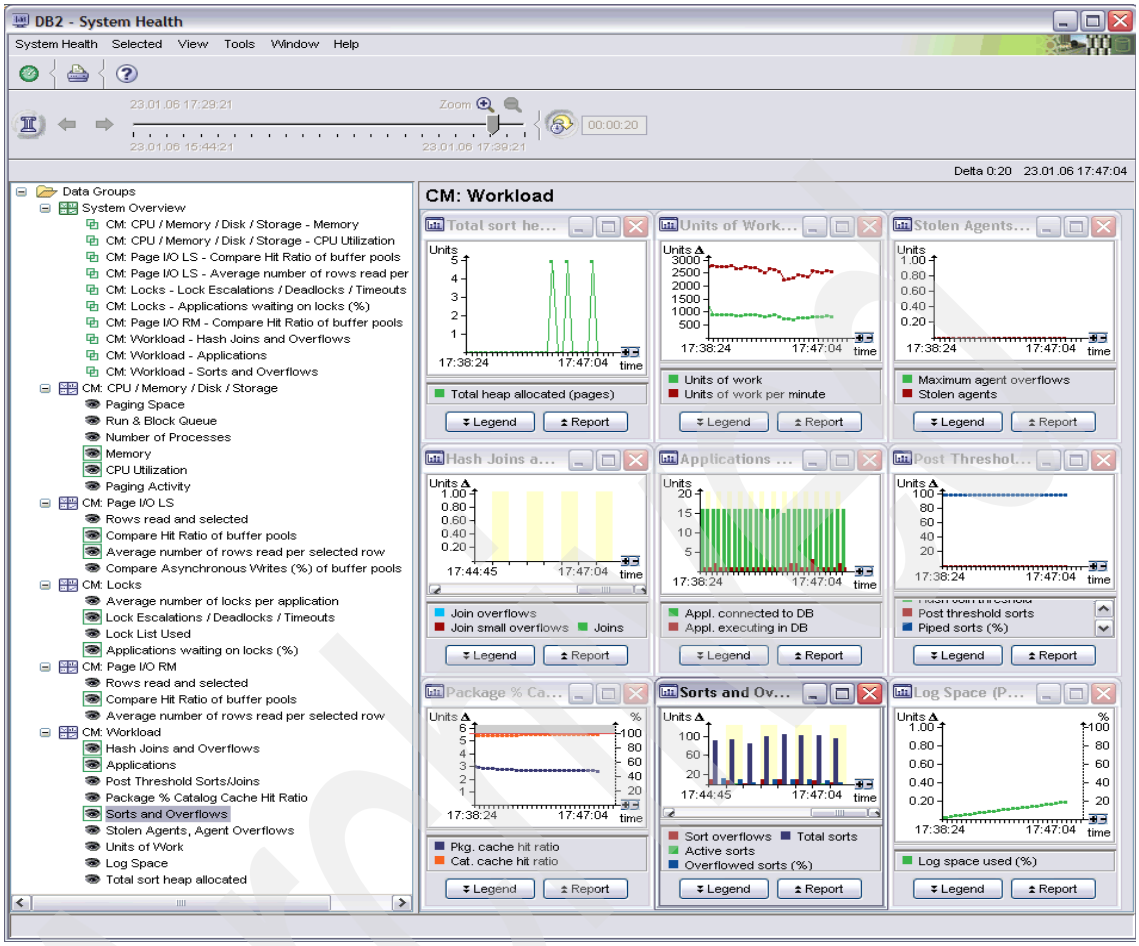


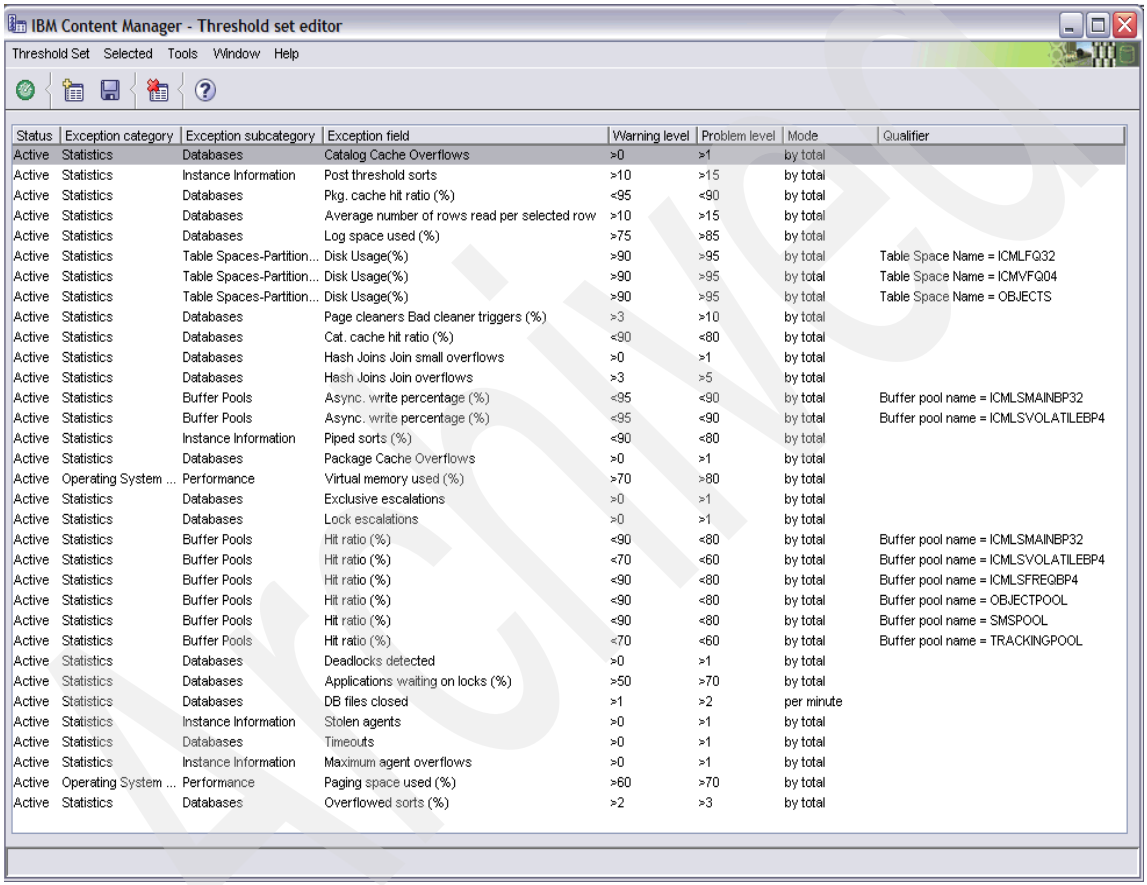
Figure B-5 System Health imported data views

Monitoring the health using exception processing

Performance Expert provides capability of monitoring by exception. The Performance Expert server can periodically compare the warning and problem thresholds that are set on DB2 snapshot values or calculated values out of snapshot values against actual snapshot values. Exceptions are raised on the Performance Expert client if a threshold is exceeded. Thresholds on different snapshot counters can be combined in threshold sets. Performance Expert offers predefined threshold sets that can be applied *as-is* or adapted to your own needs.

From the Tools menu, select **Exception Processing** and create a new threshold set by using the predefined threshold set Statistics OLTP (if you are using Performance Expert V2.2) or IBM Content Manager (if you are using Performance Expert V2.2.0.1). Activate the threshold set for your CM Library Server and CM Resource Managers.

Figure B-6 shows the content of the predefined threshold set 'IBM Content Manager'.



Status	Exception category	Exception subcategory	Exception field	Warning level	Problem level	Mode	Qualifier
Active	Statistics	Databases	Catalog Cache Overflows	>0	>1	by total	
Active	Statistics	Instance Information	Post threshold sorts	>10	>15	by total	
Active	Statistics	Databases	Pkg. cache hit ratio (%)	<95	<90	by total	
Active	Statistics	Databases	Average number of rows read per selected row	>10	>15	by total	
Active	Statistics	Databases	Log space used (%)	>75	>85	by total	
Active	Statistics	Table Spaces-Partition...	Disk Usage(%)	>90	>95	by total	Table Space Name = ICMLFQ32
Active	Statistics	Table Spaces-Partition...	Disk Usage(%)	>90	>95	by total	Table Space Name = ICMVFQ04
Active	Statistics	Table Spaces-Partition...	Disk Usage(%)	>90	>95	by total	Table Space Name = OBJECTS
Active	Statistics	Databases	Page cleaners Bad cleaner triggers (%)	>3	>10	by total	
Active	Statistics	Databases	Cat. cache hit ratio (%)	<90	<80	by total	
Active	Statistics	Databases	Hash Joins Join small overflows	>0	>1	by total	
Active	Statistics	Databases	Hash Joins Join overflows	>3	>5	by total	
Active	Statistics	Buffer Pools	Async. write percentage (%)	<95	<90	by total	Buffer pool name = ICMLSMAINBP32
Active	Statistics	Buffer Pools	Async. write percentage (%)	<95	<90	by total	Buffer pool name = ICMLSVOLATILEBP4
Active	Statistics	Instance Information	Piped sorts (%)	<90	<80	by total	
Active	Statistics	Databases	Package Cache Overflows	>0	>1	by total	
Active	Operating System ...	Performance	Virtual memory used (%)	>70	>80	by total	
Active	Statistics	Databases	Exclusive escalations	>0	>1	by total	
Active	Statistics	Databases	Lock escalations	>0	>1	by total	
Active	Statistics	Buffer Pools	Hit ratio (%)	<90	<80	by total	Buffer pool name = ICMLSMAINBP32
Active	Statistics	Buffer Pools	Hit ratio (%)	<70	<60	by total	Buffer pool name = ICMLSVOLATILEBP4
Active	Statistics	Buffer Pools	Hit ratio (%)	<90	<80	by total	Buffer pool name = ICMLSFREQBP4
Active	Statistics	Buffer Pools	Hit ratio (%)	<90	<80	by total	Buffer pool name = OBJECTPOOL
Active	Statistics	Buffer Pools	Hit ratio (%)	<90	<80	by total	Buffer pool name = SMSPOOL
Active	Statistics	Buffer Pools	Hit ratio (%)	<70	<60	by total	Buffer pool name = TRACKINGPOOL
Active	Statistics	Databases	Deadlocks detected	>0	>1	by total	
Active	Statistics	Databases	Applications waiting on locks (%)	>50	>70	by total	
Active	Statistics	Databases	DB files closed	>1	>2	per minute	
Active	Statistics	Instance Information	Stolen agents	>0	>1	by total	
Active	Statistics	Databases	Timeouts	>0	>1	by total	
Active	Statistics	Instance Information	Maximum agent overflows	>0	>1	by total	
Active	Operating System ...	Performance	Paging space used (%)	>60	>70	by total	
Active	Statistics	Databases	Overflowed sorts (%)	>2	>3	by total	

Figure B-6 IBM Content Manager threshold set

If an exception occurs, it is displayed on your System Overview panel, from which you can drill down into the exception. Additionally, you can specify that an e-mail should be sent in case of an exception or, with Performance Expert V2.2.0.1, a user exit can also be executed that allows you to execute appropriate actions or route the exception to other applications. If an exception occurs, use the

recommendations that are described in “Monitoring the health of the CM DB2 databases” on page 480 to determine how your system could be improved.

Figure B-7 shows the display of an exception on the System Overview panel and a drill down into one exception.

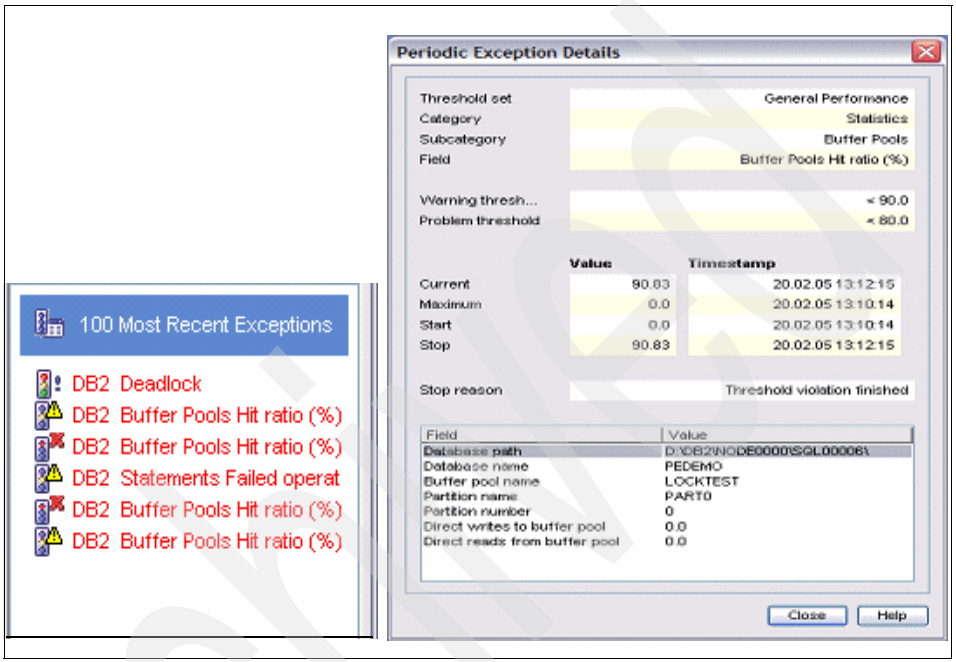


Figure B-7 List of exceptions and exception details

Monitoring the connections and agents of your CM library server

Monitoring your connections and agents on the CM Library Server machine has two aspects:

- Determine whether you should turn on the DB2 connection concentrator and whether connection pooling configurations are optimal
- Determine whether your agent, application, and connection settings are correct in the database and database manager configuration

Determining the need of using the DB2 connection concentrator

Figure B-8 depicts two different CM Library Server environments.

- In one environment, a “fat” client is used to connect to the CM Library Server machine. This client does not have the ability to do connection pooling. If many users are using a “fat” client, then as many connections are kept to the CM Library Server database, which are idle very often. With each connection, a db2fmp process and an agent is associated, which consumes system resources all the time. In this environment, connection pooling on the DB2 side might be useful if too many system resources are allocated.
- In the other environment, a mid-tier server is used between the CM Library Server machine and a group of Web applications that are using eClients. The mid-tier server has the ability to do connection pooling, which must explicitly be configured. This is the recommended approach, because it saves resources on the CM Library Server. Depending on the connection pooling configuration, only a few connections are kept to the CM Library Server database, which are shared among all the Web applications. In an ideal connection pooling configuration, connections are busy most of the time.

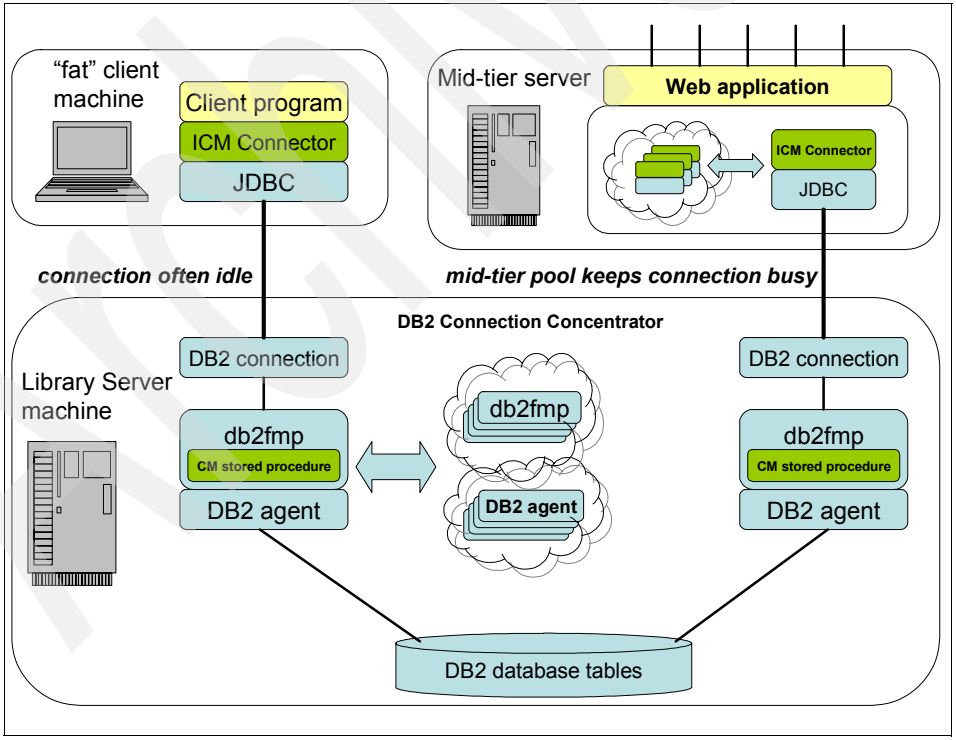


Figure B-8 CM connection pooling environments

You can use Performance Expert to monitor both types environments and to detect whether a connection concentrator should be used for the “fat” client connections or whether the connection pooling configuration on the mid-tier server is optimal.

In Performance Expert, select **Statistics Details** → **Databases.** Select the **CM Library Server** database and double-click it. The Application window is displayed (Figure B-9), which contains useful information about the number of connections and agents in your Library Server database.

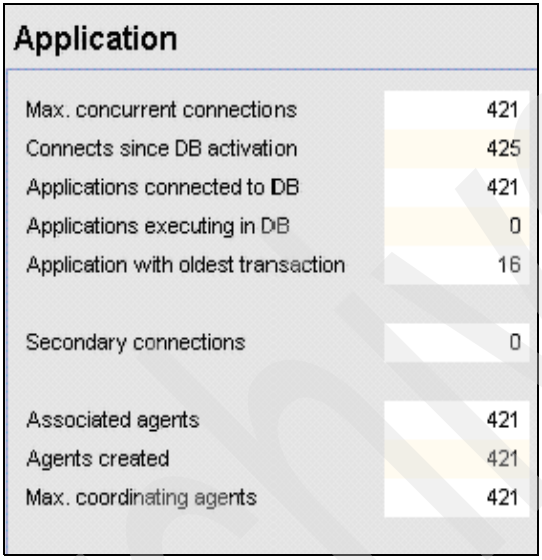


Figure B-9 Connection information shown in PE with connection concentrator

The Application pane shows that 421 applications are currently connected to the CM Library Server database (Applications connected to DB). The high water mark of connections since the database was activated is 421 connections (Max. concurrent connections).

The connection high water mark determines the value that you should use for the MAX_APPLS parameter in the database configuration of the CM Library Server database. If the MAX_APPLS value is currently set to a lower value, increase it. If it is set to a higher value, consider decreasing it. But before doing so, monitor the connection high water mark value for several days or weeks to determine the correct value.

Because in most cases your CM Library Server database is the only database in your DB2 instance, the connection high water mark also determines the setting of the MAX_CONNECTIONS value in the database manager configuration.

The coordinating agents high water mark is also currently 421 (Max. coordinating agents). Because it is the same value as the connection high water mark, it shows that the connection concentrator is not turned on. The number of associated agents (Associated agents) is also 421, so one agent is associated with one connection, and no agent sharing takes place between connections. On the other hand, no application is currently executing in your database (Applications executing in DB), which means that all connections are idle, but consuming resources (agents and db2fmp processes).

Because the CM Library Server uses stored procedures to execute SQL statements, on the operating system level, one stored procedure process named db2fmp is associated with one coordinating agent. So the coordinating agent high water mark determines the maximum number of db2fmp processes running on your CM Library Server system. As a rule of thumb, each db2fmp process consumes 10 MB of memory, so you can calculate the amount of memory that is used on your system by db2fmp processes. If this amount of memory is close to the maximum memory available on your system, consider turning on the connection concentrator.

By using Performance Expert, you can also monitor the db2fmp processes, how much memory and CPU they consume, and also the overall memory consumption if operating system monitoring is enabled. See “Monitoring the operating system” on page 520.

There is some additional information that is displayed in this window:

- If you turn the connection concentrator on, the number of applications executing in your database gives you the approximate ratio that you should use for the connection concentrator. Of course, monitor this value for some time before deciding which ratio to use.
- If you use CM desktop clients (“fat” client) without a mid-tier server, the connection high water mark tells you how many users are using the CM system.
- If you use eClients together with a mid-tier server, the connection high water mark should be much less than the number of users, because the mid-tier server has the ability to do connection pooling. If the connection high water mark equals the number of users, connection pooling is not configured on the mid-tier server, but should be done in order to save system resources.
- If you use an eClient together with a mid-tier server doing connection pooling, the ratio of applications connected to database (Applications connected to DB) and applications executing indicates how efficient your connection

pooling is. If you have many more connections than executing applications, you might consider optimizing it to save resources. Using the connection concentrator makes no sense in this environment because connection pooling in the mid-tier server is the right way to go.

Figure B-10 shows an example where the connection concentrator is turned on with a ratio of 2:1. The connection high water mark is 602, and the coordinating agents high water mark is 310. Only one connection is currently executing in the database, and one agent is associated with it. The agent goes back into the agent pool after the connection has finished the execution, although the application is still connected but idle.

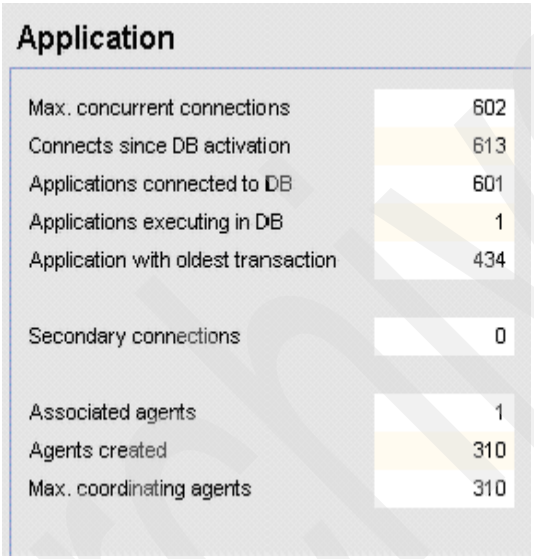


Figure B-10 Connection information shown in PE with connection concentrator

Refer to the *IBM DB2 Content Manager V8.3 Enterprise Edition Performance Tuning Guide* for more information about turning on the connection concentrator. Parameters that are affected are MAX_CONNECTIONS and APP_CTL_HEAP_SZ. *IBM DB2 Content Manager V8.3 Enterprise Edition Performance Tuning Guide* can be found at:

`http://www.ibm.com/support/docview.wss?uid=swg27006452`

Verifying agent settings of the CM library server DB2 instance

In Performance Expert, select **Statistic Details** → **Instance Information** to display information that you can use to verify your agent settings in the database manager configuration. This information is shown in Figure B-11.

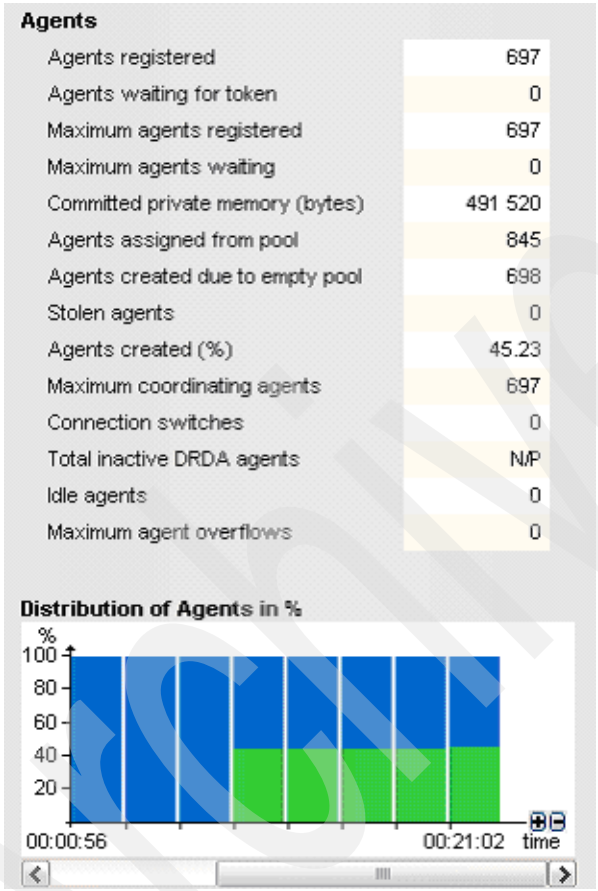


Figure B-11 Agent information in PE

The agent's registered high water mark (Maximum agents registered) helps you to determine the appropriate value for the MAXAGENTS database manager configuration parameter. The coordinating agents high water mark (Maximum coordinating agents) determines the appropriate value for the MAXCAGENTS configuration parameter. In a CM Library Server environment, both values should be the same, as shown in Figure B-11.

If you see agent overflows (Maximum agent overflows), you should increase the value for the MAXAGENT parameter.

The number of agents assigned from the agent pool is 845 (Agents assigned from pool), while the number of agents that had to be created because the pool was empty is 698 (Agents created due to empty pool). The percentage of agents created is approximately 45% (Agents created). Because creating agents causes some overhead, a high number of creations should be avoided by having the agents preallocated in the agent pool (NUM_POOLAGENTS parameter). On the other hand, a low number of agent creations suggests that some agents in the pool are rarely used and are wasting system resources. Look at the number of idle agents (Idle agents). If most of the agents are idle, you should consider decreasing the NUM_POOLAGENTS value.

Also, the usage of the connection concentrator affects the NUM_POOLAGENTS setting. It should be set to a higher value if the connection concentrator is not used. When the connections concentrator is off, NUM_POOLAGENTS specifies the maximum size of the agent pool. When the concentrator is on, it is used as a guideline for how large the agent pool will be when the system workload is low.

NUM_INITAGENTS and NUM_POOLAGENTS should be set to the average number of expected concurrent connections.

Analyzing deadlocks and locking conflicts

Sometimes locking situations or deadlocks occur in the CM Library Server database. When they occur, the CM Library Server issues the following message:

```
Library Server Return Code = 7015, Extended Return Code = -911
```

Return code 911 points to an SQL error message and means that the current transaction has been rolled back because of a deadlock or timeout.

If CM users get this message very often, you should analyze it to determine if the CM application can be improved and whether the Library Server needs to be tuned. Deadlock and timeouts in the CM Library Server can occur for the following reasons:

- The CM application is written using explicit transactions. If this is the case:
- Determine if you can shorten the explicit transactions.
- Determine if you can do some actions outside explicit transactions.
- Void user interactions inside an explicit transaction.
- If your application is already in production, then check whether the recommendations given in the next bullet can serve as a workaround.

- The CM application is written using implicit transactions, but the user is executing long queries. Possible explanations include:
  - The queries are well tuned but just taking longer than the current setting of the LOCKTIMEOUT value in your database configuration. You should determine if you can execute these queries at times where the system workload is low. If this is not an option, you should increase the LOCKTIMEOUT value to get these queries finished.
  - The queries are not well tuned. For example, table scans or sorts occur because an index is missing. See “Identifying long-running SQL statements” on page 512.

But first use Performance Expert to analyze the reason for the deadlock or timeout, and to determine what applications and statements are involved.

Performance Expert offers various possibilities to discover and analyze deadlocks and locking conflicts. To get an overview about the amount of locking and deadlock situations, the System Health graph “Applications waiting on locks (%)” shows how many applications are waiting on locks. The **Databases** → **Locks** window in Statistic Details shows snapshot counters for your database, for example, how many locks held, lock waits, or deadlocks occurred in the database since activation. If you see deadlocks and you have not turned on event exception processing in Performance Expert, you should do that immediately so that you can analyze the details of the next deadlock that happens.

Analyzing deadlocks

If event exception processing is enabled in Performance Expert, then Performance Expert alerts you if a deadlock occurred and let you analyze the details. When event exception processing is enabled, a DB2 deadlock event monitor is started on your CM Library Server database. A deadlock event monitor causes little overhead on your CM Library Server and writes data only if a deadlock occurred. Therefore, you should enable event exception processing.

Enable event exception processing by doing the following steps:

1. On the Performance Expert server machine, in the **`peconfig`** utility, set **Event monitor on** for your CM library server database
2. On the Performance Expert client machine, select **Tools** → **Exception Processing** → **Activation** and enable event exception processing for the CM Library Server. Additionally, you can set up e-mail notification.

If a deadlock occurs, you are alerted on the Performance Expert System Overview window. If you set up e-mail notification, an e-mail will be sent that contains the deadlock details. From the System Overview window, you can drill

down into the deadlock details and see which applications and statements are involved in the deadlock (Figure B-12).

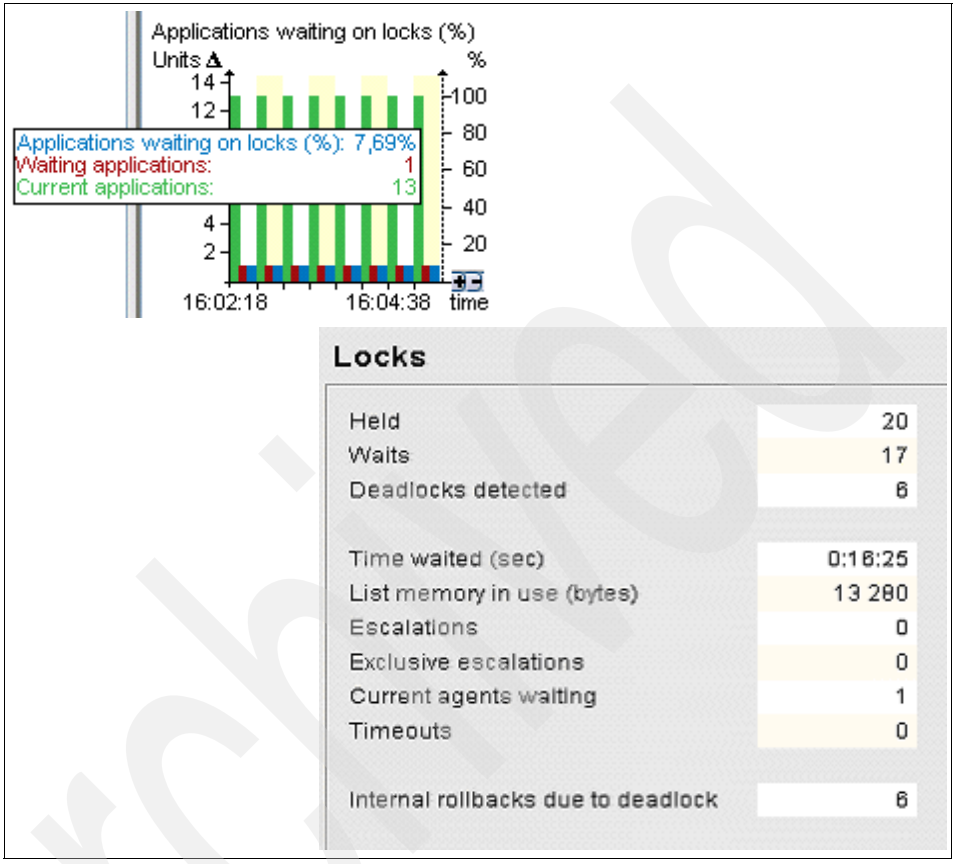


Figure B-12 Locking information in PE

Analyzing locking conflicts

If two or more applications have a locking conflict, the transactions are either rolled back because the locking conflict is a deadlock or they time out (if LOCKTIMEOUT is not set to -1, which is the case for the CM Library Server by default).

You can analyze the details of these applications by using the Performance Expert “Applications in lock conflicts” or “Locking conflicts” functions. Both functions can be called from the System Overview window. The information provided is based on DB2 snapshot information. Therefore, you will only get locking information using these functions if the DB2 snapshot was executed in the time frame where the applications were waiting for the lock to be resolved before there were rolled back. The DB2 snapshot is taken if you use the refresh button in the “Applications in lock conflicts” or “Locking conflicts” functions or if you have configured your system to collect historical data for these functions.

If CM users are complaining about the 911 error and the 911 errors happened because of a lock timeout and not because of a deadlock, either try to catch the next one by pressing refresh or use the history slider in “Applications in lock conflicts” or “Locking conflicts” to determine if Performance Expert already caught them. For debugging purposes, you might consider increasing the LOCKTIMEOUT database configuration parameter value to make sure that Performance Expert catches the lock timeout situations so that you are able to analyze them. When you are finished debugging, decrease LOCKTIMEOUT to its previous value.

“Applications in lock conflicts” starts from the application point of view and shows you holder and waiter applications, as shown in Figure B-13. From there you can drill down into the details of an application down to the SQL statements and the activity of the statements (for example, number or rows read and selected) that are holding the lock and that are waiting for a lock.

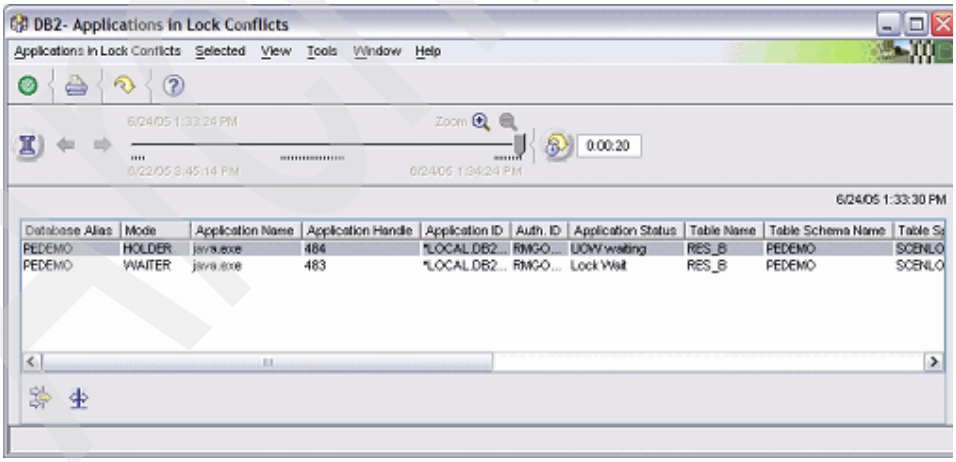


Figure B-13 Applications in Lock Conflicts (holder and waiter)

"Locking conflicts" starts from the resource. It shows you on which tables locking conflicts exist. From there you can drill down to the holder and waiter applications and further drill down to the application details, including the SQL statements.

Detecting the need for running reorg and runstats

The *IBM DB2 Content Manager V8.3 Enterprise Edition Performance Tuning Guide* recommends that you run **runstats**, **rebind**, and **reorgchk** regularly. Performance Expert also provides some information that indicates when it is necessary to run these tools.

For example, if you discover high sorting activity together with sort overflows (Refer to "Monitoring the health using System Health graphs" on page 480) or if you discover a decrease in your stored procedure throughput (Refer to "Monitoring stored procedure call throughput" on page 507), these are indicators that your database statistics are outdated.

Run **runstats** together with **rebind** or **db2rbind** and check whether performance or sort behavior improves afterwards. The *IBM DB2 Content Manager V8.3 Enterprise Edition Performance Tuning Guide* describes in detail how to call **runstats** and **rebind** or **db2rbind**.

From time to time, it might also be necessary to reorganize tables of the CM Library Server and CM Resource Manager. In Performance Expert, under **Statistic Details** → **Tables**, you get an indication that this might be necessary, as you can see in Figure B-14.

If you see many “Overflowed rows” and “Page reorg” for your tables, run **`reorgchk`** and reorganize the tables afterwards if necessary.

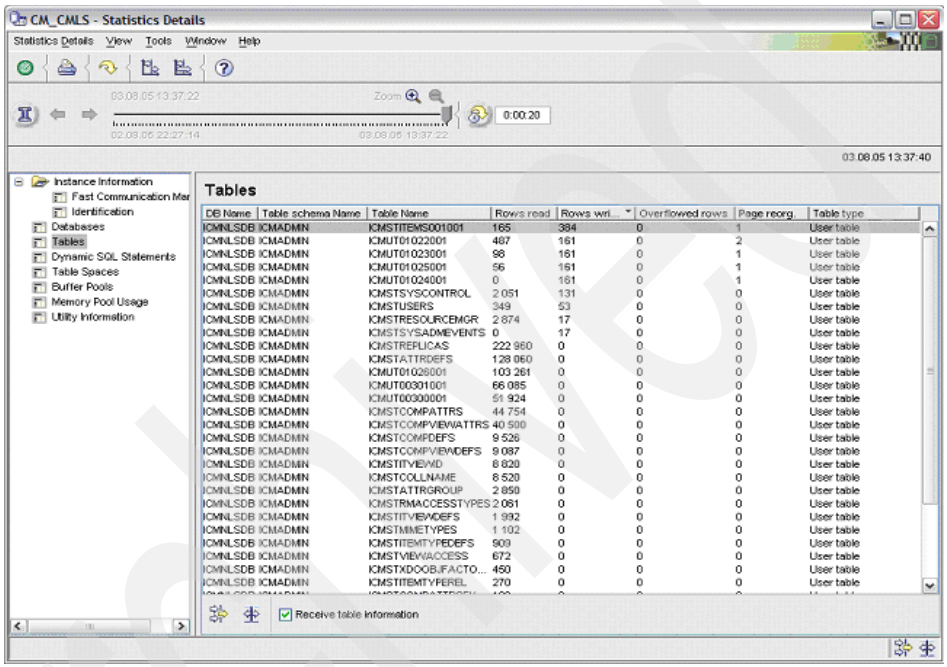


Figure B-14 List of tables with overflowed rows and page reorganization info

Monitoring buffer pool effectiveness

If CM users are complaining about slow response time, this could be the result of reading data from disk instead of from memory (buffer pool).

During the installation of the CM Library Server and CM Resource Manager, buffer pools are created with sizes that are based on 2 GB RAM being available on the machine.

Figure B-15 on page 503 shows which buffer pool is used for which tables spaces and includes a description about the usage of the tablespace.

Library Server Default Tablespaces		
Bufferpool	**Tablespaces**	**Tablespaces Description**
ICMLSMAINBP32	ICMLFQ32	Holds all the large, persistent, "frequently used" tables – most notably, all the item type tables. When a new item type is defined, the system administration client defines the new tables in this tablespace. If you customize the DDL, this tablespace should be defined across multiple containers for performance and scalability.
	ICMLNF32	Holds the large but less-frequently used tables, such as event logs, versions, and replicas. If you don't use any of these features, this tablespace won't need much space.
ICMLSVOLATILEBP4	ICMVFQ04	Holds the "volatile" tables, those whose size is usually small but which can vary widely, and for which both updates and deletes are very common. For example, the tables associated with document routing "in progress" items, checked out items, etc. Putting this on a separate physical disk should help performance.
ICMLSFREQBP4	ICMSFQ04	Holds all the small "frequently used" but seldom updated tables, such as system information, user definitions, ACLs, etc, that don't take up much space but that are always needed in memory for good performance.
CMBMAIN4	CMBINV04	Holds the tables used for EIP federated administration
Resource Manager Default Tablespaces		
Bufferpool	**Tablespace**	**Tablespaces Description**
OBJECTPOOL	OBJECTS	Primary tablespace that holds the tables that grow very large, with information about all the objects stored on this Resource Manager.
SMSPOOL	SMS	SMS holds small but frequently read administrative information.
TRACKINGPOOL	TRACKING	TRACKING holds volatile tables related to transactions.
PARTSPOOL	PARTS	PARTS holds tables used by the validation utility, which is infrequently used but can create a very large volume of data.
BLOBPOOL	BLOBS	BLOBS is reserved for future use.

Figure B-15 Buffer pools and tablespaces created in CM databases

For the CM Library Server, the ICMLSMAINBP32 buffer pools is the buffer pool that holds most of the active user data and therefore should be monitored frequently (especially in peak times) to determine if the hit ratio is always good. Additionally, after adding new item types, you should monitor the buffer pool to see whether the buffer pool can still handle the additional new data without decreasing hit ratio. If you consider increasing the buffer pools, then this is the buffer pool that could best exploit the extra space.

The ICMLSVOLATILEBP4 buffer pool holds tables, which sizes are usually small, but which can very widely. Monitor this buffer pool frequently as well, although receiving always good hit ratios for this buffer pool is difficult since the content is so volatile.

The ICMLSFREQBP4 buffer pool is used for frequently used but seldom-updated tables. When this buffer pool receives good hit ratios (which means that almost all frequently used data is available in the buffer pool), it probably does not need much attention any more.

The CMBMAIN4 buffer pool is not used much, so you might see low hit ratios, which should not be a problem in this case.

For the CM Resource Manager, the OBJECTPOOL and TRACKINGPOOL buffer pools are the buffer pools that should be monitored frequently (especially in peak times) to determine if the ratios are always good.

For the SMSPOOL buffer pool, the same is true as for the ICMLSFREQBP4 buffer pool of the CM Library Server. When this buffer pool receives good hit ratios, it does not need much attention any more.

The PARTSPOOL buffer pool might receive low hit ratios because it is infrequently used. If it starts being used heavily, this buffer pool might be too small to hold all the data.

If you see buffer pool hit ratios consistently lower than 80%, this is an indication that not all required data is available in the buffer pool when needed.

Possible reasons are:

- Table scans are done when queries are executed by the CM user, which means that much more data is read than is necessary. A missing index could be the reason for a table scan.
- The queries are well tuned, but the buffer pool is simply too small. In this case, if your CM Library Server or CM Resource Manager has more than 2 GB of RAM, consider increasing its buffer pool sizes.

First, you should analyze your buffer pool behavior before you make any changes. With Performance Expert, you have various options for monitoring your buffer pools. (Chapter 5, “Features and functions - long-term monitoring” on page 239 of this redbook describes them in detail.)

One option is to use Buffer Pool Analyzer (BPA). BPA analyzes historical buffer pool, tablespace, and table data that is already collected in Performance Warehouse and creates different types of reports based on a user-specified time frame.

One report is a hierarchical view that shows you which table resides in which tablespace and which tablespace uses which buffer pool. Another report is the snapshot view that allows you to graphically view the evolution of a snapshot counter in the time frame you specified. If you put both reports together, then start with the buffer pool with the low hit ratio, look for the tablespaces that are using this buffer pool and look at their hit ratios. Pick the one that has the same low hit ratio and look at the read/write activity of the tables residing in this tablespace. Although the hit ratios shown in Figure B-16 on page 505 are very good, can be used as an example for the method described here.

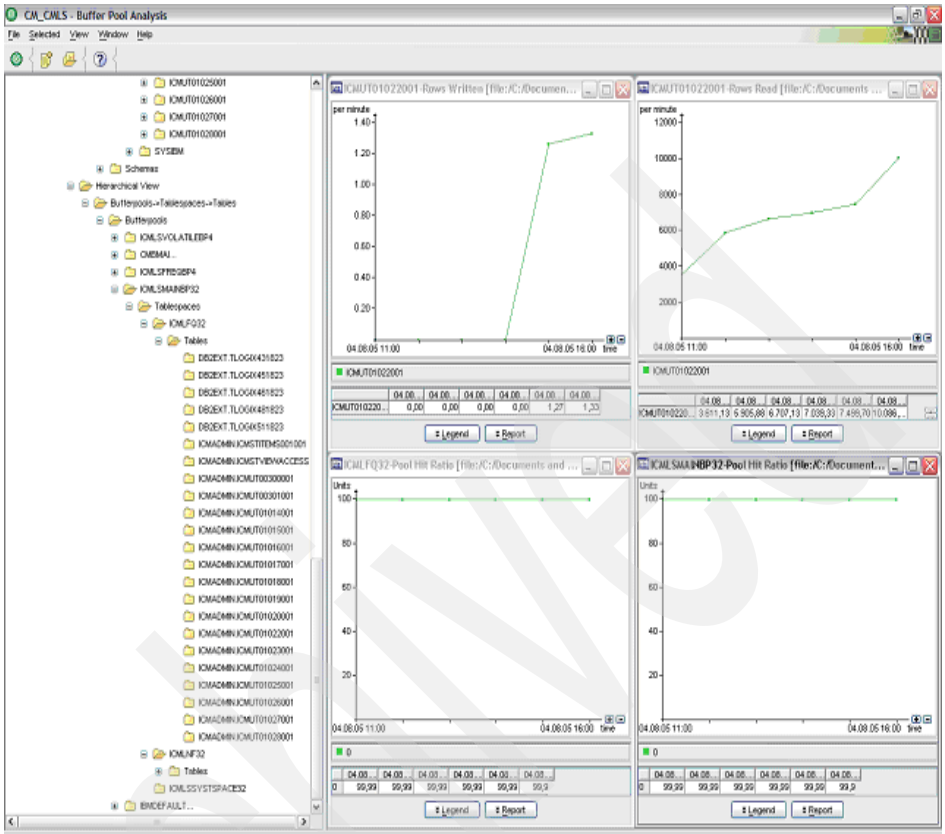


Figure B-16 Buffer Pool Analyzer report of LS database

If you detect tables that have a heavy read or write activity, analyze them further to determine if these reads or writes are a result of a table scan or sort. Refer to “Detecting heavy hitter tables” on page 514 for more information. If this is the case, you should first improve your queries before considering increasing the buffer pools.

Discovering peak and normal workload times and monitoring the performance in these times

The Performance Warehouse is the component of Performance Expert that saves historical monitoring data over a long period of time (long-term history) and provides tools to analyze the data, including reports, queries, and rules-of-thumb. Analyzing the data in the Performance Warehouse helps you to identify potential performance problems or trends and to determine the times at which you have heavy workload on your system.

The following data is stored in Performance Warehouse:

- Database activity based on snapshot data (automatically stored and aggregated)
- Buffer pool activity snapshot data together with tablespace and table snapshot data (automatically stored and aggregated)
- SQL Activity based on the statement event monitor (collected and stored only if user starts an SQL Activity Trace)
- DB2 configuration data
- OS system data (optional)

You can analyze the data that is stored in Performance Warehouse by using reports, queries, or rules of thumb. Chapter 5, “Features and functions - long-term monitoring” on page 239 describes them all in detail. Using the SQL Activity trace is also described in “Monitoring stored procedure call throughput” on page 507.

For your CM Library Server, the Database Activity Report is useful for helping you with the following tasks:

1. Monitoring connections and agents of your CM Library Server. A database activity report helps you to determine at what times the most connections were open to your Library Server and how busy they were, which ultimately tells you when your peak and normal times are. By using this data, you can determine, for example, whether the connection concentrator is needed and to which values the connection and agent parameters in your database and database manager configuration must be set.
2. When you have determined your peak times from the database activity report, then, for these times, look at performance counters that indicate whether your system is running healthy even in peak times. If not, you might need to adjust database and database manager configuration values. The system health performance counters are described in “Monitoring the health using System Health graphs” on page 480.
   - Some sample values you should look at for your peak times:
     - Rows read
     - Average number of rows read per selected row
     - Total sorts and sort overflows and the percentage of them
     - Buffer pool hit ratios
     - Deadlocks, Lock waits, Lock timeouts, and Lock escalations
     - Maximum number of coordinating agents

– If you discover low buffer pool ratios during peak times or normal times, run the Buffer Pool Activity Report or use Buffer Pool Analyzer to get further insight.

3. Determining whether your workload on your CM system is stable or whether, for example, the number of connections increases over time. If this is the case, you might also need to adjust configuration parameters from time to time and plan for additional capacity.

Figure B-17 shows you the part of the Database Activity report in which “Applications connected currently” is reported over time. Use this report to determine your peak times together with “Applications executing in the database currently”.

Partition ID	Snapshot Time	Connects Since Database Activation	Maximum Number of Concurrent Connections	Database Files Closed	Maximum Database Heap Allocated	Binds/Precompiles Attempted	Applications Connected Currently	Applications Executing in the Database Currently	Status of Database
member_id	interval_to	total_cons	connections_top	files_closed	db_heap_top	binds_precompiles	appls_cur_cons	appls_in_db2	db_status
0	08:00:00.0	216	10	0	1053169	0	8	0	0
0	09:00:00.0	223	10	0	1053169	0	8	0	0
0	10:00:00.0	223	10	0	1053169	0	8	0	0
0	11:00:00.0	117	10	0	1053169	0	6	0	0
0	12:00:00.0	195	10	0	1053169	0	7	0	0
0	13:00:00.0	217	10	0	1069553	0	8	0	0
0	14:00:00.0	231	13	0	1069553	0	10	0	0
0	15:00:00.0	235	13	0	1069553	0	10	0	0
0	16:00:00.0	218	13	0	1069553	0	10	0	0
0	17:00:00.0	217	13	0	1069553	0	11	0	0

Figure B-17 Part of a PE Database Activity Report

Monitoring stored procedure call throughput

The CM Library Server uses stored procedures to issue any request of the CM user to the CM Library Server database. Monitoring the stored procedure throughput over time helps you to:

- Monitor the overall workload mix, which is useful information for capacity planning and application tuning issues.
- Detect degraded performance before it becomes noticeable by end users.
- Detect trends in terms of changing throughput times or changing workload mix.

If you detect degraded performance over time in terms of a stored procedure throughput decrease, then first run **runstats** and **rbind** to make sure that the database statistics are up to date. You might see a throughput improvement right after running these utilities. Additionally, check the health of your DB2 system, as described in “Monitoring the health of the CM DB2 databases” on page 480 and do some tuning if required.

For monitoring the stored procedure call throughput with Performance Expert, you can either list the dynamic SQL statements in the statement cache online and step back in the short-term history or you can use Performance Warehouse functionalities to examine a long-term history.

Using Dynamic SQL Statement information

In Performance Expert, select **Statistic Details → Dynamic SQL Statements** and check the **Receive statement cache information** check box. All the statements that are available in the dynamic statement cache of the CM Library Server database are displayed together with performance metrices. You can use the filter function to display only stored procedure calls, as shown in Figure B-18 on page 509. If you sort the stored procedure calls by average execution time, you can discover the “heavy hitter” calls. By using the history slider and stepping back in history, you can discover whether this average execution time changed over time.

Note that only the statements that are currently in the cache are displayed and the average execution time is calculated since database activation. If you have peaks where the execution time increased heavily, you will not see it much in the average values. Note also that history information is available only for the time frame that you have configured in the properties of the monitored system. For discovering peaks and for a long-term analysis, use the Performance Warehouse.

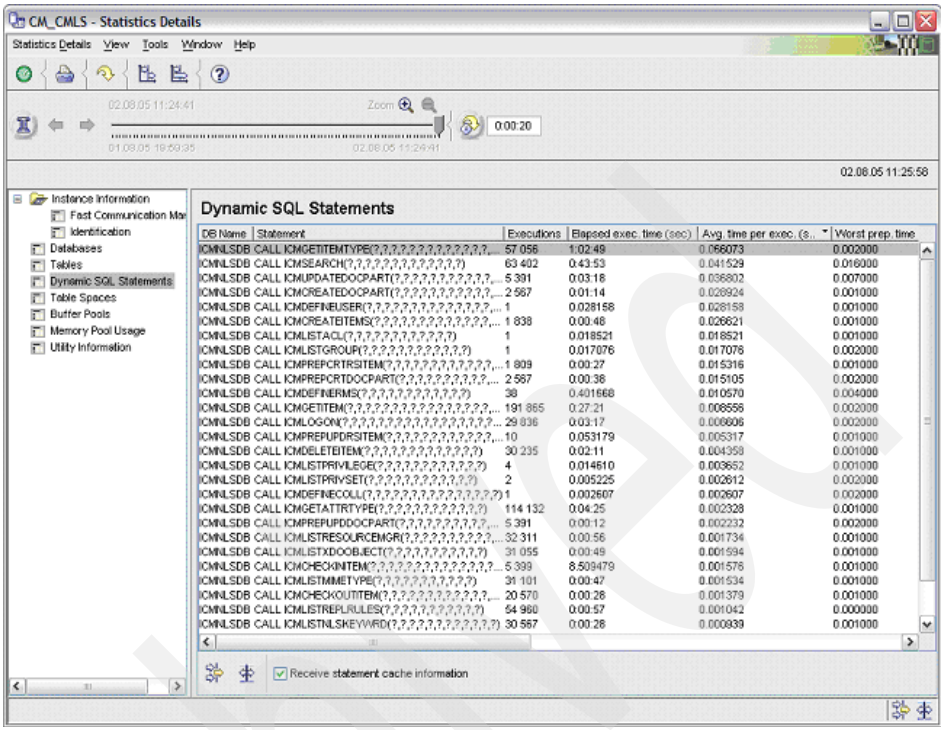


Figure B-18 List of dynamic SQL statements in package cache of LS database

Using the Performance Warehouse

You should frequently create SQL Activity reports to get a baseline for your stored procedure throughput for your workloads. You can do this from Performance Warehouse. Create you own process group and copy, from the public process group either the SQL Activity Summary Report or the SQL Activity Trace Report. Edit the three steps and specify, in the CRD step, the database name and elapsed time. In the Load step, make sure that you select **Retrieve the text of the static SQL statements**. Then execute the process and check in Process Executions when it is finished. When it has finished, open it and display the report.

Using the SQL Activity Summary report

The first part of the SQL Activity Summary Report is a summary that shows the statements that were executed together with the total execution time, and number of executions and other summary information. This data can be used to calculate the average execution time. Use the hyperlinks to drill down to the statement operations. Metrices for each SQL statement that were executed from the stored procedure are available in this report. The metrices are not ordered by execution time stamp, but are instead ordered by execution time. To get the statements ordered by execution time stamp in order to know which stored procedure executes which statements, use the SQL Activity Trace Report.

Note that you can also create a SQL Activity Summary report from the Application Summary or Application Details windows for a single application only. Doing so might limit the information in the SQL Activity report. Consider using this method if you want to run the SQL Activity trace for debugging purposes or if you know which applications currently have a performance or throughput problem.

Using the SQL Activity Trace report

The SQL Activity Trace Report contains a list of all the SQL statements that were executed, ordered by execution time stamp. It shows you which SQL statements are executed by which stored procedure, and in the statement operations, you see the execution time of each statement together with other performance metrices. If some stored procedures take longer to execute than expected, you can figure out which SQL statements executed in the stored procedure are responsible for that. If the statements are SQL queries, a next step might be to run Explain for these queries or to run the DB2 Design advisor to detect additional indices that must be added.

Note that the SQL Activity Trace contains the statements for all applications running in the database. The list of SQL statements is not ordered by application ID; it is ordered by execution time stamp. You can use filters before starting the SQL Activity Trace to include only certain applications into the trace.

Using Performance Warehouse queries

If you have collected SQL Activity information over time by creating the reports mentioned in the previous two sections, you can use the following query from the Performance Warehouse to analyze the throughput over time.

To execute one of the following Performance Warehouse queries, create a new Query Group, right-click **Queries**, and select **Create**. Copy and paste one of the following queries and execute it.

The query in Example B-1 lists the execution times of all stored procedures ordered by agent ID, execution time, and start time stamp. If you are not interested in which application called the stored procedure, you can simply delete this information from the query. Also, enhance the WHERE clause if you are interested only in specific stored procedures.

Example: B-1 Query listing execution times of all stored procedures

```
SELECT
T1.AGENT_ID, T1.START_TIME, T1.STOP_TIME, TIMESTAMPDIFF( 1, CHAR(T1.STOP_TIME -
T1.START_TIME )) AS TOTAL_TIME_IN_MS, T2.STMT_TEXT, T1.STMT_TEXT_ID, T1.APPL_ID
FROM
PWH.EVM_STMT_OPERATIONS AS T1 LEFT OUTER JOIN PWH.EVM_STMT_TEXTS AS T2  ON
T1.STMT_TEXT_ID = T2.STMT_TEXT_ID AND T1.LL_ID = T2.LL_ID
WHERE
T2.STMT_TEXT LIKE 'CALL %'
ORDER BY
T1.AGENT_ID, TOTAL_TIME_IN_MS DESC, T1.START_TIME
```

The query in Example B-2 lists the average execution times of all stored procedures over time grouped by load log ID (LL_ID), application ID, and statement text ID. A load log ID is assigned each time you load SQL Activity data into Performance Warehouse by executing an SQL Activity Report process.

To get the statement text, use the predefined query GetSqlStatement and pass the load log ID and the statement text ID.

If you are not interested in which application called the stored procedure, you can simply delete this information from the query. Also, enhance the WHERE clause if you are interested only in specific stored procedures.

Example: B-2 query lists the average execution time

```
SELECT
AVG(TIMESTAMPDIFF(1, CHAR(T1.STOP_TIME - T1.START_TIME))) , T1.LL_ID,
T1.STMT_TEXT_ID, T1.APPL_ID
FROM
PWH.EVM_STMT_OPERATIONS AS T1,  PWH.EVM_STMT_TEXTS AS T2
WHERE
T1.LL_ID = T2.LL_ID   AND T1.STMT_TEXT_ID = T2.STMT_TEXT_ID  AND T2.STMT_TEXT
LIKE 'CALL %'
GROUP BY
T1.LL_ID, T1.APPL_ID, T1.STMT_TEXT_ID
```

Identifying long-running SQL statements

If a CM client user experiences bad response times when they issue queries or perform other actions that return results, there are several possible reasons:

- Your overall query throughput is decreased, which might be a good indication that you should check the overall health of your system and run **runstats** and **rbind**.
- You have previously created new item types, attributes, and inserted documents, but the proper indices on the tables are not set yet. Another indication of this problem is when “Average rows read per selected row” on your System Health data view is high.
- You are doing inserts or updates and the disk on which the data is physically written has a bottleneck.

To get more detailed information about the cause of bad response times, start a SQL Activity Trace on your CM Library Server. Refer to “Monitoring stored procedure call throughput” on page 507 for information about starting the SQL Activity Trace. By using the SQL Activity Summary Report, you can identify which SQL statements are mostly executed and consume the most execution time.

If these statements are SELECT statements, look at the following performance counters:

- Rows_read
- Fetch_count
- Rows_written

You should consider:

- If the ratio Rows_read/fetch_count is bad (<5 is optimal), this could be an indication that a table scan has been done. Explain the statement to see whether this is the case. Also, determine whether an index must be added on the tables used in the statement (for example, by using the DB2 Design Advisor).
- If rows_read is high, this could also indicate that a sort has been done. Run Explain to determine if an index would help. If a sort has been done and the SQL Activity Report also shows values for rows_written, then the sort was probably done on disk and not in memory, which causes performance degradation. Analyze the SORTHEAP and SHEAPTHRES values of your database and database manager configuration. The System Health graphs in “Monitoring the health of the CM DB2 databases” on page 480 can help you here.
- If the fetch_count is really big (for example, 1 000 000), a CM user is possible issuing SELECT * queries. These queries should be rewritten. If this is not the case, then contact your IBM CM representative to determine if the internal CM query can be improved.

If these statements are INSERT/UPDATE/DELETE statements, check your tablespaces and disks to see if you have any bottlenecks there. “Monitoring tablespaces and file systems” on page 516 provides more information about how to do that task. “Detecting heavy hitter tables” on page 514 elaborates on how you discover which table resides in which tablespace.

Detecting heavy hitter tables

Performance Expert helps you detect which tables in your databases are heavily used. You can do this as a performance maintenance task or you can do it as a troubleshooting task.

In **Statistics Details** → **Tables**, the active tables are listed when you click the **Receive table information** button. Figure B-19 shows a list of CM Library Server tables. The number of rows read and rows written determines which tables are heavily used.

If you are doing troubleshooting (for example, because response times are bad), you should also run SQL Activity Traces to analyze the statements that are using these tables. See “Identifying long-running SQL statements” on page 512.

If you are doing performance maintenance, you can do some analysis based on this information.

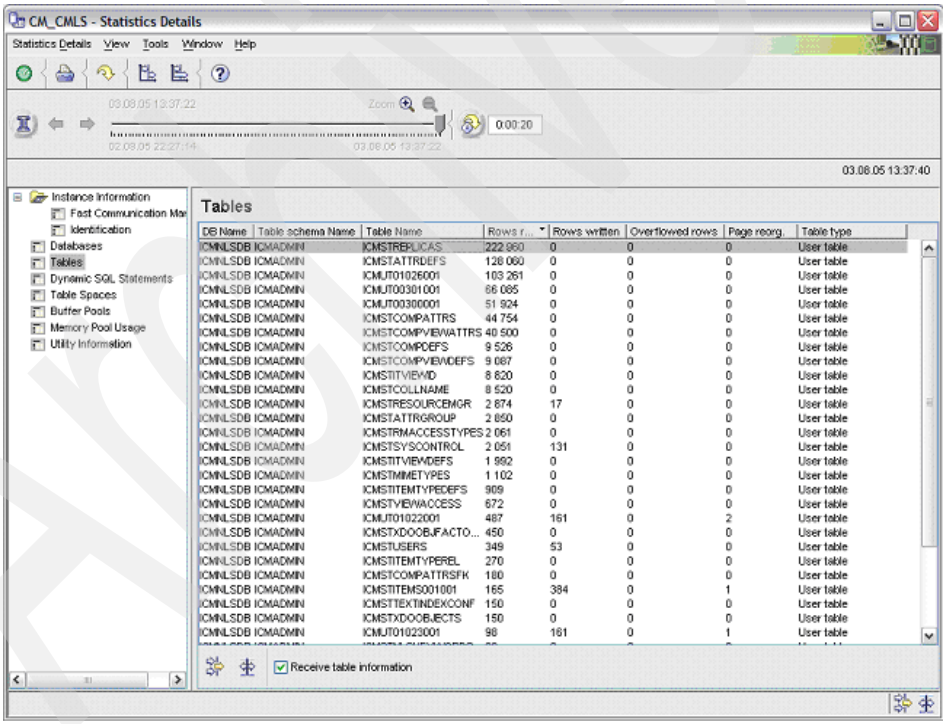


Figure B-19 List of active tables in LS database

A ICMUT* table holds the data for one CM item type. If a CM user introduces a new item type, a new table is created. Also, a user can add new attributes to an

item type, which means that new columns are added to the table. If you see a lot of rows_read for an ICMUT* table, then this table is used heavily in queries and it could be the case that some indices are missing on the columns of the table because it represents a new or changed item type. To assess this, determine the item type of this table, determine which attributes are used in queries by the CM users, and add an index if appropriate.

The ICMSTITEMS table contains all item types of all item type tables with attributes. It is a global table. If this table is heavily used, the CM user queries are most likely not item-specific; therefore, the global item table is used. If possible, the CM user queries should be improved by executing them as item-specific.

If you see a high number of rows_written on a table, in most cases this table is used for insert/update operations. You might want to monitor how your tablespaces of these tables are performing after you have determined in which tablespace this table resides. Monitoring tablespaces is described in “Monitoring tablespaces and file systems” on page 516.

Determining the item type of a ICMUT* table

To determine which table holds which item type, do the following steps:

1. Connect to the CM Library Server database and issue the following SELECT statement:

   ```
   SELECT itemtypeid from ICMADMIN.ICMSTCOMPDEFS where componenttypeid=1025
   ```

2. Issue the following SELECT statement to determine the item type name:

   ```
   SELECT keywordname, keyworddescription from ICMADMIN.ICMSTNLSKEYWORDS where
   keywordcode=<itemtypeid> and keywordclass=2
   ```

The name of an ICMUT* table contains the component type ID, which is the four characters that appear before the last three characters (which are always 001).

For example, the component type ID of table ICMUT01025001 is 1025.

Determining which table resides in which tablespace

Currently, only the Performance Warehouse component of Performance Expert provides information about which table resides in which table. But you can write a simple Performance Warehouse query to get the information. In Performance Warehouse, create a new query and copy/paste the following query text. Then execute the query:

```
SELECT DISTINCT T1.DB_NAME, T1.TABLESPACE_NAME,T2.TABLE_NAME FROM
PWH.TABLESPACE T1, PWH.TABLE T2 WHERE
T1.TABLESPACE_ID=T2.TABLESPACE_ID AND
T1.DB_NAME=T2.DB_NAME AND
T1.MEMBER_ID=T2.MEMBER_ID AND
T1.INTERVAL_TO=T2.INTERVAL_TO AND
T1.INTERVAL_FROM=T2.INTERVAL_FROM
ORDER BY T1.DB_NAME
```

Monitoring tablespaces and file systems

Monitoring your tablespaces, tablespace containers, and file systems where the tablespace containers are located is important for capacity planning and for detecting disk contention.

You can use Performance Expert to monitor the disk or file system usage of SMS and DMS tablespaces and the used-space percentage of DMS tablespace containers.

In general, you should keep an eye on the used space of your tablespace, file system, and disks to detect when a file system must be enlarged or a new container must be added. You should especially pay attention to those tablespaces that contain tables with a lot of write activity. Refer to “Detecting heavy hitter tables” on page 514 for information about detecting these tables. Additionally, look at Figure B-16 on page 505, which describes the usage of the different tablespaces, for example, which tablespaces have a lot of write activity (ICMLFQ32, ICMVFQ04, and OBJECTS).

In Performance Expert, select **Statistic Details** → **Tablespaces** to display a list of the active tablespaces in your database. Figure B-20 shows the tablespaces of a CM Library Server database. By default, they are all SMS tablespaces, and therefore the Used Space %' is 100%.

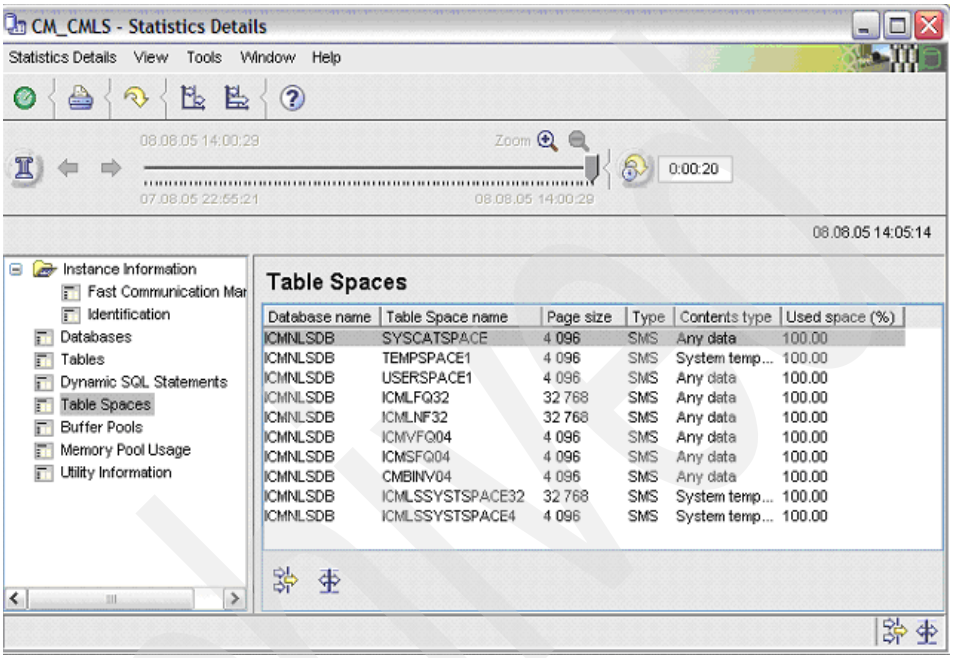


Figure B-20 List of active tablespaces in LS database

For SMS tablespaces, it is important to monitor the free or used space on the file system that the container resides on to know early enough when it must be enlarged. If you have operating system monitoring enabled for Performance Expert, you can set a threshold on the Disk Usage% counter of your file system to be alerted when your file systems are getting full.

To look at the Disk Usage% counter, drill down into a tablespace, select the **Containers** pane, and drill down into the container. If you have operating system monitoring enabled for Performance Expert, you get operating system information for your tablespace, including the disk usage and the free space, as shown in Figure B-21.

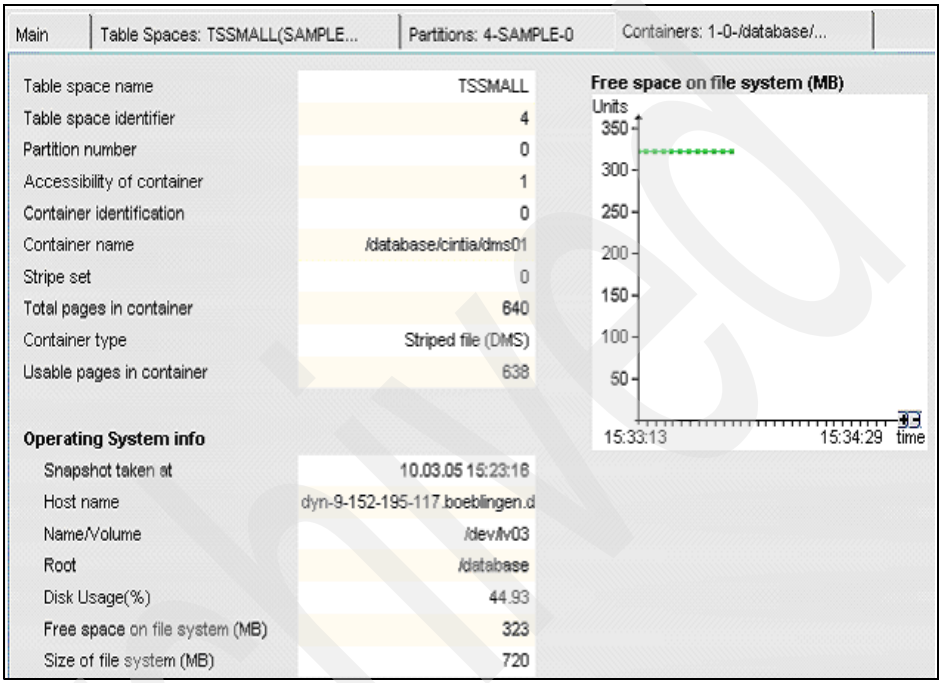


Figure B-21 Container details of tablespace

If you have changed the data model from SMS to DMS tablespaces, you can use Performance Expert to monitor the used pages within a DMS container over time in order to determine when the container must be enlarged or when a new container must be added, as shown in Figure B-22 on page 519.

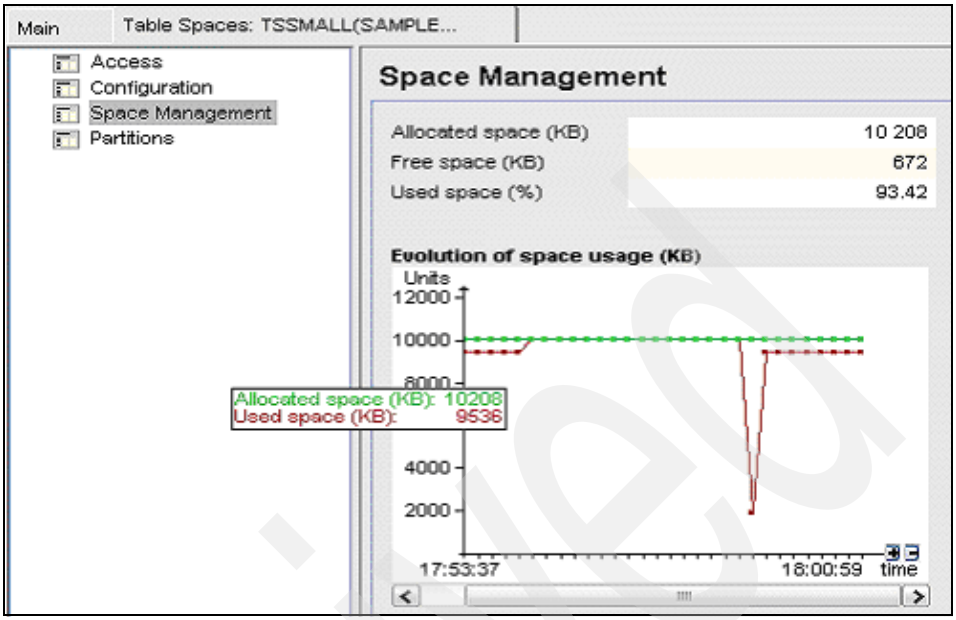


Figure B-22 DMS tablespace used and free space information

There are several reasons why you might decide to use DMS tablespaces instead of SMS tablespaces.

For example, if CM users are experiencing slow response times in their query and update/insert operations, and the analysis of the SQL statements and the tables that they are using has shown that the tables are all residing in the same tablespace, you have disk contention, which can result in I/O wait times. Take a look especially at tablespace ICMLFQ32 for disk contention. The recommendation in Figure B-16 on page 505 is to define this tablespace across multiple containers. For maximum performance, the containers should reside on multiple disks. Although you can still use SMS tablespaces with multiple containers, DMS tablespaces give you more flexibility to add new containers if your containers are getting full.

Performance Expert can help to determine which table resides in which tablespace and how full the tablespace container and their file systems are. Additionally, if you use Performance Expert V2.2.0.1 and you have operating system monitoring enabled, you can also monitor I/O activity and disk utilization on your physical disks to determine whether there is an I/O bottleneck and whether it would make sense to spread the tablespaces over more containers and disks. The CM Performance Tuning provides a list of operating system tools that can be used to accomplish this.

Monitoring the operating system

You can use Performance Expert to monitor the operating system provided that the database that is monitored resides either on the AIX or Solaris platform, has Performance Expert V2.2.0.1 also on Linux, and that a CIM object manager is running on these platforms.

The operating system information that is provided by Performance Expert in online monitor panels, System Health, Exception Processing, and Performance Warehouse include:

- Configured and used memory information
- File system information
- Process information
- CPU Utilization information
- Disk and disk utilization information (Performance Expert V2.2.0.1)

Especially for the task shown in “Determining the need of using the DB2 connection” on page 492, it is helpful to pay attention to the operating system as well. Display the processes that are running on your CM Library Server and look for the CPU and memory usage of the db2fmp processes. Additionally, set exception thresholds on the overall memory usage of your operating system by using the periodic exception processing component of Performance Expert. Then you are alerted either on the Performance Expert Client or via an e-mail when your CM Library Server is going to run out of memory.

As mentioned in “Monitoring tablespaces and file systems” on page 516, monitoring the file systems together with disk utilization and I/O activity is important to detect disk contention and tablespace full conditions.

The following figures show a few Performance Export windows that display operating system information.

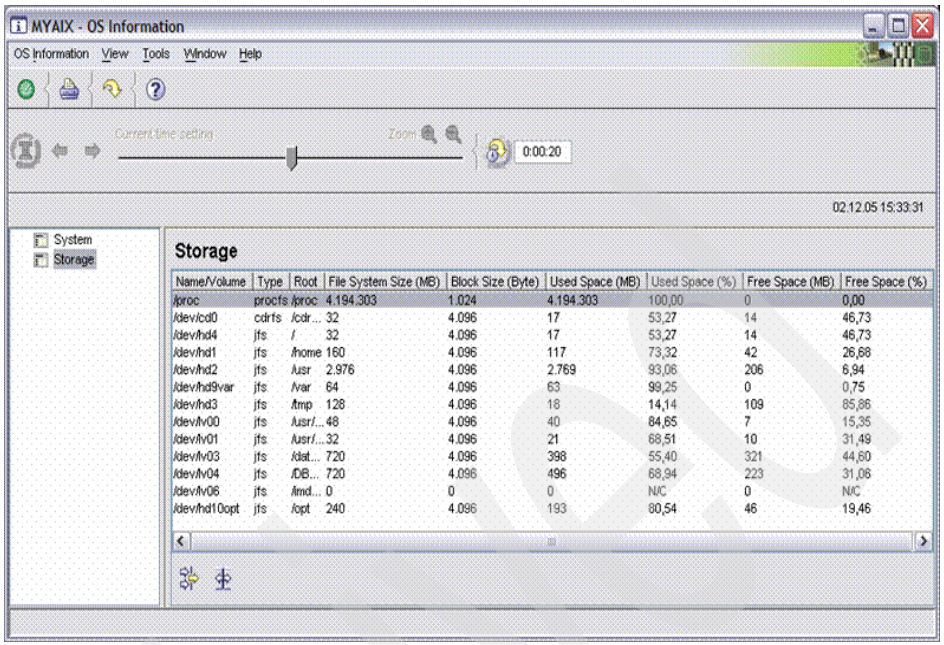


Figure 6-82 File system information

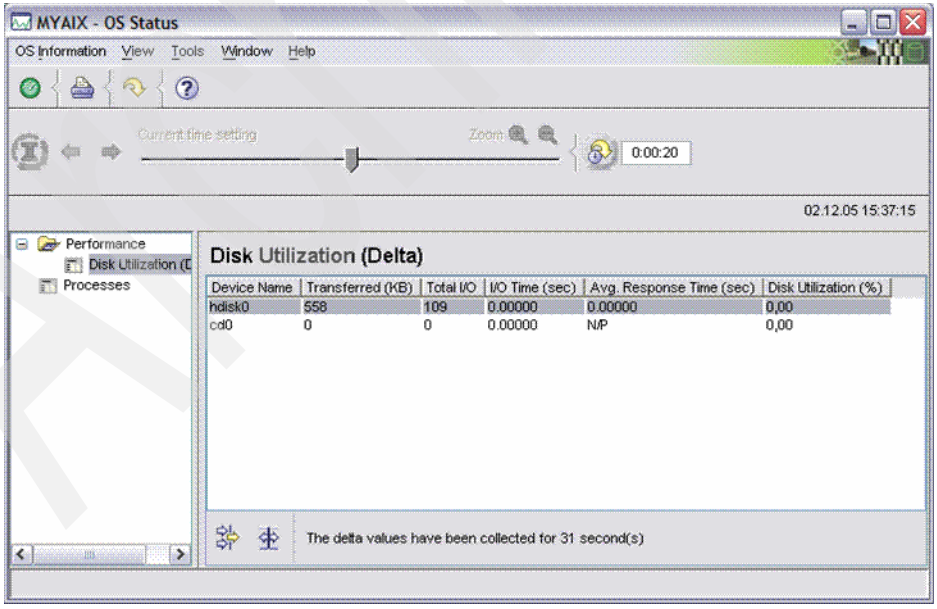


Figure 6-83 Disk utilization information (AIX, Linux)

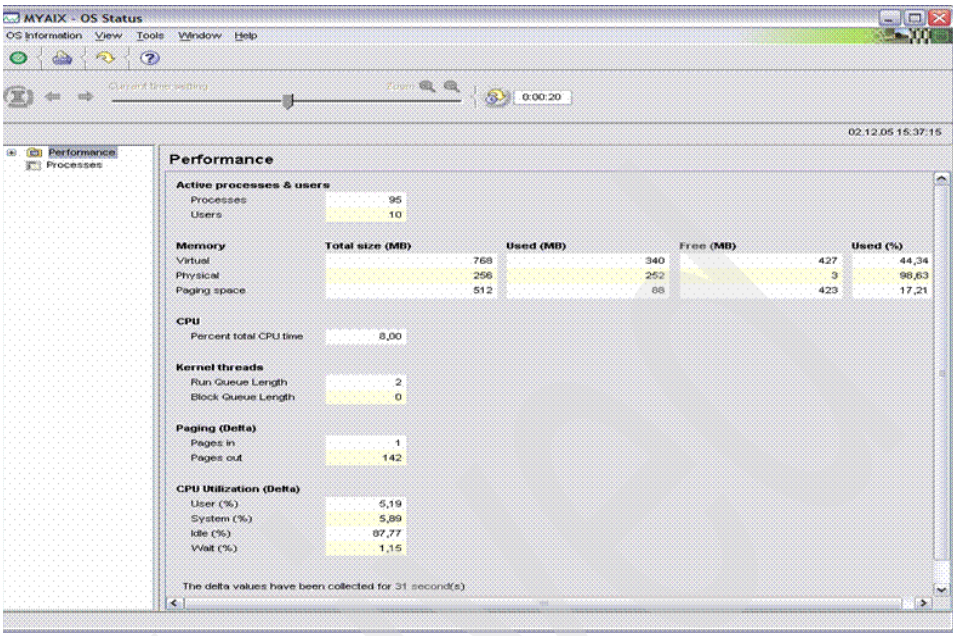


Figure 6-84 Memory consumption and CPU Utilization information

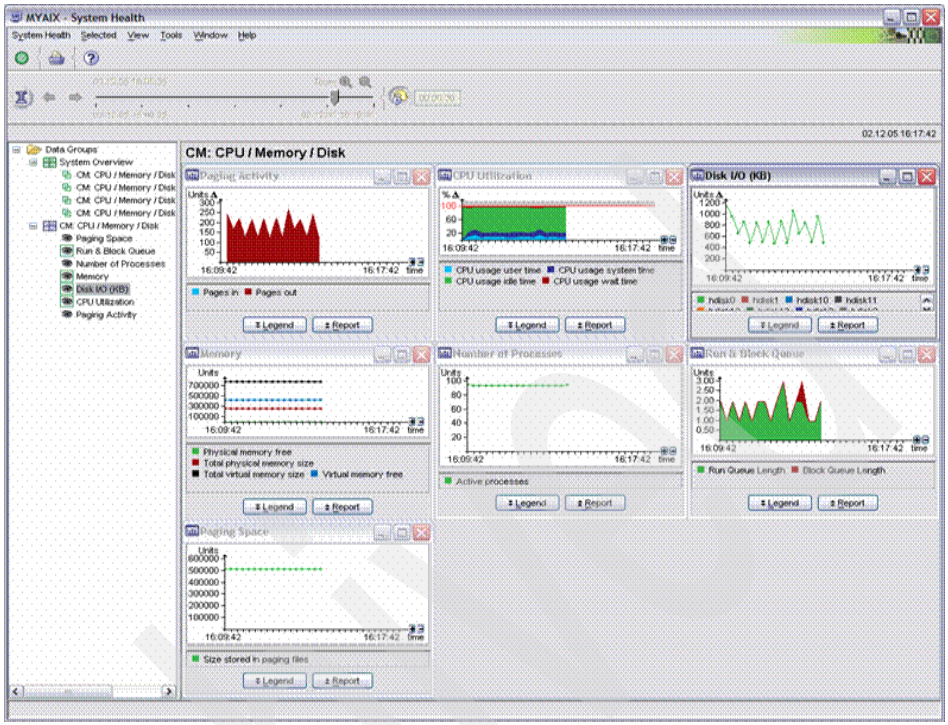


Figure 6-85 System Health data views on operating system information

C

DB2 Performance Expert For Business Intelligence

This appendix focuses on the typical and most important performance issues that occur in Business Intelligence (BI) environments and how DB2 Performance Expert (PE) can be used to analyze and fix them.

The appendix contains two main parts. The first part introduces DB2 PE and the features that are most relevant to BI systems. The second part contains a set of helpful and detailed scenarios that show you how to use DB2 PE to monitor and tune a BI environment.

This appendix is not intended to be a complete DB2 BI tuning guide or a complete DB2 PE guide.

The appendix is based on DB2 Performance Expert for Multiplatforms Version 2.2 with FixPack 1 and DB2 Version 8 with FixPack 10. Some of the descriptions and screen captures that illustrate operating system-level monitoring are specific to AIX and Linux and are slightly different than those for Solaris.

Introduction

DB2 PE is a general purpose DB2 performance monitoring and analysis framework. It is flexible enough to be applied in all types of DB2 environments. In this redbook, the focus is on how to leverage PE in DB2 BI environments.

PE is based on a three-tier monitoring architecture. Tier 1 is the monitored DB2 system. Tier 2 is the DB2 PE server that remotely monitors tier 1. It also enables all the advanced features like performance history capturing, performance warehousing, and exception processing. Tier 3 is the PE client that connects to the PE server and is used to perform all monitoring tasks.

BI systems usually have typical data models and workloads, as well as typical performance problems. PE can help you to sort out these problems and find the best solutions.

DB2 PE features for BI

This section describes the monitoring and analysis features of DB2 PE and explains why these features are valuable for BI systems.

The entry point for all PE features is the PE System Overview window (Figure C-1 on page 527). You use this window to select and monitor different DB2 systems from the navigation tree, which is located in the left most pane. The center pane contains links to the major monitoring and analysis features, as well as a number of diagrams that display key performance indicator data. The right most pane displays the performance exceptions that PE has detected on the currently selected system.

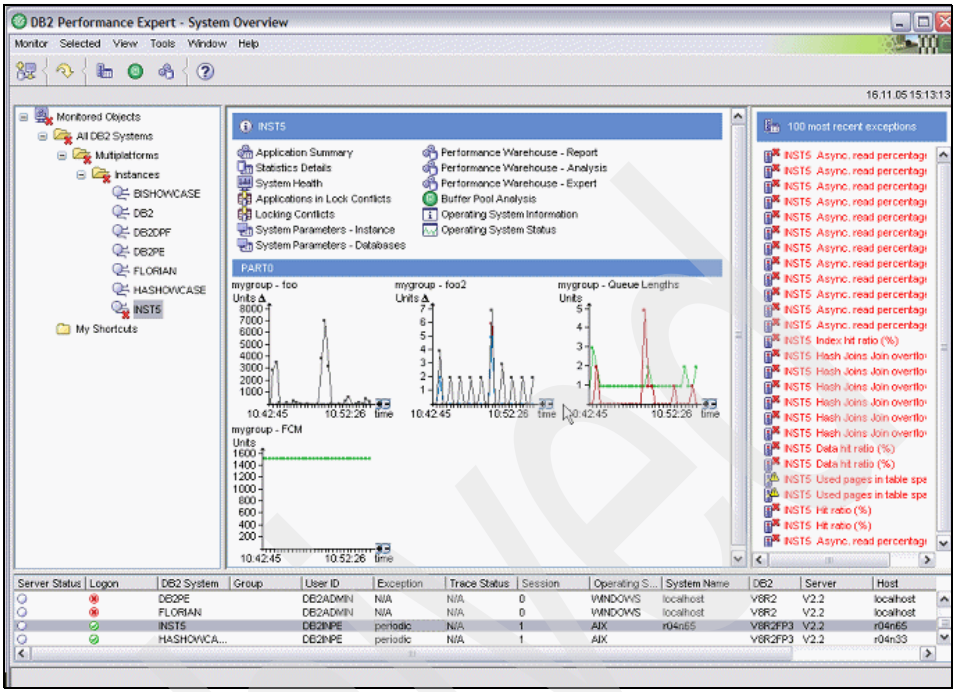


Figure C-1 PE System Overview

DPF monitoring and skew detection

PE is fully enabled to monitor a multi-partitioned DB2 system. With a single mouse click, you can retrieve monitoring information for the whole system. You can conveniently get an overview of your entire distributed system.

This capability also is the basis for quick and easy detection of skews. A skew is an uneven distribution of behavioral or residual distribution of performance indicators over the distributed system components. Skews are always potential reasons for performance problems.

Partition level skews

Most DB2 performance indicators can be retrieved for each partition. PE allows you to display an aggregated overview for these performance indicators (called a *global view*), or you can display the current values for a selected partition (called a *partition view*). In addition, PE provides a special view mode called the *group view* (Figure C-2 on page 528), which provides a matrix view of individual values. This view allows you to look for skews in the different performance indicators.

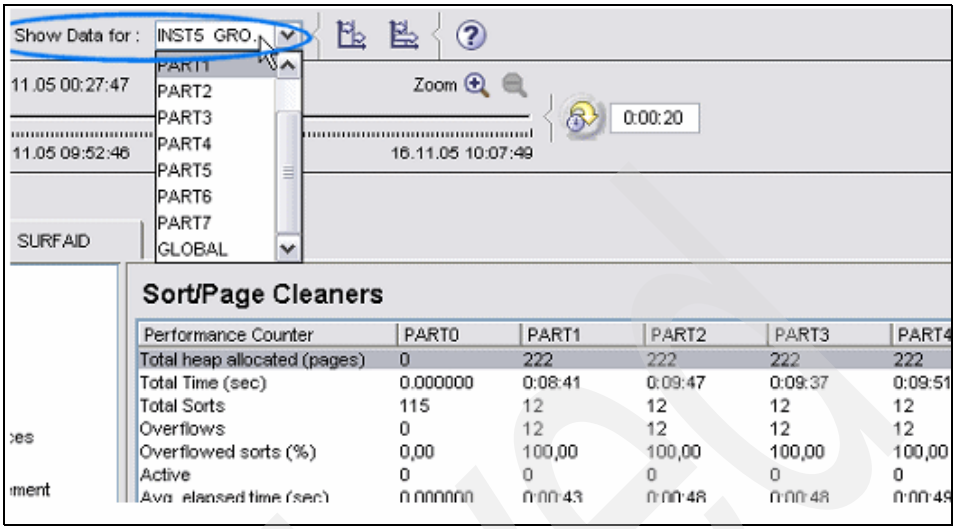


Figure C-2 PE group view

System-level skews

Because PE also provides operating system level information, it also offers you a special group view for the data group members that are comprised of machines rather than partitions (Figure C-3). The list is comprised of all distinct machines that are part of the distributed DB2 instance.

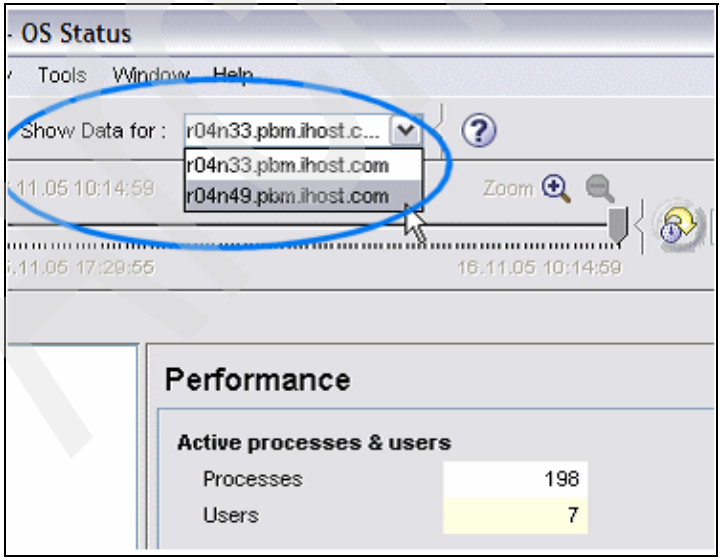


Figure C-3 System-level skews

Engine monitoring

PE provides monitoring at the DB2 engine level (that is, performance metrics on the level of instances, databases, and subsequent entities, such as tablespaces, buffer pools, and tables). You can access this data by double-clicking Statistic Details (Figure C-4) on the PE System Overview window.

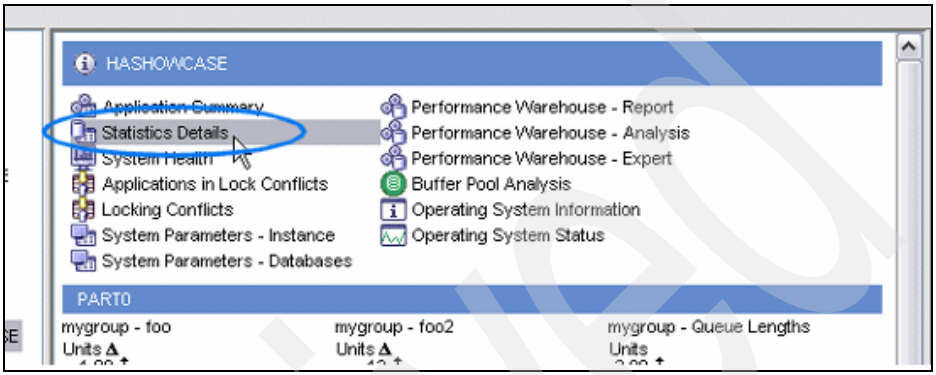


Figure C-4 PE Statistic Details

The Statistic Details windows are divided into categories in a navigation tree that is located in the left most pane. You can find the metrics that are related to a subsequent level (for example, a database) by double-clicking one of the entries in the associated category (Figure C-5).

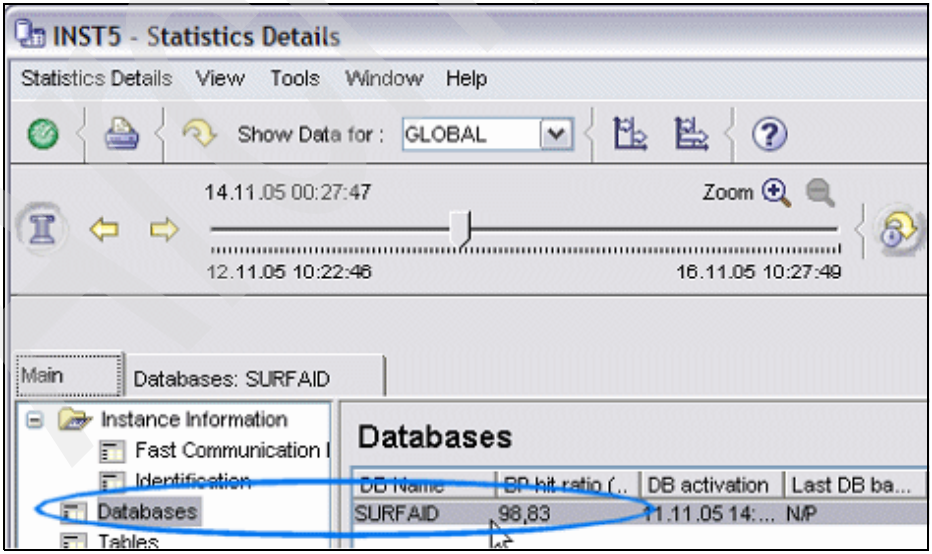


Figure C-5 PE Statistic Details - Databases

Doing so opens another tabbed pane in the same window that contains details about the selected database.

At the top of the panel is an auto-refresh option (Figure C-6).

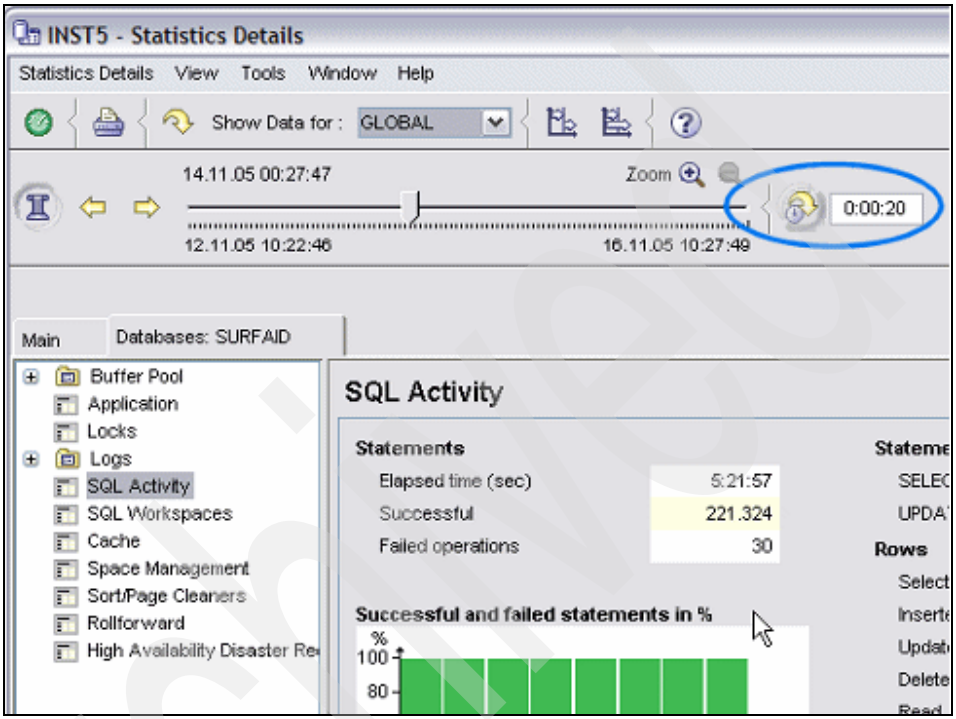


Figure C-6 PE Statistic Details - auto fresh

Some of the data can be seen as key performance indicators that you typically check first in a BI environment. Some of the most important ones are shown in the following sections.

BP hit ratio

Buffer pool hit ratio indicates the effectiveness of the buffer pool. Even though it is desirable to have the ratio at a rather high value (> 80%), this is typically not achievable for larger BI systems. Increasing the buffer pool size might have a minimal effect on the hit ratio. The number of pages might be so large that the statistical chance of a hit is not improved by increasing the size.

This metric can be retrieved on three different levels of the DB2 engine: database, tablespace, and buffer pool (Figure C-7 on page 531).

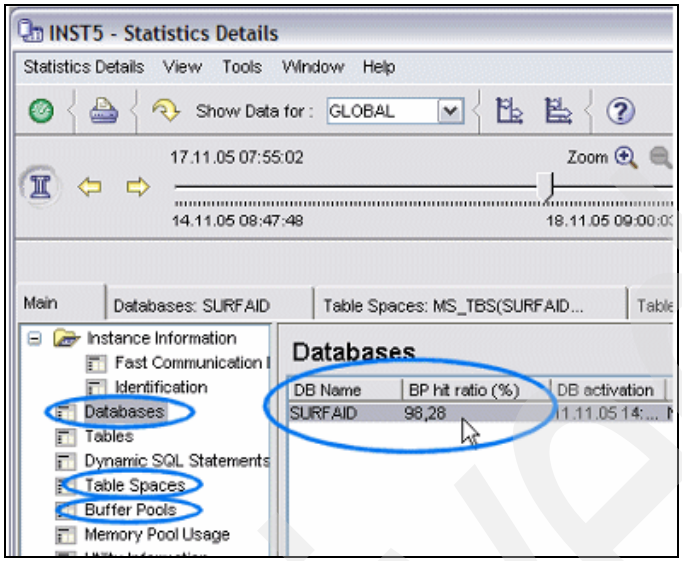


Figure C-7 Buffer pool hit ratio

Asynchronous read ratio

In BI systems, a large amount of table data usually needs to be read. DB2 provides the feature of sequential prefetching that allows asynchronously fetching pages ahead of access if the optimizer expects the next rows requested to be on these consecutive pages. With the asynchronous read ratio (synchronous versus asynchronous reads) metric, you can see to what extend pages are read via this method. A high number is usually desired for BI systems.

This metric is available on database, tablespace, and buffer pool level, like the buffer pool hit ratio. Figure C-8 on page 532 shows an asynchronous read ratio.

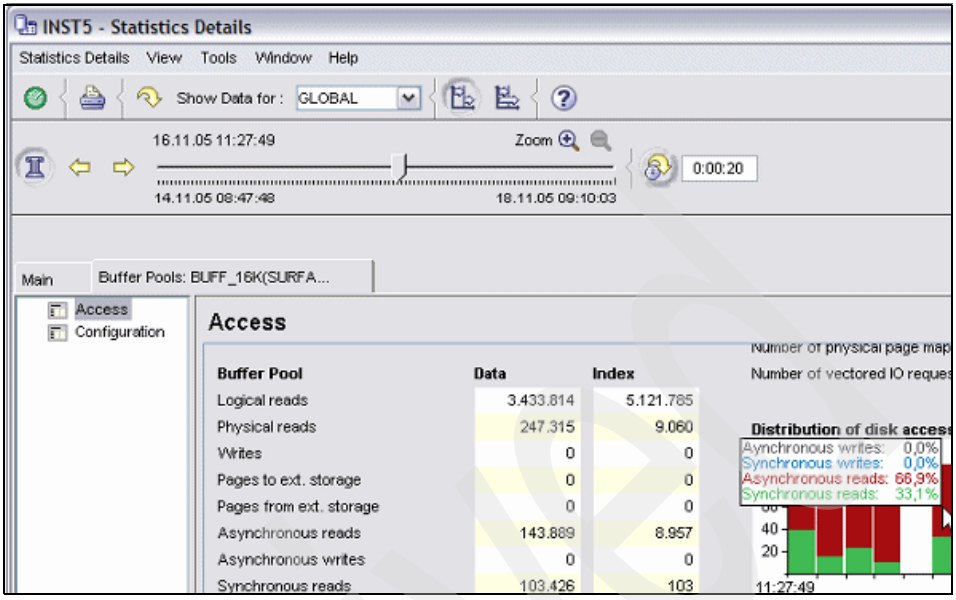


Figure C-8 Asynchronous ratio

Read efficiency

How efficiently queries are executed with regard to how many rows have to be read from disk compared to how many rows are actually returned (that is, are part of the result set) is a crucial metric. If this number is considerable, DB2 reads much more data than the user actually requests (Figure C-9 on page 533).

You can check this metric on the database, tablespace, buffer pool, and table level.

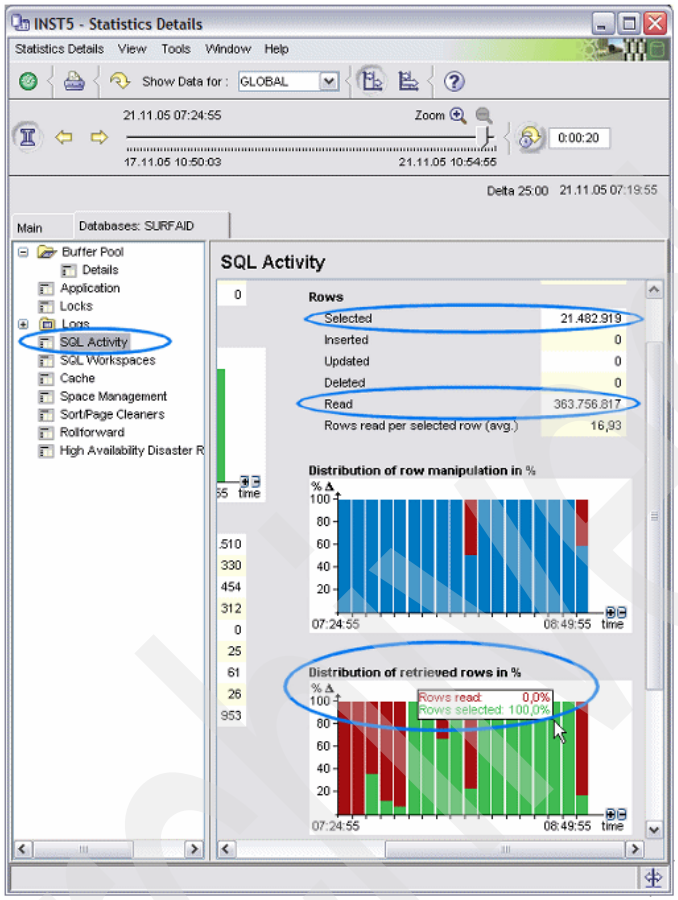


Figure C-9 SQL Activity

FCM buffers

The Fast Communication Manager (FCM) is a key component of the DB2 data partitioning feature (DPF). It is used for communication between the partitions. That is why PE provides a dedicated pane for its performance metrics (Figure C-10 on page 534).

“Check if a system is CPU bound” on page 545 describes FCM tuning in more detail.

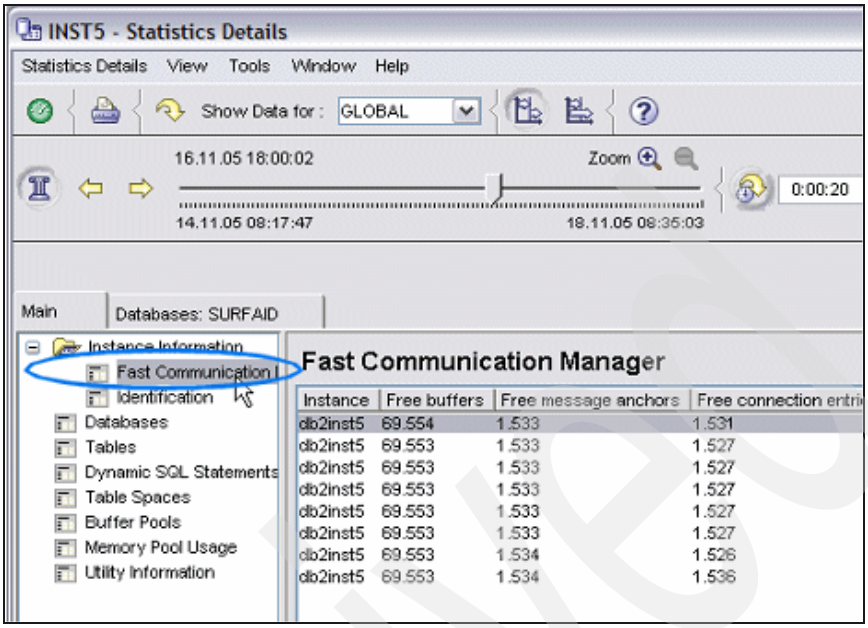


Figure C-10 FCM buffers

Sort

Sorting data is typically necessary, to a large extent, for execution of BI SQL workloads. Tuning the engine accordingly is a key task for BI administrators. For this reason, PE displays the key sort performance indicators for the DB2 engine on a dedicated pane in the Statistic Details window (Figure C-11 on page 535).

“Sort and hash join tuning” on page 550 describes how to address sorting issues.

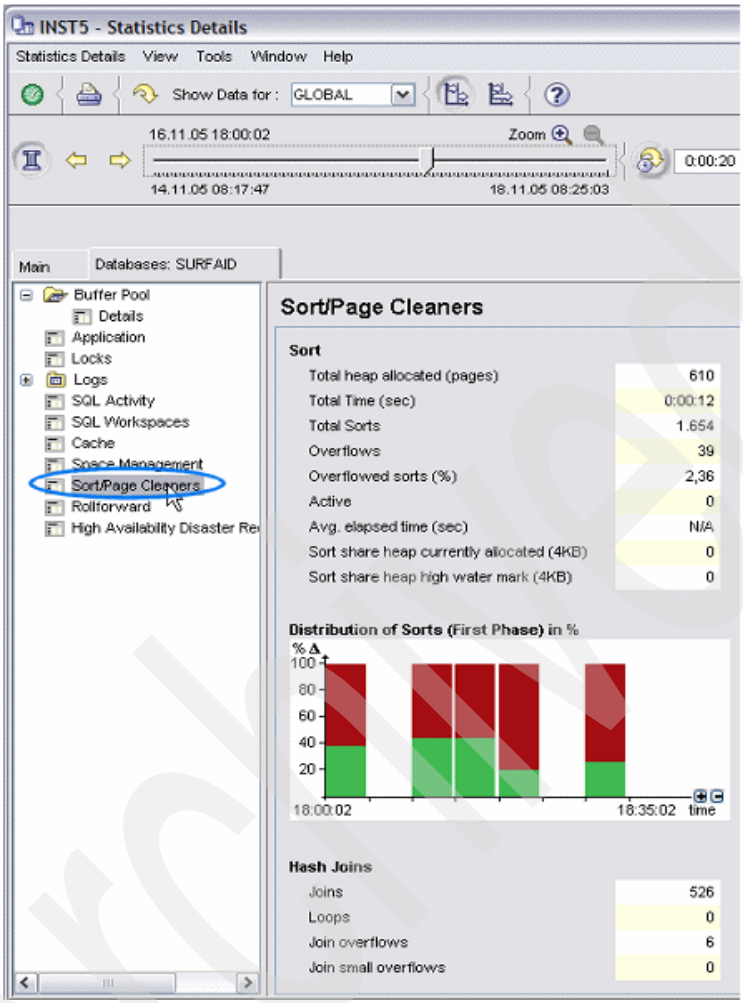


Figure C-11 Sort statistics

Locking

When BI systems require concurrent incremental loads of new data while the query workload continues, you must also check for locking issues. A dedicated Locks pane (Figure C-12 on page 536) gives you the best overview of this task.

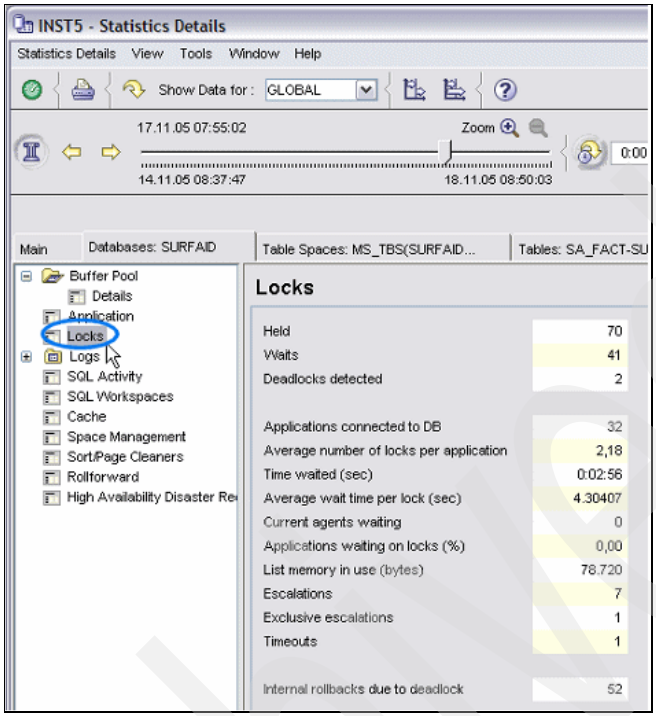


Figure C-12 Locks

In addition, PE proactively informs you about deadlocks, including the relevant details. Furthermore, the System Overview window provides access to special dialogs for analyzing locking conflicts (Figure C-13).

For more details about tuning load, see “Monitoring and tuning load” on page 585.

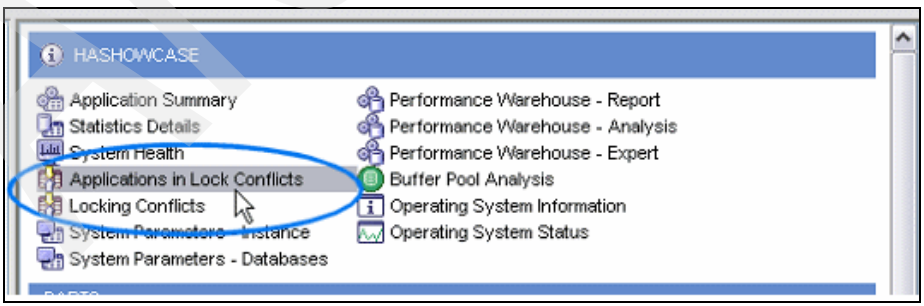


Figure C-13 Lock conflicts

Dynamic SQL statements

The main purpose of a BI system is to return the answers for analytic queries. Therefore, it is crucial to keep track of the queries that the BI system is running and how they are performing. The best starting point is the Dynamic SQL Statements pane (Figure C-14) in the Statistics Details window. It shows you each statement that was recently executed along with such key properties as number of executions, compile time, and execution time.

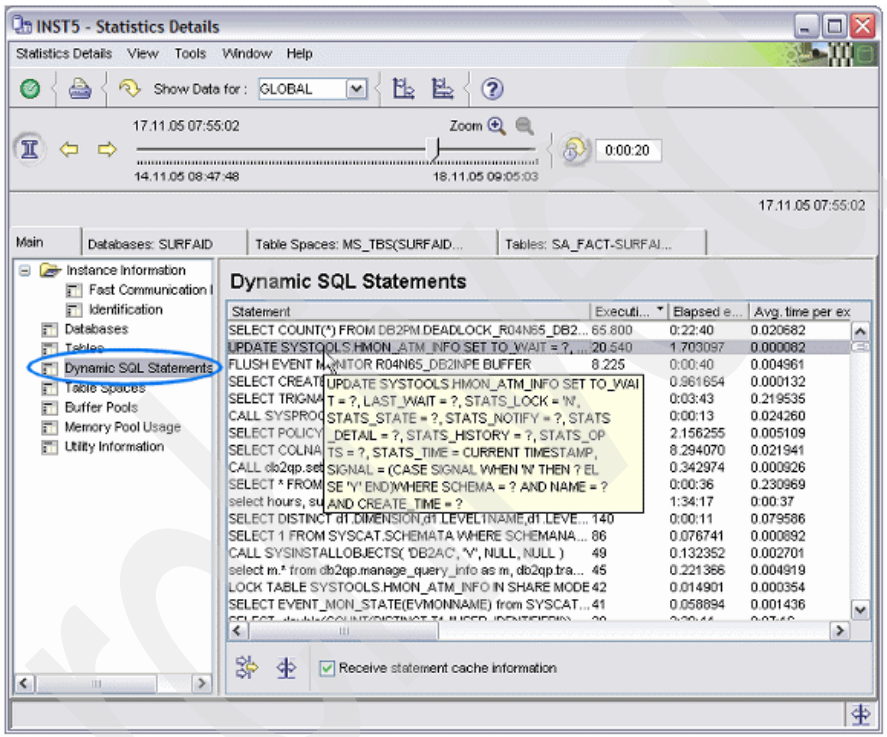


Figure C-14 Dynamic SQL statements

Application monitoring and tracing

Applications are a crucial element to monitor in BI environments because they typically execute long running queries. PE provides a special view for application specific metrics that can be accessed by clicking the Application Summary from the PE System Overview window.

The window that opens shows a list of all active applications and includes a couple of important performance metrics (Figure C-15 on page 538). You can see all details about a certain application by double-clicking it. The Application Details pane opens, which can be used in a manner that is similar to the Statistic

Details pane to navigate to different types of application metrics. Most of the metrics that are described above for the DB2 engine level are available on a per-application level here (buffer pool hit ratio, asynchronous read ratio, read efficiency, and sort metrics).

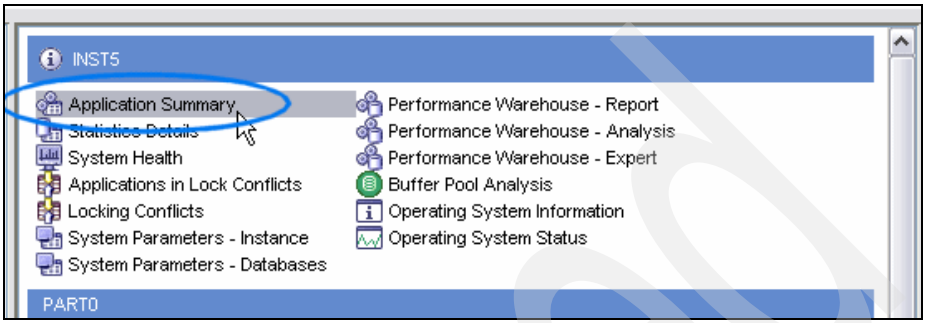


Figure C-15 Application Summary

History analysis

PE provides a convenient way to monitor the historical performance of a DB2 system. The panels that are opened from the PE System Overview window are opened in online monitoring mode by default. With a single mouse click, you can switch to history mode. A history slider allows you to conveniently browse through historical data.

By default, this history data wraps after 50 hours. You can configure to any other number. Just be aware, the more hours that you keep, the larger the performance database on the PE server will grow.

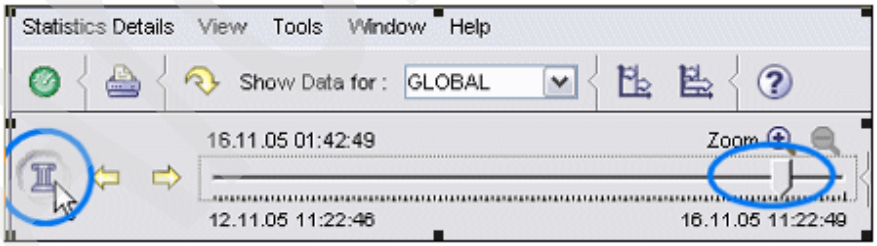


Figure C-16 History analysis

Performance Warehouse analysis

PE has special long-term storage for performance data called the Performance Warehouse (PWH). Along with the data, a set of analyzing features is provided (Figure C-17). A predefined set of HTML reports can be generated and predefined, and custom queries can be executed, and predefined and custom rules of thumb can be evaluated to check for performance problems that were experienced by the BI system over time.

The query capability is referenced extensively throughout the remainder of this redbook.

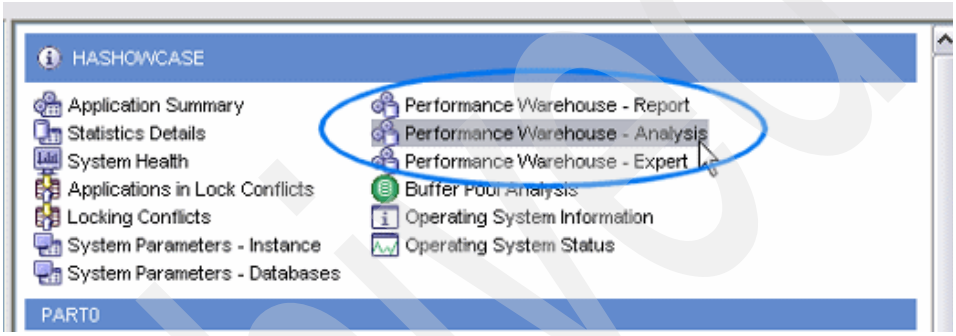


Figure C-17 Performance warehouse analysis

Operating system monitoring

In addition to a large variety of DB2 metrics, PE also keeps track and displays OS-level metrics. There are two links in the PE System Overview window to view OS-level data. *Operating System Information* links to the rather static data, such as configured memory, installed CPUs, and file system information. *Operating System Status* links to more volatile metrics about paging, processes, CPU consumption, and disk utilization (Figure C-18). All OS-level data is enabled for history monitoring.

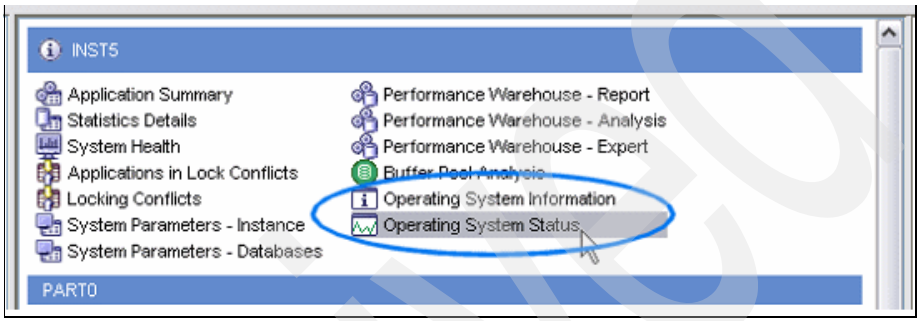


Figure C-18 Operating System monitoring

File system usage

As your BI system grows over time, you typically want to verify that the storage resources are still sufficient. You might also want to keep an eye on storage that is needed only temporarily. This is important because BI systems tend to perform large sorts that might require some dedicated disk space. The OS Information window provides an overview of storage resources (Figure C-19 on page 541).

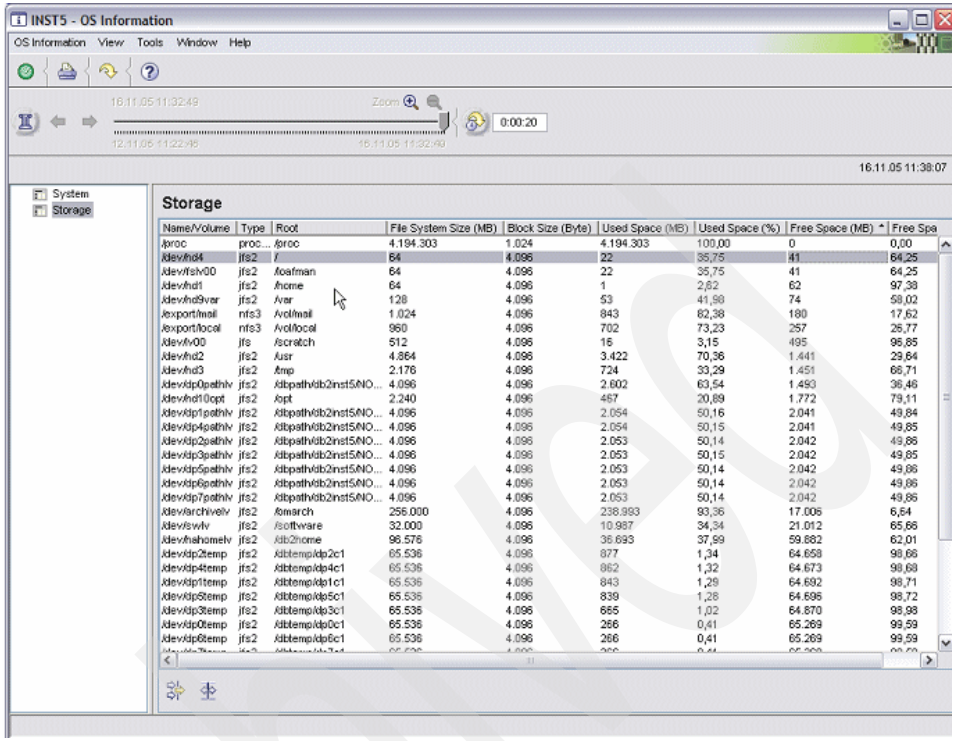


Figure C-19 File system usage

Disk I/O

Click the **Disk Utilization (Delta)** category of the OS Status window to display disk I/O metrics (Figure C-20). This window shows the transferred (that is, read and written) kilobytes and total number of I/O operations. Note that this data is based on delta intervals. You can either manually refresh the window or see this data based on the time between the recent two refreshes, or you can use the auto-refresh option.

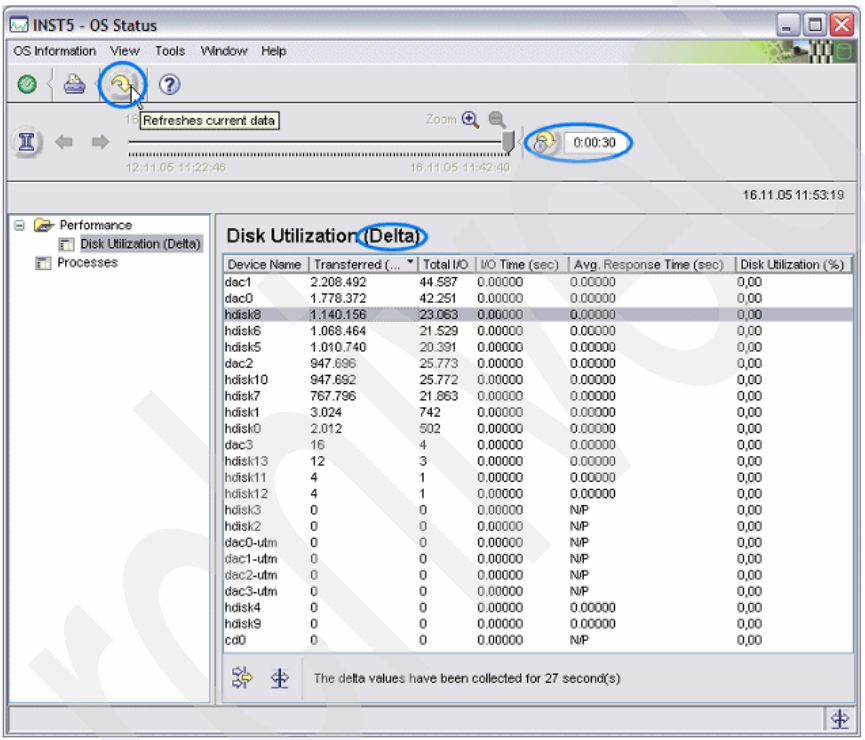


Figure C-20 Disk I/O

CPU utilization, queues, and paging

The OS Status window also contains information about CPU and paging in the Performance category (Figure C-21 on page 543).

The Run Queue Length field indicates how many processes are actually ready for execution, and the Block Queue Length field shows the number of processes that are waiting for I/O to finish.

Paging is counted in the number of pages being paged in or paged out.

The CPU utilization is divided into user time, system time, I/O wait time, and idle time.

Both CPU utilization and paging data is based on delta values. You can either click the **Refresh** button to see the data based on the time between previous and current refresh, or you can use the auto-refresh feature.

The paging rate and block queue length data can also be visualized in the System Health pane, and they can be used for threshold definitions for exceptions.

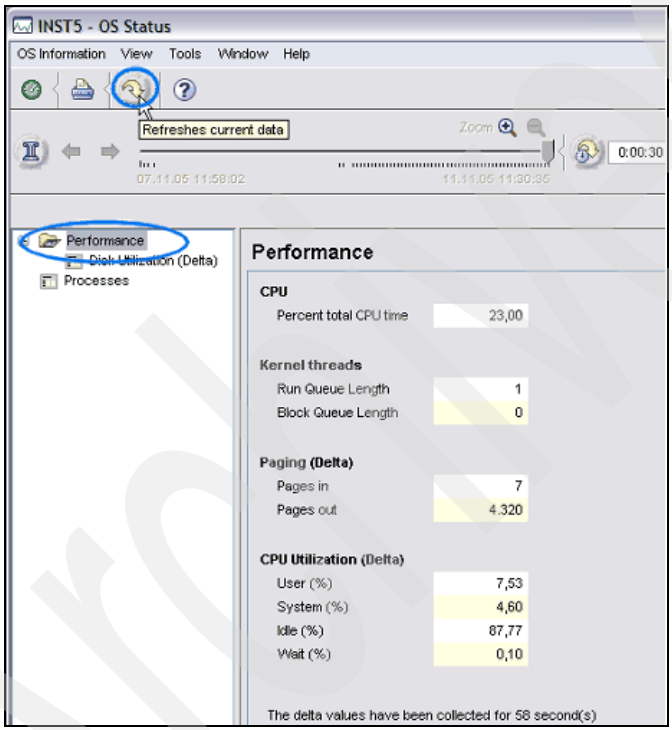


Figure C-21 CPU utilization

Visualization

For all performance indicators that have been described so far, PE also provides a visual way of displaying them in the System Health window (Figure C-22).

In this window, you can set up and view a number of dashboards with predefined or custom defined diagrams (called data views). Like the other dialogs that have been described so far, the System Health window is fully enabled for history monitoring and for DPF monitoring (group views).

This allows you to set up dashboards with the BI key performance indicators as described in “Dashboard monitoring of key BI performance indicators” on page 598.

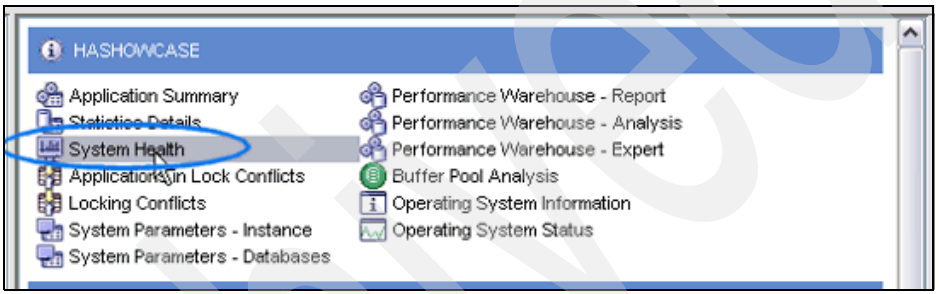


Figure C-22 system Health view

Threshold exceptions/predefined threshold set

A BI system typically has certain threshold values for performance indicators that you should be aware of when they are exceeded. For this purpose, PE has the built-in feature of periodic exception processing, which basically means that it checks defined performance indicators against threshold definitions and raises an exception (visually, by e-mail, or by executing custom code).

To enable this feature, create a set of threshold definitions and activate them so that they are considered in periodic exception processing (Figure C-23).

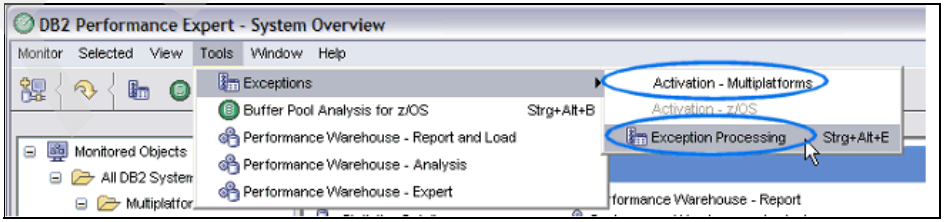


Figure C-23 Exception Processing

When you create such a new threshold set, you can use the built-in BI template so that you do not need to start from scratch (Figure C-24).

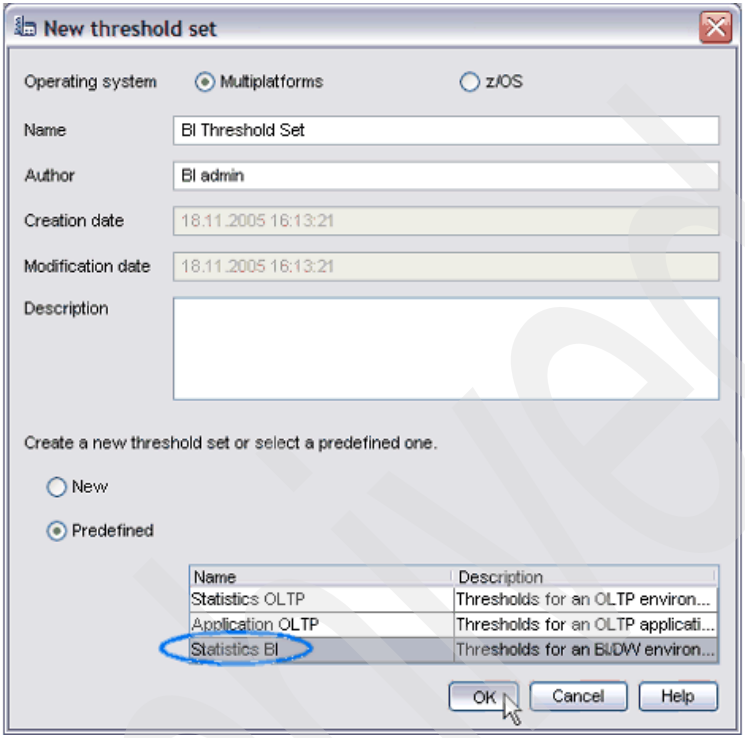


Figure C-24 Exception processing template

BI Performance Tuning scenarios with DB2 PE

This section describes a set of typical tuning scenarios and how PE facilitates them.

Check if a system is CPU bound

The desired state is for the system to be CPU bound, that is, to not spend precious CPU cycles waiting on any resources, such as I/O.

General approach

A straightforward method to determine if the monitored system is CPU bound is to open the OS Status window and to display the Performance pane. Next, switch to auto-refresh mode so that you are permanently presented with the current state of the system. Then check the Block Queue Length field. It should have a low value near 0, because it counts each process that currently has to wait for a resource, such as a disk or network.

Next, check the CPU utilization fields. The majority of the time should be consumed by user and system time. If wait time is a two-digit percentage over a longer time, the system is I/O-bound rather than CPU-bound and must be revised accordingly. Figure C-25 shows some sample OS performance data.

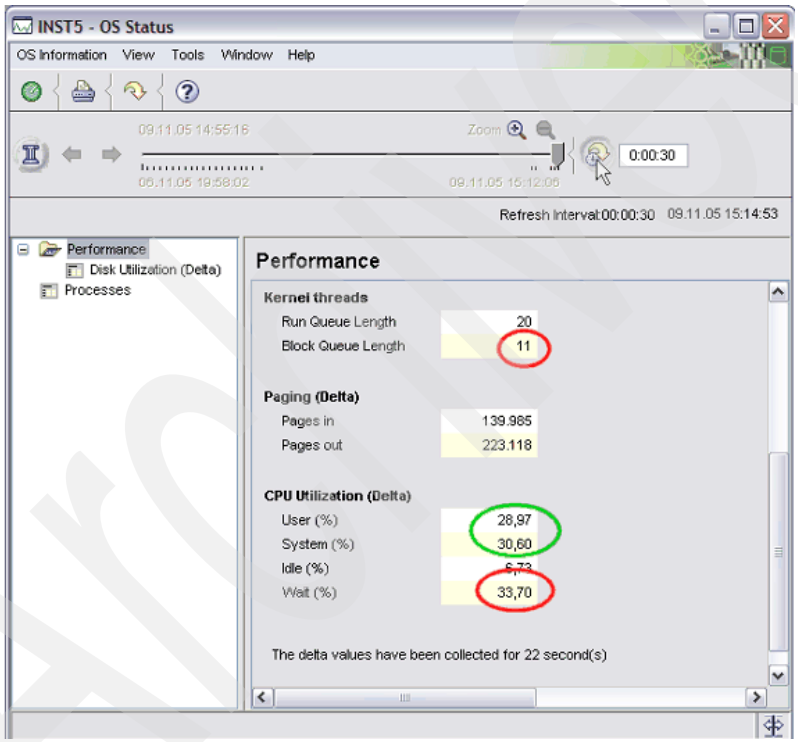


Figure C-25 Operating system performance sample data

Visualization

The queue length metrics can also be visualized in the System Health and System Overview windows. Open the System Health window, right-click a data group, and select **New…**. Name the new data view “Queue Lengths” and select the category **Operating System Status, Performance** before clicking **Next**. Now select the counters **Run Queue Length** and **Block Queue Length** and

click **Next**. Select **Dynamic scale** and click **Next**. Now select the chart type you like, for example, Line chart, to display a data view like the one shown here. Now click **Finish**. Figure C-26 shows the Queue Lengths chart.

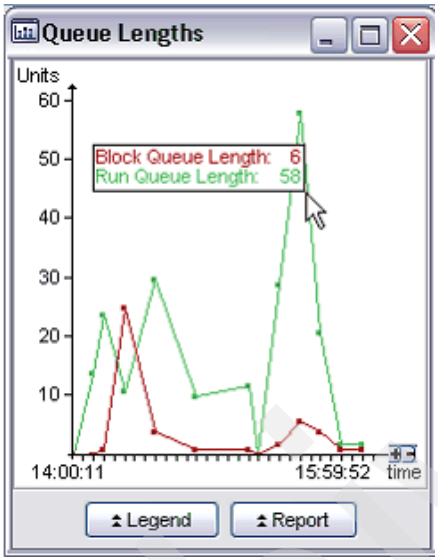


Figure C-26 Queue Lengths chart

If you want the data view that you just created to be displayed in the System Overview window, right-click it and select **Display in System Overview**.

Exceptions

The Block Queue Length can also be automatically checked by PE against a threshold. To enable this feature:

1. Click **Tools** → **Exceptions** → **Exception Processing**.
2. Double-click a threshold set and select **Threshold Set** → **New Threshold**.
3. Select the **Operating System Status** exception category and the exception field **Block Queue Length**. Specify a warning and problem threshold depending on your system.

   Typically, the block queue length should be below 10, so a warning threshold of 10 and a problem threshold of 15 might be a good start.
4. Click **OK**.
5. Activate this threshold set by clicking **Tools** → **Exceptions** → **Activation - Multiplatform**. In the dialog, select the threshold set and then press the start button for Periodic Exception Processing.

For other exception processing related features, such as e-mail notification, refer to the *IBM DB2 Performance Expert: Installation and Configuration,* SC18-0101.

C.0.0.1 Identify I/O bound partition

Use the following method to identify I/O bound partitions. This method involves checking for temporary tablespace skew.

1. Open Statistical Details pane for tablespaces and double-click the system temporary space that you want to check in that exercise.
2. The *Space Management* pane is displayed (Figure C-27). Make sure that you are in Global view mode.

   If there is repeating data skew on temporary tablespaces in the same partitions, this is an indication that these partitions are I/O bound or at least significantly more I/O bound than the other partitions.

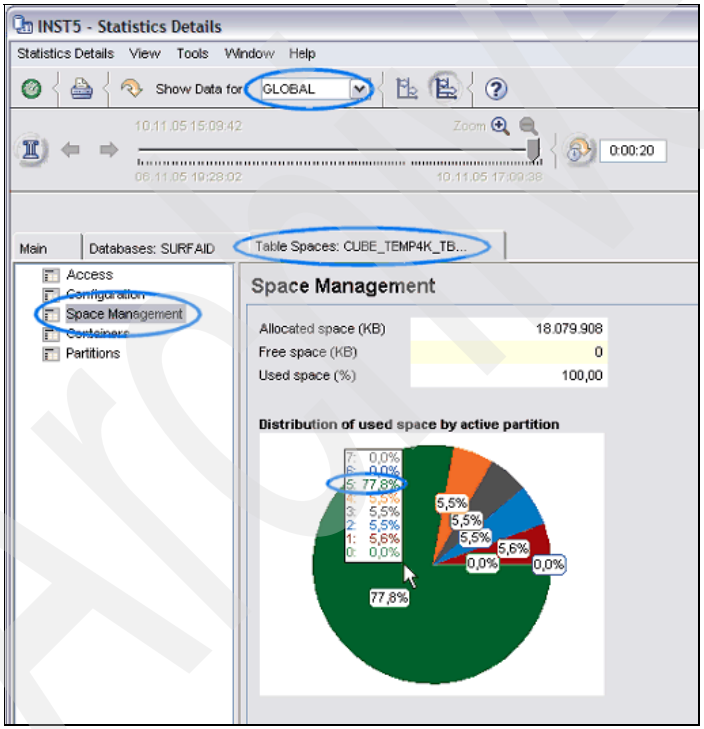


Figure C-27 Space management

Before concluding that a partition is I/O bound based on this method, you must eliminate other potential reasons for data skew on temporary tablespaces, which could be:

- Data skew on the base tables so that some partitions have to sort more than others.
- Current workloads are using only certain or even a single partition because the base tables to not span the same partitions as the temporary tablespace does.

To eliminate the first of these reasons, display the Application Summary window in group view. Sort the table on the Partition column and check all entries of the partition with skew detected above (Figure C-28). Within this group, look for the entry with the highest sort time. This partition is a good place to check out if the workload is causing the skew in the temporary tablespace.

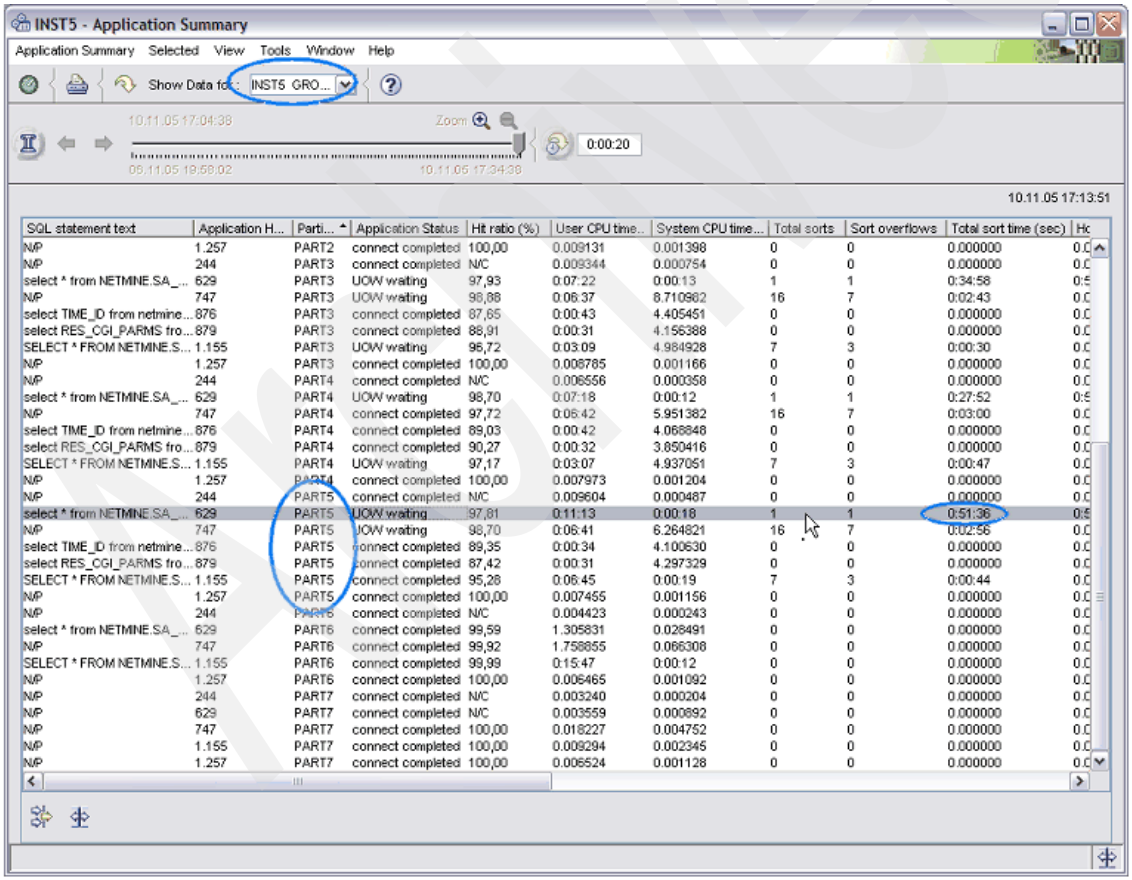


Figure C-28 Application Summary with group view

Open the details view by double-clicking the entry of interest and check the rows read and written counters (Figure C-29). Now compare these to the same counters of the details panels of the other partitions of the same application handle.

If the numbers are approximately the same, then the data skew on the base tables is likely not the cause of the skew on the temporary tablespace.

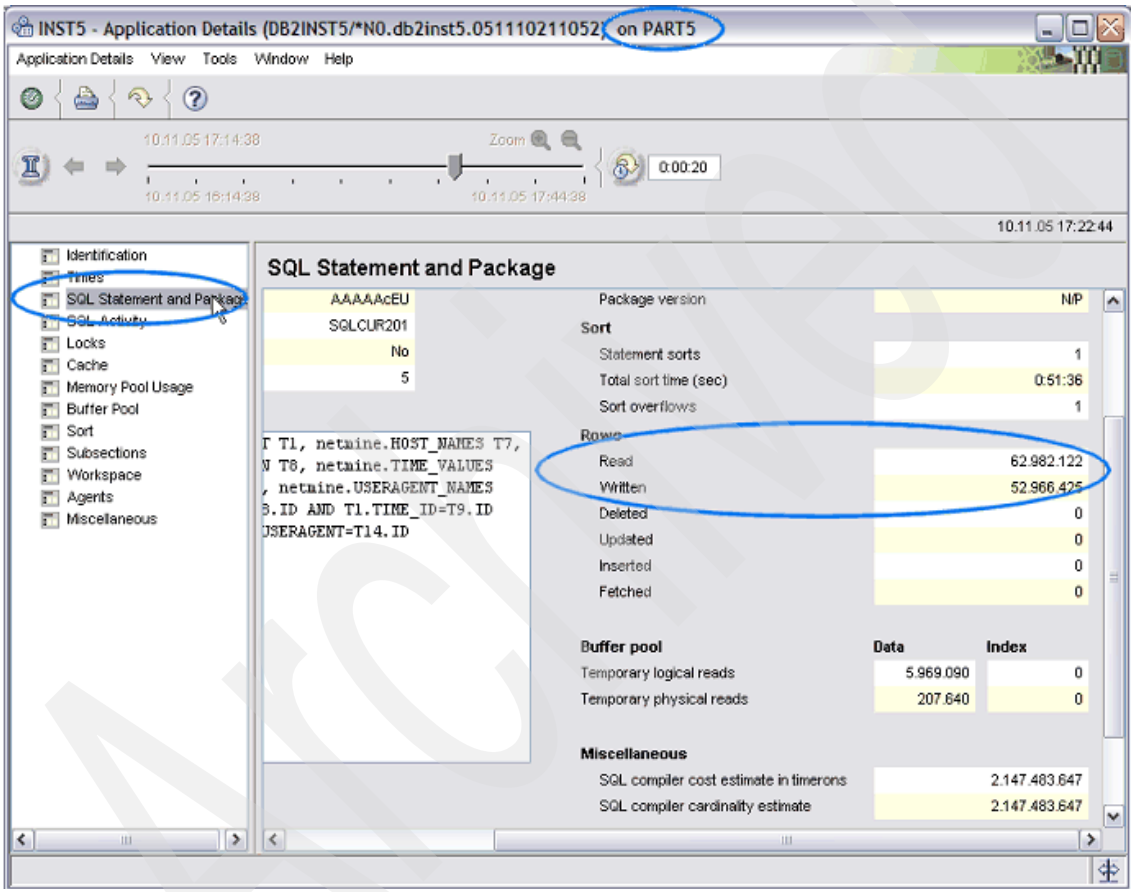


Figure C-29 Application details of a partition

Sort and hash join tuning

Sorting usually has a large impact in BI systems. For example, ORDER BY, GROUP BY, DISTICT, HAVING, or UNION clauses require the data to be sorted if there is no matching index on the according columns. It is crucial to make sure that sorting is done in an optimal way.

A good starting point is the Sort/Page Cleaner pane in the statistic details for a particular database (Figure C-30). Check for sort overflows and overflowed sorts (%). These occur when a sort cannot be done in memory anymore.

Another reason for sorting to occur is hash joining. There are special indicators for these sort types. You should look at the ratio between the number of joins and join overflows. Check for hash join overflows.

- Statistic Details window: The Join small overflows and Join Overflows fields.
- System Overview Data View for Small Join Overflows versus Join Overflows.
- If Join small overflows are more than 10% of Join Overflows, increase the sort heap.
- If Join overflows are rather high, increase the sort threshold.

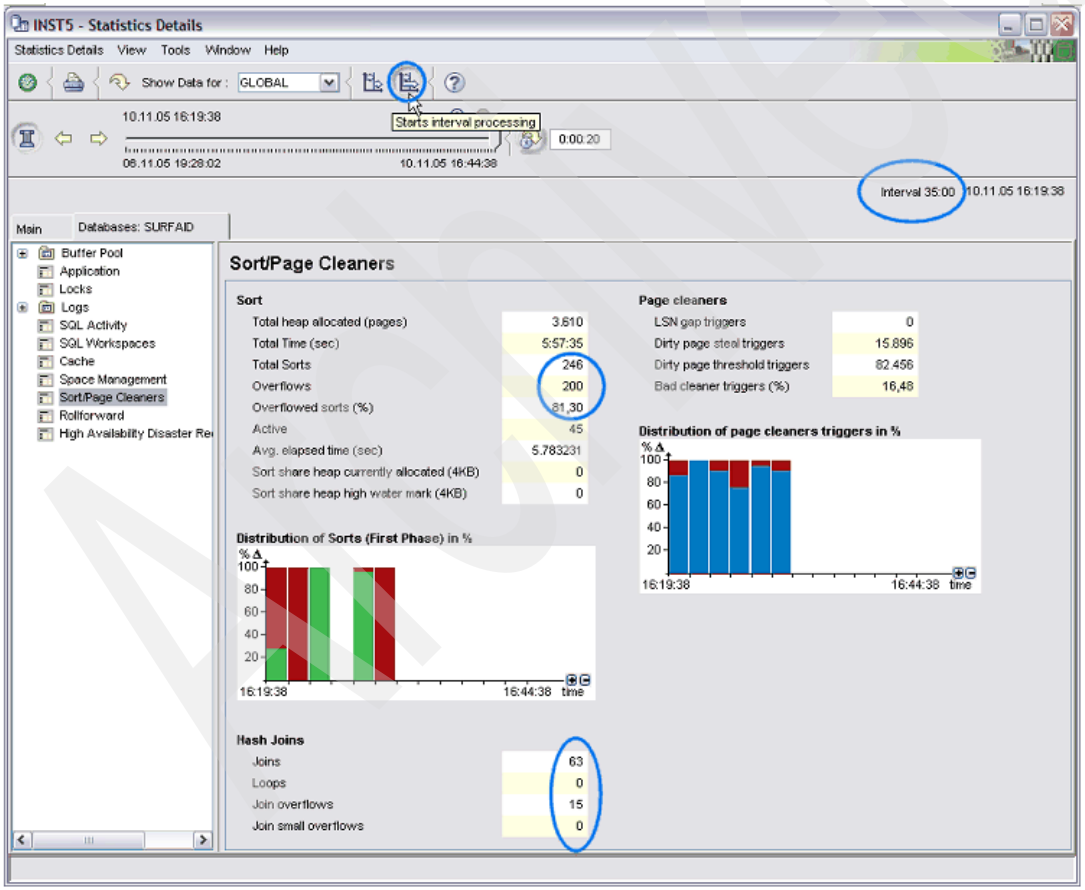


Figure C-30 Sort/Page Cleaners sample data

If you detect many overflows (of sorts or hash joins), you should determine how to lower them. Increasing the sort heap (database configuration) should help you if you see many join overflows in conjunction with many join small overflows. However, if your problem is a high number of sort overflows, you should check the overall number of post-threshold sorts, which is available in the Instance Information pane of the Statistic Details window. If the number of sort overflows significantly exceeds the number of post-threshold sorts, increasing the sort heap should help you.

Attention: When you compare these numbers, take into account that post-threshold sorts counts for all sorts on all databases, while sort overflows are always counted on a per database basis.

Before increasing the sort heap, you can use the Instance Information pane (Figure C-31 on page 553) to determine how many pages are currently allocated.

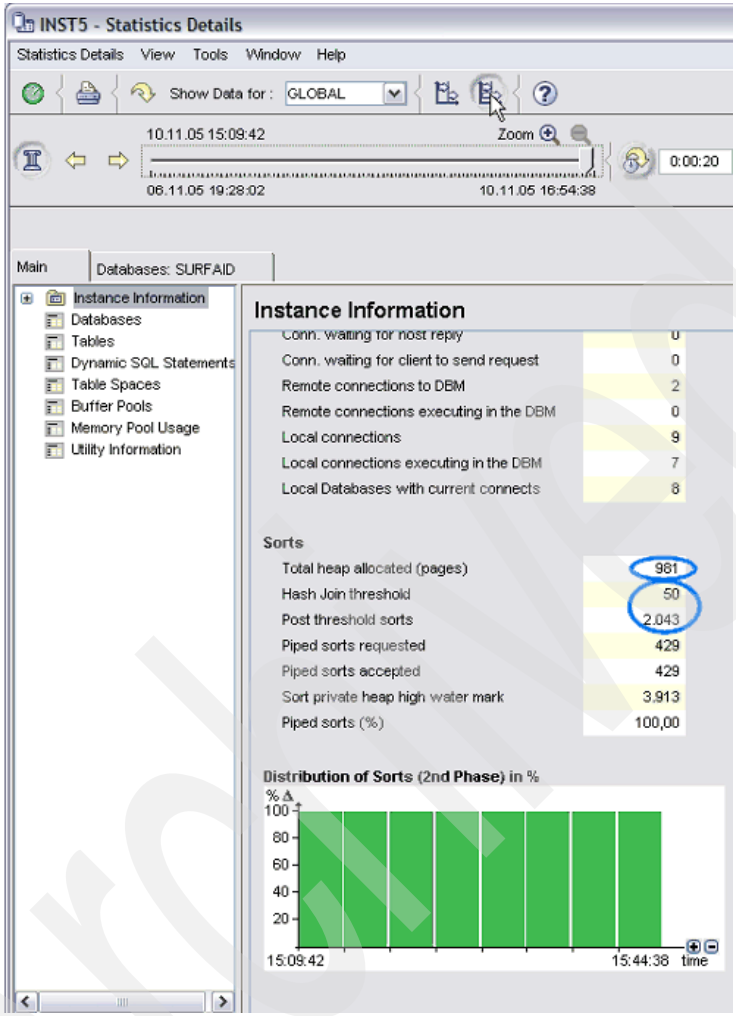


Figure C-31 Sort heap data under Instance Information

If a significant number of sorts (more than 5%) end as post-threshold sorts or as Hash join thresholds (these are post-threshold hash joins), consider increasing the sort heap threshold in the database manager configuration.

A more general way to resolve sort problems is to avoid sorting by using materialized query tables (MQTs). (You can create MQTs on typical joins, and ORDER BYs, GROUP BYs, and so on, in your base tables.) For more information about MQTs, see “Verifying MQT effectiveness” on page 595.

You can also set up data views for sort performance indicators, as shown in Figure C-32, to support a more intuitive detection of the aforementioned sort problems.

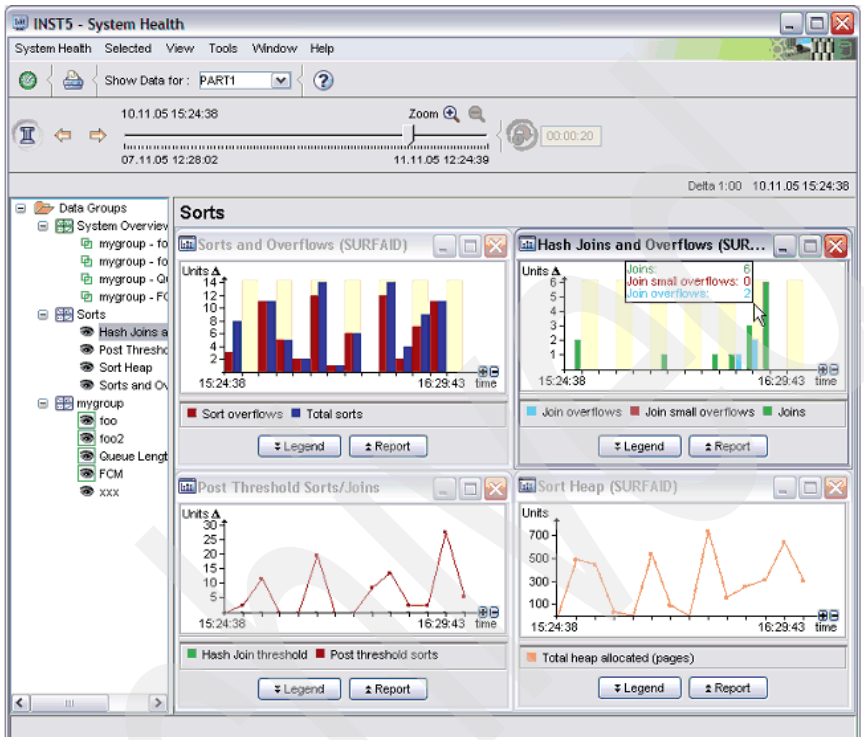


Figure C-32 Data view

If you detect many sort or hash join overflows, you might first want to check how much data is actually spilling to disk for being sorted. You can do this by looking at the temporary tablespace consumption and usage in the Statistics Details window for the temporary tablespace in the Containers and Access panes (Figure C-33 on page 555 and Figure C-34 on page 555).

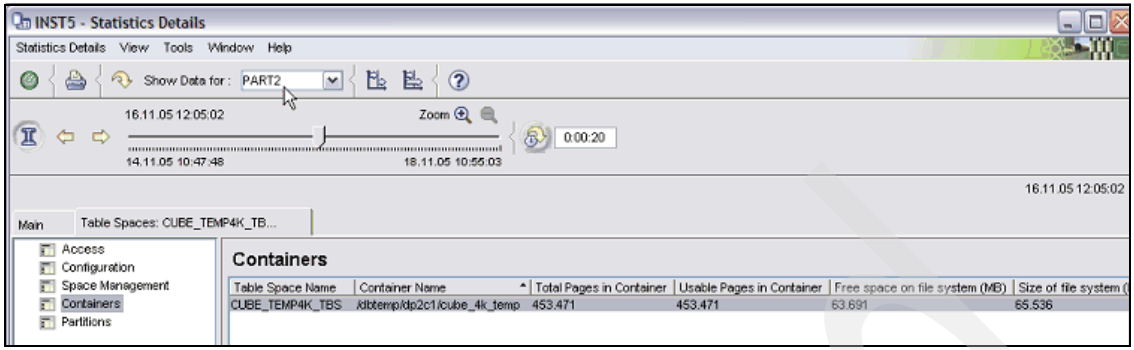


Figure C-33 Checking temporary tablespace consumption

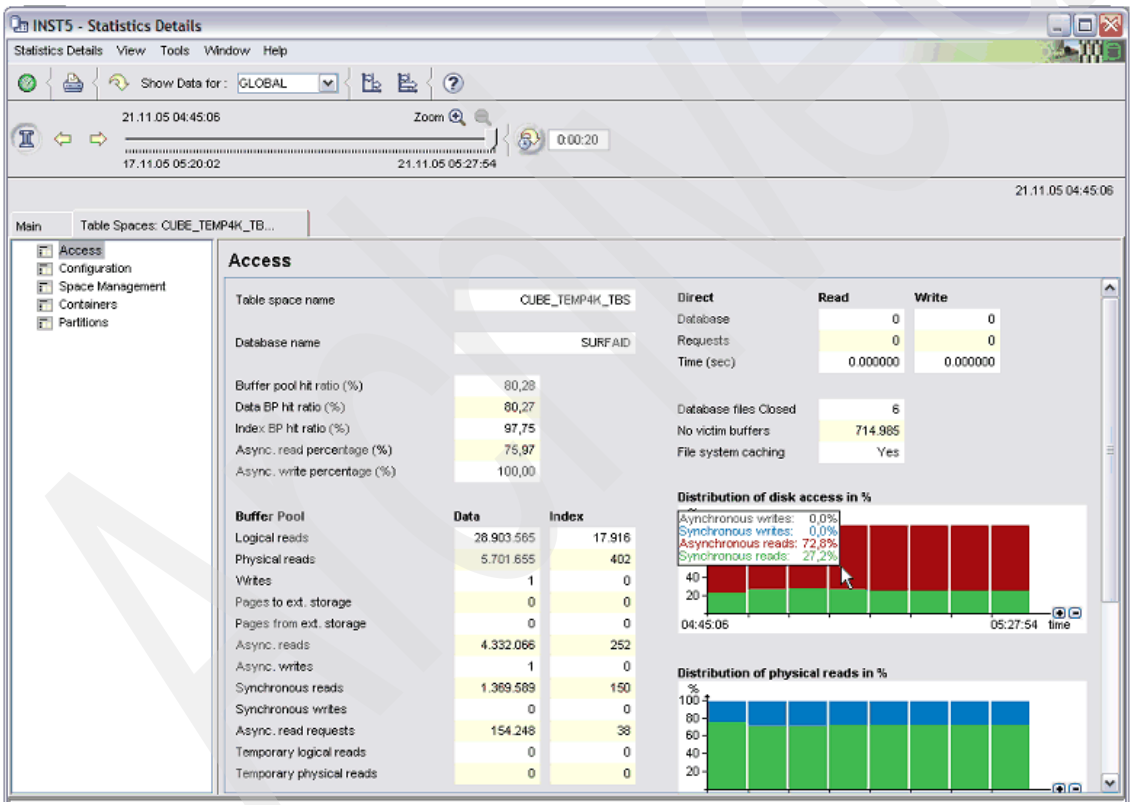


Figure C-34 Sample data in Access pane

Page I/O tuning

BI workloads impose a large amount of I/O on tablespaces. For this reason, it is essential to have a close look at page I/O metrics. You can do this analysis at the database, buffer pool, and tablespace level, as well as at the application level. This scenario focuses on the tablespace level. To check the I/O on tablespaces:

1. Open the Statistic Details window and display the Table Spaces pane.
2. Double-click a tablespace.

   You should start with your most critical tablespace (the one where the fact table or tables are located). The Access pane opens, which contains details about page I/O (Figure C-35).

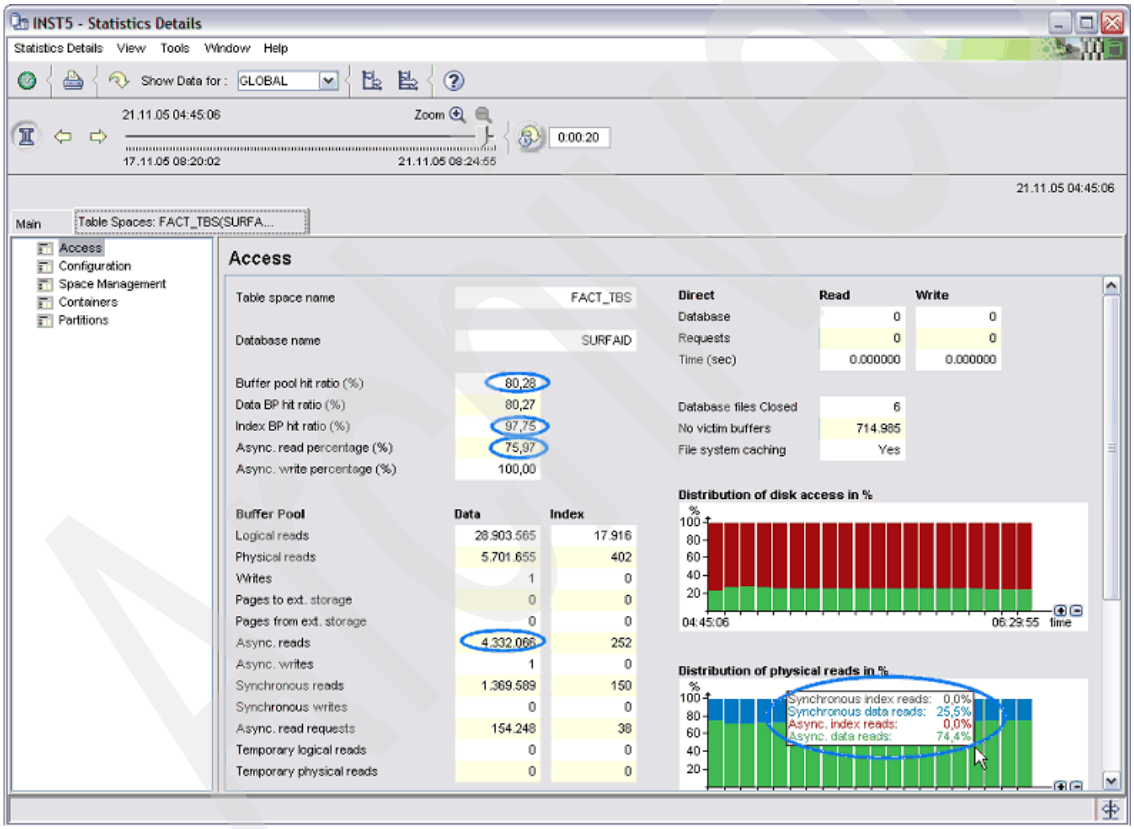


Figure C-35 Page I/O tuning

3. Evaluate the following metrics:
   – Buffer pool hit ratio

     Check the buffer pool hit ratio. The higher it is, the less page I/O has to be done to the tablespace containers. In large BI systems, it is typically not possible to achieve a high overall buffer pool hit ratio. Therefore, it is a good practice to at least tune the index page hit ratio. If your index data is located in a separate tablespace than the table data, examine the hit ratio for that tablespace and tune it by gradually increasing the buffer pool size (which can be done dynamically) and by running a test workload against the system. While you are doing this, watch the buffer pool hit ratio for the index tablespace in the Access pane. Stop when increasing the buffer pool size does not increase the hit ratio anymore.

     You can determine which buffer pool is caching data for this tablespace by looking at the Configuration pane in the *Bufferpool - Currently being used* field, which displays the ID of the responsible buffer pool.

   – Prefetching

     Prefetching is a technique that is used by DB2 to decouple I/O operations to the tablespace containers from the page requests by the process that is executing a statement. DB2 anticipates the pages that will be requested next and then asynchronously prefetches them to the buffer pool while the actual statement execution continues.

     The amount of prefetching can be seen in the *Async. Reads* and *Async. Read percentage (%)* fields and in the *Distribution of physical reads in %* graph of the Access pane.

     In addition, you can check single statements if they do prefetching if you look at the access plan of the statements (as described in “Understanding statement plan” on page 573).

     One type of prefetching is sequential prefetching, which is chosen when pages on disk are expected to be in the physically needed order (also referred to as cluster ratio). Sequential prefetching is an option for table scans. In the explain output, look for nodes of type TBSCAN and double-click them. In the *Input arguments* box, check the *Prefetch* argument to see if sequential prefetching is being performed for that table scan.

Another type of prefetching is list prefetching, which is chosen when pages on disk are not in the required order but the table is accessed through the use of an index. For list prefetching, the RIDs that are retrieved from the index are sorted, and then the list of pages that they point to is prefetched. In the output, look for chains of nodes of type IXSCAN, SORT, RIDSCN, and FETCH (Figure C-36). Double-click the FETCH node. In the *Input arguments* box, check the *Prefetch* argument to see if list prefetching is being performed for that table scan.

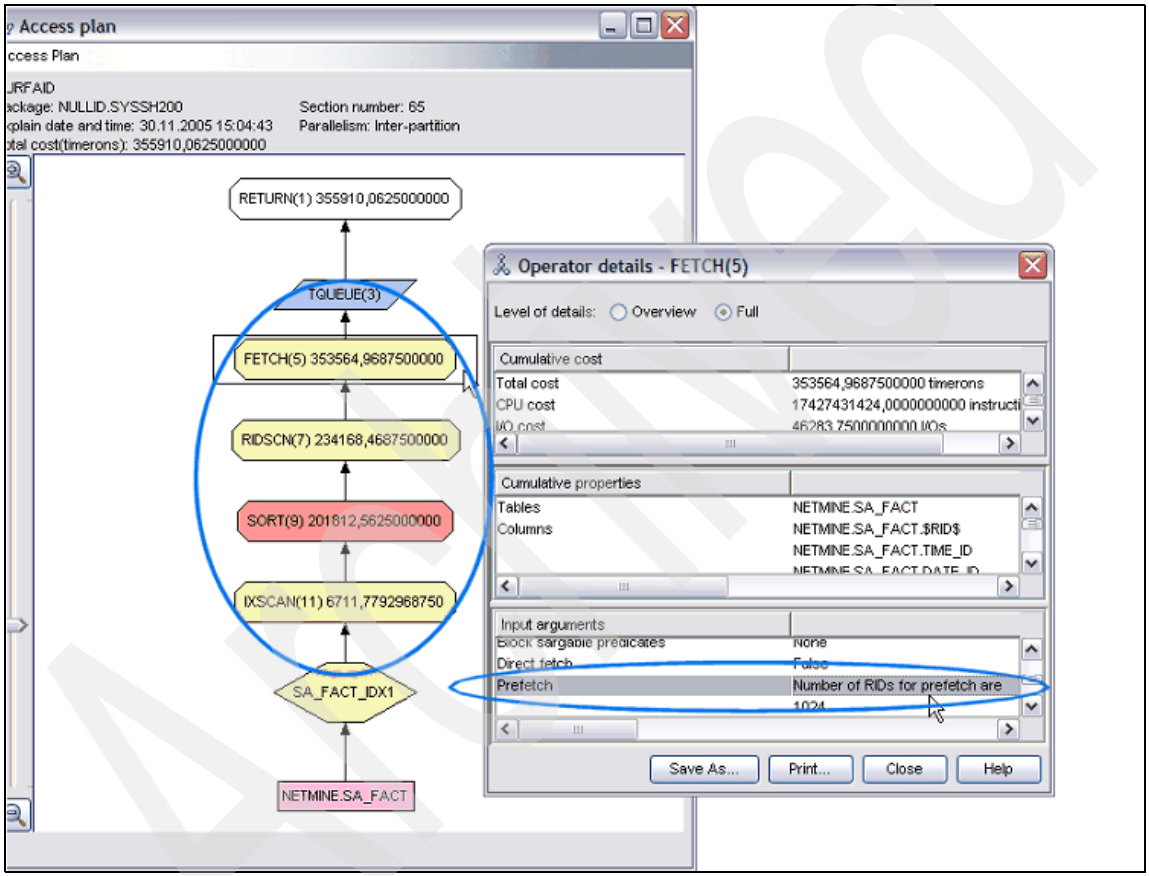


Figure C-36 Access plan

Sequential prefetching is the preferred way of performing table scans. You can support DB2 choosing and performing sequential prefetching by ensuring a good cluster ratio on the columns (the table is sorted right after a table scan). This is achieved by maintaining an index on the same columns in the same sort order and declaring it as the CLUSTERING

INDEX. If you have created a clustering index on an existing table, you should run **reorg** and **runstats** on that table after you create the index.

If you determine that DB2 does a lot of prefetching (by evaluating the metrics in the Access pane), you should determine if your system is I/O bound by looking at the I/O wait time indicator (as described in “Check if a system is CPU bound” on page 545). If this is the case, try to increase the PREFETCHSIZE parameter for the tablespaces where prefetching is done. But be aware that prefetching might not have been the cause of the I/O waits.

Increasing the PREFETCHSIZE too much leads to wasted I/O operations, which will be visible as a lower buffer pool hit ratio. So as you increase PREFETCHSIZE, also pay attention to the hit ratio. As soon as it goes down, you might have increased PREFETCHSIZE too much. If you are monitoring on a buffer pool level, you also have the counter available for *Unread prefetch pages*, which provides direct information about prefetching effectiveness.

– Block-based buffer pools

A further step to tune sequential prefetching allows the pages in the buffer pool to be in the same order as they are on disk, which allows DB2 to use single I/O operations to read entire sequences of contiguous pages into the buffer pool. To enable this function, you must create the buffer pool with the option to reserve a fraction of the entire pool for block-based I/O.

You can monitor how effective this option is set up in the buffer pool details in the Access pane of the Statistics Details window (Figure C-37 on page 560). If you have set up a block-based buffer pool and you see that the *Number of block based IO requests* field contains a very low number or almost the same as is in the *Number of vectored IO requests* field, you should consider changing the block size parameter of the buffer pool. The block size should be aligned with the extent size of the tablespaces. If this is not the case, you might waste space in the block area of the buffer pool. If the value of the *Total number of pages read by block IO* field divided by the value of the *Number of block IO requests* field is much less than the defined block size, your block size might be too high with regard to the extent size of the tablespaces.

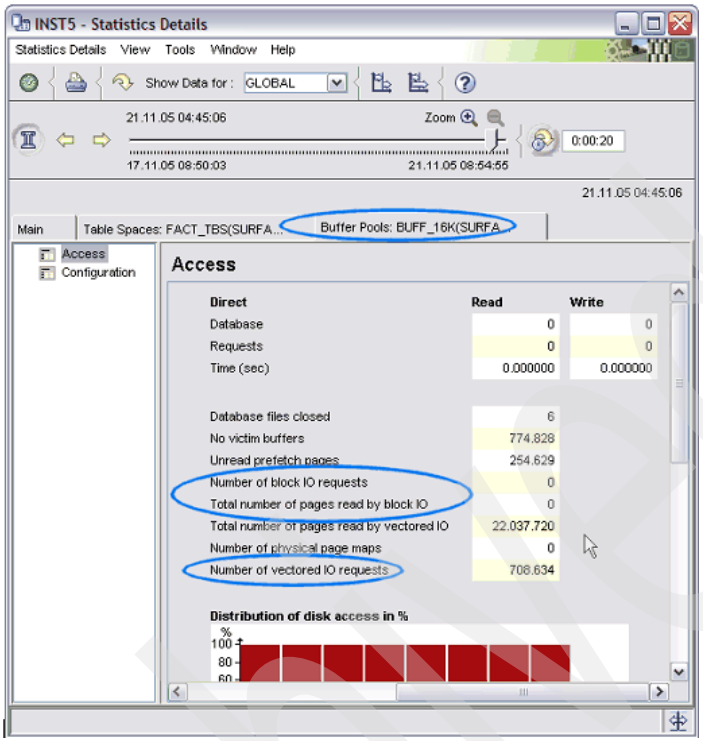


Figure C-37 Monitoring buffer pools

– Read efficiency

This important I/O-related performance indicator has already been described to some extent in “Engine monitoring” on page 529. If you encounter many read rows, but see only a few selected rows, you should revise your queries and determine if you are doing many table scans, which could be avoided by an appropriate index. Also, you might consider using MQTs for frequently joined tables. See “Understanding statement plan” on page 573 for details about the execution plan.

Detecting skews

There are two basic approaches to detect skews with PE:

- You can set up a visualization dashboard in the System Health window for performance indicators for which you want to know the skew on the partitions.
- You can use the table displays of the group views in the data panels of PE.

Visualized skew detection

Figure C-38 is an example of a dashboard that has been set up in System Health to identify skews on sort behavior. See “General approach” on page 546 for details about setting up data views.

Another interesting way to visually recognize data skew is available in the statistics details pane described in “Data skew” on page 563.

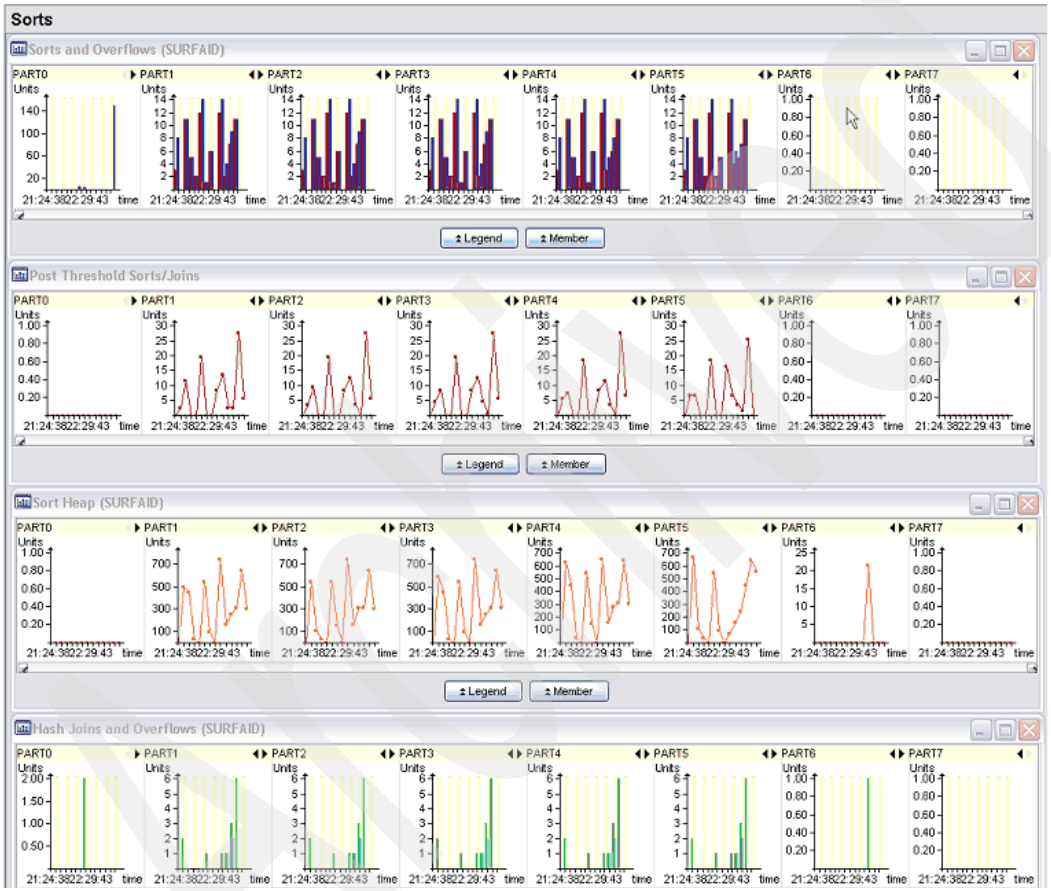


Figure C-38 System Health dashboard

Buffer pool hit ratio skew

The buffer pool hit ratio can be evaluated on different levels:

- Database
- Tablespace
- Buffer pool
- Application

For all these levels, PE also facilities checking for skews in the buffer pool hit ratio.

To check the buffer pool hit ratio, open Statistic Details, switch to group view and select the single panes for Database, Table Spaces, and Buffer Pools to verify the hit ratio distribution over the partitions (Figure C-39).

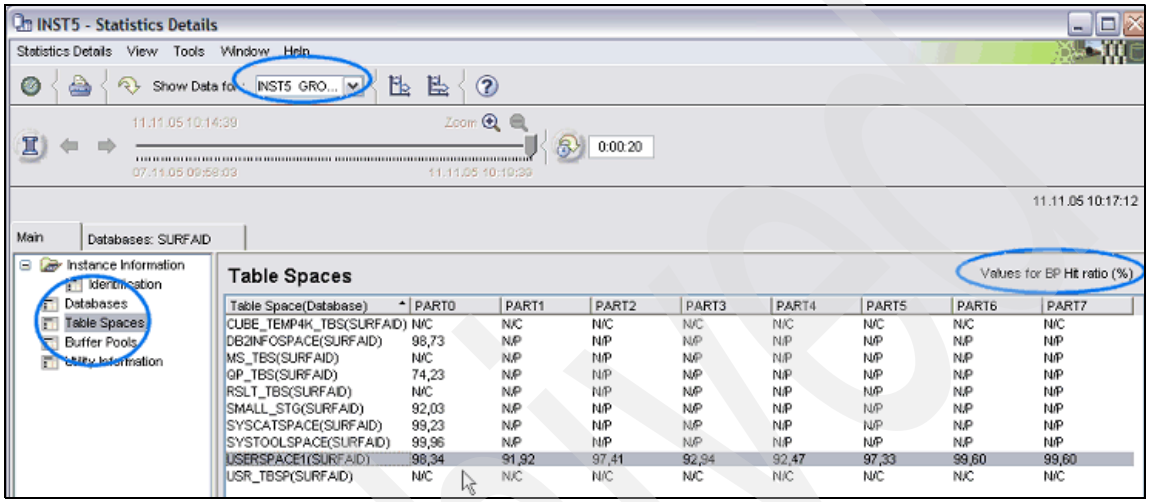


Figure C-39 Checking buffer pool hit ratio

If you are interested in hit ratio values divided by index and data, you can find them by drilling down to the database or tablespace details respectively.

To check for the hit ratio distribution on application level, open the Application Summary window, switch to group view, and sort the *Application Handle* column to see application details for each partition and grouped together per application. Then check the *Hit ratio (%)* column for uneven distribution of the hit ratio (Figure C-40 on page 563).

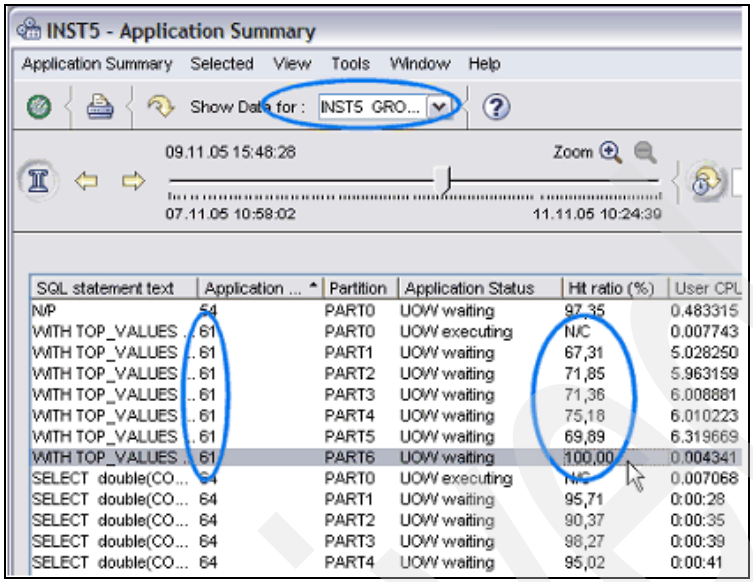


Figure C-40 BP hit ratio distribution on application level

Data skew

PE provides means to check for data distribution at the database and tablespace level. Open the Statistics Details window, select the Table Spaces pane, and double-click the tablespace that you are interested in. Switch to group view and select the Space Management pane. A table is displayed that contains the allocated and free space of the selected tablespace on each single partition (Figure C-41 on page 564).

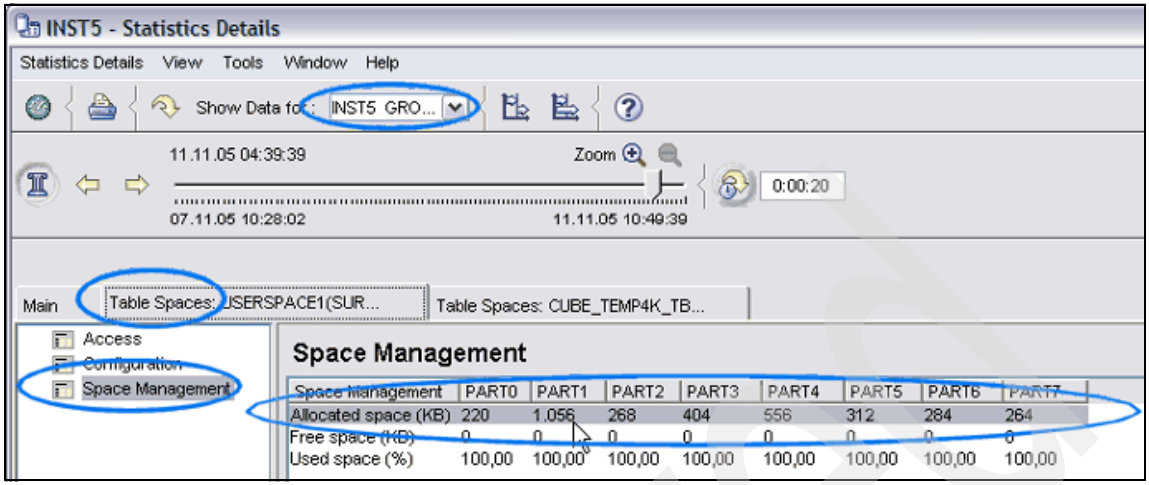


Figure C-41 Checking data distribution

There is also a visual way to check for data skews on tablespace level. Open the Table spaces - Space Management pane and switch to global view. A pie chart is displayed that visualizes the data distribution of the tablespace in terms of used pages (Figure C-42 on page 565).

Note that this pie chart is available only in online monitoring. Due to technical reasons, it is not available in history mode.

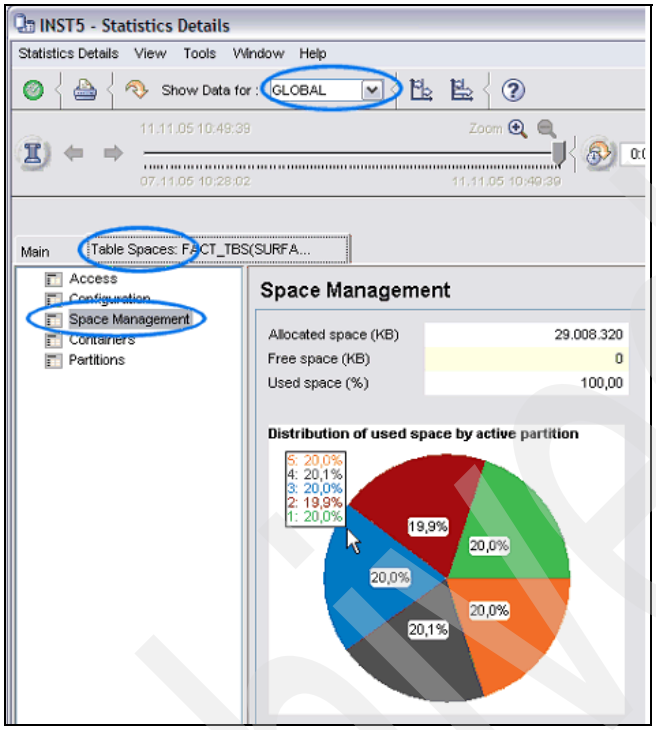


Figure C-42 View data skew in pie chart

Other skews

The group view feature of PE provides ways to look for skews on many more performance indicators. As an example, Figure C-43 shows how to check for skews on DB2 I/O data, such as rows read and rows selected.

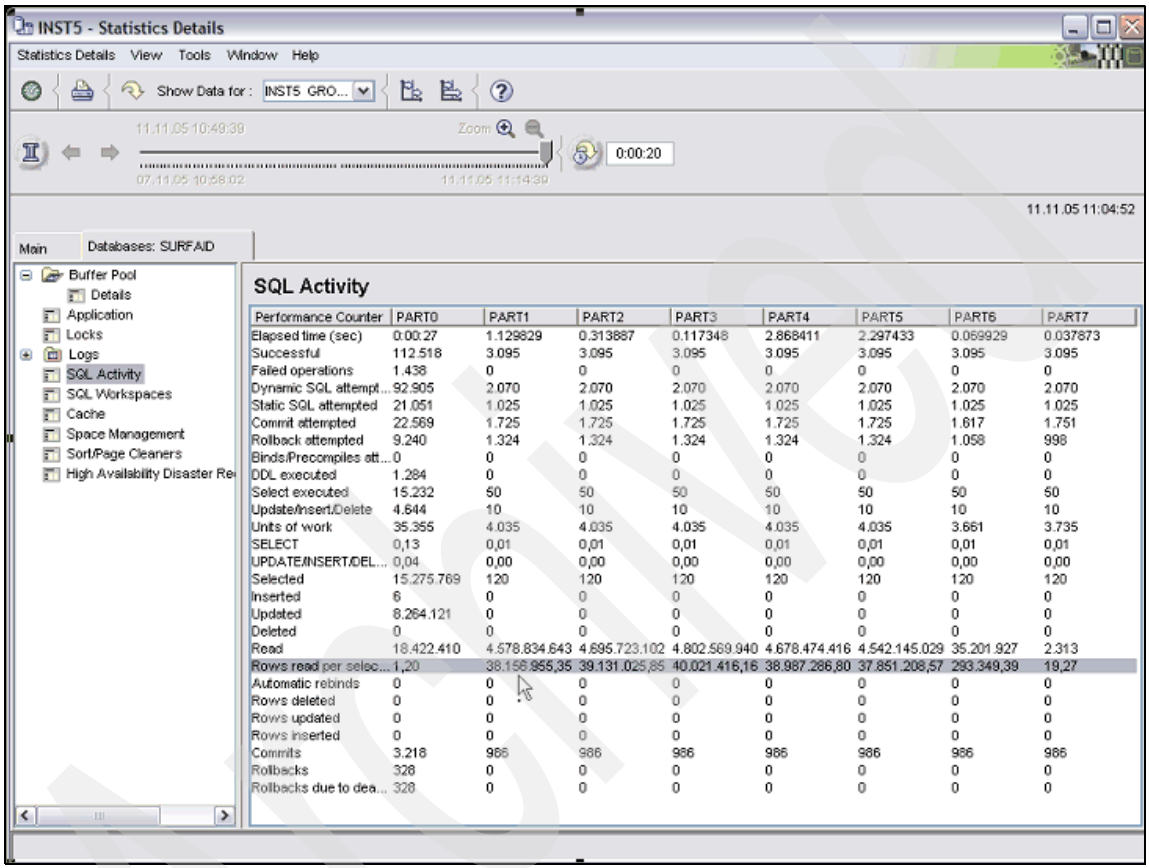


Figure C-43 Checking skews on DB2 I/O data

In this example figure, two partitions deviate from the average rows read per selected number. Partition 0 returns almost each row read because it is the coordinator partition in this particular scenario. Partition 6, however, has a multiple of the average rate and might need some attention.

You can look for skews on the other important performance indicators by opening the associated panes and switching to group view.

Understanding long-running queries

In this section, we discuss how to monitor long-running queries.

Identifying the top 10 statements

Whether you want to tune you BI system or you want to find the source for poor performance in general, you need a starting point. A good way to start is to find the top 10 statements with regard to execution time. This can easily be achieved by opening the Statistics Details window and accessing the Dynamic SQL Statements pane. Click the **Receive statement cache information** button at the bottom of the pane, and PE will request and display this data. Now sort the **Avg. time per execution** column. You might also want to use other columns to find the top 10 statements, such as the *Executions* and *Elapsed time* columns (Figure C-44 on page 568).

You can also find the top 10 statements by examining the performance history. To do so, switch to history mode and scroll to the point in time for which you want to perform this check.

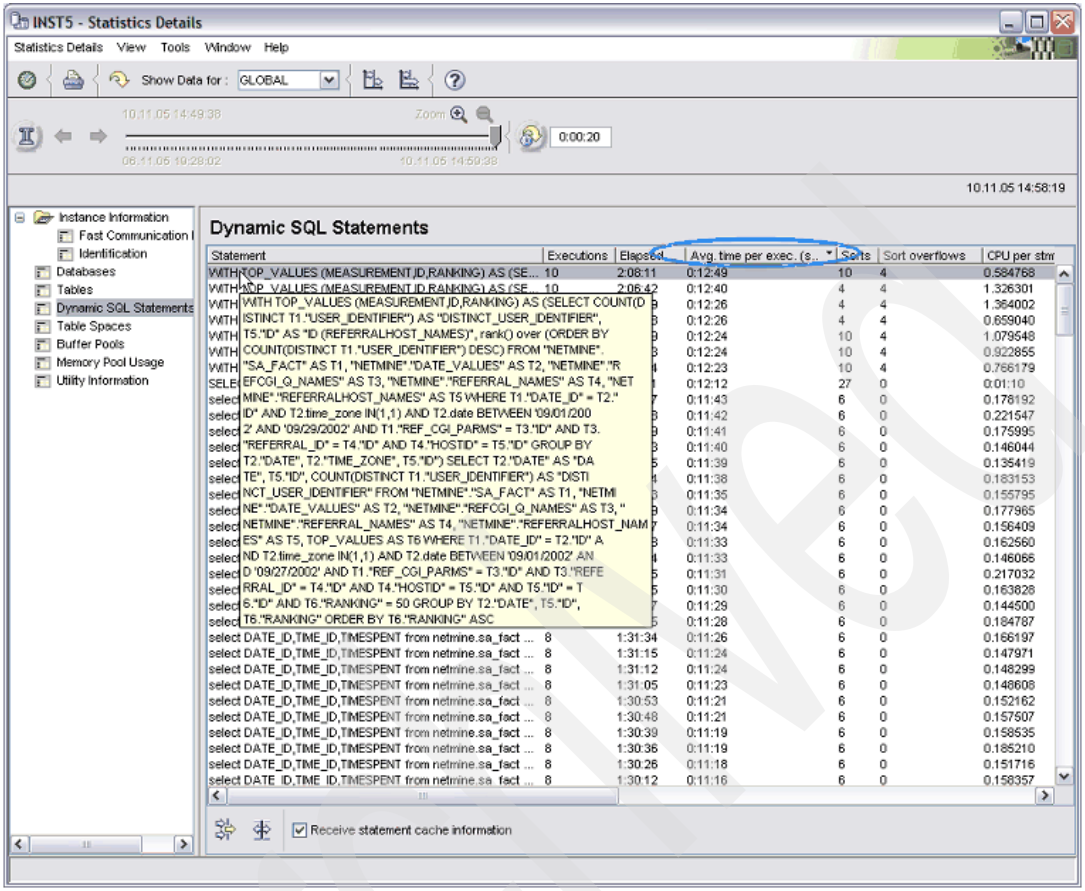


Figure C-44 Identify the top 10 statements

Linking top 10 statements to execution details

The dynamic SQL statement information shown in the previous section represents aggregated data over all executions of these statements. This means that it is not directly possible to drill down to the applications (as displayed in the application summary) that executed these statements.

But you can leverage the Query facility in Performance Warehouse to define and execute a query that shows you all entries when one of the top 10 statements has been captured in the application summary. To do so:

1. Open the Performance Warehouse - Analysis window, go to Query groups, and create a private group, if one does not already exist.

2. Right-click the group and select **Create**. Give the new query a name and description and paste the query shown in Example C-1 into the appropriate entry field on the Definition pane.

Example: C-1 SQL to shows entries on the top 10 statements

```
WITH LAST(last_ts) AS
(SELECT     MAX(HT_TIMESTAMP)
 FROM       DB2PM.HISTORYTOC
 WHERE      HT_DATA = 'DYNAMICSTATEMENTCACHE'
), TOP10(statement, ms_per_execution, top) AS
(SELECT     DISTINCT sql.STMT_TEXT, sql.ATIMEP_EXECUTIONS, ROW_NUMBER( ) over (
 ORDER BY   ATIMEP_EXECUTIONS DESC
                                                                          )
            AS TOP
 FROM       DB2PM.DYNSQL sql, LAST
 WHERE      sql.INTERVAL_TO = LAST.last_ts
            AND MEMBER_ID = - 2
 ORDER BY   ATIMEP_EXECUTIONS DESC
FETCH     FIRST 10 ROWS ONLY
) SELECT   TOP10.top top_stmt_number, MAX(applstmt.INTERVAL_TO)
last_captured_at
           , applstmt.AGENT_ID application_handle, applstmt.STMT_START
           statement_started_at, applstmt.STMT_TEXT text
FROM       DB2PM.STATEMENT applstmt, TOP10
WHERE      applstmt.STMT_TEXT = TOP10.statement
           AND MEMBER_ID = - 2
GROUP BY   applstmt.STMT_START, applstmt.STMT_TEXT, applstmt.AGENT_ID,
           top10.statement, top10.top
ORDER BY   top10.top
```

3. Click **OK** to save the query.

4. Right-click the query and select **Execute**. In the menu that opens, click **Execute** again (Figure C-45). Depending on how much data you have recorded in the history, this query might run for several minutes.

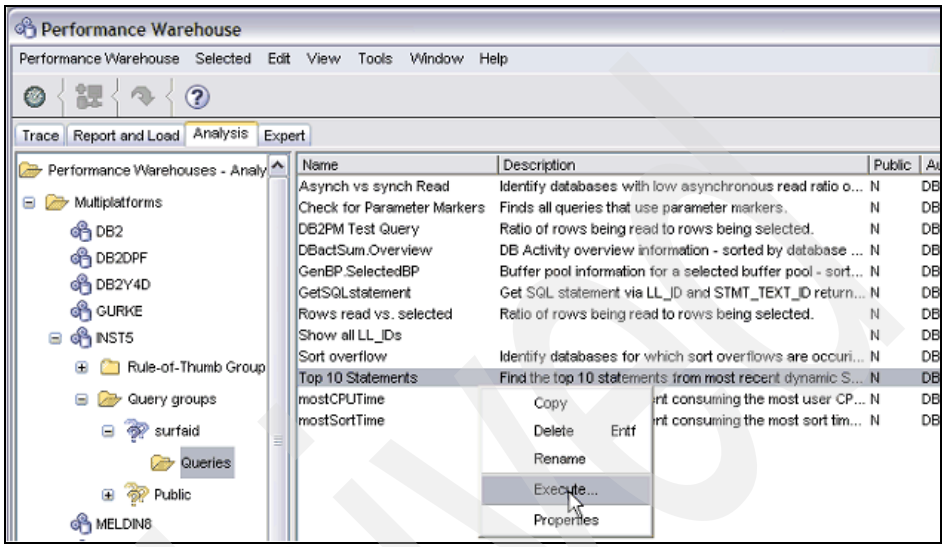


Figure C-45 Execute SQL saved in PWH

When the query finishes running, the results shown in Figure C-46 on page 571 are displayed. These results show one row per point in time when an application executed one of the top 10 statements.

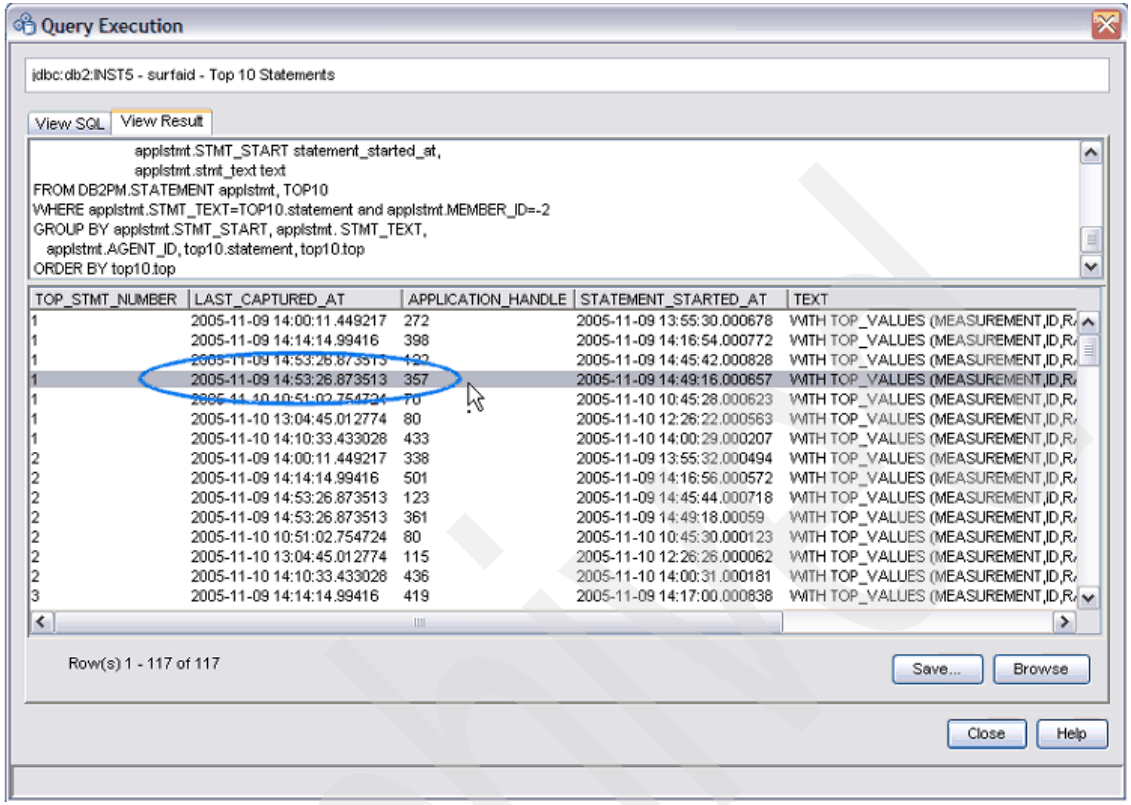


Figure C-46 View SQL execution result

5. Select one of these entries, note the point in time, and open the Application Summary window. Switch to history mode and scroll to the same point in time. Look for the same application handle as in the selected entry of the query results. Double-click the entry to display the details about the application and you are done (Figure C-47).

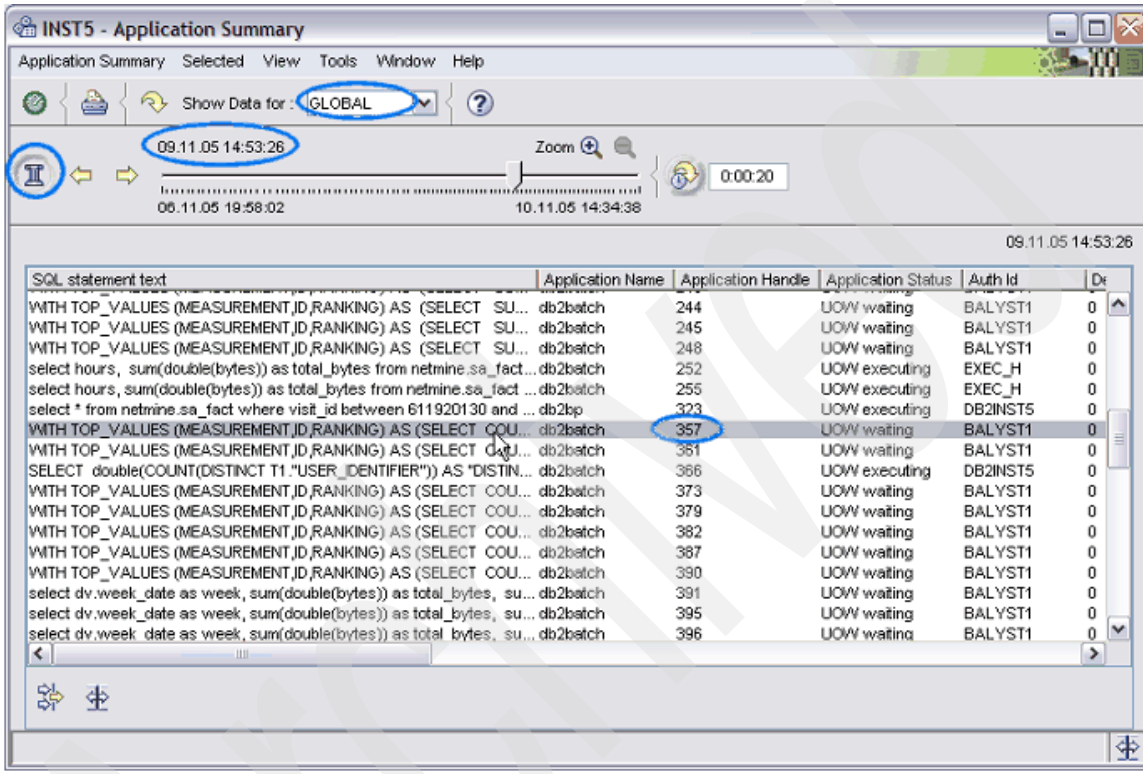


Figure C-47 View the application details

Checking for skews per query execution

In the Application Summary window, you can check for different types of skews regarding query executions.

1. Switch to group view and sort the **Application handle** column.
2. Scroll to the application that you are interested in. You will see one row per partition involved in the query execution for the current statement.
3. Check the columns of important performance indicators, such as user CPU time, system CPU time, sort overflows, and hit ratio for skews over the different partitions (Figure C-48 on page 573).

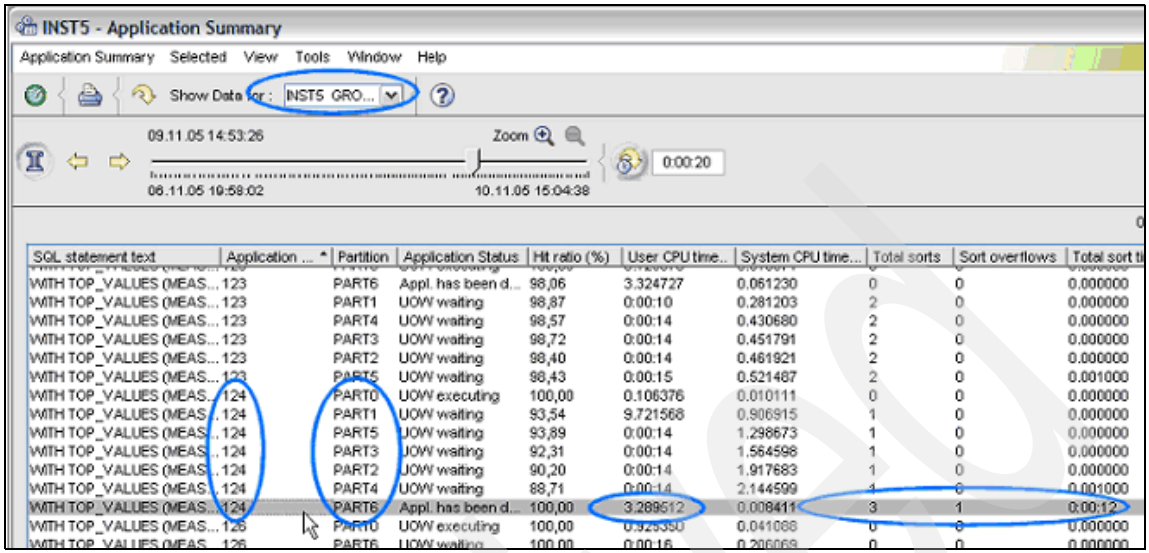


Figure C-48 Checking for skews per query execution

If you have a higher number of partitions, you might want to use the filter function to limit the table content to just the application handle that you are interested in. You can then sort more freely over the columns with performance indicators that you want to check for skews.

Understanding statement plan

After you have found an entry of interest in the Application Summary window you can drill-down into details by double-clicking the entry. In addition to other details, you will see information about the *SQL statement and package* in the associated category on the left most pane. From here you can also explain the statement by clicking **Explain** (Figure C-49 on page 574).

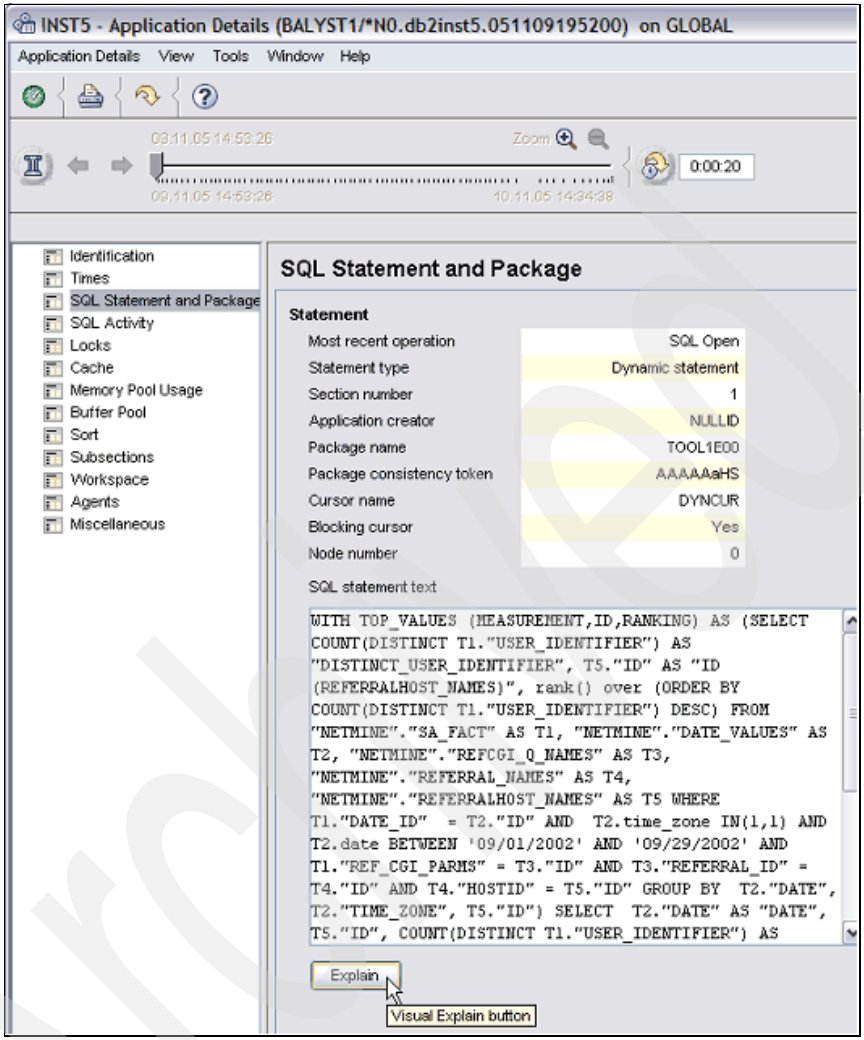


Figure C-49 SQL statement plan

Clicking **Explain** opens the Visual Explain tool (as known from the DB2 GUI tools), which you can use to explore the plan details (Figure C-50 on page 575). The numeric values in each box represent the estimated effort for that operation in an imaginary unit called *timerons*. You can identify the hot spots of the plan by looking for the big increases of those numbers that are higher in the tree. Boxes that are higher in the hierarchy include the aggregated timerons on their direct children boxes (that is, if the number of a parent box is just slightly more than the sum of the numbers in all children boxes, the operation that is represented by that parent box is very cheap).

For general information about interpreting the output, please refer to the DB2 documentation.

Note that the same help for PE is also available directly from the Dynamic SQL Statements pane of the Statistics Details window after you have opened the details pane for a selected statement.

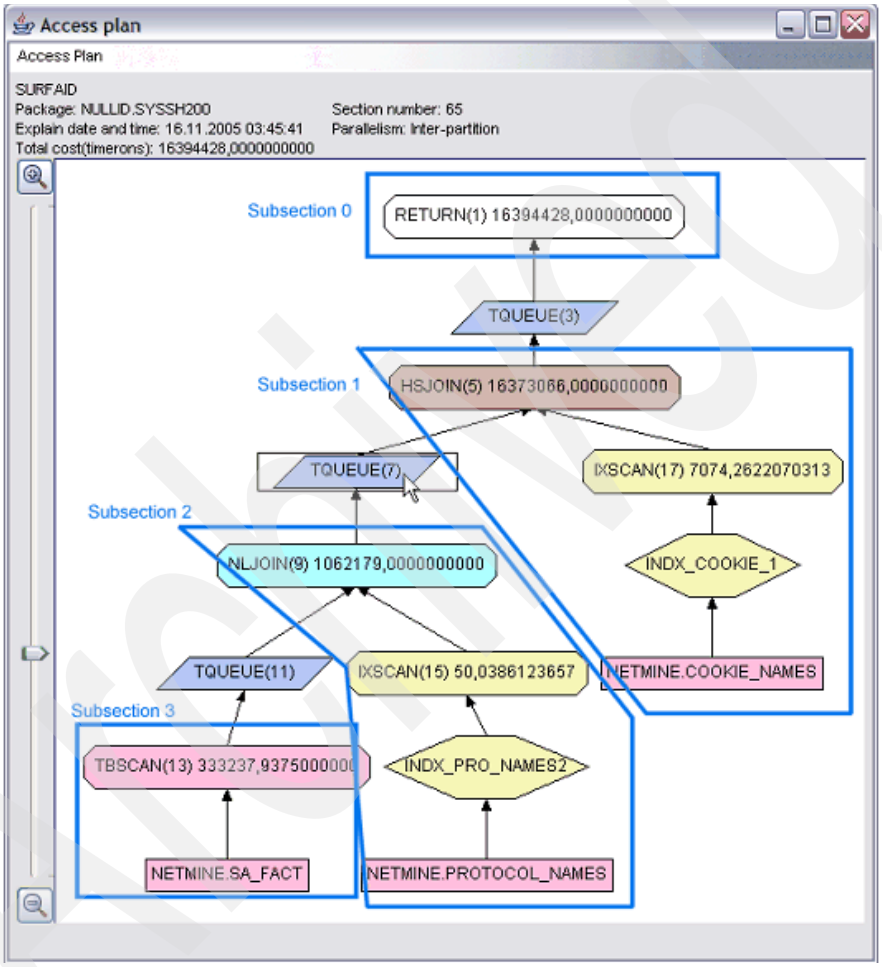


Figure C-50 Access plan

In DPF environments, a statement plan is broken down into subsections. Each subsection is executed in a dedicated agent; potentially, the same subsection is executed many times in parallel by multiple partitions. The results of one subsection are the input data for the following subsection. The data between subsections is exchanged through the use of table queues. A table queue can be imagined as a never-materialized temporary table within a communication buffer between partitions. Visual Explain allows you to detect the different subsections. To do so, look for table queues. Each part of the plan tree between table queues is a single subsection. Figure C-50 on page 575 shows a statement with four subsections outlined in blue boxes.

An interesting component of the plan is the table queue (TQUEUE), of which there are different types. To determine the type, double-click a table queue box and look at the *Table queue send type* (Figure C-51).

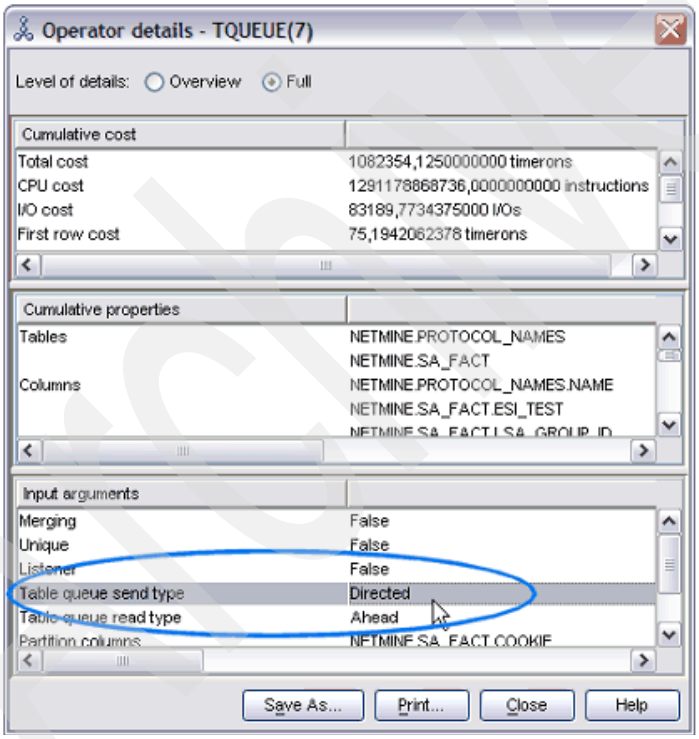


Figure C-51 Operator details

You should look for table queues send types of Directed because this type might impose a potential performance issue. They are usually inevitable if two larger tables need to be joined over columns where at least one of them is not part of a partition key. As a consequence, one or even both of the tables must first be

scanned and dynamically hashed on the join column. Then the hashed data is sent via table queues to the according partition where the actual join is performed. This operation results in a lot of data being transferred over the wire if the tables are large.

Requirement: Even if both tables have the join column in the partition key, a directed table queue is required if the two tables have different partition maps (that is, if they are stored in tablespaces of different node groups).

If you detect directed table queues, you might want to reconsider the partition keys or the distribution of tables to node groups, or you might want to revise the queries. The optimum goal to strive for is collocated joins between the tables. You can recognize them by using Visual Explain to look for joins between tables that do not involve a table queue. A common approach to working with different partition keys of tables that you need to join is to create and maintain an MQT on one of the tables with a different partition key than the other joined table. The optimizer will then decide to rewrite queries to these tables to a join that accesses the MQT instead of the base table, which results in a collocated join.

If you detect broadcast table queues for small dimension tables, consider creating replicated MQTs on them (CREATE TABLE … AS … REPLICATED), which means that they physically exist redundantly on each partition and which allows an optimizer to choose collocated joins when these tables are joined (for example, with the fact table).

Monitoring subsections

In addition to the approach to that is described in “Checking for skews per query execution” on page 572, you can also perform a detailed analysis of the execution per subsection by using the Subsections pane of the Application Details window. The table displays one entry per subsection per partition.

In the following example (which represents the same query that is presented in the Visual Explain example in “Understanding statement plan” on page 573), you can see that subsection 3 is executed in parallel by partitions one through five. Within these five entries, you can look for skews in the areas of CPU, rows read/written, and table queue length. The latter indicates how much data is read and written from/to a table queue (as shown in the Number of Rows Read from Tablequeues and Number of Rows Written to Tablequeues columns). If there are significant deviations for the same subsection (for example, partition 5 of subsection 1 in Figure C-52 on page 578) a partition might be over-utilized. The reason could be data skew.

Generally, high numbers of table queue lengths are an indicator for directed table queues that might need your attention (as described in “Understanding statement plan” on page 573).

The same properties can also be used to understand more about the selectivity of the subsections. For example, subsection 2 (nested loop join with table PROTOCOL_NAMES) has a selectivity of 100% because it reads the same number of rows from its input table queue as it writes to its output table queue. On the other side, subsection 1 (hash join with COOKIE_NAMES) has a very low selectivity so far (which might change because the query is still executing and subsection 1 is still waiting to send data via the table queue to subsection 0). See Figure C-52 and Figure C-53.

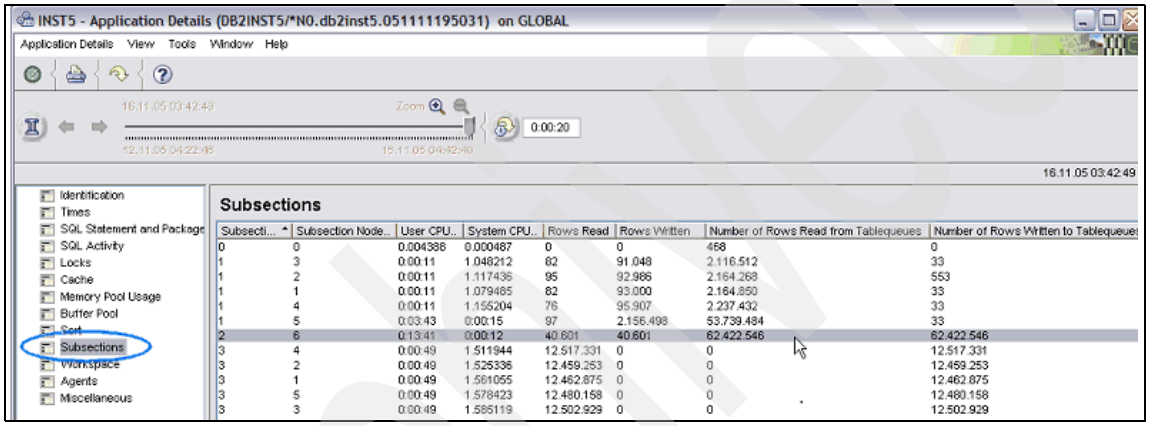


Figure C-52 Monitoring subsections

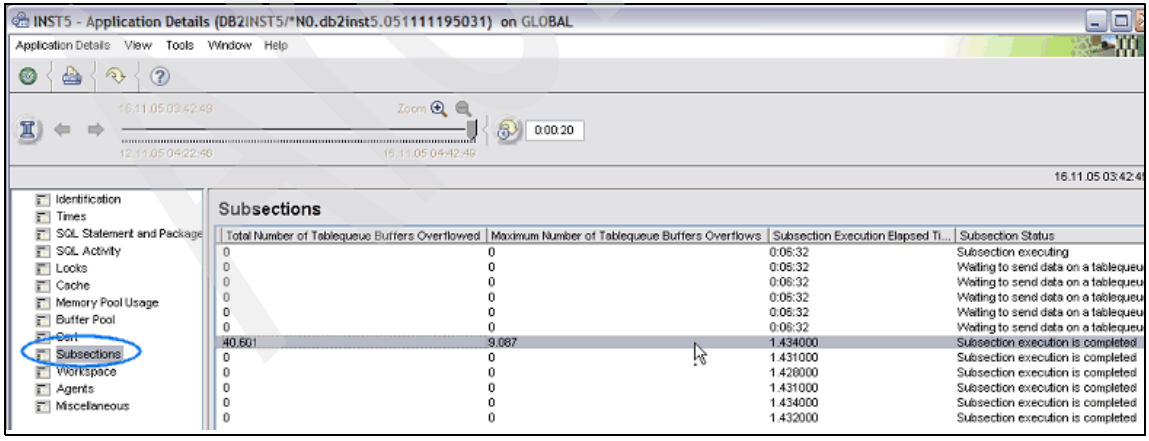


Figure C-53 Monitoring subsections continue

By scrolling horizontally, you can check for other properties. The Subsection Status column indicates the current state of query execution. In the previous figure, you can see that subsections 3 and 2 are already done. Subsection 1 is actually done as well, but still needs to send some data to subsection 0, which is the coordinator subsection and which is currently executing (that is, receiving data).

The *Total Number of Tablequeue Buffers Overflowed* and *Maximum Number of Tablequeue Buffers Overflows* columns indicate if data had to be written to temporary tablespace. This can occur if there are multiple partitions reading for the same table queue and some of them are not fast enough to read the data. The sending subsection is then spilling the data for the slower receivers to disk so that it can continue to send the data to the other receivers.

Check for SORT issues

A general indicator that large SORTs are happening and might be the cause for a slow query execution is a high rows written rate. Even though the statement is just a SELECT, rows are written to disk if a SORT or HASH JOIN cannot be done in memory. Figure C-54 on page 580 shows a rather high number of rows being written to disk, which is indicated in the application details.

Note: You can also check for Rows Written in dynamic SQL table in the Statistics Details window. Here you can likely sort this column to see the statements that are writing to disk the most.

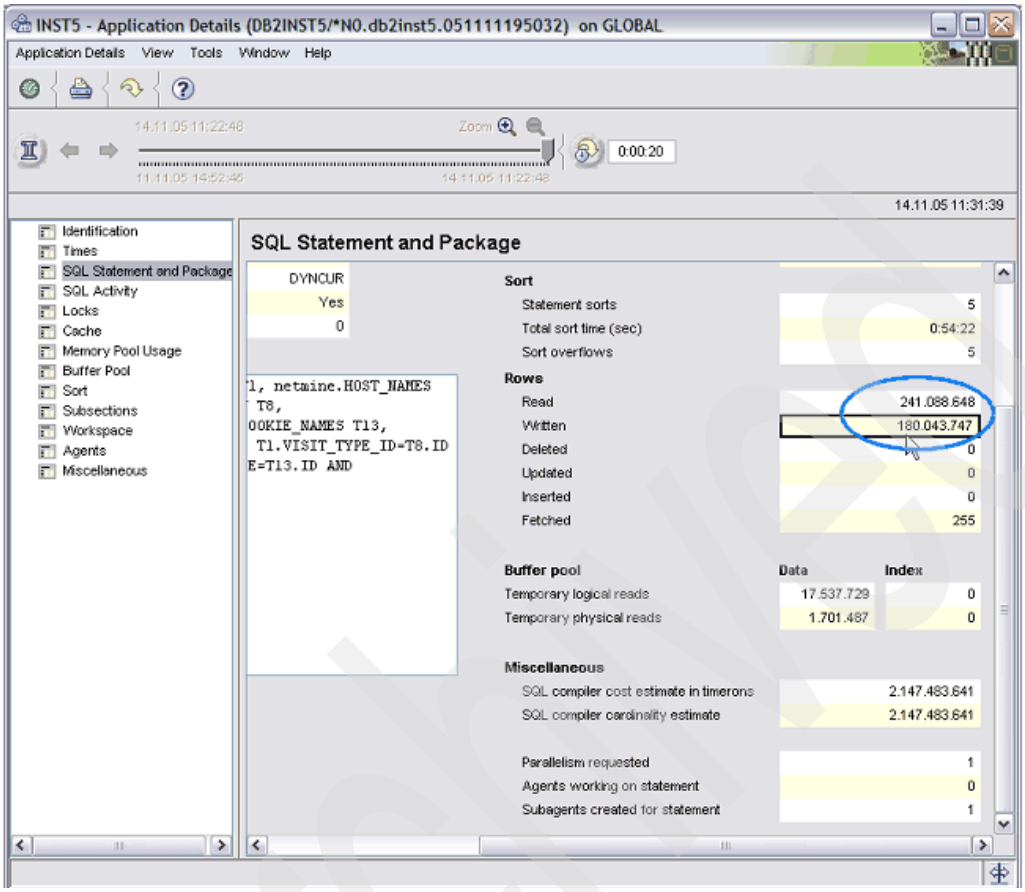


Figure C-54 Rows written to disk

To verify your suspicion of a sort problem, you should determine if the application that is executing this statement has many sort or hash join overflows. You can find this information in the Sort category of the Application Details window (Figure C-55 on page 581).

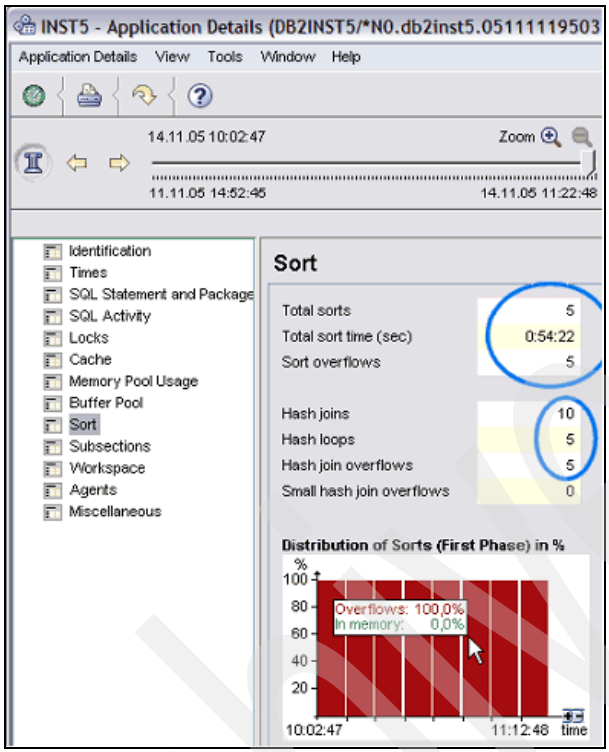


Figure C-55 Sort details in Application Details

If you are interested in sort indicators (for example, if you want to find the top statements with regard to sort issues), you can also use the Dynamic SQL Statements pane of the Statistics Details window and sort the **Sort overflows** or **Rows written** columns. See “Identifying the top 10 statements” on page 567 for more information.

Canceling long-running queries

If you identify a long-running query that is having a severe negative impact on the entire performance of your system, and you do not have access to or influence on the person or application that has issued this query, you can use PE to cancel this query. The associated application will be rolled back and closed immediately.

To do so, open the Application Summary window in global view, right-click the application that issued the query, and click **Force Application**. Then either enter the user ID (and password) that started that application or a user ID with at least SYSMAINT authority on the monitored instance (Figure C-56).

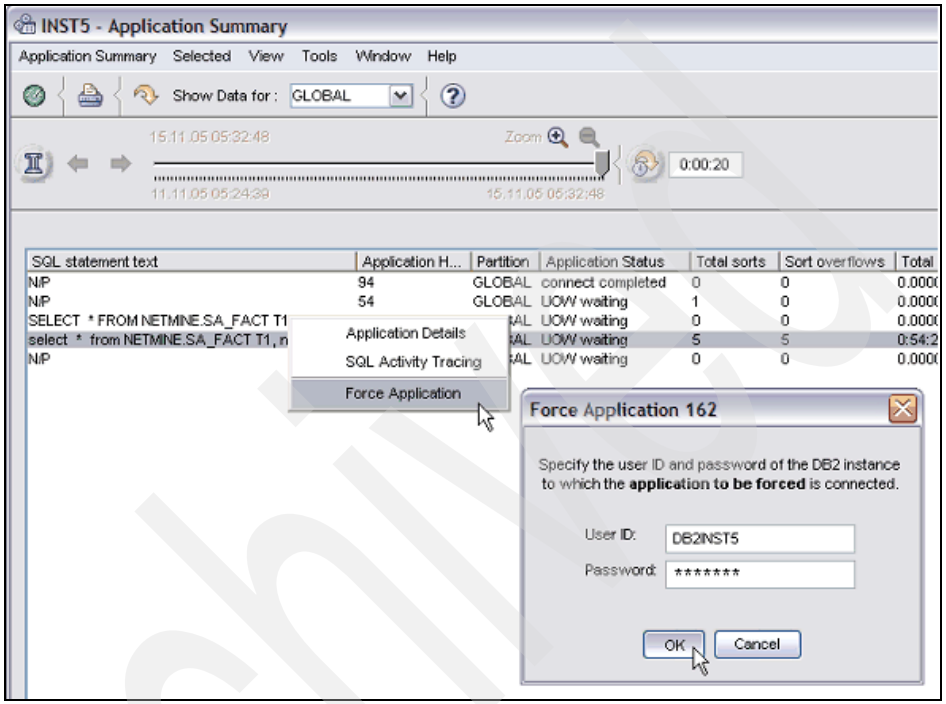


Figure C-56 Cancelling long-running queries

SQL tracing

Sometimes you need to perform a complete SQL trace of an application or, in some cases, entire BI systems in order to better understand a performance problem. PE provides means to run *ad-hoc* SQL traces for a selected application, as well as to schedule and perform complete traces for an entire BI system. PE offers a set of features to help you analyze the data that is collected in these SQL traces. These features include the ability to generate SQL Activity HTML reports and a set of predefined analyzing queries.

The simplest way to do *ad-hoc* tracing of a current application is to open the Application Summary window in partition or group view, right-click the application that you want to trace, and select **SQL Activity Tracing**. In the SQL Activity Report Generation dialog that opens, specify how long the trace should run, if it

should also capture static SQL (in case your application uses it), and if the data should be deleted from the PE database after the report has been created.

Now let the application perform the action that you want to trace and click **OK** in this dialog to start the trace. (Figure C-57).

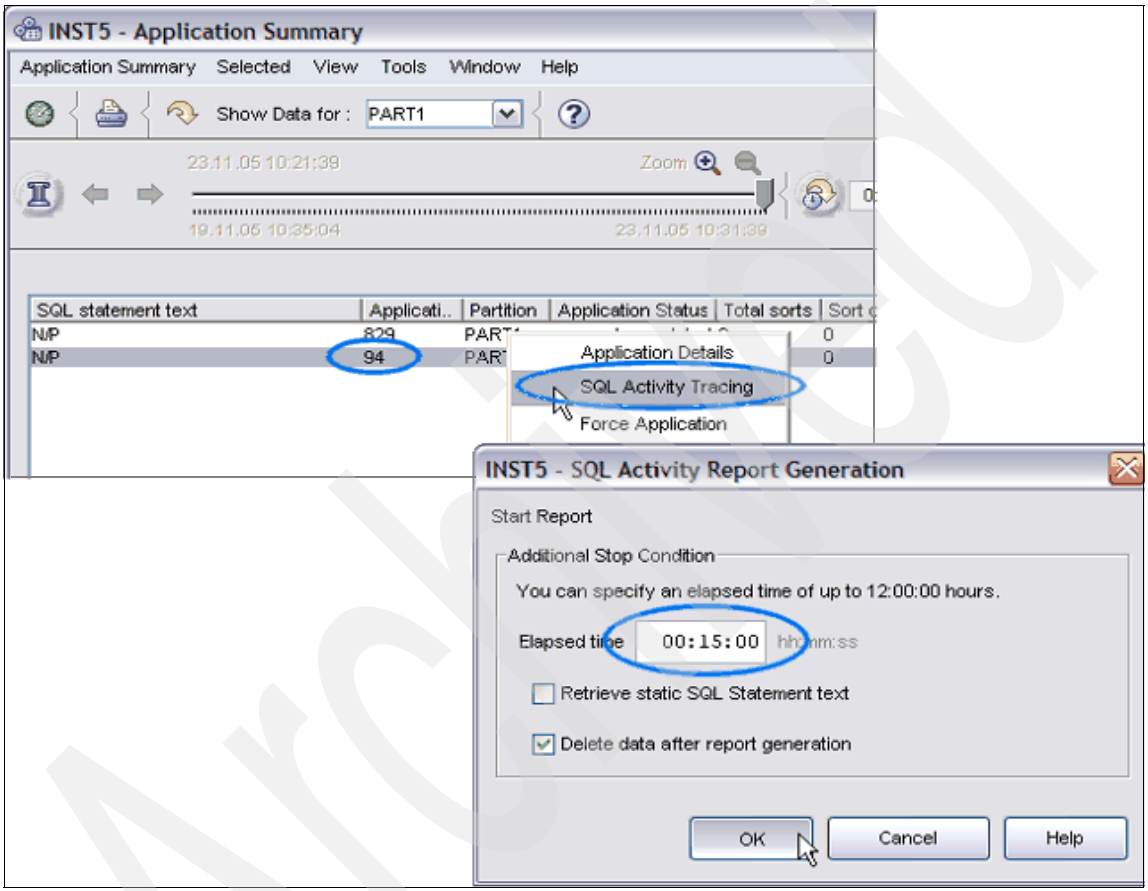


Figure C-57 SQL tracing

After the specified tracing time is reached and after some internal processing of the data, the results are displayed in an HTML report that outlines all of the SQL statements of the selected application and their execution metrics during the trace time.

If you do not have a specific application that you want to trace, but rather want to trace the entire SQL workload, do this by scheduling a process in the PE Performance Warehouse dialog. The PE Performance Warehouse dialog offers additional options for the data that you collect. For example, in addition to generating reports, you can also just collect the SQL activity trace and then run some analytic queries or rules of thumb against it.

To generate an SQL activity report in Performance Warehouse:

1. Double-click **Performance Warehouse - Expert** in the System Overview window, navigate to your instance, and expand the Process Groups folder.
2. If you have only the predefined Public group, create a custom group by selecting **Create** from the context menu and providing a name for the group.
3. Right-click **Public** → **Processes** → **DB2PM.Template SQL Activity Summary Report** and select **Copy** from the menu that opens (Figure C-58).
4. Select the previously created new process group and click **OK**.

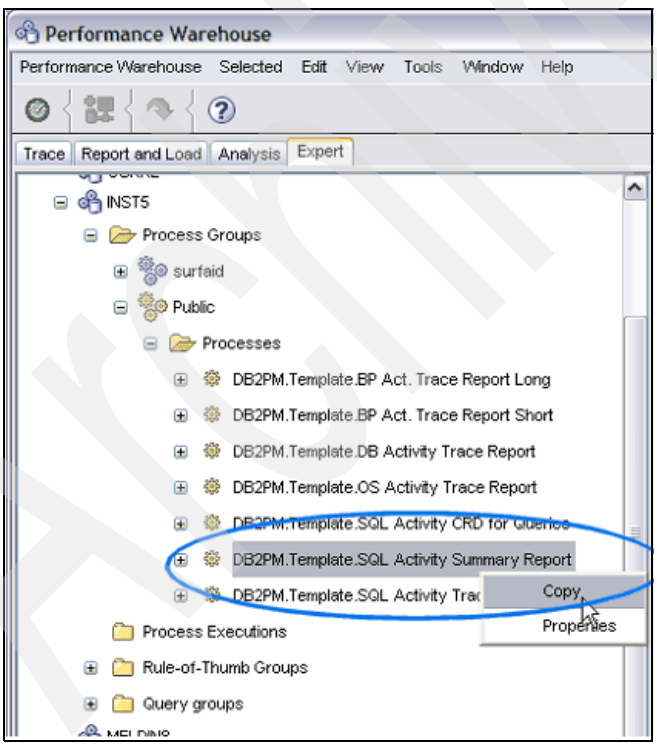


Figure C-58 Define the process

5. Navigate to the previously created process group, right-click **Processes** → **SQL Activity Summary Report** and select **Execute…** from the menu that

opens. In the dialog that opens, select whether you want to start the trace immediately (similar to the option that was described above for a single application), to run it at a scheduled time, or to set up a schedule if you want to have the trace collected periodically (Figure C-59).

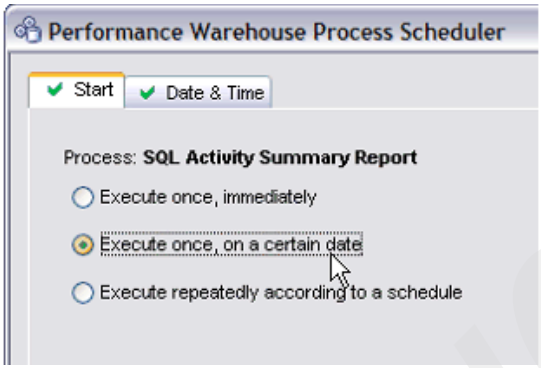


Figure C-59 Execute the Activity Summary report

Monitoring and tuning load

Not only do BI systems need to handle analyzing the workload, they also need to accommodate the periodical incremental loading of new data into the tables. The load actually consists of three steps. First, there is the actual load, import, or plain insert of new data to the tables. Then the MQTs need to be refreshed. Finally, the statistics need to be updated. The data might first be loaded to staging tables, followed by an INSERT…SELECT statement that moves the data from the staging table to the actually queried table.

PE provides dedicated means to monitor load processes. To do so:

1. From the Statistic Details window, select the Utility Information pane and display it in global view (Figure C-60). If the load is currently running, you will see one entry per partition that the load affects, as shown in Figure C-60. If there were multiple load processes running at the same time, you can distinguish them by examining the *Utility ID* column.

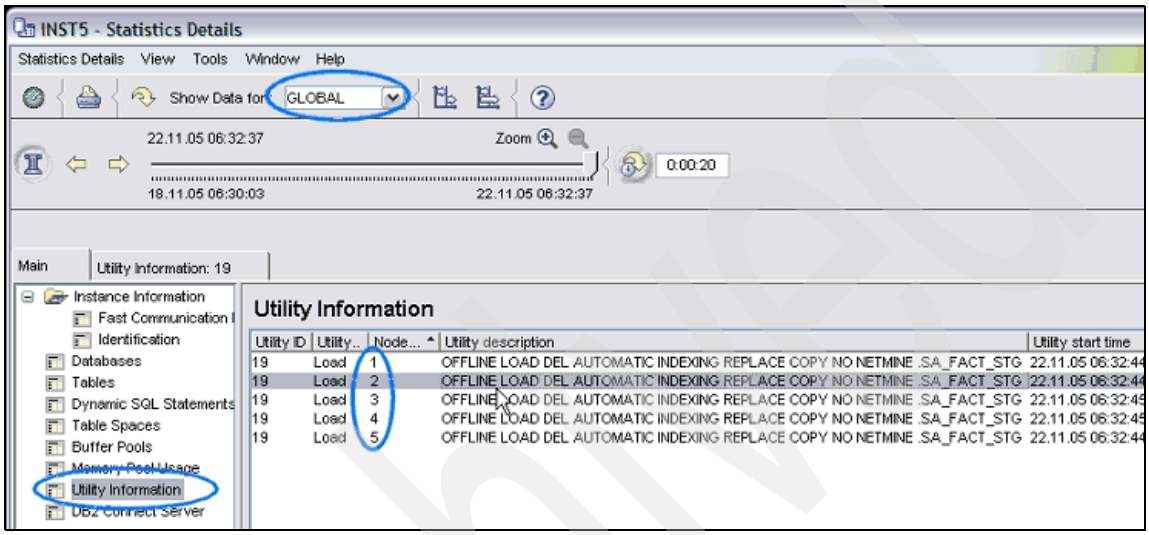


Figure C-60 Utility information

2. Double-click any of the entries to display details about the current status of the selected load process on each partition. As shown in Figure C-61 on page 587, you can track the progress of the load steps on the single partitions by looking at the *Progress percentage complete* column for the different nodes. Use the manual or auto-refresh option to track the progress over time.

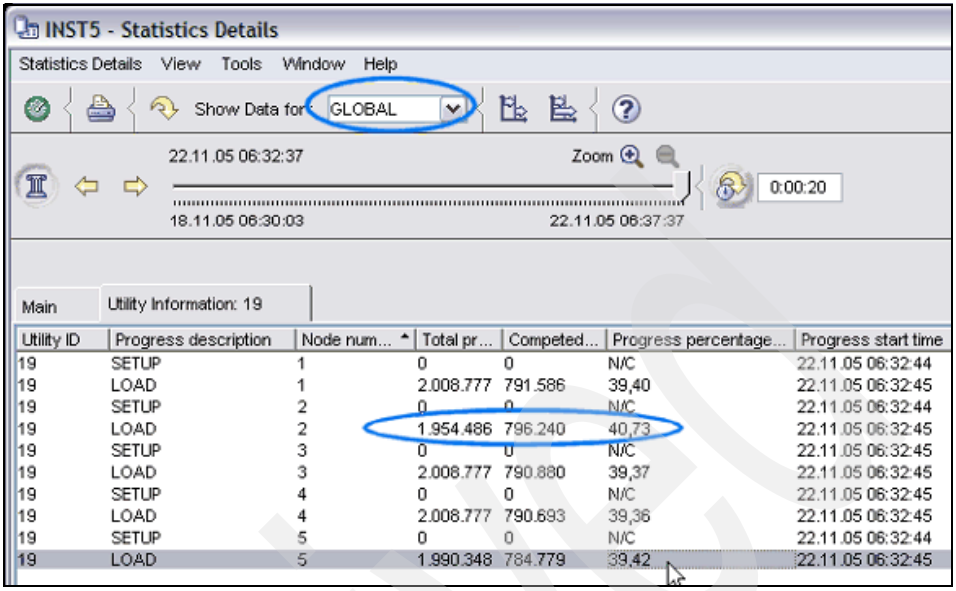


Figure C-61 Load process details

Utility heap

The load process uses the utility heap as memory buffer. When you load data into MDC tables, this buffer is heavily employed to build the MDC blocks. You can monitor the utility heap in the Memory Pool Usage pane of the Statistic Details window. Look for the heap type Backup / Restore / Util Heap and double-click it (Figure C-62).

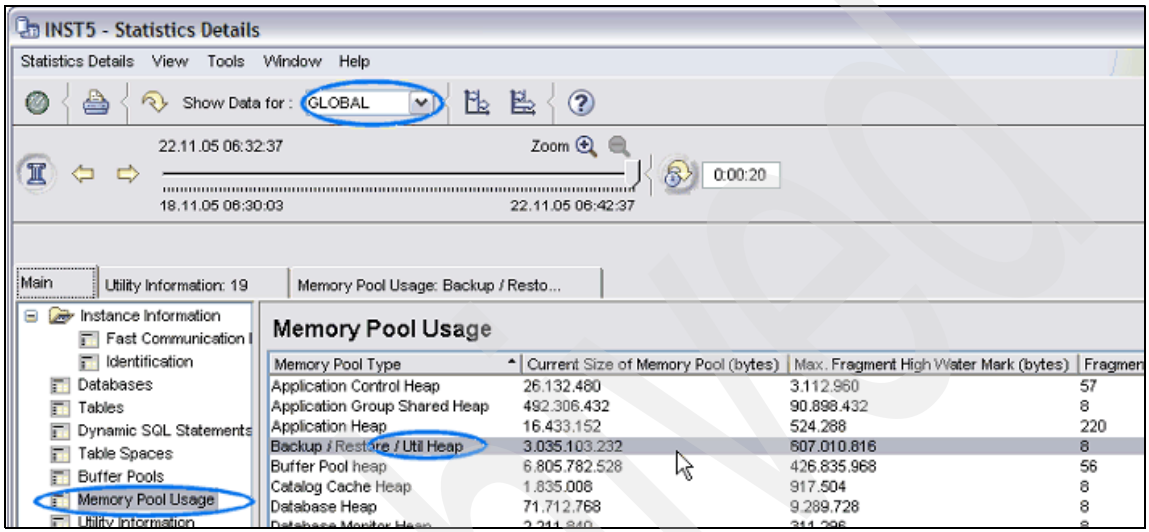


Figure C-62 Utility heap

The Statistics Details window that opens contains a High Water Mark (bytes) column that displays the heap on each partition (Figure C-63 on page 589). Unfortunately, due to the way DB2 exposes these metrics, PE cannot display the actual node number here (the fragment number is not the node number). But if you want to see the high water mark for a dedicated partition, you can switch to partition view for that partition.

Check the high watermark against the values in the *Configured size (bytes)* column. If it is near or equal to that size, you should consider increasing the utility heap size.

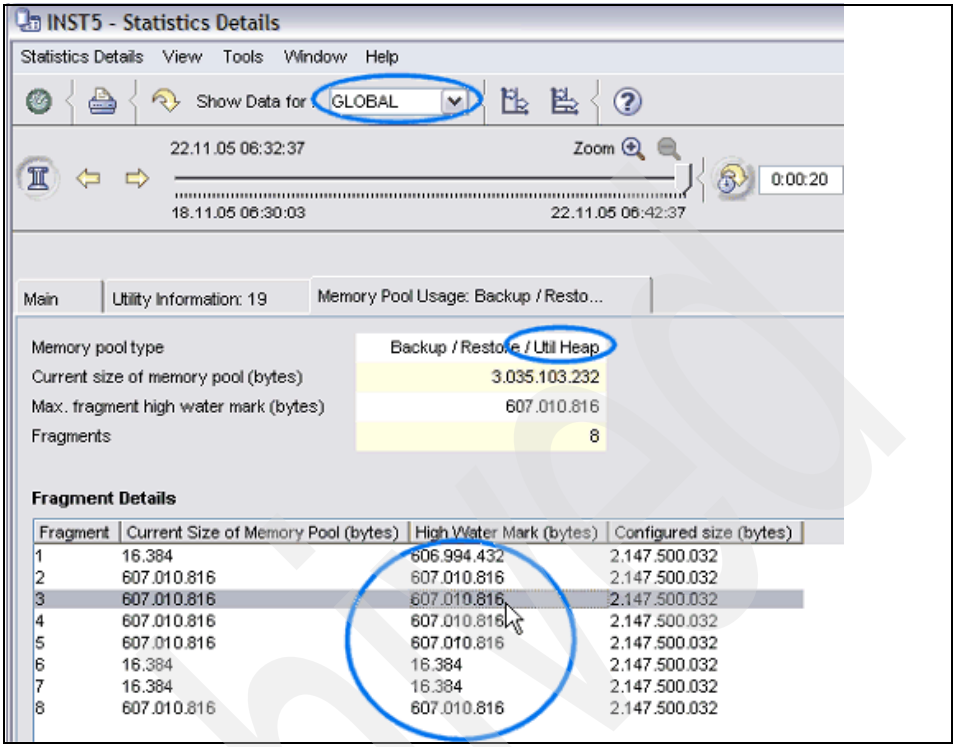


Figure C-63 High water mark

Monitoring tablespaces

Before you start the load or as the load is running, you should make sure that the tablespaces are not nearing their maximum capacity. You can track the size of the tablespaces in the Statistics Details window. To do so, select the Table Spaces category and double-click the tablespace. Then select the Containers category. Check the *Free space on file system (MB)* column to see how much space is left for the containers to grow (Figure C-64).

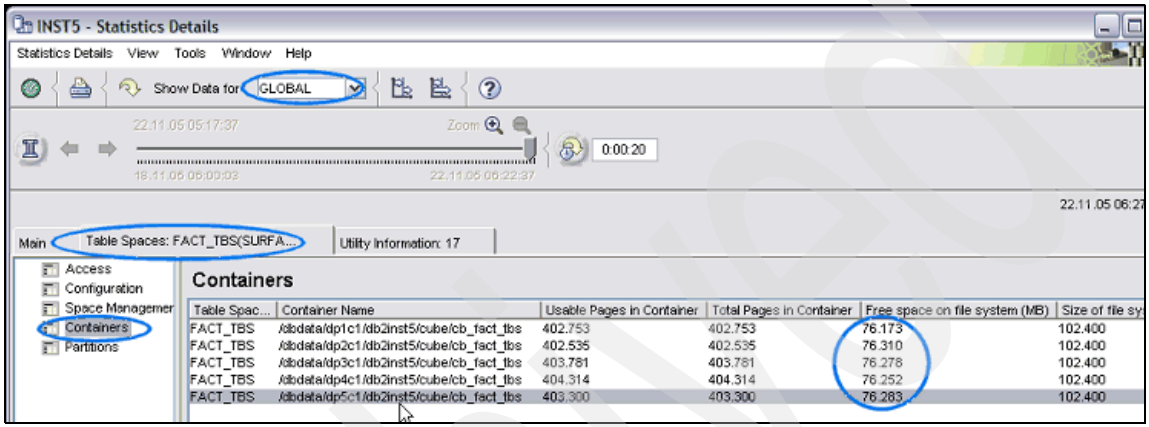


Figure C-64 Monitoring tablespaces

Log space

If you are loading to staging tables first, you will do the INSERT...SELECT from the staging tables to the actually queried tables after the load is complete. This is a transactional process, which leads to transaction logging. There are two potential issues to be aware of:

- The first is that your available transaction log space might run full. This can happen if you do the incremental INSERT in one or a few big transactions (for example, you have one INSERT…SELECT for the entire data of the staging table). You can check your transaction log space usage for your database in the Logs pane of the Statistics Details window (Figure C-65 on page 591).

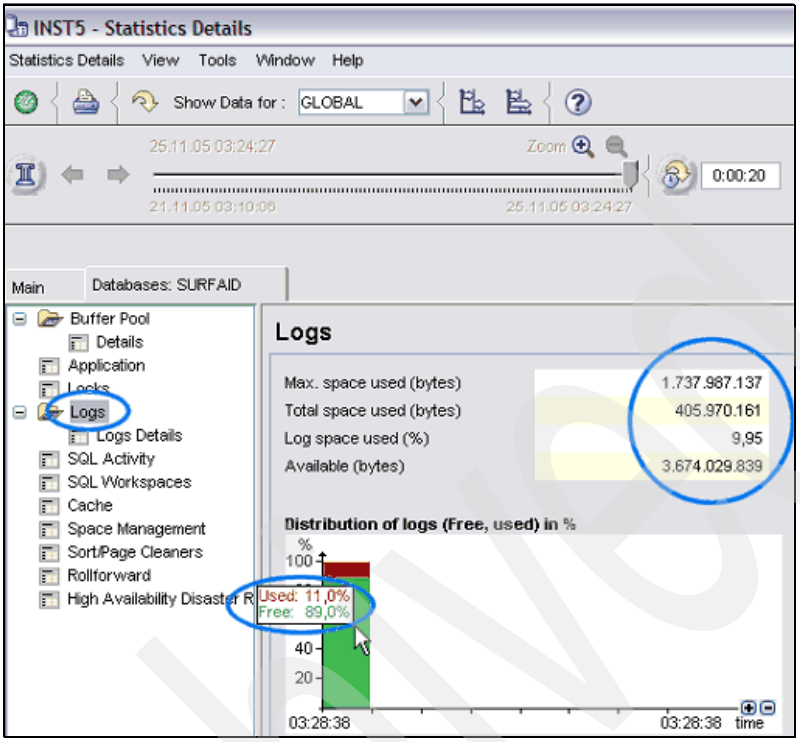


Figure C-65 Monitoring log spaces

The problem of log space running full can be resolved either by increasing it or by having more granular transactions (for example, by splitting up the INSERT…SELECT from the staging table to many statements with according range predicates).

- The second potential problem is that logging can cause a bottleneck due to I/O. You should watch the logging page activity (in the same pane as the log space usage) and the disk I/O of your disks where the transaction log files are located. See “Disk I/O” on page 542 for more information.

Logging I/O problems can be countered by distributing the file system for transaction log files over more physical disks. You can also try to increase the size of the log buffer (LOGBUFSZ in the database configuration).

Concurrent load and analytics

Oftentimes, the incremental load must run concurrently with the ongoing analytic workload. In these cases, you must also check for concurrency issues in which the queries are blocked by a load or vice versa. It is especially true for concurrent load and analytics that the previously mentioned staging tables are used for loading the data to the actual tables and then doing INSERT…SELECT operations into the actual tables.

There are different strategies to avoid concurrency issues. For example, you can issue the queries with isolation level Uncommitted Read (UR). This strategy causes the queries to read rows that have been inserted but not yet committed, as though the commit already happened. Another strategy is to set the DB2 registry variable DB2_SKIPINSERTED=YES, which causes the query to ignore the not-yet-committed rows. In both strategies, the queries are not blocked by the concurrent process that is inserting new data into the table.

But before thinking about this task, you should determine if you have concurrency problems and, if so, what the cause is.

Detecting concurrency issues

One potential issue is that some applications must wait for others for a long time. You can check this in the Locks pane of the Statistics Details window (Figure C-66 on page 593). Look at the values in the *Waits* and the *Time waited (sec)* fields and see if the wait time is high with regards to a selected delta interval.

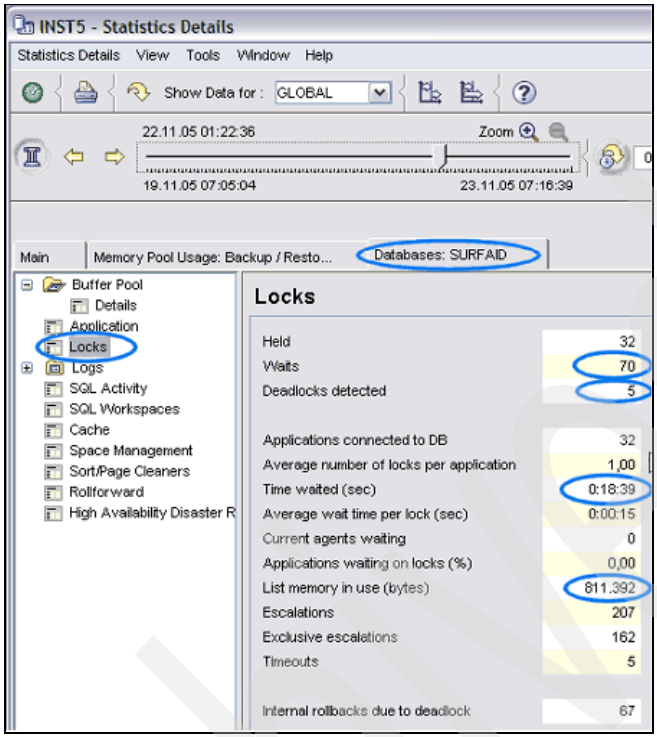


Figure C-66 Checking locks

Another potential issue is that you are experiencing deadlocks (as determined by looking at the *Deadlocks detected* field).

A potential reason for both problems could be that you are experiencing lock escalations too (as shown in the *Escalations* field). If you are, you should tune your lock list size. Check the *List memory in use (bytes)* field. If it is near the configured lock list size, you should increase your lock list size (LOCKLIST). To determine the current lock list size, look at the *Maximum storage for lock list (4 KB)* field in the System Parameters - Database window, which can be opened from the System Overview window. If not, you should increase the *Max. percent of lock list before escalation* (MAXLOCKS) parameter to a higher value, because it limits how much of the lock list can be consumed by a single application in percentage points.

Another way to mitigate problems with lock list size is to avoid locking altogether by using the Uncommitted Read (UR) isolation level.

It is not always possible to modify the lock list to solve concurrency issues. When it is not possible, you must analyze these issues in more detail. If you want to discover which applications are conflicting with each other, you can use the Applications in Lock Conflicts pane from the System Overview window. If you would rather know which tables that the lock waits are happening on, you can use the Locking Conflicts pane.

If you want to analyze deadlocks, you should start the event exception processing for deadlocks in the Activation dialog (from the System Overview window, click **Tools** → **Exceptions** → **Activation - Multiplatform**; see Figure C-67). As soon as a deadlock is detected, PE will raise an exception and display it in the System Overview window, where you can double-click it to drill down to the deadlock details.

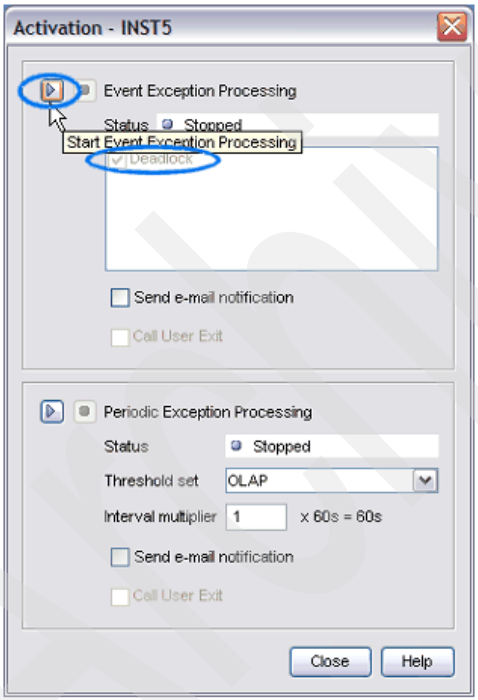


Figure C-67 Event exception processing

Parameter marker check

BI systems typically execute analysis statements that take longer per definition. For that purpose, the optimization of statements by DB2 is a crucial performance factor. Therefore, the DB2 optimizer should get as much information as possible for the statement compilation, including the values of all predicates. It is normally

better to not use parameter markers in the statements, but rather to specify the values as literals directly. This leads to DB2 statement compilation with literally each execution. But the time that is required by this approach is normally less than the time that would be spent executing a less optimal plan that the optimizer has created due the missing knowledge of parameter marker values.

The following query can be used to check all statements that are executed by the system for the use of parameter marker values. Refer to “Linking top 10 statements to execution details” on page 568 for information about creating new queries. The result list shows one entry per such statement along with its average execution time. Use the PWH query facility to store and execute this statement.

```
SELECT     sql.STMT_TEXT statement_using_par_markers,
AVG(sql.ATIMEP_EXECUTIONS)
          avg_exec_time, MIN (INTERVAL_TO) first_captured_at, MAX(INTERVAL_TO)
           last_captured_at
FROM       DB2PM.DYNSQL sql
WHERE      MEMBER_ID = - 2
           AND LOCATE(STMT_TEXT, '?') <> 0
GROUP BY   sql.STMT_TEXT
ORDER BY   AVG(sql.ATIMEP_EXECUTIONS) )
```

Verifying MQT effectiveness

Materialized query tables (MQTs) are an elementary building block for BI performance. However, it is not desirable to maintain MQTs that provide zero or even rather low performance benefit to the actual BI workload that is executed. You can use PE to verify if and to what extend a MQT is used.

Use the explain capability (see “Understanding statement plan” on page 573) to determine if certain queries are using MQTs or not by looking in the explain output for the base tables that the query text actually refers to. If there is an eligible MQT for the query, the optimizer rewrites it to run against the MQT. In the output, you will see MQT(s) that are being accessed instead of some or all base tables.

If you want to see to what extent MQTs are used in total, open the Statistics Details window and navigate to the Tables category. Select the check box at the bottom to make PE retrieve and display table access statistics. Then, locate the MQTs and compare their I/O statistics (in the Rows read column) with those of the base tables to understand to what extent MQTs are used over base tables (Figure C-68).

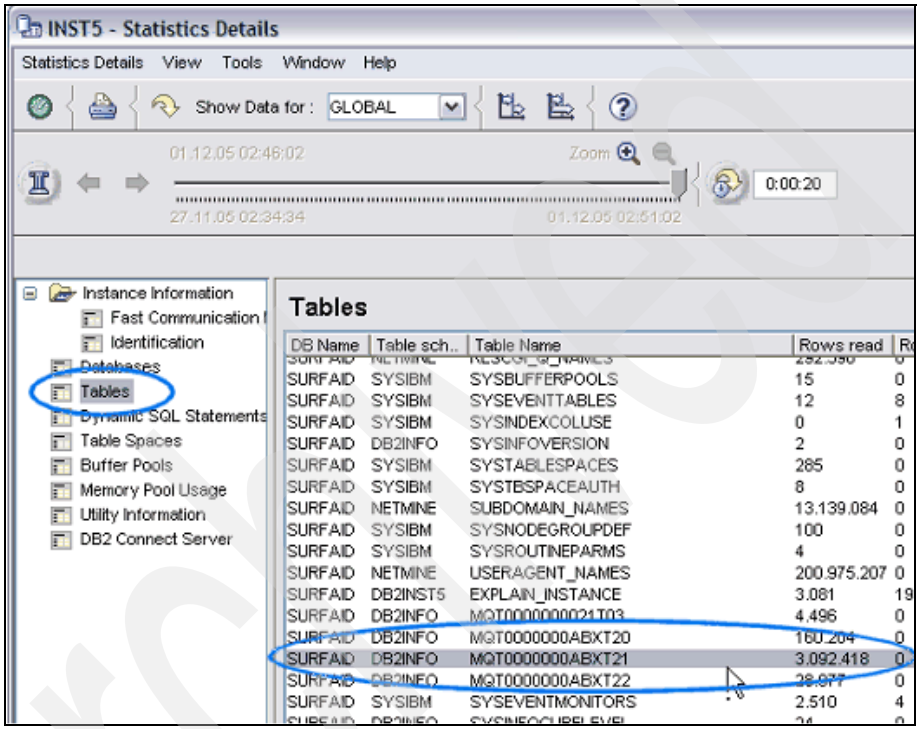


Figure C-68 Verifying MQT effectiveness

FCM tuning

Between partitions on different physical machines, the underlying transport infrastructure for the table queues (see “Understanding statement plan” on page 573) is the fast communication manager (FCM). The main configuration parameter for it is the number of FCM buffers (num_fcm_buffers) on the database manager level. You can determine if this parameter is set appropriately by opening the Statistics Details window, navigating to **Instance Information** → **Fast Communication Manager**, and switching to global view (Figure C-69 on page 597).

The key property to monitor is the *Free buffers low water mark*. It should not be too low (for example, less than 5% of num_fcm_buffers) or even 0, because this means that the number of FCM buffers are not sufficient to satisfy the actual inter-partition communication. You should increase the num_fcm_buffers parameter, or, alternatively, try to achieve more collocated joins to avoid some need for shipping data between partitions.

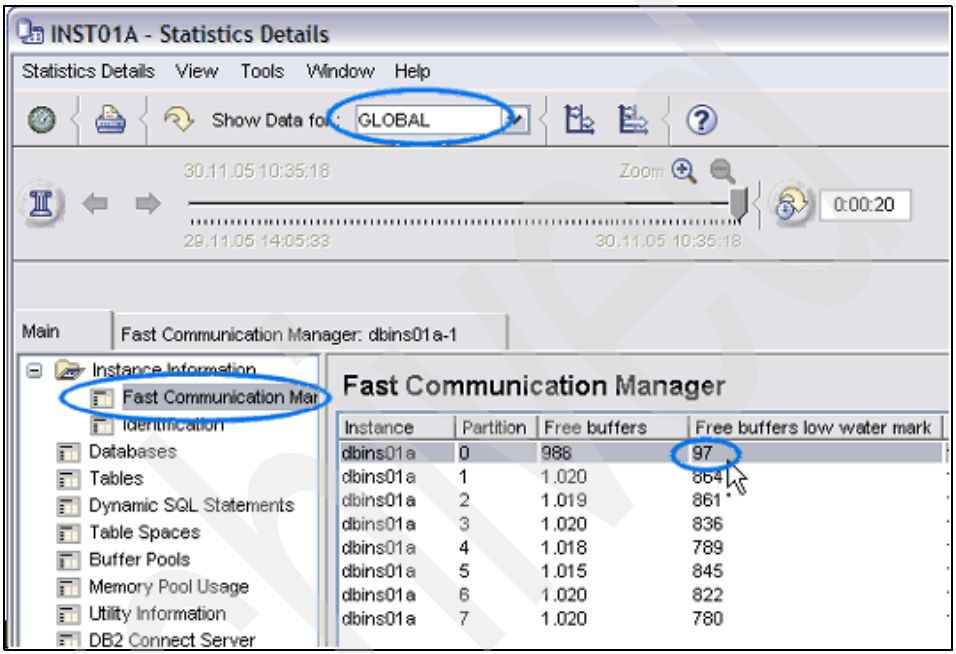


Figure C-69 Monitoring FCM

Alternately, a rather high number for the low water mark (for example, more than 50% of num_fcm_buffers) is also a cause for concern. There are several reasons for a high number for the low water mark, including:

- Too many FCM buffers are configured (your workload just does not need that many).
- All partitions are on the same machine (no FCM communication is needed).
- No workload is running.
- Network capacity is not sufficient to handle FCM communication.

If you see skew on the low water mark on certain partitions, check these partitions for such issues as I/O bound execution.

Dashboard monitoring of key BI performance indicators

At several places throughout this document, you have probably noticed the capability of PE to visualize performance data in different types of diagrams based on the System Health window (see “Visualization” on page 544). This window allows you to set up data visualization in a very flexible way, including the custom setup of dashboards for different BI monitoring topics.

To set up your BI performance dashboards:

1. Open the System Health window from the System Overview window.
2. Create new data groups (one for each dashboard) by right-clicking **Data Groups** and selecting **New…** (Figure C-70). A suggested set of dashboards is:
   - Sorts
   - CPU / Memory / Disk
   - Page I/O
   - Workload
   - FCM
   - Incremental Load
   - Storage

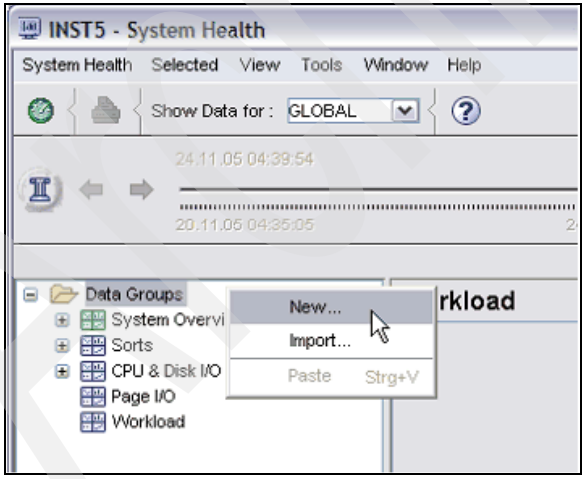


Figure C-70 Setting up BI performance dashboard

3. Now you can define the single thresholds in the data groups. PE provides a set of predefined data views, as well as the ability to freely define custom ones (by clicking **New…**) (Figure C-71 on page 599). For the custom data

views, you can choose between different data categories and then select one or multiple counters of that category. There is a rich set of options for displaying the diagram. For details, refer to the *IBM DB2 Performance Expert: Installation and Configuration,* SC18-0101.

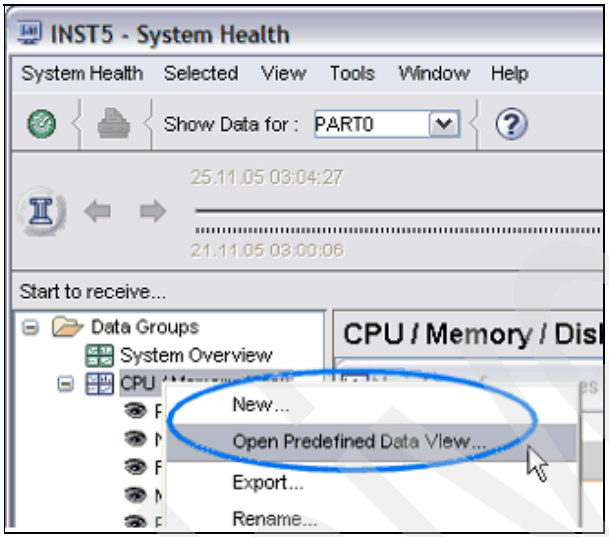


Figure C-71 Define data view

The System Health window supports exporting data view definitions per group to an XML file, which allows you to distribute your own dashboard definitions to other PE clients and to import them there. This document is also accompanied with a BI-specific set of data view definitions that you can use to set up your BI dashboards with minimal effort.

Figure C-72 on page 601 shows the data view definitions that are found in these supplementary XML files. PE provides a set of BI data view definitions that you can use to help you to set up your BI dashboard quickly and easily. These definitions are provided to you as a set of XML files in the installation directory of your client in the samples\SystemHealth directory.

To deploy them to your DB2 PE installation:

1. Open the System Health window, right-click **Data Groups**, and select **Import...**.
2. Navigate to the directory where you unpacked the zip file, select an XML file, and then click **Open**.

3. Depending on the XML file that you selected, additional dialogs might be displayed. Enter the appropriate information in these dialogs:
   - For BI_DataViews_CPU_Memory_Disk.xml: Specify the disks that you want to monitor I/O for.
   - For BI_DataViews_FCM.xml: Specify the partitions that you want to monitor FCM buffers for.
   - For BI_DataViews_Sorts.xml, BI_DataViews_IncrementalLoad.xml, BI_DataViews_PageIO.xml and BI_DataViews_Workload.xml: If you have configured multiple databases of your monitored DB2 instance for monitoring in PE, select the database that you want to monitor sorting, data loading, page I/O, and general workload for.
   - For BI_DataViews_Sorts.xml: Specify the disks where your temporary tablespaces are placed.
   - For BI_DataViews_Storage.xml: Specify the file systems for which you want to monitor the usage.

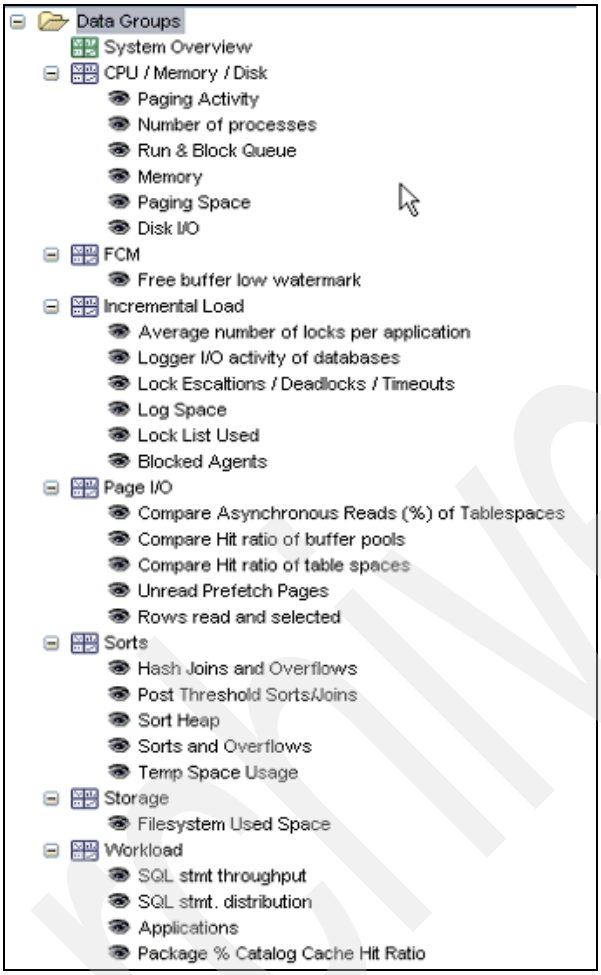


Figure C-72 Data view definitions

By using the System Health window, you have a very powerful means of intuitive access to your BI performance because it supports online monitoring via the auto-refresh option as well as history browsing, as is the case with any other PE monitoring panel. As demonstrated in “Visualized skew detection” on page 561, you can also apply the partition group view to the System Health window, which allows you to see a visualized display of your monitoring data for each single partition side by side.

The System Health window is also the dialog that you use to set up the most important data views for being displayed in the System Overview window for your system. Just check **Display in System Overview** in the context menu of a data view. The icon visually indicates that it is now also available in the *System Overview* window. The System Overview node holds all of the data views that have been marked in this way (Figure C-73).

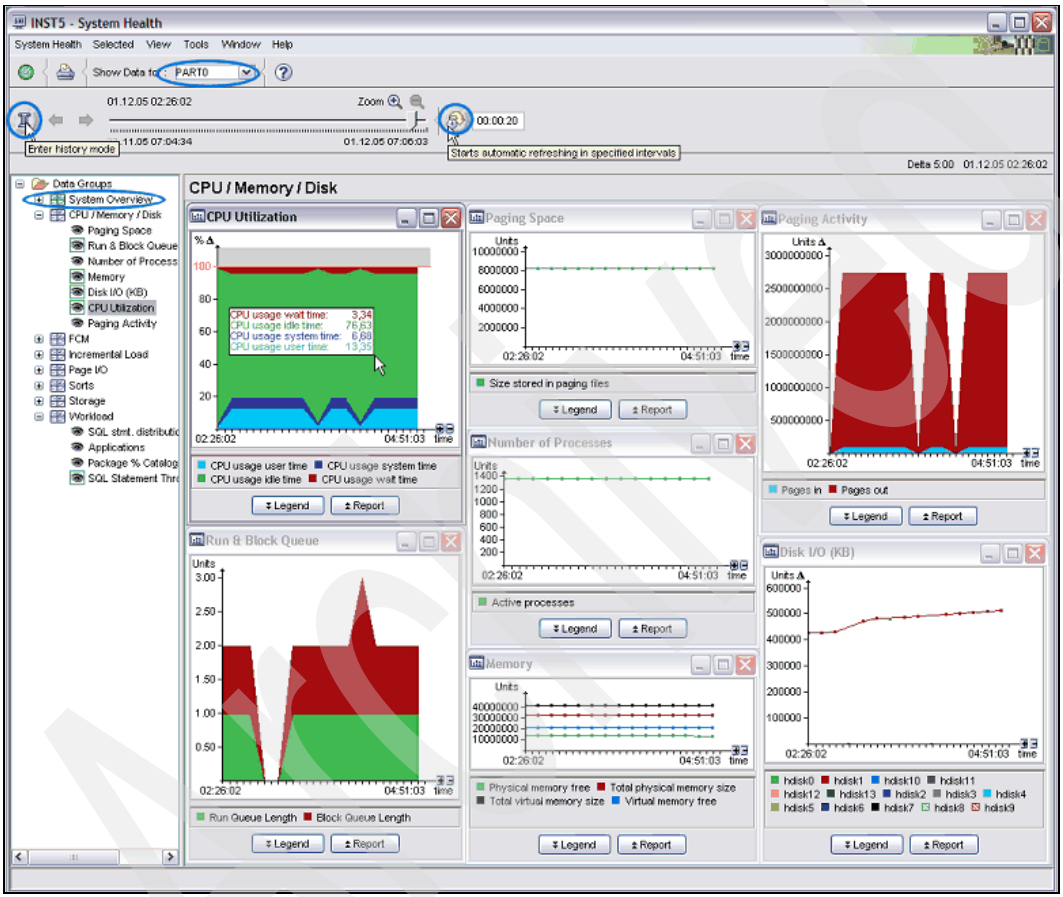


Figure C-73 System overview

Conclusion

BI performance is heavily related to DB2 performance, both of which require tooling support to understand and resolve performance issues. DB2 Performance Expert provides a very rich set of DB2 performance monitoring capabilities that you can use to perform many BI performance tuning tasks. An

especially convenient feature is DB2 PE's DPF-monitoring capability, which makes DB2 PE an extremely useful component of BI environments. DB2 PE's advanced monitoring techniques, such as history monitoring, exception processing, performance warehousing, and performance visualization, provide a great deal of flexibility and value.

Related publications

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

IBM Redbooks

For information about ordering these publications, see “How to get IBM Redbooks” on page 606. Note that some of the documents referenced here may be available in softcopy only.

- *DB2 UDB ESE V8 non-DPF Performance Guide for High Performance OLTP and BI,* SG24-6432
- *DB2 UDB Performance Expert for Multiplatform: A Usage Guide*, SG24-6436
- *IBM DB2 Performance Expert for z/OS Version 2,* SG24-6867

Other publications

These publications are also relevant as further information sources:

- *IBM DB2 Performance Expert: Installation and Configuration,* SC18-0101
- *IBM DB2 Performance Expert for Multiplatforms IBM DB2 Performance Expert for Workgroups Installation and Configuration*, SC18-9191
- *IBM DB2 Performance Expert for Multiplatforms, Workgroups, and z/OS IBM DB2 Performance Monitor for z/OS IBM DB2 Buffer Pool Analyzer for z/OS Messages*, SC18-7974
- *IBM DB2 Performance Expert for Multiplatforms, Workgroups, and z/OS IBM DB2 Performance Monitor for z/OS Monitoring Performance from the Workstation*, SC18-7976
- *IBM DB2 UDB Administration Guide: Implementation V8*, SC09-4820
- *IBM DB2 UDB Administration Guide: Planning V8.2*, SC09-4822
- *IBM DB2 UDB System Monitor Guide and Reference V8*, SC09-4847
- *IBM DB2 Universal Database Administration Guide: Performance V8.2*, SC09-4821

- Snow, et al, *Advanced DBA Certification Guide and Reference: for DB2 Universal Database v8.1 for Linux, UNIX, and Windows*, Pearson Education, 2003, ISBN 0130463884

Online resources

These Web sites and URLs are also relevant as further information sources:

- *AIX 5L Version 5.3 Common Information Model Guide*, SC23-4942, found at:

  http://publib.boulder.ibm.com/infocenter/pseries/topic/com.ibm.aix.doc/aixbman/cim/cim.pdf

- AIX Toolbox for Linux Applications

  http://www-1.ibm.com/servers/aix/products/aixos/linux/download.html

- DB2 Performance Expert

  http://www.ibm.com/software/data/db2imstools/db2tools/db2pe/index.html

- *IBM DB2 Content Manager V8.3 Enterprise Edition Performance Tuning Guide*, found at:

  http://www.ibm.com/support/docview.wss?uid=swg27006452

- IBM DB2 Universal Database™ for Linux, UNIX, and Windows

  http://www.ibm.com/software/data/db2/udb/

- IBM System p5 - AIX 5L and Linux server features and benefits

  http://www.ibm.com/pseries/en_US/aixbman/cim/cimtfrm.htm

- IBM Web Membership Signup

  https://www6.software.ibm.com/reg/swgmail/swgmail-r?S_TACT=1O4CBW68

- rows_read Help

  http://publib.boulder.ibm.com/infocenter/db2help/topic/com.ibm.db2.udb.doc/admin/r0001317.htm?

How to get IBM Redbooks

You can search for, view, or download Redbooks, Redpapers, Hints and Tips, draft publications and Additional materials, as well as order hardcopy Redbooks or CD-ROMs, at this Web site:

ibm.com/redbooks

Help from IBM

IBM Support and downloads

ibm.com/support

IBM Global Services

ibm.com/services

Index

D

E

F

G

H

N

O

P

Q

R

S

T

U

V

W

X

Z

DB2 Performance Expert for Multiplatforms V2.2

(1.0” spine)
0.875”<->1.498”
460 <-> 788 pages

DB2 Performance Expert for Multiplatforms V2.2

Step-by-step DB2 PE installation and configuration guide

Detailed DB2 PE features and functions usage guide

Performance tuning scenarios

DB2 Performance Expert (PE) for Multiplatforms V2.2 is a workstation-based performance analysis and tuning tool for managing a heterogeneous mix of DB2 systems with a single end-user interface. DB2 PE simplifies DB2 performance management by providing you with the ability to monitor applications, system statistics, and system parameters using a single tool.

This IBM Redbook provides an overview of the architecture of DB2 Performance Expert. We highlight key considerations in planning DB2 PE V2.2 for your environment and provide a step-by-step installation and configuration guide

We discuss, in detail, the DB2 PE V2.2 functions and features. Recommendations and tips for DB2 performance tuning are also introduced.

Finally, we discuss some of the commonly encountered problems faced by a DBA when managing a DB2 environment, and describe how the tool can be used to diagnose and resolve these performance problems.

INTERNATIONAL TECHNICAL SUPPORT ORGANIZATION

BUILDING TECHNICAL INFORMATION BASED ON PRACTICAL EXPERIENCE

IBM Redbooks are developed by the IBM International Technical Support Organization. Experts from IBM, Customers and Partners from around the world create timely technical information based on realistic scenarios. Specific recommendations are provided to help you implement IT solutions more effectively in your environment.

For more information:
ibm.com/redbooks

SG24-6470-01 ISBN 0738496413