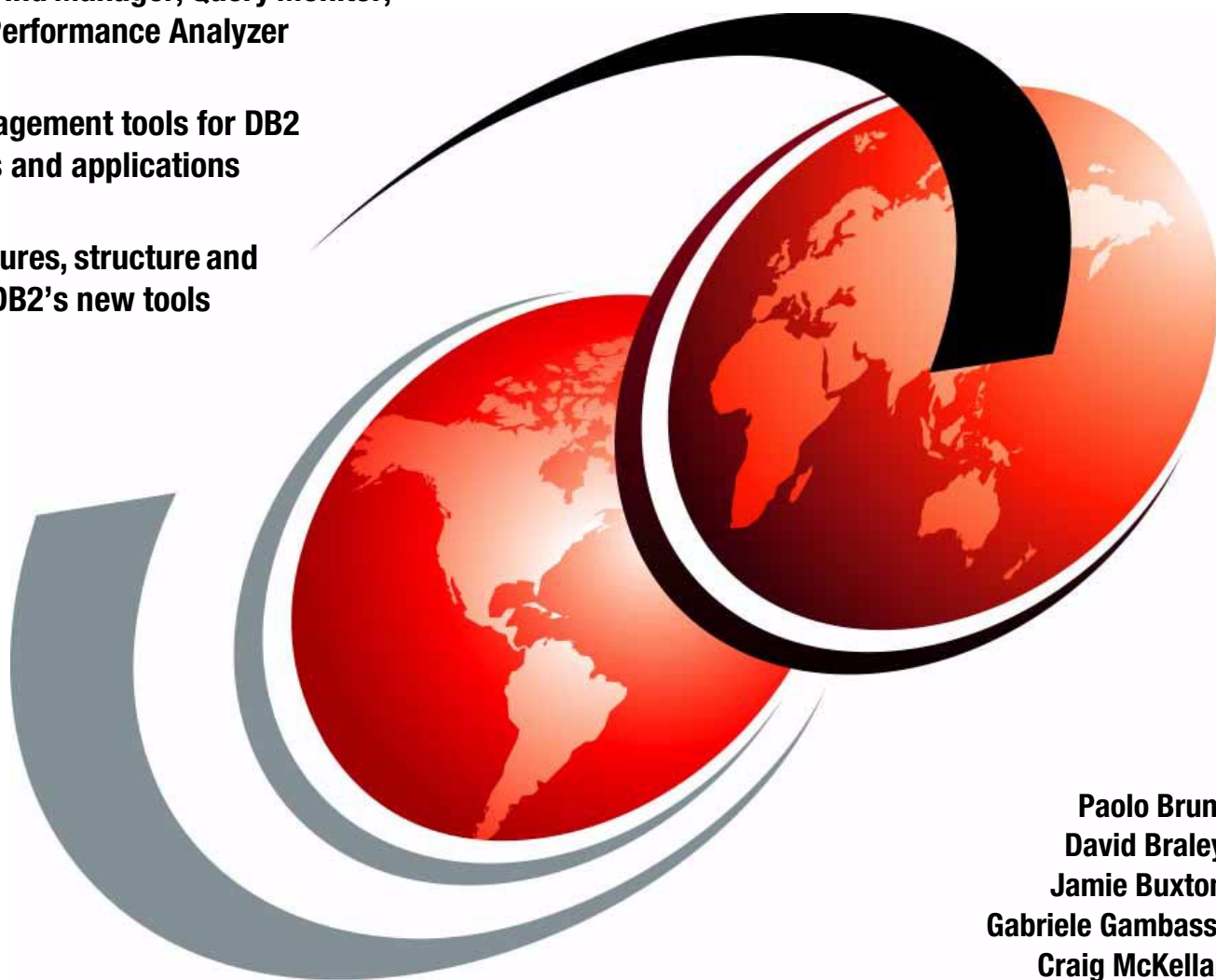


New Tools for DB2 for OS/390 and z/OS Presentation Guide

IBM DB2 Bind Manager, Query Monitor,
and SQL Performance Analyzer

Data Management tools for DB2
databases and applications

Major features, structure and
usage of DB2's new tools



Paolo Bruni
David Braley
Jamie Buxton
Gabriele Gambassi
Craig McKellar

Redbooks



International Technical Support Organization

SG24-6139-00

**New Tools for DB2 for OS/390 and z/OS
Presentation Guide**

March 2001

Take Note!

Before using this information and the product it supports, be sure to read the general information in Appendix E, "Special notices" on page 237.

First Edition (March 2001)

This edition applies to IBM DB2 Bind Manager, Program Number 5655-D38, IBM DB2 Query Monitor, Program Number 5655-E67, and IBM DB2 SQL Performance Analyzer, Program Number 5697-F57, for use with DB2 UDB Server for OS/390 and z/OS.

Note

This book is based on pre-GA versions of the products and may not apply when the products become generally available. We recommend that you consult the products documentation or follow-on versions of this redbook for more current information.

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

When you send information to IBM, you grant IBM a non-exclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

© Copyright International Business Machines Corporation 2001. All rights reserved.

Note to U.S Government Users - Documentation related to restricted rights - Use, duplication or disclosure is subject to restrictions set forth in GSA ADP Schedule Contract with IBM Corp.

Contents

Preface	ix
The team that wrote this redbook	ix
Comments welcome	x
<hr/>	
Part 1. Introduction	1
Chapter 1. Data Management tools	3
1.1 The zSeries server	5
1.2 Why use IBM Data Management tools?	6
1.3 Data Management categories	7
1.3.1 Database administration	8
1.3.2 Performance management	8
1.3.3 Recovery and replication	8
1.3.4 Application management	9
1.3.5 The hidden work	9
1.4 DB2 for OS/390 Management Clients Package	10
Chapter 2. Introduction to DB2 tools	11
2.1 The new DB2 tools at a glance	12
2.2 IBM DB2 Administration Tool	14
2.3 IBM DB2 Object Restore	16
2.4 IBM DB2 High Performance Unload	17
2.5 IBM DB2 Log Analysis Tool	19
2.6 IBM DB2 Table Editor	20
2.7 IBM DB2 Automation Tool	23
2.8 IBM DB2 Archive Log Compression Tool	24
2.9 IBM DB2 Object Comparison Tool	25
2.10 IBM DB2 Performance Monitor	27
2.11 IBM DB2 SQL Performance Analyzer	29
2.12 IBM DB2 Query Monitor	31
2.13 IBM DB2 DataPropagator	32
2.14 IBM DB2 Row Archive Manager	35
2.15 IBM DB2 Recovery Manager	37
2.16 IBM DB2 Change Accumulation Tool	39
2.17 IBM DB2 Bind Manager	40
2.18 IBM DB2 Web Query	41
<hr/>	
Part 2. IBM DB2 Bind Manager	43
Chapter 3. Overview	45
3.1 DB2 Bind Manager objectives	46
3.1.1 Objectives versus components	47
Chapter 4. Structure and configuration	49
4.1 Bind Manager - Flow	50
4.1.1 Bind required	51
4.1.2 No bind required	52
4.2 DBRM Checker - Purpose	54
4.2.1 DBRM Checker - JCL	55
4.2.2 DBRM Checker - Output	56

4.3 Path Checker - Purpose	57
4.3.1 Path Checker - JCL	58
4.3.2 Path Checker - Report JCL	59
4.3.3 Compare JCL	62
4.3.4 Path Checker - Test JCL	65
Chapter 5. Usage and considerations	69
5.1 Step 1 - Minimal impact	69
5.2 Step 2 - Closer integration	71
5.3 Step 3 - Final solution	72

Part 3. IBM DB2 Query Monitor 73

Chapter 6. Overview	75
6.1 Introduction to Query Monitor	76
6.2 Presenting DB2 Query Monitor	77
6.3 Major features of DB2 Query Monitor	78
6.4 DB2 Query Monitor terminology	80
Chapter 7. Structure and configuration	83
7.1 Installing Query Monitor	84
7.2 Query Monitor start up JCL	85
7.2.1 Controlling storage	85
7.2.2 CQMPARMS data set	86
7.2.3 Definition of start up parameters	86
7.2.4 Other information	86
7.3 Query Monitor control file	88
7.4 Query Monitor processing	89
7.4.1 Description of operations	89
7.4.2 DB2 objects	90
7.4.3 Managing DB2 objects	90
7.4.4 Managing past data	90
7.5 Operation of Query Monitor	92
7.6 DB2 subsystem monitoring	94
7.7 Updating DB2 subsystem information	95
7.8 Entering the DB2 specific information	96
7.9 Initial profile creation	97
7.9.1 Profile summary	97
7.10 Monitoring and application profiles	98
7.10.1 Categories for filters	98
7.10.2 Using wildcards	99
7.10.3 Altering filters	99
7.11 Creating and updating profiles	100
7.11.1 Creating a PLAN filter	100
7.11.2 Other filtering parameters	101
7.12 A completed filter	103
7.13 Exception profiles	104
7.14 Exception profile processing	105
7.15 Updating profiles	106
7.16 Checklist to start QM subsystem	107
Chapter 8. Usage and considerations	109
8.1 Running and managing Query Monitor	110

8.2	Activity screen introduction	112
8.2.1	Current and Past Activity panels	112
8.2.2	Commands	113
8.2.3	Data columns	114
8.2.4	Line commands	115
8.2.5	Function key navigation	115
8.3	Thread navigation	116
8.4	Thread Activity options	117
8.4.1	The detail panel or summarize options group	117
8.4.2	Active or interval	117
8.5	Thread Activity panel	118
8.6	Plan detail panel	119
8.7	DBRM - Detail (1)	120
8.8	DBRM - Detail (2)	121
8.9	SQL statement text	122
8.10	Examples of detail screens	123
8.11	Dynamic SQL caching	124
8.12	Open cursor	126
8.13	Thread, plan, or DBRM activity	127
8.14	Case studies	128
8.15	Current Activity - Case study (1)	129
8.16	Current Activity - Case study (2)	130
8.17	Current Activity - Case study (3)	131
8.18	DB2 Query Monitor and DB2 Performance Monitor	132
8.18.1	DB2 PM Online Monitor functions	132
8.18.2	Batch reporting	133
8.18.3	Why you need two monitors?	133
8.18.4	Using the two monitors	133
8.19	DB2 Query Monitor and DB2 PM comparison	134
8.19.1	Observed variations	134
8.19.2	Other variations	135
8.19.3	Conclusion	135
8.20	Past Activity panel	136
8.20.1	Intervals panel	136
8.20.2	Unload of data	137
8.20.3	Load of data	137
8.20.4	Summary	137
8.21	Unloading selected intervals	138
8.22	Unloading all available intervals	139
8.23	Loading selected intervals	140
8.24	Load via JCL	141
8.25	Past Activity considerations	142
8.26	Command Activity processing	144
8.27	DB2 commands executed	146
8.28	DB2 Command Activity	147
8.29	DB2 Utilities and Binds	148
8.30	Summary of Query Monitor features	149
8.31	Query Monitor hints and tips	151

Part 4. IBM DB2 SQL Performance Analyzer	153
---	------------

Chapter 9. Overview	155
9.1 Scope and objectives	156

9.1.1	Prediction and prevention	156
9.1.2	SQL Advisor	157
9.2	Objectives	158
9.3	SQL cost components	160
9.3.1	DB2 tracing and SMF records	161
9.4	SQL/PA and other tools	162
9.4.1	DB2 PM	162
9.4.2	DB2 Estimator	163
9.4.3	QMF Governor	163
9.4.4	QMF HPO	163
9.5	How SQL/PA determines cost	164
Chapter 10. Structure and configuration		165
10.1	The approach	166
10.2	Programs and interfaces	167
10.2.1	Parsing program ANLSCAN	167
10.2.2	Costing program ANLCOST	168
10.2.3	QMF Intercept program ANL4QMF	168
10.2.4	DB2 stored procedure ANLPROC	169
10.2.5	Binding the programs	169
10.3	Special features	170
10.3.1	Generic PLAN_TABLEs	170
10.3.2	Using the Registry table	172
10.3.3	Catalog access	173
10.3.4	QMF Interface response time	173
10.4	Input and output flow	174
10.4.1	Input SQL	174
10.4.2	ANLCNTL and ANLPARM parameters	175
10.4.3	ANLCNTL parameters	176
10.4.4	ANLPARM parameters	176
10.4.5	Parameters files allocation and storage	177
10.4.6	Special parameters: SQL Advisor	179
10.5	Programs output	182
10.5.1	Reports activation	182
10.5.2	QLIMIT report	184
10.5.3	Cost summary report	185
10.5.4	Enhanced Explain report	187
10.5.5	Detail trace report	188
10.5.6	QMF and Stored Procedure output	192
10.6	Working with the ISPF interface	193
10.6.1	Starting a session	193
10.6.2	Control panel	194
10.6.3	User parameter in ISPF	195
10.6.4	Input SQL	196
10.6.5	Report menu	197
10.7	The Batch Interface	198
10.7.1	The ANLSCAN step	198
10.7.2	The ANLCOST step	199
10.8	The stored procedure interface	200
10.8.1	The ANLSTP sample program	200
10.8.2	Using DB2 Stored Procedure Builder	202

Chapter 11. Usage and considerations	205
11.1 Using SQL/PA	206
11.2 A practical example	207
11.2.1 Using QLIMIT report	207
11.2.2 Identify the big hitters	208
11.2.3 Using the enhanced Explain report	209
11.2.4 A better access path	210
11.2.5 The final result	211
11.3 Considerations	212
11.3.1 What influences cost estimates	213
11.4 Limitations	215
Part 5. Appendices	217
Appendix A. Bind Manager	219
A.1 Path Checker Report Column mapping against Plan_Table	219
Appendix B. Query Monitor	221
B.1 Query Monitor sample output	221
B.2 DB2 PM long accounting report sample	223
Appendix C. SQL Performance Analyzer	227
C.1 ANLPARM user parameters	227
C.2 ANLCNTL configuration parameters	230
Appendix D. Using the additional material	235
D.1 Locating the additional material on the Internet	235
D.2 Using the Web material	235
D.2.1 System requirements for downloading the Web material	235
D.2.2 How to use the Web material	235
Appendix E. Special notices	237
Appendix F. Related publications	239
F.1 IBM Redbooks	239
F.2 IBM Redbooks collections	239
F.3 Other resources	239
F.4 Referenced Web sites	240
How to get IBM Redbooks	241
IBM Redbooks fax order form	242
Abbreviations and acronyms	243
Index	245
IBM Redbooks review	249

Preface

IBM's recent focus on tools has brought enhancements to current DB2 UDB Server for OS/390 ancillary products and also the introduction of new tools. A project was conducted that resulted in this IBM Redbook, in the form of presentation guide, where some of the newest of these tools, namely DB2 Bind Manager, DB2 Query Monitor, and DB2 SQL Performance Analyzer, are described in detail with references to usability scenarios.

This redbook is primarily a skills transfer vehicle that will help you understand the functionality provided by the three tools, and it can be used as a basis for preparing presentations.

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.

Paolo Bruni is a Certified Senior Consultant IT Architect currently on assignment as a Data Management Specialist for DB2 for OS/390 at the International Technical Support Organization, San Jose Center, where he conducts projects on all areas of DB2 for OS/390. During his many years with IBM, in development and in the field, Paolo's work has mainly related to database systems.

David Braley is a DB2 Database Administrator working within IBM Global Services on an outsourced contract with UK Insurance company, Royal & Sun Alliance. He has three years of experience working with DB2 within IBM and he has previously worked with the UK Government.

Jamie Buxton is a Systems Programmer working for IBM Global Services in the UK. He has over 10 years of experience working with DB2 and IMS in the manufacturing industry before joining IBM in 1997. As part of the SSO team based throughout the UK, Jamie helps support a variety of outsourced and internal IBM systems.

Gabriele Gambassi is a Senior DB2 Specialist working for IBM Global Service, Strategic Outsourcing in New Zealand. He has 14 years of experience working with DB2 as System Programmer and Database Administrator in various industries across the world. Gabriele joined IBM in 1998 after a successful Disaster Recovery project for a telecommunication company.

Craig McKellar is a DB2 Database Administrator with IBM Global Services in Australia. He has 14 years of experience with DB2 as a Systems Programmer and Database Administrator. He worked for 16 years with the Health Insurance Commission and Medibank Private prior to joining the IBM team in 2000, following government outsourcing. His experience includes many years of working with the integration and automation of DB2 tools and utilities.

Thanks to the following people for their invaluable contributions to this project:

Emma Jacobs

Claudia Traver

International Technical Support Organization, San Jose Center

Vasilis Karras
International Technical Support Organization, Poughkeepsie Center

Gabrielle Velez
International Technical Support Organization, Rochester Center

James Guo
Leta Hermes
Gopal Khrisnan
Terri Leamon
Annie Tsang
Dave Schwartz
Tom Ulveman Jensen
IBM Silicon Valley Laboratory

Jeff Krug
Mike Skopec
Rocket Software

Frank J. Ingrassia
IMSI Inc.

Gerald Hodge
HLS Technologies

Comments welcome

Your comments are important to us!

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

- Fax the evaluation form found in “IBM Redbooks review” on page 249 to the fax number shown on the form.
- Use the online evaluation form found at ibm.com/redbooks
- Send your comments in an Internet note to redbook@us.ibm.com

Part 1. Introduction

Chapter 1. Data Management tools



DB2 Universal Database is the data store for mission-critical applications that are available on all major computing platforms used in modern economy. Since its first implementation DB2 has been enhanced at each release through the introduction of sophisticated and effective technology in response to business trends and customers requirements.

Building on this tradition, IBM has recently identified the opportunity to enhance the area of Data Management tools for OS/390, in response to increasing software costs and a renewed demands for business applications on this highly available and cost effective platform.

The two objectives of this redbook are:

1. To introduce the new DB2 tools for OS/390 and z/OS with an overview of the tools and present the major features available to help you, the IBM customer, better manage DB2 databases and database applications while contributing actively to improve the cost efficiency of your business operations
2. To cover in more detail three of the recently announced and brand new products.

Therefore, the information provided in this part of the redbook is relevant to the IBM Data Management tools for DB2 for OS/390 listed in the September 2000 announcement (letter 200-302 in USA) and includes an introduction to the general areas of all products, followed by more information by product.

A summary of March 2001 announcement is also included.

The following three parts of this redbook present in more detail the IBM products:

- DB2 Query Monitor
- DB2 SQL Performance Analyzer
- DB2 Bind Manager

Each part is structured in three chapters which illustrate the main objectives of the tool, their structure and configuration, and a set of possible usage scenarios.

The sections that follow in this chapter introduce the IBM @server zSeries, as well as Data Management tools and categories, and the DB2 for OS/390 Management Tools Package.

IBM @server zSeries

World's most scalable server

Bullet-proof reliability

Dynamic workload management

Industry-leading security

New workload pricing

Unmatched availability



ibm.com/redbooks

© 2000 IBM Corporation

1.1 The zSeries server

The software products discussed in this redbook are for the OS/390 operating system and for the new z/OS which has been recently introduced to exploit the functions of the new hardware for the high end platform. It is appropriate to spend a few words to highlight the recent announcement of the zSeries since the same products are available for the new environment.

The zSeries is the latest platform for enterprise data processing operations that builds on many years of successful experience with the S/390 Enterprise Server. The unique strengths of this proven platform, its ability to run multiple workload types with exceptionally high reliability, throughput, security and manageability have been imitated but never matched by other server platforms.

In more recent times, "traditional" strengths have been extended by incorporating a number of open interfaces which allow interoperation and integration in an open, heterogeneous computing environment. Key among these are the UNIX System Services and Linux/390. These services allow the zSeries to run commercial applications including Enterprise Resource Planning (ERP) and Customer Relationship Management (CRM), such as SAP R/3, PeopleSoft and Siebel, together with popular Web and application servers like WebSphere and Lotus Domino, as well as the "best of breed" business intelligence applications.

The database engine of choice for the zSeries platform is once again DB2. Through a tight integration with the operating system, it provides higher levels of performance, integrity and availability to support your choice of e-business applications.

Why IBM Data Management Tools

The problems

- To contain and manage software costs
- To increase productivity of skilled premium resources
- To respond adequately to increased application and system demands

The solution

- New and enhanced suite of tools
 - comprehensive capabilities
 - affordable and competitive
- Data Management tools grouped into four functional areas

*Database
Administration*

*Performance
Management*

*Recovery &
Replication*

*Application
Management*



ibm.com/redbooks

© 2000 IBM Corporation

1.2 Why use IBM Data Management tools?

It is evident that raising software costs dominates today's data center budgets, forcing decisions that directly effect architecture efficiency, business initiative and operational flexibility.

The different priorities between technical planning and purchasing departments have, over the years, highlighted the lack of an adequate software asset management discipline. This weakness has been challenged by independent software vendors (ISV) for years.

The offering documented in this redbook is testimonial of IBM's commitment to help customers to reduce cost and enable them to exploit the power of the zSeries platform without compromising the quality of the database and application.

For convenience and flexibility, the tools have been grouped in four categories representing major disciplines in Data Management. They are:

- Database Administration
- Performance Management
- Recovery and Replication
- Application Management

Data Management categories

Database Administration

- Install and maintain the RDBMS
- Implement physical design
- Control access and security
- Establish standard, policy and procedures

Performance Management

- Maintain response as per SLA
 - perform REORG and RUNSTATS
 - perform monitoring and tuning
 - estimate resources

Recovery and Replication

- Maintain availability as per SLA
 - implement backup and recovery strategy
 - implement Business Recovery
- Assure data consistency across the business

Application Management

- Manage changing application
- Optimize resource usage
- Support Rapid Application Development
- Provide application quality



ibm.com/redbooks

© 2000 IBM Corporation

1.3 Data Management categories

The fast approaching evolution of traditional business into fully automated and connected e-business is causing higher demands of expertise and the consequent turnover of technically skilled and knowledgeable personnel. In response to this situation companies are changing their organizations and reviewing their business priorities causing, among other things, a different perception of the role and responsibility of Data Management (DM) professionals.

The separation line between System Programming, Database Administration and Application Development is becoming finer and because of the nature of our work and the changing technology, it is quite common today for Data Management professionals to be involved in activities ranging from the installation of software to the development of business applications.

Cognizant of their expanded role, DM professionals in general tend to agree on the activities required (and the challenges) to provide professional Data Management services, irrespective of the nature of business they support.

The IBM DB2 Tools offering aims to enhance your productivity and your value to the business for each one of the Data Management categories. In the following sections, we provide a more detailed definition of these categories.

Data modeling and logical database design are outside the scope of this redbook and therefore are not considered as topics. However, it is essential to remember that these areas represent the foundation of any efficient and cost effective business application.

1.3.1 Database administration

This category is commonly used to define most, if not all, Data Management activities. It is always associated with one or more very busy professionals that often perform more than what is listed here. Activities in this category may include:

- Installation, customizing and maintenance of the Database Management System (DBMS) software and associated products and tools
- Definition of standards, policies and procedures about the usage and sharing of database resources, including the documentation of the physical design
- Implementation of security at both system and database level, with administration and monitoring of the relevant data access
- Implementation of the physical design, involving creation of the required database structure, management of their changes and choice of the optimal physical parameters

1.3.2 Performance management

This is more an art than a discipline, and essentially it aims to maintain the agreed to and documented service level for the system and each application. Activities in this category include:

- Generation of regular reports about system and application performance, with implementation of exception processing for critical applications.
- Tuning of system configuration and allocations.
- Review and tuning of SQL data access path for both new and existing applications.
- Planning of additional capacity through regular review of system and application usage. This includes the estimation of resources for new applications.
- Implementation of data reorganization and a statistics gathering strategy to always guarantee optimal access; this activity is often perceived to be under the Database Administration category.

Data archival is an activity that has positive impact on performance. However, as it is often associated with backup of aging information, we associate it within the category of recovery and replication.

1.3.3 Recovery and replication

The activities grouped in this category are vital to safeguard the integrity of the data and they represent the most important activities performed by a Database Administrator. Irrespective of data being centralized or distributed, its integrity and availability is essential to the business and, given the importance, these activities are often covered by a Service Level Agreement that provides input to the overall strategy. Activities in this group include:

- Implementation of a local backup and recovery strategy with the main objective to preserve both physical and logical integrity of the data. Backups should be planned using both business processes documentation and an application schedule.

- Design and periodically test a business recovery strategy, also known as disaster recovery. This involves tight coordination with other IT components of a business infrastructure and it always includes the recovery of the system followed by the recovery of the applications and their data. For cost efficiency reasons, the local backup and recovery strategy is integrated in the business recovery strategy.
- Definition of data archival guidelines and implementation of the appropriate procedures to backup aging data and to restore it, if and when required by the business. The restore function is often automated and can be operated by non technical personnel.
- Administration of data replication across multiple platforms, including definition of the data transformation rules and resolution of potential conflict resulting from complex replication architecture. It is not unusual today to find implementation of an operational data store and data warehouse on different platforms and sometimes even on different RDBMS; this situation places a few challenges to the Database Administrator.

1.3.4 Application management

The introduction of Stored Procedure in DB2 have created new opportunity and new activity for the DM professionals. The Application Management category defines, from a database perspective, all activity required to support all kinds of applications including rapid developments and integration with the Web and distributed environments. These activities include:

- Provision of the necessary connectivity for both client/server and Web enabled applications
- Management of changes for database applications, particularly when deploying plan or package structures and including stored procedures definition
- Administration of rapid development tools that rely on the database for storing the application components

1.3.5 The hidden work

Despite the attempt to provide a comprehensive list of activities we still miss some of the “given” works that too often are forgotten: usually they become evident only when not performed.

- Fire fighting: This is the indispensable activity performed to solve all sorts of urgent problems, irrespective of your carefully planned tasks and responsibility.
- Project management: This is the activity that when associated with fire fighting makes your work a living contradiction in term.
- Documentation: We consider it the final touch of a DM professional service but unfortunately is too often relegated in the “nice-to-have” basket for time reasons.
- Training: This is too costly and very hard to get; in most of the cases requires personal commitment. However, you are kindly asked to provide it for your colleague. It is the best way to guarantee full return on technology investment.
- Funding request and justification: This is the measure of how passionate you are for your work.

1.4 DB2 for OS/390 Management Clients Package

Besides the tools mentioned previously, it is worth mentioning that DB2 for OS/390 also incorporates a feature which helps you exploit the full potential of your DB2 subsystem by providing five *workstation-based* tools for tuning, administering, developing, and managing DB2. The DB2 Management Clients Package, formerly called DB2 Management Tools Package, is a no-charge feature of DB2 for OS/390 Version 5, Version 6, and Version 7. It delivers the DB2 UDB for OS/390 Control Center, DB2 Installer, DB2 Visual Explain, DB2 Estimator, and DB2 Stored Procedure Builder.

Here is a brief description of these client tools:

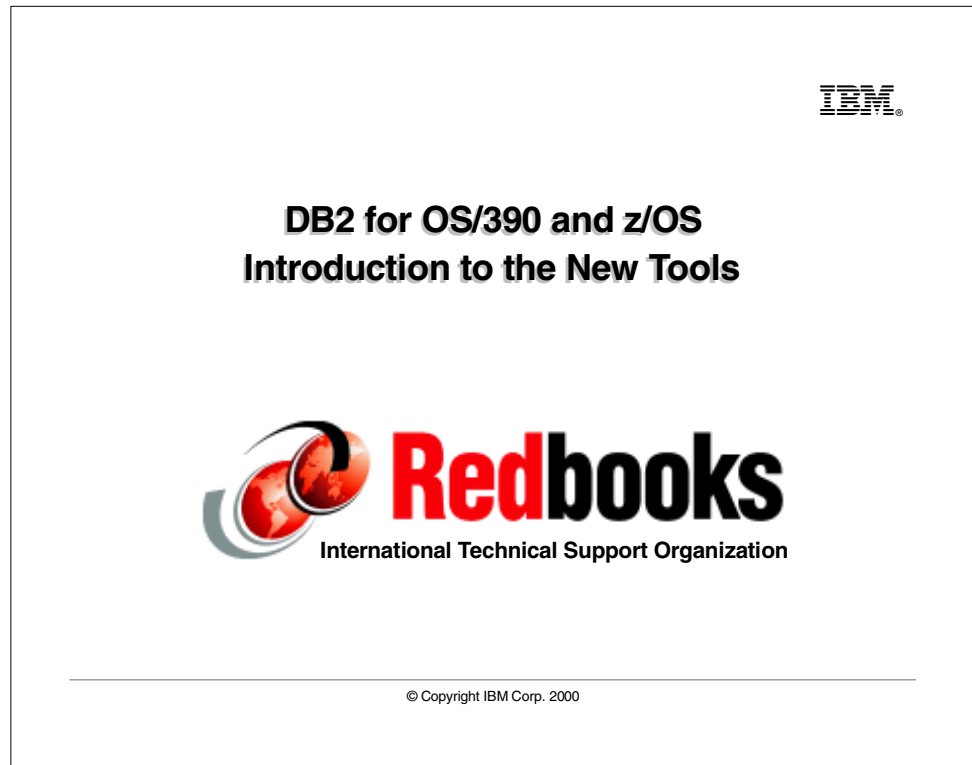
- **DB2 Control Center:** The DB2 UDB Control Center has been extended to support DB2 UDB for OS/390. The same graphical tool can now manage DB2 UDB for OS/390, UNIX, Windows, and OS/2.
- **DB2 Installer:** The DB2 Installer helps new DB2 users to install and migrate DB2 in a user-friendly fashion, and may increase the productivity of experienced installers. DB2 Installer also helps when installing DB2 Performance Monitor (PM) and Data Propagator.
- **DB2 Visual Explain:** DB2 Visual Explain is a workstation based tool that displays:
 - An easy-to-understand graph of the access paths of SQL statements
 - Catalog statistics for referenced objects from the graph
 - A list of explainable statements from plans and packages, optionally filtered by cost or access path criteria
 - Visual Explain can invoke EXPLAIN for dynamic SQL statements, provides suggestions for improving the access paths, and allows you to browse DB2's subsystem parameter values
- **DB2 Estimator:** DB2 Estimator estimates performance, resource usage, efficiency, and the cost of both existing and planned DB2 for OS/390 applications and presents them graphically.
- **Stored Procedure Builder:** The Stored Procedure Builder is a graphical tool for rapid application development on Windows that supports the coding and debugging of DB2 stored procedures written in Java and SQL Procedures language across the various platforms. It supports Java Database Connectivity (JDBC) and SQL Java (SQLJ) interfaces.

For up-to-date information on the DB2's features and contents, you can check the URL:

<http://ibm.com/software/db2os390/>

For technical details on the components of this feature, refer to *DB2 UDB for OS/390 Version 6 Management Tools Package*, SG24-5759.

Chapter 2. Introduction to DB2 tools



The Data Management utilities and tools address the most common database requirements of DB2 and Information Management System (IMS) users.

IBM has announced a new focus on database tools to complement DB2 UDB for OS/390 and IMS database engines. IBM seeks to be your preferred supplier of tools by providing competitive, high-performance, and functionally rich products.

IBM's strategy is to focus on data management tools by:

- Defining areas of importance to our customers
- Consolidating existing product offerings
- Providing a consistent set of Terms and Conditions
- Standardizing on industry consistent price methodology
- Being responsive to customer needs

This chapter provides an overview of the recently announced DB2 Tools.

The new DB2 tools at a glance

Database Administration

- DB2 Administration
- DB2 Object Restore
- DB2 High Performance Unload
- DB2 Log Analysis
- DB2 Table Editor
- DB2 Automation
- DB2 Archive Log Compression
- DB2 Object Comparison

Application Management

- DB2 Bind
- DB2 Web Query

Performance Management

- DB2 Performance Monitor
- DB2 Query Monitor
- DB2 Performance Analyzer

Recovery & Replication

- DB2 DataPropagator
- DB2 Recovery Manager
- DB2 Row Archive
- DB2 Change Accumulation



ibm.com/redbooks

© 2000 IBM Corporation

2.1 The new DB2 tools at a glance

As previously mentioned, IBM is delivering new and repositioned products for DB2 (and IMS) which can be divided into the following four categories:

• Database administration tools

They help you maximize the availability of your systems and address the most common tasks required to service and support database operations. These include such tasks as unloading, reloading, reorganizing, copying, and catalog management. Many of these are operations where performance is critical in meeting your company's availability commitments. The tools in this area are:

- DB2 Administration
- DB2 Object Restore
- DB2 High Performance Unload
- DB2 Log Analysis Tool
- DB2 Table Editor
- DB2 Automation
- DB2 Archive Log Compression
- DB2 Object Comparison

• Performance management tools

While the administration tools cover most of your maintenance and service operations, performance management tools are equally important in keeping

your database environment operating at peak performance. The tools in this area are:

- DB2 Performance Monitor
- DB2 Query Monitor
- DB2 Performance Analyzer

- **Recovery and replication management tools**

Normal backup and recovery operations are handled by the image copy and recovery tools you choose. Many customers, however, have additional requirements or situations requiring special solutions. These might include: archiving data, point in time recovery, online recovery, and added image copy capabilities. In addition, asynchronous replication of data to where it is needed can enable your distributed applications. The tools in this area are:

- DB2 DataPropagator
- DB2 Recovery Manager
- DB2 Row Archive
- DB2 Change Accumulation

- **Application management tools**

Tools helpful in the development, testing, operation, and connectivity of database applications, by providing the ability to build simple reporting applications, provide e-business connections, examine and change data, and control examine and change data, and control checkpoint activities. The tools in this area are:

- DB2 Bind
- DB2 Web Query

Database Administration



DB2 Administration Tool V2.1 (5655-E70)

- Designed for:
 - System and Database Administrator
 - application Developers
- Features includes:
 - online help
 - catalog navigation
 - reverse engineering
 - utility generation
 - alter objects definition
 - data migration
 - security administration
 - commands and dynamic SQL
 - flexibility to extend functions



ibm.com/redbooks

© 2000 IBM Corporation

2.2 IBM DB2 Administration Tool

DB2 Administration Tool Version 2.1 is the new release of DB2 Administration Tool, providing a comprehensive set of database management functions to help DB2 systems personnel manage your DB2 environment efficiently and effectively. It saves time in DB2 administration, simplifies routine day-to-day DB2 tasks, and increases knowledge and understanding of your DB2 system.

DB2 Administration has these features:

- Helps you derive the maximum performance and results from your DB2 system, dramatically reducing the effort required to support DB2.
- Provides a comprehensive set of functions that help DB2 systems personnel efficiently and effectively manage their DB2 environments.
- Offers complete DB2 catalog query and object management.
- Runs under Interactive System Productivity Facility (ISPF) and uses dynamic SQL to access DB2 catalog tables.

These are the main functions of DB2 Administration:

- Provides catalog visibility and navigation
- Lets you explore unknown databases, get a quick overview of a database, and discover database problems quickly
- Executes dynamic SQL statements
- Issues DB2 commands against database and table spaces
- Simplifies creation of DB2 utility jobs and run most DB2 utilities

- Allows you to copy tables from one DB2 to another
- Allows complex performance and space queries
- Performs the EXPLAIN function
- Performs various system administration functions, such as managing DDF, and updating Limits
- Lets you extend existing DB2 Administration applications or rapidly develop new applications using ISPF and DB2 interface
- Permits reverse engineering:
 - Recreating DDL for objects in the DB2 catalog
 - Generating Dells for underlying indexes, views, synonyms and aliases
 - Generating authorization statements for the objects
 - Adjusting space allocations
 - Changing the name of the owner
 - Changing the database name

Version 2 of DB2 Administration has two new functions:

- *Alter* supports modification of tables and their attributes.
- *Migrate* provides facilities to copy data and objects to other DB subsystems.

Version 2 of DB2 Administration has two major enhancements:

- Sort and search capabilities are improved.
- Installation-defined line commands are supported.

Recent maintenance has introduced further enhancements in the areas of restartability for Alter and Migrate, space management functions, and the possibility of saving and restoring parameters. DB2 Admin is also providing the option to launch other DB2 tools that have an ISPF interface.

Database Administration



DB2 Object Restore Tool V1 (5655-E72)

- Restores previously dropped objects, including dependencies
- Objects creation
 - includes depended objects
 - performs binds
 - performs data restore
 - restores authorization
- Managed through Catalog copy
 - changes only after first copy
 - easy navigation through Catalog versions
 - usable for all objects



ibm.com/redbooks

© 2000 IBM Corporation

2.3 IBM DB2 Object Restore

IBM DB2 Object Restore can automatically restore previously dropped objects and all related dependencies. DB2 Object Restore is superior to other tools that are available in the marketplace since it eliminates the need for a duplicate shadow copy of the catalog to recover objects. This means that DB2 Object Restore saves disk space. With DB2 Object Restore you can have confidence in cleaning up your DB2 system since you can now restore discarded DB2 objects.

Database Administration



DB2 High Performance Unload V2 (5655-E69)

- Input from two sources
 - table space
 - image copy
- Multiple outputs from one invocation
- VSAM level rather than DB2
- Full use of SELECT statement
- Many options for output format
 - DSNTIAUL
 - variable (usable by LOAD)
 - delimited (workstation loaders)
 - user (custom tailored output)
- Output data types controlled by user
 - can be different than source
 - can be internal DB2 representation
 - numeric can be converted to character
- Row level User Exit
 - to modify row
 - to skip row
- Row selectivity beyond SQL
 - number of rows
 - selected intervals



ibm.com/redbooks

© 2000 IBM Corporation

2.4 IBM DB2 High Performance Unload

DB2 High Performance Unload really can make a difference to your unload management tasks. DB2 High Performance Unload offers sequential reading and accessing of your DB2 data at top speed. It can scan a table space and create output files in the format that you need. The major capabilities provided are:

- High speed unloads of DB2 data using native virtual storage access method (VSAM)
- Ability to unload from image copies as well as active tables
- Multiple output files during a single unload with minimal additional cost
- Multiple output formats, including the opportunity to tailor your own
- Comprehensive and powerful SELECT statement
- More efficient unloads of DB2 data
- Help with batch windows constraints
- Ability to unload multiple tables each to a separate file or a single table many times

Sequential reading of DB2 tables requires long periods of time. This makes it difficult to schedule unloads of large tables in the ever shrinking batch windows of DB2 installations. Large scan performance can become critical when several unloads have to read the same table space concurrently.

The performance symptoms can sometimes be caused by multiple reads of the same DB2 pages. Sometimes there are even channel conflicts.

DB2 High Performance Unload really can make a difference to your unload management tasks.

DB2 High Performance Unload offers sequential reading and accessing of your DB2 data at top speed. It can scan a table space and create output files in the format that you need. All you have to do is select the criteria. Do you want the format to be DSNTIAUL compatible? Or do you need standard variable length records? Or is your choice a delimited file for export to another platform?

You can elect almost any type of conversion, giving your output the appearance that you want. You can code as many select statements as you want for any tables belonging to the same table space. So different output files can be created during the same unload process at almost no additional cost.

You can also unload multiple tables at once, each to its own file, as long as all the tables belong to the same table space. You can even unload the same table many times with different select statements, using only one table space scan.

If you are worried that all this activity will affect DB2 production, do not be. You can run your DB2 High Performance Unload against image copies (incremental or any full image copy) as well as active tables. So DB2 production databases are unaffected.

Database Administration



DB2 Log Analysis Tools V1.1 (5655-E66)

- Simple ISPF interface
- Automatic JCL generations
- Reads DB2 Logs and DB2 pages directly
- Compressed tables support
- Filtering using various criteria
- Generates UNDO/REDO SQL
- Targeted restoration without table space unavailability
- Report generations
 - summary with drill-down capability
 - can be saved as JCL for execution
- Audit capability



ibm.com/redbooks

© 2000 IBM Corporation

2.5 IBM DB2 Log Analysis Tool

IBM DB2 Log Analysis Tool provides the DB2 administrator with a powerful tool to ensure high availability and complete control over data integrity. It allows you to monitor data changes at a glance by providing the facilities to:

- Automatically build reports of changes made to database tables
- Specify reports by various database resource criteria, such as date, user, or table
- Quickly isolate accidental or undesired changes to your database tables
- Avoid the processing overhead often associated with data change monitoring
- Heighten confidence in data integrity while keeping DB2 for OS/390 systems at optimal efficiency
- Maximize uptime for e-business availability

Recent enhancements have added data sharing support, new summary reports, and additional filtering options.

Database Administration



DB2 Table Editor V4.1 (5697-G65)

- Table Editor and RAD tool designed with DBA and Developers needs in mind
- Designed for Windows and Java users
 - drag and drop
 - full screen table editor
 - forms with command buttons
 - wizards
- Direct access to DB2 on multiple platforms
- Centralized administration
 - version control
 - user permissions
- Referential Integrity support
- Development module for Rapid Application Development
- Console module for centralized administration
- User module provides run-time environment for any DB2 Forms application
- Java Player module for serving Java-based DB2 Table Editor applications to Web browsers



ibm.com/redbooks

© 2000 IBM Corporation

2.6 IBM DB2 Table Editor

DB2 Table Editor, program number 5697-G65, is the new generation of DB2 Forms, program number 5697-G52. It is IBM's multipurpose table editing environment that offers database administrators, developers, and the entire enterprise, direct update and data creation operations on DB2 UDB for OS/390 and z/OS databases from within Java, ISPF or Windows-based interfaces. With DB2 Table Editor you can create Windows front-end applications.

DB2 Table Editor enables developers to quickly construct new applications, often in just minutes with a drag-and-drop interface. Add buttons, labels, text boxes and controls for containing data and drop-down lists. Behaviors, data sources and data validation rules are assigned to controls from easy-to-use dialogs that require no programming. Applications can be centrally stored for user access at the database server, and periodically updated and improved at will. It then provides users with access to finished applications.

DB2 Table Editor applications are stored centrally at the DB2 server and then launched from any Windows workstation. End users select applications from the catalog of custom forms. Each application presents specific data associated with it at development time. Typical applications include table editing, inventory or product catalog access, order entry, customer invoice retrieval, or query-by-example front ends. Access is available to local or remote locations, including users connecting from any location via the Internet to TCP/IP supported DB2 databases.

Administration functions enable administrators to set up user groups, permissions for those groups, and time-of-day and day-of-week schedules. Applications can be bound to their respective databases; user groups and permissions for users and groups can be configured. Governing settings are stored centrally at the database server and allow administrators to make entire applications (or just specific capabilities across all applications) unavailable to selected groups.

These are the main functions of DB2 Table Editor:

- Build applications and graphical user interfaces to any DB2 data warehouse or DB2 operational data.
- Create controls, data validation rules, and application behaviors within a drag-and-drop environment.
- Build in advanced database techniques and commands without programming or SQL knowledge.
- Satisfy universal requirements, such as transactions, table editing, QBE, and data entry.
- Slash development time while creating applications that set new standards for performance.
- Distribute finished applications freely like a browser to in-house or remote users.
- Set up users in minutes without database gateways, middleware, or ODBC drivers.
- Applications can connect directly to databases over the Internet (including, via dial-up to a local ISP).
- Restrict user/application permissions and track user activity at the server with centralized governing.
- Roll out with TCP/IP or SNA connectivity and full DB2 security support.
- Use IBM's DB2 Data Joiner to include multi-vendor data sources, such as IMS, VSAM, Oracle, Informix, Sybase, Microsoft SQL Server, and more.

Whether using table layouts in the full screen table editor, wizards, or forms with command buttons rapidly built in the DB2 Table Editor drag-and-drop development environment, both Windows and Java-based users now can have direct access to multiple DB2 database tables on multiple platforms, including OS/390, VSE and VM, and Windows workstation databases.

DB2 Table Editor includes enhanced data editing and referential integrity capabilities, full screen table editing interface, and new form components. It continues to provide a robust table editing and database *front end* building environment that offers:

- Reading and writing directly to IBM DB2 UDB database tables, through a choice of connectivity options
- Transparent cross-platform support for multiple IBM DB2 UDB database platforms, versions, and native DB2 security
- Rapid building of business and data validation rules into table editing forms, without programming or compiling

- Centralized management and administration, providing excellent concurrency and control over user permissions, database access, and versioning of table editing forms
- Server-based licensing for distribution to unlimited users
- The most rapid path in the industry to providing controlled, direct data operations upon DB2 tables from within Windows, Java-based workstations, or Web browsers.

Database Administration



DB2 Automation Tool V1 (5697-G63)

- Automation of COPY and REORG based on user specified thresholds
- Runstats history with trend analysis and optional forecasting reports
 - it uses the new DB2 V7 history table if available
- DB2 data page browser in hexadecimal



ibm.com/redbooks

© 2000 IBM Corporation

2.7 IBM DB2 Automation Tool

DB2 Automation Tool for S/390 and z/OS, program number 5697-G63, helps you realize the full potential of your DB2 system by continuously and automatically coordinating the execution of DB2 tools on an around-the-clock basis. The primary capabilities provided are:

- Automatic execution of DB2 tools against specified objects
 - Image Copy
 - REORG
 - RUNSTATS
- Manual, periodic, or rules-based execution of any number of tools, with a combination of parameters and options
- Easy creation of job specifications as *profiles* that may be Updated, Deleted, and Imported or Exported across subsystems and test or operational environments
- Development of job profiles through intuitive ISPF panels and special syntax without needing JCL skills
- Support that allows multiple database administrators to securely maintain their own sets of profiles

Now, without relying on repeated manual interventions, every DB2 administrator is able to add maximum value to the enterprise by extracting full performance constantly from even the most heavily-used database environment.

Database Administration



DB2 Log Archive Compression Tool (5655-F54)

- Lowers auxiliary storage cost
- Allows DB2 logs to be kept on disks
- Can reduce the size up to 95 %



ibm.com/redbooks

© 2000 IBM Corporation

2.8 IBM DB2 Archive Log Compression Tool

DB2 Archive Log Compression Tool for S/390 and z/OS, program number 5655-F54, makes highly compressed off-site copies of DB2 logs allowing administrators to reduce the volume of archived logs resulting in:

- Shorter I/O and recovery times
- Lower storage costs
- Making storage of DB2 logs on DASD affordable in many cases instead of tape

Archive Log Compression uses high performance compression technology designed especially for DB2 logs to maximize the:

- Reduction of your offload and retrieval time
- Value of your storage media

Administrators may elect to keep SQL UNDO entries in the log to be compressed and achieve compressions that may exceed 40 percent. Or they may choose to have DB2 Archive Log Compression remove SQL UNDO (which is not needed for disaster recovery) in order to obtain even higher rates of compression. The tool provides disaster recovery support by restoring directly from the compressed logs.

Database Administration



DB2 Object Comparison Tool (5697-G64)

- Allows comparisons of DB2 objects
 - Catalog vs. Catalog
 - DDL file vs. DDL file
 - DDL vs. Catalog
- Reports on differences
- Generates and migrates changes from a source to a target



ibm.com/redbooks

© 2000 IBM Corporation

2.9 IBM DB2 Object Comparison Tool

DB2 Object Comparison for OS/390 and z/OS, program number 5697-G64, helps you keep your test and development system as a mirror image of the production system. New applications, application modifications, or mistakes can cause DB2 objects in one of these systems to have different attributes than on other systems. DB2 Object Comparison Tool allows you to compare objects, and dependent objects, from one source to another. Once a difference file is generated, the product can be used to generate the DB2 commands needed to bring the catalogs back into synchronization.

Masking and ignore files are supported to account for intentional differences and/or naming conventions that exist between the two sets of objects to be compared. For example, primary and secondary quantities usually are different between a test and production system. Likewise, the same object might have an owner name of TESTxxx on the test system and an owner name of PRODxxx on the production system. Use of the mask and ignore files allow you to compare only on real differences that might exist.

Version files are used for comparison. A version file is created from either a file of DDL or from objects in a DB2 catalog. The *known-to-be-correct* source version file is compared to a target version file that may be back level. Using version files as a base you may compare DDL, catalog object definitions, and other version files in any pair-wise combination desired.

The ability to do comparisons with previously generated version files gives you the opportunity to:

- Restore application objects to a previous version (backout)
- Compare a new version with several production versions (clones) of the objects

DB2 Object Comparison runs as an extension to DB2 Administration Tool Version 2.1 and consists of:

- An ISPF frontend for specification of the objects to be compared
- A DB2 catalog extract function that pulls definitions from the catalog into a version file to support the compare process A DDL extract function that reads DDL statements and converts them into a version file
- A batch compare function that compares two version files, produces a report that describes the differences found and generates the information needed to apply changes to the target
- A batch job generator that provides everything necessary to apply the changes to the target

Performance Management



DB2 Performance Monitor (5655-E61)

- Analysis, control and tune performance of DB2 system and application
- Wide variety of reports for in-depth analysis
- Explain feature to analyze and tune SQL
- Data sharing groups support through single Parallel Sysplex connection
- Application programming interface
- Utility tracing facility
- Realtime online monitor
 - choice of host or work station based
 - snapshot view of DB2 activity
 - display thread activity
 - subsystem statistics
 - history facility
 - display DSNZPARM
 - exception processing



ibm.com/redbooks

© 2000 IBM Corporation

2.10 IBM DB2 Performance Monitor

DB2 Performance Monitor for OS/390 Version 7 is IBM's strategic tool for analyzing, controlling, and tuning the performance of DB2 for OS/390 systems as well as DB2 applications. DB2 PM Version 7 supports full performance monitoring and problem analysis for all functions of DB2, including the new DB2 enhancements introduced in V7. For example, DB2 PM supports all instrumentation, catalog, and PLAN_TABLE changes.

DB2 PM can be used in two main ways: batch editing and online monitoring. Online monitoring allows you to obtain a *snapshot* view of DB2 activities, while the history facility allows you to view events both recently and in a more distant past. With the workstation-based monitor, which is replacing the traditional and ISPF interface, you can monitor all your DB2 subsystems in parallel. It has interfaces to other IBM DB2 tools, such as Visual Explain for explaining SQL statements, and it can be launched by other tools such as the DB2 Control Center.

You can use the wide variety of reports provided or you can customize them for even more in-depth performance analysis.

DB2 Trace data can be stored in the DB2 Performance Monitor performance database for further investigation and trend analysis.

Threshold-based and event-based exception event processing allows you to be notified of exceptional system situations immediately via online alerts, via user exits to system monitors, such as immediately via online alerts, via user exits to system monitors, such as Netview, or during report processing.

DB2 PM Version 7 also provides an application programming interface (API) to the Online Monitor data collector. Now you can retrieve performance information about the subsystem and the applications running on it and pass it to an application program. You can obtain raw data and derived performance information including snapshot information as well as recent history data. This includes exception alerts based on DB2 events and thresholds. DB2 PM Version 7 enhancements include:

- Data Sharing (Sysplex) Monitoring Online with group scope view
- Dynamic SQL statement cache monitoring
- Buffer pool data set statistics
- New Event exceptions:
 - Activity Log data set full
 - Activity Log data set full
 - Data set extent activity
 - Units of recovery in flight/in doubt

DB2 Performance Monitor continues to provide you with a powerful set of functions to do your daily work in analyzing, controlling, and tuning your DB2 environment.

Performance Management



DB2 SQL Performance Analyzer (5697-F57)

- Forecast SQL performance
 - response times
 - CPU times
 - I/O counts
- Expert advice to improve SQL
- Warn users of long running query
- Evaluate future production volume performance
- Illustrate incremental components of cost
- Governing for any DB2 application
- Helps the design of new queries
- Assists in tuning SQL via DBRM scans
- Resolves database and index design problems
- Identify poor use of predicates and clause
- Uncovers changing data patterns affecting performance
- Prevents runaway query before cancellation
- Eliminates prototyping and stress testing



ibm.com/redbooks

© 2000 IBM Corporation

2.11 IBM DB2 SQL Performance Analyzer

DB2 SQL Performance Analyzer for OS/390 Version 1 delivers performance analysis for all phases of database application design and development. Developing applications under DB2 today requires the cooperative skills of the system architect, the database administrator, the programming staff, and members of the systems support team. In the daily quest to produce applications on time and within budget, too often corners are cut such that performance is not adequately considered.

Once put into production, these applications fail to perform adequately, particularly at high transaction volumes. In addition, even applications that originally performed well may deteriorate over time due to changes in data distribution, data volume, system changes, and even changes to the DB2 product itself. Therefore, installations spend a great deal of their budget managing the performance of their applications.

All DB2 problem queries have one thing in common. They run too long. These queries cause the batch production window to shrink. Too often the online queries take what seems like forever to execute, causing customers and users to become frustrated. The most cost effective solution to the problem is prevention. IBM is introducing DB2 SQL Performance Analyzer to aid in preventing queries from running too long. With this tool you can find out how long queries will take:

- Before you run them
- Before resources are consumed
- Before the query exceeds your installations governor settings

Query cost can be determined regardless of which DB2 attach is used and regardless of whether static or dynamic SQL is used. Estimates are given in familiar units like CPU time, I/O count, and elapsed time and in even simpler terms, such as a single number representing overall cost. In addition, a monetary cost for each query is computed and delivered.

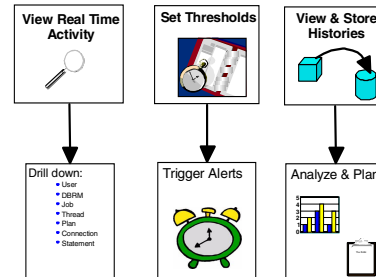
Recent enhancements have added the cost analysis for indexes other than those chosen by the optimizer and the interface to DB2 Bind Manager.

Performance Management



DB2 Query Monitor (5655-E67)

- Captures performance data in real time
- Pinpoint problematic plans in seconds
- Threshold based alerts and actions
- Quickly and accurately terminates problems
- Low memory/CPU overhead
- Easy drill down navigation



ibm.com/redbooks

© 2000 IBM Corporation

2.12 IBM DB2 Query Monitor

DB2 Query Monitor for OS/390 Version 1 helps you to maximize DB2 availability. Query monitoring facilities can generate mountains of data, most of which may be insignificant for your specific purposes. The key to generating information that is useful to you is to customize what is gathered during the query monitoring process and to make sure that activity of interest grabs your attention.

IBM DB2 Query Monitor provides extensive choices in determining what data is gathered during activity monitoring, when it is gathered, about what database resources, and then what alerts and corrective action should be executed.

Monitoring agents, that can be started and stopped dynamically, use menu-driven criteria to watch up to 64 DB2 subsystems. As defined by administrators, data gathered is offloaded at intervals from memory to storage. IBM DB2 Query Monitor provides powerful, real-time views into the query processing events occurring in your enterprise OS/390 environment.

Recovery and Replication



DB2 DataPropagator V7 (5655-E60)

- Key replication solution for Data Warehouse and distributed database
- Replicates across diverse platforms
- Enables sophisticated data transformation
 - derive
 - aggregate
 - convert
 - consolidate
- High performance log-based change capture component
- Support of heterogeneous replication via DataJoiner technology
- Support of update-anywhere type of scenario with strong conflict resolution and automatic compensation
- Subscription administration using DB2 Control Center
- Enables replication with occasionally connected mobile databases (satellites)



ibm.com/redbooks

© 2000 IBM Corporation

2.13 IBM DB2 DataPropagator

DB2 DataPropagator for OS/390 Version 7 provides a key replication solution for data warehousing and other distributed database environments. Use it for highly efficient maintenance of consistent copies of relational data in the DB2 family of databases, automatically capturing and applying data changes.

It can help you leverage your data assets for decision making by enabling sophisticated data transformation. DB2 DataPropagator provides a powerful replication capability for the DB2 family of databases. The need to keep multiple copies of the same data in separate physical databases grows as you implement data warehouses and e-business. Data replication is an essential technology for putting timely enterprise data into the hands of your mobile worker.

DB2 DataPropagator, the core component of IBM's replication solution, unites your distributed relational databases into a cohesive and integrated database solution. It automatically captures your data changes in a source database and propagates those changes to any specified target database, keeping the two consistent. With DB2 DataPropagator, you can support these activities:

- **Re-engineering business processes**

With DB2 DataPropagator, you can replicate transactional data to servers across your enterprise. You can also improve availability and responsiveness by moving data and applications to the point of each business transaction. Many replication products support the subsetting of data only according to information that is contained in the replicated data. Unlike these products, DB2 DataPropagator subsets data based on join predicates or subselects, allowing you to distribute data efficiently from normalized databases.

- **Going mobile**

DB2 DataPropagator supports the unique needs of mobile users and occasionally connected systems and accommodates the infrequent, unpredictable, and expensive connections from these systems. Specifically, DB2 DataPropagator enables on-demand replication, automates connection and disconnection, and minimizes connection time. DB2 DataPropagator also lets you initiate all data transfers from the mobile units and infrequently connected users. Your mobile users can download data from a central server or upload data for consolidated processing.

- **Building powerful distributed applications**

DB2 DataPropagator supports mobile computing, one component of a new breed of distributed applications. The update-anywhere replication capability of DB2 DataPropagator provides rigorous conflict detection and automatic compensation for offending transactions. This capability helps you maintain the integrity of your primary database and its many distributed replicas.

- **Improving your decision-making effectiveness**

DB2 DataPropagator enables you to tailor data for maximum usability, letting you automate data enhancement as the tool copies data to the target table. For example, you can do the following tasks:

- Derive data by using arithmetic, Boolean operators, or any valid SQL expression
- Aggregate data to produce sums or averages by using SQL column functions
- Convert data by translating encoded fields to descriptive fields
- Consolidate data through joins or unions
- Generate histories to support trend analysis

- **Managing replication with a graphical interface**

An intuitive graphical user interface simplifies definition of replication scenarios including sources, targets, frequency and timing, replication-events, and pre-change and post-change processing. DB2 DataPropagator automatically creates and loads target tables. You can add, delete, or change replication requests while the rest of the system continues to run. See Figure 9 on page 24, which depicts how you can select settings for your replication sessions.

- **Minimizing impact on production systems and networks**

DB2 DataPropagator uses a log-based change-capture technique that minimizes impact on transaction performance, therefore avoiding contention with source tables and inline transaction processing. DB2 DataPropagator has optimization features to support various networked environments. You can specify distribution timing on a copy-by-copy basis. This action minimizes use of peak network periods or allows you to take advantage of economy network prices.

- **Integrating mixed database environments**

DB2 DataPropagator expands the range of solutions available to you by supporting an open architecture. In particular:

- DB2 DataPropagator uses standard SQL to leverage the database engine for data enhancement, network connectivity, and data security.
- The data staging function supports interoperability among heterogeneous sources and targets, between relational and non relational formats, and among products from independent software vendors.
- DB2 DataPropagator replication directly supports multifilament sources and targets through DataJoiner, IBM's multi database server product.

- **DB2 DataPropagator and IBM's data replication solution**

DB2 DataPropagator establishes the base architecture for IBM's data replication solution that is based on individual components that work together. As changes occur in the source, the *Capture* component stores them in the staging table. The *Apply* component reads the staging area and applies those changes to targets, or copies data directly from the source in full-refresh mode. The *Administration* component provides a user interface for defining replication requests. DataRefresher and DataPropagator Non Relational (IMS) facilitate replication of non relational data from enterprise servers. The components can populate the data staging area for DB2 DataPropagator with IMS or VSAM data. DB2 DataPropagator then enhances, distributes, and applies data to the target tables to provide an end-to-end replication solution from multiple database sources.

New with this version is support of UNICODE and ASCII encoding schemes, which minimizes the need for data conversion in the replication environments.

Recovery and Replication



DB2 Row Archive Manager V1 (5655-E65)

- Complete solution to select, archive, manage and retrieve aged data
- Large databases
 - manage cost of DASD
 - maintain data access performance and availability
- Choice of retention
- Row and column level granularity
- Supports Referential Integrity
- Easy access to archived data



ibm.com/redbooks

© 2000 IBM Corporation

2.14 IBM DB2 Row Archive Manager

DB2 Row Archive Manager for OS/390 can save storage, improve performance, simplify maintenance, and reduce the overall costs of your DB2 environment. DB2 RAM provides a simple method to control the separation of aged data from your active DB2 data. You can use DB2 RAM to move seldom used data to a less costly storage medium.

DB2 Row Archiving Manager gives you a facility for separating aged data from active data, and archiving the aged data onto a less costly storage medium. The archived data can be selectively retrieved on demand. Archiving aged data results in less active storage requirements and less active data for DB2 to process. This can mean lower cost and better performance for your DB2 environment.

DB2 Row Archive Manager implements a set of rules specified by the administrator to determine what data is eligible to be archived. It allows data to be selected at a low level of granularity, for example, at a row level. This allows the administrator to precisely control which aged data is archived. Only selected rows are archived at any specified time and not all columns of a row need to be archived, rows from related tables can be archived as a unit. Individual tables or systems of related tables can be archived.

DB2 Row Archive Manager archives selected aged data into archive table spaces. It manages a catalog used to determine how to retrieve the aged data if it is needed by an application.

Besides, this tool performs storage management functions to effectively manage the physical storage used by the archived table spaces.

It also includes additional utilities, such as the REMOVE utility, which can remove erroneous or obsolete archive specifications from the system.

Recovery and Replication



DB2 Recovery Manager V1 (5697-F56)

- Coordinates the recovery of both DB2 and IMS
- Can be used to recover individually DB2 or IMS objects
- Eliminates complexity of managing different logs
- Automates generation of JCL and controls their execution
- Speeds up overall recovery time
- Uses Virtual Image Copy



ibm.com/redbooks

© 2000 IBM Corporation

2.15 IBM DB2 Recovery Manager

DB2 Recovery Manager for OS/390 simplifies and coordinates the recovery of both DB2 and IMS data to a common point, cutting the time and cost of data recovery and availability. It eliminates the error-prone complexity of managing different logs, utilities and processes to do recovery from both databases.

Many businesses use both DB2 and IMS in their online transaction environment. An application can access data in both databases. When the application commits, IMS and DB2 coordinate the data changes so that all changes occur or none occur. However, if at some later time, you need to recover both IMS and DB2 to the same point, then you must deal with different logs, different utilities, and different processes to do the recovery. This leads to complex recovery scenarios that are time-consuming and error-prone. Each product must have its data recovered separately. The DB2 Recovery Manager is a new feature that simplifies this process.

The DB2 Recovery Manager works with IMS, DB2, or both. The DB2 Recovery Manager uses image copies for either product or both products. The tool processes the individual logs and works with incremental image copies for DB2 and the output from the change accumulation utility for IMS. Recovery Manager establishes synchronization points for the recovery. Recovery Manager calls such a synchronization point a virtual image copy. In the situation as described, after the application runs, you invoke the DB2 Recovery Manager to establish a virtual image copy. The DB2 Recovery Manager does not require image copies.

The DB2 Recovery Manager establishes a quiesce point and records that point on the respective logs. During recovery the user specifies the need to recover to

this virtual image copy. The DB2 Recovery Manager applies the appropriate image copies and causes the database to apply the log to this point. If you prefer not to use virtual image copies for recovery, you can use the DB2 Recovery Manager to automate the recovery of resources for either DB2 or IMS. The DB2 Recovery Manager generates the JCL, locates the proper image copies, and controls execution of the jobs.

Recovery and Replication



DB2 Change Accumulation Tool (5655-F55)

- Recovery with little or even no Log Apply phase
- Creates full image copies SHRLEVEL REFERENCE with no overhead or data locking
- Facilitates point in time recovery



ibm.com/redbooks

© 2000 IBM Corporation

2.16 IBM DB2 Change Accumulation Tool

DB2 Change Accumulation for OS/390 and z/OS, program number 5655-F55, provides DB2 administrators with a powerful tool for restoring database objects in the most precise and least disruptive manner possible by:

- Making precise point-in-time recovery of database objects simple and reliable
- Allowing recovery routines to focus on single objects and previous states
- Producing SHRLEVEL REFERENCE image copies without the associated overhead and data locking
- Controlling the scope and specificity of image copy creation precisely via control cards
- Maintaining data integrity without recovery to RBA
- Reducing recovery session times significantly in many cases
- Providing low overhead and minimizing downtimes for high-volume, complex databases with large numbers of tables and dependencies

Application Management



DB2 Bind Manager V1.1 (5655-D38)

- Automatically analyze BIND impact and determines if BIND is required
- Organized in three functions
 - BIND Manager, determines if BIND is required
 - DBRM Checker, checks consistency between DB2 subsystem and DBRM library
 - Path Checker forecasts access path changes before BIND execution



ibm.com/redbooks

© 2000 IBM Corporation

2.17 IBM DB2 Bind Manager

Use of DB2 Bind Manager can pay major dividends in change management, since it will automatically detect production application changes requiring a bind. This frees DBAs from analyzing bind impacts and allows them to concentrate only on the changes that affect the SQL structure.

Consistency checking may be done between an existing DBRMLIB and a DB2 subsystem using the DBRM Checker function of DB2 Bind Manager. DBRM Checker will identify DBRMs by plan that have consistency tokens that are inconsistent with those in the DB2 catalog tables. You can determine which of the DBRMs you need to bind. An application may have hundreds of packages (that is, DBRMs) making up one plan. If you only changed one, then you only need to bind that one package.

Using the Path Checker function of DB2 Bind Manager, you can quickly determine whether a bind of a DBRM will result in a changed access path. This is usually done when a new release (or version) of DB2 is installed, service is applied and/or you are migrating a large application from one system to another. You've spent many hours fine tuning the SQL for performance only to have the optimizer select a path you didn't expect. By using Path Checker, you can see the effects of doing a bind (or having done a bind). Path Checker does an EXPLAIN into your plan table of the *new* DBRM and gives you a report on those that changed. You can then use any EXPLAIN tool to look at the result and determine whether you need to take action or not.

Application Management



DB2 Web Query V1 (5655-E71)

- Enables users to access, create, store, share and execute SQL query from their Web browser
- Support DB2 servers on various platforms
- DB2 servers made available by administrator
- Compliant with DB2 security
- Result set can be viewed or downloaded in
 - TXT format
 - CSV format
 - XML



ibm.com/redbooks

© 2000 IBM Corporation

2.18 IBM DB2 Web Query

Connecting to enterprise data most likely involves a large effort: complex configurations, obscure procedures, and in depth user training are necessary preparation steps in order to access, and finally be productive using, enterprise data.

The resulting time, effort, expense and, above all, waiting creates a gap between opportunity and action that cost your organization every time it happens, perhaps thousands of times a day.

DB2 Web Query tool changes this old paradigm and helps eliminate the pain previously associated with data access. With DB2 Web Query tool, end users and administrators have a single, powerful tool for bringing data access into the e-business age with speed, reliability and simplicity.

DB2 Web Query tool sets a new standard for business responsiveness because now anyone in your organization can take robust data access for granted, virtually anywhere and at anytime. Less latency; less waiting; and less waste means more opportunity for you and your organization.

With the trusted architecture of DB2, the DB2 Web Query tool enables pervasive connectivity over the Internet to every desktop from the novice user to the expert.

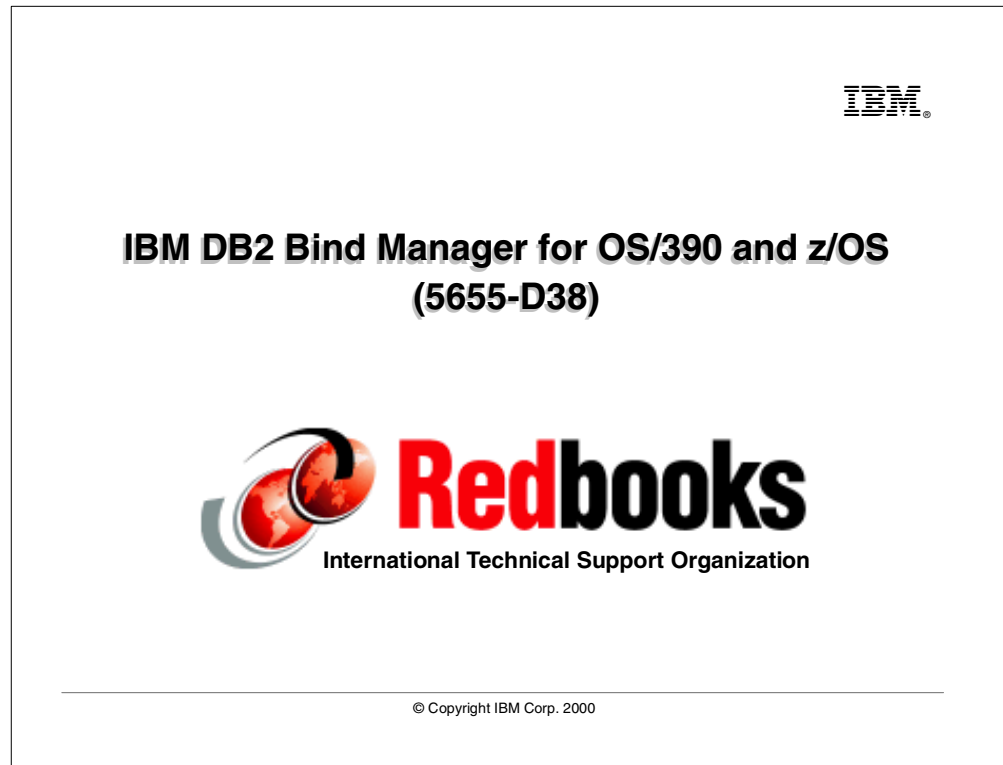
Part 2. IBM DB2 Bind Manager

In this part of the book we provide:

- An overview
- Structure and configuration
- Usage and consideration

of IBM DB2 Bind Manager for OS/390 and z/OS.

Chapter 3. Overview



DB2 Bind Manager is an application productivity tool designed to improve and streamline the development, testing and implementation of DB2 applications. It attains this goal by alleviating one of the main bottlenecks in the life cycle of an application, that bottleneck is *the DB2 BIND*.

Bind Manager allows application developers to safely bypass the bind process when their existing application programs do not alter the SQL structure, to verify DBRMs against the DB2 Catalog, and to perform binds on only those that require it and provide information about potential access path changes before they occur.

To address these issues the tool has been developed into three interrelated components:

- DB2 Bind Manager
- DBRM Checker
- Path Checker

Individually, the components are useful tools in their own right, but as the saying goes, the whole is greater than the sum of its parts. When used throughout the application development life cycle, the benefits start to accumulate.

Overview

DB2 Bind Manager objectives

- Eliminate unnecessary application rebind
 - Remove the requirement to rebind application when source code changes but SQL code does not
- Reduce unnecessary DB2 catalog access
 - DBRM consistency token checking against plans/packages by direct catalog access.
 - Removal of compulsory rebind reduces catalog read and write I/O, extremely useful in a data sharing environment
- Determine access path selection without rebind
 - DBRM's can be compared against Plan_Table data with automatic detection of access path changes.
- Reduced cost and improved reliability
 - Reduce the overhead during development and implementation
 - Fewer Bind failures



ibm.com/redbooks

© 2000 IBM Corporation

3.1 DB2 Bind Manager objectives

The main purpose of Bind Manager is to eliminate unnecessary application rebind and the effects that are associated.

Each component is designed to minimize the effect a rebind has on the application and the DB2 catalog. This is achieved in a number of ways: by removing the requirement for a rebind during pre-compile, by confirming that the DBRM consistency token matches the DB2 catalog before performing a rebind, and by providing current and potential access path changes without invoking BIND.

By delivering this functionality the product addresses a number of problems. The cost associated by performing multiple REBIND during application development, the problem of rebinding an invalid DBRM against a currently valid plan or package and detection of access path change during rebind. By reducing the requirement to rebind we also gain the reduction in CPU, the requirement to quiesce active plans or packages when promoting new programs.

The tool can also be used as a migration aid between DB2 releases to highlight where existing DBRMs would choose new access paths based on the new optimizer code.

Overview

DB2 Bind Manager objectives vs. components

- Eliminate unnecessary application rebind
 - Bind Manager
- Reduce unnecessary DB2 catalog access
 - Bind Manager
 - DBRM Checker
 - Path Checker
- Determine access path selection without rebind
 - Path Checker
- Reduced cost and improved reliability
 - Bind Manager
 - DBRM Checker
 - Path Checker



ibm.com/redbooks

© 2000 IBM Corporation

3.1.1 Objectives versus components

From the outset of its development, Bind Manager has been designed to minimize any DB2 Catalog access. By helping reduce all unnecessary access to the catalog the product achieves a number of benefits, some of these translate into CPU costs saving and others into greater application stability.

Here we have mapped the objectives of Bind Manager against each of its components and now explain how it achieves them.

With Bind Manager removing the requirement to rebind for applications that haven't changed their SQL since the last compilation, the Catalog doesn't need to be accessed and CPU cycles burnt by BIND.

DBRM Checker improves application productivity and reliability by verifying which DBRM requires binding against a target subsystem, therefore, only binding the DBRMs which require it and ensuring DBRMs which would cause invalidation not to be bound. This is achieved outside of DB2 and does not require an active subsystem.

Last but not least, Path Checker is designed to ensure stable application performance by ensuring all access paths are explained and can be continually tested to see if a rebind would have a positive or negative effect. It provides this without requiring the rebind itself, helping to reduce catalog access and ensuring only known access paths are used.

Path Checker also helps to ensure application stability between DB2 releases by its use as a migration aid, highlighting any differing access paths chosen by the old and new optimizer.

Chapter 4. Structure and configuration

DB2 Bind Manager

Components

- Bind Manager
 - Replacement module for DB2 pre-compiler.
 - Removes requirement to REBIND if previous SQL statements match in precompile
 - Existing compilation and change control procedures will require modification to benefit from bind management
- DBRM Checker
 - Reads DBRMLIB and DB2 catalog directly and performs consistency check
 - Reports on plan and package validity, versioning, precompiler timestamp and DBRM/catalog mismatches
- Path Checker
 - Utilizes primary and secondary Plan_Tables to provide access path information by reading DBRM's and invoking explain without the need for rebind.



ibm.com/redbooks

© 2000 IBM Corporation

In this chapter we look at the main components of Bind Manager and provide considerations on its installation and customization.

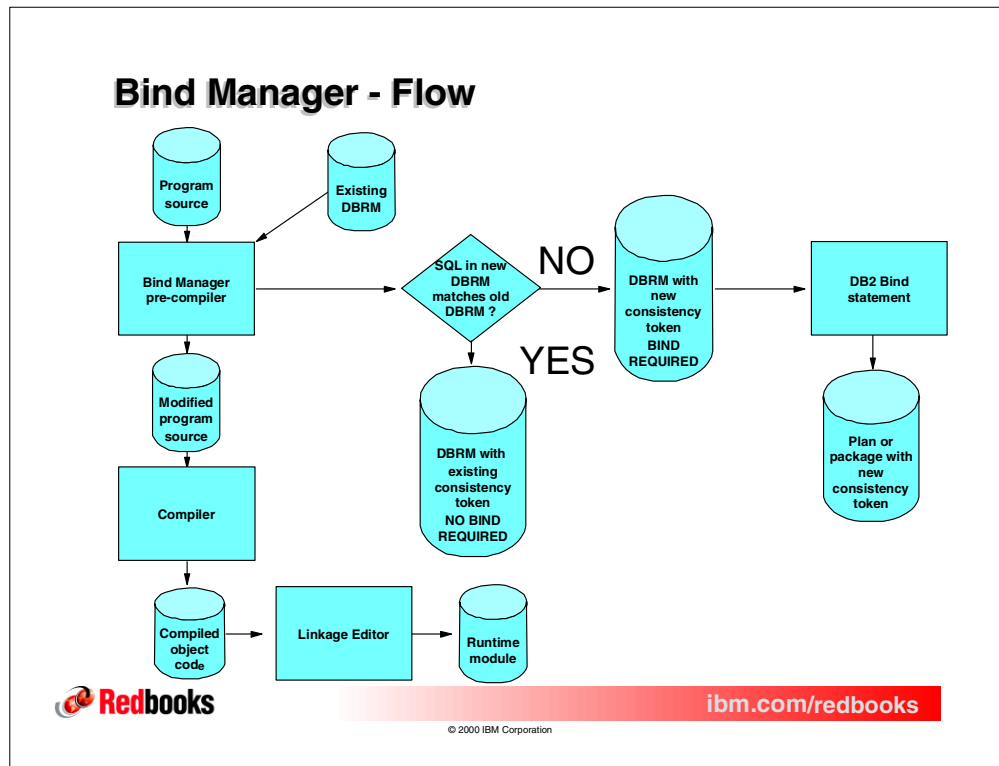
As standard for IBM products, Bind Manager is a SMP/E installed product which requires minimal effort to install and customize. After using the SMP/E apply process, the following actions are required to complete the installation of Bind Manager.

After the successful installation of Bind Manager and to make use of the product, you will be required to enable the product, bind the product into DB2 and modify or create new compilation procedures. We also suggest that you APF authorize the program library, though this is optional.

To enable the product you are required to modify the LIBDEF keyword in the install clist to correctly invoke your install library names. The install clist is stored in member BNDC001 within the SBNDJCL library.

You will then be required to bind the Path Checker into any DB2 subsystem that you want to utilize the product on. Some sample JCL is provided in the SBNDJCL library to bind the plan and grant authority.

You will then need to replace all invocations of DSNHPC with the Bind Manager precompiler module BNDAVB. Before changing your existing environment, we suggest you create duplicates and modify these for your initial testing.



4.1 Bind Manager - Flow

The Bind Manager component of Bind Manager for OS/390 helps control when BIND, or to be more accurate when REBIND, is required. This is achieved by replacing the DB2 precompiler module DSNHPC with the Bind Manager module BNDAVB, the module provides the same functionality as the pre-compiler with a few subtle changes.

When invoked BNDAVB reads in both the source code and the previous DBRM (if it exists in the DBRMLIB) and issues a call to the DB2 precompiler. Once the standard precompiler has produced a new DBRM, it performs a comparison between the previous DBRM and the newly generated DBRM. If the SQL matches, the new DBRM is created but with the existing consistency token. By utilizing the existing consistency token, the requirement to rebind the plan or package is removed.

Precompiler report - Bind required

Display Filter View Print Options Help

```
-----
SDSF OUTPUT DISPLAY TESTER  JOB07767  DSID   114 LINE 675      COLUMNS 02- 81
COMMAND INPUT ===>                                SCROLL ==> CSR
BND002I OLD DBRM INFORMATION :
      TIMESTAMP : CHAR=2000.11.11.00.17.21.369045  HEX=169DA4F4 0269F554 DEC=
      USERID=PAOLOR2
BND003I NEW DBRM INFORMATION :
      TIMESTAMP : CHAR=2000.11.15.19.08.56.266751  HEX=169E6580 1044FFE8 DEC=
      USERID=PAOLOR2
BND005E DBRM COMPARE FAILED,  SOURCE MODIFICATIONS HAVE CHANGED THE CONTENT OF T
OLD DBRM RECORD:
      | C4C2D9D4 0000004A 00000000 0000011F | DBRM...φ..... |
      | 00000014 0000002E C4C5C3D3 C1D9C540 | .....DECLARE |
      | E3C5D3C5 F140C3E4 D9E2D6D9 40C6D6D9 | TELE1 CURSOR FOR |
      | 40E2C5D3 C5C3E340 5C40C6D9 D6D440E5 | SELECT * FROM V |
      | D7C8D6D5 C5400000 00004040 40404040 | PHONE .... |
NEW DBRM RECORD:
      | C4C2D9D4 00000056 00000000 0000011F | DBRM..... |
      | 00000014 0000003A C4C5C3D3 C1D9C540 | .....DECLARE |
      | E3C5D3C5 F140C3E4 D9E2D6D9 40C6D6D9 | TELE1 CURSOR FOR |
      | 40E2C5D3 C5C3E340 5C40C6D9 D6D440E5 | SELECT * FROM V |
      | D7C8D6D5 C540D8E4 C5D9E8D5 D640F1F0 | PHONE QUERYNO 10 |
BND098S BIND MANAGER PROCESSING TERMINATED,      !!! BIND REQUIRED !!!
```



ibm.com/redbooks

© 2000 IBM Corporation

4.1.1 Bind required

In this section, we first discuss how Bind Manager emulates the existing precompiler when it detects SQL changes, then we look at how it functions when no SQL change is detected.

Bind Manager provides reporting information during the precompile indicating what action it has taken, how it determined what action to take and what is required by the BIND process. The preceding illustration is an example of the reporting information provided by Bind Manager when it determines a change in the SQL code. The diagnostic information provided in the sysout report always begins with the old and new consistency tokens written to the new DBRM. There are two easy methods for determining which consistency token has been written to the DBRM, they are:

- By checking the return code from the precompile step (0 indicates the same consistency token will be written and a RC of 4 indicates a new one)
- By comparing the consistency token reported in BND003I against the one reported in BND002I.

If the SQL has changed, then Bind Manager will produce further diagnostic information including a mini hex dump of the DBRM, indicating the first change found. In the preceding example, notice that QUERYNO has been added to the source and has been detected by Bind Manager's comparison process.

Bind Manager then issues a BND098S, indicating that an application rebind is required.

Precompiler report - No Bind required

```
Display Filter View Print Options Help
-----
SDSF OUTPUT DISPLAY TESTER  JOB08452  DSID   114 LINE 674      COLUMNS 02- 81
COMMAND INPUT ==>                                SCROLL ==> CSR
BND001I BIND MANAGER STARTED FOR DBRJIB8BC3 XX
BND002I OLD DBRM INFORMATION :
      TIMESTAMP : CHAR=2000.11.15.19.08.56.266751  HEX=169E6580 1044FFE8  DEC=
      USERID=PAOLOR2
BND003I NEW DBRM INFORMATION :
      TIMESTAMP : CHAR=2000.11.20.23.53.39.722467  HEX=169F369F 0DB338D5  DEC=
      USERID=PAOLOR2
BND004I PROCESSING COMPLETED SUCCESSFULLY,          *** BIND NOT REQUIRED ***
***** BOTTOM OF DATA *****
```



ibm.com/redbooks

© 2000 IBM Corporation

4.1.2 No bind required

If the Bind Manager precompiler detects that no SQL changes have occurred since the last precompile, the DBRM written to the DBRMLIB contains the existing consistency token. This is indicated by the BND004I message and the RC of 0 from the precompiler.

You will probably be thinking as we did when first presented with the tool: Shouldn't the new timestamp match the old timestamp in the display if a bind is not required?

The reason they are different is that the precompiler *had* to build a new DBRM. The comparison and timestamp is displayed from this newly built DBRM. It is only after the build that Bind Manager determines which token will be written out to the DBRMLIB.

[illegible]

© 2000 IBM Corporation

For all the skeptics, we confirmed the DBRM and its token by using ISPF EDIT, setting the hexadecimal display on, and performing a hex find for the old token (as gleaned by the BND002I message and the catalog).

By using standard SMF diagnosis tools, you will then be able to analyze the effectiveness of the Bind Manager and keep history on the changes that occur.

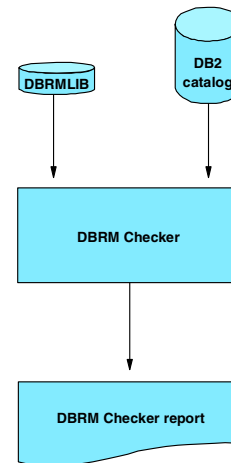
DBRM Checker - Purpose

Input

- DBRMLIB
- DB2 catalog
- Plan name(s) for comparison

Output

- Plan and package validity, catalog and DBRM consistency tokens



ibm.com/redbooks

© 2000 IBM Corporation

4.2 DBRM Checker - Purpose

The DBRM Checker provides reporting information from the DB2 catalog and the DBRMLIB to verify plan and package consistency against the DBRMLIB. Its purpose is to verify DBRM versioning against the catalog and help in determining which DBRMs require a rebind during application migration. If used in conjunction with the Bind Manager precompiler, this tool helps identify the number of rebinds required on target DB2 subsystems during application migration.

The tool is operated by executing JCL with two parameters, the DB2 catalog VCAT and a wildcard prefix of any plans you wish to compare against the DBRMLIB. To reduce the overhead against an active subsystem, the tool accesses the catalog directly using VSAM I/O routines, the checker program allocates the catalog dynamically by utilizing the VCAT and by following the standard naming protocol for the catalog objects.

With the tool accessing the DB2 catalog outside of DB2 and not via the buffer pool, you will need to take the following into account. If you suspect recent application BIND activity for the plans you want to query, they may have not been externalized yet from the buffer pool to the catalog objects, so you may have to wait for a system checkpoint or quiesce the catalog.

However, the reason you will probably be utilizing the DBRM Checker, will be to compare DBRMs before performing a BIND statement, so this event will be highly unlikely.

DBRM Checker - JCL

```
File Edit Edit_Settings Menu Utilities Compilers Test Help
=====
EDIT      BND110.SBNDJCL(BNDCKBTS) - 01.02          Columns 00001 00072
Command ==>                                     Scroll ==> CSR
***** ***** Top of Data *****
000001 //BNDCKBZZ JOB ,BINDCK,CLASS=A,MSGCLASS=X,MSGLEVEL=(1,1),
000002 // NOTIFY=&SYSUID
000003 //*****
000004 /** THIS JCL will check a subsystem utilizing          *
000005 /** a DB2 VCAT of DB2V610Y                               *
000006 /** all Plans with a prefix of DSN will be reported on *
000007 /** and any DBRM's in the plans or associated          *
000008 /** collections will be compared against the DBRMLIB    *
000009 /** coded on the DBRMLIB DD                               *
000010 //*****
000011 //STEP1 EXEC PGM=BNDCKB,PARM='DB2CAT(DB2V610Y),PLAN(DSN*)',
000012 //          REGION=0M
000013 //STEPLIB DD DSN=BND110.SBNDLOAD,DISP=SHR
000014 //DBRMLIB DD DSN=DSN610.SDSNDBRM,DISP=SHR
000015 //SYSPRINT DD SYSOUT=*
000016 //SYSUDUMP DD SYSOUT=*
000017 //
***** ***** Bottom of Data *****
```



ibm.com/redbooks

© 2000 IBM Corporation

4.2.1 DBRM Checker - JCL

The above JCL shows how simple the tool is to invoke, yet does not indicate to the submitter just how useful the tool really is. Although beneficial when using the standard DB2 precompiler, the DBRM Checker becomes almost essential once the Bind Manager precompiler is implemented and you begin to only REBIND when necessary.

This will become more apparent when viewing the output from the tool with the JCL from above and the test system used for this exercise.

Unfortunately, the DBRM Checker does not yet support concatenation on the DBRMLIB DD, though this is only a minor problem and should be resolved in a later release of the code.

DBRM Checker - Output

```

Display Filter View Print Options Help
-----
SDSF OUTPUT DISPLAY ENDCKBZZ JOB08738 DSID 101 LINE 1 COLUMNS 02- 81
COMMAND INPUT ==> SCROLL ==> CSR
END101I DBRM CHECKER STARTED FOR PLADSN* XX
END102I DB2CAT=DB2V610Y PLANS PROCESSED=00000017 DBRMS PROCESSED=00000021
PLAN=DSNTIA61 VALID=Y OPERATIVE=Y
    DBRM=DSNTIAD * DBRM/PACKAGE NOT FOUND IN DBRMLIB *
        DB2 TIMESTAMP=2000.02.14.21.55.41.253782 HEX=1673320C 0F3C25A9
PLAN=DSNESPCS VALID=Y OPERATIVE=Y COLLECTION-ID=DSNESPCS
    DBRM=DSNESM68
        DB2 TIMESTAMP=1991.12.19.07.01.51.774713 HEX=149EEA90 1A79FE48
        PDS TIMESTAMP=1991.12.19.07.01.51.774713 HEX=149EEA90 1A79FE48
PLAN=DSNWZP VALID=N OPERATIVE=Y COLLECTION-ID=DSNWZP
    DBRM=DSNWZP * MISMATCH BETWEEN CATALOG AND DBRMLIB *
        VERSION-ID=UQ33984
        DB2 TIMESTAMP=1999.08.19.23.33.21.021338 HEX=1657130B 1A716688
        PDS TIMESTAMP=2000.03.30.17.15.58.449525 HEX=167A3CB9 10E2DD40
        VERSION-ID=UQ42009
PLAN=DSNEDCL VALID=Y OPERATIVE=Y COLLECTION-ID=DSNEDCL
    DBRM=DSNECP68
        VERSION-ID=V6R1
        DB2 TIMESTAMP=1998.10.26.19.22.55.943681 HEX=16285ECE 1EBA0078
        PDS TIMESTAMP=1998.10.26.19.22.55.943681 HEX=16285ECE 1EBA0078
        VERSION-ID=V6R1
    
```



ibm.com/redbooks

© 2000 IBM Corporation

4.2.2 DBRM Checker - Output

As you can see from the output, we have a number of matches and mismatches between the Catalog and the DBRMLIB. It is important to note that the DB2 catalog is the primary source for most of the output displayed and provides the validity on the objects searched. The reasons for this are fairly obvious, because it contains your existing plan/package validity and consistency token information.

Until the DBRM Checker supports multiple DDs for the DBRMLIB, you will need to be wary of any application versioning control methods you may have implemented on your system, particularly if you move DBRMs from library to library.

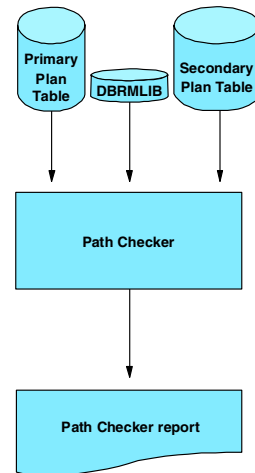
Path Checker - Purpose

Input

- DB2 SSID
- Report type
- DBRMLIB
- Authid
- Plan or collection name
- Secondary plan table name
- Program/package name
- Reporting options

Output

- Access path information with new and old comparison



© 2000 IBM Corporation

ibm.com/redbooks

4.3 Path Checker - Purpose

The DB2 Path Checker is designed to identify changes to access path selection without requiring the package or plan to have to go through the BIND process. By avoiding this requirement, both existing and new DBRMs can be analyzed and reported on without affecting the DB2 subsystem.

Path Checker achieves its goal by reading the DBRM directly and invoking DB2 Explain dynamically against the SQL, because the Explain executes with optimizer at runtime, and against the current catalog statistics, any change in access path selection can be spotted before it occurs. This also includes change brought about by maintenance or even a completely new release of DB2.

Path Checker utilizes PLAN_TABLEs for the explained output and supports all versions of the Plan Table (up to V6), this includes comparing differing release level plan tables against each other.

As previously mentioned, Path Checker is a useful tool in its own right, but becomes more beneficial when used in conjunction with the Bind Manager precompiler and DBRM Checker.

With Bind Manager removing the requirement of enforced bind, the user has more control of when bind occurs and how it will affect the access path. By utilizing DBRM Checker to confirm when rebinds are actually required on the target subsystem, you can feed the changed DBRM's into Path Checker and determine the effect on the access path.

It does all of this without requiring an actual BIND on the target system!

```

File Edit Edit_Settings Menu Utilities Compilers Test Help
#####
EDIT      BND110.SBNDJCL(BNDPTHCK) - 01.05      Columns 00001 00072
Command ==>                                         Scroll ==> CSR
***** Top of Data *****
000001 //BNDPTHCK JOB ,TSCICS,CLASS=A,MSGCLASS=T,REGION=8M,
000002 // NOTIFY=&SYSUID
000003 //STEP1      EXEC PGM=BNDPTHCK,
000004 //      PARM='ssid,Request,creator-id,secondary plan table name,plan or collid,dbnm
name,ALL,SHORT'
000005 //STEPLIB DD      DSN=DSN610.SDSNLOAD,DISP=SHR
000006 //      DD      DSN=BND110.SBNDLOAD,DISP=SHR
000007 //SYSPRINT DD      SYSOUT=*
***** Bottom of Data *****

```



© 2000 IBM Corporation

The Path Checker program can be invoked in three different modes (six if using packages).

- Report: Path Checker reads the existing Plan_table and presents the output.
- Compare: Path Checker compares the primary plan table against a secondary plan table.
- Test: Path Checker invokes Explain against a DBRM, populates the plan table then performs a comparison against the Explain data in the secondary plan table.

Full parameter descriptions are listed in the *DB2 Bind Manager for OS/390 User's Guide Version 1*, SC27-0899. An example of JCL and output follow.

If they do not already exist, you will need to define the primary and secondary Plan tables. Path Checker supports all versions of the plan table up to V6.

We will now discuss the operating modes of Path Checker in sequence, starting with reporting.

Path Checker - Report JCL

```
File Edit Edit_Settings Menu Utilities Compilers Test Help
=====
EDIT      BND110.SBNDJCL(BNDPTHCK) - 01.05          Columns 00001 00072
Command ==>                                         Scroll ==> CSR
***** ***** Top of Data *****
000001 //BNDPTHCK JOB ,TSCICS,CLASS=A,MSGCLASS=T,REGION=8M,
000002 // NOTIFY=&SYSUID
000003 //STEP1 EXEC PGM=BNDPTHCK,
000004 // PARM='DB2Y,RPTPLAN,PAOLOR2,,JIB8BH61,JIB8BC3,ALL'
000005 //STEPLIB DD DSN=DSN610.SDSNLOAD,DISP=SHR
000006 // DD DSN=BND110.SBNDLOAD,DISP=SHR
000007 //SYSPRINT DD SYSOUT=*
000008 //STEP2 EXEC PGM=BNDPTHCK,
000009 // PARM='DB2Y,RPTPLAN,PAOLOR2,,JIB8BH61,JIB8BC3,,SHORT'
000010 //STEPLIB DD DSN=DSN610.SDSNLOAD,DISP=SHR
000011 // DD DSN=BND110.SBNDLOAD,DISP=SHR
000012 //SYSPRINT DD SYSOUT=*
***** ***** Bottom of Data *****
```



ibm.com/redbooks

© 2000 IBM Corporation

4.3.2 Path Checker - Report JCL

The reporting parameters are positional and although not mandatory are required, or unexpected results may occur. Path Checker uses the creator-id parameter to determine the Plan_table which to read for the explained output, and it defaults to detailed reporting unless SHORT is coded. All of the required parameters are listed on the example JCL.

If the corresponding plan table is not populated with the selected plan or packages Explain data, then the program will return no information. This may occur for plans or packages that have not been bound with EXPLAIN(YES). Under normal circumstances, this would be a problem and require a REBIND, but this can be overcome when executing Path Checker in test mode (see discussion in 4.3.4.1, “Test mode - Report short” on page 66).

For the purpose of this execution we will assume that the plan or package have been bound with Explain and the table is populated, also that the reader has a reasonable knowledge of Explain and the data it places within the plan table.

Path Checker - Report (short)

Display Filter View Print Options Help

SDSF OUTPUT DISPLAY BNDPTHCK JOB09725 DSID 102 LINE 0 COLUMNS 01- 134

COMMAND INPUT ==>

SCROLL ==> CSR

***** TOP OF DATA *****

BND031I EXECUTING LICENSED BIND MANAGER

BND201I INPUT PARAMETERS	JIB8BC3	PROGNAME
BND201I INPUT PARAMETERS	SHORT	ACCEPTED
BND201I INPUT PARAMETERS	DB2Y	DB2 SUBSYSTEM ID
BND201I INPUT PARAMETERS	RPTPLAN	REQUEST
BND201I INPUT PARAMETERS	PAOLOR2	SQLID
BND201I INPUT PARAMETERS		COMPARISON PLAN_TABLE
BND201I INPUT PARAMETERS	JIB8BH61	APPLNAME
BND201I INPUT PARAMETERS	JIB8BC3	PROGNAME

11	2000/11/28	SQL ID - PAOLOR2	ACCESS PATH FOR JIB8BC3	APPLNAME - JIB8BH61	OLD PLAN_TABLE -
IN QRYNO M	CREATOR	TNAME	TBNO AC MC CREATOR	ACCESSNAME	IO SORTUJGO LK PF QBNO FLNO MXSQ
100 0	DSN8610	DEPT	2 R 0		N NNNNNNNN IS S 1 1
100 0	DSN8610	EMP	1 R 0		N NNNNNNNN IS S 1 2
200 0	DSN8610	EMP	1 R 0		N NNNNNNNN IS S 1 1
200 0	DSN8610	DEPT	2 I 1 DSN8610	XDEPT1	N NNNNNNNN IS 1 2
300 0	DSN8610	EMP	1 R 0		N NNNNNNNN IS S 1 1
300 0	DSN8610	DEPT	2 I 1 DSN8610	XDEPT1	N NNNNNNNN IS 1 2
400 0	DSN8610	EMP	1 R 0		N NNNNNNNN IS S 1 1
400 0	DSN8610	DEPT	2 I 1 DSN8610	XDEPT1	N NNNNNNNN IS 1 2



ibm.com/redbooks

© 2000 IBM Corporation

4.3.2.1 Basic reporting

When reporting on applications using the short report format, Path Checker provides only basic access path reporting. This can be seen in the preceding report output. The original idea behind this reduced reporting was to display the explained row on a single line in the report, we suggest that you avoid short reporting as it only supplies limited information.

For a translation of report headings into plan table field names, please refer to Appendix A, "Bind Manager" on page 219

Path Checker - Report (detailed)

```

SDSF OUTPUT DISPLAY REPORTAL JOB10190 DSID 101 LINE 0
COMMAND INPUT ==>
***** TOP OF DATA *****
BND031I EXECUTING LICENSED BIND MANAGER
BND201I INPUT PARAMETERS JIB8BC3 PROGNAME
BND201I INPUT PARAMETERS ALL ACCEPTED
BND201I INPUT PARAMETERS DB2Y DB2 SUBSYSTEM ID
BND201I INPUT PARAMETERS RPTPLAN REQUEST
BND201I INPUT PARAMETERS PAOLOR2 SQLID
BND201I INPUT PARAMETERS COMPARISON PLAN_TABLE
BND201I INPUT PARAMETERS JIB8BH61 APPLNAME
BND201I INPUT PARAMETERS JIB8BC3 PROGNAME

2000/11/29 SQL ID - PAOLOR2 ACCESS PATH FOR JIB8BC3 APPLNAME - JIB8BH61 OLD PLAN_TABLE -
IN QRYNO M CREATOR TNAME TBNO AC MC CREATOR ACCESSNAME IO SORTUJGO LK PF QBNO PLNO MXSQ
100 0 DSN8610 DEPT 2 R 0 N NNNNNNNN IS S 1 1
ACC-DEG 0 ACC-PGP 0 JN-DEG 0 JN-PGP 0 SRTC-PGP 0 SRTN-PGP 0 PRLL-MD MRG-JCL 0 PG-RNG JN-TYP WH-OPT
100 0 DSN8610 EMP 1 R 0 N NNNNNNNN IS S 1 2
ACC-DEG 0 ACC-PGP 0 JN-DEG 0 JN-PGP 0 SRTC-PGP 0 SRTN-PGP 0 PRLL-MD MRG-JCL 0 PG-RNG JN-TYP WH-OPT
200 0 DSN8610 EMP 1 R 0 N NNNNNNNN IS S 1 1
ACC-DEG 0 ACC-PGP 0 JN-DEG 0 JN-PGP 0 SRTC-PGP 0 SRTN-PGP 0 PRLL-MD MRG-JCL 0 PG-RNG JN-TYP WH-OPT
200 0 DSN8610 DEPT 2 I 1 DSN8610 XDEPT1 N NNNNNNNN IS 1 2
ACC-DEG 0 ACC-PGP 0 JN-DEG 0 JN-PGP 0 SRTC-PGP 0 SRTN-PGP 0 PRLL-MD MRG-JCL 0 PG-RNG JN-TYP WH-OPT

```



ibm.com/redbooks

© 2000 IBM Corporation

4.3.2.2 Detailed reporting

As you can see, the detailed reporting contains more information but wraps the output over a couple of lines, which makes the report slightly harder on the eyes. This needs to be taken in context though. Would you prefer a few relatively simple lines to read or have to struggle and scroll several times in reading the output via tools like SPUFI?

A slight inconsistency in the reporting is the fact that the second report line is displayed across the report and not in column style like the first line. This makes it slightly more difficult to read and interpret. This is especially true for fields in the plan table that contain blank values.

Path Checker - Compare JCL

```
File Edit Edit_Settings Menu Utilities Compilers Test Help
#####
EDIT      BND110.SENDJCL(BNDPTHCK) - 01.06          Columns 00001 00072
Command ==>                                         Scroll ==> CSR
***** ***** Top of Data *****
000001 //BNDPTHCK JOB ,TSCICS,CLASS=A,MSGCLASS=T,REGION=8M,
000002 // NOTIFY=&SYSUID
000003 //*
000004 //STEP1 EXEC PGM=BNDPTHCK,
000005 // PARM='DB2Y,COMPPLAN,PAOLOR2,PLAN_TABLE2,JIB8BH61,JIB8BC3,,SHORT'
000006 //STEPLIB DD DSN=DSN610.SDSNLOAD,DISP=SHR
000007 // DD DSN=BND110.SBNDLOAD,DISP=SHR
000008 //DERMIN DD DISP=SHR,DSN=DB2LOGA.SALADBRM(ALASQL6)
000009 //SYSPRINT DD SYSOUT=*
000010 //*
000011 //STEP2 EXEC PGM=BNDPTHCK,
000012 // PARM='DB2Y,COMPPLAN,PAOLOR2,PLAN_TABLE2,JIB8BH61,JIB8BC3,ALL'
000013 //STEPLIB DD DSN=DSN610.SDSNLOAD,DISP=SHR
000014 // DD DSN=BND110.SBNDLOAD,DISP=SHR
000015 //DERMIN DD DISP=SHR,DSN=DB2LOGA.SALADBRM(ALASQL6)
000016 //SYSPRINT DD SYSOUT=*
***** ***** Bottom of Data *****
```



ibm.com/redbooks

© 2000 IBM Corporation

4.3.3 Compare JCL

The compare option is probably the least utilized of the options of Path Checker, because it requires the definition of a secondary plan table and the requirement to propagate data to the secondary table after a successful BIND. More often Path Checker will be either invoked in report mode or test mode.

This will become more apparent as compare mode is explained.

When invoked in compare mode the program reads the creator-id.plan_table and the secondary plan table. The program determines the name of the secondary table by using the parameters supplied (creator-id and secondary plan table name).

It then performs searches on the plan tables for the correct DBRM name and performs a like for like comparison using the query numbers to match the statement.

By reading both plan tables it assumes that the secondary plan table contains the previous explained data, and this means that after your last successful bind (with Explain), you need to copy the Explain data from the real plan table to the secondary plan table.

So a bit of intervention and additional work is needed to compare the two explains.

However, this is not a problem, because the developers have addressed the situation when executing Path Checker in test mode.

Path Checker - Compare (short)

Display Filter View Print Options Help

SDSF OUTPUT DISPLAY BNDPTHCK JOB10500 DSID 101 LINE 0 COLUMNS 02 - 81

COMMAND INPUT ==>

SCROLL ==> CSR

***** TOP OF DATA *****

BND031I EXECUTING LICENSED BIND MANAGER

BND201I INPUT PARAMETERS JIB8BC3 PROGRAMNAME

BND201I INPUT PARAMETERS SHORT ACCEPTED

BND201I INPUT PARAMETERS DB2Y DB2 SUBSYSTEM ID

BND201I INPUT PARAMETERS COMPPPLAN REQUEST

BND201I INPUT PARAMETERS PAOLOR2 SQLID

BND201I INPUT PARAMETERS PLAN_TABLE2 COMPARISON PLAN_TABLE

BND201I INPUT PARAMETERS JIB8BH61 APPLNAME

BND201I INPUT PARAMETERS JIB8BC3 PROGRAMNAME

1 2000/11/30 SQL ID - PAOLOR2 ACCESS PATH FOR JIB8BC3 APPLNAME - JIB8BH61 OLD PLAN TABLE -

IN QRYNO M CREATOR TNAME TENO AC MC CREATOR ACCESSNAME IO SORTUJGO LK PF QBNO PLNO MKSQ

2 475 0 DSN8610 EMP 1 R 0 N NNNNNNNN IS 1 2

2 475 0 DSN8610 DEPT 2 I 1 DSN8610 XDEPT1 N NNNNNNNN IS 1 2

1 500 0 DSN8610 EMP 1 I 1 DSN8610 XEMP2 Y NNNNNNNN IX 1 1

2 500 0 DSN8610 EMP 1 I 1 DSN8610 XEMP1 Y NNNNNNNN IX 1 1

BND203I STATEMENTS WITH SAME ACCESS PATH 5 STATEMENTS WITH DIFFERENT ACCESS PATH 1

BND204I QUERIES WITH MATCHING EXPLAIN 6 QUERIES WITHOUT MATCHING EXPLAIN 1

COMPLETE RET CODE= 04

***** BOTTOM OF DATA *****



ibm.com/redbooks

© 2000 IBM Corporation

4.3.3.1 Compare - Basic report

As you can see from the output the comparison has detected differences in the Explain output. This is indicated by passing a return code of 4 and displaying the Explain statements that differ in the output.

In the preceding compare message BND203I, it notes that there is an SQL statement which has no match and would provide you with an indication that the last bind didn't have the same number of SQL statements as the previous bind.

BND204I is used to indicate where the query numbers are matching, but the explained rows indicate the SQL has chosen a different access path.

The advantage that short reporting provides over detailed reporting, when in comparison mode, is that only non matching Explain rows are returned. As you will see on the detailed report, all rows are returned.

Path Checker - Compare (detailed)

Display Filter View Print Options Help

```
-----
SDSF OUTPUT DISPLAY ENDPTHCK JOB10500 DSID 102 LINE 0 COLUMNS 03- 82
COMMAND INPUT ==> SCROLL ==> CSR
***** TOP OF DATA *****
BND031I EXECUTING LICENSED BIND MANAGER
BND201I INPUT PARAMETERS JIB8BC3 PROGNAME
BND201I INPUT PARAMETERS ALL ACCEPTED
BND201I INPUT PARAMETERS DB2Y DB2 SUBSYSTEM ID
BND201I INPUT PARAMETERS COMPLAN REQUEST
BND201I INPUT PARAMETERS PAOLOR2 SQLID
BND201I INPUT PARAMETERS PLAN_TABLE2 COMPARISON PLAN_TABLE
BND201I INPUT PARAMETERS JIB8BH61 APPLNAME
BND201I INPUT PARAMETERS JIB8BC3 PROGNAME
```

```
2000/11/30 SQL ID - PAOLOR2 ACCESS PATH FOR JIB8BC3 APPLNAME - JIB8BH61 OLD PLAN_TABLE -
IN QRYNO M CREATOR TNAME TBNO AC MC CREATOR ACCESSNAME IO SORTUJGO LK PF QBNO PLNO MXSQ
* 100 0 DSN8610 DEPT 2 R 0 N NNNNNNNN IS S 1 1
ACC-DEG 0 ACC-PGP 0 JN-DEG 0 JN-PGP 0 SRTC-PGP 0 SRTN-PGP 0 PRLL-MD MRG-JCL 0 PG-RNG JN-TYP WH-OPT
* 100 0 DSN8610 EMP 1 R 0 N NNNNNNNN IS S 1 2
ACC-DEG 0 ACC-PGP 0 JN-DEG 0 JN-PGP 0 SRTC-PGP 0 SRTN-PGP 0 PRLL-MD MRG-JCL 0 PG-RNG JN-TYP WH-OPT
2 475 0 DSN8610 EMP 1 R 0 N NNNNNNNN IS S 1 1
ACC-DEG 0 ACC-PGP 0 JN-DEG 0 JN-PGP 0 SRTC-PGP 0 SRTN-PGP 0 PRLL-MD MRG-JCL 0 PG-RNG JN-TYP WH-OPT
2 475 0 DSN8610 DEPT 2 I 1 DSN8610 XDEPT1 N NNNNNNNN IS S 1 2
ACC-DEG 0 ACC-PGP 0 JN-DEG 0 JN-PGP 0 SRTC-PGP 0 SRTN-PGP 0 PRLL-MD MRG-JCL 0 PG-RNG JN-TYP WH-OPT
```



ibm.com/redbooks

© 2000 IBM Corporation

4.3.3.2 Compare - Detailed report

Here is a part of a detailed report from Path Checker, and the reason for the exclusion is that most of the detail is identical. The two main differences between the reports are:

- All Explain statements are displayed, with matching columns being indicated by an * in the in column and changed or new rows being indicated by their table reference number.
- By the inclusion of the additional columns previously seen in the detailed report.

Path Checker - Test JCL

```
SDSF EDIT    TESTPLAN (JOB10614) JCLEEDIT           Columns 00001 00072
Command ==>                                     Scroll ==> CSR

***** ***** Top of Data *****

000001 //TESTPLAN JOB ,TSCICS,CLASS=A,MSGCLASS=T,REGION=8M,
000002 // NOTIFY=&SYSUID
000003 //*
000004 //STEP1 EXEC PGM=BNDPTHCK,
000005 // PARM='DB2Y,TESTPLAN,PAOLOR2,PLAN_TABLE,JIB8BH61,JIB8BC3,,SHORT'
000006 //STEPLIB DD DSN=DSN610.SDSNLOAD,DISP=SHR
000007 // DD DSN=BND110.SBNDLOAD,DISP=SHR
000008 //DBRMIN DD DISP=SHR,DSN=PAOLOR2.TESTING.DBRM(JIB8BC3)
000009 //SYSPRINT DD SYSOUT=*
000010 //*
000011 //STEP2 EXEC PGM=BNDPTHCK,
000012 // PARM='DB2Y,TESTPLAN,PAOLOR2,PLAN_TABLE,JIB8BH61,JIB8BC3,ALL'
000013 //STEPLIB DD DSN=DSN610.SDSNLOAD,DISP=SHR
000014 // DD DSN=BND110.SBNDLOAD,DISP=SHR
000015 //DBRMIN DD DISP=SHR,DSN=PAOLOR2.TESTING.DBRM(JIB8BC3)
000016 //SYSPRINT DD SYSOUT=*

***** ***** Bottom of Data *****
```



ibm.com/redbooks

© 2000 IBM Corporation

4.3.4 Path Checker - Test JCL

This is almost certainly the most useful feature of Path Checker and provides the DBA with the ability to assess the effect a BIND would have on a DBRM's access path, but it does this without performing the bind.

Path Checker achieves this by reading the DBRM via the DBRMIN DD and performing an Explain on the SQL found within. It then compares the newly created Explain data against the table specified in the secondary plan table field.

Path Checker - Test mode report short

Display Filter View Print Options Help

```
-----
SDSF OUTPUT DISPLAY TESTPLAN JOB10614 DSID 101 LINE 0 COLUMNS 02- 81
COMMAND INPUT ==> SCROLL ==> CSR
***** TOP OF DATA *****
BND031I EXECUTING LICENSED BIND MANAGER
BND201I INPUT PARAMETERS JIB8BC3 PROGNAME
BND201I INPUT PARAMETERS SHORT ACCEPTED
BND201I INPUT PARAMETERS DB2Y DB2 SUBSYSTEM ID
BND201I INPUT PARAMETERS TESTPLAN REQUEST
BND201I INPUT PARAMETERS PAOLOR2 SQLID
BND201I INPUT PARAMETERS PLAN_TABLE COMPARISON PLAN_TABLE
BND201I INPUT PARAMETERS JIB8BH61 APPLNAME

1 2000/11/30 SQL ID - PAOLOR2 ACCESS PATH FOR JIB8BC3 APPLNAME - JIB8BH61 OLD PLAN_TABLE -
IN QRYNO M CREATOR TNAME TBNO AC MC CREATOR ACCESSNAME IO SORTUJGO LK PF QBNO P1NO MXSQ
1 200 0 DSN8610 EMP 1 R 0 N NNNNNNNN IS S 1 1
1 200 0 DSN8610 DEPT 2 I 1 DSN8610 XDEPT1 N NNNNNNNN IS 1 2
2 200 0 DSN8610 EMP 1 R 0 N NNNNNNNN IS S 1 1
2 200 0 DSN8610 DEPT 2 I 1 DSN8610 XDEPT1 N NNNNNNNN IS 1 2
1 300 0 DSN8610 EMP 1 R 0 N NNNNNNNN IS S 1 1
1 300 0 DSN8610 DEPT 2 I 1 DSN8610 XDEPT1 N NNNNNNNN IS 1 2

BND203I STATEMENTS WITH SAME ACCESS PATH 1 STATEMENTS WITH DIFFERENT ACCESS PATH 2
BND204I QUERIES WITH MATCHING EXPLAIN 3 QUERIES WITHOUT MATCHING EXPLAIN 0
COMPLETE RET CODE= 04
```



***** BOTTOM OF DATA *****

ibm.com/redbooks

© 2000 IBM Corporation

4.3.4.1 Test mode - Report short

Path Checker really earns its money while running in test mode. As you can see from the report, it has detected some change, but if you read the report carefully you will not be able to find out why it determined that change.

This is why we recommend SHORT reporting only for a preliminary analysis. The program has detected a change, but it is not in one of the fields displayed on the short report.

You will be able to notice the detected change on the detailed report which follows.

Path Checker - Test mode detailed report

```

1      2000/11/30  SQL ID - PAOLR2  ACCESS PATH FOR JIB8BC3  APPLNAME - JIB8BH61  OLD PLAN_TABLE -
IN QRYNO M  CREATOR  TNAME          TENO AC MC CREATOR  ACCESSNAME          IO SORTUJGO LK PF QBNO PLNO MXSQ
*    100 0 DSN8610  DEPT              2 R  0              N NNNNNNNN  IS S    1  1
ACC-DEG  0 ACC-PGP  0 JN-DEG  0 JN-PGP  0 SRTC-PGP  0 SRTN-PGP  0 PRLL-MD  MRG-JCL  0 PG-RNG  JN-TYP  WH-OPT
*    100 0 DSN8610  EMP              1 R  0              N NNNNNNNN  IS S    1  2
ACC-DEG  0 ACC-PGP  0 JN-DEG  0 JN-PGP  0 SRTC-PGP  0 SRTN-PGP  0 PRLL-MD  MRG-JCL  0 PG-RNG  JN-TYP  WH-OPT
1    200 0 DSN8610  EMP              1 R  0              N NNNNNNNN  IS S    1  1
ACC-DEG  0 ACC-PGP  0 JN-DEG  0 JN-PGP  0 SRTC-PGP  0 SRTN-PGP  0 PRLL-MD  MRG-JCL  0 PG-RNG  JN-TYP  WH-OPT
1    200 0 DSN8610  DEPT              2 I  1 DSN8610  XDEPT1  N NNNNNNNN  IS    1  2
ACC-DEG  0 ACC-PGP  0 JN-DEG  0 JN-PGP  0 SRTC-PGP  0 SRTN-PGP  0 PRLL-MD  MRG-JCL  0 PG-RNG  JN-TYP  WH-OPT
2    200 0 DSN8610  EMP              1 R  0              N NNNNNNNN  IS S    1  1
ACC-DEG  4 ACC-PGP  1 JN-DEG  0 JN-PGP  0 SRTC-PGP  0 SRTN-PGP  0 PRLL-MD  MRG-JCL  0 PG-RNG  JN-TYP  WH-OPT
2    200 0 DSN8610  DEPT              2 I  1 DSN8610  XDEPT1  N NNNNNNNN  IS    1  2
ACC-DEG  4 ACC-PGP  1 JN-DEG  4 JN-PGP  1 SRTC-PGP  0 SRTN-PGP  0 PRLL-MD  MRG-JCL  0 PG-RNG  JN-TYP  WH-OPT
1    300 0 DSN8610  EMP              1 R  0              N NNNNNNNN  IS S    1  1
ACC-DEG  0 ACC-PGP  0 JN-DEG  0 JN-PGP  0 SRTC-PGP  0 SRTN-PGP  0 PRLL-MD  MRG-JCL  0 PG-RNG  JN-TYP  WH-OPT
1    300 0 DSN8610  DEPT              2 I  1 DSN8610  XDEPT1  N NNNNNNNN  IS    1  2
ACC-DEG  0 ACC-PGP  0 JN-DEG  0 JN-PGP  0 SRTC-PGP  0 SRTN-PGP  0 PRLL-MD  MRG-JCL  0 PG-RNG  JN-TYP  WH-OPT
2    300 0 DSN8610  EMP              1 R  0              N NNNNNNNN  IS S    1  1
ACC-DEG  4 ACC-PGP  1 JN-DEG  4 JN-PGP  1 SRTC-PGP  0 SRTN-PGP  0 PRLL-MD  MRG-JCL  0 PG-RNG  JN-TYP  WH-OPT
2    300 0 DSN8610  DEPT              2 I  1 DSN8610  XDEPT1  N NNNNNNNN  IS    1  2
ACC-DEG  4 ACC-PGP  1 JN-DEG  4 JN-PGP  1 SRTC-PGP  0 SRTN-PGP  0 PRLL-MD  MRG-JCL  0 PG-RNG  JN-TYP  WH-OPT
BND203I STATEMENTS WITH SAME ACCESS PATH          1  STATEMENTS WITH DIFFERENT ACCESS PATH          2
BND204I  QUERIES WITH MATCHING EXPLAIN          3  QUERIES WITHOUT MATCHING EXPLAIN          0
COMPLETE RET CODE=                                04

```



ibm.com/redbooks

© 2000 IBM Corporation

4.3.4.2 Test mode - Detailed report

For this test we used the same DBRM as used in the initial bind, to highlight how easy it is for an access path to change even using the same SQL. This is where Path Checker becomes a very useful tool and helps identify change where you wouldn't expect it.

The best way to describe how this product works is by this simple sentence:

If I ran my BIND now, what would happen?

This tool answers that question and does so without the bind. The benefit this brings is enormous, because you can detect any change **before** it becomes active and make a decision on what action may be required before actually performing the bind.

For our test case, we used Path Checker to determine what options the optimizer would choose for the exact same DBRM, if we changed the DB2 setup. For the example, we changed the default parameter for parallelism from 1 to ANY.

The purpose of this iss to highlight how ZPARM changes and how maintenance can effect applications when they are rebound and potentially cause a drastic change in the way the application works.

As it happens, in our example case, if the application plan was rebound it would have a positive effect!

Chapter 5. Usage and considerations

Usage and considerations

DB2 Bind Manager implementation

- Benefits versus change
- Phased approach to reduce impact
- Step 1- minimal impact
 - OS/390 changes
 - Precompiler changes
- Step 2 - closer integration
 - Changes to DBRM Checker
 - Changes to Path Checker
- Step 3 - final solution
 - Changes to precompiler
 - Changes to DBRM and Path Checker



ibm.com/redbooks

© 2000 IBM Corporation

The implementation of Bind Manager will have an impact on your system since it requires changes to the operational procedures. However, you have to consider the benefits that the tool can bring. We believe that by following the implementation proposal outlined here you will be able to assess the tool's benefits prior to full implementation. We recommend a phased approach to minimize the impact, as the adoption of a full Bind Manager solution will require changes in your existing procedures for application development and promotion to production. By implementing a poor or partial solution, Bind Manager could potentially be detrimental to application stability and performance.

By following the recommended steps we believe that the benefits that Bind Manager brings will be maximized and the impact upon your current application change management process will be reduced.

5.1 Step 1 - Minimal impact

Although the initial recommendations are designed to be as minimally intrusive as possible, a number of changes will be required to your existing procedures, but they will have no detrimental effect on your DB2 application environment.

DBRMs will be created with existing and new consistency tokens, but you will still bind all DBRMs as applications would normally. The new Bind Manager messages will appear in the precompiler output, but they will be ignored during the first stage of the implementation.

The purpose of this step is to become familiar with the tool, install it, and use the tool initially only as a reporting mechanism to evaluate benefits in the application development environment. Once adopted and verified, the tool can be deployed and fully utilized.

OS/390 system changes

The initial changes will be as minimally intrusive as possible and make the product's benefits easier to assess. After successfully installing the Bind Manager product, we recommend the following:

- APF authorization of the Bind Manager load library
- Addition of the load library to the linklist
- Agreement on a dedicated SMF record number that the product will be able to utilize

The tool can still fully function without these optional steps, however we recommend these to allow easier product installation and provide the ability to report the success or failure rate for application rebinds.

Changes for the precompiler

Once the OS/390 changes are in place we can then start to use the product as a reporting tool. Without the APF authorization and agreed SMF record number, you will not be able to implement all of the required JCL or Clist changes.

- Required Changes:
 - Change all occurrences of the module DSNHPC to BNDAVB.
 - Add a DD card of SMFnnn, where nnn is the agreed record number.
- Optional Changes:
 - If the load library was not added to linklist, change any STEPLIB or JOBLIB DDs to include the Bind Manager load library.
 - Get auditing defined in your security product for module DSNHPC. This is to ensure nobody is invoking the old module without your knowledge.

What effect will the changes have?

These initial changes will have a very minor impact on your existing development workload and should not require any change in the way they work. The reason for this is that although Bind Manager will be intervening in the precompiler process we still recommend that all packages or plans are bound.

This effectively makes the replacement precompiler redundant but allows you to report on its decision process. We achieve this by reading the generated SMF records to see what decision we could have made while still performing the BIND statement until we can make a decision on how to deploy the tool.

By following this recommendation you can start to build a trend analysis of how many times the precompiler will allow the bind to be bypassed. This analysis will then allow you to make a determination on how effective the Bind Manager precompiler will be in your development environment.

During this trial you will need to take into account how much development is occurring during this initial phase or you will have insufficient SMF data on which to base your findings. You will also need to balance the length of the trial against using the full function product, because the longer this initial analysis period is

while still invoking bind, then the longer the period is where the tool is not producing an actual reduction in the number of rebinds.

Once you are sufficiently happy with the introduction of the precompiler to your existing setup you can then proceed to the next step.

5.2 Step 2 - Closer integration

After implementing our initial suggestions your attention can then be brought into focus on the two other components of Bind Manager and how they will affect and improve your existing setup. Both of the components can be utilized separately though probably will be used in conjunction with each other.

Once you are comfortable with the DBRM Checker and its ability to determine differences between the DBRMLIB and the DB2 Catalog, you can then begin to make a decision about how to implement it into your existing application change management procedure.

Changes for DBRM Checker

As we have established earlier, the DBRM Checker performs validation of the DBRM against the DB2 Catalog. This means that we can use its validation ability when promoting applications to determine whether or not a bind should occur.

However, we initially suggest that the DBRM Checker is added to your process, but the decision on whether or not to perform a bind based on the DBRM Checkers report is ignored for now.

In your existing promotion procedures prior to the bind step, you will need to insert a DBRM Checker step.

Changes for Path Checker

As with the DBRM Checker, we suggest that you modify your existing procedures to add in the additional step, but as yet do not utilize the tool to make a decision.

To utilize the Path Checker, you will need to do the following:

- Define Plan_Table(s) for all plans which currently do not use EXPLAIN(YES).
- Insert a Path Checker step into your application promotion procedures after the DBRM Checker step.

The purpose of implementing both the DBRM Checker and the Path Checker steps together will help reduce the number of changes you have to implement, but that later on provide the ability to determine what action will be required on the BIND execution as either step may flag the DBRM as a potential problem if rebound.

We also suggest that you add a Path Checker test step to your compilation procedures to help the developers improve or spot any changes in the access path prior the applications promotion. Another suggestion is to use QUERYNO to take maximum advantage of Bind Manager. It is not critical to use it, but it does make the Path Checker diagnosis easier.

5.3 Step 3 - Final solution

After assessing the impact and understanding of how Bind Manager can be integrated into your system, the final decision must be made to start to use the tools productively.

Once these final suggestions have been implemented, Bind Manager should be functioning as designed and helping to reduce the number of issues raised when developing and promoting applications.

Final changes for precompiler

This is the crunch point where you have now finally decided to commit to using Bind Manager and to gain the benefits it produces. If you followed the initial suggestions, you will have still been performing all binds, regardless of which consistency token the DBRM has decided to write into the DBRM.

To finally gain the benefit from the precompiler, you will have to modify your compilation procedures to execute the application bind based on the return code passed from precompiler.

This will be providing the benefit in your development environment, so you will now also need to alter your change control procedures to reflect these changes.

Final Changes for DBRM and Path Checker

As with the compilation procedures, a simple decision based on the return code will allow you the ability to determine whether or not to bind the application. However, with our suggestion of placing both the DBRM Checker and Path Checker steps into your application promotion process they provide a simple but secure approach in determining whether to perform the bind.

If the both DBRM Checker and Path Checker provide a return code of 0 then you can completely bypass the BIND step.

If DBRM Checker gets a RC greater than 0 then you know that it has determined the DBRM does not match the DB2 Catalog. In our situation this will almost certainly mean that a new DBRM is being promoted and we must perform the bind. We can ignore a decision from the Path Checker return code, because we were required to perform the bind anyway.

However, should the DBRM Checker step return with 0 and the Path Checker return a RC of 4. It means the DBRM matches the Catalog but the access path has changed.

From here you have a decision to make:

- Do you automatically assume the new access path is better and perform a bind?
- Or,
- Do you bypass the bind and flag it for further investigation and possibly bind it later?

This decision probably will be based on how well you trust the design of the application being promoted and if the change had been previously noted during the compilation process.

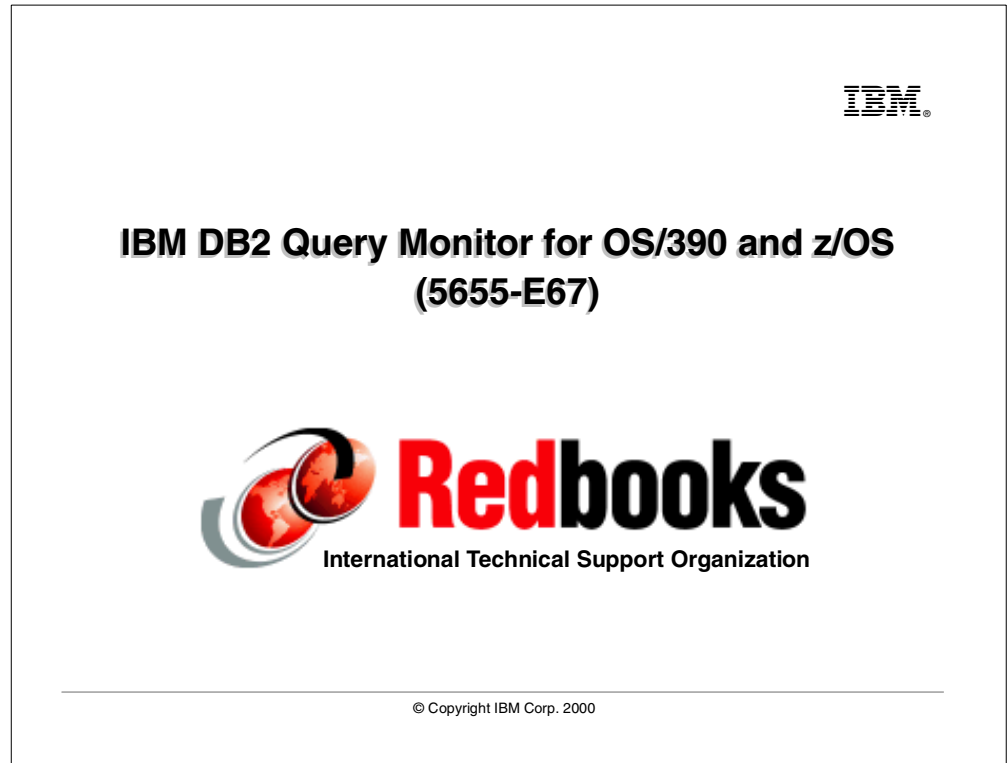
Part 3. IBM DB2 Query Monitor

In this part of the book we provide:

- An overview
- Structure and configuration
- Usage and consideration

of IBM DB2 Query Monitor for OS/390 and z/OS.

Chapter 6. Overview



This redbook presentation describes the functions and operation of DB2 Query Monitor. It will walk through the installation and initial configuration of the tool and the advantages it can bring to the work of a DBA or application developer.

The underlying principles are also documented to gain a better understanding of its operations.

This redbook presentation has been designed so that it can be used additionally as teaching or workshop material.

For reference, DB2 Query Monitor was installed on a multiple virtual storage (MVS) Sysplex with OS/390 V2R10 and DB2 for OS/390 Version 6.

Introduction to Query Monitor

Problem

Multiple DB2?
Multiple monitors?
Complicated panels?
Expensive traces?
Overhead?
Information overload?



Solution

Monitor 64 DB2s
Few simple panels
Low overhead
Information filters
Threshold processing



**DB2 Query Monitor
for OS/390**



ibm.com/redbooks

© 2000 IBM Corporation

6.1 Introduction to Query Monitor

Query Monitor is a low overhead thread monitoring tool which can concurrently monitor up to 64 DB2 subsystems. All SQL that is issued is reported. This includes both dynamic and static SQL. Commands that have been issued and utilities invoked are also detected.

Only the accounting trace is required to be started, with classes 1, 2 and 3 selected. Historical data is stored in either DB2 table spaces or on DASD.

Information on SQL statements is presented in a very user friendly way, with information on the CPU time elapsed, buffer statistics, delays and locking all immediately available.

There is no need for extensive training or familiarization with the tool, information is easily available and options are intuitive.

All information is available via ISPF panels, there is no need to run batch jobs to gather it. The only batch is for unloading and loading past information.

Query Monitor complements the functions of DB2 Performance Monitor (DB2 PM); Query Monitor provides information quickly, DB2 PM can then be used to perform a more in depth investigation if necessary.

Thresholds can be set to highlight threads which have exceeded certain limits so that quick detection is possible.

Presenting DB2 Query Monitor

Purpose of Query Monitor

- Overview of Query Monitor
- Functions of Query Monitor
- Terminology

Structure and configuration

- Installing Query Monitor
- Structure of Query Monitor
- Setting up profiles

Running and managing Query Monitor

- Monitoring DB2 subsystems
- Updating monitors
- Examples of using Query Monitor



ibm.com/redbooks

© 2000 IBM Corporation

6.2 Presenting DB2 Query Monitor

This presentation introduces the concepts and operations of Query Monitor as given here.

Purpose of Query Monitor

The functions of Query Monitor are documented together with the benefits that the product brings to DB2 installations. To aid in usage of the product and to maintain consistency in this presentation, some key terminology is also defined.

Structure and configuration

This presentation does not repeat all the installation information in the *DB2 Query Monitor for OS/390 User's Guide Version 1, Release 1, SC27-0968*, but in some areas gives additional details to aid the user in configuring the tool for the first time. We also give information about how the tool fits together, in terms of the underlying DB2 tables and how the monitoring agents interact. Instructions about how to initially set up Query Monitor is given, because some profiles need to be created to permit operation.

Running and managing Query Monitor

The following points are covered in more detail in this presentation:

- Examples and explanations of Query Monitor functions
- Instructions about how to change views of the data being captured
- How to change the data being captured by the agents in the individual DB2s
- Some useful features of DB2 Query Monitor are shown with screen shots
- How the application can be used to detect and interpret problems in DB2

DB2 Query Monitor

Major functions of Query Monitor

- Easy to use, menu driven, intuitive system
- Monitor up to 64 DB2 subsystems
- Minimum overhead for monitors
- No expensive traces necessary
- Dynamically start and stop subsystem monitoring
- Dynamically filter data collected from DB2
- Dynamically filter display of data to aid problem determination
- Pin point problems as they execute
- Terminate threads of problem queries
- Exception profiles highlight threads exceeding thresholds
- Historical data can be unloaded to data sets and reloaded for later analysis



ibm.com/redbooks

© 2000 IBM Corporation

6.3 Major features of DB2 Query Monitor

The major features of DB2 Query Monitor are documented in the following paragraphs. Examples of the features are shown and discussed in more detail in this section.

Ease of use

DB2 Query Monitor is a menu driven system with online help (by pressing F1) and meaningful error messages. All the information can be viewed and filtered from a small number of ISPF panels. There is no need for extensive training or an introduction to the system. Threads are color coded for allow for easy interpretation of the display. The coding is as follows:

Active thread: Yellow

Completed thread: Blue

Exception thread: Red (plan name and value exceeding threshold)

Monitor DB2 subsystems

DB2 Query Monitor has the ability to monitor up to 64 DB2 subsystems from one DB2 Query Monitor subsystem. The DB2s do not have to be on the same MVS LPARs to be monitored.

Low overhead

The only trace DB2 Query Monitor requires to be active is the accounting trace classes 1, 2 and 3. However, only one DB2 Query Monitor subsystem can be defined to monitor a specific DB2 subsystem.

The monitors have a very low overhead and the code works with memory structures avoiding the need to invoke OS/390 services. DB2 Query Monitor uses MVS dataspace storage to gather SQL statistical information, this can be limited by setting parameters in the startup JCL and by filtering the information collected by using Monitoring profiles.

Dynamic monitoring

Once Monitoring Agents have been defined to DB2 at start-up, the information which these agents gather about the DB2 subsystem can be dynamically changed, and the monitors can be deactivated and reactivated from a DB2 Query Monitor panel.

The data that is gathered about specific DB2 subsystems can be filtered by an application profile so that only information pertinent to a particular user or problem is displayed. This does not affect the data actually gathered by the monitor, only the data is displayed under a particular application profile. Numerous application profiles can be created which can be implemented with little effort. They can be tailored to suit the particular problem being monitored. Housekeeping functions are available to tidy up unwanted monitoring, application and exception profiles.

Thread activity is collected and displayed in real time which can greatly improve problem determination and resolution. You should note that Query Monitor will only refresh the display at the completion of each statement. For example, if the thread is executing a very long open cursor, the thread will be displayed in yellow, but it may look as if the monitoring has ceased. This is not the case. Statistics are being collected and will be updated when the open has completed.

Problem detection and determination

An Exception Profile can be assigned with suitable thresholds, so that any thread exceeding any one of the values is flagged and easily identified.

If a problem thread is identified, the ability to cancel the thread exists, with or without a dump. Note that the user must have sufficient DB2 authority in the target DB2 subsystem; DB2 Query Monitor supports DB2 security constraints.

Historical data analysis

Past activity data is written to DB2 tables, and this can be retained for further periods by unloading intervals to disk. A generation data group (GDG) is recommended with a suitably high LIMIT parameter to keep sufficient historical data.

A GDG with the following DCB parameter is required:

```
DCB= (LRECL=255,DSORG=PS,RECFM=FB)
```

There are two methods of unloading and reloading data. The first is to use the supplied Unload and Load JCL. This will unload all available data that has not been previously unloaded. It will be unloaded into one generation of the GDG. The load JCL will load all the data specified in the JCL. Data sets can be concatenated.

The other method is to use commands `U` and `L` on the past activity panel in Query Monitor to unload and load selected intervals.

DB2 Query Monitor

Terminology

- Monitoring Agent

An interface to the DB2 subsystem being monitored

- Monitoring Profile

A filter for information being gathered by the agent

- Application Profile

A filter of what the user will see from the Monitoring Profile

- Exception Profile

Determined limits when exceeded highlight the thread on the monitor

- Interval

A defined time period in which Query Monitor collects information

- Current Activity

Threads monitored within the current monitoring period

- Past Activity

Threads monitored in previous monitoring periods



ibm.com/redbooks

© 2000 IBM Corporation

6.4 DB2 Query Monitor terminology

At this point we define the terminology used throughout the presentation. This will avoid confusion later on in the presentation.

Monitoring agent

This is the interface between the DB2 Query Monitor subsystem and a target DB2 subsystem. It is initially defined at the start-up of the Query Monitor (QM) started task. There is one monitoring agent per DB2 subsystem.

Monitoring profile

This is the information that the monitoring agent gathers about the target DB2 subsystem. There is one monitoring profile assigned to each DB2, however the same monitoring profile can be assigned to more than one DB2.

Application profile

This is all or a filtered subset of the information gathered by the monitoring agent. There can be many application profiles but only one active per user.

Exception profile

Once certain user defined thresholds have been reached, the thread becomes an exception and is displayed in red on the DB2 Query Monitor screen. The exception profile is allocated at QM subsystem start-up. It can be updated interactively, but only refreshed when DB2 Query Monitor is dropped and raised. There can be many profiles but only one can be active for all DB2s.

Interval

This is a start-up parameter that determines the time frame for which DB2 Query Monitor will collect information. Once this interval has passed, Query Monitor writes the current activity from memory to DB2 tables.

Current activity

This is the DB2 activity that is being monitored in the current time interval. This information is held in memory.

Past activity

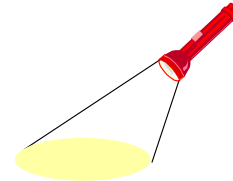
This is the information for previous time intervals. It is held in DB2 tables. The amount of past activity is determined by the user at installation time and is a combination of the number of partitions defined to the QM table spaces and the length of the monitoring interval. Once all the partitions have been used, in rotation style, the first partition is reused. Data can be offloaded to data sets for longer term storage and subsequently reloaded into history tables.

The next chapter discusses the details of the Query Monitor components and its initial configuration.

Chapter 7. Structure and configuration

Structure and configuration

- Process flow
- Installation
- Configuration
- Operation



ibm.com/redbooks

© 2000 IBM Corporation

In this chapter we describe the main components of Query Monitor and provide considerations on its installation, customization, and operation.

Installing Query Monitor

- Installation and setup
 - Start up JCL and parameter setting
This can run as a started task or batch job
 - Function of the Query Monitor control file
This defines profiles and configuration information
 - Structure and function of underlying DB2 objects
Past data held in partitioned table spaces, 1 partition per interval
Historical data can be unloaded to DASD and loaded to history database
 - Operation of Query Monitor
- Setting up of initial profiles
 - Monitoring profile and exception profile



ibm.com/redbooks

© 2000 IBM Corporation

7.1 Installing Query Monitor

This section discusses the processes necessary to set up Query Monitor, its initial configuration and how it interacts with the underlying DB2 objects.

The following charts give advice about how to configure the start up JCL, some of the steps that must be taken before the Query Monitor subsystem is raised and those that can be done afterwards.

Further information is also included to explain how some of the components of Query Monitor fit together. These components include the QM parameter file and the DB2 objects that underlie Query Monitor.

The concept of profiles is also discussed and how these profiles can be used to filter the information that Query Monitor collects.

Query Monitor start up parameters

Example JCL:

```
//CQMPROC PROC HILEVEL='DB2QM'  
//CQMPROC EXEC PGM=CQMMAIN  
//CQMPARMS DD DSN=&HILEVEL..PARMS.DATA,DISP=SHR  
//STEPLIB DD DSN=&HILEVEL..SCQMLoad,DISP=SHR  
//DB2PARMS DD DSN=CQM.DB2.CONTROL,DISP=SHR  
//SYSUDUMP DD SYSOUT=*
```

```
SUBSYS (DBQM) -  
AUTHID (PAOLR3) -  
MONITOR (DB2Y,ALLTHRD2,DBG1,ALLTHRDS,DBG2,ALLTHRDS) -  
EXPROF (LOWPROF) -  
INTERVAL (60) -
```

Specify a CQMPARMS file if exceed
100 characters in PARM list

```
//CQMPROC PROC HILEVEL='DB2QM'  
//CQMPROC EXEC PGM=CQMMAIN,  
// PARM= ('SUBSYS (DBQM) ',  
// 'AUTHID (PAOLR3) ',  
// 'MONITOR (DB2Y,ALLTHRD2,DBG1,ALLTHRDS) ',  
// 'EXPROF (LOWPROF) ',  
// 'INTERVAL (60) ')  
//STEPLIB DD DSN=&HILEVEL..SCQMLoad,DISP=SHR  
//DB2PARMS DD DSN=CQM.DB2.CONTROL,DISP=SHR  
//SYSUDUMP DD SYSOUT=*
```

← Query Monitor subsystem name
← DB2 authorization id, min DBADM
← DB2 subsystems and monitoring profiles
← Exception profile
← Monitoring interval



ibm.com/redbooks

© 2000 IBM Corporation

7.2 Query Monitor start up JCL

There are two main components of Query Monitor. One is the ISPF panels which the collected information can be viewed through and the system can be maintained through. The other is the Query Monitor started task (or batch job) which collects the information from the DB2's. The ISPF panels can be used without the started task/batch job being active, although not all the functions will be available.

The Query Monitor subsystem can run in one of two forms:

- Started Task
- Batch job under JES control

Work for this redbook has been carried out with Query Monitor running as a started task. The JCL for it needs to reside in a cataloged procedure library.

All of the parameters that can be coded are included in the *DB2 Query Monitor for OS/390 User's Guide Version 1 Release 1*, SC27-0968.

7.2.1 Controlling storage

There are a number of parameters which control the amount of storage which query monitor can use for certain gathering tasks. These parameters are optional and should be used when you want to limit the amount of storage. These are for

- Plans
- DBRMS
- SQL

These determine the size of the image block pool for the information gathered for each of these areas. If the image block pool size is not large enough, some of the captured information may be lost during the monitoring interval.

7.2.2 CQMPARMS data set

There is an MVS limit of 100 characters that can be passed via the 'PARM=' statement. This will only allow at most two DB2 subsystems to be monitored with no storage restrictions. The recommended way to code the start-up JCL is to use a parameter file coded, defined as DD CQMPARMS.

This data set needs to be created as LRECL=80 and with NUMBER OFF in the TSO profile. The continuation characters are important and the last line should also be coded with one.

7.2.3 Definition of start up parameters

The definitions of the start up parameters are reported here.

SUBSYS

The SUBSYS parameter defines the name of the Query Monitor subsystem, it is NOT the name of a DB2 subsystem.

MONITOR

The MONITOR parameter determines which DB2 subsystems are to be monitored. Monitoring agents are assigned a monitoring profile here to determine the type of information to be gathered.

Up to 64 DB2 subsystems can be monitored concurrently, there must be a profile associated with each subsystem, but an individual Monitoring Profile can be associated with many DB2 subsystems if the information to be gathered is the same.

DB2 subsystem definition

The first DB2 subsystem specified in the MONITOR parameter is where the past information is written to. The QM DB2 objects have to be created in this subsystem, the QM plan and packages bound there, and the AUTHID must have DBADM (or greater) authority.

EXPROF

The EXPROF defines the exception profile to be used to flag threads when they exceed the specified thresholds. The thread/plan is shown in red on the Query Monitor screen when it exceeds any of the thresholds. The value which has been exceeded is also changed to red. There is a further parameter, EXCEPTIONS, which defines the interval in which Query Monitor will refresh the Current Activity screen. This defaults to 30 seconds if it is not included in the startup JCL. This has proved to be a suitable interval.

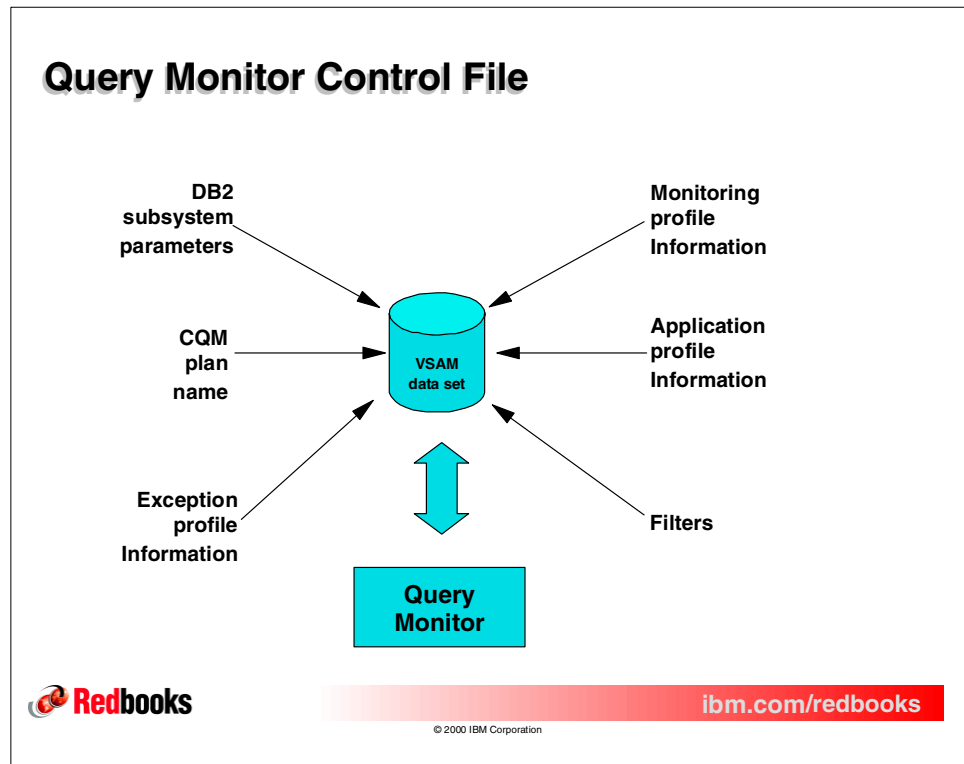
7.2.4 Other information

The load library needs to be APF authorized.

The coding of the DB2PARMS data set is not necessary in the start up JCL, because it is allocated when the Query Monitor CLIST is invoked.

Certain functions within Query Monitor can (and should) be performed without the started task running. This includes creating the initial profiles. This is done by invoking the delivered CLIST, CQMCLIST, which starts the Query Monitor ISPF panels.

Once the necessary profiles have been created, the started task or batch job can be run to start the collection of information.



7.3 Query Monitor control file

The control file is a VSAM data set which is created during the installation process. It is initially created and populated with a dummy row by supplied JCL. The control file is the data set allocated as DB2PARMS on the previous chart.

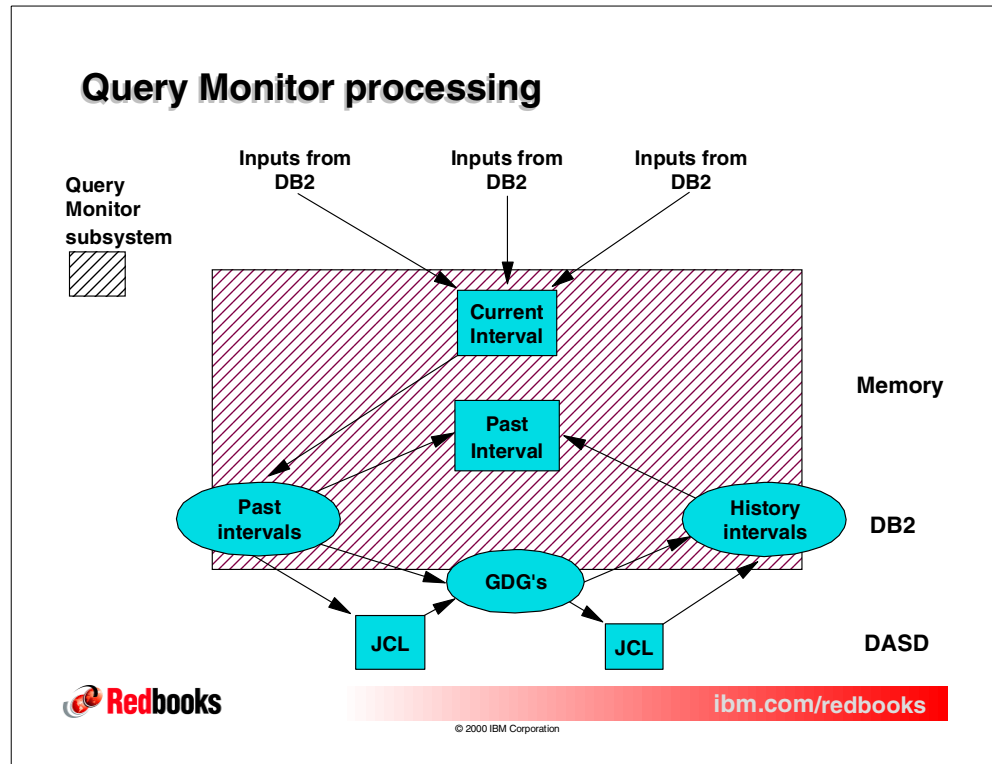
Query Monitor uses the control file to coordinate all sources of information.

It needs to be populated with at least one monitoring profile and one exception profile before the subsystem is started. This is explained in 7.4, “Query Monitor processing” on page 89.

It contains details of the DB2 subsystem that will hold the captured information and the name of the bound Query Monitor plan.

As subsequent profiles (application, monitoring and exception) are created, the control file is updated.

When in operation, details are read in from and written to this control file. Multiple control files can be created, but the profiles stored in it must match those in the startup JCL. We do not recommend that you have more than one control file, but if you do, ensure that the correct one is defined to the subsystem.



7.4 Query Monitor processing

In this section we provide an overview of the general processing flow with Query Monitor.

7.4.1 Description of operations

To minimize impact to system resources, Query Monitor records captured data to memory, to DB2 tables and to DASD (or tape).

For the current monitoring interval, the data gathered from the monitoring agents is held in data spaces in memory. This can be limited by startup parameters as discussed earlier. After the monitoring interval has completed, Query Monitor writes the information to DB2 tables in the CQMDB database. The table spaces are partitioned by monitoring interval and Query Monitor determines the least recently used partition, deletes by data currently in it, and inserts the data currently in memory. There is no observable outage when this happens.

Data is held in the CQMDB database until is overwritten in its partition, after all the other partitions have been used.

The recommended interval time is 1 hour, and 24 partitions are defined. This enables the past 24 hours of data to be kept in DB2 at any one time. It also limits the amount of captured data that is held in memory before being written out to DB2. Having one hour of data in each partition also reduces the amount of DASD space the table spaces are going to use.

Another advantage is that the current data is accessed immediately without any I/O waits. This allows for fast and effective problem determination.

7.4.2 DB2 objects

The table below gives a summary of which DB2 tables the collected data is stored in. The *DB2 Query Monitor for OS/390 User's Guide Version 1 Release 1*, SC27-0968 should be consulted for more information on sizing considerations and on how to partition the objects if more than the default 24 partitions are required.

If more partitions are required to be added, the easiest way to do this is to unload the past interval's data using Query Monitor and then to drop and recreate the objects with the additional partitions defined.

Default values for PCTFREE=5 and FREEPAGE=0 are suitable as Query Monitor performs mass inserts and deletes into a partition and does not insert individual rows during processing, (except into the non-partitioned table space INTRVL11, where one row is added during offloads and updates to the availability indicator are made. Default free space values are still appropriate).

Adding LOCKPART YES to the table space definitions may also be considered.

Information on	Database	Table space	Table
Past Intervals	CQMDB	THRDS11 BUFFPL11 DBRMS11 STMTS11	THREADS_V11 BUFFERPOOLS_V11 DBRMS_V11 STATEMENTS_V11
History	CQMDB	HISTORY	THREADS_V11 BUFFERPOOLS_V11 DBRMS_V11 STATEMENTS_V11
Intervals	CQMDB	INTRVL11	INTERVALS_V11

7.4.3 Managing DB2 objects

Once all the partitions have been populated with data, it is recommended that Runstats is run, followed by a Rebind of the packages to obtain the optimal access path.

The table spaces are created with compression enabled, REORG the table spaces when they are populated to compress the data.

Table spaces should be image copied according to installation standards.

7.4.4 Managing past data

There are two methods to retain captured data before it gets overwritten in DB2. One method is to type `U` next to the interval in the Past Activity screen.

The second method, and the most efficient, is to use the supplied JCL to unload all available and not previously unloaded partitions. A GDG base should be defined, with enough generations to retain all the information required.

If a month's worth of data is required, define the GDG base to have a LIMIT of 30 and schedule the unload JCL job to run once per day, ideally when no users are using the Query Monitor system.

The DCB for the GDG should be:

```
DCB= (LRECL=255,DSORG=PS,RECFM=FB)
```

When data is offloaded to disk, Query Monitor flags that the data is no longer available, but it can be reloaded into the history table spaces as long it was unloaded to GDGs, and that the physical GDG it was written to still exists.

Loading data to the history database will mean that all data previously held in the History table space will be deleted, and the requested data will be loaded. This ensures that obsolete data is not held in DB2 for longer than is necessary. Multiple data sets can be concatenated to be loaded into the history tables.

If the history tables are used heavily, the underlying table space should be REORGed to reclaim space.

As with the unload of data, there are two methods of loading it. One method is to type `L` next to the interval in the Past Activity screen. The second method, and most efficient, is to use the supplied JCL to load history database from the selected GDGs.

As the past and historical data is stored in DB2 tables, these tables can be queried, using any SQL processor like QMF or SPUFI, to perform more complex analysis on the data.

Operation of Query Monitor

```
CQM$MAIN V1R1 ----- IBM DB2 Query Monitor ----- 2000/11/15 18:06:42
Option ===> _____
```

```
DB2 QM Subsystem ID: DBQM
```

```
User: PAOLOR3
```

- 1. View Current Activity
- 2. View Past Activity
- 3. Work with Monitoring Agents
- 4. View DB2 Command Activity
- 5. Work with Profiles
- S. Setup
- X. Exit Query Monitor

Activity for the current monitoring period

Previous periods and unloaded historical data

Display subsystems monitored and refresh profiles

DB2 commands issued in current monitoring period

Create, update, delete all types of profile

Enter DB2 configuration information

```
Enter END command to return to ISPF.
```



ibm.com/redbooks

© 2000 IBM Corporation

7.5 Operation of Query Monitor

This is the primary Query Monitor menu. The first time a user visits this panel, the name of the Query Monitor subsystem needs to be entered. A brief description of the options follows:

Option 1

Option 1 displays the threads which have run or are currently active during the current monitoring period.

Option 2

Option 2 displays all the previous intervals which have been collected. It allows you to see which are stored in DB2, which have been unloaded to DASD, which have been reloaded and which data is no longer available.

Option 3

Option 3 gives a display of all DB2 subsystems that are defined to the OS/390 operating system, which are active and which are being monitored by agents. As long as the DB2 subsystem and a monitoring profile have been defined in the start up JCL, monitoring can be interactively switched on and off and Monitoring Profiles can be refreshed. (They first have to be updated via option 5.)

Option 4

Option 4 displays the DB2 command activity for the current interval. Commands that are entered via DB2/DB2 Commands, DSN or from the Console are captured here.

Option 5

Option 5 is where profiles can be created, updated or deleted. A sub menu will be displayed giving the choice of monitoring, application or exception profiles. This is where the initial monitoring and exception profiles should be created.

Option S

Option S allows you to enter DB2 configuration information which will be stored in the control file. When initially configuring Query Monitor, this is where the startup data is entered.

The next charts show the DB2 subsystem monitoring agent screen and the DB2 configuration information required when the system is initially being configured. The next step is to create an initial monitoring and exception profile.

DB2 subsystem monitoring

```
CQM$DDB2 V1R1 ---- DB2 QM Monitoring --- 2000/11/27 12:06:09
Option ==> _____ Scroll ==> PAGE
```

Valid Cmds: Activate, Deactivate, Refresh prof, Change prof

Cmd	Subsystem	Active	Monitored	DB2 QM SSID	Profile
—	DB2Z	NO	NO		
—	DB2X	NO	NO		
—	DB2Y	YES	YES	DBQM	ALLTHRD2
—	DB2A	YES	NO		
—	DBG1	NO	NO		
—	DBG2	NO	NO		
—	DBG3	NO	NO		

***** Bottom of Data *****

Whether DB2 active
DB2 subsystems defined

Whether QM monitoring it

QM profile being used



ibm.com/redbooks

© 2000 IBM Corporation

7.6 DB2 subsystem monitoring

This panel displays information about all DB2 subsystems defined and their current status. This panel changes dynamically as DB2 subsystems are stopped and started and as monitoring of these subsystems is stopped and started.

As long as the DB2 subsystems are defined to Query Monitor via the start up JCL monitoring can be dynamically stopped and started from this panel.

The panel also shows the Query Monitor profile that is defined to each subsystem.

Updating DB2 subsystem information

```
-----Enter DB2 System Parameters-----  
Command ==>  
  
GDG Base Model DSN      ==> DB2QM.UNLOAD.DATA  
DB2 Control Dataset     ==> CQM.DB2.CONTROL  
(Pre-allocated)  
  
Enter DB2 Subsystem Info:  
  
DB2 Subsystem ID       ==> DB2Y      (1-4 Character Subsystem ID)  
  
  
Valid command selection values are:  
1: ZPARM, BSDS, and Load Library Information  
2: DB2 Query Monitor Parameters
```



ibm.com/redbooks

© 2000 IBM Corporation

7.7 Updating DB2 subsystem information

The upper part of this screen allows a model GDG base to be input.

The DB2 subsystem ID refers to the DB2 subsystem that the monitoring agents will write the data to, that is, the one where the objects are created, the plan is bound and the first one in the MONITOR parameter list.

Updating DB2 information (contd)

OPTION 1

Enter or Update Specific DB2 Parameters :

```
DB2 ZPARMs Member      ==> DSNZPARM
DB2 Bootstrap DSN #01   ==> DB2V610Y.BSDS01
DB2 Bootstrap DSN #02   ==> DB2V610Y.BSDS02
DB2 Loadlib1            ==> DSN610.SDSNEXIT.SC63
DB2 Loadlib2            ==> DSN610.SDSNLOAD
DB2 Loadlib3            ==>
DB2 Loadlib4            ==>
DB2 Loadlib5            ==>
```

OPTION 2

Enter or Update Specific DB2 Parameters :

```
Plan #1 Name           ==> CQMPLAN1
```



ibm.com/redbooks

© 2000 IBM Corporation

7.8 Entering the DB2 specific information

Option 1 is reached from the previous chart and contains example system information for the DB2 specified in the previous panel.

Option 2 is reached from the previous panel and contains the name of the plan that was bound for Query Monitor. Remember to grant execute access on the plan as appropriate to your installation standards.

This information needs to be entered before the Query Monitor subsystem is raised. It only needs to be done once.

Initial profile creation

Monitoring Profile

- Create at least 1 monitoring profile
- Should match the profiles in MONITOR parameter
- Additional profiles and amendments can be created subsequently

Exception Profile

- Create at least 1 exception profile
- Should match profile in EXPROF parameter in start up JCL
- One active exception profile per QM subsystem

Application Profile

- No initial profiles need to be created before raise QM subsystem



© 2000 IBM Corporation

ibm.com/redbooks

7.9 Initial profile creation

This section explains the concepts of the profiles and how to set up each one. When the Query Monitor subsystem is initially created there must be a minimum of one monitoring profile and one exception profile. These must correspond to the ones defined in the startup JCL. Once they have been created, the Query Monitor subsystem can be started, and data collection will commence.

At this point, it is probably a good idea to review the definitions of the profiles.

7.9.1 Profile summary

The monitoring profile controls the information gathered by the agents. If an application profile is being used, this filters the monitored information being displayed by the user.

The exception profile sets upper limits and when a thread exceeds any of these, it is flagged as an exception.

The next charts go through the concepts of filters, how to create them and their operation. Although monitoring and application profiles are different in function, they are the same in creation, and will be dealt with together.

Profile processing is accessed via Option 5 from the main menu and then either Option 1 for monitoring, 2 for application, or 3 for exception profiles.

Monitoring and Application Profiles

CQM\$PRFT V1R1 --- Update NEW --- 2000/11/15 20:27:34
Option ===> _____ Scroll ===> PAGE

Description PROFILE _____

Valid Cmds: Create,Update,Delete

```
-----  
Cmd  Type      Inc/Exc Fltr Data  QM Pln SQLOrd Interv Host V  
-    AUTHID    INC    *          Y    N    N    N    N  
-    CONNECTN  INC    *          Y    N    N    N    N  
-    JOB       INC    *          Y    N    N    N    N  
-    PLAN      INC    *          Y    N    N    N    N  
-    SSID      INC    *          Y    N    N    N    N  
***** Bottom of Data *****
```

Select the
required
filter type.
Repeat
as
necessary



ibm.com/redbooks

© 2000 IBM Corporation

7.10 Monitoring and application profiles

Monitoring and application profiles allow you to include or exclude data based on certain criteria. This is known as a filter. Remember that a monitoring profile will include/exclude the data from being collected and an application profile will include/exclude it from being displayed. The description below applies for both of these types of profile.

7.10.1 Categories for filters

There are five types of filters that can be used to filter data:

- **Authorization ID (Authid):** The DB2 authorization ID of the thread
- **Connection type (Connectn):** The type of connection being used: TSO, Batch, Utility
- **Job name (Job):** The name of the batch job or the userid, if entered dynamically
- **Plan name (Plan):** The name of the DB2 plan issuing the call
- **Subsystem (SSID):** The DB2 subsystem identifier

Not all of the filters need to be used, for example, if you are creating a monitoring profile, it would not be logical to use the SSID to exclude the subsystem you are monitoring.

7.10.2 Using wildcards

The use of wildcards is permitted. Note that not all of these filter types are required, because the default will include all the information.

7.10.3 Altering filters

If a `u` is placed next to one of the filters, it can be updated. To create a new filter, a `c` can be placed next to any of the lines; it does not need to be the filter type you want, because this is specified on the next panel displayed.

Monitoring and Application Profiles (contd)

```
CQM$PRFF V1R1 ----- Update Detail ---- 2000/11/15 20:30:2
Option ==> _____
```

```
Include or Exclude I      (Include/Exclude)
```

```
Filter Type      P      (Job,Plan,Authid,
                          Ssid,Connection)
```

```
Filter Detail    *      (Enter wildcard criteria)
```

```
Exclude Display of DB2 QM Threads      Y  (Yes/No)
```

```
Display SQL in order of execution      N  (Yes/No)
```

```
Flag threads active in prev interval    N  (Yes/No)
```

```
Display value of DB2 Host Variables    N  (Yes/No)
```

```
Press Enter/PF3 to update or CAN to cancel
```

For each
filter type
add the
required
'include'
and
'exclude'
data



ibm.com/redbooks

© 2000 IBM Corporation

7.11 Creating and updating profiles

The following example shows how to exclude and include certain plans, but the same process is true for all the filter types. The process is repeated until all of the filters have been created.

7.11.1 Creating a PLAN filter

As an example, say that there are three plans which run in a subsystem, PLANA, PLANB, PLANC.

- To include all plans:
 - Set the Include/Exclude to an **I**.
 - Ensure the filter type is a **P**.
 - Leave the filter detail as a *****.
 - Press Enter to save this.

This will now display or monitor all plans.

To exclude PLANA two lines are needed. One line to include all plans (as created above), the second is to exclude PLANA. The above process can be repeated to achieve this, but change the **I** to an **E** in the include/exclude field and enter PLANA in the filter detail.

Be careful when excluding items as excluding ***** will exclude everything but subsequent includes will not have any effect (everything will still be excluded). So

using the above example of PLANA, B and C, to include only PLANC, do not code:

- EXCLUDE *
- INCLUDE PLANC

Instead, generate filters as:

- INCLUDE *
- EXCLUDE PLANA
- EXCLUDE PLANB

The next chart shows what this would look like as a completed filter. This process can be repeated for all of the filter types.

7.11.2 Other filtering parameters

There are four other items that can be changed on this screen.

Exclude display of DB2 QM threads

This default is to exclude Query Monitor threads from being reported on. We recommend that this default is retained.

Display SQL in order of execution

If **Y** is selected, all SQL calls for the threads will be listed in the order that they were executed. If **N** is selected a summary by statement type is produced. The following example illustrates this.

This query, consisting of two SQL statements, was issued through SPUFI:

```
SELECT * FROM SYSIBM.SYSTABLES
WHERE DBNAME = 'CQMDB'
AND NAME = 'INTERVALS_V11' ;
SELECT COUNT(*) FROM QM.DBRMS_V11
WHERE PARTITION = 1
AND INTERNAL_DBRM_NBR IN (1,2)
```

Query monitor produced the following information on the thread:

Cmd	DBRM/PKG	Collection ID	DB2 CPU Time	DB2 Elapsed	SQL Calls
_	DSNESM68	DSNESPCS	0.057833	0.064300	11

Having an application filter with display SQL in order as **Y** this produces a display of every statement type by SQL call. If a large number of rows were returned by the query, there is one line for each fetch.

Stmt #	Statement Type	SQL Calls
116	PREPARE	1
190	OPEN	1
183	FETCH	1
183	FETCH	1
197	CLOSE	1
116	PREPARE	1
190	OPEN	1
183	FETCH	1
183	FETCH	1
197	CLOSE	1
229	COMMIT	1

If the filter was set to **N**, a summary by statement type is produced.

Stmt #	Statement Type	SQL Calls
116	PREPARE	2
190	OPEN	2
183	FETCH	4
197	CLOSE	2
229	COMMIT	1

The advantages and disadvantages are:

- Having the filter set to **N**, produces a much shorter report, but all the statistics are summarized to the statement type level; if there are multiple SQL statements, it is not possible to distinguish between the two. In addition, only the SQL text of the last issued statement is obtainable.
- Having the filter set to **Y** gives all the information about each statement, including the SQL text for each. However, the report will be very long if a lot of rows are returned.
- Rule of thumb: If there is only one statement set if to **N**, multiple statements set it to **Y**. If you are in doubt, set it to **Y**. This can be changed easily via an application profile if necessary.

Flag threads active in previous interval

This will show if an active thread was active in the previous monitoring interval. If a thread is long running, it may be “lost” if it is still running when the interval changes. If this flag is set to **Y**, it will still be monitored in the next interval; the **INTV** field will be flagged to indicate that it spans intervals. In general, the default value of **N** should be retained unless you have many long running threads.

Display value of DB2 host variables

This option is not operational in this version of Query Monitor.

For the above parameters, change all of the values for all filter types and for both monitoring and application profiles.

Monitoring and Application Profiles (contd)

CQM\$PRFU V1R1 --- Update PROFILE --- 2000/11/16 18:05:10
Option ==> _____ Scroll ==> PAGE

Description	EXAMPLE OF A PROFILE
Exclude Display of DB2 QM Threads	Y (Yes/No)
Display SQL in order of execution	Y (Yes/No)
Flag threads active in prev interval	N (Yes/No)
Display value of DB2 Host Variables	N (Yes/No)

Change default
of Display SQL
to Y

Valid Cmds: Create,Update,Delete

Cmd	Type	Inc/Exc	Fltr	Data
—	AUTHID	INC	*	
—	CONNECTN	INC	*	
—	JOB	INC	*	
—	PLAN	INC	*	
—	PLAN	EXC	PLANA	
—	PLAN	EXC	PLANB	
—	SSID	INC	*	

Plans A and
B are excluded
from being
monitored

***** Bottom of Data *****



ibm.com/redbooks

© 2000 IBM Corporation

7.12 A completed filter

This chart shows a completed profile which could be for either a monitoring or application profile. Remember that at least one monitoring profile must be created before raising the Query Monitor subsystem.

Exception Profiles

```
CQM$EXCU V1R1 --- Update LOWPROF --- 2000/11/15 20:14:08
Option ==> _____
```

Profile Name LOWPROF

Description LOW VALUES FOR EXCEPTIONS _____

Total SQL Calls	000000000500
Total Getpages	_____
Lock Request	_____
Lock Escalations	000000000001
CPU Time	__ : __ : 05
Elapsed Time	__ : __ : __
Application CPU Time	__ : 01 : __
Application Elapsed Time	__ : __ : __

Press Enter/PF3 to create or CAN to cancel

Values for
exceptions
can be entered
as required



ibm.com/redbooks

© 2000 IBM Corporation

7.13 Exception profiles

Multiple exception profiles can be created, but only one can be active at any one time. The same exception profile applies across all the monitored subsystems.

When any of the categories above are exceeded, the thread is flagged as an exception, the plan name field and the value which has been surpassed are displayed in red.

The SQL calls, getpages, lock requests, and escalations represent the number of occurrences of each of these per thread.

The CPU and Elapsed Time relates to the DB2 (class 2) time and the Application CPU and Elapsed is the total thread (class 1) CPU and Elapsed times. Times are in HH:MM:SS.

Exception Profile processing

```

CQM$DTHR V1R1 ----- DB2/QM Thread Activity ----- 2000/11/17 19:51:14
Option ==> Scroll ==> PAGE
-----+-----
DB2 QM Subsystem: DBQM Current Interval Started: 11/17/2000 19:00:00

-----+-----

Display Threads: I (Active/Interval) D (Detail/Summarize)

-----+-----

Cmd SSID Intv Plan Name Job Name DB2 CPU Time DB2 Elapsed SQL Calls
- - - - -
- DB2Y - DGOPMOM PAOLOR6 0.000000 0.000000 0
- DB2Y - DSNTEP61 PAOLOR6F 28.588955 35.772352 103,719
- DB2Y - DSNESPCS PAOLOR6 0.030060 1.179937 8
- DB2Y - DSNESPCS PAOLOR6 0.087251 1.917180 18
- DB2Y - DSNESPCS PAOLOR6 0.140601 1.972357 18
- DB2Y - DSNESPCS PAOLOR6 0.128551 4.236213 18
- DB2Y - DISTSERV DB2YSPAS 0.000000 0.000000 0
- DB2Y - DGOPMOM PAOLOR6 0.299174 0.326938 236
- DB2Y - DSNTEP61 PAOLOR6E 1:02.345012 1:20.862367 256,065
                                     (mm:ss.dddddd) (mm:ss.dddddd)

Valid Line Commands: (Sql detail, Instruction totals, Buffers, Delays, Locks)

```

Exceptions
are
highlighted
in red on
screen



ibm.com/redbooks

© 2000 IBM Corporation

7.14 Exception profile processing

The above chart shows an example screen from the current data processing, using the exception profile shown previously. Each thread that exceeds any of the thresholds gets flagged, by the plan name turning red, together with the value that has been exceeded. Exceptions are therefore, very identifiable.

Note that the CPU and elapsed times on this screen will be in hh.mm.ss.dddddd format, although the hour columns will be mostly blank. The annotation under the columns on this chart is to signify that this has been added to aid in your interpretation and it does not appear on the screen itself.

Updating Profiles

Profile type	Create new profile	Swap to another profile	Update the active profile
Monitoring	Y	N <small>(note 1)</small>	Y <small>(note 2)</small>
Application	Y	Y <small>(note 3)</small>	Y <small>(note 3)</small>
Exception	Y	N <small>(note 4)</small>	N <small>(note 4)</small>

Notes

1. Need to drop QM subsystem and update start up JCL. Change not active on Option3
2. Select 'U' profile in Option 5, then refresh 'R' in Option 3
3. Select 'S' in Option 5 to refresh
4. Need to drop QM subsystem and update start up JCL



ibm.com/redbooks

© 2000 IBM Corporation

7.15 Updating profiles

All types of profiles can be created when the Query Monitor subsystem is active. However, there are a number of factors to take into account when changing the contents of filters, especially when they take effect. This is summarized as follows.

Monitoring profile

The current release of Query Monitor does not allow the user to swap to a new monitoring profile. The only way to do this is to define it in the start up JCL and refresh the started task. Later releases will allow profiles to be changed via Option 3 from the main menu.

Updating the active monitoring profile is possible without having to stop and start Query Monitor. Update the profile via Option 5 and when complete, go to Option 3 from the main menu and type an **R** next to the subsystem to refresh the profile.

Application profile

Application profiles can be updated as with any other profile via option 5 from the main menu. When the updates have been made, type an 'S' next to the profile to select it as the active one. Now go back to display the current interval and the information will be filtered as requested. Each individual user of Query Monitor can have their own profile allocated if necessary.

Exception profile

Exception profiles can be updated when the Query Monitor subsystem is active, but can only be refreshed by dropping and raising the address space. If a different profile is required, then the start up JCL needs to reflect this.

Checklist to start QM subsystem

Start up JCL tailored?
DB2 Objects created?
Plans and packages bound?
Parameter file created?
Monitoring profile created?
Exception profile created?



Raise
QM
subsystem



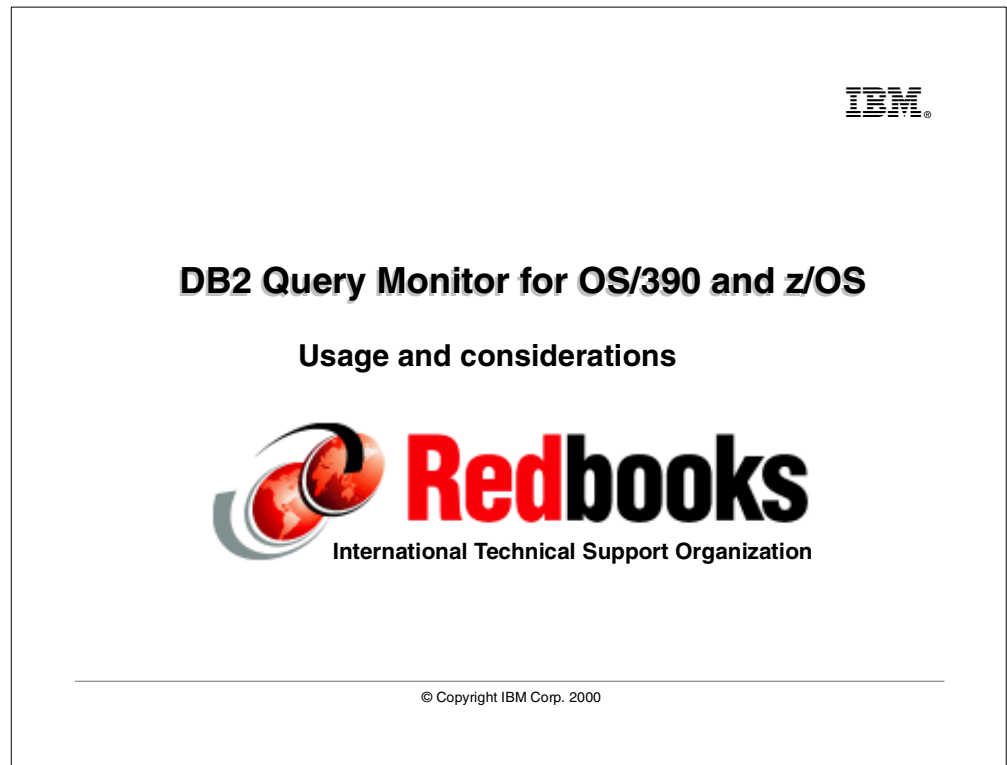
ibm.com/redbooks

© 2000 IBM Corporation

7.16 Checklist to start QM subsystem

Once all these steps have been completed, the Query Monitor subsystem can be started. These steps only need to be completed once. Subsequently, Query Monitor can be raised and dropped with no changes to parameters.

Chapter 8. Usage and considerations



In this chapter, we look at the data from the Query Monitor reports and show examples of its use.

Running and managing Query Monitor

- What can be monitored
 - Screen introduction
 - Thread navigation
 - Detail and summary screens
 - Example use of information
- Current activity
 - Case study
 - Comparison to DB2 PM output
- Past activity
 - Case study
- Command activity
- Utilities and binds
- Summary



ibm.com/redbooks

© 2000 IBM Corporation

8.1 Running and managing Query Monitor

This section goes through the monitoring screens and demonstrates the functions available within Query Monitor. This section is divided into parts, each dealing with a specific area of Query Monitor.

What can be monitored

These charts show the thread navigation that is possible for both current and past activity analysis. The information that is available at each level of detail is shown.

The ISPF commands that can be issued in the panels are also documented.

Current activity

This part of the section will go through a case study, looking at a thread and the information that can be gathered about it.

For comparison, a section comparing the same thread with information gathered using Query Monitor and DB2 Performance Monitor is shown. The PM reports and output from Query Monitor is included in Appendix B, "Query Monitor" on page 221.

Past activity

A case study using the historical data is shown. The same analysis can be performed as current activity, but to avoid duplication, this case study is concerned with unloading and loading historical data.

Command Activity

A discussion about DB2 Command Activity gives the user knowledge about where to find information of what DB2 commands have been issued and who issued them.

Utilities and Binds

Detection of Utilities and Binds is also possible.

Summary

Finally there is a summary of the main Query Monitor functions, together with some hints and tips to bear in mind when using the product.

Activity screen introduction

- Current v past activity
 - Intermediate screen for intervals
 - Navigation between intervals
- Commands
 - These are entered on the 'Option' line
- Data columns
- Line commands
 - These are entered under the 'CMD' column
 - Valid commands for screen via help or screen text
- Function key navigation
 - Standard left, right, up, down



ibm.com/redbooks

© 2000 IBM Corporation

8.2 Activity screen introduction

Before going any further into the function of Query Monitor, some of the basic ISPF commands that can be issued and the navigation that can be performed on the activity screens are discussed. Also, the columns that appear are briefly described.

8.2.1 Current and Past Activity panels

The same information is provided on the Current and Past Activity panels. This is discussed below. There is an intermediate screen when entering the Past Activity panels. This allows you to select the interval you are interested in. To select an interval it must be available, indicated by a \surd . Further discussion about unloading and loading history data forms is part of one of the case studies later in this chapter. At this point, assume that one of the intervals has been selected (with an 'S'). The next panel will contain the thread information. The same information about the threads can be found here for past intervals as well as for the present one.

On the Past Activity panels, information is provided about the interval start and stop times and dates for the interval you are currently in.

An additional feature for Past Activity is that you can navigate between the intervals and shortcut to the current interval. The commands to do this are as follows:

Next interval

Type `NEXT` as a command to go to a later interval. Alternatively, the F6 key can be used. This can also take you to the Current Activity panel when you are in the most recent history interval.

Previous interval

Type `PREV` as a command to go to an earlier interval. Alternatively, the F4 key can be used.

Current interval

From the Past Activity panels, a short cut to the Current Activity panel is possible. Type `CURRENT` on the command line.

Past interval

From the Current Activity screen, there is a short cut to the past intervals screen. From the command line, type `INTV` (short for intervals).

8.2.2 Commands

There are an additional number of ISPF commands that can be issued on the panels.

PRINTX

Typing `PRINTX` from any panel will print the panel to the RSCPRINT DD of your TSO session. No feedback message will confirm this. The output can be found by browsing your userid's job output in SDSF. We found that the easiest way to manipulate this data is as follows:

1. Go to your userid's job output in SDSF.
2. Type a `?` against the job. This should display the component parts of the output.
3. Against the part with DDNAME CQMPRINT type `xd`.
4. The SDSF Open Print Data Set dialog is displayed.
5. Fill in the appropriate information to direct the output to a data set. This can then be manipulated, viewed or reformatted via ISPF.

SORT

It is possible to sort the screen by any of the columns displayed. Default is by `plan_start_time` descending, so that the most recent threads are always displayed at the top of the list. Sort can be invoked by specifying the column name, replacing any spaces with an underscore. For example:

```
SORT DB2_CPU_TIME D
```

This sorts the threads so that the highest CPU intensive thread is at the top of the list. The sort sequence will remain until the results are re-sorted, even if swapping between active and past intervals.

`SORT PLAN_START_TIME` will return to the standard sequence.

The columns also have a number associated with them, according to the list in 8.2.3, “Data columns” on page 114. For example:

`SORT 20 D` would have the same effect as `SORT PLAN_START_TIME D`.

FIND

Entering the *FIND string* as a command will find the first occurrence of the string on the panel. For example:

`FIND DSNE`

This will find the first occurrence of the string DSNE; in this case it will probably find the first occurrence of the SPUFI plan.

HELP

Typing `HELP` on any panel will display the help panel for that screen. The F1 key also has the same function.

8.2.3 Data columns

The following information is collected on each thread, this is also stored as historical data after the current monitoring interval as ended. Most of this information is to the right of the activity screen, using the F10 key will display these columns.

1. CMD
2. SSID
3. Intv
4. Plan Name
5. Job Name
6. DB2 CPU Time
7. DB2 Elapsed
8. SQL Calls
9. X Exec
10. Connectn
11. APPL CPU
12. APPL Elapsed
13. Total Getpages
14. Token
15. Accounting token
16. Requesting site
17. NETID
18. LUNAME
19. AUTHID
20. Plan Start Time
21. Plan End Time

8.2.4 Line commands

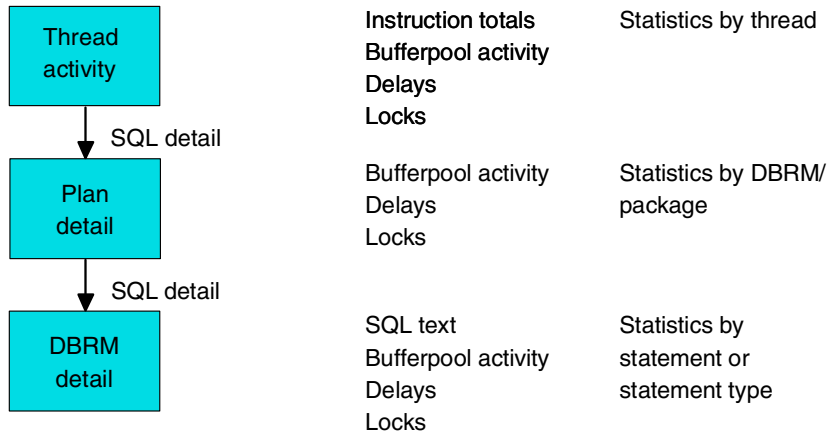
These can be entered alongside any thread under the CMD column. To see what are valid line commands for a particular screen, press the help key, F1. Additionally, if an invalid command is entered, a panel of the valid commands is displayed.

8.2.5 Function key navigation

Standard navigation using the F7, F8, F10 and F11 keys for up, down, left and right respectively is available. There are characters underneath the scroll value in the top right corner of the screen to indicate where there is more information outside of the panel window.

- < data to the left
- > data to the right
- + data above
- data below

Thread navigation



ibm.com/redbooks

© 2000 IBM Corporation

8.3 Thread navigation

This chart (and subsequent ones) gives information on where to find specific information in both the current and Past Activity panels. The screen flow is the same for both options off the main menu. If you are in Past Activity, there is the intermediate panel to select the interval you are interested in.

The threads panels allow for easy drilling down to lower levels of detail. In general the same information is available at each level, but at a lower level of detail as you drill down. Conversely, the higher level panels are a summation of those panels beneath.

At the lowest level of detail, the DBRM statement, it is possible to export the text of the SQL statement to a data set.

Thread activity options

Two choices on display of data

- Detail or summarize
 - Summary groups threads by plan
 - Multiple job names replaced by '*****'
- Active or interval
 - Active displays only currently active threads (should be displayed in orange)
 - Interval shows all threads in the current monitoring interval

Display type retained until options reset

Available in current and past data



© 2000 IBM Corporation

ibm.com/redbooks

8.4 Thread Activity options

The Current Activity and Past Activity screens give two options (four combinations) of how to view the data. These are invoked by changing the values in the *Display threads* line of the screen. These values have no effect on the Past Activity screen, although the values remain until they are changed back, even when going to the Current Activity screen. For general work and analysis, the default values of interval and detail are the most useful.

8.4.1 The detail panel or summarize options group

Detail is the standard display and shows one line per threads invocation. The summarize option groups the information by plan. Numeric values are summed and where multiple jobs have invoked the same plan, this is signified by asterisks. Columns, such as plan start and end times and authid, give the values for the last invocation of the plan. This option is useful to quickly find which plans have run in the interval, but the detail option is the more useful one.

8.4.2 Active or interval

The active option will only show the threads that are actually active at the time, they will be shown in orange. This is useful if the throughput of threads is high and there are a number of long running threads. These may have disappeared off the active window, and so they can be traced more easily. Again the more useful option is the default: interval.

Thread activity

```

CQM$DTHR V1R1 ----- DB2/QM Thread Activity ----- 2000/11/21 12:28:24
Option ==> _____ Scroll ==> PAGE
----->
DB2 QM Subsystem: DBQM Current Interval Started: 11/21/2000 12:02:30

-----

Display Threads: I (Active/Interval) D (Detail/Summarize)

-----

Cmd  SSID Intv Plan Name Job Name DB2 CPU Time DB2 Elapsed SQL Calls
S   DB2Y - QMF610 PAOLOR3 0.040701 0.076397 107
-   DB2Y - ANL4QMF PAOLOR3 0.067818 0.632690 27
-   DB2Y - QMF610 PAOLOR3 0.010870 0.092626 3
-   DB2Y - QMF610 PAOLOR3 0.071552 1.336483 34
-   DB2Y - DISTSERV DB2YSPAS 0.000000 0.000000 0
-   DB2Y - DISTSERV DB2YDIST 0.041771 2.601946 90
***** Bottom of Data *****

```

Interval
details

Display
options

Threads
run during
this
interval



ibm.com/redbooks

© 2000 IBM Corporation

8.5 Thread Activity panel

This is the main panel, and the one that will be most often visited. By default, new threads appear at the top of the list, unless the columns have been sorted in another order. To ease understanding, do not resort the columns unless doing so for a specific purpose, general monitoring is much easier this way. There is more information off to the right of the screen, and the navigation function keys can be used as discussed earlier.

The next few charts will give an example of drilling down to lower levels of detail into the thread. Remember that locking will delay instruction totals, and buffer pool statistics can also be viewed at each of these layers. A summary chart at the end of this section gives some more information about this.

In our example, we are going to find more information about the top thread in the list on the chart. To get some more SQL detail, type an `s` as a line command.

The details, so far:

This thread was run on subsystem DB2Y, by user PAOLOR3 and was executing the QMF plan. Total DB2 elapsed time was 0.076 seconds.

Plan detail

```
CQM$DPLN  V1R1 ----- Query Monitor Plan Detail ----- 2000/11/21 12:29:37
Option  ===> _____ Scroll ===> PAGE
```

```
Jobname: PAOLOR3   Plan: QMF610   DB2 SSID: DB2Y   Plan Execution Count: 0001
```

```
Totals or Averages: A   (Total of all executions/Average of each execution)
```

```
-----
Cmd DBRM/PKG  Collection ID          DB2 CPU Time      DB2 Elapsed  SQL Calls
S  DSQBFSQL  Q                      0.039934       0.075623     106
_  DSQBTMGT  Q                      0.000766       0.000773      1
***** Bottom of Data *****
```



ibm.com/redbooks

© 2000 IBM Corporation

8.6 Plan detail panel

At the plan detail panel, we can see the same thread, but broken down by DBRMs and packages used. In this example the thread used two packages in the QMF Q collection. The elapsed time and the SQL calls add to the values in the previous panel.

Selecting the DSQBFSQL package with an *s* will give a greater level of detail:

The DB2 elapsed time for this package was 0.075623 and issued 106 SQL calls. The total for elapsed with and SQL calls is the value from the previous screen.

DBRM detail - 1

```
CQM$DSTM V1R1 ----- DB2 QM DBRM Detail ----- 2000/11/21 12:30:48
Option ==> _____ Scroll ==> PAGE

Plan: QMF610      DBRM: DSQBFSQL      DB2 SSID: DB2Y      Plan Execution Count: 0001

Totals or Averages: A      (Total of all executions/Average of each execution)

-----

Cmd  Stmt #  Statement Type          SQL Calls  DB2 CPU Time      DB2 Elapsed
S   1,346  PREPARE                  1          0.014169          0.017896
-   1,728  DESCRIBE                   2          0.000333          0.000336
-   3,089  OPEN                      1          0.000223          0.000273
-   3,494  FETCH                     102        0.025208          0.057118
***** Bottom of Data *****
```



ibm.com/redbooks

© 2000 IBM Corporation

8.7 DBRM - Detail (1)

The DBRM detail screen, now breaks the thread down by statement type, and the time that each took is recorded, together with the number of SQL calls issued.

The total number of calls and the DB2 times will be equal to that from the previous screen.

Typing `s` as a line command on this screen will display the SQL instruction text screen. This will be displayed in two charts. The next chart will show a different format of the DBRM detail screen.

DBRM detail - 2

CQM\$DSTM V1R1 ----- DB2 QM DBRM Detail ----- 2000/11/21 12:32:06
 Option ==> _____ Scroll ==> PAGE

Plan: QMF610 DBRM: DSQBFSQL DB2 SSID: DB2Y Plan Execution Count: 0001

Totals or Averages: A (Total of all executions/Average of each execution)

Cmd	Stmt #	Statement Type	SQL Calls	DB2 CPU Time	DB2 Elapsed
-	1,346	PREPARE	1	0.014169	0.017896
-	1,728	DESCRIBE	1	0.000172	0.000173
-	1,728	DESCRIBE	1	0.000160	0.000162
-	3,089	OPEN	1	0.000223	0.000273
-	3,494	FETCH	1	0.000357	0.000392
-	3,494	FETCH	1	0.000287	0.000312
-	3,494	FETCH	1	0.000274	0.000310
-	3,494	FETCH	1	0.000317	0.002759
-	3,494	FETCH	1	0.000274	0.000302
-	3,494	FETCH	1	0.000244	0.000268
-	3,494	FETCH	1	0.000287	0.000488
-	etc. etc.				



ibm.com/redbooks

© 2000 IBM Corporation

8.8 DBRM - Detail (2)

As a side note, in an earlier section on profiles, there was a field that allowed you to display SQL statements in the order of execution. On the previous chart, this was set to N in the application profile.

To change it follow this procedure:

- Go to profiles, option 5 from the main menu.
- Select option 2 to work with application profiles.
- Update the application profile you are using by typing a U as a line command.
- One of the lines displayed on the next panel will be to display the SQL in order of execution.
- Change the N to a Y and exit the profile.
- Type an S to select it from the screen with the list of profiles. You will then receive a message that the profile is now active.

When you now navigate to the DBRM detail screen, each individual SQL call is listed.

SQL statement text

```
CQM$DSQL V1R1 ----- SQL Instruction Text ----- 2000/11/21 12:33:01
Option  ==> _____ Scroll ==> PAGE

Plan: QMF610      DBRM: DSQBFSQL   DB2 SSID: DB2Y   Plan Execution Count: 0001

-----

      SELECT * FROM SYSIBM.SYSTABLES FOR FETCH ONLY
***** Bottom of Data *****
```



ibm.com/redbooks

© 2000 IBM Corporation

8.9 SQL statement text

If an **s** was typed next to one of the statements on the DBRM detail screens, the text of the SQL statement is displayed. It is possible to display the SQL text at higher levels also, but from this screen, the statement can be additionally exported to a data set.

Type **EXP** as a command and a panel will be displayed prompting for a data set name and a member name to be entered. The data set has to have been allocated previously; the member name is only required if the data set is a PDS.

Examples

Dynamic SQL caching

- Initial statement
- Cached statement
- Repeated statement with different plan executing

Sorting example

- Open cursor with sort in SQL
- Open cursor without sort in SQL



ibm.com/redbooks

© 2000 IBM Corporation

8.10 Examples of detail screens

Two examples follow of the use of these detail screens. The first is to demonstrate the use of the statement cache for dynamic SQL and the second looks at the effect of an ORDER BY clause.

Dynamic SQL caching....

Cmd	SSID	Intv	Plan Name	Job Name	DB2 CPU Time	DB2 Elapsed	SQL Calls
-	DB2Y	-	DSNESP RR	PAOLOR3	2.221620	3.262403	388 (3)
-	DB2Y	-	DSNESP CS	PAOLOR3	2.073618	3.020702	388 (2)
-	DB2Y	-	DSNESP CS	PAOLOR3	2.255547	6.703290	388 (1)

Cmd	Stmt #	Statement Type	SQL Calls	DB2 CPU Time	DB2 Elapsed
-	116	PREPARE	1	0.062017	0.218405
-	190	OPEN	1	2.115267	6.389214
-	183	FETCH	384	0.076725	0.094034
-	197	CLOSE	1	0.000233	0.000289
-	229	COMMIT	1	0.001304	0.001346

***** Bottom of Data *****

Cmd	Stmt #	Statement Type	SQL Calls	DB2 CPU Time	DB2 Elapsed
-	116	PREPARE	1	0.000623	0.000733
-	190	OPEN	1	1.990825	2.919826
-	183	FETCH	384	0.078667	0.096579
-	197	CLOSE	1	0.000224	0.000276
-	229	COMMIT	1	0.003276	0.003286

***** Bottom of Data *****

Cmd	Stmt #	Statement Type	SQL Calls	DB2 CPU Time	DB2 Elapsed
-	116	PREPARE	1	0.054376	0.136405
-	190	OPEN	1	2.073508	3.022045
-	183	FETCH	384	0.080884	0.090787
-	197	CLOSE	1	0.000227	0.000246
-	229	COMMIT	1	0.012625	0.012919

***** Bottom of Data *****



ibm.com/redbooks

© 2000 IBM Corporation

8.11 Dynamic SQL caching

The dynamic SQL cache is enabled via a DSNZPARM parameter, CACHEDYN. Having it set to `Y`, enables the cache. For more information on the dynamic statement cache, refer to the *DB2 UDB for OS/390 Version 6 Application Programming and SQL Guide*, SC26-9004. However, for the purposes of this illustration, dynamic caching avoids having DB2 reprepare the statement if an identical has been issued previously and it still exists in the cache.

In the above example, the same SQL statement was issued (a simple select from the catalog with an ORDER BY). Thread (1) was invoked first, thread (2) second and thread (3) third. The first two times via the SPUFI plan with cursor stability (DSNESP CS) and the third time via SPUFI with repeatable read (DSNESP RR). Under the conditions for dynamic statement caching, statement two is eligible to use the cache.

The same procedure was then repeated with the cache disabled via the DSNZPARM parameter. This was to determine if the difference in preparation time was due to the cache. In this case, all three preparations were of a similar time. These figures are not documented here.

Thread (1)

DB2 prepares the statement and returns the rows satisfied by the query. The prepared statement is placed in the dynamic statement cache in the EDM pool for future use.

Thread (2)

DB2 checks the cache to see if an identical statement is contained in the cache. In this case it is, and DB2 does not have to prepare it. This is shown by the preparation times being about 100 times quicker than for the first statement.

Thread (3)

The same statement was issued a third time, but under a different plan, therefore, invalidating one of the rules for using the cache. DB2 had to reprepare the statement; this is shown by the preparation time being of a similar order of magnitude to the first statement.

Open cursor

Cmd	SSID	Intv	Plan Name	Job Name	DB2 CPU Time	DB2 Elapsed	SQL Calls
-	DB2Y	-	DSNESPSCS	PAOLOR3	2.122145	3.188787	388 (2)
-	DB2Y	-	DSNESPSCS	PAOLOR3	2.082310	3.177917	388 (1)

Cmd	Stmt #	Statement Type	SQL Calls	DB2 CPU Time	DB2 Elapsed
-	116	PREPARE	1	0.000601	0.000770
-	190	OPEN	1	1.999041	3.084969
-	183	FETCH	384	0.078458	0.087790
-	197	CLOSE	1	0.000233	0.000241
-	229	COMMIT	1	0.003975	0.004146

***** Bottom of Data *****

Cmd	Stmt #	Statement Type	SQL Calls	DB2 CPU Time	DB2 Elapsed
-	116	PREPARE	1	0.054546	0.114061
-	190	OPEN	1	0.000257	0.000551
-	183	FETCH	384	2.065850	3.072275
-	197	CLOSE	1	0.000231	0.000255
-	229	COMMIT	1	0.001259	0.001644

***** Bottom of Data *****



ibm.com/redbooks

© 2000 IBM Corporation

8.12 Open cursor

In this example, the same SQL statement was issued, the first time with an ORDER By clause, the second time without.

As can be seen, for statement one, the OPEN statement took considerably longer than for statement two as DB2 had to retrieve the rows and sort them when the cursor was opened. The fetch, for statement one was subsequently quicker than for statement two.

Thread/Plan/DBRM Activity

The following line commands are also available

- Either by thread, plan or DBRM
 - B - To view buffer pool statistics.
Information by buffer pool on pages read, pre-fetch etc.
 - D- To view thread delays.
Numbers and lengths of delays
 - L- To view locking statistics for the thread.
Numbers of lock and latches, claims and drains
 - I- To view SQL instruction totals.
Numbers of instructions by instruction type (at thread level only)



ibm.com/redbooks

© 2000 IBM Corporation

8.13 Thread, plan, or DBRM activity

It is possible to obtain information about the threads that Query Monitor monitors by thread, plan or DBRM depending on the screen you are currently in by entering the line commands as shown on the chart.

As a summary, the following table shows what information can be obtained at which level.

Panel	Information by:
Thread Activity	Thread
Plan detail	DBRM or package
DBRM detail	Statement type

You should also be aware that going to too low a level of detail may not give the results you were expecting. For example, if you look at buffer pool statistics at the statement level, you will get the information for each statement issued, and some of the statements may not have buffer pool statistics associated with it. In this situation looking at a higher level will be more informative as the values will be summarized to the DBRM/package level.

Case studies

Current Activity

- Looking at 'problem' threads
- Comparison with DB2PM
- Examples of DB2 PM and Query Monitor output

Past Activity

- Unloading and loading partitions
 - Options are available to use panels or supplied JCL and programs

Command Activity

- Where to find information on DB2 commands issued

Utilities and Binds

- Where to find information



ibm.com/redbooks

© 2000 IBM Corporation

8.14 Case studies

This section will look in detail at some of the functions of Query Monitor and where to find some of the information.

Current activity

A number of areas will be investigated here. First, a case study will investigate problems with a couple of threads and what information can be gathered about them. The third example shows how the Query Monitor statistics can be used answer frequently asked questions.

Secondly a comparison with the output of DB2 PM is discussed, and how these two products can be used together. Respective reports from both products will be included for comparison.

Past activity

The section will look into the different methods of retaining and reusing historical data within Query Monitor.

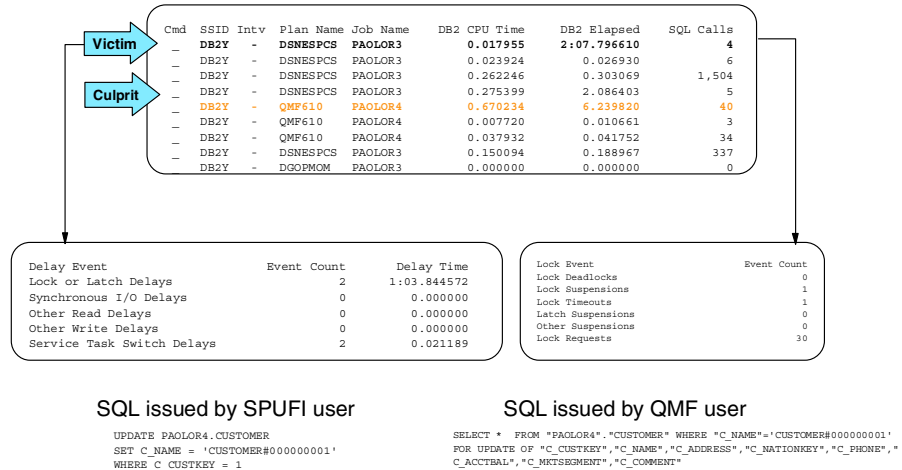
Commands

Information on DB2 commands can be found in a number of places, this section will look in detail as to where this information is recorded.

Utilities and Binds

Query Monitor only records limited information on Utilities and Binds, this is expanded further in this section.

Current Activity - Case study (1)



8.15 Current Activity - Case study (1)

In this scenario, a SPUFI user is reporting problems with an update taking a long time and then receiving a error message.

The user with the problem is identified as the top thread on the panel (PAOLOR3). Looking at some of the thread information available reveals what the problem is.

The lower left panel is the information from entering the line command for delays for the thread. This shows that the thread spend over a minute waiting for a lock or latch. The lower right panel is the thread lock activity and here it can be seen that a time-out occurred.

Looking at the Current Activity screen again, a QMF user is displayed in orange, signifying it as being active (PAOLOR4). Further investigation of this thread revealed that this thread was holding the locks preventing the SPUFI user from making the update.

The SQL the two threads were executing further confirms this.

Current Activity - Case study (2)

A thread exceeding thresholds is highlighted

Cmd	SSID	Intv	Plan Name	Job Name	DB2 CPU Time	DB2 Elapsed	SQL Calls
C	DB2Y	-	QMF610	PAOLOR4	1.903603	7.335825	108
-	DB2Y	-	ANL4QMF	PAOLOR4	0.109348	0.217317	15
-	DB2Y	-	QMF610	PAOLOR4	0.013150	0.019040	4
-	DB2Y	-	ANL4QMF	PAOLOR4	0.038606	0.148929	14

Cancel thread issued

The following thread is about to be canceled:

Jobname: PAOLOR4 Plan: QMF610 Token: 00085

Cancel Thread Y with Dump N



ibm.com/redbooks

© 2000 IBM Corporation

8.16 Current Activity - Case study (2)

In this example, the thread at the top of the list has been flagged as an exception as it has exceeded the total number of SQL calls a thread can make. As this is a QMF user and the thread is still active and causing problems for users of the online applications, it is decided to cancel the thread.

This can be performed from within Query Monitor as long as the QM user has the appropriate DB2 authority to issue the cancel (SYSOPR, SYSCTRL or SYSADM). Again Query Monitor does not override DB2 security, if you are not authorized to cancel threads, an error message will be displayed.

To cancel the thread, enter a **c** as a line command against the thread, and a confirmation box will appear. The change the Cancel thread field to **Y** and the dump field to **N** and press enter and the thread will be canceled.

Query Monitor allows you to quickly identify problems and terminate threads, from the same panel, allowing for quick and effective resolution.

Current Activity - Case study (3)

Q: Which was the most expensive thread this hour?

Cmd	SSID	Intv	Plan Name	Job Name	DB2 CPU Time	DB2 Elapsed	SQL Calls	
-	DB2Y	-	DSNTEP61	PAOLOR4B	15.511868	17.672098	7	Culprit
-	DB2Y	-	DISTSERV	DB2YDIST	3.246567	12.454918	8	
-	DB2Y	-	DSNTEP61	PAOLOR4B	1.737339	6.480142	14	
-	DB2Y	-	DSNTEP61	PAOLOR4B	1.714923	6.281369	14	
-	DB2Y	-	DSNTEP61	PAOLOR4B	1.706969	3.616753	5	
-	DB2Y	-	DSNTEP61	PAOLOR4B	1.694167	4.014660	14	
-	DB2Y	-	DSNTEP61	PAOLOR4B	1.622472	5.244860	5	
-	DB2Y	-	QMF610	PAOLOR4	1.194245	2.200839	379	
-	DB2Y	-	QMF610	PAOLOR4	1.038618	1.869663	7	

A: PAOLOR4 running an SQL query from batch job PAOLOR4B using plan DSNTEP2



ibm.com/redbooks

© 2000 IBM Corporation

8.17 Current Activity - Case study (3)

Who was running the most expensive query this hour? If this or similar questions are asked, then it is very easy to obtain this information from Query Monitor.

From the Past or Current Activity screen, sort the columns by the criteria necessary to display to information. In this case the display was sorted by DB2_CPU_TIME.

DB2 Query Monitor and DB2 PM

Discussion of the two products

- Introduction to DB2 Performance Monitor
 - On-line monitor
 - Batch reporting
- Output from the two products
 - Reports are reproduced in Appendix 3
- QM and DB2 PM
 - Why do I need two monitors?
- Using the two monitors
 - Information from DB2 PM long accounting report
 - Information extracted from Query Monitor



ibm.com/redbooks

© 2000 IBM Corporation

8.18 DB2 Query Monitor and DB2 Performance Monitor

One of the main DB2 monitors available today is DB2 Performance Monitor (DB2 PM). The following brief extract about DB2 PM is taken from the *DB2 Performance Monitor for OS/390 Online User's Guide*, SC26-9168.

8.18.1 DB2 PM Online Monitor functions

DB2 generates data about its own performance but it does not provide any reporting facilities for analyzing this data. The PM Online Monitor provides you with the capability to view an active DB2 subsystem and identify performance problems online.

The PM Online Monitor displays DB2 performance information in a comprehensive form that is easy to understand and analyze.

Summary of DB2 PM functions

You can use the DB2 PM Online Monitor to:

- Determine total DB2 system performance and efficiency.
- Measure an application's performance and resource use.
- Evaluate an application's impact on other applications and the system.
- Analyze and improve SQL statements.
- Identify potential problems.
- Determine tuning requirements for DB2.

When changes are made to an application or to the DB2 subsystem, the Online Monitor can help you determine the impact. This is very important for determining whether the changes increased or decreased performance.

When DB2 performance is not satisfactory, the Online Monitor can help you identify areas where tuning is required to optimize the performance of DB2. The Online Monitor can log DB2 activities and events and provide this information for later viewing to assist you in determining the cause of potential problems. You can also perform a thread diagnosis to view an analysis of a thread's performance and suggested improvements.

For a long-term view of DB2 performance, your needs are best served by the DB2 PM Batch reporting capabilities.

8.18.2 Batch reporting

DB2 PM has the ability to fully edit the data produced by the Instrumentation Facility Component of DB2, and produce meaningful reports which are easy to read and comprehend.

8.18.3 Why you need two monitors?

From the above discussion, and an experienced user of DB2 PM will confirm, DB2 PM can provide an incredible depth of information about every aspect of DB2's operation.

However, with this amount of information comes a cost, in terms of overhead of running the monitor and in training and use of it.

This is where Query Monitor can complement the use of DB2 PM. Query Monitor can provide first line assistance to DBAs on the DB2 subsystems, by providing information in a clear and concise way, and high-lighting problems as they are happening.

The majority of the time, the depth of information the DB2 PM collects is not required so Query Monitor can provide all the necessary data with much less overhead.

For the purposes of an illustration between DB2 PM and Query Monitor, the DB2 PM long accounting report was created. The same thread in the report was also captured within Query Monitor, and output was sent to SDSF by issuing the PRINTX on the selected ISPF panels. The reports have been 'tidied' up so that only the relevant information has been retained for presentation. These reports are contained in B.2, "DB2 PM long accounting report sample" on page 223.

8.18.4 Using the two monitors

There are going to be reporting differences between the two monitors, especially as Query Monitor does not start all the traces that DB2 PM does. The next section gives a brief discussion of the output from the two products.

Query Monitor and PM comparison

Table of comparisons:

	Source	Comment
DB2 time	Class 2	Query monitor differs in 2nd/3rd decimal place
Application time	Class 1	QM records differently (see note 1)
Delays (times)	Class 3	Query monitor differs in 6th decimal place
Delays (events)	Class 3	QM events = twice PM events

Note 1: QM Appl = PM Class1 - PM Class 2
(for both CPU and elapsed)



ibm.com/redbooks

© 2000 IBM Corporation

8.19 DB2 Query Monitor and DB2 PM comparison

This section discusses some of the variations that can be found between the two products: DB2 Query Monitor and DB2 PM.

In general the information from the two monitors will be mostly comparable, especially in the recording of buffer pool statistics, instructions and delays. Referring to Appendix B, "Query Monitor" on page 221, the preceding table is a summary of some of the differences that can be found between the two monitors and an explanation for it.

8.19.1 Observed variations

The chart highlights the variations that might be observed between the output from the two monitors. This is important to know if you are comparing output from the two system.

The difference in accuracy is due to the way that Query Monitor collects the information. The following is a mapping when comparing both sets of figures.

Class 1 (Application)

Query Monitor records this differently. The value is calculated as Class 1 minus Class 2 times. This is true for both elapsed and CPU time. It has the same accuracy as the class 2 times, that is to the second or third decimal place.

PM Appl CPU time = QM Appl CPU time + QM DB2 CPU time

PM Appl elapsed time = QM Appl elapsed time + QM DB2 elapsed time

Class 2 (DB2)

Times and CPU figures may differ in the second or third decimal place of seconds.

Class 3 (Delays)

Elapsed times may differ in the sixth decimal place of seconds. Events in Query Monitor are twice those in DB2 PM, so to compare the values, divide the Query Monitor figures by two.

8.19.2 Other variations

Smaller variations may also be detected for individual threads, for example, in the buffer pool or locking statistics. These are not significant for operational purposes and are due to the different ways in which the two monitors collect the data. Performance Monitor starts the necessary traces to collect all the information where as Query Monitor does not. This difference then manifests itself in the way Query Monitor classifies certain events.

8.19.3 Conclusion

The variations between the monitors is essentially in the representation of the data. The two products are perfectly complementary as monitoring agents. Query Monitor with its ease of use for rapid problem determination, DB2 PM with its detail and reporting capabilities.

Past Activity

Outline of scenario

- Introduction to past activity panel
- Unload of data to disk
 - Unload selected intervals
 - Unload of all unloaded intervals via JCL
- Load of data to history tables
 - Load an interval via Query Monitor panels
 - Load GDG generations via JCL
- Summary
 - Summary of past activity processing
 - Considerations to take into account



ibm.com/redbooks

© 2000 IBM Corporation

8.20 Past Activity panel

In this section we look at some detail the Past Activity processing within Query Monitor. All Thread Activity details are from option 2 off the main Query Monitor menu. The detail on the Thread Activity screens is essentially the same as for the Current Activity, except the interval start and stop times are displayed.

8.20.1 Intervals panel

This is an intermediate panel before the thread details screen. It gives a summary of the intervals that have been collected by Query Monitor. Apart from the interval start and stop times there are also a number of other pieces of information available.

Available column

This has three values as to where the data currently is.

- A **V** indicates that the data is still within DB2. That is, the data that is available for viewing through the past intervals panels. With Query Monitor defined as previously mentioned in this redbook (with 24 partitions and an interval equal to one hour), the past 24 hours of data is available.
- When the data has been overwritten in the partition, this value is changed to **N**.
- If the data has been loaded into the history table space, an **H** is displayed in this column.

Off-loaded data set

If the partition(s) has been unloaded to DASD, then this field will be populated with the data set name of the off-loaded data. The process of doing this will be

described later. If the available column is **N** and the off-loaded data set column is populated, then the data can be reloaded to the history table space (provided that the GDG generation still exists).

Interval summary data

Further to the right of the data set name, is a summary of the information contained in the interval. The total number of threads, total CPU time as well as other information is provided for reference.

8.20.2 Unload of data

There are two methods of unloading data stored in the DB2 tables to GDGs. The first method is to use the past interval panel to unload selected intervals, the second is to use the supplied JCL to unload all non-unloaded intervals.

8.20.3 Load of data

As with unload of data, there are two methods of reloading data from the GDG data set into the DB2 history table space. When either method of loading is selected, any data currently in the HISTORY table space is deleted before the selected intervals are loaded. This is to have as small an overhead on the system as possible. The data can always be reloaded if the GDGs exist, or data sets can be concatenated if more data is required for analysis. Remember that the load takes place via a program call and not through the LOAD Utility.

8.20.4 Summary

A short summary of Past Activity processing will look into some of the considerations to bear in mind when working with historical data.

Unloading selected intervals

```
CQM$INTV V1R1 --- DB2/QM Intervals --- 2000/11/22 17:29:31
Option ==> _____ Scroll ==> CSR
```

Valid Cmds: Select,Unload Data,Load history

Cmd	Avail	Start Date	Intv Nbr	Offloaded Dataset
-	Y	11/21/2000	0000000225	
U	Y	11/21/2000	0000000224	
-	Y	11/21/2000	0000000223	
-	Y	11/21/2000	0000000222	
-	N	11/20/2000	0000000221	
-	N	11/20/2000	0000000220	
-	N	11/20/2000	0000000219	
-	N	11/20/2000	0000000218	

Screen becomes populated with dataset name

Cmd	Avail	Start Date	Intv Nbr	Offloaded Dataset
-	Y	11/21/2000	0000000225	
U	Y	11/21/2000	0000000224	
-	Y	11/21/2000	0000000223	



ibm.com/redbooks

© 2000 IBM Corporation

8.21 Unloading selected intervals

An individual interval can be unloaded to a GDG from the Intervals Panel. A line command, U, is entered against the particular interval required. It should be noted that the data has to be available first. It can not be unloaded if it has been overwritten.

Unloading all available intervals

- Sample JCL supplied with Query Monitor
- Unload all available and non-previously unloaded data
- History interval panel populated with dataset name
- Tailored JCL below

```
//CQM@UDB2 JOB ,CQM,CLASS=A,MSGCLASS=X,
// NOTIFY=PAOLOR3
//*****
//* Sample unload of intervals
//*****
//UNLDINTV EXEC PGM=CQM@UDB2, PARM='DB2Y' (DB2 subsystem containing QM tables)
//STEPLIB DD DSN=DB2QM.SCQMLoad,DISP=SHR (QM load libraries)
//DB2PARMS DD DSN=DB2QM.DB2.CONTROL,DISP=SHR (QM parameter dataset)
//SYSOUT DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//CQM#DATA DD DSN=DB2QM.UNLOAD.DATA(+1), (GDG details)
// DISP=(NEW,CATLG,DELETE),
// DCB=PAOLOR3.PRINT1,
// UNIT=SYSDA,SPACE=(CYL,(50,100),RLSE)
***** Bottom of Data *****
```



ibm.com/redbooks

© 2000 IBM Corporation

8.22 Unloading all available intervals

This is done via a batch job, using the JCL supplied with Query Monitor. The supplied JCL needs to be tailored to individual installation standards, and those lines that need to be updated are annotated on the chart.

The batch job will create one GDG generation for all the intervals that have not previously been unloaded and have the status of available. This method of unloading the data is recommended over the unload of individual intervals as discussed earlier.

As all information is written to one GDG generation, the data stored in the GDGs can go back further in time. This job can be scheduled to run once per day, ideally when there are no Query Monitor users on the system, to unload all the day's data. If done at night, this ensures that the overhead of running Query Monitor remains very low.

Once the JCL has completed, the unloaded data set field on the intervals panel will be populated with the generation of the GDG.

Load of selected intervals

```
CQM$INTV V1R1 --- DB2/QM Intervals --- 2000/11/22 17:39:29
Option ===> _____ Scroll ===> PAGE
```

```
Valid Cmds: Select,Unload Data,Load history
```

```
-----
Cmd  Avail  Start Date  Intv Nbr  Offloaded Dataset
L    N      11/09/2000 0000000019 DB2QM.UNLOAD.DATA.G0002V00
-    N      11/09/2000 0000000028 DB2QM.UNLOAD.DATA.G0003V00
-    N      11/09/2000 0000000021 DB2QM.UNLOAD.DATA.G0002V00
-    N      11/09/2000 0000000018 DB2QM.UNLOAD.DATA.G0002V00
-    N      11/09/2000 0000000017 DB2QM.UNLOAD.DATA.G0002V00
-    N      11/09/2000 0000000016 DB2QM.UNLOAD.DATA.G0002V00
-    N      11/08/2000 0000000002 DB2QM.UNLOAD.DATA.G0001V00
-    N      11/08/2000 0000000001 DB2QM.UNLOAD.DATA.G0001V00
-----
```

- Existing data in HISTORY table space deleted
- Selected interval loaded and Avail indicator changes to 'H' as data loaded



ibm.com/redbooks

© 2000 IBM Corporation

8.23 Loading selected intervals

As with unloading of data, it is possible to load a selected interval into the history table space.

This can be done from the intervals panel by placing an **L** as a line command. When an interval is selected for loading (via panel or JCL), Query Monitor will delete all the data currently in the HISTORY table space.

Once the load is complete, the available indicator is changed to an **H**. Only individual intervals can be loaded this way, if more than one data set is required to be loaded, then the supplied JCL should be used. The next chart will demonstrate this.

Load via JCL

- Tailor JCL as appropriate to installation
- Data sets containing the unloaded data can be concatenated to load more than one generation
- Existing data in HISTORY table space deleted before data loaded
- Past interval panel populated with 'H' for intervals loaded

```
//CQM@LDB2 JOB ,CQM,CLASS=A,MSGCLASS=X,  
// NOTIFY=PAOLOR3  
//*****  
//*          INVOKE RQM@LDB2          *  
//*****  
//UNLDINTV EXEC PGM=CQM@LDB2,PARM='DB2Y'  
//STEPLIB DD DISP=SHR,DSN=DB2QM.SCQMLoad  
//DB2PARMS DD DISP=SHR,DSN=DB2QM.DB2.CONTROL  
//SYSOUT DD SYSOUT=*  
//SYSPRINT DD SYSOUT=*  
//SYSUDUMP DD SYSOUT=*  
//CQM#DATA DD DISP=SHR,DSN=DB2QM.UNLOAD.DATA(0)  
//          DD DISP=SHR,DSN=DB2QM.UNLOAD.DATA(-1)
```



ibm.com/redbooks

© 2000 IBM Corporation

8.24 Load via JCL

The supplied JCL should be tailored as with the unload JCL to reflect installation specifics. The tailoring will essentially be the same as the unload JCL. The advantage of loading using the JCL is that multiple data sets can be concatenated to be loaded into the HISTORY table space.

Summary of Past Activity

Points to consider

- Ensure GDG base has LIMIT high enough
 - consider how many days of data is required
- Command text is not stored
- Reorg/Runstat/Rebind table spaces
 - reclaim space and ensure optimal performance
- Schedule unload JCL once per day
 - will unload 'all' data to one GDG generation
- Concatenate GDGs to load more than one offload data set



ibm.com/redbooks

© 2000 IBM Corporation

8.25 Past Activity considerations

The following notes should be taken into account when using past and historical data.

GDG limit

Ensure that the GDG is defined to have a LIMIT large enough to ensure that the data is retained for the correct period. Using the formula below:

$$\text{LIMIT} = (\text{Number of offloads per day}) * \text{intervals/hour} * (\text{Number of days retention})$$

A LIMIT of 31 would allow a month's worth of data to be retained in the interval — one hour and one unload of the all partitions is scheduled per day.

To reduce the amount of DASD the unloaded data sets use, they could be defined to be archived to tape.

DB2 commands

The text of DB2 commands is not stored in the Past Activity.

DB2 housekeeping

All of the past and historical data is stored in DB2 tables, these should be periodically reorganized, RUNSTATED and dependant plans and packages rebound to optimize performance and reclaim space.

Unloading of past data

Schedule the JCL once per day to unload all available intervals to disk. This is the most efficient method of doing this.

Loading of data

Using the supplied JCL enables you to load more than one interval into the history table space.

DB2 Command Activity - 1

Monitoring DB2 command activity

- Command screen
 - Command issued displayed with timestamp
 - Text of command retained for current interval
 - Issuer identified
- Current thread activity screen
 - Commands are identified as thread with plan name blank or zero values
 - Subsystem command issued on identified
- Past activity screen
 - Commands are identified as thread with plan name blank
 - Text of command not retained


ibm.com/redbooks

© 2000 IBM Corporation

8.26 Command Activity processing

There are a number of places with Query Monitor, where DB2 Command Activity can be investigated. The amount of information that each provides varies, and this is discussed now.

Command screen

The command screen is Option 4 from the main menu. The most complete information about DB2 commands that have been issued in the current monitoring interval will be found on this panel. Although commands can be picked up on the Current Activity screen, this screen should be used in preference. An example screen is included in a following chart.

Current activity

Commands issued in the current monitoring interval also appear on the Current Activity screens. If it was issued interactively as a DSN command or from the DB2I panels, the thread will be picked up here. It is identifiable as having no plan associated with it, and zeros as the DB2 times. Commands issued via the stored procedure address space (for example from the Control Center) are detected by having zero DB2 times and a plan of DISTSERV. However, commands issued from the console are not collected, these can only be detected on the Command Activity screen. Commands issued via batch jobs are displayed on both the Current Activity and command screens.

Past activity

Only command information available from the current interval is available through the Past Activity panel; that is only DSN, DB2I and stored procedure issued commands. The text of the command is not stored. It does however give a

timestamp of when the command was issued to aid investigation, so that the correct part of the MSTR address space or SYSLOG can be found.

DB2 Command Activity - 2

CQM\$DCMD V1R1 ----- DB2 Commands Executed ----- 2000/11/21 12:41:53
 Option ==> _____ Scroll ==> PAGE

DB2 QM Subsystem: DBQM Current Interval Started: 11/21/2000 12:02:30

Job Name	Authid	Command Timestamp	Command Text	
PAOLOR3	PAOLOR3	11/21/2000 - 12:37:12	-TERM UTIL(UTIL1)	← DB2I
PAOLOR3	PAOLOR3	11/21/2000 - 12:36:51	-DIS THREAD	← Console
PAOLOR1	PAOLOR1	11/21/2000 - 12:36:39	-START DATABASE(PAOLOR12)	
DB2YMSTR	SYSOPR	11/21/2000 - 12:36:32	-DIS UTIL(*)	
PAOLOR1	PAOLOR1	11/21/2000 - 12:36:27	-STOP DATABASE(PAOLOR12)	← Stored
DB2YSPAS	RMRES01	11/21/2000 - 12:22:06	-STOP TRACE(MON) TNO(04)	proc
DB2YSPAS	RMRES01	11/21/2000 - 12:22:06	-START TRACE(MON)	

***** Bottom of Data *****



ibm.com/redbooks

© 2000 IBM Corporation

8.27 DB2 commands executed

The command screen (option 4 from the main menu) for the current monitoring interval should be used to display the commands issued. It gives the most complete picture of activity. It is the only place where the text of the command issued is reported. However to determine the subsystem the command was issued on (if monitoring more than one subsystem), the Current Activity screen needs to be used. As commands are asynchronous, this screen does not imply that the command was successful, only that it was issued.

DB2 Command Activity - 3

```

CQM$DTHR  V1R1  ----- DB2/QM Thread Activity -----      2000/11/21  12:40:43
Option  =====> _____ Scroll =====> PAGE
-----
DB2 QM Subsystem: DBQM      Current Interval Started: 11/21/2000  12:02:30
-----

Display Threads:  I  (Active/Interval)      D  (Detail/Summarize)
-----

Cmd  SSID Intv  Plan Name Job Name   DB2 CPU Time   DB2 Elapsed   SQL Calls
--  --  --  --  --  --  --  --  --  --
-   DB2X  -   ALASQLP  PAOLOR1B    0.023081     1.359726       3
-   DB2X  -           PAOLOR3    0.000000     0.000000       0
-   DB2Y  -           PAOLOR3    0.000000     0.000000       0
-   DB2Y  -           PAOLOR1    0.000000     0.000000       0
-   DB2Y  -           PAOLOR1    0.000000     0.000000       0
-   DB2Y  -   DSNESPCS  PAOLOR1    0.068521     3.040212      13
-   DB2Y  -   ANL4QMF  PAOLOR3    0.067818     0.632690      27
-   DB2Y  -   DISTSERV  DB2YSPAS    0.000000     0.000000       0
-   DB2Y  -   DISTSERV  DB2YDIST    0.041771     2.601946      90

```

Commands


ibm.com/redbooks

© 2000 IBM Corporation

8.28 DB2 Command Activity

From the Current Activity screen, commands issued from DSN, DB2I, batch or distributed threads. These are highlighted with an arrow on the chart. In general they are identified by having zeros in the elapsed and CPU columns. Commands issued interactively also have the plan name field blank.

This panel additionally supplies the subsystem where the command was issued.

Note that commands issued from the console are only displayed for the current interval in the Command Activity panel.

DB2 Utilities and Binds

```

CQM$DTHR V1R1 ----- DB2/QM Thread Activity ----- 2000/11/21 13:06:25
Option ==> _____ Scroll ==> CSR
-----
DB2 QM Subsystem: DBQM      Current Interval Started: 11/21/2000 13:02:41
-----

Display Threads:  I  (Active/Interval)      D  (Detail/Summarize)
-----

Cmd  SSID Intv Plan Name Connectn      APPL CPU      APPL Elapsed  Total Getpa
-   -   -   -   -   -   -   -   -   -   -   -   -   -   -   -   -   -
-   DB2Y -   DSNUTIL UTILITY          0.000000      0.000000
-   DB2Y -   DSNUTIL UTILITY          0.113922      0.616575
-   DB2Y -   DSNUTIL UTILITY          0.704231      6.796758
-   DB2Y -   DSNUTIL UTILITY          7.762895     1:47.960518
-   DB2Y -   DSNUTIL UTILITY          1.667687     15.307225
-   DB2Y -   DSNBIND BATCH             0.000000      0.000000
-   DB2Y -   DSNBIND BATCH             0.000000      0.000000
-   DB2Y -   DSNBIND BATCH             0.000000      0.000000
-   DB2Y -   DSNBIND BATCH             0.000000      0.000000
-   DB2Y -   DSNBIND BATCH             0.000000      0.000000

```


ibm.com/redbooks

© 2000 IBM Corporation

8.29 DB2 Utilities and Binds

Query Monitor also provides information about Current Activity panel by their plan name, for example, DSNUTIL for Utilities and DSNBIND for Binds. In addition the connectn column also shows the connection typed as UTILITY for Utilities.

Query Monitor displays each call to the Utility. In the above example, for the Utility, the batch job ran once, but issued five Utility statements, in the example it was actually executing RUNSTATS for five table spaces. It is therefore possible to see the length of time each statement took very clearly.

Although no detailed information is collected on these threads by Query Monitor, the time frame when the thread was active is easily identified.

Summary of Query Monitor

- Flexible and easy to use monitor
 - ISPF panel driven with on-line help
- Monitor many DB2 subsystems concurrently
 - can define upto 64 to one QM subsystem
- Low system overhead
 - combining MVS data spaces, DB2 tables and offloads to DASD
- Thread exceptions easy to spot
- Dynamic filtering of collected information
 - eliminates 'information overload'
- Historical data available for analysis


ibm.com/redbooks

© 2000 IBM Corporation

8.30 Summary of Query Monitor features

In this section we summarize our considerations on the features of DB2 Query Monitor.

Ease of use

Query Monitor is an ISPF panel driven system with extensive online help. It is very easy to use and very little training and familiarization is required to exploit its features. Color coding allows for quick identification of active and exception threads.

Monitor up to 64 DB2 subsystems

Up to 64 DB2 subsystems can be monitored concurrently, with all the collected data available from one source.

Low system overhead

A combination of the use of MVS data spaces for the current interval, DB2 table spaces and off-loads to DASD allow for a very efficient and low overhead subsystem monitor.

Exception processing

As soon as threads have exceeded pre-determined limits, they are flagged on the screen as an exception, and can be easily identified.

Dynamic filtering

“Information overload” can be eliminated by using application profiles to dynamically reduce the information displayed.

Historical data

Unloads of the past intervals can be performed to retain historical data for subsequent loading and analysis.

Query Monitor for DB2 for OS/390

Query Monitor hints and tips

- If in doubt, press <F1>
- Ensure Accounting traces class 1,2,3 are active
- Collect all information, filter via application profile
- Individuals can have own application profile
- Unload all day's intervals once per day
- If 'Display SQL in order...'
 - N - SQL text will be last statement issued only
 - Y - SQL of all statements displayed
- Remember DB2 maintenance of QM objects
- Cancel threads through Query Monitor to resolve problems
- Filtering - don't exclude all and include specifics
- Choice of where to store DB2 information


ibm.com/redbooks

© 2000 IBM Corporation

8.31 Query Monitor hints and tips

The following tips and hints may prove useful when working with Query Monitor.

Help

Remember extensive online Help is available by pressing the F1 key or typing `HELP` as a command on the panels.

Traces

The accounting trace with classes 1, 2 and 3 should be active. If threads are being displayed with elapsed times of several hours when they have only been active for seconds, then this probably means that not all of the correct traces classes are active (probably class 2). Query Monitor dynamically detects when the traces have been started.

Monitoring versus application filter

As long as there is not too great a system overhead, do not filter via a monitoring profile, but filter via an application profile. This ensures that all necessary data is collected that can be used for analysis. If the data has not been collected it can not be used later! The application profile can be used to filter unwanted data for specific investigations and can be changed dynamically without interrupting the collection of data.

Individual users of Query Monitor can all have an individual application profile if necessary.

Unload of Past Intervals

Schedule one unload per day of all the past intervals. Using the supplied JCL is the most efficient method of doing this.

Display SQL in order of execution

Remember to switch the Y/N via an application profile to see all SQL instructions (if Y) or a summary by statement type (if N). If N is selected, the SQL text is of the last statement issued.

DB2 housekeeping

The underlying DB2 objects should have the standard Utilities run against them for performance and recoverability according to the installation standards.

Cancel thread

Threads can be canceled through Query Monitor, but only if you have the appropriate DB2 authorizations.

Filtering

Remember not to exclude all with a wildcard and then to include specifics, but rather include all with a wildcard and then exclude specifics.

Where to store information in DB2

Query Monitor gives you the choice of which DB2 subsystem to store the past information in. The only condition is that it has to be the first subsystem specified in the MONITOR parameter in the start-up JCL. It can be defined to a less heavily used subsystem if required.

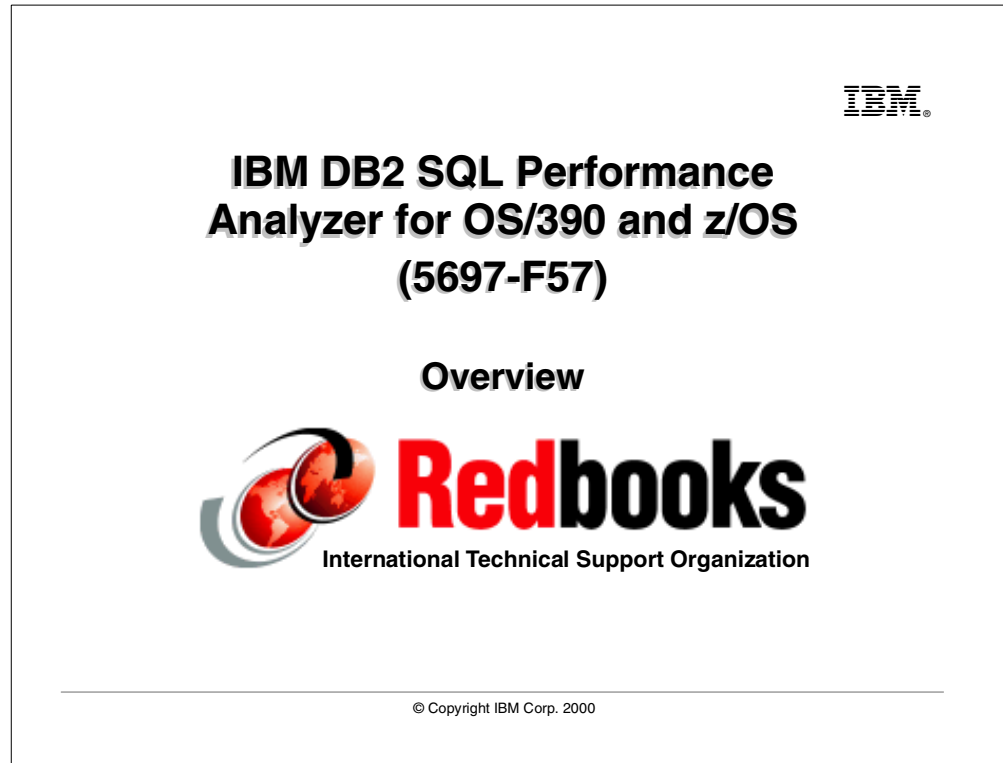
Part 4. IBM DB2 SQL Performance Analyzer

In this part of the book we provide:

- An overview
- Structure and configuration
- Usage and consideration

of IBM DB2 SQL Performance Analyzer for OS/390 and z/OS.

Chapter 9. Overview

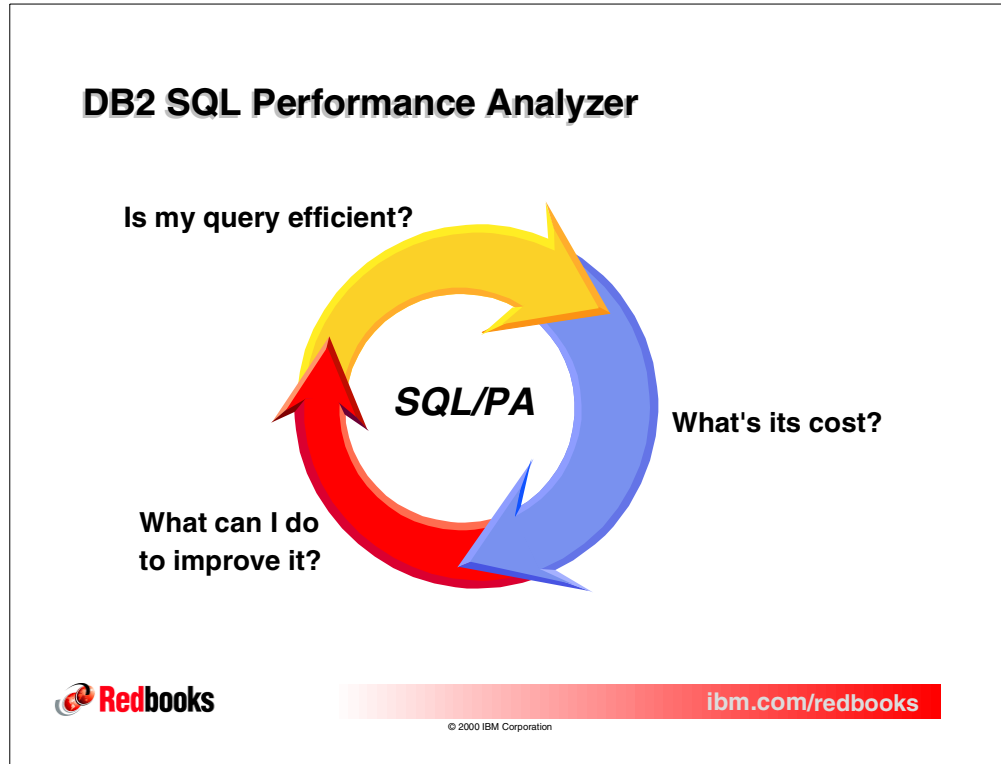


Today the IT department's approach to application implementation is dominated by limited budgets and short delivery times forcing Data Management professionals to cut corners. A common victim of this approach is the consideration for performance.

Performance analysis and tuning is an activity that requires skilled personnel, time and system resources; typically these assets are not always available and when considering the associated costs, they are not easily granted.

End users performing ad-hoc queries for decision support tend also to ignore resource usage issues, often causing a waste when their queries are cancelled.

The purpose of DB2 SQL Performance Analyzer for OS/390 and z/OS is to help database users, developers and administrators give the right consideration to SQL performance in an effective and timely manner, providing expert advice and easy to understand cost estimates.



9.1 Scope and objectives

DB2 SQL Performance Analyzer (SQL/PA) provides realistic costs and resource usage for DB2 SQL statements *without* having to execute them.

It also incorporates a comprehensive set of warnings, alerts, recommendations and guidelines, collectively known as **SQL Advisor**, to assist users in understanding what their queries are going to do and how to improve their performance.

SQL/PA can be executed in batch or through its own TSO/ISPF interface as well as through a Stored Procedure, allowing interface to IMS, CICS, and network based applications.

In addition SQL/PA provides a real time QMF Intercept module that preempts the QMF Governor and may avoid execution of resource intensive queries that otherwise would be cancelled, preventing in this way loss of useful CPU cycles.

9.1.1 Prediction and prevention

DB2 SQL Performance Analyzer is based on the belief that “trial-and-error” is not the most cost effective solution to resource intensive SQL access. Likewise, the existing facilities to limit the amount of resource used by a given query can only stop its execution. They cannot avoid the loss of already used resources.

SQL/PA is designed to give users a realistic assessment of the cost associated with SELECT, INSERT, UPDATE and DELETE statements along with Joins, Unions and Subqueries, *before* they are executed.

The prediction capability of SQL/PA is accomplished through embedded technology that merges skills of predictive modeling for online systems and a collection of extensive benchmarking results of DB2 operation, covering every release since V1.2

9.1.2 SQL Advisor

Over the past years, we have seen a relatively rapid sequence of DB2 releases, introducing new features and enhancement for the benefit of the business community.

The increasing demand placed on Data Management specialists does not always allow the necessary time and exposure to keep up with the innovations: for this reason SQL/PA integrates a very useful and knowledgeable 'expert system'.

SQL Advisor is a fully automated collection of advises that can pinpoint trouble spots, anomalies, poor design, and general weaknesses in SQL and database existing structures.

SQL Advisor is both SQL context and DB2 version sensitive so that it limits advice to the specific SQL statement type and the DB2 releases you choose for the run.

Users ranging from novice to expert can find useful advice in the three levels of detail that the component provides:

- Alerts and warnings
- Recommendation and performance notes
- Guidelines and good news

A good news type of message when the job is well done provides a way to appreciate progression in your skills and expertise with SQL for DB2 for OS/390.

DB2 SQL/PA Objectives

- To provide easy to understand resource usage information and costs associated with SQL queries ***without having to run them***
- To avoid trial-and-error scenarios and the consequent resource waste
- To supply an expert system to overcome lack of skills and to quickly pinpoint opportunity for improvement
- To help end-users to understand the real cost of their database requests, providing the monetary value associated with each query before execution
- To speed up and enhance the work of application programmers and database designers, providing SQL and database design guidelines and recommendations
- To increase accuracy of capacity planning models, providing realistic base input information



ibm.com/redbooks

© 2000 IBM Corporation

9.2 Objectives

SQL Performance Analyzer evaluates SQL statements executing from any attach facility predicting their costs without engaging an actual execution. The costs provided are in terms of:

- Elapsed time
- CPU time
- I/O count
- Monetary value
- QUNITS (Query CPU units)

While Elapsed time, CPU time and I/O count are self-explanatory, we have to clarify the meaning of Monetary value and QUNITS.

Monetary value is actual cost of the query, reported using the currency of your choice. It is calculated using configured base costs for CPU and Connect time as well as I/O counts.

Query Units (QUNITS) is a cost unit introduced by SQL/PA. It is a specially derived valued similar to *timerons* that help users, who are not familiar with resource usage values, to get a quick comparison of two different access paths. It is based primarily on CPU processing so that an I/O bound query will produce a lower QUNITS value rather than a CPU bound one. Please note that the CPU processing associated with I/O is also included in this value.

SQL/PA is complemented by an expert system, SQL Advisor, that saves time in identifying trouble spots as well as opportunities for improvement, providing

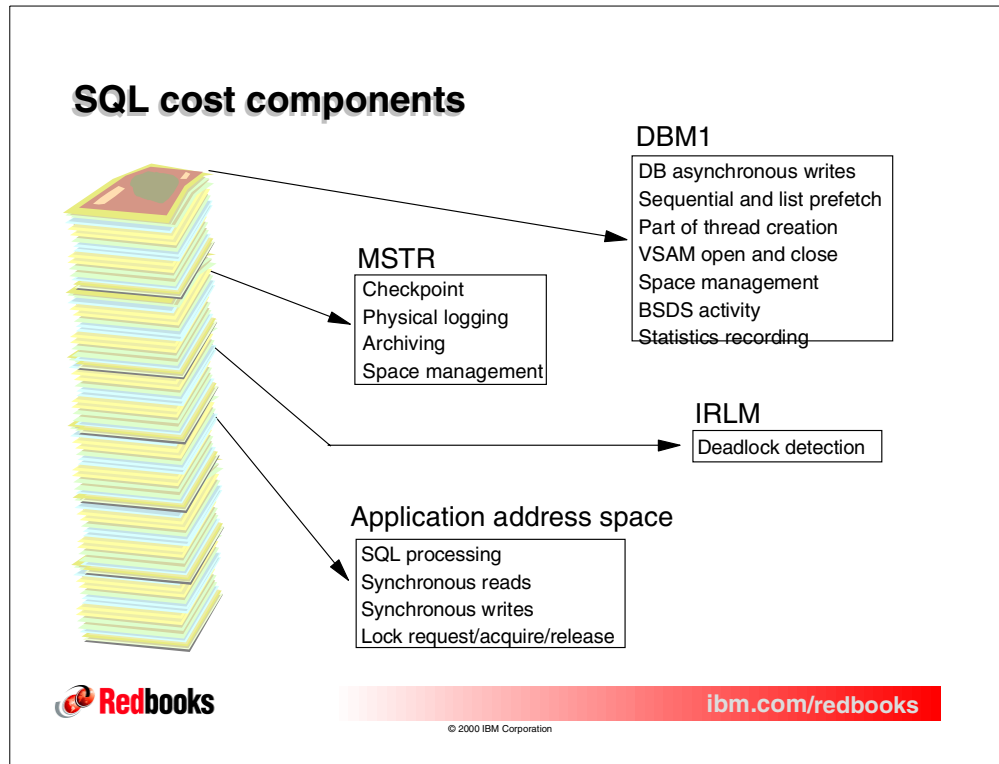
recommendations and guidelines built upon several years of experience in performance and tuning.

The product helps to use valuable computing resources appropriately, evaluating SQL statement behavior and its performance in advance of its execution. This technology can also be enabled in QMF to inform the end user of the monetary cost of the work that is going to be introduced into the system and of any exceeding resource usage thresholds.

The program can then ask the user to confirm the execution or, optionally, can be configured to cancel the request automatically without any prompt.

SQL/PA is useful in several circumstances:

- Application analysts can use SQL/PA to highlight those statements which consume excessive resources and to code and test alternative ways of processing. This process can save time and resources during the development cycle while improving the application quality and reducing the need for future maintenance
- Database designer can also use this tool to define proper indexing structures by testing critical SQL statements and comparing the cost estimates of the selected access paths
- The ability to configure the target host environment together with the option to populate the DB2 catalog with the most appropriate statistics, allows database analysts to simulate the expected performance of a given SQL statement under different hardware and/or data volume.
- When performing capacity planning for a DB2 application, SQL/PA can assist to gather input information of the basic building blocks of the predictive model, the SQL statements.



9.3 SQL cost components

The impact of an SQL statement execution is wider than what can appear from an initial inspection; it's a matter of fact DB2 performs several activity on behalf of that same SQL statement.

The most visible costs of an SQL statement are those directly associated with the SQL execution itself, that is with the application task. Some of these costs are:

- SQL processing, including the activity performed by the Relational Data System (RDS) and the Data Manager (DM)
- All synchronous I/O, both reads and writes
- Locking activity, including lock request, acquire and release

Other less apparent activity are performed by the DB2 tasks that, in some cases, have to carry the associated cost instead of the application task that originated the request. An example of these costs are:

- Sequential and list prefetch cost is not directly associated with the originating task, however it can be determined from DB2 tracing information
- Asynchronous writes often are performed when the triggering task has already completed, therefore their cost is charged to the DBM1 task

Only part of the thread management is actually recorded in the DB2 SMF records and therefore this overhead is usually pro-rated within the general cost of the DB2 tasks.

DB2 SMF records

SMF 100

Sequential and List prefetch, EDM Pool I/O, Asynchronous I/O ...
Thread logging, Lock Management ...

SMF 101

Open/Close, Getpages, Stage 1 and 2 processing ...
Fetch row, Sync/Commit, Lock request ...

SMF 102

Performance and Audit details



ibm.com/redbooks

© 2000 IBM Corporation

9.3.1 DB2 tracing and SMF records

To find out all costs components of an SQL statement, or generally the cost of the activity performed by a DB2 thread, normally requires the deployment of traces and the execution of the query.

DB2 traces collect statistics, accounting and performance information associated with DB2 threads while they are running and stores them in System Management Facility (SMF) records. The cost and resource usage associated with an application thread are known as DB2 Accounting information and are recorded by the DB2 Accounting trace.

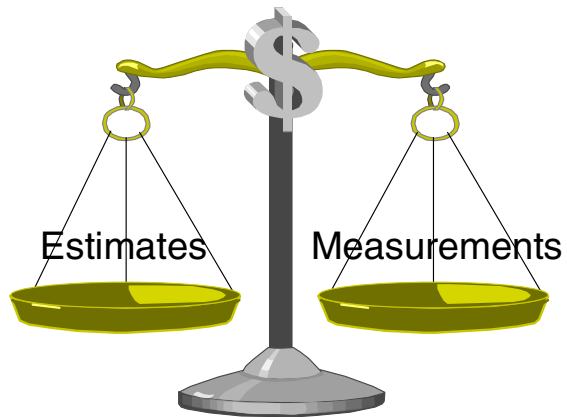
There are three types of SMF records used by DB2: 100, 101 and 102. The Accounting information is spread out over the first two (100 and 101) depending on the Accounting trace classes activated.

SMF record type 102 is used by Performance and Audit traces and it stores detailed information associated with each single SQL statement.

It is important to remember that tracing activity produces additional overhead on the system workload; its cost is directly related to the type and class of the DB2 trace used, and in general terms we can assume that the more details you want the bigger price you have to pay.

Note: SQL/PA does not use SMF records.

SQL/PA and other tools



ibm.com/redbooks

© 2000 IBM Corporation

9.4 SQL/PA and other tools

There are other tools that have functionality similar to SQL/PA and in most of the cases they are not comparable because of the different fundamental approach adopted for their design. It is the typical case of comparing “apples” with “oranges”. In fact, SQL/PA projects *what it will be* while other tools report *what has been*, and in some case *what it is*.

Let us look at some of the other tools and at their similarities and differences.

9.4.1 DB2 PM

To assemble a thorough and complete picture of the costs associated with an SQL operation you must execute the statement and then collect the resource usage information from the various SMF records written by DB2 traces.

There are specialized tools available to perform such work, an example of these is DB2 Performance Monitor (DB2 PM), however they are often deployed by System Programmer and specialized DBA because their complexity.

They also tend to be deployed primarily for production environments and often in a reactive way given the impact they impose on the system.

SQL Performance Analyzer is a simpler tools that addresses the needs of fast development and software cost control. It does not require DB2 tracing activity, nor the actual execution of the SQL statement, and therefore the system is not burdened with additional workload.

The results SQL/PA produces are *estimates* and could be different than the real execution measurements, in fact SQL/PA's strength is to provide an indication of poor performance and to advise about correction, not to report actual execution measurement.

SQL/PA does not compete with DB2 Performance Monitor, but it offers the opportunity to develop better applications while assisting less experienced staff to walk through the dungeons of DB2 performance.

9.4.2 DB2 Estimator

Very similar to SQL/PA, DB2 Estimator is the most comparable of all. It provides performance modeling capability but not in an easy way.

The model definition in DB2 estimator is almost entirely manual and therefore can be very cumbersome, it requires download of the catalog information and the SQL. Conversely SQL/PA runs directly on the host where DB2 and your applications and catalog are, and there is very minimum manual intervention to define the models.

In addition SQL/PA provides governing capability, warnings and guidelines through SQL Advisor and Explain information in a sentence-like style.

9.4.3 QMF Governor

The governing capability of QMF is limited to CPU time and number of records fetched. It provides thresholds for which the user receives warnings of the imminent termination of the thread: the choices are to self-terminate or wait for QMF to do it. The work and the resource used up to the termination point are then wasted.

As SQL/PA works on a prevention basis it informs you when exceeded the limits before executing the query, providing easy to understand information and giving the possibility to run the query anyway, or automatically prevent the execution.

The governing capability of SQL/PA are more advanced and usable than those provided by the QMF Governor.

9.4.4 QMF HPO

QMF High Performance Option has two main components, it converts SQL queries into Cobol programs and it governs executions based on historical data.

While the first component is unique to QMF HPO and not provided by SQL/PA, the governing capability are based on different assumptions. QMF HPO collects historical performance data and then compares the current execution, if it goes beyond the query is terminated. Once again the resource used up to the termination point are wasted.

QMF HPO provides an exit through which it could be possible to integrate the governing capability of SQL/PA with the transformation functions of QMF HPO in order to get the best of the two products.

How does it do it ?



Operations path length



Power rating of the target system



Overheads



ibm.com/redbooks

© 2000 IBM Corporation

9.5 How SQL/PA determines cost

The key factors that SQL/PA uses to determine the cost of an SQL statement are:

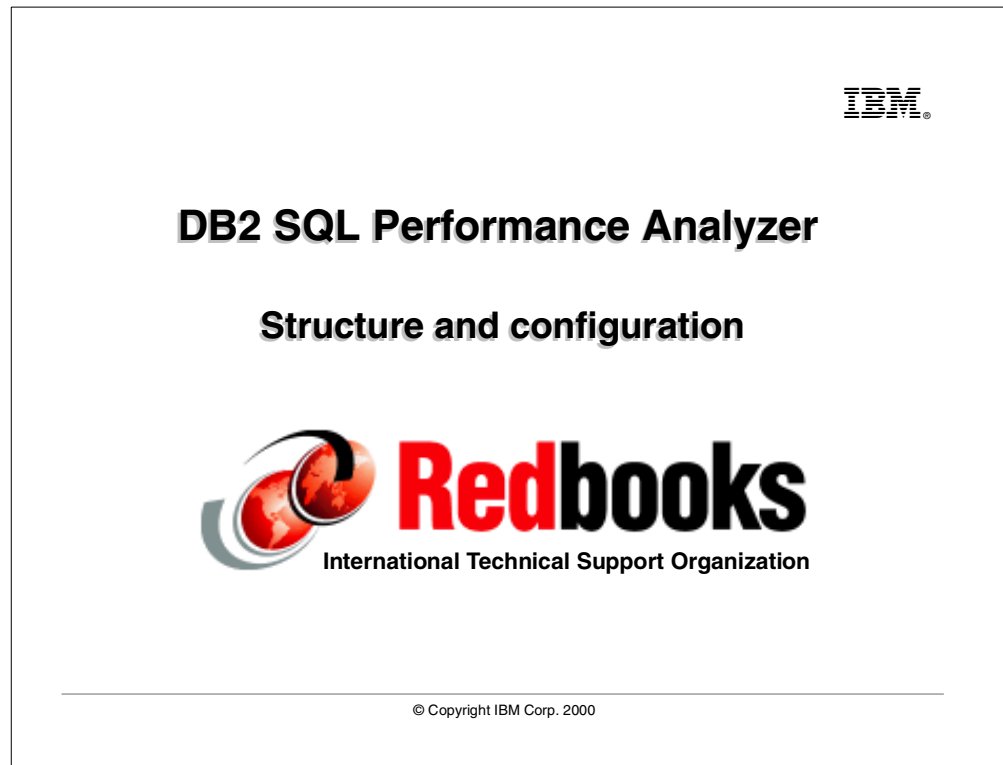
- The path length of the DB2 operations for various releases
- The power rating of the target system, partial or total
- The overhead involved with any of the DB2 operations

Using Explain information SQL/PA assembles, in an incremental way, the complete picture of the query processing in terms of path length, that is the number of instructions needed to perform the query.

It then adds the appropriate overheads and finally uses the rating of the target system to calculate the costs, simply performing an operation like:

$\text{Path Length} / \text{MIPS} = \text{CPU seconds}$

Chapter 10. Structure and configuration



The purpose of this chapter is to list all key components of DB2 SQL Performance Analyzer as well as to highlight some aspects of its configuration with the objective to assist Data Management specialists to better understand the tool and its functions.

The tool is covered by IBM Support and the reader should refer to the standard IBM channels for problems and fixes. Furthermore, SQL/PA is installed using a standard SMP/E process and we assume that the reader is familiar with this facility.

Customizing SQL/PA for your local environment is covered in *DB2 SQL Performance Analyzer for OS/390 Installation Guide Version 1*, SC27-1003. Refer to this manual when installing and activating the SQL/PA.

The approach

Parse input

- sequential , PDS member or DBRM
- identify SQL, key clauses and predicates

Explain

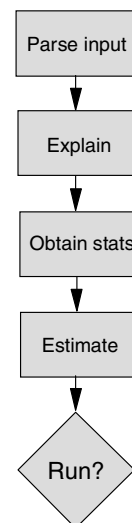
- perform explain against generic or private plan tables
- identify access path

Obtain stats

- retrieve catalog statistics for objects used in access path

Estimate

- use collected data with internal cost formulas to generate output


ibm.com/redbooks

© 2000 IBM Corporation

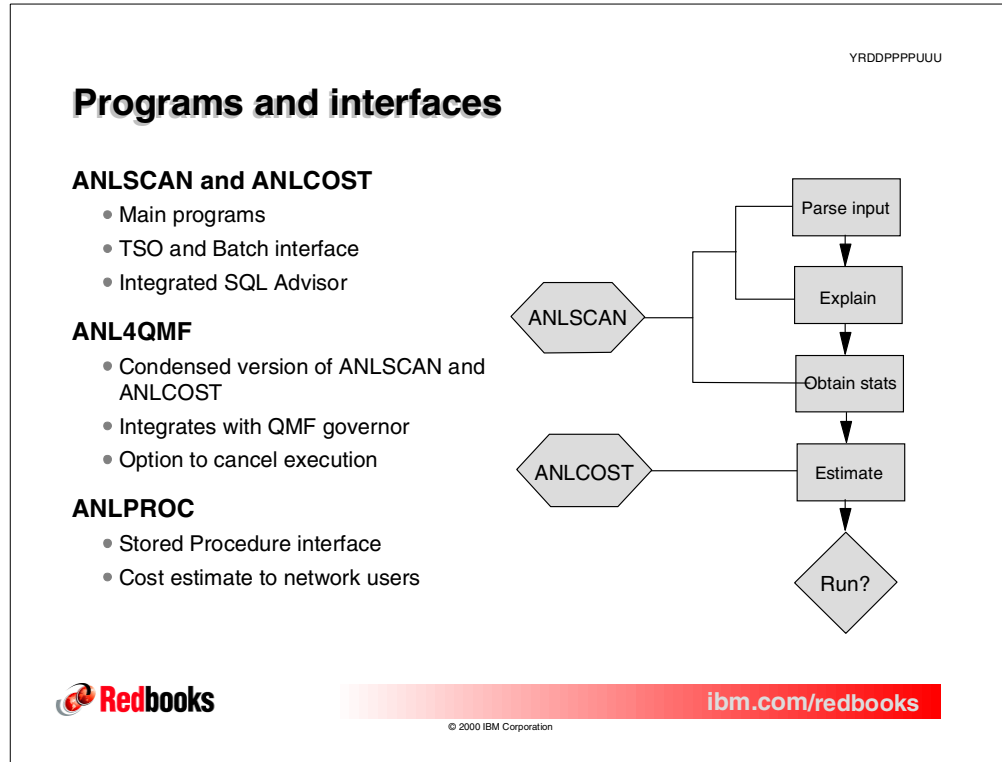
10.1 The approach

The main objective of SQL/PA is to answer the question: *What is the cost of this SQL statement?*

The product performs a series of operations that can be summarized as:

1. It takes input from one or more SQL statement(s) in the form of a sequential file, a PDS member or a DBRM and automatically extracts the executable SQL statements.
It then parses each statement collecting information about key clauses like OPTIMIZE FOR n ROWS, ORDER BY, DISTINCT as well as predicates, such as =, >, BETWEEN and stores them for later use.
2. It invokes EXPLAIN for each statement using a generic or primary PLAN_TABLE, and collects the DB2 optimizer's selection of access paths.
3. It obtains information, such as size and cardinality of the objects participating in the selected access by accessing the DB2 catalog.
4. It produces the resource usage estimate and the cost assessment of each SQL statement, using the information collected and various formulas built upon extensive benchmarking of DB2 operation.

At the end of this process, the user will be in a position to make an informed decision about the quality and potential impact of the SQL statement in question.



10.2 Programs and interfaces

SQL/PA is made up by two main programs: ANLSCAN and ANLCOST; together they cater for SQL/PA execution in batch and under the product's TSO/ISPF interface.

SQL/PA can also be executed in QMF and using Stored Procedure; for this purpose the two main programs have been condensed in a lighter, but still efficient, version: ANL4QMF for the QMF environment and ANLPROC for Stored Procedure.

10.2.1 Parsing program ANLSCAN

This program performs the input parsing, the Explain invocation and the DB2 catalog access. During the input parsing phase it assigns Explain query numbers to each SQL statement starting from 100,000,001 for SQL found in sequential files or PDS members or using 100,000,000 plus the statement number for SQL found in a DBRM.

Let us consider the following two examples:

- One sequential file with two SQL statements will produce the following statement numbers:

```
100,000,001
100,000,002
```

- One DBRM containing three statements numbered 813, 1578, and 2104 will produce the following:

```
100,000,813
```

100,001,578
100,002,104

Using this expedient your own explained statements can be preserved when using primary PLAN_TABLE instead of generic PLAN_TABLE. For this reason we recommended not to use Explain query numbers above 100,000,000 for your own Explain.

The DB2 catalog access is optimized to perform one access for each object even if there are many SQL statements referring to the same table or index for the same SQL/PA execution.

10.2.2 Costing program ANLCOST

This is a very fast and efficient program that produces all the final costs, resource usage and relevant reports generated by SQL/PA. The program performs various calculations and it combines them with the use of intelligence data collected in several years of DB2 operation benchmarking, under different conditions, and with different versions of DB2.

Using the key SQL clauses and predicates together with the DB2 catalog statistics for the given objects it calculates the filtering effects and estimates the portion of table and indexes that will be accessed as well as the number of rows.

The program takes also in consideration a number of parameters describing the target host environment that may affect the final performance prediction. Among these are:

- The MIPS rating of the target host machine
- The total number of engines as well as the number of engines dedicated to a specific logical system (LPAR)
- The type of disk used

In addition the user can request to replace the program's calculated CPU estimates with the one generated directly by the DB2 optimizer. In this case the user must also have a generic or private DSN_STATEMENT_TABLE created and accessible.

10.2.3 QMF Intercept program ANL4QMF

The QMF Intercept program essentially provides a small report about the cost associated with the SQL submitted and in case any of the configured cost limits is exceeded it prompts the user to confirm its execution.

The control parameter can be configured to make this program completely transparent, this means that the prompt and associated report can be displayed only when at least one threshold is exceeded. Furthermore the QMF Intercept program can be customized to automatically cancel the query execution instead of prompting the user for confirmation.

In any case ANL4QMF displays a message to clarify the reason of the cancelled request.

Such integration with the QMF Governor is achieved through a modification to the governor exit DSQUEGV1 that triggers the calculation of the query costs and

relative comparison with the configured limits before the query is actually executed.

The modification to the QMF Governor exit consists of a few additional lines of code for the QMF exit DSQUEGV1 and the driving program ANLGOV1 that provides the parameter lists.

The QMF Governor exit is executed only for interactive sessions, so that any batch processing can bypass the governor control. The same convention applies to the QMF Intercept program of SQL/PA.

Considering the interactive nature of the QMF application and the focus on response time of its users, the additional steps of costing the query may raise some concern. For this reason the QMF Intercept program has been designed to minimize as much as possible any overhead.

10.2.4 DB2 stored procedure ANLPROC

This program is essentially very similar to ANL4QMF because it combines the functions of both ANLSCAN and ANLCOST. It allows SQL/PA functions to be operated from anywhere in the network, including IMS, CICS and remote client applications.

A simple SQL CALL instruction is required to interface SQL/PA's stored procedure, passing a parameter list that includes the SQL statement you want to analyze, its length and the required user parameter list.

The program uses the Call Attach Facility (CAF) and it executes in the DB2 Stored Procedure Address Space.

Note: when installing SQL/PA you should avoid using the SQL/PA load library SANLLOAD in the DB2 Stored Procedure Address Space procedure. In most of the cases when you install SQL/PA DB2 is already up and running, therefore you should copy the modules ANLPROC and ANLMODA to the load library already allocated to the STEPLIB or JOBLIB of the DB2 Stored Procedure Address Space.

10.2.5 Binding the programs

A bind is required as the programs perform accesses to the PLAN_TABLEs as well as a special Registry table that will be discussed in more detail later. The only programs that require a bind are ANLSCAN, ANL4QMF and ANLPROC.

SQL/PA user must have the appropriate authorization to execute the relevant plan, therefore the plans needs to be granted accordingly. Failing to do so will cause a return code 500 from the Call Attach Facility. The same return code is also produced if you change the registry table without rebinding the programs.

In case you do not want to use generic PLAN_TABLE you must replace the QUALIFIER value of the bind with a valid qualifier identifying an existing PLAN_TABLE; if such PLAN_TABLE does not exist the bind will fail.

Special features

Generic PLAN_TABLEs

- Designed for concurrency avoiding one PLAN_TABLE per user
- Registry table to govern usage and prevent conflicts
- Preserve personal PLAN_TABLE
- Requires modification to DSN3@ATH exit
- Not usable with Stored Procedure interface

Registry table

- A special DB2 Registry table is used to record the status and usage of each generic PLAN_TABLE to avoid any conflict with concurrent users.

DB2 Catalog access

- One access per object regardless of how many query reference it
- Information on up to 100 objects cached in memory

QMF interface response time

- PL/1 environment set up and removal minimized
- Disk I/O avoided by using memory resident array
- Explicit parameters hard-coded into exit


ibm.com/redbooks

© 2000 IBM Corporation

10.3 Special features

Special solutions have been adopted to address potential performance and administration issues of SQL/PA. This section briefly describes the most important features.

10.3.1 Generic PLAN_TABLEs

The purpose of the Generic PLAN_TABLEs is to minimize the administration effort of an environment with many SQL/PA users and to provide performance improvements.

One of the step involved in the analysis process of SQL/PA is to perform an Explain of the input SQL statement and the tool can use either personal or generic PLAN_TABLEs.

In this context a personal PLAN_TABLE is owned by the primary authorization ID executing the Explain command, while the generic PLAN_TABLEs are owned by some of its secondary authorization IDs.

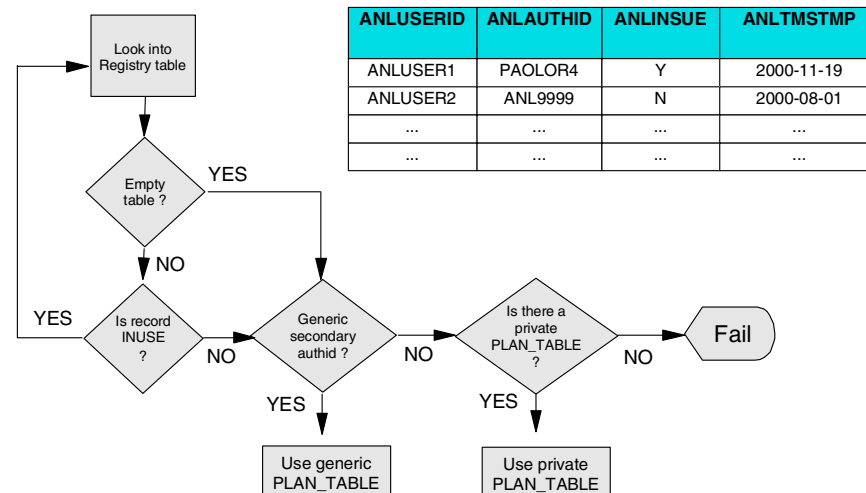
The mechanism revolves around an optional modification to the DB2 Authorization exit DSN3@ATH. The SQL/PA modification intervenes almost at the end of the exit process checking for available (empty) secondary authorization ID slots. If any are available it populates as many as required (default is 10) with generic SQL/PA ID: these are the owners of the generic PLAN_TABLEs.

SQL/PA detects automatically if generic PLAN_TABLEs are used and in this case it performs a SET CURRENT SQLID instruction before to perform the Explain, so that the Explain output can be addressed to one of the generic PLAN_TABLE.

Note: The modification to the DB2 authorization exit DSN3@ATH and the consequent usage of generic PLAN_TABLEs is completely optional as SQL/PA works well with either generic or private PLAN_TABLE. However generic PLAN_TABLEs enable better performance because they all reside in a segmented table space and are cleaned up with a mass delete instruction.

Furthermore generic PLAN_TABLEs cannot be deployed when using the Stored Procedure interface. This is due to the restriction in using SET CURRENT SQLID within Stored Procedures using CAF.

Using the Registry table


ibm.com/redbooks

© 2000 IBM Corporation

10.3.2 Using the Registry table

A special DB2 Registry table is used to record the status and usage of each generic PLAN_TABLE to avoid any conflict with concurrent users.

The table contains four columns:

- **ANLUSERID:** The value of this column identifies the creator of the PLAN_TABLE selected for use; at initialization this column is populated, by default, with ten IDs ranging from ANLUSER1 to ANLUSERA.
- **ANLAUTHID:** Indicates the primary authid using a given PLAN_TABLE; after installation this column contains the generic value ANL9999 that will be replaced every time the relevant table is used with the primary authid of the SQL/PA analysis requester.
- **ANLINUSE:** This flag indicates if the relevant PLAN_TABLE is in use
- **ANLTMSTMP:** This is the timestamp of last update. The record is updated when the relevant PLAN_TABLE starts to be used (ANLINUSE=Y) and when it is released (ANLINUSE=N).

Under particular circumstances, for example the thread is cancelled during the Explain process, SQL/PA may be prevented from updating the Registry table correctly and therefore one or more generic PLAN_TABLE may appear to be constantly in use.

A simple SQL update statement can fix this problem, an example of this is provided in the SANLSQL member ANLRESET.

Note: The Registry table is required by the ANLSCAN program even if you do not modify the DB2 authorization exit DSN3@ATH and therefore you do not use the generic PLAN_TABLE. When using personal PLAN_TABLE, you may optionally add the primary authid of the SQL/PA user as ANLUSERID or let the ANLSCAN program select the *Current SQLID*. In this case you must keep the Registry table empty and make sure that all SQL/PA users have their own PLAN_TABLE and DSN_STATEMNT_TABLE. Since you have no generic PLAN_TABLE, when binding the SQP/PA programs you need to set the QUALIFIER to one of the existing PLAN_TABLE and DSN_STATEMNT_TABLE owners.

10.3.3 Catalog access

The DB2 catalog is a shared resource and its access can impact the response time of SQL/PA, especially when analyzing several SQL statements in one run.

To minimize the access, SQL/PA deploys a caching mechanism to store the catalog information of up to 100 objects for each type in memory, that is 100 table spaces, 100 tables, 100 indexes and 100 relationships.

This feature allows the ANLSCAN program to avoid additional access for the same object when multiple SQL statements, within the same analysis execution, refer to it.

10.3.4 QMF Interface response time

The QMF Intercept program is a condensed version of ANLSCAN and ANLCOST. For the QMF interface these two programs have been stripped of any non-essential parts, given that for this interface the program deals with one single SQL statement at the time and has minimal reporting capability.

The costing component of the program requires a PL/1 environment for the analysis and the QMF Intercept program has been designed to set this environment up only once at the first invocation and to dismantle it at the end of the QMF session. In this way only the first SQL analysis pay the additional cost of environment set up.

Another expedient deployed to minimize potential performance impact is the usage of program's embedded parameters lists instead of using external files. In this way then program can execute without performing any external I/O.

Input and output flow

SQL statements

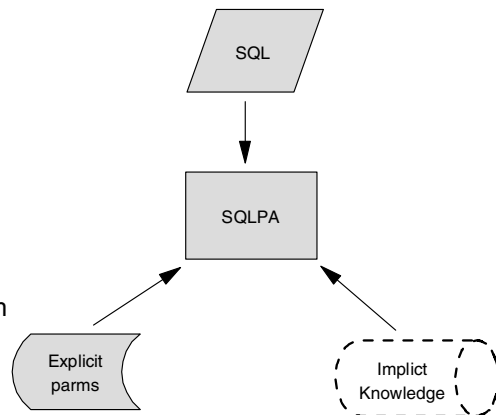
- sequential data set
- PDS member
- DBRM
- QMF query

Parameters

- ANLCNTL: target host system
- ANLPARM: user specific

Implicit knowledge

- predictive modeling techniques
- costing formulas
- path length benchmarks



ibm.com/redbooks

© 2000 IBM Corporation

10.4 Input and output flow

This paragraph provides information about input and output flow of SQL/PA and details of their structure and contents.

10.4.1 Input SQL

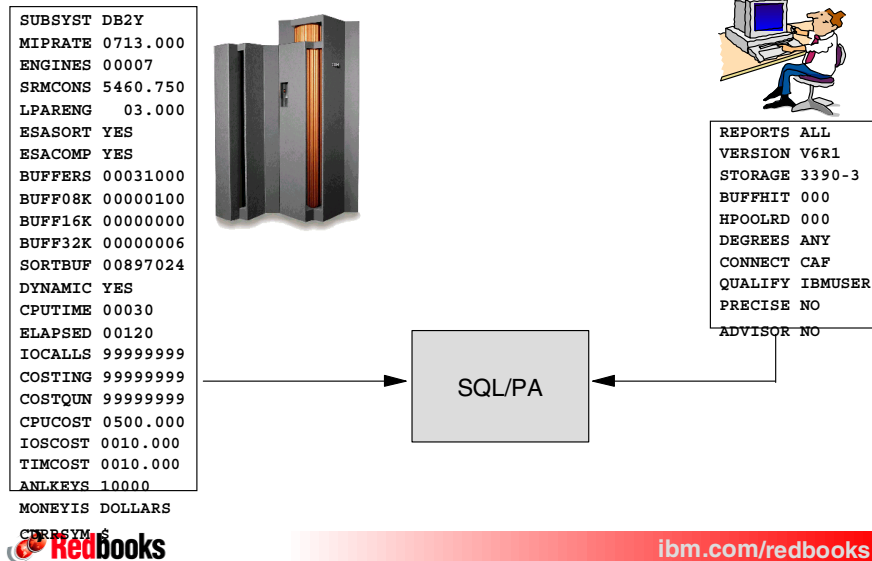
The Parser program ANLSCAN accepts DBRMs, sequential files, and PDS members as valid forms of input for SQL statements. The data set format must be Fixed Block with a record length of 80 and for Sequential files and PDS members the actual SQL statement must appear between column 1 and 72.

Note: ANLSCAN program doesn't support Variable Blocked format data sets as well as whole source code programs as input; source programs however can still be used with the SQL/PA ISPF Interface.

Definition of entire library as input is not supported, however you can concatenate multiple homogeneous data sets. This means that Sequential files and PDS data sets with same RECFM and LRECL can be concatenated but you cannot mix DBRM with them. The program doesn't know that the input is made up by multiple files, this can be specified only in the JCL, therefore the format is determined by the first file.

When using SQL statements in multiple concatenated Sequential or PDS data sets, we recommend the use of a semicolon at the end of each statement as this will allow the Parser program to identify the beginning and end of each SQL statement.

ANLCNTL and ANLPARM



10.4.2 ANLCNTL and ANLPARM parameters

Another very important input are the two sets of parameters identified as ANLCNTL and ANLPARM.

The first set describes the target host environment against which SQL/PA will perform its analysis; this can be the test environment or the production environment or even a *virtual* environment. For example you can analyze the performance of a given query in a more powerful computer or configuration without the actual hardware.

The second set of parameters is primarily user-oriented and describes features that are associated with the application and its usage.

When stored in a file all parameters have common characteristics:

- Name must begin in column 1
- Name length is exactly 7 characters
- Name must be followed by one blank
- Value begins in column 9
- Value maximum length is 11 alphanumeric and can be variable
- Comments can be added after column 20 and can be up to 60 characters in length

The following information intends to provide a summary of the usage and meaning of some of the SQL/PA parameters, for a complete list of the parameters and relevant descriptions, refer to Appendix C, "SQL Performance Analyzer" on page 227.

10.4.3 ANLCNTL parameters

These set of parameters define the target host system providing information such as MIPS rating of the target computer, number of engines, presence of hardware assisted sort and compression, number of buffer pools and many more. Usually these parameters are set up by the personnel installing SQL/PA and don't change very often.

For a full list and description of the target host system parameters, refer to Appendix C, "SQL Performance Analyzer" on page 227.

SQL/PA use the MIPRATE and ENGINES parameters together to calculate the processing capability of the computer, however also the SRMCONS parameter provides the same information and this is only considered when MIPRATE and ENGINES are set to 0.

Among these parameters you can find two particular categories:

- Parameters that establish the thresholds for specific resources; these are CPUTIME, ELAPSED, IOCALLS, COSTING, and COSTQUN.
Note: SQL/PA will flag any activity that exceeds the limits defined by these parameters. In the case of the QMF Intercept program these are the parameters controlling the prompt for query execution confirmation or optionally the automatic cancel.
- Parameters that are used to calculate the monetary value of a given SQL statement; these thresholds must be set according to your local charge back guideline. The parameters are CPUCOST, IOS COST and TIMCOST.

10.4.4 ANLPARM parameters

These parameters identify the individual application environment and they control characteristics such as type of attachment used, version of DB2, type of SQL/PA reports to be produced, type of disks, degree of parallelism and so forth. These parameters can, and often must, change for each individual execution.

For a full list and description of the user parameters, refer to Appendix C, "SQL Performance Analyzer" on page 227.

Parameter files allocation and storage

Interface	ANLCNTL	ANLPARM
Batch	external file default SANLPARM library	external file default SANLPARM library
ISPF	external file default SANLDATA library	set up in ISPF panel saved in ISPF table
QMF	embedded in ANLGOV1	embedded in ANLGOV1
Stored Procedure	external file allocated by DSNPAS	provided by calling program



ibm.com/redbooks

© 2000 IBM Corporation

10.4.5 Parameters files allocation and storage

It is important to note that each product's interface allocates the parameter files in a different manner in order to satisfy either the functionality of the interface or the appropriate performance requirements. The parameters are exactly the same as well as their value, the difference is in the way they are stored and allocated as detailed below:

- Library SANLPARM contains two members ANLCNTL and ANLPARM, these are the two sample parameter lists and are used for batch execution. ANLCNTL should be maintained centrally in this library, however copies of ANLPARM can be used from different libraries in order to accommodate the users requirements.
- The equivalent of ANLCNTL for the TSO/ISPF interface, it is stored in a special library SANLDATA that is allocated under the ISPMLIB DD together with the ISPF message libraries. The member names usually identify the system or LPAR name.

The ANLPARM values are instead stored in the ISPF user profile and can be configured dynamically through ISPF panels. This includes the selection of different target host system configuration (ANLCNTL file).

- With QMF Intercept both parameter lists are embedded in the program ANLGOV1 so that any changes require compile and link. This particular solution has been adopted for performance reasons so that the program doesn't have to perform additional IO to external files.

Note: the QMF Intercept program ANLGOV1 is statically linked in the QMF governor exit module therefore after reassembling ANLGOV1 you must always reassemble the modified DSQUEGV1 exit. For further information,

please refer to *DB2 SQL Performance Analyzer for OS/390 Installation Guide Version 1*, SC27-1003.

- For Stored Procedure execution the ANLCNTL target host system configuration must be allocated by the DB2 Stored Procedure Address Space under an ANLCNTL DD card. The address space has exclusive control of the allocated data set and therefore it is strongly recommended to use a parameter file separate from those used in batch and ISPF.

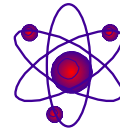
Note: STOP and START of DB2 is required to activate any changes to this file.

The user parameter ANLPARM are provided by the calling program and passed to the Stored Procedure ANLPROC as part of the standard parameter list of the CALL statement.

SQL Advisor

Expert system that provides:

- Alerts and Warnings
 - identify anomalies or weaknesses in the coding or design
 - always active, ADVISOR NO
- Notes and Recommendations
 - performance related advice to enhance the SQL statement
 - activated by ADVISOR YES
- Guidelines and Good News
 - assist novice users in proper coding and design
 - activated by ADVISOR ALL



ibm.com/redbooks

© 2000 IBM Corporation

10.4.6 Special parameters: SQL Advisor

Between both ANLCNTL and ANLPARM parameters there are special parameters that require some additional comments.

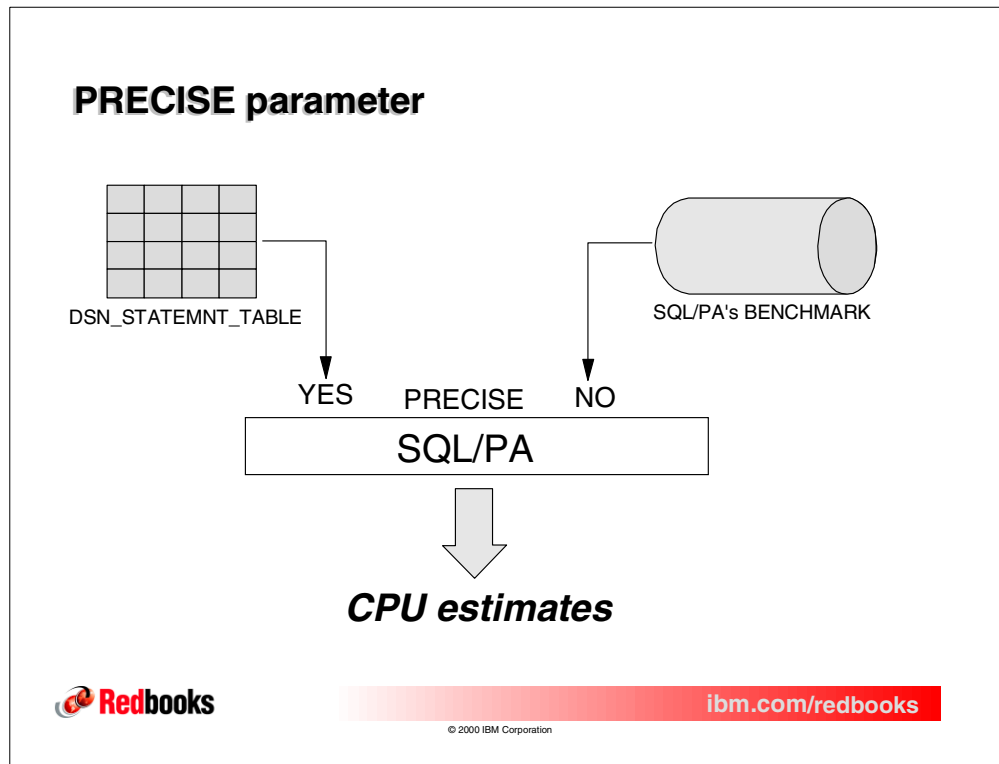
10.4.6.1 ADVISOR parameter

This is a user parameter that controls the level of assistance required from the SQL Advisor component.

The SQL Advisor is an expert system that assists you in identifying potential problems and advising about improvement to both the SQL statements and the underlying database design.

The messages provided by the SQL Advisor are displayed in the SQL/PA reports and their generation is controlled by the ADVISOR parameter in ANLPARM. The messages generated are grouped in the following levels:

- Alerts and Warnings are always active (ADVISOR NO) and provide assistance to identify potential problems and areas that will cause performance degradation
- Notes and Recommendations are activated by setting ADVISOR YES and suggest ways to enhance the SQL statement or the underlying database design, often highlighting how to take advantage of DB2 version related features or enhancements
- Guidelines and Good News are activated by setting ADVISOR ALL and provide general design and coding techniques advice as well as providing an always welcomed appreciation for any job well done. This level is particularly indicated for novice users.



10.4.6.2 PRECISE parameter

Still amongst the ANLPARM parameters we find the flag that allows SQL/PA to replace its own internal cost estimates with the CPU time estimate from the DB2 optimizer. The parameter is PRECISE and the default is NO, meaning SQL/PA internal cost estimate are enabled. YES or ALL enables the usage of the DB2 optimizer's CPU estimates.

Note: table DSN_STATEMNT_TABLE is required for PRECISE YES. This is special table introduced by DB2 for OS/390 Version 6 in addition to the PLAN_TABLE to provide processor milliseconds (PROCMS) and Service Units (PROCSU) estimates for the explained SQL statement.

In essence this parameter allows SQL/PA to use the estimated path lengths of the DB2 optimizer instead of those internally provided. The main differences between the two is that the DB2 optimizer's estimates are focused only data access and therefore are usually less extensive.

Beside the CPU estimates SQL/PA will also produce the Service Unit (QUNIT) from the same table instead of its own QUNITS.

It is important to remind the reader that estimates are not actual measurements, therefore even values from the DB2 optimizer can be different than those reported by an actual execution.

10.4.6.3 ANLKEYS parameter

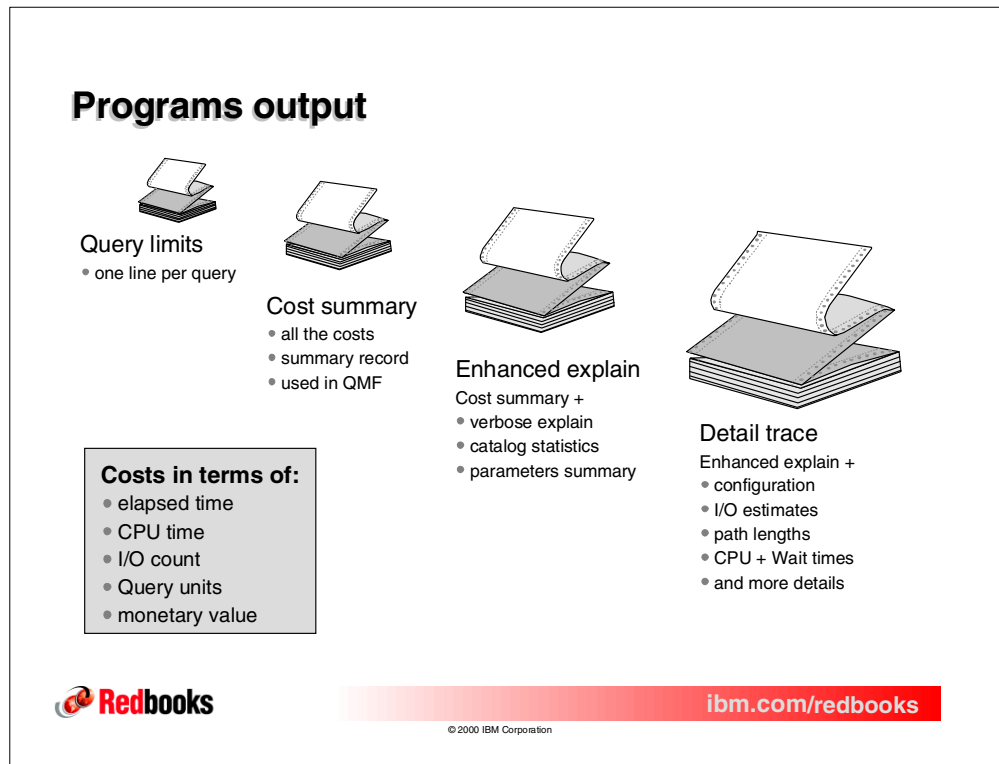
This parameter is part of the target host system set ANLCNTL and you can use it to modify the way SQL/PA performs the cost analysis.

It allows the elimination of certain overhead that you don't want to consider in your investigation.

It can assume only four values:

- 00000 is used for normal processing (or omit parameter entirely)
- 10000 specifies that only CLASS 1 times must be reported. This can be used for example to validate the CPU estimate against values produced by real execution
- 02000 eliminates all system overheads and its purpose is to feed base information to a capacity planning model. When using this parameter with this value it is advisable to specify CONNECT NONE in order to eliminate also the estimated connect overhead
- 12000 combines 10000 and 02000 so that only CLASS 1 times will be reported and any additional overhead will be eliminated.

Note: Do not use values different than those reported here, because they could cause unpredictable results.



10.5 Programs output

The main reporting capabilities of DB2 SQL Performance Analyzer are available through the TSO/ISPF interface and in batch. The program generates four types or levels of reports. These include:

- Query limits report is a quick eye-catcher report that shows costs and thresholds violations in one simple line for each SQL statement
- Cost summary report provides only the costs in terms of elapsed time, CPU time, I/O count, Query units and monetary value. In addition it presents warnings for any exceeded limit including the threshold value.
- Enhanced Explain report enriches the Cost summary report with Explain information provided in a sentence-like structure that include catalog statistics and the list of parameters used for the SQL/PA execution
- Detail trace complements the Enhanced Explain informations providing estimates for several components such as the total path length, number of getpages, synchronous and asynchronous I/O and much more.

10.5.1 Reports activation

The generation of the reports is controlled by the REPORTS parameter in the user parameter list (ANLPARM) and it can assume the following values:

- YES or ONE: These values generate always the Cost summary report and the QLIMIT report when requested.
- EXP or TWO: These values generate the Enhanced Explain report in addition to the previous ones.

- ALL or TRE: These values generate the Detail trace report in addition to the previous ones.

Furthermore the reports are always written to a file that is identified by a specific DD card for the batch execution while in ISPF must be defined by the user in the appropriate field of the ISPF set up panel.

Please note that the disposition of the report's files in batch is fully controlled by the user and the same is in ISPF, with the exception of the Cost summary report that in ISPF is always appended within the same user's ISPF session.

The Enhanced Explain report and the Detail trace report are incremental reports, meaning that both provide also the information of the previous report's level and therefore they can be quite extensive in size.

The following table summarize the reports activation requirements.

Report name	REPORTS parm	Batch DD card	ISPF default file name
QLIMIT	YES or ONE	QLIMIT	userid.ANLI.QLIMIT
Cost Summary	YES or ONE	SYSPRINT	userid.ANLCOST.LOG
Enhanced Explain	EXP or TWO	ANLREP	userid.ANLI.EXPLAN
Detail trace	ALL or TRE	QTRACE	userid.ANLI.DETAIL

Note: The QLIMIT report under ISPF and the Cost Summary report, both in batch and in ISPF, are mandatory and automatically produced. During the same ISPF session of SQL/PA the *userid.ANLCOST.LOG* file is allocated with disposition MOD, resulting in each analysis execution to be appended to the same file, this allows comparison of analysis during tuning exercises.

All other reports are optional.

QLIMIT report

```
BROWSE -- IBMUSER.ANLI.QLIMIT ----- Line 00000000 Col 001 080
Command ==>                               Scroll ==> PAGE
***** Top of Data *****
CEIQ$ ERROR QUERYNO    CPU TIME    ELAPSED    PHYS I/O    QUNITS MONETARY COST
NNNNN    0 100000001      223.502      491.970       3747      12205        69.88
***** Bottom of Data *****
```



ibm.com/redbooks

© 2000 IBM Corporation

10.5.2 QLIMIT report

The output of the Query Limit report consists of one record for each SQL statement analyzed that shows the following fields:

- CLIM, indicates if CPU time limit has been exceeded; values are Y or N
- ELIM, indicates if Elapsed time limit has been exceeded; values are Y or N
- ILIM, indicates if I/O count limit has been exceeded; values are Y or N
- QLIM, indicates if Query units limit has been exceeded; values are Y or N
- MLIM (\$LIM), indicates if Monetary value limit has been exceeded; values are Y or N
- RETCODE (ERROR), is the SQL/PA return code from query analysis processing; it is a four digit value that corresponds to the four digit in the ANL messages
- QUERYNO, is the query number assigned by SQL/PA for the Explain process
- CPUTIME, is the CPU time estimate for this query
- ELAPSED, is the ELAPSED time estimate for query
- IOCUNT (PHYS I/O), is the I/O count estimate for this query
- QUNITS, is the Query units estimate for this query
- COSTS (MONETARY COSTS), is the Monetary value estimate for this query

Cost summary report (Parser program)

```
BROWSE -- PAOLOR4.ANLCOST.LOG ----- Line 00000000 Col 001 080
Command ==>                               Scroll ==> PAGE
***** Top of Data *****
*** SQL PARSE, EXPLAIN & CATALOG FACILITY ***
ANLSCAN PROGRAM BEGINS AT: 15:00:58.921 ON 11-22-2000 VERSION 1.1.0

ANL0000I ANLUSER2.PLAN_TABLE WILL BE USED TO HOUSE THE EXPLAIN OUTPUT.

ANLSCAN ** STATISTICS ** TOTAL STATEMENTS EXAMINED = 1
** FOR THIS ** GOOD STATEMENTS PREPARED = 1
** PARSE RUN ** BAD STATEMENTS W/ERRORS = 0
ANLSCAN PARSE COMPLETE AT: 15:00:59.796 ON 11-22-2000 VERSION 1.1.0

ANL2000I TOTAL NUMBER OF PLAN RECORDS PROCESSED WAS 2
TOTAL NUMBER OF PARSED RECORDS SCANNED WAS 1
TOTAL NUMBER OF ERRORS ENCOUNTERED IN RUN 0
SQL/PA CATALOG ACCESS: TABLES = 1 TABSPACES = 1
INDEXES = 0 RELATIONS = 0

ANLSCAN PROGRAM ENDS AT: 15:01:00.212 ON 11-22-2000 VERSION 1.1.0
```



ibm.com/redbooks

© 2000 IBM Corporation

10.5.3 Cost summary report

The Cost summary report is divided into two parts, the first one shows the statistics from the execution of the parsing process and it is produced by the Parser program ANLSCAN.

This part of the report highlights information such as:

- Number of statements examined
- Number of statements that were not parsed (Bad)
- Number of PLAN_TABLE records processed
- Number of processing errors
- Number of table spaces, tables, indexes and relations for which the DB2 catalog statistics were retrieved

```

*** SQL/PA -- SQL PERFORMANCE ANALYZER ***
ANLCOST PROGRAM BEGINS AT: 15:01:00.778 ON 11-22-2000 VERSION 1.1.0

EXPLAIN PLAN SET QUERYNO = 100000001 FOR
SELECT L_RETURNFLAG, L_LINESTATUS, SUM(L_QUANTITY) AS SUM_QTY,
       SUM(L_EXTENDEDPRICE) AS SUM_BASE_PRICE,
       SUM(L_EXTENDEDPRICE*(1-L_DISCOUNT)) AS SUM_DISC_PRICE,
       SUM(L_EXTENDEDPRICE*(1-L_DISCOUNT)*(1+L_TAX)) AS SUM_CHARGE,
       AVG(L_QUANTITY) AS AVG_QTY, AVG(L_EXTENDEDPRICE) AS AVG_PRICE,
       AVG(L_DISCOUNT) AS AVG_DISC, COUNT(*) AS COUNT_ORDER
FROM LINEITEM
WHERE L_SHIPDATE <= '1998-09-02'
GROUP BY L_RETURNFLAG, L_LINESTATUS
ORDER BY L_RETURNFLAG, L_LINESTATUS ;

*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
* QUERY 100000001 WILL REQUIRE 491.96209 SECONDS OF ELAPSED TIME *
* DURING WHICH 223.49649 SECONDS OF CPU TIME WILL BE CONSUMED AND *
* A TOTAL OF 3747 PHYSICAL I/O REQUESTS WILL BE ISSUED TO DISK *
* QUNITS 12205 ESTIMATED PROCESSING COST $ 69.88 DOLLARS *
*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*

ANLCOST PROGRAM ENDS AT: 15:01:01.228 ON 11-22-2000 VERSION 1.1.0

(C) COPYRIGHT 1993-2000 BY IMSI *
* ALL RIGHTS RESERVED WORLDWIDE *

***** Bottom of Data *****

```



© 2000 IBM Corporation

Elapsed time is commonly used for batch executions, while for online system this is often referred to as response time. In SQL/PA Elapsed time is used to indicate the forecasted end-to-end total time, this includes:

- If any limit is exceeded a warning message is also displayed in this report, showing the estimated value and the relevant threshold value with a description.

Enhanced Explain report (partial report)

```

A TABLESPACE SCAN IS REQUESTED ON A PARTITIONED TSPACE OF 3 PARTS.
IN TABLE PAOLORA4 .LINEITEM                                33832  4K PAGES WILL BE READ.
TABLE HOLDS                                1951548 ROWS OF 135 BYTES, WITH 16 COLS.
TYPE = T, LOCK SIZE = A, CLOSE = N, TS LOCK MODE = IS , LOCK PART = .
PCT WITH ROWS = 99% , ENCODE = E, MAXROW = 255, PCT COMPRESSED = 99%.

```

ANL5012W *** ALERT:

This partitioned tablespace is not being accessed using any form of parallelism. Possible reasons include: insufficient parallel buffers defined in pool, or Degree Any not specified on the bind. Elapsed time would be considerably enhanced by choosing a parallel processing option on this partitioned tablespace.

A SORT WILL BE PERFORMED ON THE RESULTS.

```

*-----*
* QUERY 100000001 WILL REQUIRE 628.04552 SECONDS OF ELAPSED TIME *
* DURING WHICH 223.50237 SECONDS OF CPU TIME WILL BE CONSUMED AND *
* A TOTAL OF 3747 PHYSICAL I/O REQUESTS WILL BE ISSUED TO DISK *
* QUNITS 12205 ESTIMATED PROCESSING COST $ 70.26 DOLLARS *
*-----*

```

ANL5027W *** WARNING:

ESTIMATE OF 70.26 EXCEEDS "MONETARY" LIMIT OF 50.00 DOLLARS !

ANL5029W *** WARNING:

ESTIMATE OF 628.046 EXCEEDS "ELAPSED TIME" LIMIT OF 600 SECONDS!



ibm.com/redbooks

© 2000 IBM Corporation

10.5.4 Enhanced Explain report

The report displays the list of the parameters used for the analysis (not shown in the slide) followed by the catalog statistics and the Explain information. Any Alerts and Warnings messages (if any) from the SQL Advisor are also displayed in this report.

The sentence-like format of the Explain caters to the following type of access:

- Table space scan
- Cluster matching index scan
- Cluster non-matching index scan
- Random matching index scan
- Random non-matching index scan
- Multiple index scan
- Joining tables
- Subquery processing
- Sorting results
- Insert, Update and Delete

For more details about the access paths reports please refer to *DB2 SQL Performance Analyzer for OS/390 User's Guide Version 1*, SC27-1002.

Detail trace report (configuration details)

ANL3098I *** CONFIGURATION NOTE:

The overall rating for this system is 203.714 Million Instructions/Second
Each individual engine will operate at 101.857 Million Instructions/Second

ANL3036I: The assumed Overheads for Connection Type of CAF have been
estimated in Millions of Instructions for this BATCH Statement:
Thread Mgmt = 0.096400 Attach = 0.300000 Appl = 0.150000

ANL6042I *** NOTE:

Dynamic statement caching is assumed for this set of estimates.

ANL3037I: Estimated 3390-3 DASD Service Times are:

Sync Read 4K Page = 0.0231	Prefetch block of 32 4K Pages = 0.0526
Sync Read 8K Page = 0.0240	Prefetch block of 4 8K Pages = 0.0297
Sync Read 16K Page = 0.0259	Prefetch block of 0 16K Pages = 0.0221
Sync Read 32K Page = 0.0297	Prefetch block of 1 32K Pages = 0.0297
Async Wrt 4K Pages = 0.0526	Async Write of 16K Page Block = 0.0221
Async Wrt 8K Pages = 0.0297	Async Write of 32K Page Block = 0.0297



ibm.com/redbooks

© 2000 IBM Corporation

10.5.5 Detail trace report

In addition to the Cost summary and the Enhanced Explain information, the Details trace report shows the estimates for several aspects of the processing. Its main purpose is to illustrate the components and the steps involved in the estimation process and it is not used under normal circumstances.

This reports has two parts, the first one provides details about the estimated base values of the overall configuration, the second one displays the estimates for the steps involved in the resolution of the data access.

Based upon the target host system configuration and the user parameters, particularly the disk configuration and the type of attachment, SQL/PA displays a series of Configuration Notes in this first part.

The slide highlights the calculated speed per engine based on the total MIPS rating for the processor and the connection overhead base value whom estimates are based on the connection type identified. In addition we can see the forecasted synchronous and asynchronous I/O speed per unit of I/O based on the configured type of disk.

Detail Trace report (processing details)

```

ROWS PROCESSED =      644011  PERCENT TABLE PROCESSED =    0.330000000
COLS PROCESSED =         10  BOOLEAN FILTER FACTORED =    0.330000000
DATA PAGES READ =      33832  INDEX LEAF PAGES READ =          0

SYNC READ I/OS =         2  TABLE =         2  INDEX =         0
PREFETCH I/OS =       1058  TABLE =       1058  INDEX =         0
ASYNC WRT I/OS =         0  TABLE =         0  INDEX =         0
GET PAGE CALLS =      33869  SYSIO =        11  LOGIO =         0
PROCESSES =20042.411100 CLASS 1 =21644.090866 OTHER O/H = 30.258800
SYNC READ =   0.014700 PREFETCH = 30.258800 ASYNC WRT = 0.000000
DECOMPRESS= 173.881791 COMPRESS = 0.000000 HIPERPOOL = 0.000000
GET PAGES =   82.979050 SYSTEMS = 0.037800 LOG WRITE = 0.000000
FETCH ROW =  869.414850 LOCK/ETC = 475.351575 PREDICATE =      1

QUERYNO: 100000001  QBLOCKNO:      1  PLANNO:      1  MIXSEQ:      0
SUMMARY ->
THE TOTAL CUMULATIVE PATH LENGTH FOR THIS QUERY IS 21674.896066M INS.
RESULTING IN A TOTAL CPU TIME OF 212.79702 SECONDS CONSUMED OVERALL.
DB2 WILL PUT CLASS 1 CPU TIME OF 212.49458 SECONDS IN SMF 101 RECORD.
DB2 SHOWS ADDITIONAL CPU TIME OF 0.30244 SECONDS IN SMF 100 RECORD.
ESTIMATED TOTAL LOGICAL I/O CALLS = 1060 (EXCLUDING SYSTEM) AND
ESTIMATED TOTAL PHYSICAL I/O CALLS = 1060 WITH HIT RATIO = 1.000.
WAIT TIME FOR SYNC READ I/O = 0.04610 PREFETCH I/O = 55.62561
WAIT TIME ON ASYNC WRITE I/O = 0.00000 TOTAL IWAIT = 55.67171
WAIT TIME FOR VSAM OPEN/CLOSE MACROS, BINDING AND LOCKING = 2.60791
*****

```



ibm.com/redbooks

© 2000 IBM Corporation

In the second part of the report SQL/PA there are four groups of information, normally preceded by the Enhanced Explain report; the four groups are:

- **I/O estimates**

These values are calculated using the Explain and catalog information and they show the number of rows, columns and pages (data and index) involved in the process as well as the amount of synchronous and asynchronous I/O to be expected.

Note: the number of rows processed is a very important information for a proper estimation, for this reason the tool allows the user to establish a greater degree of precision through the use of OPTIMIZE FOR n ROWS. SQL/PA detects this clause and uses it as the actual number of rows to be processed. For example, if you expect 543,700 records to be returned then you should let SQL/PA know using OPTIMIZE FOR 543700 ROWS.

- **Path Length estimates**

The path length equates to the number of instructions needed to perform a given process; this part of the report shows the estimated path length for each component involved in the resolution of the SQL statement.

All numbers are given in millions of instructions and the CLASS 1 and OTHER O/H (short for OverHead) values are totals of the other fields in the same section.

CLASS 1 summarizes the fields normally compounding the CLASS 1 time, such as PROCESSES, GETPAGES, DECOMPR, LOCK/ETC.

OTHER O/H instead provides the summary estimate of the processes charged to PREFETCH, ASYNC WRT, LOG WRITE, COMPRESS.

PROCESSES provides the estimate for the primary path length; as an example this includes Stage 1 and Stage 2 predicate resolution, index and data page scans, built-in function resolution and any calculation.

- **Summary CPU estimates**

TOTAL CUMULATIVE PATH LENGTH value is the estimated path length for each *incremental* step involved in the SQL process.

Note: the CUMULATIVE attribute of the field's title refers to the path length components displayed in the previous part of the report, and not to the incremental steps of and SQL plan.

The incremental SQL Plan steps associated with this estimate can be:

1. The main or driving SQL
2. Multiple index operational sequence (MIXOPSEQ) steps
3. Subquery (SBQRY)
4. Sort operation

These operations are presented as separate path length totals to provide greater granularity of the incremental costs of the whole query.

Usually this value is larger than the sum of CLASS 1 and OTHER O/H and this because the total includes the overhead associated with the attach facility connection, thread management, and the estimates of the application logic associated with the attach method. These are one-time charges applied to the base total path length.

The resulting TOTAL CPU TIME is provided in seconds and it is computed simply dividing the TOTAL PATH LENGTH by the MIPS (Millions Instructions Per Second) as determined by the target host system parameters.

Note: In spite of the total path length being individual for each step (MIXOPSEQ, SBQRY or Sort), the resulting CPU time is always cumulative.

- **Wait time**

The last group of information provides estimates for the wait time. When totalized and compounded with the total CPU, these values forecast the expected Elapsed time.

The wait times reported here are greatly influenced by a series of user parameters such as BUFFHIT, DEGREES, STORAGE and VERSION as well as by some of the target host system configuration parameters.

Detail trace report (additional sort step)

```
A SORT HAS BEEN REQUESTED: SORT COST = 1059.611400 I/O =      2687.
ROWS SORTED =      644011 COLUMNS =      3 SORT WORK PAGE =    10745.

QUERYNO: 100000001 QBLOCKNO:      1 PLANNO:      2 MIXSEQ:      0
SUMMARY ->
THE TOTAL CUMULATIVE PATH LENGTH FOR THIS QUERY IS    1090.416600M INS.
RESULTING IN A TOTAL CPU TIME OF    223.50237 SECONDS CONSUMED OVERALL.
DB2 WILL PUT CLASS 1 CPU TIME OF    222.89750 SECONDS IN SMF 101 RECORD.
DB2 SHOWS ADDITIONAL CPU TIME OF      0.60487 SECONDS IN SMF 100 RECORD.
ESTIMATED TOTAL LOGICAL I/O CALLS =      3747 (EXCLUDING SYSTEM) AND
ESTIMATED TOTAL PHYSICAL I/O CALLS =      3747 WITH HIT RATIO = 1.000.
WAIT TIME FOR SYNC READ I/O =      0.04610 PREFETCH I/O =    189.09212
WAIT TIME ON ASYNC WRITE I/O =      0.00000 TOTAL IWAIT =    189.13823
WAIT TIME FOR VSAM OPEN/CLOSE MACROS, BINDING AND LOCKING =    2.60791
*****

* QUERY 100000001 WILL REQUIRE    628.04552 SECONDS OF ELAPSED TIME *
* DURING WHICH    223.50237 SECONDS OF CPU TIME WILL BE CONSUMED AND *
* A TOTAL OF      3747 PHYSICAL I/O REQUESTS WILL BE ISSUED TO DISK *
* QUNITS    12205 ESTIMATED PROCESSING COST $      70.26 DOLLARS *
*****
```



ibm.com/redbooks

© 2000 IBM Corporation

To illustrate the point of the SQL plan's incremental steps, here we show the report portion associated with the additional sort step required by this particular SQL.

The TOTAL CUMULATIVE PATH LENGTH is the total path length for the sort step only, while the resulting TOTAL CPU TIME, as well as the other values in the same part of the report, are cumulative of the previous step.

Eventually, the Cost Summary is also displayed and this refers to the SQL as a whole, that is inclusive of all incremental steps.

QMF and Stored Procedure output

```
ANL5025W *** WARNING:
ESTIMATE OF      258.163 EXCEEDS "CPU TIME" LIMIT OF    30 CPU SECONDS!

*-----*
* QMF ONLINE QUERY WILL REQUIRE    614.92575 SECONDS OF ELAPSED TIME *
* DURING WHICH    258.16280 SECONDS OF CPU TIME WILL BE CONSUMED AND *
* A TOTAL OF      1947 PHYSICAL I/O REQUESTS WILL BE ISSUED TO DISK *
* QUNITS    14098 ESTIMATED PROCESSING COST $      57.03 DOLLARS *
*-----*

*** WOULD YOU LIKE SQL/PA TO CANCEL THIS QUERY (Y/N)?
```

```
* CAF OPEN RETCODE IS          0

* EXPLAIN PLAN FOR (LENGTH 160)
* SELECT NAME, CREATOR, TSNAME, COLCOUNT, CARD, NPAGES
* FROM SYSIBM.SYSTABLES WHERE TSNAME = 'ANLSPACE'

* ANLPROC SQLCODE IS          0
* ANLPROC RETURNS ==> WARNING FLAGS: NNNNN
  ELAPSED:    1.83993 CPU TIME:    0.06617
  I/O COUNT:    18 QUNITS:        4
  MONETARY:    0.01
  ANL CODE:    0 SQL CODE:        0

* CAF CLOSE RETCODE IS          0
* PROGRAM TERMINATION
```



ibm.com/redbooks

© 2000 IBM Corporation

10.5.6 QMF and Stored Procedure output

As mentioned previously the QMF Intercept program and SQL/PA Stored Procedure are not designed to produce extensive reporting. The output provided by these two program is essentially a cost summary.

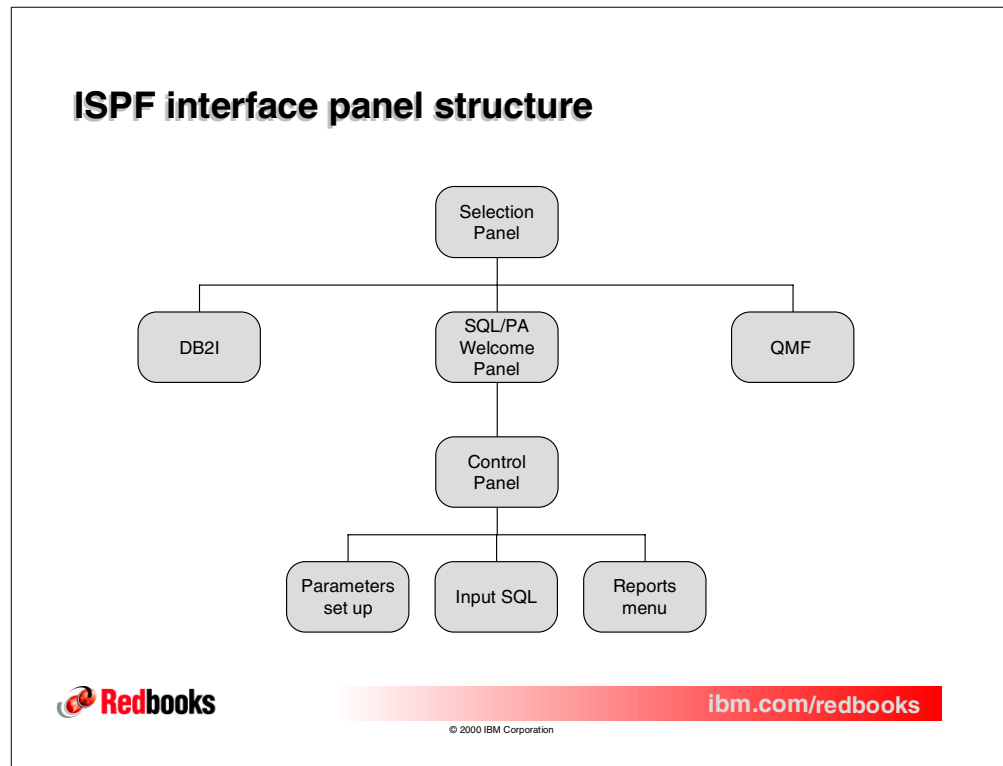
The main differences are in the format and in some of the contents:

- The QMF Intercept program uses the same format of the Cost Summary report, and in addition it has the capability to prompt for confirmation when any of the defined thresholds are exceeded.

Note: Optionally you can configure the program to automatically cancel without asking confirmation.

- The stored procedure offers the same cost summary information assembled in a different way. The sample program ANLSTP displays the SQL statement analyzed and the return code from the SQL/PA analysis, the DB2 SQL parsing and the Call Attachment Facility OPEN and CLOSE instructions.

The SQL/PA stored procedure ANLPROC returns all its values as variables and the ANLSTP sample program formats and displays them as reported in this foil.



10.6 Working with the ISPF interface

The ISPF interface of SQL Performance Analyzer is an interactive instrument oriented towards the development and maintenance of applications or SQL query.

The standard set up of the appropriate ISPF environment is prerequisite to use this interface. Please refer to *DB2 SQL Performance Analyzer for OS/390 User's Guide*, SC27-1003 for further information regarding installation of the ISPF interface.

10.6.1 Starting a session

The first panel displayed is a selection of items:

- DB2 Interactive (DB2I)
- QMF with the integrated functions of SQL/PA QMF Intercept program
- SQL/PA ISPF interface

Note: you can integrate SQL/PA in your existing menu and avoid the display of this first panel. Please make sure you carefully read and understand the recommendations provided in the installation manual.

The SQL/PA welcome panel is displayed when selecting option S.

Control panel

```

----- SQL/PA CONTROL PANEL -----
COMMAND ==>

INPUT DATA SET NAME ==> 'DB2PA.SANLSQL(COBOL) '
EDIT INPUT DATA SET ==> YES          (YES or NO)

OUTPUT EXPLAIN FILE  ==> ANLI.EXPLAN
OUTPUT QLIMITS FILE  ==> ANLI.QLIMIT
OUTPUT DISPOSITIONS ==> OVERLAY      (Overlay or Append)
RESET SQL/PA PARMS   ==> YES          (YES or NO)
GO TO REVIEW REPORTS ==> NO          (YES or NO)

>>> Press PF3 or PF12 to Exit SQL/PA, or Press Enter to Continue <<<

----- Current SQL/PA Parameter Settings -----

Report Level = ALL      DASD Storage = 3390-3    DB2 Version = V6R1
Buffer Hits   = 100     Attach Methd = CAF       DB2 Sys ID  = DB2Y
Default Name  = PAOLOR4 Target Host  = SC63       Trace Stmt = YES
Hiperpool Rd  = 0       Parallel I/O = ANY       Precise Est = NO
IO Seek Time  = 0.0000  Rotate Delay = 0.0000    Trnsfr Rate = 0000.0
Data Shr Pct  = 0       Data Shr Mem = 0         SQL Advisor = NO

SQL/PA Cost Summary File ==> PAOLOR4.ANLCOST.LOG (always in APPEND mode)

```



ibm.com/redbooks

© 2000 IBM Corporation

10.6.2 Control panel

From this panel you can control the operation of the ISPF interface and it presents two groups of information:

- Input, output and navigation controls
- Current user parameters settings (ANLPARM); these are in display mode only, any changes must be performed through the appropriate panel.

From here you can select the input file or member that contains the SQL to analyze. You can decide to either edit it or go straight for the analysis execution. Furthermore you can define the name for the output data set that will hold the Enhanced Explain report and the QLIMIT report.

Note: the input data set or member must already exist, if not an error panel will be displayed.

In terms of navigation you can change the user parameter settings before the analysis or you can review existing report data set.

While the other reports are optional, depending to the selection of the Report Level in the user parameters, the Cost Summary report is always generated and the panel display at the bottom the default name.

Note: this file is always in append mode for the duration of the session; the file will be overlaid at the first execution after exiting and re-entering the ISPF interface.

User parameters in ISPF

```

----- SQL/PA PARAMETERS PANEL -----
COMMAND  ==>

DB2 SUBSYSTEM ID  ==> DB2Y          (DSN or other DB2 system name)
OUT REPORT LEVEL  ==> ALL           (Choose Level: YES, EXP or ALL)
SQL ADVISOR LEVEL ==> NO            (Choose Level: YES, NO, or ALL)
DEFAULT QUALIFIER ==> PAOLOR4       (Owner or High Level Qualifier)
ATTACH OVERHEAD   ==> CAF           (Attach via DSN, WFI, CICS, etc.)
BUFFER HIT RATIO  ==> 100           (Pages Found in Buffer Pool, 0-100)
HIPER POOL READS  ==> 0            (Pages Read from Hiperpool, 0-100)
PARALLEL DEGREES  ==> ANY           (Parallel I/O Degrees = ANY or ONE)
EXTRA PRECISION   ==> NO           (More Precise Answer: YES, NO, ALL)
DB2 DASD STORAGE ==> 3390-3       (DASD pool: 3390-2, etc. or NEWDSK)
NEWSTOR SEEK TIME ==> 0.0000        (if NEWDSK, Avg Seek Time in Msec)
NEWSTOR ROTATION  ==> 0.0000        (if NEWDSK, Rotation Delay in Msec)
NEWSTOR TRAN RATE ==> 0000.0        (if NEWDSK, Transfer Rate in KB/Sec)
DB2 TARGET HOST   ==> SC63         (Configuration Describing the Host)
DATA SHARING PCT  ==> 0            (Percent DB2 Work uses Data Sharing)
DATA SHARING MBRS ==> 0            (Number Data Sharing Group Members)
DB2 VERSION NAME  ==> V6R1         (DB2 Version: V6R1 or V3R1 -> V7R1)
DETAIL TRACE FILE ==> ANLI.DETAIL

>>> Press PF3 to Return to Main Menu, or PF12 to Exit SQL/PA <<<

```



ibm.com/redbooks

© 2000 IBM Corporation

10.6.3 User parameter in ISPF

This panel is displayed when selecting the RESET SQL/PA PARMS option.

We want to draw your attention on the Report Level parameter: when set to ALL this will trigger the display of DETAIL TRACE FILE parameter through which you can define the relevant file name.

Note: If you have REPORT LEVEL set to ALL or TRE, even if in the previous Control Panel you specify RESET SQL/PA PARMS NO, this User Parameter panel will be displayed anyway to allow you to change the name of the DETAIL TRACE FILE.

Please also take note of DB2 TARGET HOST parameter that specifies the name of target host system configuration (ANLCNTL); the ANLCNTL configuration must be in a member of the SANLDATA library. This is the place where the tool will search.

It is a good practice to name the member with a meaningful name such as the host name, machine type or the LPAR id.

The NEWSTOR parameters are only considered if DB2 DASD STORAGE is set to NEWDSK, otherwise SQL/PA will use the predefined values for the DASD model selected.

For a full list and description of the user parameters please refer to Appendix C, "SQL Performance Analyzer" on page 227.

Input SQL

```
EDIT ---- DB2PA.SANLSQL(COBOL) - 01.00 ----- Columns 001 072
Command ==> anl                               Scroll ==> CSR
028200 *****
028300 * SQL CURSORS *
028400 *****
028500 *** CURSOR LISTS ALL EMPLOYEE NAMES
028600
028700 EXEC SQL DECLARE TELE1 CURSOR FOR
028800 SELECT *
028900 FROM DSN8610.VPHONE
029000 END-EXEC.
029100
029200 *** CURSOR LISTS ALL EMPLOYEE NAMES WITH A PATTERN (%) OR (_)
029300 *** FOR LAST NAME
029400
029500 EXEC SQL DECLARE TELE2 CURSOR FOR
pp9600 SELECT *
029700 FROM DSN8610.VPHONE
029800 WHERE LASTNAME LIKE :LNAME-WORK
pp9900 AND FIRSTNAME LIKE :FNAME-WORK
030000 END-EXEC.
030100
030200 *** CURSOR LISTS ALL EMPLOYEES WITH A SPECIFIC
030300 *** LAST NAME
```



ibm.com/redbooks

© 2000 IBM Corporation

10.6.4 Input SQL

The Parser program ANLSCAN is not designed to scan program source code. The diverse convention and standards implemented by the various languages makes the parsing task complex and prone to error, such as the use of semicolons in PL/1. For this reason SQL/PA provides special editing capability through its ISPF interface.

The SQL/PA's ISPF interface allows you to edit a program's source code and to identify the SQL statement to analyze using the character **p** in the line number column of the edit session. The analysis is then activated by the execution of the ISPF Edit macro **anl**.

Usual ISPF conventions are implemented by SQL/PA so that multi-lines statements can be selected either with a pair of **pp-pp** markers or indicating the number of lines like **p6**. For further information and details about SQL/PA's ISPF Interface please refer to *DB2 SQL Performance Analyzer for OS/390 User's Guide*, SC27-1002.

You can edit source code programs as well as sequential and partitioned data set members containing just SQL statement such as a QMF export or import query library

Note on F keys: F3, F15, and END indicate the end of the current step. F12 and F24 exit all panels. Once you are in editing, the input member of the PF3 key will trigger the analysis execution instead of ending the edit. At the beginning of the edited file SQL/PA displays a series of temporary notes providing further details about line selection and F keys usage.

Report menu

```

----- SQL/PA REPORTS MENU -----
OPTION  ==>

Choose a Report to Browse:

1      SQL/PA  COST  Report -   A Costing Summary for each SQL statement
2      SQL/PA  EXPLAIN Report - Enhanced Explains for each SQL statement
3      SQL/PA  TRACE  Report - Detailed Trace Analysis of SQL statement
4      SQL/PA  LIMITS Report - One line per SQL cost vs. SQL/PA run limits

Other options:

T      TUTORIAL
X      EXIT

      To return to main SQL/PA processing panel, Press END key

```



ibm.com/redbooks

© 2000 IBM Corporation

10.6.5 Report menu

For the brief duration of the analysis SQL/PA displays the following message:

```

* * * * *
*
*   THE SQL PERFORMANCE ANALYZER IS NOW CONTEMPLATING THE
*   COST OF YOUR SQL STATEMENTS... THIS SHOULD JUST TAKE
*   A FEW MOMENTS...
*
* * * * *

```

At the end of the process you can press enter and the Report menu is displayed. All four reports can be displayed from this panel depending of the selection performed via the Report level parameter.

For details about the report's contents, refer to 10.5, "Programs output" on page 182, in this redbook, or refer to *DB2 SQL Performance Analyzer for OS/390 User's Guide Version 1*, SC27-1002.

Batch interface: ANLSCAN step

```
//*
//ANLSTEP1 EXEC PGM=ANLSCAN,COND=(4,LT)
//STEPLIB DD DSN=DB2PA.SANLLOAD,DISP=SHR
//          DD DSN=DSN610.SDSNLOAD,DISP=SHR
//          DD DSN=CEE.SCEERUN,DISP=SHR
//SYSPRINT DD SYSOUT=*
//ANLPARM DD DSN=DB2PA.SANLPARM(ANLPARM),DISP=SHR
//ANLCNTL DD DSN=DB2PA.SANLPARM(ANLCNTL),DISP=SHR
//ANLIN DD DSN=DB2PA.SANLSQL(Q01),DISP=SHR
//ANLOUT DD DSN=&&ANLOUT,DISP=(NEW,PASS),UNIT=SYSDA,
//          SPACE=(TRK,(5,1),RLSE),DCB=(LRECL=80,RECFM=FB,BLKSIZE=9040)
//ANLPAS DD DSN=&&ANLPAS,DISP=(NEW,PASS),UNIT=SYSDA,
//          SPACE=(TRK,(5,1),RLSE),DCB=(LRECL=140,RECFM=FB,BLKSIZE=11200)
//ANLSEP DD DSN=&&ANLSEP,DISP=(NEW,PASS),UNIT=SYSDA,
//          SPACE=(TRK,(1,1),RLSE),DCB=(LRECL=257,RECFM=FB,BLKSIZE=11308)
//ANLREL DD DSN=&&ANLREL,DISP=(NEW,PASS),UNIT=SYSDA,
//          SPACE=(TRK,(1,1),RLSE),DCB=(LRECL=83,RECFM=FB,BLKSIZE=11869)
//SYSPRINT DD SYSOUT=*,DCB=LRECL=133
//*
```



ibm.com/redbooks

© 2000 IBM Corporation

10.7 The Batch Interface

Batch execution of SQL/PA consists of three steps:

- Delete existing report data sets (optional)
- Execute the Parser program ANLSCAN
- Execute the Costing program ANLCOST

Besides the SQL/PA load library, for both ANLSCAN and ANLCOST steps you must allocate to STEPLIB or JOBLIB the DB2 load library and the Language Environment runtime library, unless they are defined in the link list.

Both steps also require the allocation of the parameters files ANLPARM and ANLCNTL to the appropriate DD card.

10.7.1 The ANLSCAN step

The Parser program execution step takes the input from the ANLIN DD card, this can be a sequential files and member of a partitioned data set or a DBRM. It produces staging information into four temporary files that are passed to the resource estimation program in the following step.

The SYSPRINT DD is required to print the Parser program statistics.

Note: Do not modify the DCB parameters for the temporary files, because this could lead to unpredictable results.

ANLCOST step

```
//*
//ANLSTEP2 EXEC PGM=ANLCOST
//STEPLIB DD DSN=DB2PA.SANLLOAD,DISP=SHR
//          DD DSN=DSN610.SDSNLOAD,DISP=SHR
//          DD DSN=CEE.SCEERUN,DISP=SHR
//ANLPARM DD DSN=DB2PA.SANLPARM(ANLPARM),DISP=SHR
//ANLCNTL DD DSN=DB2PA.SANLPARM(ANLCNTL),DISP=SHR
//ANLPAS DD DSN=*.ANLSTEP1.ANLPAS,DISP=(OLD,DELETE)
//ANLSEP DD DSN=*.ANLSTEP1.ANLSEP,DISP=(OLD,DELETE)
//ANLREL DD DSN=*.ANLSTEP1.ANLREL,DISP=(OLD,DELETE)
//ANLOUT DD DSN=*.ANLSTEP1.ANLOUT,DISP=(OLD,DELETE)
//SYSPRINT DD SYSOUT=*,DCB=LRECL=133
//ANLREP DD DSN=PAOLOR4.ANLREP.SQL,DISP=(,CATLG,DELETE),UNIT=SYSDA,
//          SPACE=(CYL,(10,5),RLSE),DCB=(LRECL=133,RECFM=FBA,BLKSIZE=23275)
//QTRACE DD DSN=PAOLOR4.QTRACE.SQL,DISP=(,CATLG,DELETE),UNIT=SYSDA,
//          SPACE=(CYL,(10,5),RLSE),DCB=(LRECL=120,RECFM=FB,BLKSIZE=23520)
//QLIMIT DD DSN=PAOLOR4.QLIMIT.SQL,DISP=(,CATLG,DELETE),UNIT=SYSDA,
//          SPACE=(TRK,(2,1),RLSE),DCB=(LRECL=80,RECFM=FB,BLKSIZE=9600)
//*
```



ibm.com/redbooks

© 2000 IBM Corporation

10.7.2 The ANLCOST step

The Costing program step uses the information stored in the temporary files produced by the Parser program and performs the analysis delivering the reports in the files allocated by the appropriate DD card.

The SYSPRINT DD is required to print the Cost Summary report.

The name of the report files can be change as you prefer, however do not change the DD card for the temporary and report files.

User parameters in ANLSTP

```
DCL SQL_LEN BIN FIXED(15);
DCL 1 PARM_AREA,
    3 PARM1 CHAR(16) INIT('REPORTS STP '),
    3 PARM2 CHAR(16) INIT('VERSION V6R1'),
    3 PARM3 CHAR(16) INIT('STORAGE 3390-3'),
    3 PARM4 CHAR(16) INIT('BUFFHIT 070 '),
    3 PARM5 CHAR(16) INIT('HPOOLRD 000 '),
    3 PARM6 CHAR(16) INIT('DEGREES ANY '),
    3 PARM7 CHAR(16) INIT('QUALIFY PAOLOR4'), /* <=== YOUR AUTHID */
    3 PARM8 CHAR(16) INIT('CONNECT DSN '),
    3 PARM9 CHAR(16) INIT('PRECISE NO '),
    3 PARMX CHAR(96) INIT(' '); /* ROOM FOR UP TO 15 PARMS */
DCL ANL_PARM CHAR(240) DEF PARM_AREA;
DCL ANL_ELAPS BIN FLOAT(53);
DCL ANL_CPUTM BIN FLOAT(53);
DCL ANL_IOCNT BIN FLOAT(53);
DCL ANL_QUNIT BIN FLOAT(53);
DCL ANL_MONEY BIN FLOAT(53);
DCL ANL_WARN CHAR(5);
DCL ANL_CODE BIN FIXED(31);
DCL SQL_CODE BIN FIXED(31);
```



ibm.com/redbooks

© 2000 IBM Corporation

10.8 The stored procedure interface

SQL/PA can be invoked using the sample batch program provided with the tool (ANLSTP) or you can invoke it from client work station.

In order to use SQL/PA from a client machine you must have DB2 Connect installed, then the simplest way to invoke SQL/PA is to use DB2 Stored Procedure Builder.

The DB2 Stored Procedure Builder (SPB) is a graphical tool designed to help with the development of DB2 stored procedures. It provides all the functions required to create, build, test, and deploy new stored procedures. It also provides functions to work with existing stored procedures.

DB2 SPB is client tool that is part of the DB2 for OS/390 Version 6 Management Tools Package feature and can also be downloaded from the Web. See the redbook *DB2 UDB for OS/390 Version 6 Management Tools Package*, SG24-5759, for more details.

10.8.1 The ANLSTP sample program

The ANLSTP program offers an example about calling the SQL/PA stored procedure in batch mode.

The program needs to be modified before use, in fact the user parameters are hard coded in the program therefore you must change them according to your environment as well as to supply the SQL statement you want to analyze.

In particular you must make sure that the REPORTS parameter is set to STP to enable SQL/PA to format the output correctly. The PARMX variable provides the space for additional parameters, for example the DYNAMIC parameter; in this case you must make sure that the total length of the PARM_AREA adds up correctly to the maximum of 240 bytes.

The program serves as just an example of how to call the ANLPROC stored procedure.

Using the sample program as guideline you can create your own calling program that, for example, might use a DB2 table or just a sequential file to store the user parameters and the input SQL statements.

DB2 SPB output

```

SYSPROC.ANLPROC - The value(s) of the output parameters:
var4              = 9.78579564569489
var5              = 12.997846715398397
var6              = 61.0
var7              = 553.0
var8              = 1.35913828412429
var9              = NNNNN
var10             = 0
var11             = 0

```


ibm.com/redbooks

© 2000 IBM Corporation

10.8.2 Using DB2 Stored Procedure Builder

DB2 SPB gives you the opportunity to quickly call SQL/PA's Stored Procedure to verify the costs of an SQL statement from a work station. The following are the main prerequisites:

- DB2 for OS/390 Version 5 with TCP/IP connectivity enabled and working
- DB2 Connect (any edition) on your work station, installed and configured to with connect to the appropriate DB2 for OS/390 subsystem

Note: Usually DB2 SPB is included with either DB2 Connect or DB2 Software Development Kit. For further information about DB2 SPB, its installation and configuration please refer to *Developing Cross-Platform DB2 Stored Procedures: SQL Procedures and DB2 Stored Procedure Builder*, SC24-5485.

Once configured DB2 SPB will present you a list of stored procedures already existing on the target DB2 subsystem, amongst these you will see SYSPROC.ANLPROC: the SQL/PA stored procedure.

Click with the right-hand side button on procedure name and chose the RUN option: a window will be displayed where you can enter the three required parameters:

1. SQL statement to analyze (CHAR 32000)
2. The length of the statement (SMALLINT)
3. The user parameter (CHAR 240)

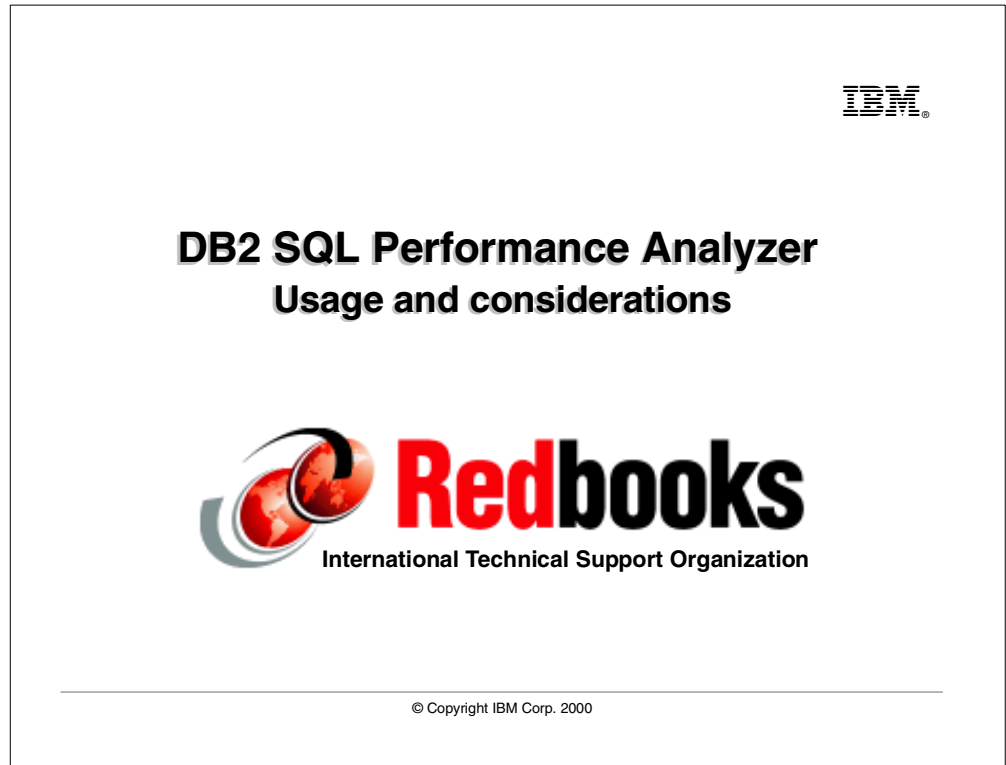
Please remember that each parameter and its relevant value must be separated by one blank.

Click OK and the SQL/PA estimates will be displayed in the Result windows at the bottom of the screen in the form of eight variables. Following the display order, they are:

1. var4 = CPU time estimate
2. var5 = Elapsed time estimate
3. var6 = I/O count estimate
4. var7 = Query Unit estimate
5. var8 = Monetary cost estimate
6. var9 = Limits flags (Y/N)
7. var10 = Return code from ANLPROC
8. var11 = SQL Code of the Prepare or Explain statement

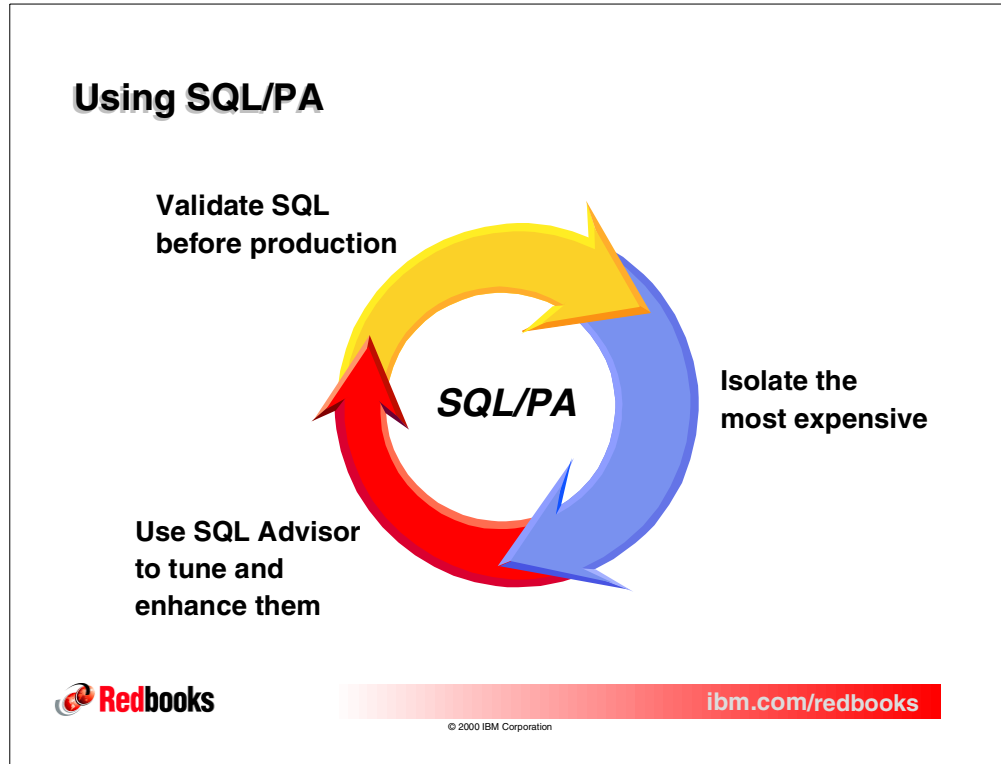
Note: The ANLPROC procedure produces a return code = 777 when any of the limits are exceeded. On certain errors it may also give a greater than 0 return code.

Chapter 11. Usage and considerations



The purpose of this chapter is to demonstrate the use of SQL/PA, to suggest some deployment scenarios, and to further help you understand the main functions and capabilities of SQL/PA.

In addition, we will present some considerations and the limitations found in the current version of SQL/PA.



11.1 Using SQL/PA

DB2 SQL Performance Analyzer for OS/390 is an application tuning tool. You can use it to validate SQL before going into production as well to review and tune existing production applications by scanning their DBRM.

After you have installed SQL/PA and established the resource limits for your system, you may want to verify which one of the existing production applications exceed the thresholds and therefore isolate those that may require some tuning. The same concept can be applied to a QMF query export (or import) data set.

You can also use SQL/PA to estimate the impact of new hardware, either a processor or DASD or even both; simply create a new set of parameters to reflect the new configuration and compare the before and after analysis; there is no need to actually have the hardware.

Furthermore, SQL/PA is useful in those situations where you do not have access to the program(s) source code and you need to investigate what kind of SQL the program performs. This is a common scenario when implementing “off-the-shelf” applications or when the source program is lost.

You can even edit the Cost Summary to extract any SQL statements and reuse them in other programs.

Using the QLIMIT report

Menu Utilities Compilers Help							
BROWSE PAOLOR4.QLIMIT.SQL				Line 00000000 Col 001 080			
Command ==>				Scroll ==> PAGE			
***** Top of Data *****							
CEIQ\$	ERROR	QUERYNO	CPU TIME	ELAPSED	PHYS I/O	QUNITS	MONETARY COST
NNNNN	0	100000001	0.001	0.096	4	2	0.00
NNNNN	0	100000002	0.001	0.072	3	2	0.00
YNNNN	0	100000003	100.482	158.761	1060	567693	41.87
NNNNN	0	100000004	9.636	67.915	1060	54438	4.01
NNNNN	0	100000005	17.829	76.108	1060	100724	7.43
NNNNN	0	100000006	9.637	19.789	1064	54445	4.02
YNNNN	0	100000007	29.010	39.138	1063	163896	12.09
NNNNN	0	100000010	1.724	12.724	201	9736	0.72
***** Bottom of Data *****							

→ SELECT * FROM LINEITEM ;



ibm.com/redbooks

© 2000 IBM Corporation

11.2 A practical example

In our example we take a program in development that accesses an existing Supply Management database. For convenience we have extracted the SQL statements and put them all together in one member of a PDS library and submitted the SQL/PA batch analysis.

11.2.1 Using QLIMIT report

The first step in using SQL/PA is to look at the QLIMIT report. This report shows you at a glance which are the most expensive SQL statements in the input file you used for the analysis.

Using the QUERYNO of the identified SQL we then search either the Enhanced Explained report or just the Cost Summary provided in the job output to find out the SQL statement.

In this case it is an easy pick: `SELECT *` is not really a good practice, in addition the query is not qualifying any rows.

Identify the big hitters

Menu Utilities Compilers Help							
BROWSE PAOLOR4.QLIMIT.SQL				Line 00000000 Col 001 080			
Command ==>				Scroll ==> PAGE			
***** Top of Data *****							
CEIQ\$	ERROR	QUERYNO	CPU TIME	ELAPSED	PHYS I/O	QUNITS	MONETARY COST
NNNNN	0	100000001	0.001	0.096	4	2	0.00
NNNNN	0	100000002	0.001	0.072	3	2	0.00
NNNNN	0	100000003	19.004	77.283	1060	107367	7.92
NNNNN	0	100000004	9.636	67.915	1060	54438	4.01
NNNNN	0	100000005	17.829	76.108	1060	100724	7.43
NNNNN	0	100000006	9.637	19.789	1064	54445	4.02
YNNNN	0	100000007	29.010	39.138	1063	163896	12.09
NNNNN	0	100000010	1.724	12.724	201	9736	0.72
***** Bottom of Data *****							

→ SELECT L_DISCOUNT, L_TAX, L_QUANTITY
FROM LINEITEM
WHERE L_SHIPMODE LIKE 'TR%' ;



ibm.com/redbooks

© 2000 IBM Corporation

11.2.2 Identify the big hitters

The best thing to do in this case is try to be more specific and define exactly which columns we want to retrieve and maybe defining a WHERE condition to qualify only a subset of rows.

In a small amount of time we tested the validity of your new SQL and when browsing the QLIMIT report again we immediately appreciated the impact of the change.

Another way to identify SQL statements that may require tuning is to establish the thresholds beyond which you want SQL/PA to raise a flag.

There are five flags at the beginning of each SQL summary record in the QLIMIT report that can assume Y/N values. In this case we have the CPU time limit set to 20 seconds and two statements were flagged as exceeding the CPU time limit.

We fixed one and there is still another to go.

Using the Enhanced Explain report

```
EXPLAIN ALL SET QUERYNO = 100000007 FOR
SELECT L_TAX, L_EXTENDEDPRICE, L_COMMENT, O_ORDERDATE, O_CUSTKEY
FROM PAOLOR4.LINEITEM, PAOLOR4.ORDER
WHERE L_ORDERKEY = O_ORDERKEY
AND L_PARTKEY = 112777 ;

QUERYNO: 100000007 QBLOCKNO: 1 PLANNO: 1 MIXSEQ: 0
PROCESS ->

A TABLESPACE SCAN IS REQUESTED ON A PARTITIONED TSPACE OF 3 PARTS.
IN TABLE PAOLOR4 .LINEITEM 33832 4K PAGES WILL BE READ.
TABLE HOLDS 1951548 ROWS OF 135 BYTES, WITH 16 COLS.
TYPE = T, LOCK SIZE = A, CLOSE = N, TS LOCK MODE = IS, LOCK PART = .
PCT WITH ROWS = 99%, ENCODE = E, MAXROW = 255, PCT COMPRESSED = 99%.

A JOIN OF 2 TABLES HAS BEEN DETECTED. THIS WAS THE FIRST TABLE ACCESS.

QUERYNO: 100000009 QBLOCKNO: 1 PLANNO: 2 MIXSEQ: 0
PROCESS ->

A MATCHING INDEX SCAN AGAINST A CLUSTERED INDEX, WITH LOCK MODE = IS.
1 OF 5 COLS ARE MATCHED IN THE UNIQUE INDEX, IXTYPE = 2, CLOSE = N.
INDEX PAOLOR4 .PXOKODCKSPOP HAS CLUSTER RATIO 100, SUBPAGE 4096.
THERE ARE 3 LEVELS IN THE INDEX AND 3269 LEAF PAGES IN THE TREE.
FIRST KEY = 336519, AND FULL KEY = 336519.

THIS IS THE CLUSTERING (INSERT) INDEX FOR TABLE PAOLOR4 .ORDER

THIS IS AN "INDEX ONLY" ACCESS: NO DATA PAGES ARE READ FROM THE TABLE.

THIS TABLE IS JOINED WITH THE PRIOR TABLE VIA THE "NESTED LOOP" METHOD.
THIS IS THE INNER TABLE IN THE NESTED LOOP JOIN.
```



ibm.com/redbooks

© 2000 IBM Corporation

11.2.3 Using the enhanced Explain report

For the second expensive query, we look at the Explain report and immediately see that this query performs a join between two tables LINEITEM and ORDER.

The plan for this query is split into two steps:

- A table space scan is performed on the LINEITEM table which contains almost two million records
- A matching index scan is then performed on the ORDER table using the clustering (and clustered) index and accessing only the index

It is evident that the access to the ORDER table is well performed, in fact this portion of the query is not only using the clustering index but is also finding the requested columns in the same index, therefore avoiding access to the table.

The problem seems to be in the first step where the query performs a table space scan in spite of the rows qualification using the L_PARTKEY column. LINEITEM is a partitioned table and therefore must have a partitioning and clustering index but the L_PARTKEY column is not probably part of the key (in spite of the column name).

A better access path

```
EXPLAIN ALL SET QUERYNO = 100000007 FOR
SELECT L_TAX, L_EXTENDEDPRICE, L_COMMENT, O_ORDERDATE, O_CUSTKEY
FROM PAOLOR4.LINEITEM, PAOLOR4.ORDER
WHERE L_ORDERKEY = O_ORDERKEY
AND L_PARTKEY = 112777 ;

QUERYNO: 100000009 QBLOCKNO: 1 PLANNO: 1 MIXSEQ: 0
PROCESS ->

A MATCHING INDEX SCAN IS APPLIED TO RANDOM INDEX WITH LOCK MODE = IS.
1 OF 6 COLS ARE MATCHED IN THE UNIQUE INDEX, IXTYPE = 2, CLOSE = N.
INDEX PAOLOR4 .SXL#PKSKEPDSQN HAS CLUSTER RATIO 50, SUBPAGE 4096.
THERE ARE 3 LEVELS IN THE INDEX AND 15489 LEAF PAGES IN THE TREE.
FIRST KEY = 199987, AND FULL KEY = 1951548.

THIS INDEXED ACCESS IS APPLIED TO A PARTITIONED TSPACE OF 3 PARTS.
IN TABLE PAOLOR4 .LINEITEM 33832 4K PAGES WILL BE READ.
TABLE HOLDS 1951548 ROWS OF 135 BYTES, WITH 16 COLS.
TYPE = T, LOCK SIZE = A, CLOSE = N, TS LOCK MODE = IS, LOCK PART = .
PCT WITH ROWS = 99%, ENCODE = E, MAXROW = 255, PCT COMPRESSED = 99%.

A JOIN OF 2 TABLES HAS BEEN DETECTED. THIS WAS THE FIRST TABLE ACCESS.

QUERYNO: 100000009 QBLOCKNO: 1 PLANNO: 2 MIXSEQ: 0
PROCESS ->

A MATCHING INDEX SCAN AGAINST A CLUSTERED INDEX, WITH LOCK MODE = IS.
1 OF 5 COLS ARE MATCHED IN THE UNIQUE INDEX, IXTYPE = 2, CLOSE = N.
INDEX PAOLOR4 .PXO#OKODCKSPOP HAS CLUSTER RATIO 100, SUBPAGE 4096.
THERE ARE 3 LEVELS IN THE INDEX AND 3269 LEAF PAGES IN THE TREE.
FIRST KEY = 336519, AND FULL KEY = 336519.
```



ibm.com/redbooks

© 2000 IBM Corporation

11.2.4 A better access path

A quick investigation with the DB2 catalog confirms that the partitioning index is the only index for this table and that the L_PARTKEY column is not even part of the key.

Depending on the database design it may be appropriate or not to create an additional index. This must be evaluated and decided by the Database Administrator. In this case we decide that it was appropriate to create a secondary index using the L_PARTKEY column as well as some other columns for the benefits of the whole design.

After the index creation we execute the RUNSTATS and reevaluate the statement using SQL/PA. The Enhanced Explain report tells us that we have done the right thing.

A matching index scan is now used to identify the records qualified by L_PARTKEY and then only the relevant records in the tables are accessed with a direct hit and then joined with the ORDER table.

A big improvement

Menu Utilities Compilers Help									

BROWSE		PAOLOR4.QLIMIT.SQL				Line 00000000 Col 001 080			
Command ==>						Scroll ==> CSR			
***** Top of Data *****									
CEIQ\$	ERROR	QUERYNO	CPU TIME	ELAPSED	PHYS I/O	QUNITS	MONETARY	COST	
NNNNN	0	100000001	0.001	0.096	4	2		0.00	
NNNNN	0	100000002	0.001	0.072	3	2		0.00	
NNNNN	0	100000003	19.004	77.283	1060	107367		7.92	
NNNNN	0	100000004	0.001	0.096	4	4		0.00	
NNNNN	0	100000005	11.291	11.386	4	63791		4.70	
NNNNN	0	100000006	0.002	0.101	8	11		0.00	
NNNNN	0	100000007	0.002	0.078	7	11		0.00	
NNNNN	0	100000008	1.724	12.724	201	9736		0.72	
***** Bottom of Data *****									

*... KILLING 'FOUR' BIRDS WITH ONE STONE ...
EVEN WITHOUT THROWING IT!!*



ibm.com/redbooks

© 2000 IBM Corporation

11.2.5 The final result

An even quicker verification of the new index validity to check again the QLIMIT report.

Immediately we see that not only the estimates had a dramatic improvement and that the statement is not flagged as having exceeded the limits any longer, we also improved the performance of another three SQL statements within the same program.

We achieved this result in a matter of a few minutes, the bulk of which was the index creation and the RUNSTATS execution and without ever executing the SQL statements.

Considerations

Generic PLAN_TABLE performs better

- however Stored Procedure interface works only with personal PLAN_TABLE

Maintain very accurate statistics in the DB2 catalog

Use the estimates as relative values and not as absolute values

- SQL/PA is a performance model tool, not a real time monitor

QLIMIT and Enhanced Explain reports are the most useful

- the QTRACE report is only need for particular investigations and provides far too many details

Plan to experiment with the parameters

- SQL/PA needs to be tuned for your own environment

The User's Guide manual provide very good information about SQL performance and tuning



ibm.com/redbooks

© 2000 IBM Corporation

11.3 Considerations

There are a number of considerations to keep in mind when working with SQL/PA, they are summarized below.

Either generic or personal PLAN_TABLEs are needed if your DB2 has development performed remotely on a client work station. The generic PLAN_TABLEs provide better performance than the personal ones.

Accurate catalog statistics are the foundations of accurate access path selection, therefore the usability of SQL/PA's estimates is directly linked to the validity of the objects statistics.

Estimates are the projection of what it will be while measurements are the evaluation of what it has been. Do not expect SQL/PA to estimate exactly the same value displayed by a DB2 PM report. You can compare two distinct SQL/PA's analysis of an SQL statement but you cannot compare the estimates with the actual without considerable effort.

The QLIMIT report is very useful when working with multiple SQL statements with the same analysis execution; you can then use the Enhanced Explain to go into the details of the access path. With these two reports you will perform most of the tuning. The QTRACE report provides an higher level of sophistication and in some case can be overwhelming; so use it only when necessary.

Do not use the tools without appropriate tuning for your environment. The following page provide some additional information about parameters.

What influences cost estimates

Elapsed time

- STORAGE
 - or NEWxxxx parameters
- BUFFHIT
- HPOOLRD
- CONNECT
- BUFFERS
 - 8K, 16K, 32K and SORTBUF
- DATASHR and DSGROUP
- DEGREES

CPU time

- VERSION
- MIPRATE + ENGINES
 - or SRMCONS
- LPARENG
- ESASORT and ESACOMP
- DYNAMIC
- PRECISE
- ANLKEYS



ibm.com/redbooks

© 2000 IBM Corporation

11.3.1 What influences cost estimates

Most of the parameters specified in both the ANLPARM and ANLCNTL parameter sets will have some impact on the internal formulas and costing algorithms used by SQL/PA. The following list reviews where the impact will be.

VERSION selects which set of path lengths are used, with different instruction counts on some functions.

CONNECT imposes overhead charges for thread management, attach facility and application logic, different for each connection chosen.

STORAGE provides the basis for service times calculated for all physical I/O calls, based on the seek, revolution speed and transfer rate of the devices; also impacts the wait time estimates and contributes to overall elapsed time.

BUFFHIT governs the logical-to-physical I/O ratio, allocating a portion of the total logical I/O to disk; impacts the I/O waits, elapsed time, and other calculations.

HPOOLRD governs read time of physical I/O: are they found in expanded storage, or read from disk? Impacts elapsed times and I/O waits; also affects asynchronous writes back to pool after insert, update, or delete activity.

DEGREES enables the optimizer to consider parallel processing; this has an impact on sequential prefetch, I/O wait, elapsed time for table scans.

NEWSTOR, **NEWSEEK**, **NEWROTA** and **NEWXFER** redefine the basis for service times calculated for all physical I/O calls, based on the seek, revolution speed and transfer rate of the new devices; they also impact the wait time estimates and contribute to overall elapsed time.

MIPRATE sets the processor speed of the complex, and the aggregate MIPS (millions of instructions) processed by the target host system.

ENGINES (along with MIPRATE) contributes to determining the speed of a single processing engine, the fastest any instructions will execute.

SRMCONS provides a more conservative alternative to the MIPRATE and ENGINES method of computing the individual processor speeds.

LPARENG helps rate individual processor speed on processor complexes with multiple system images.

ESASORT invokes different path length algorithms within SQL/PA for systems with the hardware-assisted DB2 sort facility.

ESACOMP invokes different path length algorithms within SQL/PA for table spaces using the hardware-assisted data compression facility.

BUFFERS (4K, 8K, 16K and 32K) affects the number of pages contained in a prefetch block for user requests.

SORTBUF affects the number of sort/merge passes and sort work files I/O.

CPUCOST establishes the base for the monetary cost estimates of CPU Time.

IOSCOST establishes the base for the monetary cost estimates of physical I/O.

TIMCOST establishes the base for the monetary cost estimates of Elapsed Time.

DATASHR and **DSGROUP** affect the overhead applied to data sharing in a Sysplex environment, and the overall processing capacity of the system.

DYNAMIC affects the overhead for dynamic SQL processing.

PRECISE causes SQL/PA's calculated CPU Times and QUNITS to be replaced with DB2 optimizer's estimates.

ANLKEYS alters consideration of certain path lengths and instruction counts, leaving out some overhead depending upon option selected.

Limitations

Catalog information in memory

- 100 object per type of objects per run

Computational limits

- 24 hours for CPU and Elapsed time; 99,999,999 for I/Os

Wildcarding

- not supported

Host variables

- mostly managed but still a few cases

Limits for QMF Intercept and Stored Procedure per single SQL

- 64 query blocks, 64 relationships, 128 plan records, 128 tables and indexes, 100 errors



ibm.com/redbooks

© 2000 IBM Corporation

11.4 Limitations

DB2 SQL Performance Analyzer Version 1 has a number of limitations:

1. It will store catalog information for a maximum of:

- 100 separate tables spaces
- 100 separate tables
- 100 separate indexes
- 100 separate relationships

Additional access to the catalog will be required when these limits are exceeded.

2. The maximum computations permitted are:

- 24 hours or 86,400 seconds for CPU time
- 24 hours or 86,400 seconds for Elapsed time
- 99,999,999 physical calls for I/O count

3. Wildcards are not supported, particularly with the ISPF interface.

4. In most of the cases SQL/PA handles host variables in DBRMs replacing them with a ?; however, there are still cases when the DB2 prepare returns and SQL code -104, they are:

- Both end of equality use host variables (? =?)
- The operand of a Date/Time arithmetic operation is an host variable
- The argument of a scalar function in a SELECT or ESCAPE clause is an host variable

In addition, the QMF Intercept Exit and Stored Procedure imposes the following limits:

- A maximum of 64 Query Blocks can be handled within a single SQL statement
- A maximum of 64 RI relationships can be handled by the exit per SQL statement
- A maximum of 128 plan records generated by the prepare of the SQL statement can be handled
- A maximum total of 128 combined tables and indexes can be referenced by a single SQL statement
- A maximum of 100 errors will be allowed by SQL/PA for all SQL statements before the program terminates

Part 5. Appendices

Appendix A. Bind Manager

This appendix maps the report columns against the plan table columns.

A.1 Path Checker Report Column mapping against Plan_Table

- QRYNO - Query Number
- M - Method
- Creator - Creator
- TNAME - TNAME
- TBNO - TABNO
- AC - ACESSTYPE
- MC - MATCHCOLS
- CREATOR - ACCESSCREATOR
- ACESSNAME - ACCESSNAME
- IO - INDEXONLY
- SORTUJGO -
 - SORTN_UNIQ
 - SORTN_JOIN
 - SORTN_ORDERBY
 - SORTN_GROUPBY
 - SORTC_UNIQ
 - SORTC_JOIN
 - SORTC_ORDERBY
 - SORTC_GROUPBY
- LK - TSLOCKMODE
- PF - PREFETCH
- QBNO - QUERYBLOCK NO
- PLNO - PLANNO
- MXSQ - MIXOPSEQ
- ACC-DEG - ACCESS_DEGREE
- ACC-PGP - ACCESS_PGROUP_ID
- JN-DEG - JOIN_DEGREE
- JN-PGP - JOIN_PGROUP_ID
- SRTC-PGP - SORTC_PGROUP_ID
- SRTN-PGP - SORTN_PGROUP_ID
- PRLL-MD - PARALLELISM_MODE
- MRG-JCL - MERGE_JOIN_COLS
- PG-RNG - PAGE_RANGE
- JN-TYP - JOIN_TYPE
- WH-OPT - WHEN_OPTIMIZE

Appendix B. Query Monitor

This appendix contains output from DB2 Performance Monitor (DB2 PM) and from Query Monitor. The PM report is the long accounting report, and the output from Query Monitor was generated from using the PRINTX command on selected screens to generate the information.

B.1 Query Monitor sample output

Query Monitor thread details

Cmd	SSID	Intv	Plan Name	Job Name	DB2 CPU Time	DB2 Elapsed	APPL CPU	APPL Elapsed
—	DB2Y	-	DSNESPCS	PAOLOR3	17.916116	20.317978	0.020870	0.077569
—	DB2Y	-	DGOPMOM	PAOLOR3	0.000000	0.000000	0.000000	0.000000
—	DB2Y	-	DGOPMOM	PAOLOR3	0.000000	0.000000	0.000000	0.000000
—	DB2Y	-	DSNESPCS	PAOLOR3	17.082057	18.753023	0.019725	0.071651
—	DB2Y	-	DSNESPCS	PAOLOR3	11.797311	17.181197	0.175377	0.290936
—	DB2Y	-	DSNESPCS	PAOLOR3	1.135894	11.927914	0.020642	0.146119
—	DB2Y	-	DSNESPCS	PAOLOR3	17.120323	20.602996	0.020429	0.074960
—	DB2Y	-	DSNESPCS	PAOLOR3	0.010415	0.022707	0.020069	0.110804
—	DB2Y	-	DISTSERV	DB2YDIST	0.548523	1.844274	0.168847	1:25.266325

Query Monitor delays

Delay Event	Event Count	Delay Time
Lock or Latch Delays	118	0.002393
Synchronous I/O Delays	8	0.040665
Other Read Delays	40	0.085061
Other Write Delays	4	0.113152
Service Task Switch Delays	0	0.000000
Archive Log Quiesce Delays	0	0.000000
Archive Log Read Delays	0	0.000000
Drain Lock Delays	0	0.000000
Claim Release Delays	0	0.000000
Page Latch Delays	0	0.000000
Stored Procedure Delays	0	0.000000
Notify Message Delays	0	0.000000
Global Contention Delays	0	0.000000
Total Delays	170	0.241272

Query Monitor bufferpools

Bufferpool:	BP0	BP2	All BP
Get Page Requests	40,042	3,775	43,817
Buffer Pages Updated	608	0	608
Synchronous Pages Read	131	0	131
Synchronous Pages Written	0	0	0
Sequential Prefetch Requests	1,302	0	1,302
List Prefetch Requests	0	0	0
Dynamic Prefetch Requests	0	0	0
Successful Hiperpool Reads	0	0	0
Hiperpool Read Failures	0	0	0
Successful Hiperpool Writes	0	0	0
Unsuccessful Hiperpool Writes	0	0	0
Asynchronous Pages Read	39,574	0	39,574
Async Pages Read by Hiperpool	0	0	0

Query Monitor instructions

Instruction	Totals	Instruction	Totals	Instruction	Totals
-----	-----	-----	-----	-----	-----
Select	—	Comment On	—	Drop View	—
Open	—	Connect	—	Explain	—
Close	—	Connect Reset	—	Fetch Alloc Cs	—
Fetch	—	Connect To	—	Implicit Conne	—
Insert	—	Create Alias	—	Intopen	—
Update	—	Create Databas	—	Label On	—
Delete	—	Create Index	—	Release All	—
Prepare	—	Create Stogrou	—	Release All Pr	—
Lock	—	Create Synonym	—	Release All SQ	—
Commit	—	Create Table	—	Release Curren	—
Rollback	—	Create Tablesp	—	Release Locati	—
Grant	—	Create View	—	Remote SQL	—
Revoke	—	Describe	—	Rename Table	—
Execute	—	Describe Alloc	—	Set Connection	—
Execute Immedi	—	Describe Input	—	Set Cur Degree	—
Allocate Curso	—	Describe Proce	—	Set Cur Pkgset	—
Alter Database	—	Drop Alias	—	Set Cur Rules	—
Alter Index	—	Drop Database	—	Set Cur SQLID	—
Alter Stogroup	—	Drop Index	—	Set Host Varia	—
Alter Table	—	Drop Package P	—	Set Special Re	—

Query Monitor locking

Lock Event	Event Count
—Lock Deadlocks	— 0
—Lock Suspensions	— 0
—Lock Timeouts	— 0
—Latch Suspensions	— 1
—Other Suspensions	— 0
—Lock Requests	— 18,867
—Unlock Requests	— 18,862
—Query Requests	— 0
—Change Requests	— 1
—Other Requests	— 0
—Claim Requests	— 9
—Claim Failures	— 0
—Drain Requests	— 0
—Drain Failures	— 0
—XES Lock Requests	— 5
—XES Change Requests	— 0
—XES Unlock Requests	— 1
—IRLM Global Resource Contentio	— 0
—XES Global Resource Contention	— 0
—False Resource Contention	— 0
—Incompatible Retain Lock	— 0
—Shared Lock Escalations	— 0
—Exclusive Lock Escalations	— 0

B.2 DB2 PM long accounting report sample

```
1  LOCATION: DB2Y                      DB2 PERFORMANCE MONITOR (V6)                      PAGE: 1-1
    GROUP: DB2V610Y                   ACCOUNTING TRACE - LONG                      REQUESTED FROM: NOT SPECIFIED
    MEMBER: DB2Y                                                                TO: NOT SPECIFIED
    SUBSYSTEM: DB2Y                                                            ACTUAL FROM: 12/01/00 18:56:35.41
DB2 VERSION: V6
```

```
----- IDENTIFICATION -----
ACCT TSTAMP: 12/01/00 18:56:35.41    PLANNAME: DSNESPCS          WLM SCL: 'BLANK'          CICS NET: N/A
BEGIN TIME : 12/01/00 18:56:15.02    PROD ID : N/P              CICS LUN: N/A
END TIME   : 12/01/00 18:56:35.41    PROD VER: N/P              CICS INS: N/A
REQUESTER  : DB2Y                   CORRNAME: PAOLOR3          LUW NET: USIBMSC          LUW LUN: SCPDB2Y
MAINPACK   : DSNESM68               CORRNMBR: 'BLANK'          LUW INS: B50747057690     ENDUSER : 'BLANK'
PRIMAUTH   : PAOLOR3                CONNTYPE: TSO              LUW SEQ: 1                TRANSACT: 'BLANK'
ORIGAUTH   : PAOLOR3                CONNECT : TSO              WSNAME : 'BLANK'
```

```
MVS ACCOUNTING DATA : ACCNT#
ACCOUNTING TOKEN(CHAR): N/A
ACCOUNTING TOKEN(HEX) : N/A
```

```
ELAPSED TIME DISTRIBUTION                                     CLASS 2 TIME DISTRIBUTION
-----
APPL |=====
DB2  |===== > 98%
SUSP |=> 2%

CPU |===== > 88%
NOTACC |===== > 10%
SUSP   |=> 2%
```

TIMES/EVENTS	APPL (CLASS 1)	DB2 (CLASS 2)	IFI (CLASS 5)	CLASS 3 SUSP.	ELAPSED TIME	EVENTS	HIGHLIGHTS
ELAPSED TIME	20.395541	20.318823	N/P	LOCK/LATCH	0.002516	62	THREAD TYPE : ALLIED
NONNESTED	20.395541	20.318823	N/A	SYNCHRON. I/O	0.181926	131	TERM.CONDITION: NORMAL
STORED PROC	0.000000	0.000000	N/A	DATABASE I/O	0.181926	131	INVOKE REASON : DEALLOC
UDF	0.000000	0.000000	N/A	LOG WRITE I/O	0.000000	0	COMMITTS : 2
TRIGGER	0.000000	0.000000	N/A	OTHER READ I/O	0.092543	21	ROLLBACK : 0
				OTHER WRTE I/O	0.113152	2	INCREM.BINDS : 0
CPU TIME	17.936899	17.916715	N/P	SER.TASK SWITCH	0.000118	1	UPDATE/COMMIT : 0.00
AGENT	17.936899	17.916715	N/A	UPDATE COMMIT	0.000118	1	SYNCH I/O AVG.: 0.001389
NONNESTED	17.936899	17.916715	N/P	OPEN/CLOSE	0.000000	0	PROGRAMS : 1
STORED PRC	0.000000	0.000000	N/A	SYSLGRNG REC	0.000000	0	MAX CASCADE : 0
UDF	0.000000	0.000000	N/A	EXT/DEL/DEF	0.000000	0	PARALLELISM : NO
TRIGGER	0.000000	0.000000	N/A	OTHER SERVICE	0.000000	0	
PAR.TASKS	0.000000	0.000000	N/A	ARC.LOG(QUIES)	0.000000	0	
				ARC.LOG READ	0.000000	0	
SUSPEND TIME	N/A	0.390256	N/A	STOR.PRC SCHED	0.000000	0	
AGENT	N/A	0.390256	N/A	UDF SCHEDULE	0.000000	0	
PAR.TASKS	N/A	0.000000	N/A	DRAIN LOCK	0.000000	0	
				CLAIM RELEASE	0.000000	0	
NOT ACCOUNT.	N/A	2.011852	N/P	PAGE LATCH	0.000000	0	
DB2 ENT/EXIT	N/A	19	N/A	NOTIFY MSGS	0.000000	0	
EN/EX-STPROC	N/A	0	N/A	GLOBAL CONT.	0.000000	0	
EN/EX-UDF	N/A	0	N/A	TOTAL CLASS 3	0.390256	217	
DCAPT.DESCR.	N/A	N/A	N/P				
LOG EXTRACT.	N/A	N/A	N/P				

1	LOCATION: DB2Y	DB2 PERFORMANCE MONITOR (V6)	PAGE: 1-2
	GROUP: DB2V610Y	ACCOUNTING TRACE - LONG	REQUESTED FROM: NOT SPECIFIED
	MEMBER: DB2Y		TO: NOT SPECIFIED
	SUBSYSTEM: DB2Y		ACTUAL FROM: 12/01/00 18:56:35.41
	DB2 VERSION: V6		

---- IDENTIFICATION ----

ACCT TSTAMP: 12/01/00 18:56:35.41	PLANNAME: DSNESPCS	WLM SCL: 'BLANK'	CICS NET: N/A
BEGIN TIME : 12/01/00 18:56:15.02	PROD ID : N/P		CICS LUN: N/A
END TIME : 12/01/00 18:56:35.41	PROD VER: N/P	LWU NET: USIBMSC	CICS INS: N/A
REQUESTER : DB2Y	CORRNAME: PAOLOR3	LWU LUN: SCPDB2Y	
MAINPACK : DSNESM68	CORRNMBR: 'BLANK'	LWU INS: B50747057690	ENDUSER : 'BLANK'
PRIMAUTH : PAOLOR3	CONNTYPE: TSO	LWU SEQ: 1	TRANSACTION: 'BLANK'
ORIGAUTH : PAOLOR3	CONNECT : TSO		WSNAME : 'BLANK'

SQL DML	TOTAL	SQL DCL	TOTAL	SQL DDL	CREATE	DROP	ALTER	LOCKING	TOTAL	DATA SHARING	TOTAL
SELECT	0	LOCK TABLE	0	TABLE	0	0	0	TIMEOUTS	0	GLB CONT (%)	0
INSERT	0	GRANT	0	TEMP TABLE	0	N/A	N/A	DEADLOCKS	0	FLS CONT (%)	0
UPDATE	0	REVOKE	0	AUX TABLE	0	N/A	N/A	ESCAL. (SHAR)	0	L-LOCKS (%)	0
DELETE	0	SET SQLID	1	INDEX	0	0	0	ESCAL. (EXCL)	0	LOCK REQUEST	0
		SET H.VAR.	0	TABLESPACE	0	0	0	MAX.LCK HELD	1	UNLOCK REQST	0
DESCRIBE	0	SET DEGREE	0	DATABASE	0	0	0	LOCK REQUEST	18867	CHANGE REQST	0
DESC.TBL	0	SET RULES	0	STOGROUP	0	0	0	UNLOCK REQST	18863	LOCK - XES	5
PREPARE	2	SET PATH	0	SYNONYM	0	0	N/A	QUERY REQST	0	UNLOCK-XES	1
OPEN	1	SET PREC.	0	VIEW	0	0	N/A	CHANGE REQST	1	CHANGE-XES	0
FETCH	2	CONNECT 1	0	ALIAS	0	0	N/A	OTHER REQST	0	SUSP - IRLM	0
CLOSE	1	CONNECT 2	0	PACKAGE	N/A	0	N/A	LOCK SUSP.	0	SUSP - XES	0
		SET CONNEX	0	PROCEDURE	0	0	0	LATCH SUSP.	1	SUSP - FALSE	0
		RELEASE	0	FUNCTION	0	0	0	OTHER SUSP.	0	INCOMP.LOCK	0
DML-ALL	6	CALL	0	TRIGGER	0	0	N/A	TOTAL SUSP.	1	NOTIFY SENT	0
		ASSOC LOC.	0	DIST TYPE	0	0	N/A				
		ALLOC CUR.	0								
		HOLD LOC.	0	TOTAL	0	0	0				
		FREE LOC.	0	RENAME TBL	0						
		DCL-ALL	1	COMMENT ON	0						
				LABEL ON	0						

RID LIST	TOTAL	ROWID	TOTAL	STORED PROC.	TOTAL	UDF	TOTAL	TRIGGERS	TOTAL
USED	0	DIR ACCESS	0	CALL STMTS	0	EXECUTED	0	STMT TRIGGER	0
FAIL-NO STORAGE	0	INDEX USED	0	ABENDED	0	ABENDED	0	ROW TRIGGER	0
FAIL-LIMIT EXC.	0	TS SCAN	0	TIMED OUT	0	TIMED OUT	0	SQL ERROR	0
				REJECTED	0	REJECTED	0		

1	LOCATION: DB2Y	DB2 PERFORMANCE MONITOR (V6)	PAGE: 1-3
	GROUP: DB2V610Y	ACCOUNTING TRACE - LONG	REQUESTED FROM: NOT SPECIFIED
	MEMBER: DB2Y		TO: NOT SPECIFIED
	SUBSYSTEM: DB2Y		ACTUAL FROM: 12/01/00 18:56:35.41
	DB2 VERSION: V6		

```

----- IDENTIFICATION -----
ACCT TSTAMP: 12/01/00 18:56:35.41  PLANNAME: DSNESPCS      WLM SCL: 'BLANK'      CICS NET: N/A
BEGIN TIME  : 12/01/00 18:56:15.02  PROD ID  : N/P        CICS LUN: N/A
END TIME    : 12/01/00 18:56:35.41  PROD VER: N/P        CICS INS: N/A
REQUESTER   : DB2Y                  CORRNAME: PAOLOR3     LUW NET: USIEMSC
MAINPACK    : DSNESM68              CORRNMBR: 'BLANK'     LUW LUN: SCPDB2Y
PRMAUTH     : PAOLOR3              CONNTYPE: TSO         LUW INS: B50747057690  ENDUSER  : 'BLANK'
ORIGAUTH    : PAOLOR3              CONNECT  : TSO        LUW SEQ: 1             TRANSACT: 'BLANK'
                                           WSNAME   : 'BLANK'

```

QUERY PARALLEL.	TOTAL	DATA CAPTURE	TOTAL	SERVICE UNITS	CLASS 1	CLASS 2
MAXIMUM MEMBERS	N/P	IFI CALLS	N/P	CPU	101231	101117
MAXIMUM DEGREE	0	REC.CAPTURED	N/P	AGENT	101231	101117
GROUPS EXECUTED	0	LOG REC.READ	N/P	NONNESTED	101231	101117
RAN AS PLANNED	0	ROWS RETURN	N/P	STORED PRC	0	0
RAN REDUCED	0	RECORDS RET.	N/P	UDF	0	0
ONE DB2 COOR=N	0	DATA DES.RET	N/P	TRIGGER	0	0
ONE DB2 ISOLAT	0	TABLES RET.	N/P	PAR.TASKS	0	0
SEQ - CURSOR	0	DESCRIBES	N/P			
SEQ - NO ESA	0					
SEQ - NO BUF	0					
SEQ - ENCL.SER	0					
MEMB SKIPPED(%)	0					
DISABLED BY RLF	NO					
REFORM PARAL-CONFIG	0					
REFORM PARAL-NO BUF	0					

OPTIMIZATION	TOTAL	DRAIN/CLAIM	TOTAL	LOGGING	TOTAL	MISCELLANEOUS	TOTAL
REOPTIMIZATION	0	DRAIN REQST	0	LOG RECS WRITTEN	0	MAX STOR VALUES	0
PREP_STMT_MATCH	1	DRAIN FAILED	0	TOT BYTES WRITTEN	0		
PREP_STMT_NO_MATCH	0	CLAIM REQST	9				
IMPLICIT_PREPARES	0	CLAIM FAILED	0				
PREP_FROM_CACHE	0						
CACHE_LIMIT_EXCEED	0						
PREP_STMT_PURGED	0						

```

----- RESOURCE LIMIT FACILITY -----
TYPE: N/P      TABLE ID: N/P  SERV.UNITS:  N/P  CPU SECONDS: 0.000000  MAX CPU SEC:  N/P

```

1	LOCATION: DB2Y	DB2 PERFORMANCE MONITOR (V6)	PAGE: 1-4
	GROUP: DB2V610Y	ACCOUNTING TRACE - LONG	REQUESTED FROM: NOT SPECIFIED
	MEMBER: DB2Y		TO: NOT SPECIFIED
	SUBSYSTEM: DB2Y		ACTUAL FROM: 12/01/00 18:56:35.41
	DB2 VERSION: V6		

```

----- IDENTIFICATION -----
ACCT TSTAMP: 12/01/00 18:56:35.41  PLANNAME: DSNESPCS      WLM SCL: 'BLANK'      CICS NET: N/A
BEGIN TIME : 12/01/00 18:56:15.02  PROD ID : N/P          CICS LUN: N/A
END TIME   : 12/01/00 18:56:35.41  PROD VER: N/P          CICS INS: N/A
REQUESTER  : DB2Y                  CORRNAME: PAOLOR3      LUW LUN: SCPDB2Y
MAINPACK   : DSNESM68              CORRNMBR: 'BLANK'      LUW INS: B50747057690  ENDUSER : 'BLANK'
PRIMAUTH   : PAOLOR3              CONNTYPE: TSO          LUW SEQ: 1             TRANSACT: 'BLANK'
ORIGAUTH   : PAOLOR3              CONNECT : TSO          WSNAME  : 'BLANK'

```

BP0	TOTAL	BP2	TOTAL	TOT4K	TOTAL
BPOOL HIT RATIO (%)	0	BPOOL HIT RATIO (%)	100	BPOOL HIT RATIO (%)	9
GETPAGES	40042	GETPAGES	3775	GETPAGES	43817
GETPAGES-FAILED	0	GETPAGES-FAILED	0	GETPAGES-FAILED	0
BUFFER UPDATES	608	BUFFER UPDATES	0	BUFFER UPDATES	608
SYNCHRONOUS WRITE	0	SYNCHRONOUS WRITE	0	SYNCHRONOUS WRITE	0
SYNCHRONOUS READ	131	SYNCHRONOUS READ	0	SYNCHRONOUS READ	131
SEQ. PREFETCH REQS	1302	SEQ. PREFETCH REQS	0	SEQ. PREFETCH REQS	1302
LIST PREFETCH REQS	0	LIST PREFETCH REQS	0	LIST PREFETCH REQS	0
DYN. PREFETCH REQS	0	DYN. PREFETCH REQS	0	DYN. PREFETCH REQS	0
PAGES READ ASYNCHR.	39574	PAGES READ ASYNCHR.	0	PAGES READ ASYNCHR.	39574
HPOOL WRITES	0	HPOOL WRITES	0	HPOOL WRITES	0
HPOOL WRITES-FAILED	0	HPOOL WRITES-FAILED	0	HPOOL WRITES-FAILED	0
PAGES READ ASYN-HPOOL	0	PAGES READ ASYN-HPOOL	0	PAGES READ ASYN-HPOOL	0
HPOOL READS	0	HPOOL READS	0	HPOOL READS	0
HPOOL READS-FAILED	0	HPOOL READS-FAILED	0	HPOOL READS-FAILED	0

DSNESM68	VALUE	DSNESM68	TIMES	DSNESM68	TIME	EVENTS	TIME/EVENT
TYPE	PACKAGE	ELAPSED TIME - CL7	20.318459	LOCK/LATCH	0.002516	62	0.000041
SUCC AUTH CHECK	NO	CPU TIME	17.916493	SYNCHRONOUS I/O	0.181926	131	0.001389
LOCATION	DB2Y	AGENT	17.916493	OTHER READ I/O	0.092543	21	0.004407
COLLECTION ID	DSNESPCS	PAR.TASKS	0.000000	OTHER WRITE I/O	0.113152	2	0.056576
PROGRAM NAME	DSNESM68	SUSPENSION-CL8	0.390256	SERV.TASK SWITCH	0.000118	1	0.000118
CONSISTENCY TOKEN	149EEA901A79FE48	AGENT	0.390256	ARCH.LOG (QUIESCE)	0.000000	0	N/C
ACTIVITY TYPE	N/A	PAR.TASKS	0.000000	ARCHIVE LOG READ	0.000000	0	N/C
SCHEMA NAME	'BLANK'	NOT ACCOUNTED	2.011710	DRAIN LOCK	0.000000	0	N/C
ACTIVITY NAME	'BLANK'			CLAIM RELEASE	0.000000	0	N/C
		CPU SERVICE UNITS	101116	PAGE LATCH	0.000000	0	N/C
SQL STATEMENTS	9	AGENT	101116	SCHED STORED PROC	0.000000	0	N/C
		PAR.TASKS	0	SCHEDULE UDF	0.000000	0	N/C
				NOTIFY MESSAGES	0.000000	0	N/C
		DB2 ENTRY/EXIT	18	GLOBAL CONTENTION	0.000000	0	N/C
				TOTAL CL8 SUSPENS.	0.390256	217	0.001798

1	LOCATION: DB2Y	DB2 PERFORMANCE MONITOR (V6)	PAGE: 1-5
	GROUP: DB2V610Y	ACCOUNTING TRACE - LONG	REQUESTED FROM: NOT SPECIFIED
	MEMBER: DB2Y		TO: NOT SPECIFIED
	SUBSYSTEM: DB2Y		ACTUAL FROM: 12/01/00 18:56:35.41
	DB2 VERSION: V6		TO: 12/01/00 18:56:35.41

TOP FIELD: PROCESSING (TCB) TIME SPENT IN APPLICATION TOP NUMBER REQUESTED: 10

	TIMESTAMP	PRIMAUTH	PLANNAME	VALUE	PAGE
1	12/01/00 18:56:35.418349	PAOLOR3	DSNESPCS	17.936899	1-1

ACCOUNTING TRACE COMPLETE

Appendix C. SQL Performance Analyzer

In this Appendix we list the user parameters and configuration parameters.

C.1 ANLPARM user parameters

When running in batch mode, these parameters are stored in the file specified in the ANLPARM DD statement. They are expected to be modified by most SQL/PA users and tailored to each particular run. In fact, SQL/PA under TSO prompts you and builds this parameter list “on the fly” (meaning at the same time) during execution.

REPORTS aaa: The levels of reporting provided by SQL/PA are defined with the REPORTS parameter. Values of YES or ONE turn on the Cost Summary Report, giving the user a variety of query costs, plus any warning messages generated. This report is always available. Values of EXP or TWO will turn on the Enhanced Explain Report, providing an explanation of each access step. Values of ALL or TRE will trigger the Detail Trace Report, containing information about each step of execution, with SQL/PA's estimates of DB2 resource consumption at the most granular level. REPORTS should be the first parameter in ANLPARM. Its default value is YES, providing Cost Summary Reporting only.

VERSION aaaa: The DB2 version and release number are specified here so that SQL/PA can select the appropriate path length values. The following values (releases of DB2) are supported: V3R1, V4R1, V5R1, V5R2, V6R1, V6R2 and V7R1. VERSION has a default value of V6R1, the latest production release.

STORAGE ddddd: Specify the most prominent DASD storage medium for a application's databases with the STORAGE parameter. This value is used for I/O and elapsed time estimates. The acceptable values include 3390, 3390-1, 3390-2, 3390-3, 3390-9, 6390, 6390-3, 6390-5, 6390-9, 390-1, 7390-2, 7390-3, or 7390-F disk device types, and solid state devices like the 4305, 4380, 6110, 6110-1, 6110-2, 6110-4, 6680-1, 7900, ICEBRG, STK, EMCX, EMCBAR, EMCELT, RAMAC, RAMAC2 and RAMAC3. If several device type are on your system, select the device which holds most of your databases. The STORAGE default value is the 3390-2 disk device.

NEWSTOR newname: Although there are a number of new device types already defined for SQL/PA, including RAMAC, EMC, ICEBERG and other solid state devices, you may still wish to define your own using this parameter. Use NEWSTOR to name a new, user-defined storage device and follow it with the NEWSEEK, NEWROTA and NEWXFER parameters giving the specifications of that device. Either this parameter, or a STORAGE NEWDSK setting, will inform SQL/PA that new disk storage is being defined.

NEWSEEK n.nnnn: Specify the average seek time in seconds for the new storage device.

NEWROTA n.nnnn: Specify the average rotational delay (half a revolution, sometimes called latency) in seconds for the new storage device.

NEWXFER nnnnn.n: Specify the average transfer rate of the device over the channel in kilobytes per second. The default values for all the new disk storage

definitions portray a 3390-2 disk system — Seek =.012 seconds, Rotation =.0071 seconds, and Transfer Rate =4200.0 KB/s. The Detail Trace file (REPORTS ALL) will carry a notation on the Disk Storage definitions.

BUFFHIT 999: The BUFFHIT parameter indicates the relationship of physical to logical I/O in a ratio, expressed as a percentage from 0 to 100. This parameter should answer the question, “What percent of my pages will be found in the buffer pool?” Technically speaking, it is the Buffer Pool Hit ratio. Specify a value for the percentage of getpages that will be satisfied by data in buffer pool without the need for a real disk I/O. BUFFHIT's default value is 000, meaning 100% of the I/O will go to disk.

HPOOLRD 999: The HPOOLRD parameter specifies what percentage of the physical I/O is satisfied by reading from a Hiperpool rather than disk, expressed as a number from 000 to 100. The pages not found in the buffer pool are either read from DASD or, if a Hiperpool exists, possibly read from that expanded storage pool instead. The default value is 000, meaning no Hiperpool.

CONNECT aaaaa: Users may specify the expected application connection to DB2 when the SQL is executed in production. This helps SQL/PA account for overheads caused by the various attach facilities — thread management, application processing, and so forth. The acceptable values are: IFP (IMS Fast Path), WFI (IMS Wait For Input), MPR (IMS Message Processing Region), BMP (IMS Batch Message Program), CICS (Online CICS/VS), SPUFI (Online TSO SPUFI), QMF (Online TSO QMF), DL1 (DL/1 Batch Program), CAF (Batch Call Attach Facility), DSN (Batch TMP with DSN CLIST), RRSF (Recoverable Resource Manager Services Attach Facility), DRDA (remote connections) or NONE (don't include any attach overhead in cost). If not specified, the default value is NONE.

DEGREES aaa: This parameter enables SQL/PA to consider parallel processing for each statement evaluated by DB2. If set to ANY, which is the default, then SQL/PA informs the optimizer that parallel processing is an option. If set to ONE, SQL/PA tells DB2 to ignore any parallel cases, even if the query could benefit from parallel processing (a table space scan against a partitioned table, for instance.)

QUALIFY qqqqqqqq: This optional parameter supplies the default high-level name qualifier for all single-level table and view names. It is used as the CREATOR prefix in front of single-level names. For example, if the SQL contained the following:

```
SELECT FROM ORDER WHERE O_ORDERKEY = 2272;
```

SQL/PA needs to know the high-level qualifier for the ORDER table, currently unspecified in the SQL. If QUALIFY is set to 'PAOLOR4', SQL/PA will interpret the query as follows:

```
SELECT FROM PAOLOR4.ORDER WHERE O_ORDERKEY = 2272;
```

The Parser section of ANLSCAN will automatically make this adjustment for you. If no value is specified for QUALIFY, it defaults to the current user's SQLID.

Synonyms processing is facilitated in SQL/PA by setting the QUALIFY parameter to blanks. SQL/PA recognizes this as a sign that the user does not want SQL/PA

to insert high-level qualifiers in front of unqualified names which, in this case, are synonyms. Because synonym names are, by definition, usable only by their creators and referred to only in unqualified terms, SQL/PA cannot prepare these statements for evaluation under the owner's SQLID and then switch over to the generic plan tables for Explain analysis. Therefore, any users using the synonyms feature of SQL/PA are required to have their own userid. PLAN_TABLE.SQL/PA cannot switch authorization ids midstream because DB2 will not allow it.

PRECISE aaa: The PRECISE parameter provides an optional, extra level of precision to SQL/PA's cost estimates. When PRECISE is set to 'YES or 'ALL', the SQL/PA cost analysis will attempt to replace its internal cost estimate with the DB2 optimizer's own CPU time estimate, drawn from the optional Explain table DSN_STATEMNT_TABLE and the value for PROCMS, or processor milliseconds. Also included in this table is the PROCSU or Service Units estimate, which replaces the value SQL/PA calculates for QUNITS.

PRECISE YES or ALL causes the optimizer's estimate of path length to be used in calculations. However, be aware that this path length is less extensive than SQL/PA's because it only considers the data access portion of the processing. Especially significant are Category B estimates, where the cost estimate is definitely incomplete due to referential integrity, triggers, user-defined functions, and so forth.

The values returned from PRECISE YES or ALL can also be used to govern dynamic queries via the IBM Resource Limit Facility, or RLF. This is accomplished by using the PROCSU (QUNIT) values output from SQL/PA evaluations to populate the RLFASUWARN and RLFASUERR columns in the DSNRLST table. Then, a threshold for warnings and errors can be set in Service Units for the actual execution of the dynamic queries evaluated by SQL/PA.

The default value for PRECISE is NO.

ADVISOR aaa: The Advisor parameter activates additional information on the performance and design of each query, and of the tables and indexes accessed by the query. The information is release-dependent. Warnings and Alerts are always active, even with ADVISOR NO, which is the default. Notes and Recommendations are given when YES is coded, and these can reveal many potential performance tuning opportunities. A value of ALL is ideal for novice or intermediate programmers who need Guidelines as well as Good News when things are done correctly.

The SQL Advisor writes to both the Explain and Detail Trace files, and is totally integrated with the TSO and batch reporting structure.

C.2 ANLCNTL configuration parameters

The following parameters may be specified as ANLCNTL parameters by the SQL/PA installation team or systems support, and are not normally modified by users of SQL/PA.

SUBSYST aaaaaaaa: This parameter specifies the DB2 subsystem name on this machine. The name can be up to eight characters long. Normally, DB2 is called DSN on most processors, but it is possible to run several copies of DB2 on the same machine. If this is the case, you need to define a separate configuration file for each DB2 subsystem. It is not necessary to specify SUBSYST if your DB2 subsystem's name is DSN, the default value.

MIPRATE 9999.999: The MIPS rate (Millions of Instructions Per Second) for the entire processor complex. This is the speed of the physical computer system, including all engines. For example, the three engine IBM 9672-Z37 is estimated at 554.40 MIPS. This parameter must be used in conjunction with the ENGINES parameter. MIPRATE has a default value of 0204.800, or 204.8 million instructions per second, the rating for an IBM 9672-Z17 processor. If your box is logically partitioned into separate systems, that will be accounted for by the LPARENG parameter.

ENGINES 99999: The total number of engines in this processor complex. For example, the IBM 9672-Z87 has eight engines and the HDS Trinium 6008 has six engines. This parameter is used in conjunction with MIPRATE, and both must be specified for SQL/PA to assess the speed of a single processor in the complex. The ENGINES default value is 00001, for a single engine processor.

SRMCONS 9999.999: This parameter reflects the MVS Systems Resource Manager (SRM) constant, CPUMSU, indicating the MVS Service Units per SRM Second available on each CPU engine. It is normally used by MVS to divide up the service available to users under MVS, DB2 users included. This is an optional parameter which replaces the combination of MIPRATE and ENGINES only if they are not both present. In that case, SQL/PA uses an internal formula to derive an estimate for MIPS. This parameter is of particular value between releases of SQL/PA when new processors are announced or installed which are not yet rated by SQL/PA. Often vendors will publish these values for common configurations in their documentation.

This CPUMSU value can also be found in the SMF type 72 record as field RMF72ADJ, or may be retrieved from the target system using the following methodology under TSO:

```
TEST 'SYS1.LINKLIB(IEFBR14)'  
L 10.#+25C?+40 F
```

Divide the result into 16,000,000 to get the CPUMSU per processor. This number will vary for different LPARs, so it may be wise to retrieve it for each system in the complex, especially when a processor complex is logically partitioned into several machines.

SRMCONS has a default value of 9667.670, or 9667.67 service units per SRM second, as does the IBM 9672-Z17.

Remember, if both MIPRATE and ENGINES parameters and the SRMCONS parameter are specified, the MIPRATE and ENGINES values will override the SRMCONS value. Refer to SQL/PA Installation Guide for the MIPS ratings of some CPUs, and formulas for estimating other values of MIPRATE and SRMCONS.

LPARENG 9999.999: This parameter specifies the number of engines from the processor (or fraction thereof) dedicated to the logical partition (or system) running DB2. The value can be specified in terms of whole or partial processors, such as 1.85 or 2.0. This parameter specifically describes when a single physical processor complex is logically divided into multiple system images. This parameter should reflect the logical system image on which the DB2 target host system is presently running, and is used to isolate the maximum single engine speed. This is an optional parameter, and has a default value of 0001.000 (one engine) when not specified.

ESASORT aaa: The ESASORT parameter specifies whether the DB2 hardware assisted sort facility, or its equivalent, is available on this processor. Most IBM processors have this hardware feature installed as standard. It is also a fielda upgrade for many processor types. This optional parameter can be set to YES or NO. The default value is NO. SQL/PA's DB2 sort algorithms will be appropriately influenced by this parameter.

ESACOMP aaa: The ESACOMP parameter specifies whether this machine has a hardware data compression feature. For those table spaces which are data compression candidates, the choice of hardware or software significantly affects the path length of encoding and decoding rows. This optional parameter can be set to YES or NO. The default value is NO, implying software compression.

BUFFERS 99999999: The total number of 4K buffers in the dominant application's 4K buffer pool. This value is used to set the number of pages per block for sequential prefetch and other I/O calculations. The most prominent 4K buffer pool should be reflected here, even if several are used. The BUFFERS default value is 00002000 buffers.

BUFF08K 99999999: The total number of 8K buffers in application's 8K buffer pool, also used for sequential prefetch and I/O calculations on the 8K page operations. The BUFF08K default value is 00000500 buffers.

BUFF16K 99999999: The total number of 16K buffers in application's 16K buffer pool, also used for sequential prefetch and I/O calculations on the 16K page operations. The BUFF16K default value is 00000250 buffers.

BUFF32K 99999999: The total number of 32K buffers in application's 32K buffer pool, also used for sequential prefetch and I/O calculations on the 32K page operations. The BUFF32K default value is 00000100 buffers.

SORTBUF 99999999: The total number of buffers in the sort buffer pool, servicing the temporary storage database, used for sort/merge pass estimates and sort I/O calculations. The SORTBUF default value is 00002000 buffers.

DSGROUP nn: The number of members in the Data Sharing Group, from 0 to 32. If this system participates in data sharing, specify the group size. The default is 0, no data sharing.

DATASHR nnn: The average percentage of the DB2 workload that participates in Data Sharing, from 0 to 100 percent. The default is 0, no data sharing. SQL/PA uses DSGROUP and DATASHR to adjust the overall processing power of the system to reflect the overhead incurred by data sharing operations, giving a more accurate cost assessment.

ANLKEYS nnnnn: This special control parameter is used to modify SQL/PA's cost analysis. A value of 10000 tells SQL/PA to use only Class 1 CPU Times in its estimates, good for validation against an online monitor. A value of 02000 tells SQL/PA to eliminate all overheads from its calculations, ideal for building capacity planning models that will consist of running the application multiple times. These values can be combined, as 12000, to gain both effects at once. A value of 01000 tells SQL/PA to process the Parser only, and may be helpful in isolating an SQL parsing problem, if requested by Technical Support.

Warning: Do NOT attempt to run SQL/PA with any other values in this parameter. There are additional, undocumented settings which will give inaccurate results to the uninformed user.

CPUTIME 99999: The maximum CPU time, in seconds, that will be permitted before the SQL statement is flagged as exceeding the installation's CPU limit by SQL/PA. For example, a value of 120 would be equivalent to 2 minutes of CPU time (2 minutes * 60 seconds/minute = 120 seconds). The maximum value that can be specified is 86400, or 24 hours. When unspecified, SQL/PA's internal default value is -1, setting no limit at all.

ELAPSED 99999: The maximum elapsed time, in seconds, that will be permitted before the SQL statement is flagged as exceeding the installation's elapsed time limit by SQL/PA. For example, a value of 300 would be equivalent to 5 minutes (5 minutes * 60 seconds/minute = 300 seconds). The maximum value that can be specified is 86400, or 24 hours. When unspecified, the SQL/PA internal default value is -1, setting no limit.

IOCALLS 99999999: The maximum physical I/O calls that will be permitted before the SQL statement is flagged as exceeding the installation's physical I/O limit by SQL/PA. For example, if you want to restrict queries to 5000 physical I/Os, specify a value of 5000. The maximum value that can be specified is 99999999. The buffer hit ratio parameter, BUFFHIT, has an impact on this IOCALLS limit. Only physical I/O, not the total getpages or logical I/O, count towards this limit. If unspecified, SQL/PA's internal default value is -1, setting no limit to the number of physical I/O issued.

COSTING 99999999: The maximum monetary limit for any query, before SQL/PA flags it as costing an excessive amount. The amount is specified in the national currency. (See MONEYIS and CURRSYM at the end of this section.) The maximum value that can be specified is 999999999. A value of 0, the default, turns off the limit warning.

COSTQUN 99999999: The maximum QUNITS for any query, before SQL/PA flags it as exceeding the QUNITS limit, a CPU service derivative. The maximum value that can be specified is 999999999. A value of 0, the default, turns off the limit warning.

A reminder: CPUTIME, ELAPSED, IOCALLS, COSTING, and COSTQUN are installation-defined limits for these resources. SQL/PA will flag all activity that

exceeds the limits set by these parameters. If any are not specified, SQL/PA will assume that there is no limit for that category.

DYNAMIC aaa: The parameter controls the use of the option to 'cache' dynamic SQL statements in SQL/PA's estimates for this configuration. This option is available with DB2 Version 5 or higher, and it avoids most of the overhead of preparing dynamic SQL for subsequent executions. The values are YES (the system uses Dynamic Statement Caching) or NO (the default).

CPUCOST 9999.999: The cost of one hour of CPU time, in the national currency, for this configuration. Used by SQL/PA for charge back and financial cost determination. SQL/PA's default value is 0000.000, providing no cost figures for CPU time.

IOSCOST 9999.999: The cost of 1000 I/O calls, in the national currency, for this configuration. Used by SQL/PA for charge back and financial cost determination. SQL/PA's default value is 0000.000, providing no cost figures for physical I/Os issued.

TIMCOST 9999.999: The cost of one hour of connect time, in the national currency, for this configuration. Used by SQL/PA for charge back and financial cost determination. SQL/PA's default value is 0000.000, providing no cost figures for elapsed connect time.

MONEYIS aaaaaaaaa: The national currency in descriptive form, such as DOLLARS, LIRE, FRANCS, POUNDS, DRACHMA, KRONES, PESOS, and the like. The default value is DOLLARS.

CURRSYM a: The currency symbol, as a single character, representing the monetary unit specified by the MONEYIS parameter. The default value is the dollar sign, \$.

Note: CPUCOST, IOSCOST and TIMCOST will be used to calculate the total monetary value of a transaction, based on your installation's charge back costing guidelines. One or more of these parameters may be set to zero or left out, changing the cost algorithm accordingly. For example, if your installation does not charge for connect time, the costing would reflect only CPU and I/O charges.

Appendix D. Using the additional material

This redbook also contains additional Web material. See the appropriate section below for instructions on using or downloading the material.

D.1 Locating the additional material on the Internet

The CD-ROM, diskette, or Web material associated with this redbook is also available in softcopy on the Internet from the IBM Redbooks Web server. Point your Web browser to:

<ftp://www.redbooks.ibm.com/redbooks/SG246139>

Alternatively, you can go to the IBM Redbooks Web site at:

ibm.com/redbooks

Select the **Additional materials** and open the directory that corresponds with the redbook form number.

D.2 Using the Web material

The additional Web material that accompanies this redbook includes the following:

<i>File name</i>	<i>Description</i>
6139_P1_INTRO.zip	Zipped Presentation, overview of DB2 Tools
6139_P2_BM.zip	Zipped Presentation of Bind Manager
6139_P3_QM.zip	Zipped Presentation of Query Manager
6139_P4_PA.zip	Zipped Presentation of SQL Performance Analyzer
6139_ALL.zip	Zipped Presentation of all of the above

D.2.1 System requirements for downloading the Web material

The following system configuration is recommended for downloading all the additional Web material.

Hard disk space:	3.6 MB minimum
Operating System:	Windows NT or 95 or 2000
Processor:	Intel 386 or higher
Memory:	16 MB

D.2.2 How to use the Web material

Create a subdirectory (folder) on your workstation and copy the contents of the Web material into this folder. Then unzip the material into the folder.

Appendix E. Special notices

This publication is intended to help managers and professionals understand and evaluate of the functions introduced by three new IBM Data Management Tools for DB2 for OS/390 and z/OS. This redbook is in the format of a presentation guide and as such can be easily utilized to propagate the information. The information in this publication is not intended as the specification of any programming interfaces that are provided by IBM DB2 Bind Manager, Program Number 5655-D38, IBM DB2 Query Monitor, Program Number 5655-E67, and IBM DB2 SQL Performance Analyzer, Program Number 5697-F57 for use with DB2 UDB Server for OS/390 and z/OS. See the PUBLICATIONS section of the IBM Programming Announcement for IBM DB2 Bind Manager, Program Number 5655-D38, IBM DB2 Query Monitor, Program Number 5655-E67, and IBM DB2 SQL Performance Analyzer, Program Number 5697-F57, for more information about what publications are considered to be product documentation.

References in this publication to IBM products, programs or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM product, program, or service is not intended to state or imply that only IBM's product, program, or service may be used. Any functionally equivalent program that does not infringe any of IBM's intellectual property rights may be used instead of the IBM product, program or service.

Information in this book was developed in conjunction with use of the equipment specified, and is limited in application to those specific hardware and software products and levels.

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


Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact IBM Corporation, Dept. 600A, Mail Drop 1329, Somers, NY 10589 USA.


Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The information contained in this document has not been submitted to any formal IBM test and is distributed AS IS. The use of this information or the implementation of any of these techniques is a customer responsibility and depends on the customer's ability to evaluate and integrate them into the customer's operational environment. While each item may have been reviewed by IBM for accuracy in a specific situation, there is no guarantee that the same or similar results will be obtained elsewhere. Customers attempting to adapt these techniques to their own environments do so at their own risk.

Any pointers in this publication to external Web sites are provided for convenience only and do not in any manner serve as an endorsement of these Web sites.

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

DB2
e (logo)® 
IBM ®
OS/390

Redbooks
Redbooks Logo 
zSeries

The following terms are trademarks of other companies:

Tivoli, Manage. Anything. Anywhere., The Power To Manage., Anything. Anywhere., TME, NetView, Cross-Site, Tivoli Ready, Tivoli Certified, Planet Tivoli, and Tivoli Enterprise are trademarks or registered trademarks of Tivoli Systems Inc., an IBM company, in the United States, other countries, or both. In Denmark, Tivoli is a trademark licensed from Københavns Sommer - Tivoli A/S.

C-bus is a trademark of Corollary, Inc. in the United States and/or other countries.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States and/or other countries.

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

PC Direct is a trademark of Ziff Communications Company in the United States and/or other countries and is used by IBM Corporation under license.

ActionMedia, LANDesk, MMX, Pentium and ProShare are trademarks of Intel Corporation in the United States and/or other countries.

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

SET, SET Secure Electronic Transaction, and the SET Logo are trademarks owned by SET Secure Electronic Transaction LLC.

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

Appendix F. Related publications

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

F.1 IBM Redbooks

For information on ordering these publications see “How to get IBM Redbooks” on page 241.

- *DB2 UDB for OS/390 Version 6 Management Tools Package*, SG24-5759
- *Developing Cross-Platform DB2 Stored Procedures: SQL Procedures and DB2 Stored Procedure Builder*, SC24-5485

F.2 IBM Redbooks collections

Redbooks are also available on the following CD-ROMs. Click the CD-ROMs button at ibm.com/redbooks for information about all the CD-ROMs offered, updates and formats.

CD-ROM Title	Collection Kit Number
IBM System/390 Redbooks Collection	SK2T-2177
IBM Networking Redbooks Collection	SK2T-6022
IBM Transaction Processing and Data Management Redbooks Collection	SK2T-8038
IBM Lotus Redbooks Collection	SK2T-8039
Tivoli Redbooks Collection	SK2T-8044
IBM AS/400 Redbooks Collection	SK2T-2849
IBM Netfinity Hardware and Software Redbooks Collection	SK2T-8046
IBM RS/6000 Redbooks Collection	SK2T-8043
IBM Application Development Redbooks Collection	SK2T-8037
IBM Enterprise Storage and Systems Management Solutions	SK3T-3694

F.3 Other resources

These publications are relevant as further information sources:

- *DB2 SQL Performance Analyzer for OS/390 Installation Guide Version 1* , SC27-1003
- *DB2 SQL Performance Analyzer for OS/390 User's Guide Version 1* , SC27-1002
- *DB2 Query Monitor for OS/390 User's Guide Version 1 Release 1* , SC27-0968
- *DB2 Bind Manager for OS/390 User's Guide Version 1* , SC27-0899
- *DB2 Recovery Manager Message and Codes Version 1 Release 1* , SC27-1114
- *DB2 Universal Database for OS/390 Administration Guide Volume 1 Version 6* , SC26-9003
- *DB2 Performance Monitor for OS/390 Online User's Guide Version 6* , SC26-9168

F.4 Referenced Web sites

These Web sites are also relevant as further information sources:

- <http://ibm.com/software/data/db2imstools/> IBM Database Tools for OS/390
- <http://ibm.com/software/data/db2/os390/> DB2 for OS/390

How to get IBM Redbooks

This section explains how both customers and IBM employees can find out about IBM Redbooks, redpieces, and CD-ROMs. A form for ordering books and CD-ROMs by fax or e-mail is also provided.

- **Redbooks Web Site** ibm.com/redbooks

Search for, view, download, or order hardcopy/CD-ROM Redbooks from the Redbooks Web site. Also read redpieces and download additional materials (code samples or diskette/CD-ROM images) from this Redbooks site.

Redpieces are Redbooks in progress; not all Redbooks become redpieces and sometimes just a few chapters will be published this way. The intent is to get the information out much quicker than the formal publishing process allows.

- **E-mail Orders**

Send orders by e-mail including information from the IBM Redbooks fax order form to:

In United States or Canada	e-mail address pubscan@us.ibm.com
Outside North America	Contact information is in the "How to Order" section at this site: http://www.elink.ibm.link.ibm.com/pbl/pbl

- **Telephone Orders**

United States (toll free)	1-800-879-2755
Canada (toll free)	1-800-IBM-4YOU
Outside North America	Country coordinator phone number is in the "How to Order" section at this site: http://www.elink.ibm.link.ibm.com/pbl/pbl

- **Fax Orders**

United States (toll free)	1-800-445-9269
Canada	1-403-267-4455
Outside North America	Fax phone number is in the "How to Order" section at this site: http://www.elink.ibm.link.ibm.com/pbl/pbl

This information was current at the time of publication, but is continually subject to change. The latest information may be found at the Redbooks Web site.

IBM Intranet for Employees

IBM employees may register for information on workshops, residencies, and Redbooks by accessing the IBM Intranet Web site at <http://w3.itso.ibm.com/> and clicking the ITSO Mailing List button. Look in the Materials repository for workshops, presentations, papers, and Web pages developed and written by the ITSO technical professionals; click the Additional Materials button. Employees may access MyNews at <http://w3.ibm.com/> for redbook, residency, and workshop announcements.

IBM Redbooks fax order form

Please send me the following:

Title	Order Number	Quantity
-------	--------------	----------

[illegible]

First name	Last name
------------	-----------

Company	Revenue	Profit	Assets	Liabilities	Equity
Company A	100	20	50	30	20
Company B	150	30	75	45	30
Company C	200	40	100	60	40
Company D	250	50	125	75	50
Company E	300	60	150	90	60
Company F	350	70	175	105	70
Company G	400	80	200	120	80
Company H	450	90	225	135	90
Company I	500	100	250	150	100
Company J	550	110	275	165	110
Company K	600	120	300	180	120
Company L	650	130	325	195	130
Company M	700	140	350	210	140
Company N	750	150	375	225	150
Company O	800	160	400	240	160
Company P	850	170	425	255	170
Company Q	900	180	450	270	180
Company R	950	190	475	285	190
Company S	1000	200	500	300	200

Address _____

City	Postal code	Country
------	-------------	---------

Telephone number	Telefax number	VAT number
------------------	----------------	------------

☐ Invoice to customer number☐ Credit card number

Credit card expiration date	Card issued to	Signature
-----------------------------	----------------	-----------

We accept American Express, Diners, Eurocard, Master Card, and Visa. Payment by credit card not available in all countries. Signature mandatory for credit card payment.

Abbreviations and acronyms

AIX	Advanced Interactive eXecutive from IBM	DNS	domain name server
APAR	authorized program analysis report	DRDA	distributed relational database architecture
ARM	automatic restart manager	DTT	declared temporary tables
ASCII	American National Standard Code for Information Interchange	EA	extended addressability
		EBCDIC	extended binary coded decimal interchange code
BLOB	binary large objects	ECS	enhanced catalog sharing
CAF	call attach facility	ECSA	extended common storage area
CCSID	coded character set identifier	EDM	environment descriptor management
CCA	client configuration assistant	ERP	enterprise resource planning
CFCC	coupling facility control code	ESA	Enterprise Systems Architecture
CTT	created temporary table	FDT	functional track directory
CEC	central electronics complex	FTP	File Transfer Program
CD	compact disk	GB	gigabyte (1,073,741,824 bytes)
CF	coupling facility	GBP	group buffer pool
CFRM	coupling facility resource management	GDG	generation data group
CICS	Customer Information Control System	GRS	global resource serialization
CLI	call level interface	GUI	graphical user interface
CLP	command line processor	HPJ	high performance Java
CPU	central processing unit	IBM	International Business Machines Corporation
CRM	customer relationship management	ICF	integrated catalog facility
CSA	common storage area	ICF	integrated coupling facility
DASD	direct access storage device	ICMF	internal coupling migration facility
DB2 PM	DB2 Performance Monitor	IFCID	instrumentation facility component identifier
DBAT	database access thread	IFI	instrumentation facility interface
DBD	database descriptor	IMS	Information Management System
DBID	database identifier	IRLM	internal resource lock manager
DBMS	Database Management Software	ISPF	interactive system productivity facility
DBRM	database request module	ISV	independent software vendor
DCL	data control language	I/O	input/output
DDCS	distributed database connection services	ITSO	International Technical Support Organization
DDF	distributed data facility		
DDL	data definition language		
DLL	dynamic load library manipulation language		
DML	data manipulation language		

IVP	installation verification process	SLA	service level agreement
JDBC	Java Database Connectivity	SMIT	System Management Interface Tool
JFS	journaled file systems	SP	stored procedure
JVM	Java Virtual Machine	SQL	structured query language
KB	kilobyte (1,024 bytes)	SQLJ	SQL Java
LAT	Log Analysis Tool	SRB	system resource block
LOB	large object	TCB	task control block
LPL	logical page list	TSO	time sharing option
LPAR	logically partitioned mode	VSAM	virtual storage access method
LRECL	logical record length	WLM	workload manager
LRSN	log record sequence number		
LVM	logical volume manager		
MB	megabyte (1,048,576 bytes)		
MVS	multiple virtual storage, often used for all operating systems on S/390 platform		
OBD	object descriptor in DBD		
OBID	object identifier		
ODBC	Open Data Base Connectivity		
OS/390	Operating System/390		
PAV	parallel access volume		
PDS	partitioned data set		
PM	Performance Monitor (DB2 Performance Monitor for OS/390)		
PSID	pageset identifier		
PSP	preventive service planning		
PTF	program temporary fix		
PUNC	possibly uncommitted		
QM	Query Monitor		
QMF	Query Management Facility		
RACF	Resource Access Control Facility		
RAD	rapid application development		
RBA	relative byte address		
RECFM	record format		
RID	record identifier		
RRS	resource recovery services		
RRSAF	resource recovery services attach facility		
RS	read stability		
RR	repeatable read		
SDK	software developers kit		

Index

Numerics

100 161
101 161
102 161
5697-G52 20

A

access path 57, 65, 210
accounting trace 76
ANL4QMF 167, 168
ANLAUTHID 172
ANLCNTL 175, 177
ANLCOST 167, 168, 199
ANLGOV1 169, 177
ANLINSUE 172
ANLKEYS 180, 214
ANLMODA 169
ANLPARM 175, 177, 194
ANLPROC 167, 169, 192
ANLSCAN 167, 198
ANLSTP 192, 200
ANLTMSTMP 172
ANLUSERID 172
ANY 67
APF 49, 86
application management 9
application profile 80, 98, 106

B

Bind required 51
BND002I 51, 53
BND003I 51
BND098S 51
BND203I 63
BND204I 63
BNDAVB 49, 50
BNDC001 49
BUFFERS 214
BUFFHIT 213

C

CAF 169
categories of filters 98
Class 1 134
Class 2 135
Class 3 135
CLIM 184
Command activity 144, 146, 147
compare JCL 62
concatenation 55
CONNECT 213
cost estimates 213
COSTS 184
CPUCOST 214
CPUTIME 184

CQMCLIST 87
CQMDB 89
CQMPPARMS 86
CUMULATIVE 190
Current Activity 81, 129, 130, 131
Customer Relationship Management 5

D

Data Management categories 7
Data Management Tools
 why 6
Database administration 8
DATASHR 214
DB2 Administration Tool 14
DB2 Archive Log Compression Tool 24
DB2 Automation Tool 23
DB2 Bind Manager 40, 45
 components 47
 final solution 72
 flow 50
 integration 71
 minimal impact 69
 objectives 46
 precompiler report 51
 structure and configuration 49
 usage and considerations 69
DB2 catalog 46
DB2 Change Accumulation Tool 39
DB2 DataPropagator 32
DB2 Estimator 10
DB2 Forms 20
DB2 High Performance Unload 17
DB2 Installer 10
DB2 Log Analysis Tool 19
DB2 Object Comparison Tool 25
DB2 Object Restore 16
DB2 Performance Monitor 27
DB2 PM 132, 162
 long accounting report sample 223
DB2 PM long accounting report 133
DB2 PM Online Monitor 132
DB2 Query Monitor 31
 activity screen 112
 case studies 128
 checklist 107
 commands 113
 commands 'U' and 'L' 79
 Control File 88
 current and past activity 112
 data columns 114
 DB2 objects 90
 DB2 subsystem monitoring 94
 exception profile processing 105
 exception profiles 104
 filtering parameters 101
 function keys 115
 hints and tips 151

- installation 84
- loading intervals 140
- main menu options 92
- major functions 78
- Monitoring and Application Profiles 98
- open cursor 126
- operational scenarios 109
- operations 89, 92
- overview 77
- past activity 81
- past data 90
- processing overview 89
- profile creation 97
- Profile summary 97
- purpose 76
- running and managing 110
- sample output 221
- start up 85
- summary of features 149
- terminology 80
- thread navigation 116
- unloading intervals 138
- updating DB2 subsystem information 95
- usage and considerations 109
- DB2 Query Monitor and DB2 Performance Monitor 132, 134
- DB2 Recovery Manager 37
- DB2 Row Archive Manager 35
- DB2 SQL Performance Analyzer 29
 - Batch Interface 198
 - considerations 212
 - cost summary report 185
 - detail trace report 188
 - generic PLAN_TABLE search 172
 - input and output flow 174
 - limitations 215
 - objective 166
 - overview 156
 - reporting functions 182
 - stored procedure interface 200
 - structure and configuration 165
 - usage 206
 - usage and considerations 205
- DB2 Stored Procedure Builder 202
- DB2 tools
 - introduction 11
- DB2 UDB Control Center 10
- DB2 Utilities and Binds 148
- DB2 Visual Explain 10
- DB2 Web Query 41
- DB2PARMS 88
- DBM1 160
- DBRM 46
- DBRM Checker 45
 - changes for 71, 72
 - JCL 55
 - output 56
 - purpose 54
- DBRM detail 120, 121
- DBRM IN DD 65
- DECOMPR 189
- DEGREES 213
- display host variables 102
- display SQL 101
- display threads 101
- DM 160
- DSGROUP 214
- DSN_STATEMNT_TABLE 168, 180
- DSN3@ATH 170
- DSNBIND 148
- DSNHPC 50
- DSNUTIL 148
- DSQBFSQL 119
- DSQUEGV1 168, 169
- DYNAMIC 214
- Dynamic SQL caching 124

E

- ELAPSED 184
- ELIM 184
- ENGINES 214
- Enhanced Explain 209
- Enhanced Explain report 187
- Enterprise Resource Planning 5
- ESACOMP 214
- ESASORT 214
- exception profile 79, 80, 106
- EXPLAIN 59
- Explain 163
- EXPROF 86

F

- features 10
- FIND 114
- flag threads 102

G

- GDG 79, 91
- Generic PLAN_TABLE 170
- Good News 157
- Governor 168
- Guidelines and Good News 179

H

- HELP 114
- HISTORY 141
- HPOOLRD 213

I

- ILIM 184
- IMS 37
- interval 81
- IOCUNT 184
- IOSCOST 214
- ISPF 53, 78

L

Linux/390 5
Lotus Domino 5
LPAR 168
LPARENG 214

M

MIPRATE 214
MIPS 188
MIXOPSEQ 190
MLIM 184
MONITOR 86, 95
monitoring agent 79, 80
monitoring profile 80, 106
MSTR 145

N

NEWROTA 213
NEWSEEK 213
NEWSTOR 213
NEWXFER 213
No Bind required 52

O

OPTIMIZE FOR n ROW 189
Option 1 92, 96
Option 2 92, 96
Option 3 92, 106
Option 4 92
Option 5 93, 97, 106
Option S 93
OS/390 70
OTHER O/H 189

P

Past activity 136, 142
Path Checker 45
 basic reporting 60
 changes for 71, 72
 Compare 62
 compare detailed 64
 compare short 63
 detailed reporting 61
 JCL 58
 purpose 57
 reporting JCL 59
 test JCL 65
 test mode detailed report 67
 test mode report short 66
PeopleSoft 5
Performance management 8
Plan detail 119
PLAN filter 100
PLAN_TABLE 57, 170
PRECISE 180, 214
precompiler 70
prediction 156

PRINTX 113, 133
PROCESSES 189

Q

QLIM 184
QLIMIT 184, 207
QMF 91, 118, 156, 168
QMF Governor 163
QMF HPO 163
QTRACE 212
Query Limit 184
Query Monitor see also DB2 Query Monitor 75
QUERYNO 51, 184
QUNIT 180
QUNITS 158, 184

R

RDS 160
REBIND 46
Recovery and replication 8
redbook
 objectives 3
 parts 3
Registry table 172
reloading 79
REPORTS 182
RETCODE 184

S

SANLLOAD 169
SANLPARM 177
SAP R/3 5
SBNDJCL 49
SBQRY 190
Service Units 180
SET CURRENT SQLID 171
SHORT 59
Siebel 5
SMF 53, 160
 records 161
SORT 113
SORTBUF 214
SPUFI 61, 91
SQL Advisor 156, 157, 179
SQL Performance Analyzer see also DB2 SQL Performance Analyzer 155
SQL statement text 122
SRMCONS 214
STORAGE 213
Stored Procedure Builder 10, 200
SUBSYS 86

T

Thread activity 117, 118
Thread/Plan/DBRM activity 127
threads panels 116
threshold 208
TIMCOST 214

Token verification 53
tools at a glance 12

U

UNIX System Services 5
unloading 79

V

VCAT 54
VERSION 213
VSAM 54

W

WebSphere 5
wildcard 99

IBM Redbooks review

Your feedback is valued by the Redbook authors. In particular we are interested in situations where a Redbook "made the difference" in a task or problem you encountered. Using one of the following methods, **please review the Redbook, addressing value, subject matter, structure, depth and quality as appropriate.**

- Use the online **Contact us** review redbook form found at ibm.com/redbooks
- Fax this form to: USA International Access Code + 1 845 432 8264
- Send your comments in an Internet note to redbook@us.ibm.com

Document Number	SG24-6139-00
Redbook Title	New Tools for DB2 for OS/390 and z/OS Presentation Guide
Review	<div></div> <div></div> <div></div> <div></div> <div></div> <div></div>
What other subjects would you like to see IBM Redbooks address?	<div></div> <div></div> <div></div>
Please rate your overall satisfaction:	<input type="radio"/> Very Good <input type="radio"/> Good <input type="radio"/> Average <input type="radio"/> Poor
Please identify yourself as belonging to one of the following groups:	<input type="radio"/> Customer <input type="radio"/> Business Partner <input type="radio"/> Solution Developer <input type="radio"/> IBM, Lotus or Tivoli Employee <input type="radio"/> None of the above
Your email address: The data you provide here may be used to provide you with information from IBM or our business partners about our products, services or activities.	<div></div> <div><input type="radio"/> Please do not use the information collected here for future marketing or promotional contacts or other communications beyond the scope of this transaction.</div>
Questions about IBM's privacy policy?	The following link explains how we protect your personal information. ibm.com/privacy/yourprivacy/



New Tools for DB2 for OS/390 and z/OS Presentation Guide

(0.5" spine)
0.475" <-> 0.873"
250 <-> 459 pages



New Tools for DB2 for OS/390 and z/OS Presentation Guide

**IBM DB2 Bind
Manager, Query
Monitor, and SQL
Performance Analyzer**

**Data Management
tools for DB2
databases and
applications**

**Major features,
structure and usage
of DB2's new tools**

IBM's recent new focus on tools has brought enhancements to current DB2 UDB Server for OS/390 ancillary products and to the introduction of new tools.

A project was conducted in order to provide a redbook, in the format of a presentation guide, where the newest of these tools, namely DB2 Bind Manager, DB2 Query Monitor, and DB2 SQL Performance Analyzer are described in detail with references to real life scenarios.

This IBM Redbook is primarily a skills transfer vehicle: it will help you understand the functions provided by the three tools, and it can be used as a basis for preparing presentations.

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

SG24-6139-00

ISBN 0738419699