

Capacity Planning for Business Intelligence Applications: Approaches and Methodologies

Understand the resource consumption
of BI applications

Learn methodologies for the
S/390 BI infrastructure

Use our capacity
planning worksheets



Seungrahn Hahn
M.H. Ann Jackson
Bruce Kabath
Ashraf Kamel
Caryn Meyers
Ana Rivera Matias
Merrilee Osterhoudt
Gary Robinson

ibm.com/redbooks

Redbooks



International Technical Support Organization

**Capacity Planning for
Business Intelligence Applications:
Approaches and Methodologies**

November 2000

Take Note!

Before using this information and the product it supports, be sure to read the general information in Appendix F, "Special notices" on page 179.

First Edition (November 2000)

This edition applies to DB2 UDB for OS/390 Version 6, 5645-DB2, for use with OS/390 V2.7.

This document created or updated on December 12, 2000.

Comments may be addressed to:
IBM Corporation, International Technical Support Organization
Dept. HYJ Mail Station P099
2455 South Road
Poughkeepsie, NY 12601-5400

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 2000. 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

Figuresix
Tablesxi
Prefacexiii
The team that wrote this redbookxiii
Comments welcomexv
Chapter 1. Introduction to capacity planning	1
1.1 What is Business Intelligence (BI)	1
1.1.1 Business Intelligence data stores	2
1.1.2 Business Intelligence processing	3
1.2 What is the Business Intelligence infrastructure	6
1.2.1 The Business Intelligence system architectures	7
1.2.2 The S/390 Business Intelligence infrastructure	8
1.3 Capacity planning for the BI infrastructure	9
1.3.1 What is capacity planning	9
1.3.2 Why capacity planning is essential for the BI infrastructure	9
1.3.3 Capacity planning for traditional workloads	10
1.3.4 Capacity planning for BI workloads	11
1.4 Information needed for capacity planning	13
1.5 The BI infrastructure capacity planning team	13
1.6 Capacity planning methods	15
1.7 Capacity planning tools	15
1.8 A word on resource sizing	16
1.9 Overview of remaining chapters	16
1.10 Summary	17
Chapter 2. Resource usage attributes of a query workload	19
2.1 Query parallelism	19
2.1.1 Types of query parallelism	20
2.1.2 Partitioning for parallelism	21
2.1.3 Query parallelism and I/O activity	23
2.1.4 Concurrency and locking	23
2.2 Response time consistency	24
2.3 Workload Manager (WLM)	25
2.3.1 Workload management issues	25
2.4 Usage patterns	26
2.4.1 Increased usage	26
2.4.2 Efficient usage	27
2.4.3 Access refinement by users	27

2.4.4 Statistics	27
2.5 Query creation	29
2.6 User population	29
2.7 Data volume	30
2.7.1 Choice of model	31
2.8 Summarization of data	32
2.8.1 Summary table design	32
2.8.2 Shifting CPU cycles	33
2.9 Tuning	33
2.9.1 Awareness of table size	34
2.9.2 Indexing	34
2.10 Summary	35
2.11 Checklist	35
Chapter 3. Data maintenance	37
3.1 Data population	37
3.2 Database utilities	40
3.3 Archiving/purging old data	42
3.4 Summary tables	45
3.5 Factors influencing the maintenance workload	46
3.6 Summary	49
Chapter 4. Characterization of existing workloads	51
4.1 Capacity planning for a business intelligence workload	51
4.2 Key characteristics of a workload	53
4.2.1 Workload profiling	53
4.2.2 Profiling system usage of a workload	53
4.2.3 Profiling data usage	58
4.2.4 User profile	59
4.2.5 Profiling query by user population	59
4.2.6 Maintenance processing	63
4.3 Summary of using the worksheet	64
Chapter 5. Establishing current system usage	65
5.1 System information sources	65
5.1.1 System Management Facility (SMF)	65
5.1.2 Resource Management Facility (RMF)	66
5.1.3 DB2 Performance Monitor	66
5.1.4 How performance data is generated	67
5.1.5 How DB2 PM reports the data	68
5.2 Developing a workload profile	68
5.3 Worksheet 1: Application profile	70
5.4 Worksheet 2: System characterization	71
5.4.1 General information	71

5.4.2	System utilization	72
5.5	Worksheet 3: Workload characterization	73
5.5.1	Data profile	73
5.5.2	User profile	73
5.5.3	Query profile	73
5.5.4	Additional information	75
5.6	Worksheet 4: Maintenance processing requirements	82
5.6.1	Table 4-1: Operational profile	84
5.6.2	Table 4-2: Critical path for batch jobs	84
5.7	Summary	85
Chapter 6. Growing existing workloads		87
6.1	Capacity planning concepts	87
6.1.1	Balanced system	87
6.1.2	Capture ratio	88
6.1.3	Relative I/O content (RIOC)	88
6.1.4	Understanding the workload	89
6.2	Data growth	90
6.2.1	Considerations for data growth	90
6.3	User query growth	93
6.3.1	Understanding user growth	93
6.4	Worksheet 5: User growth projections	94
6.4.1	Estimating the data growth factor for the query workload	95
6.4.2	Calculating CPU growth for query workload	97
6.5	Maintenance workload growth: Worksheet 4	100
6.6	Summary	101
Chapter 7. Sizing methodologies for new applications		103
7.1	Fact gathering and iterative estimation	103
7.2	Planning methodologies	104
7.3	Method 1: Rule of thumb extrapolation for the query workload	104
7.3.1	Industry rules of thumb	105
7.3.2	Rules of thumb: custom query profiles	106
7.3.3	Query ROT technique comparison	109
7.3.4	Method 2: formulas/guidelines for batch workloads	110
7.3.5	Initial load estimates	117
7.4	Method 3: Modeling with DB2 Estimator	117
7.4.1	Method 4: Proofs of Concept (POC)	119
7.4.2	Functional Proofs of Concept	119
7.4.3	Performance Proofs of Concept	120
7.5	Summary	121
Chapter 8. Estimating model size for IBM DB2 OLAP Server		123
8.1	Inflation	123

8.2 Sparsity	126
8.3 Block density	128
8.4 Gathering information	129
8.4.1 Proof of concept	130
8.5 Using the information	133
8.6 Summary	134
Chapter 9. Sizing Intelligent Miner for Data for performance	137
9.1 Factors influencing Intelligent Miner performance	137
9.1.1 Location of input data	137
9.1.2 Data volume and variables	137
9.1.3 Mining techniques	137
9.1.4 Parallelism	137
9.1.5 Database design	138
9.2 Sample response for a selected system configuration	138
9.2.1 Clustering	139
9.2.2 Sequential patterns	139
9.2.3 Associations	140
9.2.4 Statistical Analyses: Linear Regression	142
9.3 Intelligent Miner performance recommendations	143
9.4 Data considerations for data mining	145
Appendix A. Sample worksheets	147
Appendix B. Data models	161
B.1 Overview	161
B.2 Entity-relationship (ER)	161
B.3 Star schema	161
B.4 Snowflake schema	162
Appendix C. Data Propagator Relational (DB2 DPropR)	165
C.1 Architecture and components	165
C.2 Storage requirements	166
C.2.1 Log impact	166
C.2.2 New tables: CD tables and UOW tables	167
C.2.3 Target tables	167
C.2.4 Apply spill file	168
C.3 Capacity requirements	168
C.3.1 Capture	168
C.3.2 Apply	169
C.3.3 Administration	169
C.3.4 Network capacity	169
C.3.5 Throughput capacity	169
C.4 DPropR workload	170

Appendix D. Capture ratios	171
Appendix E. Using the additional material	177
E.1 Locating the additional material on the Internet	177
Appendix F. Special notices	179
Appendix G. Related publications	183
G.1 IBM Redbooks	183
G.2 IBM Redbooks collections	183
G.3 Other resources	184
G.4 Referenced Web sites	184
How to get IBM Redbooks	185
IBM Redbooks fax order form	186
Index	187
IBM Redbooks review	191

Figures

1. Data warehouse and data mart	3
2. Data-to-information process.	4
3. Data analysis techniques.	6
4. Application deployment on a BI system.	8
5. Automatic data archiving	44
6. Data updating design alternatives	48
7. Business intelligence physical solution architectures	52
8. Typical Peak to Average Ratios, and Latent Demand for OLTP	55
9. Various CPU utilization profiles of BI workloads	57
10. Composite query profile of the total query workload	60
11. Query segmentation by business units	63
12. Sample statistics and accounting commands	79
13. Sample SQL for query profile worksheet.	81
14. Understanding the Maintenance Workload in the batch Window	83
15. Resource utilization by workload	89
16. Historical data growth	91
17. Custom query profile - Table 6-1	107
18. Batch Rule of Thumb: Performance Estimate - Table 6-3.	113
19. Summary of new application sizing methods.	121
20. Cube storage requirement and calculation time versus sparsity	128
21. Way to define slices through the cube.	131
22. Sample cube for hardware estimate	133
23. Result of clustering	139
24. Result of sequential pattern	140
25. Settings for association	141
26. Rules detail list.	142
27. Result of linear regression	143
28. Data models.	163
29. DPropR components	166
30. Captured CPU time	173
31. Calculating capture ratio	175

X Capacity Planning for Business Intelligence Applications: Approaches and Methodologies

Tables

1. DW concurrency ROT	30
2. Capacity growth (Table 5-2 from Worksheet)	97
3. Calculating overall capacity requirements	99
4. Example of two-dimensional model	124
5. Example model with aggregates added	125
6. Example widgets model that exhibits sparsity	126
7. Example proof of concept model	132

Preface

Businesses of all sizes are recognizing the powerful effect that data analysis can have on decision-making, resulting in the increasing deployment of Business Intelligence (BI) applications and solutions. This IBM Redbook contains approaches and methodologies that help you do capacity planning for the system resources that support BI applications accessing DB2 on OS/390.

Written for S/390 capacity planners, it includes the following:

- A discussion of the fundamental concepts of BI environments
- A structured methodology for doing a capacity plan for the S/390 BI hardware and software infrastructure
- A detailed description of how experienced capacity planners can modify traditional S/390 capacity planning techniques to address the unique characteristics of BI environments
- A set of worksheets to help you and your capacity planning team members develop a S/390 BI infrastructure capacity plan

This redbook addresses capacity planning for the S/390 Business Intelligence infrastructure, which includes processors, memory, I/O subsystems and DASD; it does not address capacity planning of network or non-S/390 hardware platforms.

You can use the capacity planning concepts and techniques described in *DB2 for OS/390 Capacity Planning*, SG24-2244 as a guide for the collection and analysis of data.

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 Poughkeepsie Center.

Seungrahn Hahn is a Business Intelligence specialist at the ITSO. Before joining the ITSO, she worked as an IT specialist in physical database design, performance tuning, and data sharing implementation with DB2 for OS/390 and BI application solutions in IBM Korea.

M.H. Ann Jackson has over 25 years of experience assisting customers in architecting and implementing IT applications. She has specialized in delivering DB2 and Business Intelligence solutions for the past 15 years, with an emphasis on performance optimization and capacity planning.

Bruce Kabath is an I/T Specialist for IBM Global Services, Global Application Delivery. He has been with IBM for over 12 years, has 10 years of experience with DB2 for OS/390, and is DB2 UDB certified. He has worked in various capacities in application development and DBA roles. His current responsibilities include application and system performance and tuning.

Ashraf Kamel is a senior Business Intelligence Solutions Specialist with the IBM Americas BI Solutions Center in Dallas, Texas. He focuses on BI solutions for the S/390 platform and is typically involved in integrating IBM's products and platforms to develop customer solutions. Ashraf has an international exposure to many customers in the US and the Middle East with an emphasis on Business Intelligence Solutions Architecture and Consulting.

Ana Rivera Matias has been an IT Specialist in IBM Global Services, Spain since 1996. She is working on System Integration and Application Development projects for Telco&Media customers. Her areas of expertise include DB2 for OS/390, data warehousing, data modeling and DPropR.

Caryn Meyers has 20 years of experience in assisting customers in architecting, installing, and sizing technology solutions to address business needs. She has conducted capacity planning efforts for large financial, retail and distribution accounts, and has specialized in delivering Business Intelligence solutions for the past five years.

Merrilee Osterhoudt has 10 years of experience in MVS operating system development and product integration, and has been involved in the development, test and support of data warehouse solutions on S/390 for the past eight years. She is currently responsible for the S/390 Business Intelligence technical support strategy worldwide.

Gary Robinson is a Senior Software Engineer in IBM Silicon Valley Lab. He has over 12 years experience in Decision Support and Business Intelligence with IBM, and is one of the original members of the DB2 OLAP Server Development team. Gary works extensively with customers and partners deploying DB2 OLAP Server and Business Intelligence solutions. He is a regular speaker at major Business Intelligence conferences, and has written for DB2 Magazine. Before joining IBM, Gary worked in semiconductor manufacturing research in the Netherlands and Space research in the UK. He holds a BS in Computer Science.

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

Terry Barthel
Alfred Schwab
Alison Chandler
International Technical Support Organization, Poughkeepsie Center

Nin Lei
IBM S/390 Teraplex Center, Poughkeepsie

Chris Panetta
IBM S/390 Performance, Poughkeepsie

Evanne Bernardo
Eugene Azuolas
IBM S/390 Performance and Capacity Planning, Gaithersburg

Dick Pierce
Jim McCoy
IBM S/390 Large Systems Support, Washington System Center

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 191 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

Chapter 1. Introduction to capacity planning

The focus of this redbook is to provide approaches and methodologies for capacity planning of the system resources that support Business Intelligence (BI) applications accessing DB2 on OS/390. This chapter will establish a baseline of fundamental concepts related to Business Intelligence and capacity planning for the BI infrastructure on S/390. The following topics will be discussed in this chapter:

- Fundamental concepts of Business Intelligence
- The definition of the Business Intelligence infrastructure
- The reasons that capacity planning for the BI infrastructure is important
- The reasons that capacity planning for the S/390 BI infrastructure differs from traditional S/390 capacity planning for OLTP environments
- The skills and resources required to do an effective capacity plan for the S/390 BI infrastructure
- A brief overview of the remaining chapters in this book

1.1 What is Business Intelligence (BI)

The objective of this section is to familiarize the capacity planner with the fundamental concepts of Business Intelligence systems. If you are already familiar with Business Intelligence concepts, you may want to go directly to 1.2, "What is the Business Intelligence infrastructure" on page 6.

Business Intelligence (BI) is the gathering and analysis of vast amounts of data in order to gain insights that drive strategic and tactical business decisions which, in turn, will improve performance in the market place. Data can be related to all facets of the business, such as customer transactions, customer demographics, company financial information, manufacturing processes, inventory management, industry trends, supplier transactions, and competitor profiles. Business Intelligence data is collected from internal sources, such as transaction systems, manufacturing processes, customer records, as well as external sources such as consultant and industry studies, the printed media and the Internet.

The purpose of Business Intelligence systems is to support the processing that transforms data into actionable information. Some common BI applications include market segmentation, customer profiling, campaign analysis, fraud detection, risk analysis and profitability analysis. Many corporations are implementing solutions that integrate business intelligence

into business processes through automation. Some examples of these integrated solutions are Customer Relationship Management (CRM), Campaign Management Systems (CMS), Risk Management and Supply Chain Management (SCM) or Supplier Relationship Management (SRM).

Businesses of all sizes are recognizing the powerful effect that data analysis can have on decision making, which has resulted in widespread funding and deployment of BI applications and solutions.

1.1.1 Business Intelligence data stores

As shown in Figure 1 on page 3, the data warehouse and the data mart are two common types of data stores in a BI system which are optimized to facilitate access and analysis of historical, non-volatile data by business users. A third data store, the operational data store (ODS), has evolved in recent years to support the analytical processing of current, changeable data.

A data warehouse is generally characterized as a centralized repository that contains comprehensive detailed and summary data, that provides a corporatewide view of customers, suppliers, business processes and transactions from an historical perspective with little volatility. The data warehouse incorporates data from all sources and supports multiple business functions. It is accessed by many departments in the organization

The data mart, on the other hand, is typically single-subject oriented, containing a subset of the data housed in the data warehouse which is of interest to a specific business community, department or set of users, (for example, marketing promotions, finance, or account collections). The data mart may exist on the same platform as the data warehouse or in a separate environment. In a well structured BI system, the enterprise data warehouse (EDW) serves as the single source for multiple data marts.

Some BI architectures include a third structure called the operational data store, which is an intermediary data store that combines characteristics of both operational transaction systems and analytical systems. The frequency of data update, real time or near real time to overnight, is a key difference between it and the data warehouse/data mart. Data already in the ODS may be subject to frequent changes. An ODS can be used for decision support activities against current data as well as a staging area for data acquisition and transformation of operational data into the data warehouse.

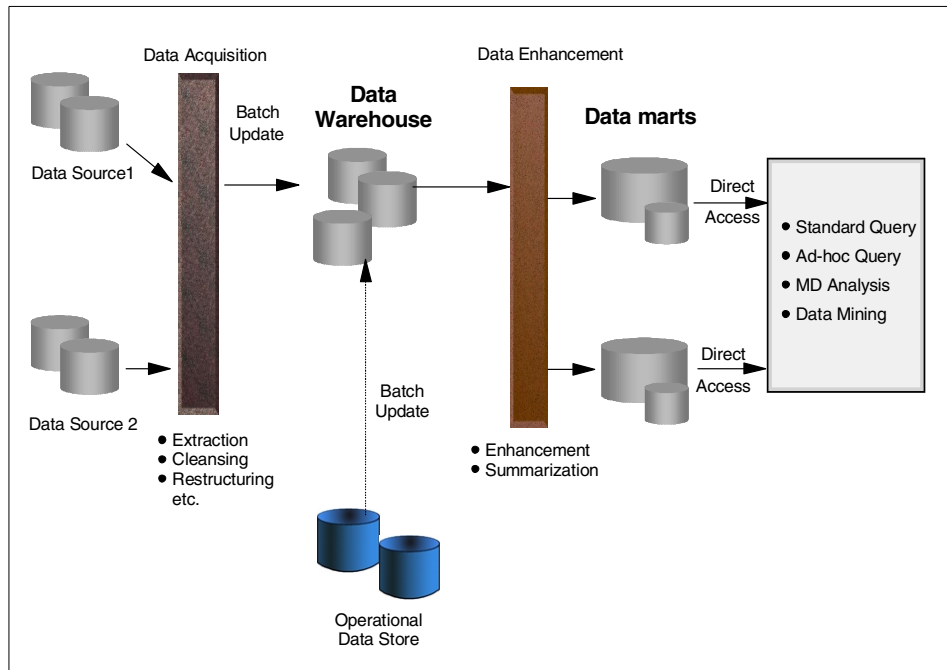


Figure 1. Data warehouse and data mart

When we refer to the data that is stored in the data warehouse, data mart and ODS, we are referring to “raw data”. Raw data is defined as the actual data that is recorded by transactions and produced by business processes (for example customer number, part number, account balance, purchase data, branch number). This is the data that will be analyzed in order to obtain insights (such as which territory was the most profitable in December?).

The processing and managing of raw data requires the creation and storing of “system data”. System data is used only by computing system software to manage its resources; however, there are still associated storage and processing costs which must be considered when planning for capacity requirements.

1.1.2 Business Intelligence processing

The processes that transform raw data into information fall into three categories:

- Data collection and maintenance
- Data analysis and presentation
- Data administration

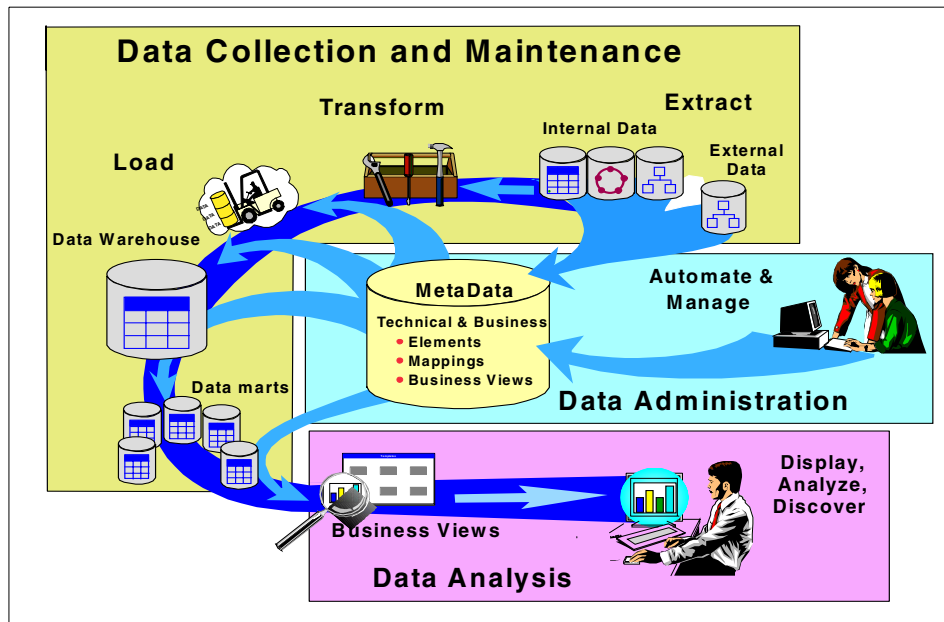


Figure 2. Data-to-information process

Figure 2 is a representation of the data-to-information process flow. This flow starts at the top right and moves counter-clockwise, as follows:

- Data collection and maintenance

First, sources of data, such as transaction systems, multimedia (text, video, audio), flat files and click streams, are identified and the relevant subset of raw data that is desired for analysis is extracted.

Raw data can come from different database formats, (DB2, IMS, Oracle, or Informix, for example), and can come in various physical formats (such as ASCII, EBCDIC, text and multimedia). The raw data must be transformed and stored into a single database format, such as DB2, for query access and analysis. Transformation also includes a cleansing process to resolve inconsistencies and remove incorrect or “dirty” data.

Once the detailed data has been transformed and cleansed, it must be loaded into the data warehouse. Data marts can be populated directly or by extracting subsets of the detailed data in the warehouse. Over time, the data warehouse and data marts must be maintained by removing obsolete data and kept current with updates from operational systems and external

sources. IBM and other software vendors provide software utilities and tools that support maintenance and incremental updating.

- Data administration

Business Intelligence systems, like transaction systems, require structured controls to maintain security, integrity, and relevancy of data. In addition to the traditional systems management and automation, BI system administration can be optimized through the use of metadata.

Metadata, put simply, is *data about data*. It includes information such as where the data comes from, where it is located, when it was last changed, who accesses it, and what access patterns exist. Products such as IBM's DB2 Warehouse Manager provide a central point of control and automation for data warehouse, data mart, and metadata management.

- Data Analysis

All of the previously discussed processes are necessary to build and maintain the structures that support data analysis. As shown in Figure 3 on page 6, the three most common techniques of data analysis, in order of complexity, are:

- Query and Reporting - Structured Query Language (SQL) tools are used to extract from a database all of the instances that match a specific set of criteria. This technique is commonly used for building reports and building lists of records that exhibit a predetermined set of characteristics. Simple query tends to be single-dimensional and can be performed using SQL-generating tools that do not require sophisticated analysis skills.
- On-Line Analytical Processing (OLAP) - This is the analysis of precalculated, summarized data in multiple dimensions and business views. This technique is used, for example, to gain insights into how events occur over intervals of time. OLAP can answer questions like, which branch office was most profitable in January? OLAP is a more complex form of analysis than query and requires more sophisticated tools, such as the IBM DB2 OLAP Server, and advanced skills.
- Data mining - This is a discovery-based analysis technique in which complex algorithms are applied to detailed data in order to reveal hidden patterns and relationships within data. Data mining is heavily used in fraud prevention, customer segmentation, and predicting behavior. This is the most complex form of data analysis, requiring very sophisticated tools (the IBM Intelligent Miner, for example) and highly skilled analysts.

Analysis tools and applications run on the desktop, across the Web and directly on host systems.

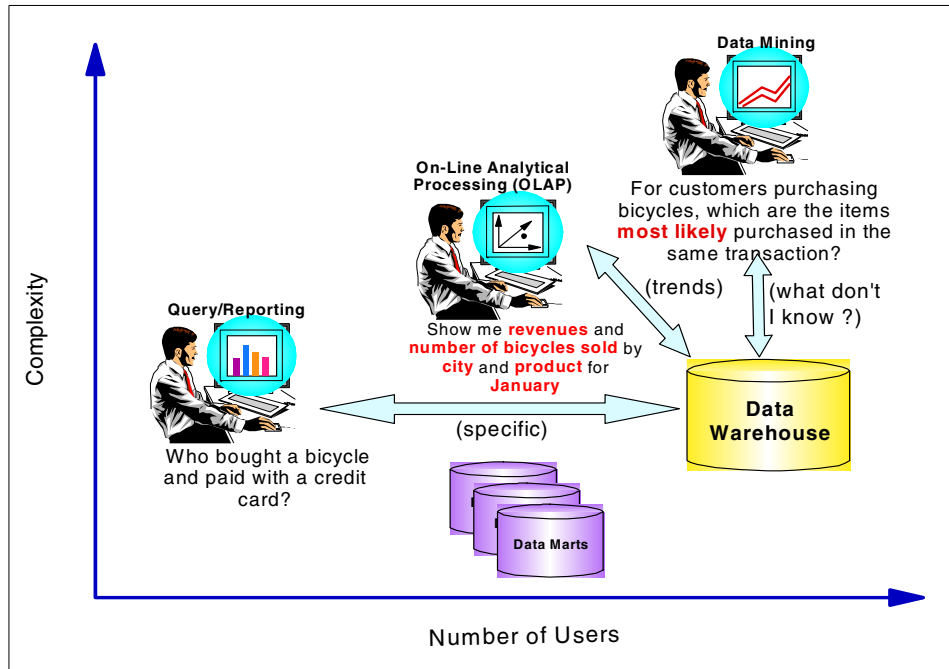


Figure 3. Data analysis techniques

An in-depth discussion of Business Intelligence architectures and processes, can be found in *The Role of S/390 in Business Intelligence*, SG24-5625, and *S/390 Business Intelligence Architecture Presentation Guide*, SG24-5641.

All of the processes we have described require processing resources, storage devices for data, and access paths to the data. These resources are the fundamental components of the Business Intelligence infrastructure.

1.2 What is the Business Intelligence infrastructure

In this section we distinguish the BI infrastructure from the other components of the overall BI system and put it in the context of the hardware and software components of a S/390 implementation.

1.2.1 The Business Intelligence system architectures

The Business Intelligence system consists of all the software that performs the processing of data described in the previous section, and the physical hardware and operating system upon which the software runs. When deploying BI applications, as with other applications, there are three logical processing layers, as shown in Figure 4 on page 8:

- *The data layer* - This layer supports all processes that move, store, format, and change the raw data. This includes all of the processes of data collection and maintenance of the data warehouse and data marts, as well as the processing of Structured Query Language (SQL).
- *The application logic layer* - This layer supports secured access to the BI application and data, the creation of SQL, and post processing of retrieved data before presentation to the business user. This may include security authorization and validation, generation of SQL, and manipulation to complete business rules processing and analysis.
- *The presentation logic layer* - This layer directly interacts with the business user. It includes the graphical user interface (GUI) to develop business queries and to present results in multiple formats, i.e. texts, charts, and graphics.

The physical implementation of a BI system architecture may include one, two or three physical tiers of hardware and software that perform the processing. In the single, 1-tiered physical implementation, the processing for all of the logical layers of the architecture is performed on one hardware and operating system server with results displayed on attached unintelligent, “dumb” terminals.

In a 2-tiered implementation, there are two physical hardware platforms performing the logical processing. The data layer and, optionally, a portion of the application layer are on one server while the second platform performs the remaining application and presentation functions. In Client/Server implementations, this configuration is often a large host server with attached intelligent “fat” clients.

In a 3-tiered implementation, the processing layers are implemented across three separate hardware and operating system platforms. Like the 2-tiered implementation, the application layer could be split across two of the tiers. However, the data layer is more often housed by itself on its own platform, while a high majority of the application function is located on a middle server, and the presentation layer on an intelligent PC or “thin” client.

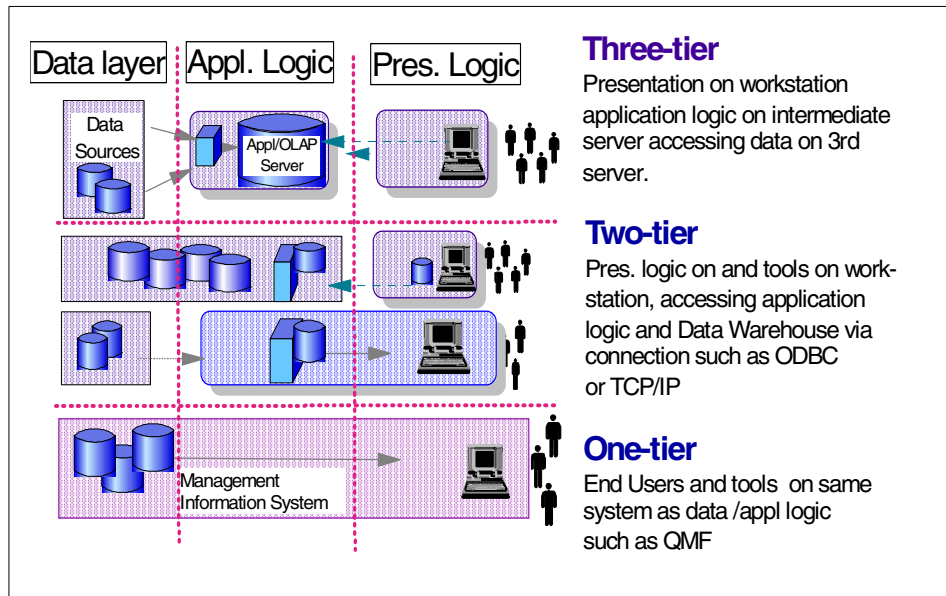


Figure 4. Application deployment on a BI system

Many customers implement a combination of architectures as they deploy applications on their Business Intelligence system, as depicted in Figure 4.

1.2.2 The S/390 Business Intelligence infrastructure

For the purpose of discussion in this book, we define the Business Intelligence infrastructure as all the hardware and software that support the processes which make up the foundation of the Business Intelligence system. This includes the Data Base Management System (DBMS) software, the operating system environment, and the physical hardware resources that support the movement, storage, formatting and altering of data. The S/390 BI infrastructure includes the DB2 on OS/390 database management system and the DB2 OS/390 family of utilities, the OS/390 operating system and its subsystems, S/390 processors, memory and I/O channels, as well as controllers and disk.

1.3 Capacity planning for the BI infrastructure

In this section we do the following:

- Establish a definition of capacity planning
- Help the capacity planner understand how BI workloads differ from traditional transaction workloads
- Help the capacity planning team understand why capacity planning for the BI infrastructure should be an integral part of their IT processes

1.3.1 What is capacity planning

A simple definition of capacity planning is “to estimate the computer resources required to meet an application’s service-level objectives over time.” The principal objective of capacity planning is to proactively maintain a *balanced* computer system that will sustain performance objectives based on priorities established by the business. A system is balanced when all of its resources are working in concert to allow the optimal amount of workload through the system to meet specific objectives.

Capacity planning is employed to estimate the amount of computing resources that will be needed to support the future processing needs of an existing workload that is in production today. Capacity planning utilizes measurement tools to understand the resource usage of the current workload, together with the collection of feedback from business users, in order to understand future requirements as they relate to the expected growth in data and users, changing data access, and performance requirements. The data provides insights that can help predict the increase or decrease in system resources that will be required to meet the Service Level Agreements for this workload in the future.

Capacity planning can either be approached in a casual, informal manner or in a very structured and disciplined fashion. The more disciplined the methodology, the more accurate the results.

1.3.2 Why capacity planning is essential for the BI infrastructure

In the past, Business Intelligence (initially called *decision support*) workloads were viewed as non-essential, discretionary work and consequently have often been given a low priority when planning computing resource requirements. Today, however, most successful businesses have come to view their BI systems as equal in importance to their operational systems.

Thus, establishing Service Level Agreements (SLAs) and maintaining balanced systems to handle BI workloads has become a critical requirement for IT. The BI infrastructure is the fundamental building block upon which the BI system is constructed. If the resources are not in place to optimize the performance of data warehouse, the BI system is in jeopardy.

While the fundamental concepts of capacity planning for traditional workloads can be applied to BI systems, there are significant differences in the characteristics of OLTP and BI workloads that must be taken in consideration when planning for the growth of the BI system.

1.3.3 Capacity planning for traditional workloads

Because transaction systems are the lifeblood of the business, meeting availability, performance, and throughput requirements to keep the critical processes operational has been a critical metric of IT's contribution to the business. Hence, capacity planning for the growth of the transaction systems has become well integrated into IT processes.

Transaction, or OLTP, systems have been running on mainframe hardware and OS/390 operating system infrastructures for decades. Over time, proven capacity planning techniques and methodologies have been developed based on the evolution of software tools to collect OS/390 operating system statistics, the accumulated experience of system programmers with the management of the S/390 system resources, and with their observation of the characteristics of the transaction workloads.

OLTP workloads facilitate capacity planning because they are predictable in nature, exhibiting the following characteristics:

- Small units of work with consistent elapsed times (which are usually very short)
- Predictable DB2 access paths to the data, using direct index lookup with virtually no scanning of large numbers of records
- Very small answer sets (often a single row), requiring little I/O
- Simple SQL, easily optimized and rarely requiring parallelism
- Users have little or no control over the SQL that is executed

Transaction systems tend to grow predictably, in a gradual and linear fashion, making planning for growth in volumes relatively straightforward. Batch workloads associated with maintaining these systems are also reasonably predictable because they are closely tied to the growth in the transaction processing.

1.3.4 Capacity planning for BI workloads

While many of the traditional techniques can be applied to capacity planning for the BI infrastructure, BI workloads have notably different characteristics from traditional workloads, and they require the capacity planner to modify his approach.

- Units of work are heterogeneous in nature, varying in elapsed times from subsecond to many hours.
- There are unpredictable DB2 access paths, sometimes using indices but frequently scanning very large volumes (gigabytes and terabytes) of data.
- Very large answer sets (millions of rows) are common, requiring a lot of concurrent I/O that affects elapsed times and resource consumption.
- Frequently there is complex SQL that is difficult to optimize and heavily dependent on parallelism, capable of consuming all available resources for extended periods of time.
- Users generate their own SQL, often with desktop tools from many different vendors, using techniques like point-and-click to build complicated predicates.

The growth of the BI infrastructure is difficult to anticipate accurately, particularly in the initial stages of deployment of a data warehouse. As a company realizes the initial benefits of running Business Intelligence applications, it is common for a dramatic growth in the BI system to occur. More business units begin to deploy BI applications, increasing the number of users accessing the data, as well as increasing the amount of data that users want available for analysis.

As users become more skilled in the use of sophisticated analysis tools, there is also an increase in the volume and in the complexity of the queries that are generated. The growth of data mining and online analytical processing (OLAP) tools requires more powerful engines and sophisticated parallelism, both in hardware and software.

Data maintenance, such as updates, backup, reorganizations and runstats, is also affected by a dramatic growth in the size of data warehouses and data marts. In addition, the trend toward globalization is driving demand for availability of business intelligence systems to business users across multiple (and often, all) time zones.

As BI techniques become integrated into real time business processes, the requirement for current data becomes more critical. These factors are quickly

closing the “maintenance window”, requiring real time updates and putting significant stress on the data maintenance processes.

1.3.4.1 Critical elements

The following are critical elements in the growth of the BI infrastructure:

- Number of users
- Amount of data
- Size and complexity of query workload
- Growth in maintenance workload

Growth in any or all of the above elements, without appropriate adjustments in the system capacity, can negatively impact query response times, as well as stress the data maintenance processes, resulting in decreased effectiveness of the BI infrastructure.

S/390 system measurement tools (such as System Management Facility, Resource Management Facility, and DB2 Performance Monitor traces and reporting capabilities) can be utilized to capture the performance and resource consumption characteristics of the existing workload and apply proven guidelines and methods to approximate the future growth in resource requirements based on projected growth of the workload. Using capacity planning on an ongoing basis enables you to build a balanced system that supports both the growth in data and the growth in users.

1.3.4.2 System resources

The critical resources to support the S/390 BI infrastructure are:

- CPU
- Storage, central and expanded
- I/O channels
- Controllers (storage directors, cache, and non-volatile storage) and disk

Having a “balanced” system implies that a relationship exists between these key system resources. *System capacity* is a measurement of the optimal amount of work that can be processed when all of the resources are being fully utilized. When business objectives are not being met, the system is out of capacity.

If your data warehouse is part of a Parallel Sysplex, you must assess the impact of Parallel Sysplex on these resources as well. Parallel Sysplex introduces the Coupling Facility (CF) as an additional resource, formed by:

- Links
- CPU
- Storage

The network capacity is also be very important. The capacity required depends on the data to be moved between the two systems, the timing window in applying the data, the desired currency of the data warehouse, and the bandwidth installed.

1.4 Information needed for capacity planning

A capacity planning study is focused on obtaining answers to the following questions:

- Which Information Technology (IT) resources are currently in use? (CPU, CF, devices, controllers, channels, coupling links, storages and so forth)
- Which parts of the workloads consume most of the resources?
- What is the expected growth rate for particular workload? In a data warehouse environment, the workloads often grow exponentially within the first months (or years) of implementation.
- When will the demand on existing resources negatively impact delivered services levels?

The interrelationships of processing power, I/O capability, processor storage size, and networks determine system performance. Capacity can be determined based on the performance objectives.

In subsequent chapters there will be detailed discussions on how to collect and analyze data to get answers to these questions

1.5 The BI infrastructure capacity planning team

Many different IT professionals are involved in the capacity planning process, each with a different task and focus. In this section we describe the roles and tasks of these individuals.

Capacity Planner

The capacity planner has responsibility for developing a hardware capacity plan for the data warehouse as requirements change over time, and is the lead member of the BI capacity planning team.

Storage specialist

The storage specialist understands the storage requirements of the data warehouse and data marts, and is responsible for making recommendations that will optimize DASD usage and storage management, such as S/390 hardware and Storage Management System (SMS).

Database administrator

The database administrator uses the DB2 planning methodology and formulas to design and calculate the size of the tables, and work with application developer to define, measure, and optimize the retrieval of data.

DB2 systems programmer

The DB2 systems programmer provides critical input to the capacity planner related to the utilization of system resources by DB2 and the DB2 system utilities. The DB2 systems programmer should provide input related to parallelism processing requirements and workload management.

System programmer for OS/390

The systems programmer for OS/390 provides the capacity planner with important information related to the overall system performance and throughput, including utilization of Workload Manager to optimize system resource allocation.

Data analyst

The data analyst provides insights about the relationships between legacy data and the data warehouse data, and input into the processing requirements of batch and query environments.

Application analyst or developer

The application analyst or developer interfaces with the business user to gather new or additional data access requirements and also identifies associated volumes. The person is responsible for communicating this information to the DBA, data analyst, DBA, and capacity planner.

Tools specialist

Business Intelligence systems often include multiple tools and applications, provided by multiple vendors, to perform the numerous functions required to build, maintain, access and analyze data. The tools specialists can provide key input to the capacity planner in regard to the characteristics of the work that is generated by various tools and applications.

Operations staff

The operations staff is responsible for running BI applications and monitoring system performance, and can provide excellent input to the capacity planner in regard to the ability of the system to meet service level agreements.

Network specialists and non-S/390 server platform specialist

In a data warehouse environment, multiple platforms may be utilized for connectivity middleware, application servers, Web servers and data marts. Developing a comprehensive capacity plan across a heterogeneous environment requires participation by support specialists for each of the server components, such as UNIX, Windows NT and LINUX systems. Capacity planning for these components of the overall BI system is not within the scope of this book.

1.6 Capacity planning methods

There are various methods and techniques that you can use to plan for the changes in capacity requirements for your DB2 Business Intelligence systems on S/390. The methodology that you choose for your capacity planning exercise will depend on the available time, cost, skill and the level of accuracy that best meets your installations needs.

For detailed descriptions of the capacity planning steps for S/390 systems and DB2 on OS/390, refer to *OS/390 MVS PSX Capacity Planning*, SG24-4680, and *DB2 for OS/390 Capacity Planning*, SG24-2244.

In our redbook, we focus on the unique aspects of Business Intelligence systems and how to modify traditional capacity techniques and methodologies to address these characteristics when doing capacity planning for the BI infrastructure.

1.7 Capacity planning tools

There are several capacity planning tools available to IBM specialists and customers. These tools vary in the amount of time that must be invested and the level of accuracy that can be expected, as do capacity planning methodologies. When making a decision about using a capacity planning tool, numerous factors should be considered. Chapter 5 discusses this topic in detail.

1.8 A word on resource sizing

Capacity planning and resource sizing, while similar activities, have different objectives and methodologies, and it is important to understand when it is appropriate to engage in a capacity planning exercise and when it is appropriate to do resource sizing.

Capacity planning is used to estimate the future computing resource requirements of an *existing* workload. In contrast, resource sizing entails estimating the computing resources required to support *a new workload that has not been run in production*. There are no production measurements available for a new workload, and the sizing estimations must rely on extrapolations of the following:

- Information about similar workloads
- Industry benchmarks, and/or
- Extrapolation from pilots or beta programs

A methodical sizing exercise has a much higher risk of inaccuracy than a capacity planning exercise, simply because of the lack of factual data available about new workloads.

When reading this redbook, it is important to remember that the techniques and methodologies we cover are discussed in the context of capacity planning. Chapter 7 discusses alternative methodologies for sizing new BI applications when little quantitative data is available.

1.9 Overview of remaining chapters

- Chapter 2, Resource Usage attributes of a Query Workload, discusses some of the characteristics of the query component of a Business Intelligence or data warehouse (DW) environment that differ from a traditional Online Transaction Processing (OLTP) application and that have an impact on the capacity planning results.
- Chapter 3, Data Maintenance, discusses the maintenance or batch workload that is part of a Business Intelligence solution, including data population, backup, recovery, REORG, and RUNSTATS.
- Chapter 4, Characterization of Existing Workloads, focuses on the unique considerations when capacity planning for a BI workload, and on how a BI workload differs from typical workloads in the S/390 environment.
- Chapter 5, Establishing Current System Usage, looks at how to gather the necessary information to understand BI workload characteristics, and

introduces a series of worksheets designed to aid in capturing that information. The monitor reports required to locate the information are identified and, when necessary, the methodology for calculating the values to complete these worksheets is provided.

- Chapter 6, Growing Existing Workloads, provides a detailed discussion of how to forecast the capacity requirements for a BI environment.
- Chapter 7, Sizing Methodologies for New Applications, discusses the challenges of estimating the computing requirements of new applications, the techniques available, guidelines for selecting the appropriate technique, and the data that is required for each technique.
- Chapter 8, Estimating Model Size for IBM DB2 OLAP Server, discusses the factors that influence model size and provides guidance for estimating the capacity requirements without building the entire model.
- Chapter 9, Sizing Intelligent Miner for Data for Performance guides you in creating a sound methodology for sizing and planning your Intelligent Miner application environment on OS/390 as a continual process. This includes giving you the ability to measure the right data and correlate Intelligent Miner application transaction events to the measured data.
- The appendices provide additional reference materials and information that you can use in you capacity planning. Appendix A contains a set of worksheets (referenced in Chapters 2 to 7), that will help you structure and document information gathering.

1.10 Summary

- Business Intelligence (BI) is the gathering and analysis of vast amounts of data in order to gain insights that drive strategic and tactical business decisions.
- BI workloads are becoming mission-critical, and effective capacity planning for the BI infrastructure is critical to meet Service Level Agreements.
- The three basic data stores in BI systems are: the data warehouse, the data mart, and the Operational Data Store. The three logical processing layers of a BI implementation are: the data layer, the application logic layer, and the presentation logic layer.
- The S/390 BI infrastructure is made up of all the S/390 hardware (processors, memory, channels and disk) and software that support the processes which make up the foundation of the Business Intelligence system.

- The objective of capacity planning is to estimate the future computing resource requirements for an existing workload. Resource sizing is the estimation of computing resource requirements for a new workload.
- Capacity planning requires a team effort by IT professionals from various disciplines.

Chapter 2. Resource usage attributes of a query workload

In order to plan for the capacity needs of a workload, it is important for you to understand the many factors that impact the resource consumption of that workload. This chapter discusses some of the characteristics of the query component of a business intelligence or data warehouse (DW) environment which differ from a traditional Online Transaction Processing (OLTP) application. These factors have an impact on capacity planning results.

2.1 Query parallelism

In an OLTP environment, the SQL statements are typically very small, resulting in fast response times. By contrast, some of the queries against a data warehouse can be long running and access large volumes of data. They can be orders of magnitude greater than the granular work found in OLTP, therefore the time necessary to execute those queries can be very lengthy: a warehouse workload can have queries with elapsed times ranging from a few seconds to many hours.

To shorten overall execution time, DB2 may optimize the queries into smaller segments of work that execute concurrently, or in parallel, on separate engines within the system. This *query parallelism* can significantly improve the performance of BI applications.

However, because of the ability to break a single query into many separate tasks, one query can potentially consume 100% of the CPU and I/O resources available in a data warehouse system. As a result, it can potentially take fewer queries to drive resources to the point of saturation.

You can expect to frequently see CPU utilizations of 100% for a DW, compared to 70% to 80% utilizations for OLTP workloads. Even during off-peak hours, the DW workload can continue to drive CPU utilization to high levels of consumption.

These factors make capacity planning for a data warehouse much more goal-oriented, rather than being driven simply by resource consumption. Therefore, when doing capacity planning in this situation, you must ask yourself:

- Are you meeting your business objectives?
- Are you meeting your Service Level Agreements (SLAs)?
- Are you providing good service?

2.1.1 Types of query parallelism

Query execution time is a critical metric in a warehousing environment. Workloads are typically composed of queries with widely varying execution times, which generally are a result of the amount of data scanned to derive an answer set.

As queries scan more data, DB2's likelihood of breaking that scan into separate segments increases. Breaking a query into segments allows the segments to execute on separate engines in the configuration, which has a dramatic impact on reducing the overall execution time.

This segmentation of a query is referred to as *parallel processing*. It is an important aspect of a data warehouse environment, where queries typically access large volumes of data. One of the goals in the design of a data warehouse should be to develop a partitioning strategy that facilitates parallelism for query performance.

DB2 for OS/390 has three flavors of query parallelism. These are:

- I/O parallelism

Introduced in DB2 V3, with I/O parallelism one task can initiate multiple simultaneous I/O requests within a single query, thereby retrieving data into the buffers in parallel and effectively eliminating much of the I/O wait time.

- CPU parallelism

Introduced in DB2 V4, CPU parallelism breaks a single query into multiple tasks that run concurrently on all available engines in an SMP server. One coordinating task can initiate two or more tasks. Each task handles one I/O stream, thus allowing I/O and processing activities to run in parallel.

- Sysplex parallelism

Introduced in DB2 V5, sysplex parallelism is similar to CPU parallelism, but is extended so that all processors in a Parallel Sysplex can be used to process a query. The query can be split into multiple tasks which run in parallel across all servers that are clustered in a data sharing group. By taking advantage of all the processing power within a Parallel Sysplex, the elapsed times of complex queries can be reduced by orders of magnitude.

The DB2 Optimizer determines the degree of parallelism (how many segments a single query can be broken in to) for a query. The degree of parallelism is related to a number of factors, including the hardware configuration underlying the database. DB2 considers factors such as the number of partitions, central processor speed, number of available engines,

estimated I/O cost, and estimated processor cost when determining the number of parallel operations for a query.

The goal of DB2 parallelism is to minimize multitasking overhead while achieving optimum throughput. A single query split into multiple tasks running simultaneously can quickly drive up your CPU utilization, even to the point of saturation.

While the CPU is usually the system resource most affected by DB2 parallelism, storage and I/O should be monitored, as in any environment. Proper allocation of DB2 virtual bufferpools helps ensure that storage is being used efficiently. Allocating DB2 hiperpools in expanded storage, or utilizing DB2's ability (beginning in Version 6) to put virtual buffer pools in data spaces, facilitates the effective use of storage.

For more information on query parallelism, refer to *DB2 for OS/390 V5 Performance Topics*, SG24-2213.

2.1.2 Partitioning for parallelism

Having multiple parallel tasks for a single query increases simultaneous I/O activity against the data. Partitioning the data into logical segments, each segment stored in its own dataset, can minimize possible I/O contention.

For optimal performance of the I/O system, it is often best to spread table and index partitions on as many different volumes (physical or logical) as possible. This can be accomplished in a number of ways.

Range partitioning

A large table or index can be broken into physically separate segments, each placed on a separate (preferably), or the same, disk volume. In DB2 for OS/390, this is known as *partitioning*. Each segment is a partition of a partitioned table or index space.

With DB2 for OS/390, partitioning is accomplished by dividing the data based on a sequential key range, referred to as the partitioning key. This is known as *range partitioning*. Determining the appropriate partitioning key to maximize query performance is done during the logical database design by the DBA team.

Data can be partitioned according to any of the following criteria:

- Time period (date, month, or quarter)
- Geography (location)
- Product (more generally, by line of business)

- Organizational unit
- Any other method that divides the data into approximately equal-sized segments

For more information on this subject, refer to one of the following IBM Redbooks: *DB2 Performance Topics*, SG24-2213, *DB2 for OS/390 Capacity Planning*, SG24-2244, and *DB2 for Data Warehousing*, SG24-2249.

Hash and random key-based partitioning

Partitioning is implemented by many relational databases, regardless of the platform, operating system or vendor. In some implementations, this may be done through a different technique called hashing, and is known as *hash partitioning*.

With hash partitioning, columns in the table are identified to be hashing key columns (this is often referred to as the *partitioning key/index*, or even more confusing, the *primary index*) and each row's corresponding hash key values will be randomized to a partition number for storage and retrieval. Like the range partitioning technique, identifying the appropriate hashing columns to maximize query performance is done during logical database design by the DBA team. The selection is based primarily on choosing a set of columns whose randomized values will result in roughly the same number of rows per partition.

Introduced in DB2 for OS/390 V6, the ROWID column containing a DB2 guaranteed unique, randomly generated, stored column value for each row can be specified as the partitioning key. This will automatically spread the row evenly across the partitions on the disk subsystem. However, random key-based partitioning techniques introduce maintenance issues related to update, backup/recovery and archiving processes, just as hashed partitioning does on the other platforms.

Because the primary objective of hash partitioning is to spread the data evenly across all partitions by randomizing the hashing column values, the addition of data occurs to many, preferably all, partitions. As a result, backups, recoveries, and mass removal of data must occur across all partitions as well. In a VLDB environment, the time required for this may be significant and can impact data and application availability. On the other hand, range partitioning, especially if a time-based limitkey is chosen, can reduce the time and system resources needed for these maintenance processes, because only the partitions containing the affected time period must be processed.

2.1.3 Query parallelism and I/O activity

While partitioning improves query performance by driving more I/Os concurrently, this can increase the I/O activity to the disk subsystems. To minimize I/O contention for parallel queries accessing the same or different partitions, newer disk subsystems have brought new functionality to the marketplace. The latest disk subsystem, the Enterprise Storage Server, shows virtually no degradation when a number of partitions located on the same physical volume are scanned simultaneously.

The Enterprise Storage Server (ESS) has some advanced features that benefit DB2 performance. These features include:

- Parallel access volumes (PAV), which drives up to 8 multiple active concurrent I/Os on a given device when the I/O requests originate from the same system. PAVs make it possible to store multiple partitions on the same volume with almost no loss of performance.

In other DASD subsystems, if more than one partition is placed on the same volume (intentionally or otherwise), attempts to read the individual partitions simultaneously result in contention which shows up as I/O subsystem queue (IOSQ) time. Without PAVs, poor placement of datasets can significantly elongate the elapsed time of a parallel query.

- Multiple allegiance, which allows multiple active concurrent I/Os on a given device when the I/O requests originate from different systems.

PAVs and multiple allegiance dramatically improve I/O performance for parallel work on the same volume by nearly eliminating IOSQ or PEND time and drastically lowering elapsed time for both transactions and queries.

In general, remember that parallelism exists to reduce elapsed times and in doing so, all resources, CPU, storage and I/O may be driven harder.

2.1.4 Concurrency and locking

With parallelism, there is a trade-off in concurrency, as each query drives system resources higher in a condensed period of time. Thus, fewer queries running concurrently can saturate the system. The higher the degree of parallelism, the lower the degree of concurrency. This is due to the intensity of the structured query language (SQL) in a data warehouse environment and its ability to consume system resources.

In OLTP environments where queries are short running, a key issue with higher levels of concurrency is lock avoidance. Locks on database records are necessary to serialize access when the data is being updated. Locks are generally not an issue for data warehousing queries which typically do not

update information in the database, but rather read the information, summarizing it in a meaningful manner.

Business intelligence queries can therefore take advantage of a DB2 function called *read through locks* or *uncommitted reads*. This allows users to access data, regardless of whether another user is in the process of updating that data. When generating total sales for a region over a month or a quarter, being off by 1 is generally not an issue. When reviewing a data warehouse system, ensure the BI application can use read through locks to minimize response time for queries.

Query concurrency can also have a dramatic effect on query execution times in a data warehouse. For example, take a single parallel query executing standalone that already utilizes the CPU resources to 100% and requires 30 minutes to complete. When running concurrently with another query that also utilizes 100% of the CPU, it will likely take twice as long to complete both queries.

2.2 Response time consistency

Any data warehouse is highly susceptible to inconsistent response times without proper management of system resources. When the system is saturated, queries will experience the effect of latent demand. Regardless of platform, it may not take many queries to reach the system CPU saturation point, at which time work will begin waiting for the availability of system resources.

It is not uncommon for the same user, executing the same query, to experience different response times due to:

- Running concurrently with varying workload
For example, being the only query in the system one day and running with many other queries the next day.
- Relative priority to concurrent workload
For example, the user may be in the top priority query one time and a lower priority another time.
- Vastly different data volumes accessed for the same query, depending on values of input parameters
For example, a query in which a user selects a highly populated state like California versus a sparsely populated state like Idaho will probably access much more data and take longer to process.

These variations in workload can have a dramatic impact on the response time a user experiences in a data warehousing/business intelligence environment.

2.3 Workload Manager (WLM)

When dealing with diverse workloads such as the query mix of a business intelligence environment, it is important for the operating system to have the ability to efficiently manage the workload.

Workload Manager (WLM) is a component of OS/390 that allows you to prioritize the business workload. You define performance goals for each type of workload or query type and assign a business importance to each goal. You set the goals for the work based on business priorities, and the system decides how much resource, such as CPU and storage, should be allocated to meet the goal.

WLM is an integrated function of OS/390 providing highly sophisticated, industry-unique features for managing complex query workloads. It introduces controls that are business goal-oriented, work-related and dynamic.

The most important step in WLM implementation planning is properly classifying your workloads; in other words, establishing an overall workload policy for your entire system (this is called a *service definition*). The service definition contains performance goals, the business importance of the work, resource requirements, and reporting information. Goals are defined in business terms and the system automatically determines the amount of resources that are allocated. WLM constantly monitors these rules and dynamically adjusts its allocations to meet the goals.

2.3.1 Workload management issues

Business intelligence workloads are diverse in nature and are characterized by dynamic mixes of ad hoc queries whose system resource consumption varies widely from query to query. This makes it worthwhile for you to understand the workload requirements and establish rules accordingly.

- Query concurrency

The system must be flexible enough to handle large volumes of concurrent queries. The amount of concurrency fluctuates as does the number of users, so the system can quickly become inundated and reach saturation. Being unable to manage these variations can lead to inconsistent response times.

- Query monopoly

Probably every DBA has their own story of the *killer query*, a query that enters the system and seems to run forever, monopolizing resources while doing so. In the meantime, this long-running query is preventing or impeding other work from completing.

- Query identification

In an OLTP environment, it is usually easier to identify pieces of work that may need to be segregated. In a data warehouse, you may not have the same granularity. Front-end tools using generic IDs make it harder to distinguish the user community. In a data warehouse, it may be challenging to isolate program names or the source of work.

It is critical to manage resources dynamically and efficiently in a data warehouse because utilization is typically at 100% for long periods of time, whereas OLTP often has some *reserve* resources available to handle spikes.

For more information on Workload Manager for OS/390, refer to *OS/390 Workload Manager Implementation and Exploitation*, *Data Warehousing with DB2 for OS/390*, and *Building VLDB for BI Applications on OS/390: Case Study Experiences*. The WLM home page also contains quite a bit of useful reference material at: <http://www.s390.ibm.com/wlm>.

2.4 Usage patterns

A data warehouse tends to be much more unpredictable than an OLTP environment. OLTP offers virtually no flexibility in how data is accessed by its end users compared to a data warehouse that often offers ad hoc query capability to its users. With this ability to submit information requests at will, the business intelligence applications resource requirements can vary dramatically over time.

2.4.1 Increased usage

With the ability to create ad hoc queries, business users often increase the number and types of queries they submit. This ability to request any information needed drives higher system utilization as users become knowledgeable about the data that is stored in the warehouse, and the types of information they can derive from it.

CPU usage may increase given the same user community working with the same rich analytical toolset, for the following reasons:

- Users become more sophisticated.

- They do more probing of the data.
- They perform more complex analysis.
- They discover new uses for data.
- They discover new functions of the toolset.

As the benefits of the data warehouse becomes known and spreads, there may also be a growth from unanticipated users. The popularity of the data warehouse increases as it grows in business value.

2.4.2 Efficient usage

As users gain an understanding of the data and toolset, they become more proficient in finding the answers to their business problems, running fewer queries, and taking advantage of key performance structures in the database. These structures that improve efficiency include summary tables and indexes.

A phenomenon typical in data warehousing is that faster response time leads to more usage. Reductions in system resources brought about through more efficient usage and tuning are quickly consumed. White space does not remain available very long as users look to expand their use of the information in these systems.

2.4.3 Access refinement by users

A data warehouse offers more flexibility than OLTP for users to change the ways in which they access data. Immature users may request more data than they need, or receive more than they expect. With experience, there is less of a tendency to submit huge reports. Users feel more comfortable working with smaller amounts of data, allowing them to make better business decisions.

In some cases, summary tables or aggregate data accessed when initially using a data warehouse, may be used less frequently. With experience, users do more probing and drilling down into detailed information to identify patterns, trends or anomalies.

Therefore the initial usage patterns of a data warehouse may not be a good representation upon which to base capacity planning. Common to both OLTP and a data warehouse, whenever dealing with new applications and new users, a learning curve is to be expected.

2.4.4 Statistics

Statistics in the DB2 Catalog, captured and maintained via the DB2 RUNSTATS utility, help the DB2 Optimizer to determine what access path will minimize elapsed time. It is important that these statistics accurately reflect

the characteristics of the data stored on a logical table basis as well as a physical partition. Statistical data stored within the DB2 catalog includes:

- The number of rows in a table - referred to as the *cardinality* of the table (CARDF column in DB2 catalog)
- The number of unique values found in specific columns - known as the *column cardinality* (COLCARDF column in DB2 catalog)
- The percentage of the table containing certain column values - known as the *frequency distribution* (FREQUENCYF column in DB2 catalog)
- The number of DASD pages per table, partition, and index that represent physical DASD being used (NPAGES column for table or NLEAF column for index in DB2 catalog)

Using these and other catalog statistics with the SQL query, the DB2 Optimizer estimates the I/O and CPU cost to retrieve the data for different access paths, choosing the least-cost method that also minimizes the elapsed time.

By default, only the first column of an index has cardinality and distribution information, which is collected during a RUNSTATS utility of an index. To collect column cardinality statistics for all columns of a table, we recommend specifying COLUMN (ALL) within the TABLE option of the RUNSTATS TABLESPACE option. Sampling option of the RUNSTATS utility allows for statistics to be generated on percent of the underlying data. In large data warehouse environment, this can save valuable resources.

Frequency or distribution statistics which get stored in the DB2 Catalog can help the Optimizer be aware of data skew and influence the access path selected. Data skew is very common, especially when columns have default values or contain nulls. Starting with DB2 V5, RUNSTATS can collect frequency and cardinality statistics for combinations of keyed columns, or correlated keys. This can help the Optimizer be aware of more skewed data situations; however, RUNSTATS only collects statistics for the leading columns of an index.

The DSTATS utility is a supported program from IBM that will collect statistics on *any column the user specifies* and populates the SYSCOLDIST catalog table. We recommend that distribution statistics be collected on any column that is suspected to be skewed or frequently referenced in query predicates (selection criteria).

In a Star schema model (described in the appendix on data modeling in this redbook), it is very important that the join columns of the dimension and fact

tables have distribution statistics in order for the Optimizer to calculate accurate filtering factors and the sequence of tables to be accessed.

DSTATS requires DB2 V5 or higher. You can get DSTATS from:

<ftp://www.redbooks.ibm.com/redbooks/dstats/statistics.zip>

2.5 Query creation

Business intelligence queries are generally dynamic ad hoc queries, rather than static SQL statements. These dynamic queries tend to be more complex and require much more processing cycles than those in an OLTP environment. While a tuning analysis can be time-consuming, the potential reduction in hardware resources and elapsed time can also be significant. When tuning, it is important to understand the source of SQL statements that are executing in the system.

Currently the trend is for SQL to be generated by front-end tools, giving users less control to refine the SQL itself. GUI front-end products remove the need for users to even know the SQL syntax. Often these tools generate generic SQL to access a variety of databases, and as a result, it is often less than efficient. These tools are designed to allow users to submit queries by pointing and clicking. These tools may limit the opportunity to capture and optimize the generated query, depending on the application or tool design. However, changing the SQL statement may not be possible.

There are exceptions, such as with Web-based tools, where SQL may be written by developers and executed based on buttons a user selects on the Web page. Additionally, DB2 Connect, a popular product used to connect end-user tools to DB2 on the OS/390, has the ability to trace or capture the SQL being submitted. However, if tools generate the syntax, there is often very little that can be done to alter how the tool formulates the query.

2.6 User population

Capacity planning for the data warehouse environment is a unique exercise, because several factors make it more difficult than capacity planning for an online transactional processing system (OLTP).

The number of users for a data warehouse may be sporadic. Anyone with access to the data is a potential user. A rule of thumb (ROT) for estimating the concurrency level is:

- 10% of the total user population is signed on to the data warehouse.
- 1% to 2% of the total user population is active.

An example is illustrated in Table 1:

Table 1. DW concurrency ROT

Description	Percentage	# of users
Total users	100	2500
Signed-on users	10	250
Active users	1 to 2	25 to 50

The magnitude of user impact is more significant than witnessed in OLTP workloads. Generally OLTP users have very similar resource consumption profiles, and there is little if any variation between two users of the same transaction in a system.

However, with BI users, the user profile can be as diverse as the workload. Heavy users of the system, or *power users*, often submit very compute-intensive queries that can take hours to process. These users may make up 20% of the user base, yet consume 80% of the system resources. Recent trends in both data warehouses and Web-based OLTP include:

- Direct end-user access to data
- External customers having access to data
- Difficulty in identifying and segregating users

2.7 Data volume

The sheer amount of data contained in a data warehouse, often in the terabytes, impacts the capacity planning efforts for an application accessing and scanning those large data volumes. In most cases, the database of the data warehouse will be larger than any OLTP application within the enterprise. Additionally, queries in a business intelligence application scan more of that data, and the amount of data scanned grows as the data volume increases over time.

The data typically stored in a warehouse is a mix extracted from OLTP applications and from external sources. The warehouse has the same data spanning years (for example, all the sales for each store for each day for the past three years). This allows trending, pattern, and forecasting analysis by

the users. Requests for increasing the years of historical data accessible online also influences the growth rate.

Following are some design considerations, that apply to all platforms, which may reduce the affects of data growth:

- Partitioning
- Summarizing historical data after a certain point
- Table design: one table containing 5 years of data versus five tables, each containing one year of data

Make sure the requirements for data growth are understood so that an appropriate design is selected. Also consider the impact on processing requirements for requests to expand the data in a warehouse. For example, storing 10 years of historical data for tax or audit purposes should have little impact on the processing requirements if the database is properly designed. Data such as this may be accessed infrequently and by a limited set of users. This data need not belong in a table that is queried daily by a large user population.

The amount of data accessed varies greatly from query to query and is dependent on the access path chosen by the DB2 Optimizer and the query itself. The impact of doubling the amount of data can range from almost no increase in response time to twice the response time, depending upon the physical design. The increase may be more proportional to the number of DB2 pages actually requested by the query rather than the total size of the tables.

Similarly, you cannot judge resource consumption by number of rows returned. The result set may be an aggregate of many detail rows and not an accurate reflection of how many rows were accessed to obtain the result. *The amount of data accessed by the query to deliver the result is the key factor.*

2.7.1 Choice of model

When dealing with these large volumes of data, the data model is an integral part of every design as it impacts the performance of any warehouse. There are three common designs that are used in the many UNIX, NT, and OS/390 relational architectures: Entity-Relationship (ER), Star-Schema, and Snowflake designs. For more information on these models refer to Appendix B, "Data models" on page 161 in this redbook, and to *Data Modeling Techniques for Data Warehousing*, or reference the many books that discuss the topic.

In general, you find the Star schema used more often in BI applications than the ER model because it is easier for business users to understand. More importantly, front-end tools tend to be geared toward the Star schema design.

The data model chosen can have an impact on the SQL. Queries in the Star and Snowflake schema will generally involve more tables (frequently 7 or 8 tables and possibly more). This is due to the fact that the model is comprised of many small dimension tables. Optimal access path selection and tuning may be more complex.

2.8 Summarization of data

Aggregating data and building summary tables can have a dramatic impact in the reduction of system resources. The amount of data accessed is dramatically reduced from the volumes of detail data a query would need to access to construct the answer.

2.8.1 Summary table design

An OLTP application usually has no need for summary data, as transactions generally access and update the individual records in the detailed data. In a data warehouse, analysis frequently begins at a high level where data may be grouped or rolled up, based on key fields. Proper design requires a knowledge of the business query patterns.

Analysis to determine the subset of fields to include in the summary table, and analysis of the data to aggregate in order to significantly reduce the number of detail rows returned for frequently executed queries, are often performed by a database administrator (DBA) with the application analyst.

Summary tables can be easy to implement and their content evolves over time. Understanding user query patterns may not be known initially at the creation of the data warehouse. Patterns may also change, making summary tables an iterative process. Summary tables often need to be modified or dropped, and new tables created to meet the changing needs of the business.

Most toolsets today are proficient at aggregate awareness. This means they are aware of, and will create, SQL to access summary tables for the user, automatically. This eliminates the need for the user to know of the existence of summary tables and to decide when to use them.

Summary tables are commonly used in the many UNIX, NT, and proprietary systems that support BI applications today. Oracle, the DB2 family, and proprietary databases such as Teradata use these structures to improve

query execution time, while minimizing processor consumption. This is a common method of addressing system throughput that should be considered for any implementation.

2.8.2 Shifting CPU cycles

Its important to understand that the use of summary tables moves the consumption of CPU cycles from query processing to the maintenance window. This shift provides tremendous benefits when the summary data is able to be accessed by multiple queries throughout the user community. The cost of producing the summary table during maintenance is paid back each time the summary table eliminates the need to query large detailed tables.

Because the impact to the maintenance window may or may not be minimal, be cautious of overdoing summarization if this window becomes constrained. The overhead of creating a summary table can be justified by monitoring its usage.

2.9 Tuning

Once you have implemented a good data model for your application, you can then consider the option of tuning an application. While this is not required on any platform, some system architectures and designs can benefit from this effort more than others.

The effort an organization puts into tuning is dependent on the availability of skilled resources versus the cost of additional hardware. It is not necessary to tune, but if you choose to more efficiently utilize your capacity, you have the option to do so in a S/390 environment.

If you have chosen to invest in performance tuning, changes in access paths often provide a tremendous reduction in capacity requirements. When in capacity planning mode, you need to determine if that tuning effort has been done. If not, it should be considered as a conscious part of your process for high resource consumers.

With proper tuning, it is not unusual to reduce a query from *hours* of elapsed time to completion in *minutes*. This type of dramatic improvement is more prevalent in a data warehouse environment than OLTP. In general, OLTP applications tend to have consistent data accesses with very short response times, which are less complex and easier to tune than BI workloads.

In OLTP applications, less data is retrieved and a direct index lookup is usually performed against the database. SQL with a bad access path that

scans a tablespace is more noticeable in the OLTP workload and is brought to the attention of a database administrator (DBA) quickly. The type of access path seen in BI is usually different from what a DBA sees in OLTP.

In a BI workload, tuning can have a dramatic impact on resource consumption. However, it can be difficult to identify queries that can benefit from tuning due to the wide variability of the query response times within the BI workload. Most query performance challenges are due to poor access paths or the volume of data requested. A majority of CPU savings is realized from tuning access paths, in some cases, by orders of magnitude.

Adjusting system resources such as bufferpools, EDM pools, or work space, may provide 5% to 10% reductions in processing times of a query. Therefore, effort should be focused on tuning access paths to reduce system resource consumption. Queries can be reviewed during testing using the EXPLAIN function to view the access path. This gives you an indication of what production performance will be like, and provides the opportunity to change the access path for efficiency. The following section discusses the additional focus areas to consider when reducing resource consumption in a BI workload.

2.9.1 Awareness of table size

In data warehousing, it becomes imperative to be more aware of the volume of data being accessed by a query. It is critical to have developers work together with DBAs and gather these metrics for an application. Together, they should estimate the number of rows being accessed by the most frequent queries, and those they expect to be the largest queries. Through this process, DBAs develop an understanding of the size of the underlying tables and the impact on resource consumption.

2.9.2 Indexing

OLTP database designs frequently minimize the numbers of rows a query accesses with non-partitioning indexes (NPI) or secondary indexes. Using indexes reduces the amount of I/O, thus minimizing the query response time.

Indexes can have a tremendous impact on improving response time in a business intelligence workload as well. A query can be reduced from running for many hours down to running for a few minutes. It is not feasible, however, to have an index satisfying every individual query's needs. Index maintenance increases with the number of indexes and may impact the batch windows of an application. Therefore, indexes must be developed that can accommodate *queries that will run frequently or queries with similar search conditions*.

When designing a database in a data warehouse, we recommend that you try to have tables that may be accessed together (joined) have the same or similar partitioning index. By definition, the partitioning index also sequences or clusters the data. As a result, when tables with similar partitioning indexes are joined together, synchronous random I/O can be minimized because rows in each table are stored and processed in the same order.

2.10 Summary

The points in this chapter can be summarized as follows:

- Do not blindly accept the resource usage of queries.
- Brute force parallelism is not the only answer.
- Make efficient use of resources.
- Check if tuning through DB2 Catalog statistics or indexes is possible.
- Proper tuning can have a dramatic and beneficial effect on system utilization.
- Data volumes and the number of users can impact resource consumption.
- End-user tools can generate inefficient SQL.
- Consider the value of summarization tables to improve query processing, while evaluating their impact on the batch window.

Benefits of the robustness and maturity of DB2 and S/390 are the options available for tuning the environment. Multiple tuning knobs exist, providing flexibility that can delay the need to purchase additional hardware.

2.11 Checklist

The following is a brief list of questions that may be helpful when considering capacity planning for a data warehouse environment.

Question	Comments
Have performance expectations been defined for both the query and batch maintenance workloads?	Requirements for query response time, critical batch window, maintenance window and concurrency all help to determine hardware requirements.

Question	Comments
Have application processing volumes been estimated?	The ability to meet processing volumes for both workloads is critical to achieving SLAs or informal user expectations.
What use will be made of aggregated data?	The use of summary tables can greatly improve query performance.
Is OLAP functionality or data mining part of the application?	Processes to build cubes or mine large volumes of data tend to be intensive and performance expectations should be reviewed with the business user and technical support staff.
Is the processor configuration optimal for query parallelism?	To achieve any CP parallelism, there must be at least two engines available to the query workload.
Are there sufficient paths/control units to meet I/O bandwidth requirements?	BI applications often process large concurrent queries with high data volumes that may place heavy demands on the I/O subsystem.
Is the OS/390 Workload Manager implemented in goal mode or compatibility mode?	WLM goal mode is highly recommended to enable resource group allocation of CPU resources and dynamic management of DB2 stored procedures and Web address spaces.
Are you using DDF thread support?	DB2 V6 introduces Type 2 inactive thread support, which uses less virtual storage than before.
How will you manage the query workload?	BI query workloads are inherently unpredictable in terms of complexity, run time and the number of concurrent queries. Use of WLM Period Aging and explicit prioritization is recommended to deliver good, consistent query performance in a mixed workload environment. WLM policies can also prevent queries from monopolizing resources.

Chapter 3. Data maintenance

This chapter discusses the maintenance, or batch workload, that is part of a business intelligence solution.

Business users generally require that the data within a BI system be kept current through periodic updates and refreshes. This update processing can run either nightly, weekly, or monthly, depending on the application design. In many cases, the update processing may need to complete before the end users are able to access the system. In VLDB (Very Large Database) warehouse environments, this batch workload is often resource-intensive. It can even be the primary driver for CPU requirements for a system; therefore, it is critical to review the batch processing requirements when conducting a capacity planning activity for a BI workload.

Data maintenance includes all the batch processes that need to be done in a BI environment, including:

- Data population
- Database utility processing: backup, recovery, reorg and runstats
- Data archiving and purging
- Summary table maintenance

Understanding the processes within the maintenance workload allows us to identify the critical path of work which must finish within the batch window. This chapter focuses on describing those processes, and in Chapter x6x, we will discuss a methodology for estimating the capacity requirements for this workload.

3.1 Data population

Data population includes all the steps necessary to add data to the business intelligence data store, starting with the extraction of data from operational systems, mapping, cleansing, and transforming it into a form that is suitable for the BI application, and finally loading the data in the BI environment. All these steps together are usually referred to as the ETL process: Extract, Transform and Load.

Extract

In order to update the BI applications data store, it is important to know what data is required from the operational system. Then, programs can capture any additions or changes to the data in the source system. There are many

options used to extract data from operational sources, such as total extracts of all the operational data, incremental extraction of a subset of data, or propagation of changed data.

These extraction runs might only be possible when the source system is not actively using its data. This window when the source application is offline may also need to accommodate other housekeeping functions for the source system as well, such as backups, reorganizations, or other necessary tasks; this can reduce the availability of the data for the extraction processes.

The extraction processes execute on the system where the source data is located, which may be a separate system than the one housing the BI solution. Understanding where these tasks run is important when conducting a capacity planning effort.

Transformation and cleansing

Business intelligence applications combine data from many independently developed systems, so the data must be transformed and cleansed to ensure that the data warehouse will have valid, consistent, and meaningful information. In this process, data that has been extracted from the source systems is standardized to a common format with consistent content. The transformation process is often very I/O intensive, and involves things such as:

- Converting data in different formats to a single consistent format.
- Data consistency checking to correct or remove bad data, such as misspelled names, incorrect addresses, and other inaccuracies.
- Detecting changes in the source data over time.

These processes need to be performed each time data is extracted from the source system, and as the number of source systems increases, the complexity of this task multiplies. These processes may run on any system where there is available capacity, and are not required to run on the same image with the business intelligence solution.

Load

The *load* component encompasses those processes that directly add the data to the data warehouse. Loads are periodically processed to keep the data in the system up-to-date: either nightly, weekly, monthly, or as often as necessary. Once a file is created from the records to be loaded, they can be sorted into partition key sequence, then split into multiple load files to exploit load parallelism. You can also run multiple DB2 utilities in parallel to fully exploit the OS/390 server and I/O subsystem bandwidth.

Generally, the load step of the ETL is performed with either a database load utility, data propagation technology, or by inserting or updating the data in the target table with SQL statements. Usually, the load step is processed during the batch window for the BI application, which is the time the online query workload will not be processing against the data. In a S/390 environment, load processing is work that is processed by the DB2 for OS/390 image that manages the BI data store for the workload, so it is part of the capacity planning for that system.

ETL capacity planning issues

Capacity planning for the ETL process must include the time constraints for extract processing, as well as data transfer times. If the source data resides on a different platform it may be necessary to transfer the data through the network. In that case, you should evaluate the capacity and available bandwidth of the network to accommodate the traffic.

The ETL process mixes a variety of methods to complete the tasks, such as:

- Sorts
- Row-oriented, hand-crafted COBOL programs or equivalent 3GL code generated by tools such as DB2 Warehouse Manager (formally as known Visual Warehouse), ETI or Vality
- Set-oriented SQL transformations in script languages
- Data Propagator
- Database utilities

If you use row-oriented 3GL programs, you can capacity plan for the transformation process using techniques described in *DB2 for OS/390 Capacity Planning*, SG24-2244 and *OS/390 MVS Parallel Sysplex Capacity Planning*, SG24-4680. Set-oriented SQL transformation processes may also lend themselves to these techniques if batch growth projections are reasonable. However, if the growth in the batch workload is anticipated to be orders of magnitude larger, we recommend that you use DB2 Estimator to provide data for the capacity planning process. DB2 Estimator is discussed in 7.4, “Method 3: Modeling with DB2 Estimator” on page 117.

The frequency of the ETL data population process varies with different environments and end-user departments. This process can execute monthly, weekly, or nightly. In some cases, the users may require updates to occur throughout the day. Understanding the requirements of the users, and how often the data must be updated, allows for proper capacity planning for the workload.

Often, the requirements for the data population process change over time, as end users require more timely access to their data. This can alter the

frequency with which this process must execute. Understanding both current and future requirements for this process is key to properly planning for the needs of the application. This topic should be discussed with each end-user department on a regular basis, to qualify how their data currency needs may be changing.

3.2 Database utilities

Database utilities perform some important data management functions for the business intelligence environment. These include backing up the database for recovery, as well as maintaining query performance by periodically reorganizing the data, and keeping the database Runstats up to date. All these functions typically execute on the system hosting the BI application, often when the data is offline to query processing.

These DB2 utilities require the same types of system resources as the query workload:

- CPU
- I/O
- DASD
- Storage

The greater the number of rows and columns, the more CPU needed to load, back up, and reorganize the tablespaces. BI environments are characterized by large volumes of data, so it is necessary to analyze the resource consumption of all the utilities when conducting a capacity planning study. When dealing with very large databases, exploiting parallelism in DB2 for the various utilities will reduce the overall elapsed time for the maintenance job stream. DB2 Estimator considers parallelism, and therefore can be used to determine the impact on each utility when the data is projected to grow.

The utilities that are frequently used in the maintenance workload include:

- Backup/Recovery
- Load (discussed earlier)
- Reorganizing the database
- RUNSTATS

Backups

Backups execute regularly, often after every population process, to ensure there is always a current copy of the data. Backups can execute concurrently with the query workload, so they are not a significant consideration when factoring the length of your batch window. When the installed disk includes features such as Snapshot or Flashcopy, you can totally disregard the time

necessary to complete this task, as these features are designed to eliminate application outages for backing up data.

Reorganization

As data is added to a table over time, it can become physically scattered through the partitions. *Reorganizing* the database encompasses the unloading and reloading of the data into a table to re-sequence it, thereby improving query performance. The frequency of data reorganization needed in BI environments can be minimized with the use of time-partitioned tables. With this database design, new data is added at the end of the last partition or in separate partitions. Thus, you only need to reorganize the data if the updates significantly change the clustering of the data within the tables.

In the past, reorganizations of the database were scheduled during the batch window. However, as an alternative with DB2 for OS/390 V6, you have the ability to reorganize the data within the partitions, while the database remains online for query users. The reorganization process takes longer using this method, but since there is no application outage the greater time period is irrelevant to the users. This feature should be looked at when you are running V6 or a higher version of DB2. This feature effectively removes this task from the batch window processing.

Recovery

Recovery processes are not part of the regularly scheduled workload for any environment. However, you should consider how are you going to recover the data in case of a disaster. You should perform the same tasks as you have for your production workloads. What is unique is that when running this workload on a S/390 server, you can contract for recovery services as you have for all your other production workloads.

RUNSTATS

In older DB2 versions (V5 or earlier), it was necessary to execute the RUNSTATS utility to update the statistics stored in the DB2 catalog following a LOAD or REORG. These statistics are used by the DB2 optimizer to efficiently access data within the database.

Now, DB2 version 6 allows the statistics to be gathered during the execution of some utilities, such as LOAD, REORG, and REBUILD INDEX. Using inline statistics eliminates the need for separately executing RUNSTATS after the utilities finish; however, it also increases the CPU capacity requirements for the LOAD, REORG, or REBUILD INDEX function, as two processes are being done at once. The total capacity requirements for completing the RUNSTATS inline with another function are approximately equivalent to executing the two

processes separately. RUNSTATS for new partitions or tables should be run, as the information is critical for queries to execute efficiently.

3.3 Archiving/purging old data

In a BI environment, data is constantly being added to the database on a regular basis. This results in significant volumes of data online, often in orders of magnitude more than in the traditional OLTP/batch environment. Today, even medium-sized enterprises are implementing multi-terabyte data warehouses.

Users generally require a finite window of data for their analysis: three to five years meets most requirements of users. (However, this should be validated with the business user population and has sometimes stretched to 10 years and longer.) For most companies, once you have accumulated the historical data needed for the application, as data continues to be added, older data needs to be removed or archived from the database.

In the past, archiving of older data generally ran infrequently for small business intelligence systems. However, in a VLDB warehouse environment, the archive process is necessary to keep the volume of data in check. In such large environments, this process may touch a significant volume of rows. With constrained batch windows and extremely large volumes of rows to archive (for example, tens of millions to hundreds of million of rows is not uncommon), a plan is needed for the archiving process; if older unused data is not removed, the volume of data will continue to grow.

Depending on the database design and the database managements system's storage architecture, constant data growth may result in linear (or worse) increases in query elapsed times and system resource consumption. In addition, if data growth is not checked, the normal batch window may not have sufficient time to complete the *regular* maintenance process for the application, let alone the archive process itself.

The advantages of archiving/purging old data are:

- Reduced storage costs
- Storage is reused for new data
- Queries scan less un-requested data
- Less data means overall improved performance

When designing the archiving/purge process, you should consider the following questions:

- What data must be archived?

- Does a logical archive already exist, so that data can be simply purged?
- When is the archiving process going to be done?
- How much data are you going to archive?
- How long will this process take?
- How are you going to perform the archive?
- How long do you keep the archive?

As an example, imagine that only the current year of data is actively being queried, so you remove all the data older than one year from the active data warehouse. First, you should determine if the older data still has value for the business. If not, you can delete it without backup. However, if it does have value, you must move the data from the active warehouse to another media, such as tape. It is also important, prior to archiving, to back up the database, tablespaces and table descriptions from which the data came. If you don't, and the table structures change, you may not be able to quickly restore the old data when you need it.

Another type of archiving consists of archiving only *aggregations* of the data. With old data, the detail becomes less useful, so some companies aggregate the data, remove the detail and thus reduce the volume of the data, and then archive only the aggregation.

A practical example of archiving data is illustrated in Figure 5 on page 44. This method assumes a time-based (Year-Quarter) range partitioning scheme for partitioning data that isolates particular quarters of data to specific ranges of partitions of a table. This avoids having data for a given quarter be scattered throughout the table as in a hash partitioning scheme, and reduces the hardware resources needed to locate the candidate archive records and remove them.

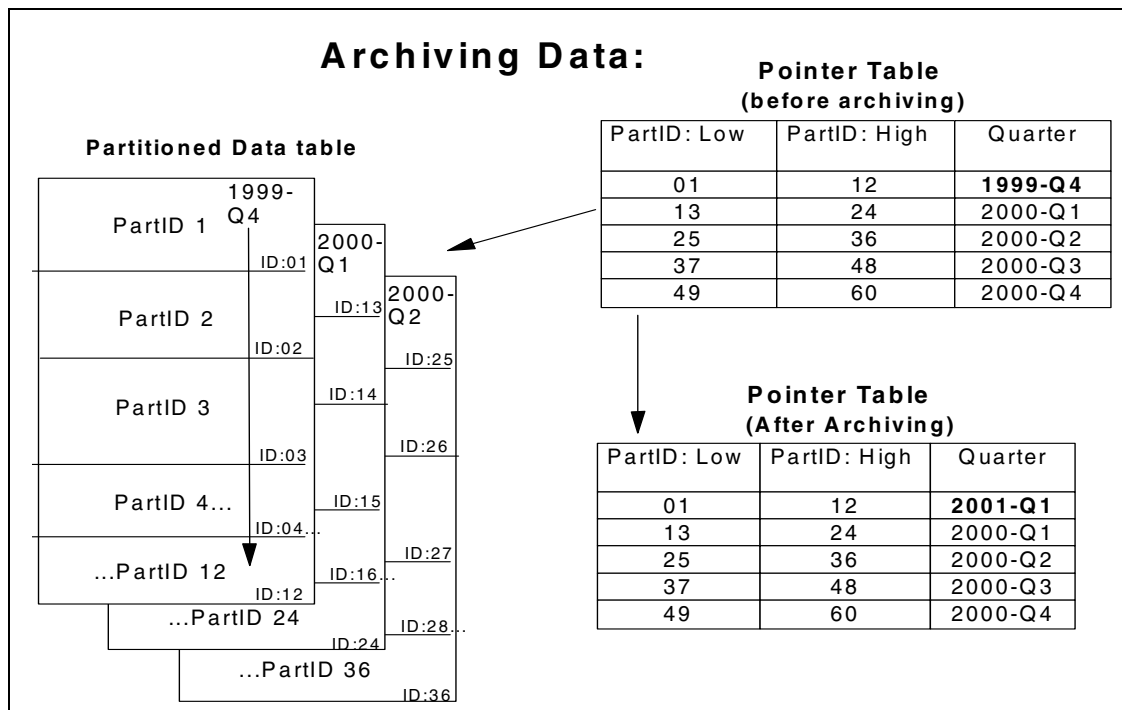


Figure 5. Automatic data archiving

This figure illustrates a common method of simplifying data archiving. Here, data is stored in a large table that has been broken into *segments*, or partitions of smaller portions of the data. Each year's quarter of information is stored in ten segments or partitions of the table.

In this example, the customer requires one year of active data (4 quarters), where they differentiate between the active quarters within that year. One additional set of partitions, above those required to handle the active data, is available to support loading of the current quarter's data while the oldest data has not yet been archived.

In this figure, Partitions 1 to 12 contain data from the fourth quarter of 1999, which is the oldest data in this table. The data for the first quarter of 2000 is spread through Partitions 13 to 24. The data for the second quarter of 2000 is in Partitions 25 to 36, and so on.

The columns in the pointer table indicate the first and last partition identifier (PartID) for the quarterly date span and the year/quarter of information within that partition. When it's time to archive data, the oldest set of partitions are

easily identified by checking for the earliest date in the Quarter column, and the data is archived to tape. The newest data can be immediately loaded into the archived partitions with a LOAD RESUME REPLACE PART function of the DB2 LOAD utility. The partitions are logically emptied in seconds, without any need to do potentially costly DELETES of the archived/purged data in those partitions.

Using the small partition pointer table as an indicator, shown at the right of Figure 11, the DBA and application development group can automate the archiving process with the following steps:

1. Archive, as needed, the oldest partitions to tape using the DB2 REORG UNLOAD EXTERNAL utility.
2. Prepare the new load file with data to be added to the table. During the Transform step of the ETL process, assign a partition ID in each load record such that the pointer table's quarter and partition ID will correspond to the record's quarter and partition ID. These are the partitions that were emptied in step 1 of this process.
3. Load the newly transformed data into these partitions (1 to 12 in this example). These partitions can be loaded in multiple parallel jobs, rather than using a single serial load job to shorten the load time.
4. Update the partition pointer table to reflect that Partitions 1 to 12 contain data for 2001-Q1.

It's important to identify both your archiving process and *when* it executes in the batch window. Understanding when this workload processes is important for overall capacity planning. The time required to execute this process is impacted by data growth.

As new data is added to the database for new applications, or when there is an increase in the number of years of information stored within the data, it will impact the time required to archive data from the database, impacting the batch profile that is discussed in 6.5, "Maintenance workload growth: Worksheet 4" on page 100.

3.4 Summary tables

Most data warehouses have summary tables that are calculated, or re-calculated, once the detail or atomic level data has been populated. Summary tables generally contain the results of frequently asked questions. They can reduce much of the processing time for queries that are frequently executed by the business users, because the results are pre-tabulated in the summary tables.

As a result, the queries do not have to read millions of rows each time to derive the same results over and over for every user asking the same question throughout the day.

As the data grows, the process of building the summary tables may become more complicated and require more resources, especially if historical summaries need to be rebuilt. If your data warehouse is being continually fed from the operational workload, the cost of building and maintaining a summary table can be very high.

Automatic Summary Tables

The newest releases of relational databases have a new function that keeps summary tables synchronized with detail data through Automatic Summary Tables. As the detail data is changed, that change is reflected into the summary table. This feature is important when updates are trickled into the warehouse as it remains online. In environments where Summary Tables are recalculated at the end of a batch update, it would not have a bearing.

A common change to data within a database occurs when an item is *reclassified* or *moved* from one department to another. For example, a product may shift from one category to another (say, women's dresses to women's sport clothes), forcing the summary table to be completely rebuilt.

A change like this may seem trivial, but it can have a significant impact on the processing cycles on the night that change is made. This will produce unpredictable spikes in processing requirements if the end-user department does not advise the systems personnel of the change. Make them aware of the processing considerations, so they can help you plan for these events. This is best determined during discussions with the end-user departments. In particular, questioning them on reclassifications, and when or if they might occur, can help identify what added resources, if any, may be needed.

It is important to identify the workload associated with updating all the summary tables in the normal population process, as well as when they need to be rebuilt.

3.5 Factors influencing the maintenance workload

Once you've identified the jobs that perform your current maintenance processes for the BI environment, you can estimate the impact that growing the workload will have on your environment. The two factors that will impact the batch workload are *factors changing the length of the batch window* and *data growth*. We discuss these in detail in the following sections.

The batch window

The batch window is composed of:

- The *transfer* window, which is the period of time when there is no, or insignificant, database updates occurring on the source data, and the target data is available for update (target data availability may not matter).

This includes the time needed to transfer the source data from the operational system to the business intelligence environment. This also includes any time needed to move data from a centralized data warehouse to distributed data mart environments.

- The *batch processing* window, which is the time the system is available for processing current source data, either exclusive of or concurrent with online queries. The period begins with the time the source data is available for transformation and load processing. Depending upon the methods used for loading the data, end users may or may not be able to access existing data for query processing.

Optimizing the processing flows during the batch window is often a critical activity in a BI environment.

The batch window in BI environments often is more constrained than in OLTP environments. In a OLTP system, if you cannot complete all your batch processing during the batch window, you can often reschedule the jobs for the next batch window, with minimal impact on the users.

However, in many BI environments, the reporting and analytical business requirements often need the nightly addition of the prior day's operational data, so you cannot start your queries until your batch processes are done. It is very important to know the time period of the actual batch window, and how it is going to be affected as the workload grows, in order to ensure that you have enough time to complete all the required batch processes.

The relationship between data volume and batch window size influences the design of the data update process. Figure 6 on page 48 shows the alternatives for updating the data.

In Figure 6, the length of the batch window is reflected by the x-axis (the length of the line), and the amount of data to be processed is reflected by the y-axis, (the height of the line). Each line represents a different batch maintenance example.

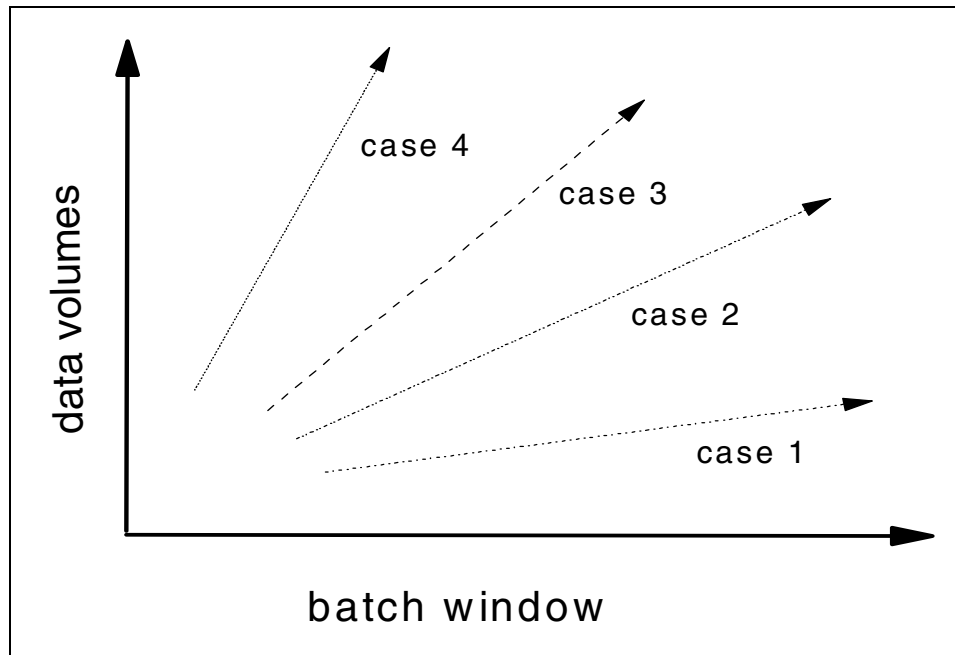


Figure 6. Data updating design alternatives

Examples

Case 1: There is no batch window constraint. In this case, there is very low volumes of workload, and enough capacity (CPU and disk) within the window both on the source and the target side. We recommend using the simplest and most familiar solution to process the batch workload.

Case 2: In this case, the batch window is shrinking relative to the data volumes. Process overlap techniques usually work well in this situation.

Case 3: In this case, the batch window is very short compared to the data to be moved. One solution to this problem is to move only the *changed* data. This approach can be supported by tools such Data Propagator Relational. In this situation it is probably sufficient to apply the changed data to the target once a day, or perhaps, throughout the day.

Case 4: In this case, the batch window is essentially non-existent. It is necessary to move the changed data through continuous data transfer, near-synchronous update of the operational data and the warehouse data. Minimizing response time and throughput impacts to the existing operational systems is a requirement.

Depending upon the volumes, tools such as Data Propagator or user-written code may be used. The design and construction of the batch processes are critical success factors needed to meet processing requirements effectively, with minimal impact to existing operational systems as well as concurrent query activity.

Understanding the approach implemented for an application will allow you to predict the impact of application growth on meeting the business windows.

Data growth

Maintenance workloads are only affected by growth in data, and in a BI environment, this dependency is very strong. Query growth generally has no impact on this component. As the data grows, the maintenance process starts to require more CPU and the elapsed time.

To avoid batch window constraints, you have to determine how your data will be growing and estimate the new maintenance workloads. This information allows you to improve the population process design to maximize the CPU usage before your batch window becomes too short. Some population designs that can help you are:

- Parallel solutions
- Piping technologies

You can find a detailed description of these techniques in the IBM Redbook *Building VLDB for BI Applications on OS/390: Case Study Experiences*.

For further description of data growth and its impact on the batch workload, refer to Chapter 6, "Growing existing workloads" on page 87.

3.6 Summary

This chapter provides an overview of the processes and issues related to the data maintenance associated with a business intelligence solution. Because accessing the data is important to the end-user community, having current and updated information within that system is equally important.

To ensure the data is updated, you should plan for the resource requirements to handle that process. This chapter outlines the steps of the update process,

and describes issues related to these business processes. It also addresses other, related processes that should be considered when a maintenance workload is being grown.

At this point, we've covered the two primary workloads associated with a business intelligence environment. The following chapters present methodologies for capacity planning for this environment.

Chapter 4. Characterization of existing workloads

In this chapter we describe the ways that business intelligence workloads differ from typical workloads in the S/390 environment, and how the unique characteristics of BI workloads affect capacity planning for your applications.

4.1 Capacity planning for a business intelligence workload

To accurately characterize workload's resource requirements, you need to consider all the architectural components implemented in the solution. Business intelligence applications tend to share common architectures. Many functional solutions feature a three tier logical architecture, where the user has a PC-based tool on the client workstation that connects to an application server, as depicted in Figure 7 on page 52. The application server then connects to the data server (data store) that can be on the same physical system as the application server, as shown at the top of Figure 7, or a completely separate physical server, as seen in the bottom figure. From a capacity planning perspective, you would need to consider each of these components.

In some cases, business intelligence solutions feature a two tier logical implementation where an end-user tool, such as QMF, has direct access to the raw data in the data store. This approach is also a common architecture for BI solutions, where business functionality for applications such as Customer Relationship Management (CRM) or Supply Chain Management (SCM) is not incorporated into the solution.

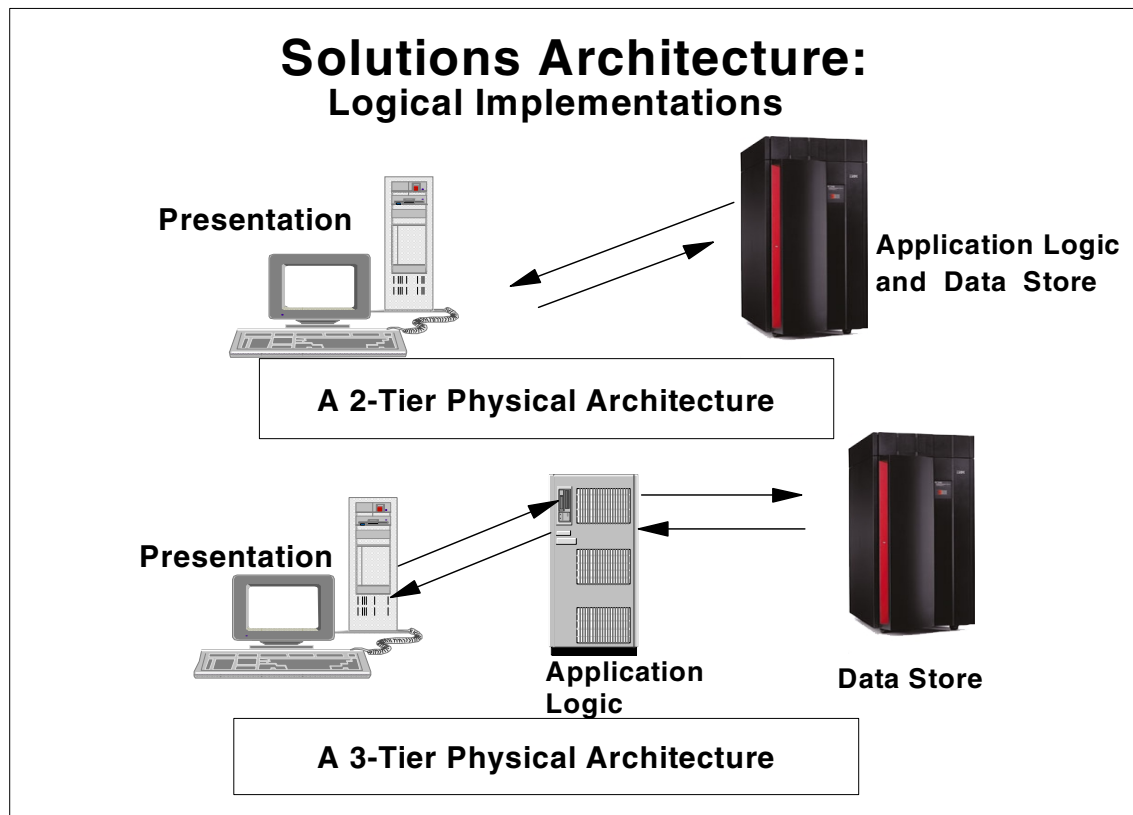


Figure 7. Business intelligence physical solution architectures

The client workstation requirements primarily depend on the end-user tool, its associated prerequisite software, and the connectivity software needed to access the application server. This generally remains constant over the life of the end-user tool. Software upgrades may alter the product requirements; changes to prerequisites should be considered as newer versions of the tools are released.

When an application server is a component in the solution, it is frequently a component offered by the end-user tool vendor to centralize many of the functions onto a single server. Capacity planning for the application server is dependent on application vendor information, which often associates processing capacity requirements to the number of users. Generally, this is a linear metric. The application servers generate the SQL statements which are then executed on the data server. Once the application server receives the answer set back from the data server, it is returned to the client workstation.

In many cases, very little processing occurs on the application server itself; however, this varies from product to product. Information to properly plan the capacity requirements for this component should be provided by the application software vendor, or if the application software is written in house, the development team should provide this information. If the application server is running on the same platform as the data store, as shown in the two-tier architecture in Figure 7 on page 52, the application server growth should be planned for separately, using the guidelines provided.

4.2 Key characteristics of a workload

To plan for added capacity in a business intelligence environment, it is first necessary to determine the current resource consumption of the workload, and to relate it to variables of the workload. The metrics to consider for a workload include the users, the number and type of queries they generate, the processing capacity they require, and the amount of data stored and accessed in the system.

4.2.1 Workload profiling

A key aspect of capacity planning is understanding the business importance of an application. This is most critical when there are diverse workloads on a system; it is necessary to prioritize the work in the system to ensure that the work is processed in an order that reflects its business priority. From a capacity planning perspective, you also want to ensure that when resources are constrained, the highest priority work is favored.

In a BI environment, this may mean differentiating between queries from different business units, or between individuals within a business unit. For example, if the Vice President of Marketing submits a query, you would want his work to take priority over long-running queries that are not required in the next 10 hours.

Capturing these relationships between business units that share access to the data, as well as within the business units, is key to prioritizing the work in the system and knowing when to review the capacity demands on the system.

4.2.2 Profiling system usage of a workload

When planning the capacity requirements for any workload, it is critical to understand its resource consumption pattern over time. This high level analysis is done to understand the business processing cycles of the application. Mapping the capacity requirements of an application over time lets you identify the peak and average resource consumption time periods for

this application. Gather detailed information about the application at selected peak times, then use these details as the basis of your projections for future growth.

Generally, to produce a credible capacity forecast, you should plan for the peaks rather than the “average” times of resource consumption. With a DB2 workload, it is important to determine not only the peak for query processing, but for maintenance processing as well. Selecting the length of the time interval you will be studying can be as significant as selecting the specific period of time to analyze: is it a peak month, week, or specific days that is most important? Will it be fifteen minutes, an hour, a week, or a month? It is important to select a sample that is statistically valid, as well as an interval that covers the critical work period for the end users.

When selecting this measurement interval, it is important to use the time span in which the critical work in the system must complete. If you have an environment where the operational (query) window is 10 to 12 hours, then this would be a valid measurement interval to ensure that the work executing within that time frame achieves the required performance goals. This also applies to the maintenance workload. The length of the maintenance window, whether for a day or a week, would be an appropriate time frame to study.

4.2.2.1 Average/peak utilization

A common concept in capacity planning is the peak to average ratio for processor utilization. During any measurement period, the workload fluctuates. The processor requirement for a sample period varies with demand and is composed of higher and lower levels of processor utilization. The peak to average ratio is the ratio of the highest value in a sample, compared to the average value for that same sample period. For example, in the upper graph in Figure 8 on page 55, the processor utilization for a sample period averages about 60%. However, the peak value approaches 100%, and it is during that time that the processor resource approaches being constrained. At this point, there is a higher probability of processor response time issues and there will probably be work in the system that is delayed in its processing (latent demand). This can be a critical issue in OLTP environments.

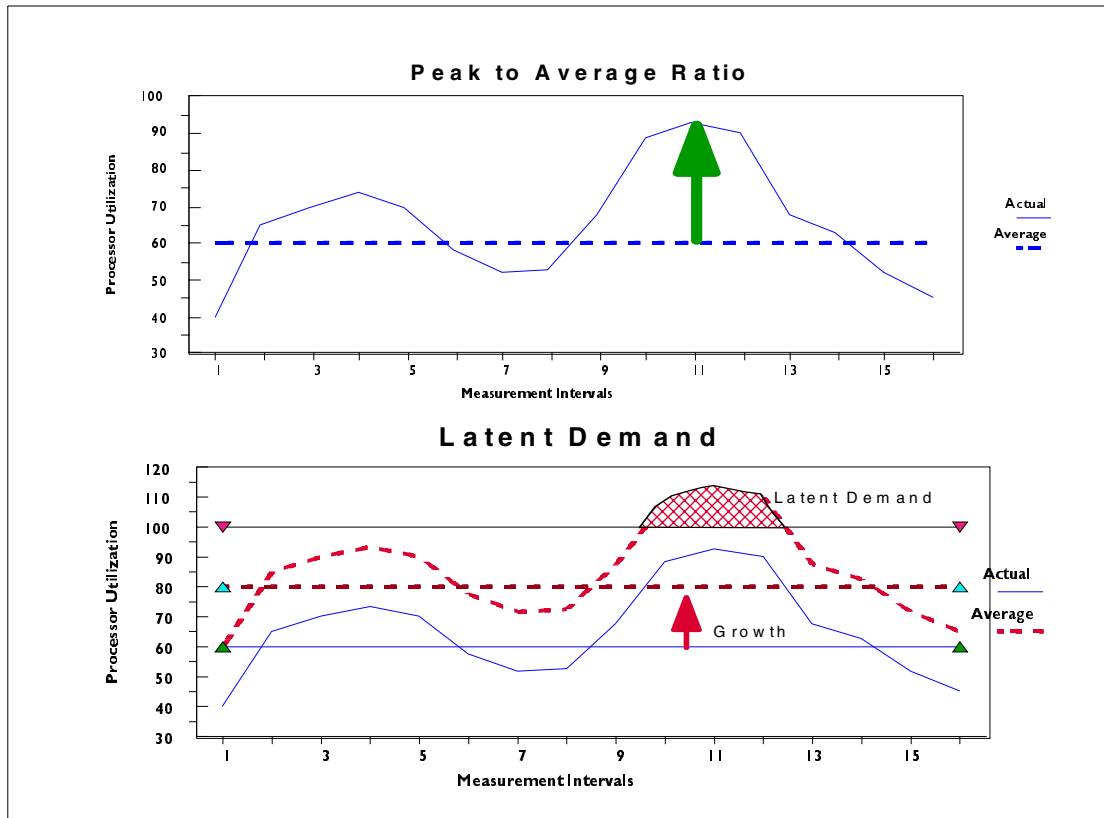


Figure 8. Typical Peak to Average Ratios, and Latent Demand for OLTP

However, as the amount of work in the system grows over time, the average processor utilization increases. If the average increases to 80%, as shown in the lower graph in Figure 8, the curve shifts up to the dotted portion of the chart, and exceeds the capacity of the installed system. In reality, work in the system is delayed and executes over longer periods of time, using system capacity that becomes available later on. When doing capacity planning, it is important to understand this ratio of peak to average, to better understand the latent demand in the system and more importantly, its impact on the users.

With production OLTP systems, this is handled by properly prioritizing the work in a system at the appropriate application level—such as the CICS address space running the order entry system and the IMS address space being associated with inventory and shipping. These functions are more critical than the general TSO users and batch jobs that are in the system.

This ability to prioritize workloads has been a differentiating characteristic of OS/390 systems for decades.

In contrast, business intelligence systems are highly erratic, often running at high levels of utilization (often 100%) for extended periods of time. In this case, it is important to leverage the Workload Manager capabilities to manage the query flow in the system through the proper implementation of workload manager goals. WLM can be used to implement period aging, which permits short queries to flow through the system, while long running queries are reduced in priority until they effectively become background (batch) work. Setting the appropriate priorities for different users allows WLM to identify the important short queries and prioritize them over other short running queries from less critical users. An effective implementation of workload manager allows customers to manage latent demand in the system without incurring response time issues. The important work is identified and executed at a higher priority, and the less critical work in the system experiences longer execution times. An upgrade is required only when the critical work can no longer complete within the parameters defined in the system goals, or when low priority work is never being executed.

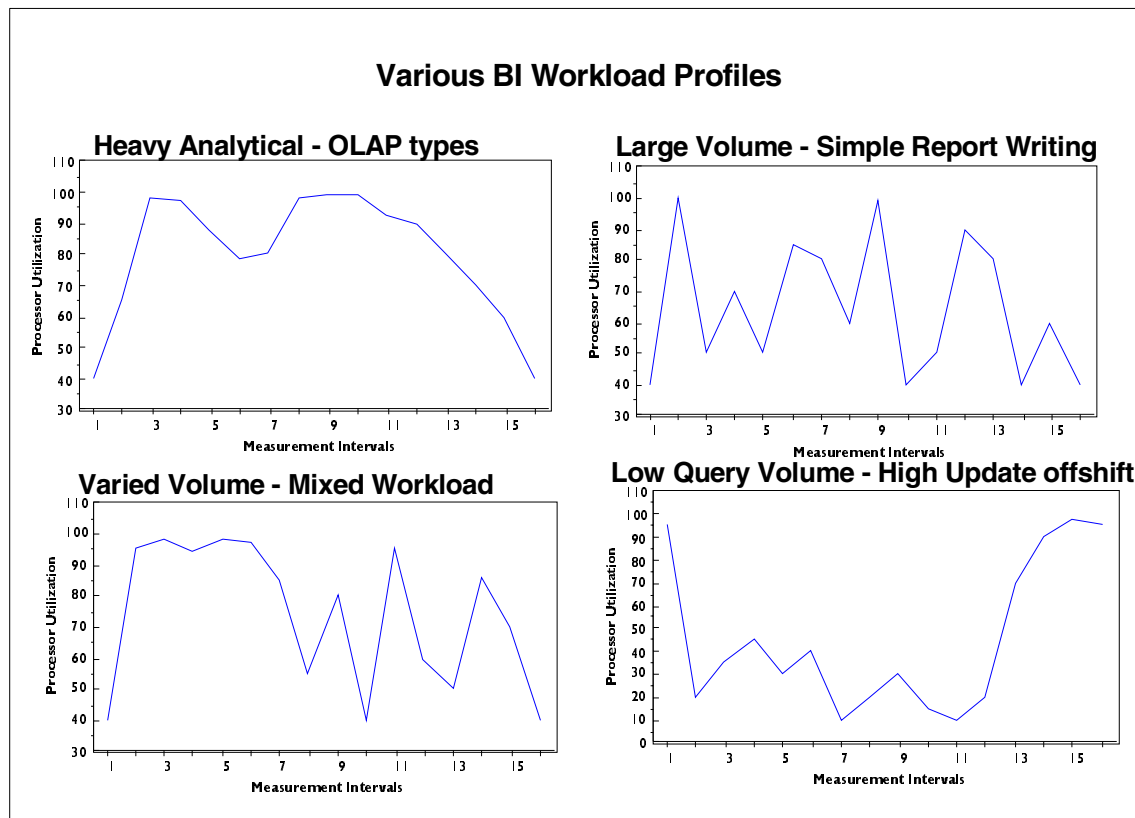


Figure 9. Various CPU utilization profiles of BI workloads

This workload capacity profile should be understood for any BI workload, just as it is for critical OLTP workloads. This analysis will identify the days in the week, month, or year where the business intelligence workload undergoes its highest capacity demands, and where it is important to determine if the service level agreements (SLAs) for that workload are being met. This level of understanding is developed by ongoing monitoring of the environment to determine how resource consumption relates to the business cycles of a workload.

The four charts in Figure 9 illustrate the diversity of profiles that can exist for various types of BI workloads. The x-axis represents a period of time (day, week, month, year), while the y-axis represents the CPU utilization at each data point in that time sample on the chart. This type of chart allows you to understand the normal activity levels for the system and when the activity deviates from these norms; this is important information for capacity planning.

4.2.3 Profiling data usage

Another key characteristic of the workload is the profile of data use associated with the application. Data volume and the data access patterns are critical in a business intelligence environment. Such factors as how the data is stored, the number of indexes providing access, and summary tables created to reduce repetitive processing of data, all impact how the data is accessed. Establishing a clear picture of the patterns of access over time will help in developing an understanding of when the largest volumes of data are needed, and what data is required. Generally, processor and data requirements should peak within the same time interval.

When considering data, it is important to understand some of the common terms related to it. *Raw data*, as it applies to BI workloads, is the amount of actual application data stored in the relational database. It does not include DB2 constructs such as indexes, catalogs, summary tables, and so forth. It is the only real data number that can be used to compare the data between systems on different platforms.

Raw data is quite different from the physical disk used. *Physical disk used* is the amount of physical disk installed to handle all the raw data, DB2 structures related to that data, and any disk space required to mirror the data (S/390 environments generally do not mirror their data). It includes all the disk capacity necessary for work spaces for users, sorts, utilities, DB2 catalogs, and summary tables. This value does not include space for operating system and database executable code and the associated maintenance libraries.

Raw data and used physical disk values are necessary to compute the ratio of physical disk required to support each gigabyte of raw data. This is an important ratio for a business intelligence system on any platform. Each architecture and environment has its own unique ratio of raw data to physical disk that is used to gauge the amount of physical disk necessary to run an application.

When capacity planning for additional data, the anticipated increased volume of data is multiplied by the disk ratio to obtain a rough number of the physical disk space that must be installed to hold the data. On S/390, customer production profiles typically range between 1.5 to 2.7 gigabytes of physical disk for every 1 gigabyte of raw data. This ratio varies depending on indexes and sort work space, and other application-dependent features. This ratio should be calculated for the existing data in a database since it is a critical metric when determining the amount of disk space required to grow the data for a business intelligence workload. The next chapter will talk more about using this ratio.

The data associated with a business intelligence workload, and how it utilizes available disk space, are important considerations for capacity planning. Having adequate disk capacity available to accommodate growth is important, but it is also critical to provide the necessary I/O bandwidth to ensure that there are adequate paths to move the large volumes of data into system memory for processing.

4.2.4 User profile

Like in OLTP systems, BI users drive the capacity requirements. This metric defines the number of people or users that have access to the system. It is important to establish clear terminology when discussing the number of users.

There are various ways to view the user population, particularly in a BI application. First, there is the raw total number of users that have access to the system. Many of them may be only occasional users, some signing on infrequently. Therefore, the number of users alone is not an indication of system activity. A better measure may be the number of users that log on concurrently to a system. This value gives an indication of how many users may be submitting queries in the same time frame. However, while there may be a significant number of users logged on to the system, many may be reviewing query results on the screen rather than executing work in the system. The most meaningful metric is the number of users that concurrently execute queries in the system. This is an indication of the level of concurrency the system must support, and directly impacts data accessibility as well as the capacity requirements for the workload.

Another key metric for capacity planning purposes is the concurrency level divided by the total user population. This gives a proportion of the activity level on the system for the population size. It allows you to estimate the impact on system activity of increasing the user population. For example, knowing a population of 2000 users has approximately 200 active users will allow you to forecast the number of active users when the population doubles.

4.2.5 Profiling query by user population

Another important assessment a BI workload is defining the dynamics of the query workload generated by the user population. Generally, there are two key characteristics of a user that relate to capacity planning.

4.2.5.1 Query type segmentation

The primary characteristic is the types of queries the user submits to the system. This characteristic may evolve over time, as individual users become

more knowledgeable about the data in the system and the kinds of business information they can derive from it. This may cause some users to alter their access patterns over time, perhaps submitting more complex or simplified queries as they develop their awareness of the data available. However, in any population there is a group of users that stabilize their usage, and maintain the same usage profiles over time, similar to the OLTP applications.

It is important to develop a “query profile” of the user community, by segmenting the types of queries all users typically submit into the data store.

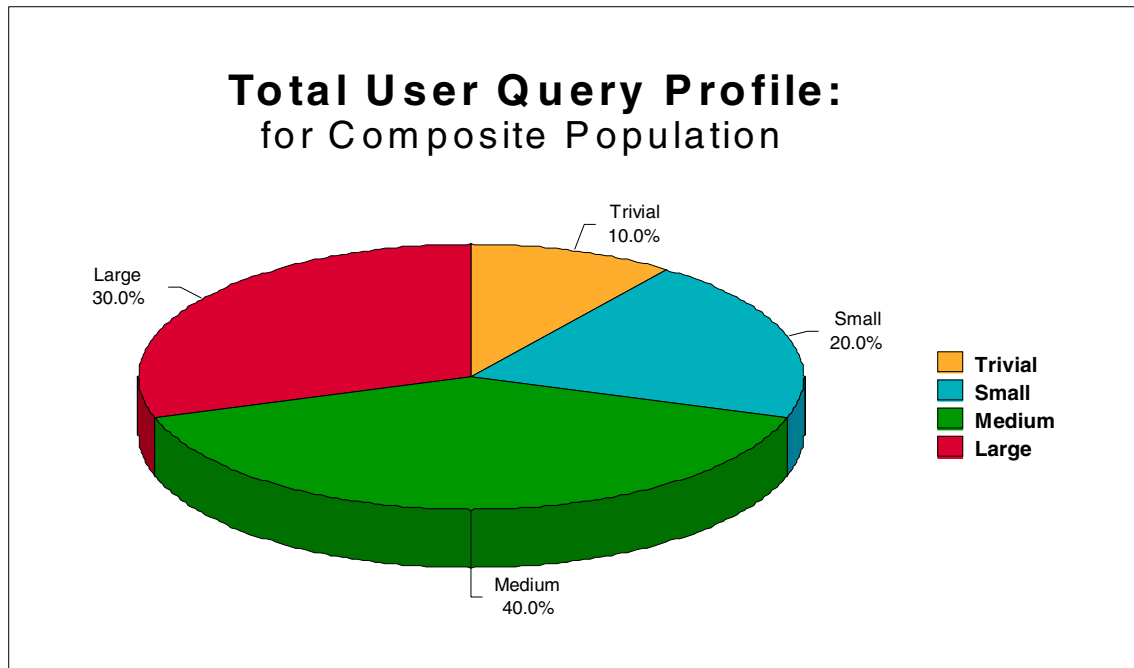


Figure 10. Composite query profile of the total query workload

Figure 10 illustrates how a query profile develops a more complete understanding of the query workload for a business intelligence system. Note that this represents all the queries being submitted by all users accessing the system. A query profile segments the total workload based on the types of queries you would like to track. These query type definitions are arbitrary ones each shop creates for their unique workload. You will want sufficient granularity in your definitions to group the majority of the query workload into 4 to 6 segments, with a reasonable distribution of queries in each category. The segmentation can be based on the total elapsed time for executing a

query, or on the total CPU consumed by the query group. It is up to the individual account to develop these definitions and adhere to them.

For example, if a shop has over 90% of their queries completing in under 10 minutes, the Trivial, Small, Medium, and perhaps even large definitions should segment the group of queries completing in under 10 minutes, and each group should have a minimum of 10-20% of the queries in them. Obviously, two or more segments will have significantly more than that.

This composite profile can be developed for specific days of the week, weeks of the year, or any other period where an analysis of the workload is needed.

4.2.5.2 Business unit segmentation

Another important aspect of the user population is that large business intelligence systems are often accessed by a number of departments or business units within an organization. The business unit to which they belong is a key characteristic of each user. This information allows you to group users from a given business function to derive the capacity requirements of that group. Once the users are grouped, you can develop a model of different business unit's workloads that can be used in calculating capacity requirements as the user population grows.

However, to incorporate a business view to this analysis is not trivial. It requires significant work by the capacity planner or DBA staff to identify the business units for each individual user ID that has access to this system. This requires that the individuals for a business unit have unique user IDs, such as TRnnnn for the customer service department, or MKT0nn for the marketing group. The variables nnnn represent unique identifiers for individual user IDs. To effectively divide the users by business unit requires some unique identifier in all the user IDs for that department.

When DB2 Connect is used to access remote databases, application programming interfaces (APIs) are available to pass tokens to S/390 that assist in differentiating generic ID workload. The Set Client Information API, `sqlseti`, can pass a client end-user ID, client application name, and client workstation name.

A default accounting string for a DB2 Connect Enterprise Edition Server can be overridden by users or applications through two mechanisms. Calling the Set Accounting String API, `sqlsact`, before the application connects to a database is the recommended method. Using the DB2ACCOUNT registry value at the client workstation is the second option. Check whether your vendor's software or local application code is exploiting these APIs. See the

DB2 Connect User's Guide, SC09-2954 and DB2 UDB Administration API Reference, SC09-2947 for more information.

Segmenting the workload by business unit requires the ability to associate query work with an individual user ID. If the application software uses a generic ID to submit work into the data store, it may not be possible to segment the workload by business units.

If you can differentiate between users, and monitor the work they are submitting into the data store, then a query profile for the users of each business unit can be created. This is the *query profile* baseline for this subset of business users.

The value of understanding the query profiles for subsets of users is illustrated in Figure 11 on page 63. The total query workload is represented by the large oval in the drawing, and the query profiles for individual business units are represented by the pie charts within the total workload for the data store.

With this understanding of the source of queries, you can quickly determine the impact of varying any component within the population at the business unit level, to develop the impact varying these user populations will have on the query mix and processor capacity at the server.

For example, if the finance department doubles their users, clearly the number of large queries in the system will increase. If the sales department doubles their users, there will be more small queries entering the system.

Workload Characterization: Query Profiles

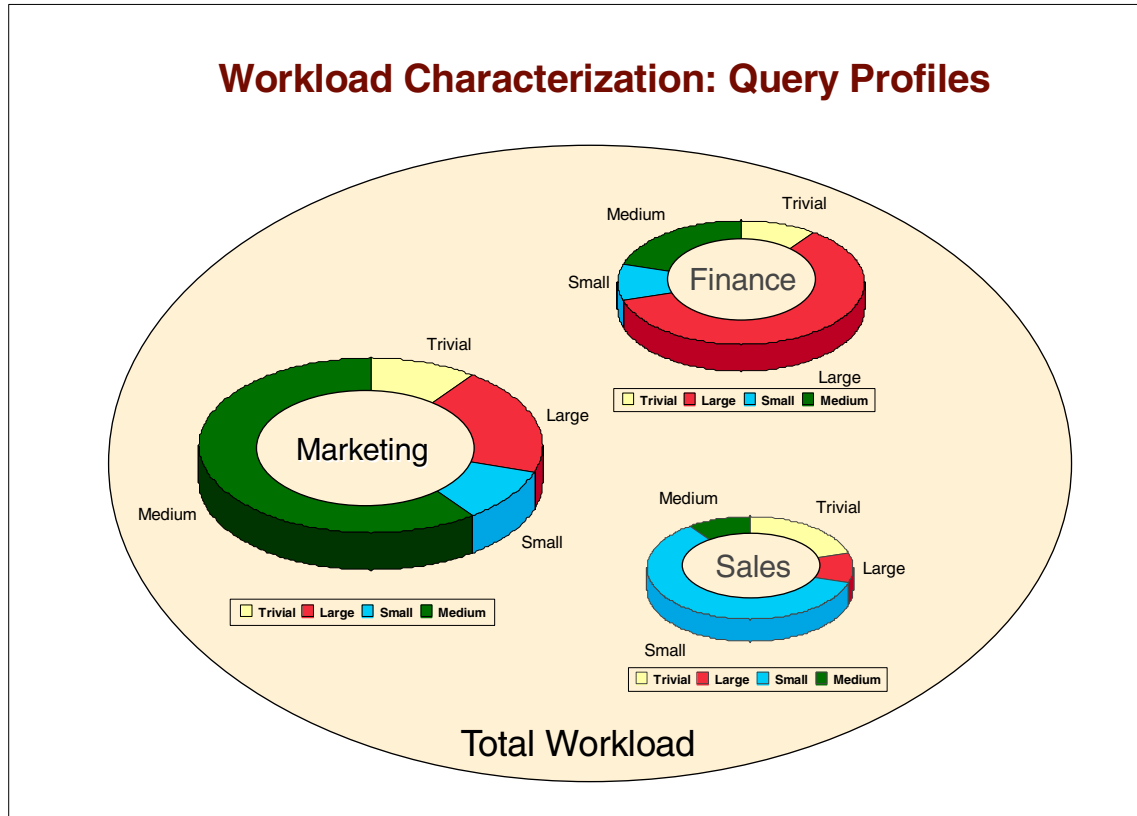


Figure 11. Query segmentation by business units

4.2.6 Maintenance processing

While online processing is often the critical workload for OLTP environments, with BI workloads the batch requirements are also critical. This includes the various maintenance tasks discussed in the prior chapter.

While the application is down for maintenance processing, the new data needed for the next business day must be gathered, cleansed, loaded, and in some systems, summarized (into summary tables) before the system can be made available to the users. Often, this serial job stream to get the data processed and loaded within the nightly batch window can demand significantly more resources than the online query component. Therefore, to effectively plan capacity, it is necessary to analyze this update processing and determine how it will be affected by the expected growth. It is important to note that growing the number of users has no impact on the maintenance/batch job stream, only data growth is a factor.

Analyzing the maintenance workload is done in the same manner as OLTP nightly batch analysis. First, you need to determine how often the system will be updated, and if the amount of records being processed varies on a nightly, weekly, or monthly basis. Once you have an understanding of when the heaviest volumes are being processed, identify the job names for those jobs that are associated with the batch job stream that represents the update process. Since each step is dependent on the successful completion of the prior step, they most likely execute serially in the system. A basic assumption is that the batch job stream has been properly designed and tuned to optimize the processing time required. If not, tuning should be considered as a separate activity.

Once the critical job stream is identified, it's important to determine if that job stream will execute in the maintenance window for the system. Note that there are instances where this batch update stream may need to wait for all data to arrive from other systems before it can begin processing. If that is the case, this must be factored in when determining if the job stream will fit in the batch window.

For example, if the system is available for update from midnight to 7:00 AM, and the batch job stream has an elapsed time of 5.5 hours, it appears that there will be no problem. However, if the last transmission of data needed to start the batch work does not arrive until 3:00 AM, you have missed the window. This is an important consideration when working with a Business Intelligence workload.

In some cases, it is the maintenance workload that drives the capacity requirements of a BI system, not the online query workload. Therefore, it is important to ensure that both the query and the maintenance workload is considered.

4.3 Summary of using the worksheet

This chapter discussed the characteristics of a business intelligence workload to help you better forecast resource consumption. By understanding the relationships between the amount of data within a system, and the capacity of the installed disk, you will be able to predict the impact of increasing the data in the system. By understanding the queries, their CPU consumption, and how they relate to business units, you will be able to determine the impact of increasing the number of users overall, and also increasing the number in specific departments.

Chapter 5. Establishing current system usage

In this chapter we look at how to gather the information needed to understand the characteristics of a business intelligence workload as described in Chapter 4. We introduce a series of worksheets that will help you capture this information. We identify what monitor reports to locate the information in and, when necessary, tell you how to calculate the values to complete these worksheets.

Blank copies of the worksheets are included in Appendix A, “Sample worksheets” on page 147.

5.1 System information sources

The OS/390 is unique among platforms in that it offers the most detailed view of how a system is being used. Other platforms lack this robust variety of hardware and software monitors and reporting tools to allow you to manage the environment. Without this level of detail, you must rely on the vendor to advise you when an upgrade is required. Often, alternative systems require total system replacements for an upgrade, which is a very disruptive approach.

This chapter focuses on the three primary monitors (SMF, RMF and DB2PM) and how to use information from these sources for capacity planning.

5.1.1 System Management Facility (SMF)

The OS/390 operating system includes the Systems Management Facility (SMF), which helps you measure various aspects of work running in the system. This component collects and records system- and job-related information that your installation can use to bill users, analyze the configuration, and perform capacity planning on system usage.

SMF stores the information that is written by other applications, subsystems, and products into system-related SMF records (or job-related records). Most SMF records contain general identification fields such as the job name, step name, programmer name, start time, and start date. By sorting and summarizing SMF records according to these types of fields, an installation can create profiles that show the use of system resources by each unit of work, such as:

- CPU time
- Storage

- I/O devices
- Service units

As the records are created, they are stored in the SMF disk datasets defined to the operating system. There are generally three datasets that are used serially. When one is full, SMF begins storing in the next dataset, and the first can be dumped to tape for processing by reporting routines.

5.1.2 Resource Management Facility (RMF)

The Resource Monitoring Facility (RMF) is also a component of the OS/390 operating system. RMF contains three separate monitors that can be operated to collect and write data into SMF records (types 70-79). These records are created and stored in the SMF datasets, and eventually are offloaded. The RMF postprocessor routine can then process these RMF records to create comprehensive reports of either a single-system or sysplex scope.

The RMF reports generated by the postprocessor are used for either capacity planning or performance tuning of an OS/390 system. For the capacity planning activity, only three reports are necessary:

- The Summary Report summarizes the key resources for an OS/390 image.
- The CPU Activity Report provides an overview of activity for all processors belonging to this OS/390 image.
- The Workload Activity Report shows workload-related data on resource consumption with different levels of detail. It provides a summary by performance group (service class in goal mode), by reporting groups, and for the entire system.

To create these postprocessor reports, code the batch job using SMF datasets as input. Refer to *RMF Users Guide* for more information. Later in this chapter we discuss each worksheet, as well as the exact reports and fields for input for these reports. For additional information on RMF reports, see either *RMF Performance Management Guide* or *RMF Report Analysis*.

5.1.3 DB2 Performance Monitor

IBM DATABASE 2 Performance Monitor (DB2 PM) is a tool for analyzing and tuning the performance of DB2 and DB2 applications. It runs on an OS/390 server and has a real-time online monitor which runs on an OS/2 or a Windows NT workstation. It includes a history facility to view recent events,

and a wide variety of reports for in-depth analysis. DB2 PM V6 has been included as a feature of the DB2 UDB for OS/390 V6 product.

DB2 generates data about its own performance, called *instrumentation data*, but it does not provide any reporting facilities for analyzing this data. The Online Monitor of DB2 PM allows you to view an active DB2 subsystem and identify performance problems online.

For a long-term view of DB2 performance, use the DB2 PM Batch reporting capabilities, which also provide an historical view of DB2 performance. When you are performing capacity planning, DB2 PM can be used to measure an application's performance, its resource usage, and the effect an application can have on the system.

5.1.4 How performance data is generated

The DB2 trace facility, also called the DB2 instrumentation facility, gathers information about data and events in the database. After DB2 has collected and externalized this data, DB2 PM reads it and generates reports, graphs, and data sets from it.

When starting the DB2 trace facility, specify the performance information you want to collect. The data is recorded in different record types, called instrumentation facility component identifiers (IFCIDs). These records are grouped into *classes*, and the classes are grouped into *types*, which are the broadest categorization of performance data. The types used by DB2 PM Batch are:

- Statistics
- Accounting
- Audit
- Performance
- Global

SMF is the default destination of trace output for Statistics and Accounting information. All DB2 trace data is found in SMF record types 100 through 102.

Traces have a performance overhead, so carefully consider which traces to run. IBM recommends regularly collecting accounting class 1 and 3 and statistics class 1, 3, and 4 data. You should also consider collecting accounting class 2 data, which provides important information about DB2 times.

5.1.5 How DB2 PM reports the data

DB2 PM generates reports, data sets, graphs, and logs. The output is controlled by means of commands. The output is grouped into *report sets*, each associated with a particular type of data. The report sets most commonly used for capacity planning are *Statistics* and *Accounting*. Statistics summarizes system-wide performance data, whereas Accounting summarizes information for particular applications. Both report sets are used to determine the efficiency of the subsystem or application.

Some forms of DB2 PM Batch output we are interested in are:

- Traces

Traces show individual DB2 events, for example, for a particular thread. All events are listed individually, usually in the order of occurrence.

- Reports

Reports show these events summarized by identifiers, such as primary authorization ID or plan name.

- Data sets

Formatted data can be stored in data sets suitable for loading into DB2 tables. The tables can then be easily queried to produce customized reports using a reporting tool such as IBM Query Management Facility (QMF).

These tables are optional, but we strongly recommend having historical performance data loaded into DB2 tables. DB2 PM provides the statements for creating the tables, loading the tables, and sample queries. The data can be used for capacity planning as well as performance and tuning. Storing the information in DB2 data sets simplifies saving the data for use for future analysis or problem determination. We will simply refer to these tables as the performance database.

For additional information refer to *DB2 Performance Monitor for OS/390 Batch User's Guide* and *DB2 Performance Monitor for OS/390 Report Reference, Volume I* and *Volume II*.

5.2 Developing a workload profile

In a typical business intelligence application, the majority of the processing is performed by the data server and its relational database. To conduct a capacity planning effort for this environment, there are a number of aspects to consider. These have been captured in a series of sample worksheets found

in Appendix A, “Sample worksheets” on page 147 of this book. The worksheets are:

- **Worksheet 1: Application profile**
This worksheet illustrates the business-related information that should be gathered for all the applications accessing the data store.
- **Worksheet 2: System characterization**
This worksheet organizes the system-level information that creates a high-level profile of the capacity requirements of a BI environment. This identifies the analysis period to be used as a basis for capacity planning of a workload.
- **Worksheet 3: Workload characterization**
This worksheet gathers information that is used to plan for the data stored in the DB2 subsystem, as well as profiling the query workload that accesses it.
- **Worksheet 4: Maintenance processing profile**
This worksheet develops a profile of the maintenance and update work associated with the data stored in the DB2 subsystem.
- **Worksheet 5: Growth projection**
This worksheet gathers information related to the future growth requirements of the application.
- **Worksheet 6: New application**
This worksheet gathers information for new applications that are being planned for. It is discussed in Chapter 7, “Sizing methodologies for new applications” on page 103.

To make it easier to identify those sheets that pertain to the same study, the worksheets all start with the same basic information at the top of the page. The identification items are:

- **Date of Analysis:**
The date of the time interval being studied for a capacity analysis.
- **Time Period of Analysis:**
The time period for the interval being studied (which hours of the day).
- **Workload ID:**
Identifies the workload that is being studied, with the associated subsystem ID (STC).

- **System ID:**
Identifies the unique system ID that is used to identify this OS/390 image in the system monitoring records.

Together, these worksheets highlight the information that should be gathered to develop an effective capacity plan for a business intelligence workload. The next several sections of this chapter describe each of the worksheets and identify the sources of information needed to complete the worksheets.

5.3 Worksheet 1: Application profile

This worksheet helps capture the relationships between the business units that share access to the data, as well as within the business units. This worksheet is completed through discussions with the business units' personnel. Monitor reports are not required for this particular spreadsheet. Complete a worksheet for each business unit accessing the data store.

The sections on the worksheet are:

End-User Information

This section of the worksheet contains information related to the end user population. It identifies the tools being used, who installed the software on the workstation (in case there are additional questions related to the software), and how the end user is connected to the application server. This information can be important when trying to determine if it is possible to segment the user population to develop a query profile for the business unit.

Application Server Information

This section is focused on the application server code. It collects information that can be used to identify the SMF data that pertains to this workload if it is running on the S/390 platform. Having this workload in a dedicated performance group/service class simplifies gathering the required RMF data for the workload.

Critical Processing Windows for Business Unit

The bottom portion of this sheet should be completed for each of the individual business units that submit queries in to the system. Understanding which time periods are important to each unit will ensure each group receives the proper priority at their critical processing times. A crucial question that this information answers is, does more than one business unit have a critical need to get information quickly during the same time interval? This information will be required for the following worksheets.

5.4 Worksheet 2: System characterization

This worksheet provides an overview of the system. Before completing this worksheet, it is necessary to develop the CPU consumption profile for the application over time to determine the peak CPU requirements for the data store. First run the RMF CPU Activity Report using a large interval of a day to get a sense of activity over a month. Identify the peak processing times and rerun at the daily and hourly level, drilling down to understand CPU usage for the application over time.

To develop a good understanding of an application, you should review its activity monthly for the first year. Keep in mind that the usage of a BI system will probably change dramatically over time. As data continues to accumulate, it may require more resources in the second and third year to process the same queries. Eventually, jobs will be added to delete or archive old data from the data store. In addition, as their understanding of the data matures, users will begin asking more complex queries, again increasing CPU requirements.

It is this adhoc nature of the workload that makes understanding this environment so different from OLTP environments. While OLTP workloads grow, the users have specific, well-tuned transactions they can run. They do not have the ability to create their own transactions at any time, as users of a business intelligence system can. Therefore, monitoring this system over long periods of time is necessary to understand how the usage patterns are evolving.

To develop this profile, run the RMF postprocessor Summary Report, which consists of one line of data that summarizes system activity for each interval within the reporting period. This will help pinpoint the critical processing times for the workload.

5.4.1 General information

Once the peak times for processing are identified, complete Worksheet 2. The top section of the worksheet captures general system information related to the LPAR or system where the workload being analyzed is running. The information you will gather includes:

Processor Model

This indicates the physical processor model number the OS/390 image is running on. This is found on the RMF CPU Activity Report or it can be obtained from a systems programmer. This includes the number of engines, memory, and channels for the OS/390 image.

#CPs

The number of engines enabled to the OS/390 image this workload is running on. If it is running in an LPAR, it is the number of engines enabled to the LPAR at the time the measurement is taken. This can also be derived from the CPU Activity Report. Count the number of engines listed in the column titled CPU Number.

Memory

The amount of memory available to the OS/390 image when the measurement is taken. Request this information from the systems programming staff. It can also be determined from the RMF Paging Report. Multiply the total frame counts for real and expanded memory, listed in the lower left quadrant of the page, by 1024 bytes per frame.

Channels

The number of channels on the processor that are available to the OS/390 image. This information should be obtained from the systems programmer.

5.4.2 System utilization

The information in this section identifies the time interval being analyzed and why it was selected, thus conveying the importance of this time interval.

Worksheet 2 has two Peak CPU Utilization tables. Table 2-1 lists the CPU utilization for each hour of each day during the peak week, and a similar table (2-2) is included for each 15 minute interval of each hour, for a single day's CPU utilization. These tables contain the data points for line graphs, similar to those in Figure 9 on page 57. Use the CPU Busy field from the RMF Summary Reports run at the appropriate intervals to complete these tables. For Table 2-1, process RMF records for the week, with an interval time set to 1 hour. For the peak day, re-run the RMF Summary Report with an interval of 15 minutes and complete the detailed information on Table 2-2.

5.4.2.1 CPU by workload

In using the RMF Summary Report to determine the peak processing for this workload, we are assuming that there is only one workload on this system image. That is often the case with business intelligence workloads. However, if the BI application is sharing a system with a diverse set of other workloads, it will be necessary to do this analysis at the *workload* level as well as the *processor* level. The RMF Workload Activity Report can be used for this. The BI workload must be in its own service class (referred to as a Performance Group if running in compatibility mode) and reporting group. The field Appl% is the percentage of a single engine used by the workload. This field can be used to develop the workload profile for the application.

It is important to profile both the query and the maintenance workloads on the system. With business intelligence systems, the maintenance workload is sometimes larger and requires more resources than the query workload in the system. By completing tables 2-1 and 2-2 for the peak week and day for *each* workload, you can identify when the maximum system resources are being consumed for the business intelligence system.

5.5 Worksheet 3: Workload characterization

Worksheet 3 develops a characterization of the BI workload: the amount of data, the number of users, and especially the details regarding the query profile of the workload.

5.5.1 Data profile

To quantify the relationship between the data and disk installed, determine the amount of raw data in the database subsystem and the size of physical disk installed, both in gigabytes. What is included in these values is discussed in the preceding chapter; refer to 4.2.3, “Profiling data usage” on page 58 for more information.

The database administrator can provide this information. A query of the DB2 Catalog can be executed to total the DASD usage of tablespaces, indexes, and the DB2 work database, DSNDB07. The DB2 Catalog statistics should be kept current through execution of the RUNSTATS utility.

5.5.2 User profile

The total user population is determined by discussions with the business unit. If more than one business unit accesses the data, include them all. The value to enter on this worksheet is the total number of persons with user IDs for the system, not the number logged on. This is the sum of the users for all the business units you interviewed when completing worksheet 1.

5.5.3 Query profile

This is a key section of the worksheet. Here you will develop the query profile for the user population, first at a total population level, then at the business unit level (if possible). Refer to 4.2.5, “Profiling query by user population” on page 59 for more information on this topic.

Table 3-1: Query segmentation

This table creates the query profile of the user community. The total queries submitted to the database are segmented into groups based on criteria that

you establish. In this table, data is entered for each segment. The columns in the table are:

Query Type

The naming convention of the query segments for the individual account. The names and definitions in this sample worksheet are arbitrary; in reality they should be created specifically for the client company, based on criteria meaningful to them.

Time limit

This is the criteria that differentiates the query types for this workload. The time parameters should be set to divide the work with a reasonable number of queries in each segment.

Max CPU (seconds)

The maximum number of CPU seconds the longest running query in this segment required to complete processing. This is obtained from a query run against the DB2PM historical tables.

Min CPU (seconds)

The CPU seconds consumed by the fastest running single query in this segment. This is derived from another query against the historical data.

Avg CPU (seconds)

The average number of CPU seconds for all the queries that completed in this segment. This is calculated by executing a query that sums the CPU seconds for all queries that qualified for the time limit for this segment as defined above. The sum is then divided by the total number of queries that executed in this segment. This information is from the DB2PM historical tables.

% of query workload CPU usage

Percentage of the CPU resource consumed only by the query workload, that is consumed by this subset of query types. To calculate, gather the RMF Workload Activity report listing, and add up the APPL% for each query type in the workload report. Divide the APPL% for this query type, by the sum of the query types in the report.

% Total CPU

The percentage of the total CPU resource this segment of the query workload consumes. It is determined from either the RMF reports or the DB2PM data.

When using the RMF reports, calculate this by finding the APPL% for the workload segment as reported in the Workload Activity Report, and divide this

by the total number of CPU seconds for the measurement interval (interval time converted to seconds, multiplied by the number of engines).

With DB2PM, you can execute a query to sum the total CPU seconds for all the queries that run in this segment, and divide by the total CPU seconds for the measured interval (interval time converted to seconds, multiplied by the number of engines).

If you do not have WLM in Goal Mode, you will need to work with the DBAs to determine this information. Once this table is completed, it can then be used to properly establish a WLM policy for your system, so you get the appropriate RMF Workload Activity Reports.

5.5.4 Additional information

This section of the worksheet, includes optional information that will help you understand the nature of the workload, however the information is not required for a capacity planning effort for a Business Intelligence workload.

5.5.4.1 Disk subsystem

The first optional section deals with the disk subsystems that are physically installed for the database workload. Ask the system programmer staff for all the information in this section. The disk subsystem types, the number of like systems installed, the number of channels, and gigabytes of disk in the subsystem. While this information is not necessary for the capacity study outlined in this manual, it will provide important information for tuning activities.

5.5.4.2 Table 3-2: Disk usage table

This table breaks down the amount (gigabytes) of used disk into its components. This breakdown will give you insights into how your disk resources are being utilized, thereby highlighting some constraining factors for growth. In addition to being useful for the database support team, this table includes two key values that are required to develop the growth requirements of an application. They are the System and Reserve/Other fields, which are discussed later in this section.

All the fields listed in this table can be determined by your database administrator. A query of the DB2 Catalog can be executed to sum the DASD usage of tablespaces, indexes, and the DB2 work database, DSNDB07. The DB2 Catalog statistics should be kept current through execution of the RUNSTATS utility.

All the fields in this table should be expressed in units of gigabytes. The fields have the following significance:

Raw data

The amount of disk space necessary to hold the raw data or actual application data stored in the DB2 subsystem tables.

Index

The amount of disk space required to hold all the indexes related to the DB2 tables.

Summary tables

The amount of disk space required to hold summary tables for a given workload.

DB2 work area

The size of the physical disk required for the DB2 work area.

Batch work area

The amount of disk used as a work area by batch programs.

System

The amount of disk space required for the operating system, executable DB2 code, and maintenance-related (DLIBs, SMPE zones) files.

Reserve/Other

The amount of disk space required for things like testing, unexpected growth, and disk space left unused.

The System and Reserve/Other fields represent the disk requirements for this application, in addition to the true data requirements. These fields should be added to the amount of physical disk needed for the data.

Note that as the data grows, the indexes, summary tables, and other DB2 constructs grow with it. However, the system and reserved disk space requirements remain constant, and generally increase only when new versions of software are installed. Take this fact into account when calculating the growth requirements of the workload.

5.5.4.3 Table 3-3: Disk subsystems physically installed

A related table deals with the disk subsystems that are physically installed for the database workload. Ask the systems programming staff for the information to complete this table: the disk subsystem types, the number of like systems installed, the number of channels, and gigabytes of disk space in the subsystem.

5.5.4.4 Table 3-4: Query average and peak data

This optional table contains interesting information that helps define the characteristics of this workload. The columns in this table have the following meanings:

Average number/day

The total average number of queries executed in the sample period, per day. It is calculated by summing the total number of all the queries executed each day of the sampling period, and dividing by the total number of days in the sample period.

Average concurrent

The average number of queries running concurrently over the entire sampling period. This is a tricky number to calculate, but you should try to do so. There is no field in DB2PM that reflects the average number of queries executing in the system concurrently. But there *is* a field that states the peak number of queries executing concurrently during a reporting interval. So we can “fudge” this number by setting the reduction interval (which is a parameter on the DB2PM job that reads the SMF records and generates a report) to run at short intervals, say every 10 or 15 minutes, and execute it to run for the duration of our sampling period. That generates a lot of data points, but it's easy math.

Then record the field marked No. of Occurrences in DB2PM Accounting reports for each interval. Next, average that value for all the reported intervals in the sample, by dividing the sum of all the interval values by the total number of intervals counted. This effectively calculates the upper boundary for the average number of queries executing concurrently in the system. This is the best we can do to derive this answer at this time.

So, to calculate the value, run the DB2PM with 10/15 minute reporting intervals, and sum up all the values in the field No. of Occurrences in Accounting report. Then divide by the number of intervals you added. This will be the average (peak) concurrent queries executing in the system.

Peak number/day

The highest total number of queries executed for a day within the sample period.

Peak concurrent

For the same day used above, this is the highest concurrency level for the queries for any interval from the specific day. It is determined from the highest value in the No. of Occurrences for the day.

Information required for the Query Avg/Peak data in Table 3-4 can be obtained from DB2 PM Batch reports. This requires familiarity with the DB2 PM commands.

The report commands have many options providing a lot of flexibility. INCLUDE/EXCLUDE options allow you to select data based on certain identifiers so you can segregate the query workload if desired. FROM/TO limits can limit the range of records to be included in the report processing by date and time. For some worksheet fields, you need to manually review the reports and calculations to derive the results. We will also describe how to query the performance database as an easier alternative and provide sample SQL.

To derive the average number of queries per day (Avg # / day), and the peak number of queries per day (Peak # / day), the Short Accounting Report can be used. It displays the number of occurrences of an activity based on what field(s) the report is ordered by. ORDER specifies which DB2 PM identifiers are used to aggregate accounting records. An interval of 1440 minutes can be specified with the REDUCE subcommand to organize the output report into daily sections. The sum of the number of occurrences divided by the total number of days in the input time period will provide the Avg # / day. The maximum value of the number of occurrences is the Peak # / day.

To derive the concurrency values in the query profile, another version of the Accounting Report Short can be used. The report can be arranged in smaller intervals defined by the customer, for example, 15 or 30 minutes. In addition to ordering the report by INTERVAL, also specify connection type (CONNTYPE), or requesting location (REQLOC), so that distributed queries and local queries are reported separately.

Using the sum of the number of occurrences (#OCCURS) for all intervals, and dividing by the number of intervals reported, will yield an approximate Avg Peak Concurrent value. The highest #OCCURS from the report would be the Peak Concurrent value for the worksheet. The smaller the interval, the more precise value the concurrency values will be, but keep in mind that the report itself can become quite lengthy and adds overhead to the report processing.

Figure 12 on page 79 shows examples of these DB2 PM Batch commands.


```
STATISTICS
  REDUCE INTERVAL (1440)
  REPORT
    LAYOUT (LONG)
    ORDER (INTERVAL)
ACCOUNTING
  REDUCE INTERVAL (15)
  REPORT
    LAYOUT (SHORT)
    ORDER (INTERVAL-PLANNAME)
```

Figure 12. Sample statistics and accounting commands

5.5.4.5 Table 3-5: Detailed query segmentation

This table provides significantly more detail pertaining to the segments that were defined in Table 3-1. Here, there is information that relates to the volume of data each query is scanning (getpages), as well as the elapsed time for the query to execute. You must use DB2PM to complete this table; the necessary information is available only from the DB2PM monitor. The rows in this table reflect the query segments described previously for Table 3-1.

The columns of the table have the following meanings:

- **Avg total/day**
The average total number of queries that executed for this segment per day. This is calculated from the DB2PM field, total number of queries, for each day in the period, averaged together.
- **% of daily total**
Of the total queries executed in a day, the percentage that qualified for this segment. It is calculated by dividing the total number of queries for this segment for the day by the sum of all the queries for all segments in this day.
- **Avg ET**
The average elapsed time for queries within this segment. It must be calculated from information in the DB2PM historical tables. Submit SQL against the tables.
- **Min ET**
The shortest elapsed time for a query in this segment.
- **Max ET**
The maximum elapsed time for a query in this segment.

- **Avg CPU**
The average CPU seconds used for queries in this segment.
- **Min CPU**
The lowest CPU seconds reported for a query in this segment.
- **Max CPU**
The maximum CPU seconds for a query in this segment.
- **Getpages per day**
The average of the total getpages for this segment per day.
- **Max GetPages per day**
The maximum number of getpages for this segment per day.
- **Avg Total GetPages per day**
The average of the total number of getpages for this segment, per day, for the entire sampling period.

The detailed information for this table can be obtained from the Accounting Trace - Short. However, to derive the table from the output would require manually reviewing each thread and categorizing it. This would be very time-consuming; instead, we recommend loading performance data into DB2 tables to create a performance database.

With the performance database you can:

- Perform additional performance evaluations
- Collect historic data
- Use SQL for fast and easy retrieval of data

Both the Accounting and Statistics groups consist of multiple tables. By joining the Accounting General Information table with the Accounting Buffer Pool table, you can gather the elapsed times, CPU times, and getpage data. With this information, you can build data for Table 3-5. Sample SQL is shown in Figure 13 on page 81. The query can be tailored based on how the query workload is identified and the range of time to be profiled.

A report generated from the data returned by the query can also sum the number of queries (nbr_queries), and count the number of days to calculate the average number of queries per day in Table 3-2.

```

-- GENERATE QUERY PROFILE FOR WORKSHEET #3

SELECT
  CORRNAME
, CATEGORY
, COUNT (*) AS NBR_QUERIES
, END_DATE
, AVG(ET) AS AVG_ET
, MIN(ET) AS MIN_ET
, MAX(ET) AS MAX_ET
, AVG(CPU) AS AVG_CPU
, MIN(CPU) AS MIN_CPU
, MAX(CPU) AS MAX_CPU
, MIN(GP) AS MIN_GP
, MAX(GP) AS MAX_GP
, SUM(GP) AS TOT_GP
FROM
(
SELECT
  G.CORRNAME
, DATE(G.TIMESTAMP) AS END_DATE
, CLASS1_ELAPSED AS ET
, (CLASS1_CPU_NNESTED + CLASS1_CPU_PARAL) AS CPU
, CASE
  WHEN CLASS1_ELAPSED < 5      THEN '1. TRIVIAL '
  WHEN CLASS1_ELAPSED < 300   THEN '2. SMALL  '
  WHEN CLASS1_ELAPSED < 1800  THEN '3. MEDIUM '
  WHEN CLASS1_ELAPSED < 3600  THEN '4. LARGE  '
  ELSE                          '5. VERY LARGE'
END AS CATEGORY
, GP
FROM creator.DB2V6FACCT_GENERAL G
, (SELECT TIMESTAMP, SUM(BP_GETPAGES) AS GP
   FROM creator.DB2V6FACCT_BUFFER
   WHERE CORRNAME = &qryid
   GROUP BY TIMESTAMP
  ) AS BP
WHERE CORRNAME = &qryid
  AND G.TIMESTAMP = BP.TIMESTAMP
) AS TMP

GROUP BY CORRNAME, END_DATE, CATEGORY

```

Figure 13. Sample SQL for query profile worksheet

As an alternative to using the DB2 PM performance database, if Workload Manager has been set up to differentiate between groups of queries, the Ended field in the RMF Workload Activity Report can be used to determine the number of queries completing in each segment.

5.6 Worksheet 4: Maintenance processing requirements

While online processing is often the critical workload for OLTP environments, with Business Intelligence workloads, the maintenance requirements are also critical. While the application is down for batch processing, the new information needed for the next business day is generally gathered, cleansed, loaded, and in some cases, summarized (into summary tables) before the system can be made available to the users.

Often, this serial jobstream to get the data processed and loaded within the nightly batch window can demand significantly more resource than the online query component. Therefore, to effectively capacity plan, it is necessary to analyze this update processing and determine if it will be impacted by the expected growth. Growing the number of users has no impact on the batch jobstream; only data growth is a factor.

The maintenance processing analysis is conducted in a similar manner as with OLTP night batch analysis. First, you need to determine how often the system will be updated, and if the amount of records being processed varies on a nightly, weekly, or monthly basis. Gather this information for the operational profile in Table 4-1 of Worksheet 4. Also note the number of hours, per day and per week, that the system is available to process queries (operational), available for backing up/archiving data (maintenance), and available for updating the data within the database (update database). This should be noted in Worksheet 4.

Once you have an understanding of when the heaviest volumes are being processed, work with the user community to identify the jobnames for those jobs that are associated with the batch jobstream that represents the update process. Since each step is generally dependent on the successful completion of the prior step, they most likely execute serially in the system.

A basic assumption here is that the batch jobstream has been properly designed and tuned to optimize the processing time required. If not, tuning should be considered as a separate activity. Another point to note is that the workload can vary dramatically from night to night, as discussed in Chapter 3. The volume of updates may fluctuate significantly from one batch stream to the next, and it is best to discuss this with the DBAs and business users of the application to understand the factors causing this. Once you have the jobstream running as planned and you understand deviations in the workload, you can effectively capacity plan for the workload.

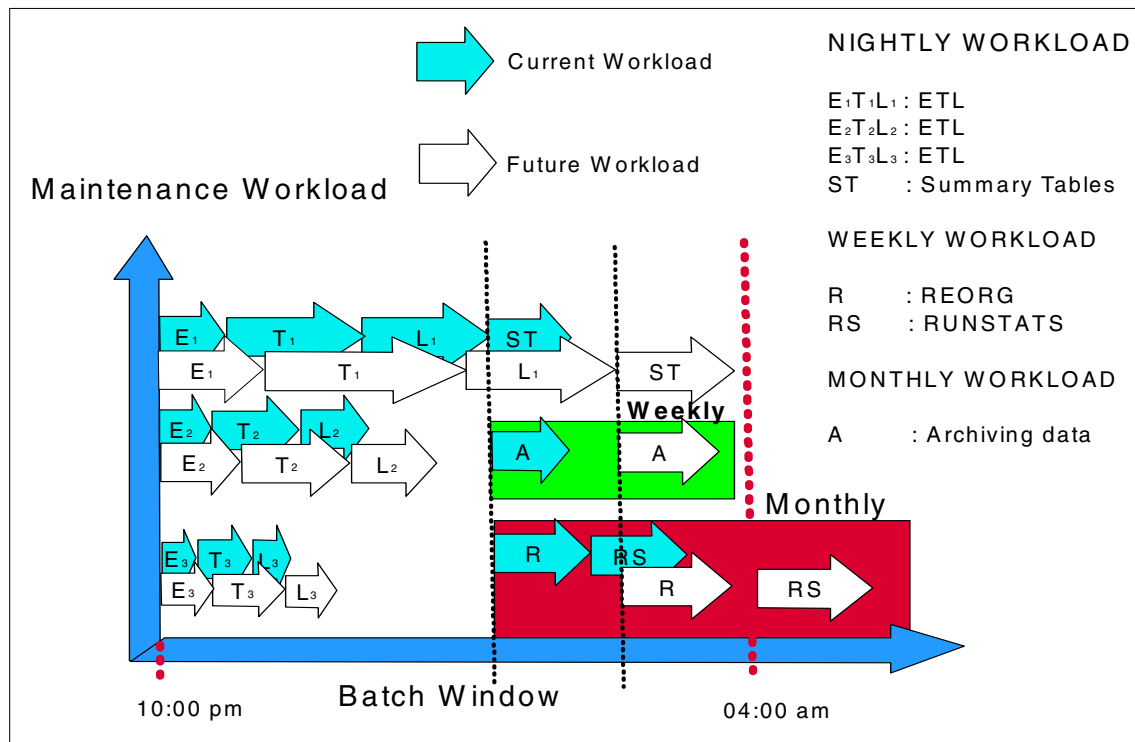


Figure 14. Understanding the Maintenance Workload in the batch Window

Figure 14 illustrates the batch jobs that are commonly found within a typical batch jobstream. Three separate ETL processes are illustrated. In each case, they must complete before the Load or Summary Table (ST) build jobs can be executed. In some cases, select jobs are run only weekly or monthly for the workload. This is particularly true for Archiving, REORG, and Runstats jobs.

The batch window for this application is from 10:00 PM to 4:00 AM. The dark arrows indicate the workload, with the current amount of data in the database. The light arrows reflect the longer execution time required when the data is increased. The first black line (on the left) identifies when the ETL processes are completed, and the final Summary Table build jobs in the batch stream can execute. As the data is increased, the time necessary to execute the batch jobs increases, and the black line is moved to the right. Note that the increase will result in the monthly jobs exceeding the batch window.

To determine if the Batch Maintenance workload will fit into the window, you typically analyze the SMF records for the workload. There are a variety of

tools to process the SMF records created for each job. Gather the information for Table 4-2 of Worksheet 4 from the SMF records. The information you require is the jobname, Start Time, Stop Time, elapsed time (Stop-Start time), and CPU seconds. Once you have this data calculated for the entire critical job stream, you will use this information in the next chapter to determine what impact growing the data will have on this workload.

The information required in Worksheet 4 should be completed for the peak time frame for the maintenance workload that was determined in Worksheet 2. It focuses on profiling the maintenance workload described in Chapter 3. This worksheet will assist you in determining if the maintenance workload will fit into the allotted window.

5.6.1 Table 4-1: Operational profile

This table is used to determine the processing time available for operational (query) executions, maintenance (archives, backups, and so forth), and data updates. Determine the number of hours for each day, and each week, that the system is available for each type of processing. The information for this table can be obtained from either the system operators or the technical support staff for the system.

5.6.2 Table 4-2: Critical path for batch jobs

This table is used to identify the steps that are critical for maintaining the quality of the data in the data store, including gathering and cleansing of updates for the data, loading the data into the database, and creating or refreshing any existing summary tables to reflect the changes in the underlying detail data. All the jobs associated with these tasks constitute the critical path for the nightly maintenance processing.

The best way to determine what jobs make up the critical path is to discuss this issue with users in the business units, as well as the system programming staff that supports this application. Once the jobs are identified, a COBOL job or SAS job can be written to process the SMF records for each job, extracting the necessary information.

The purpose of this worksheet is to gather the critical information related to the maintenance work, and to determine if that work can complete in the allotted time.

5.7 Summary

Using the monitors discussed in this chapter, you will be able to complete the four worksheets that profile the current processing requirements for a Business Intelligence workload. This information is required in the next chapter to determine the additional resource requirements to handle the projected growth of the business.

By carefully documenting the information gathered in these worksheets, you will acquire an understanding of the business flow for the work, and how to relate changes in the resource consumption to the work over time. The dynamic nature of the query workload makes it a challenging environment to manage and plan for. Only by developing a clear understanding of the characteristics of this workload can you develop a methodology to properly plan for the capacity requirements of this unique environment.

Chapter 6. Growing existing workloads

So far, we have looked at the issues related to capacity planning for a business intelligence workload. While reviewing a variety of worksheets designed to organize the information needed, we have seen how we can use a number of monitors to gather the information. Now we will look at how to forecast the required capacity for a BI environment.

When considering the capacity requirements for a business intelligence workload, we have discussed the importance of both the query and maintenance workload. In chapter 3, we looked at the impact the maintenance workload can have on a system and discussed the fact that this component can often be larger, and more resource-intensive, than the query component. When capacity planning for a workload, it is important to evaluate both the query and maintenance components of the workload, and determine which is more resource-intensive. You will capacity plan to that component since, in most solutions, they run during different windows of the 24 hour day, and generally one will drive the resource requirements for the workload. In this chapter we estimate the additional resource requirements for the expected growth, for both query and maintenance workloads.

6.1 Capacity planning concepts

Before we begin looking at the capacity requirements for either the query or maintenance workloads, it is important to discuss some fundamental concepts in capacity planning. This includes having a system that is balanced, understanding the importance of capture ratios, and knowing the relative I/O content of a workload.

6.1.1 Balanced system

A system is in balance when the processor, storage, and attached devices are configured so that nothing is over- or under-utilized. As the workload increases on such a system, you must upgrade the different resources proportionally (processor, storage, channel configuration, and DASD storage) so that the system stays in balance. What components should be upgraded, and when, is dependent on the type of workload the system supports. For example, if much of the growth is in a processor-intensive application, processor and memory upgrades will most likely be needed before DASD subsystem upgrades. If the work drives a significant amount of I/O, then the DASD subsystem and channels should be reviewed when any workload increase is being planned for.

6.1.2 Capture ratio

To effectively plan for the capacity of a workload, you need to determine the total processing requirements for the workload. CPU consumption is reported at a number of levels in various reports available through the Resource Measurement Facility of OS/390 (RMF). Within the RMF, the Workload Activity Report provides the CPU resource consumption for specific workloads in the system. System-related functions, such as the processor time used in “service” address spaces such as JES or VTAM, as well as system address spaces such as the Master address space or the many monitors and other services that exists in the system, are listed separately if the WLM controls are set up correctly.

A capture ratio attempts to equitably adjust the time directly attributed to the business unit work to include uncaptured system time. Uncaptured system time includes time not reported under discrete workloads in the RMF Workload Report, as well as the time that is reported for the system functions mentioned earlier. Using capture ratios you can then determine the actual utilization of a workload at the system level, which equates to the hardware capacity that must be installed to handle the work.

To calculate the capture ratio you need to gather system utilizations from a number of system reports. For information on calculating exact capture ratios for a system, see Appendix D, “Capture ratios” on page 171, or refer to *RMF Performance Management Guide*.

In business intelligence workloads, we are generally able to capture approximately 90% of the CPU resource at the application level. Therefore, we can use a simple rule of thumb to raise the workload CPU consumption by 10%, and have a reasonable estimate of the total capacity requirements necessary to handle the workload. When using the capture ratio to adjust the CPU resource requirements of a workload, take the APPL% from the RMF Workload Report, and multiply by 1.1, thereby increasing it by the 10% to get a more accurate level of CPU resource. This should be an acceptable method for adjusting the CPU consumption by the capture ratios for Table 5-1 on page 147; the pertinent worksheet is discussed later in this chapter. If you feel you need a more accurate estimate, refer to Appendix D, “Capture ratios” on page 171.

6.1.3 Relative I/O content (RIOC)

Another characteristic of a workload is its relative I/O content (RIOC), calculated as the DASD I/O rate divided by the relative processing power used for a workload. It is a measure of how I/O intensive the work is. If the type of work does not change, this should be constant when the workload

grows. The RMF Workload Activity Report shows the I/O activities by workload in the field “SSCHRT.” This is the number of start subchannels per second; it shows the number of DASD non-paging I/Os generated by this workload type. Dividing this by the relative processing power used by a workload yields the RIOC. While the RIOC of a workload tends to be consistent for OLTP-type applications, it can change in a business intelligence workload as the queries evolve over time, and as summary tables are created and dropped from the database design. However, the current RIOC should be consistent through the capacity planning period, and should be considered when the workload is expected to grow.

6.1.4 Understanding the workload

By correcting the CPU utilization for each workload to account for the uncaptured CPU consumption, and by understanding the RIOC for each workload, you can get a graphical picture of the nature of the types of queries in the system, as shown in Figure 15.

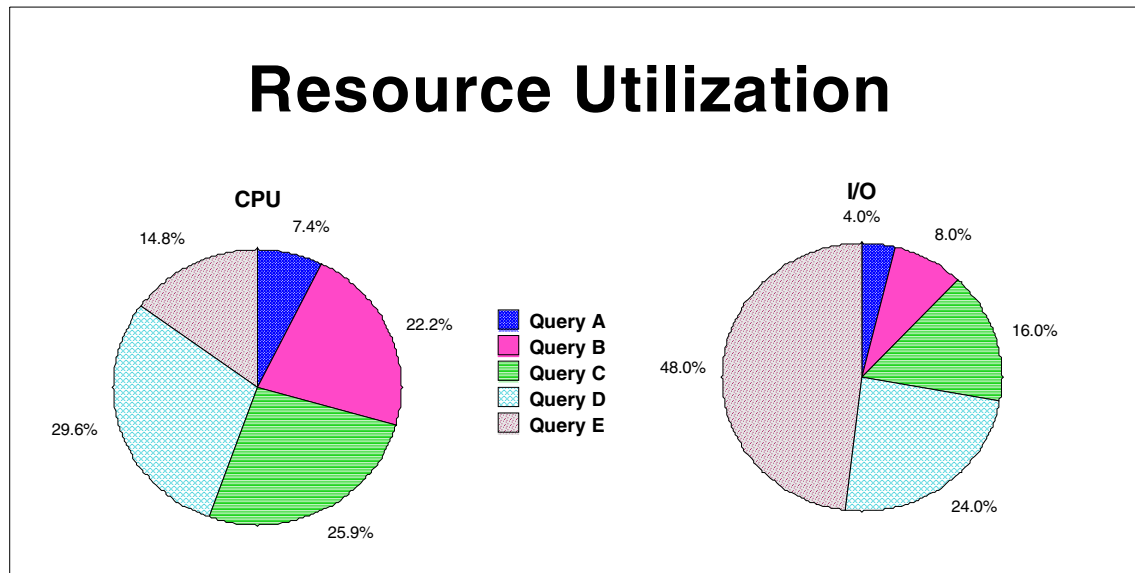


Figure 15. Resource utilization by workload

From Figure 15, you can see an increase in query workload B, which may represent a specific business unit, and which will require much more CPU resource than I/O bandwidth. On the other hand, increasing query workload E will most likely require more I/O bandwidth than CPU resource. A chart like

this simply gives you an indication of what you need to look at as workloads grow in a computing environment.

6.2 Data growth

As business areas provide their estimates for data growth, ownership of the data should be clear. Often, tables will be accessed by multiple business groups. Establishing ownership helps ensure that only one business area or application submits growth estimates for each table, to avoid overstating growth and overlooking tables. Using naming conventions for tables or table creators is one way to group a business application's tables and easily delineate ownership.

6.2.1 Considerations for data growth

When working with the owners of data to estimate the growth of that data, there are a number of aspects to consider. The volume of data can grow either through historical means (adding new data and maintaining old data as well), by the addition of new information into an existing application (such as when new customers, products, or stores are added to a database), or by increasing the data constructs within the database. Growth resulting from adding new tables and data for new applications will be considered in the next chapter.

Changes to the data in a BI environment can be due to new requirements from the users, or due to a corporate directive. Perhaps when the data warehouse was originally implemented, there were only 50,000 customers within the database, and now the company serves 150,000 customers. Perhaps analysts only needed three years of data initially, and now require five years for their analysis. Perhaps the marketing division would like to add data from all departments in a store, instead of just a few of the departments.

Typically, the historical data grows throughout the first years of an application's implementation. Generally, customers may have as much as one year of historical data when beginning an implementation. Additional data is added over the following years. When the number of years of data specified in the original design is reached, older data is archived off the system as time moves on. These examples are illustrated in Figure 16 on page 91.

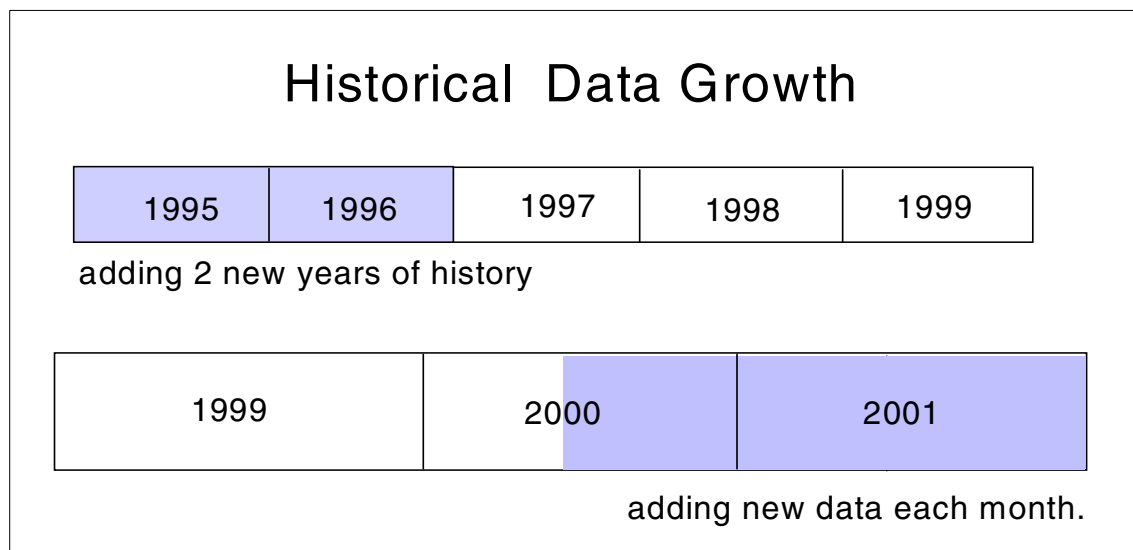


Figure 16. Historical data growth

When data is added to an environment, it can impact the DASD, and the CPU requirements related to the queries and batch workload associated with the application in the following ways:

DASD

The impact data growth has on DASD is the easiest to account for and is calculated in the same manner in any capacity planning exercise. In general, there is a direct correlation: doubling the amount of data doubles the amount of storage required. There is always a physical disk to raw data ratio that is characteristic of a particular hardware architecture and application implementation. Once this ratio is developed, it can then be used to determine the amount of physical disk needed to accommodate the application's data growth. Simply multiply the amount of raw data being added to the application by this ratio to calculate the necessary disk space. This ratio is calculated from data in Worksheet 3 of Appendix A, "Sample worksheets" on page 147 — gigabytes of used DASD divided by the gigabytes of raw data, which yields the physical disk to raw data ratio. This was described in 4.2.3, "Profiling data usage" on page 58.

CPU requirements

The CPU processing requirements of the query and maintenance workloads of an application can be impacted by the growth of the data, thus increasing the overall CPU resources for the workload.

- Queries

Data growth impacts queries, and that impact varies. To determine the effect of having more data in the data store requires taking a broad view of the query profile. Trivial and small queries are minimally impacted since these queries rely on efficient use of indexes to minimize the amount of data accessed, so additional data has little impact in this case.

Large queries, however, may see an increase in resource consumption. Large queries often involve scanning large tables in the database. A query may not be interested in the newly added data, but may experience an increase in CPU time if a majority of the tablespace is scanned. Certain table design factors can sometimes shelter queries from data growth. Partitioning by keys that are time-based may make additional data transparent. The DB2 Optimizer may be able to determine the subset of partitions that need to be accessed based on the selection criteria in the query, and thus avoid the additional data.

Queries that do access all the data in a table may experience a nearly linear growth in resource consumption. If the data in the table is doubled, and the query scans the entire table, it will probably take twice as long to execute. The determining factor is the number of additional data pages being accessed, which results in an increase in CPU time and elapsed time.

- Maintenance processing

Data growth has a significant impact on the ongoing maintenance CPU requirements of a business intelligence environment as well. The impact of data growth on batch processing varies according to the batch job function. Data growth may have no impact on daily updates flowing from an OLTP system to the data warehouse. If the data growth is historical data, there will be no impact, however if the data growth is for additional stores/departments, it may impact the volume of data added in the periodic updates.

Those maintenance processes that run against the total data store, such as backup and reorganizing the data, are significantly impacted as the data grows. These utility functions can be run at the partition level. Thus, if the data is partitioned on a time-based key, it is possible to backup/reorg only newer partitions. Older partitions can be backed up and reorganized once data is no longer being added to these partitions. This can reduce the regular batch processing requirements of the application.

Data growth can have a substantial impact on the CPU processing requirements of the query and maintenance components of a workload.

6.3 User query growth

When considering the growth of a BI workload, end users think in terms of data and user growth, rarely in terms of query growth. It's clear that the number of queries executing is related to the users that have access to a system, and it is that relationship that determines the increase in the workload and processing requirements. It is important to discuss this with the end-user department representative, to establish the user-query relationship.

6.3.1 Understanding user growth

To understand the impact of growing the users, it is necessary to consider what information we have about the users. Do you only have the total number of users accessing the system, and are you therefore capacity planning at the total population level? Or have you been successful in segmenting the users into business units? Perhaps you were able to segment the total population by the end-user tool they are using to access the system. If you have segmented the users, is the growth related to a specific segment? For example, if you were able to segment the business users into the marketing and sales departments, has the marketing department decided to grow their subset of the user population by 20%? If they are segmented by end user tool, is the growth related to one of those groups? For example, is the corporation planning to add 20% more QMF users? It is important to understand which users are growing to understand their impact on the query mix. Business departments often group users into the following classifications:

Operational users are those users whose queries perform reporting-type functions. Their actions tend to consist of shorter queries that use the indexes for highly efficient data access. From a query segmentation view, these users generally submit trivial and small queries into the system.

Analytical users tend to submit more intensive queries to the system. There are generally fewer of these individuals in an organization, and the types of queries they submit vary. Over time, as they learn more about the information, they tend to develop more resource-intensive queries. While they submit queries that span the entire spectrum of queries, these users tend to submit more medium, large, and some extra-large queries. They contribute a relatively small amount of small or trivial queries.

Data Mining users are those using software that detects patterns in the data. Generally, an extraction is performed to pull a subset (10 to 50 GB) of data from the business intelligence system; this data is then analyzed separately from the database. These extractions require significant resources, and are

generally in the large or extra-large query segment. Generally, only one or two people run data mining operations in a department.

When discussing user growth with the end-user department representative, they may know that the growth will be only operational users, or that the growth will include a specified number of analytical or mining users. Having this information will help in determining the true capacity requirements for the new users.

6.3.1.1 OLAP users: analytical or operational

A new group of software products, online analytical processing (OLAP) software, allows users to do in-depth analysis of data online. The multidimensional products now available use pre-built cubes. These users demand significantly less processing capacity to do an analysis than they did in the past. Prior to the delivery of OLAP tools, detailed analysis was done through complex processing and I/O-intensive queries. As OLAP products become more prevalent, they may reduce the resource requirements of the existing analytical users. It is important to note that the products often move the processing demands from the query window to the batch window, which is needed to update and generate the summarized data for the cubes. The actual impact of a particular OLAP tool varies, and understanding its impact should be part of the planning phase when introducing one into the business environment. Consider whether the users are doing this type of analysis through queries currently running in the system (in which case introducing OLAP tools will eliminate some of the more intense queries in the workload), or will this be additional analysis not done before, therefore generating added workload on the system. This type of product will be considered in depth in Chapter 8, “Estimating model size for IBM DB2 OLAP Server” on page 123.

6.4 Worksheet 5: User growth projections

Capturing as much information as possible related to the growth of a workload will improve the accuracy of the capacity projections. Using simple numbers, like 20% growth of users, without understanding where they fit in the query profile spectrum, would require you to assume a 20% across the board increase to the current requirements. That may be adequate. However, if that growth is in the marketing department where the users have the characteristics shown in Figure 11 on page 63, then they will most likely be submitting more processing-intensive queries, rather than trivial or small queries.

Understanding where the growth is occurring by using query segmentations based on business units, end user tools, or some other basis, will result in more accurate capacity projections.

The remainder of this section deals with the actual steps necessary to calculate the increased processing capacity to accommodate an increase in users. It describes how to complete Worksheet 5, which is found in Appendix A, “Sample worksheets” on page 147.

Data growth

The first section of Worksheet 5 provides some key questions to ask the business users related to data growth. This information can then be used to determine the impact the added data will have on the workload. Establish discussions with the user department and determine the answers to the questions in this section. You are trying to develop a percentage increase in the data that will be added to the data store. This is calculated by dividing the amount of data being added by the amount of data currently in the data store (GBs added/GBs current). The answers to the questions should help you determine the gigabytes being added to the application. Be sure you develop a composite number for all the business user groups that use the data store.

User growth

The user growth section of Worksheet 5 lists a series of questions to ask the business users accessing the system. When they state there is an increase of users, you need to understand what type of users. Are they the high-level reporting users, or the analytical users that develop complex, CPU-consuming queries? To understand how the increase in number of users will impact the capacity requirements of the system, ask knowledgeable people from the business department questions about what segment is growing. Worksheet 5 has sample questions for this discussion.

6.4.1 Estimating the data growth factor for the query workload

In general, the impact of growing the data will have significantly different impacts on the CPU requirements for the various types of queries in the workload. Trivial and small queries are hardly impacted with data growth since they often use highly efficient access paths to locate the data in the tables. Large and x-large queries will most likely scan more data to locate the needed records and are significantly impacted when the data is grown. This is reflected in the following table.

Look to the DBAs to estimate this impact for the workload you are growing. If the DBAs cannot give you an approximation for the impact of the data growth on the various types of queries, use some general numbers such as:

Query type	Growth impact
Trivial	0.05 x
Small	0.1 x
Medium	0.5 x
Large	0.8 x
X-Large	1.0 x

These estimates can be used to determine the *Data growth factor* when completing Table 5-1 on Worksheet 5. The true impact may vary for each environment, so try to determine these impact estimates by query type for your own location. The numbers in the above table are estimates from customer experiences, as seen by the Business Intelligence Technical Support Team at IBM. Generally speaking, as you descend the table, the amount of data being scanned by the queries is increasing, and the impact of added data is more significant on the query CPU resource requirements. A general observation is that as the data scanned doubles, so does the amount of CPU resource required. This relationship is reflected in the table, and could be used if there is no understanding of the overall processing requirements of the workload.

To use this estimate, the *Percent data growth* (as provided by the user department) is multiplied by the *Growth impact* listed for the types of queries, above, and the result is listed as the Data growth factor column of the following table. As you can see, for trivial queries the data growth has minimal impact, while for larger queries the impact is significant.

Query type	Growth impact	Percent data growth	Data growth factor
Trivial	0.05 x	50 %	0.025 %
Small	0.1 x	50 %	5 %
Medium	0.5 x	50 %	25 %
Large	0.8 x	50 %	40 %
X-Large	1 x	50 %	50 %

The Data growth factor is then used in Table 5-2 of Worksheet 5 to estimate the data impact on the CPU requirements of the workload, discussed in the next section.

6.4.2 Calculating CPU growth for query workload

The composite CPU requirement due to growing both the queries and the data is calculated in Table 5-2 of Worksheet 5.

Step 1

Determine if you will be working at the total population level or at a business segment level. If at the segment level, establish what user segments in the population are growing, and try to identify the types of queries those users are generally submitting. The sample questions may help you do this.

Step 2

Table 2. Capacity growth (Table 5-2 from Worksheet)

Query type	% of total CPU	User growth factor	New CPU reqmt (user growth)	Data growth factor	% Current CPU resource	Additional capacity
Trivial	5.3%					
Small	4.2%					
Medium	8.2%					
Large	24.0%					
X-large	18.6%					
Total	60.3%					

Start completing Table 5-2 from Worksheet 5 for each segment in the user population that is listed in the table. The query types in column 1 were defined by your organization when you completed Worksheet 3. On Worksheet 3 you recorded the % CPU for each of the query segments. Now you must apply the capture ratio for this environment to derive a more accurate value of the required CPU resource to run the workload. We have decided to use 10 % across all the query workloads to simplify this calculation. Enter this adjusted value in column 2 of Table 5-2.

Complete the next column by listing the growth factor based on the increase in the number of users as provided by the end-user community. Multiply the value in column 2, the CPU % for the query segment, by the increase in number of users by segment (column 3), to determine the New CPU %, to be recorded in column 4.

For example, in the above table, the % CPU of the currently installed system consumed by each query segment is listed. This adds up to a total of 60.3 % of the installed processing capacity used by this workload, as adjusted by the 10 % Capture Ratio uplift. The population of users that generate this query workload is expected to grow by 20 %. If you have no additional information about the users, then you might assume an across the board increase of 20 % for each segment of the population, and you would multiply the total by 1.2, getting a 72 % use of the currently installed system, an increase of 12 %. Applying the growth factor for the entire workload is the safest assumption when no additional information is available. It yields a conservative growth number with some room for error on the part of the users who are providing the information.

However, if the 20 % will be only operational users, who generally submit trivial or small queries, then the increase might only be against those two query types, which only account for 9.5 % of the CPU consumption. A 20 % increase in the 9.5 %, is only 1.9 % more capacity. If a data mining product is to be used more heavily, these added runs will generate additional large and x-large queries in the system. These segments of queries consume 42.6 % of the CPU resources needed for this workload already. Increasing that by 20 % requires 8.5 % more resource. Note that identifying the precise segment of users will reduce the assumed capacity upgrade needed by not growing workloads that are not projected to be increasing.

When deciding how to apply the growth in users to a population, be conservative and assume the total population unless you are very comfortable with growing by query type.

Step 3

Step 2 calculated the increase in processing resource due to the added users who will be submitting more queries into the system. It is also important to include the impact of growing the data in the database being accessed by the users.

The Data growth factor, the last column in Table 5-1, is copied into column 5 in Table 5-2 of the worksheet, to estimate the data impact on the CPU requirements of the workload. Multiply the CPU utilization in column 4 of by the impact the data growth will have on the CPU capacity.

For the entire query workload (trivial through extra-large), the projected CPU requirement to run the workload and its growth (both user and data) is 98 %

of the currently installed system, a growth of 37.7 % over the currently used resource, as illustrated in Table 3.

Table 3. Calculating overall capacity requirements

Query type	% of total CPU	User Growth factor	New CPU reqmt (user growth)	Data Growth factor	% Current CPU resource	Additional capacity
Trivial	5.3%	20%	6.4%	0.025%	6.6%	1.3%
Small	4.2%	20%	5.0%	5%	5.3%	1.1%
Medium	8.2%	20%	9.8%	25%	12.3%	4.1%
Large	24.0%	20%	28.8%	40%	40.3%	16.3%
X-large	18.6%	20%	22.32%	50%	33.5%	14.9%
Total	60.3%	20%	N/A	N/A	98.0%	37.7%

Note the significant increase in the CPU requirements for large and extra-large queries when they are scanning the data to develop the answer set. Substantial increases in data can require significantly more CPU resources for this query segment. If we ignored the impact of the data growth on the CPU requirements, we would have calculated only the 20 % user growth, for an increase of 12 % CPU, not the 37.7 % we will actually need.

Another approach to calculating the increased processing requirements due to data growth is to apply the data growth across all query types equally, over the 20 % user increase. The result would be high, but its simple and provides a cushion of CPU capacity. In this example, the calculation (72.3 % x 1.5) yields a value of 108.5 %. So, you would need 108.5 % of the currently installed system to run this workload. If you don't have the queries for the workload broken down by segment, you can see that this method would certainly give you adequate CPU resource, with a slight cushion.

If you do have information segmenting the query workload by business users or end-user tools, Table 5-2 should be completed for the composite workload accessing the data store. Knowing the relative capacity of the installed system, you can then multiply that by the required increase to figure the total capacity for the query workload.

To summarize this process, complete the columns in Tables 5-1 and 5-2, indicating the growth factor due to an increase in the number of users provided by the end-user community, and the projected impact the data growth will have on the queries. Multiply the User growth factor and the query CPU percentage in column 2 to derive the New CPU % requirement. Next,

multiply this intermediate CPU utilization by the Data growth factor to derive the capacity required for this type of query in this business segment. Once you have the capacity required to run the workload, subtract the current CPU %, to determine the increase over the current system requirements for the workload.

6.5 Maintenance workload growth: Worksheet 4

To complete Worksheet 4, described in the previous chapter, you gathered information related to the maintenance workload associated with the business intelligence application. This workload performs a diverse set of functions for the environment, including maintaining the data by updating and deleting data stored within the database, reorganizing the data to maintain performance, as well as backing up the data for disaster recovery purposes.

In Worksheet 4, you identified the key jobs that make up the batch jobstream for the workload, when they executed, and identified the overall start and stop time for this jobstream. Again, the assumption is made that the batch jobstream has been properly designed and tuned to optimize the processing time required. If not, tuning should be considered as a separate activity. The time when the peak load for update processing is completed should be evaluated since that will be the gating factor in this workload. The basic question is, will it fit in the allotted window?

From Worksheet 4, users should provide their growth rate for the data. Using that, calculate the growth in the batch jobstream by multiplying the total elapsed time from the jobstream (column 4 of Table 4-2), by the growth factor for the data (gathered from the users, in the first section of Worksheet 5, item 4 in the data growth section). The projected Elapsed Time is the clock time needed to complete the work. Compare this to the time available for update for the system (from Table 4-1). If the batch jobstream fits in the window, you are okay. This is a simplistic high level analysis, but it tells you what you need to know.

If you exceed the window, your choices are to add more capacity or review tuning efforts on the batch stream. Can piping technology be used to shorten the stream? Can you start the batch processing when you have the majority of the records received and process the last transmission in parallel once it's received? Determine if there is a way to tune the jobstream to take advantage of other engines on the system. If not, then you need to upgrade the system to an engine speed that can handle the required increase in data in the allotted window.

6.6 Summary

This chapter wraps up the methodology of capacity planning for an existing business intelligence workload. It provides a methodology to calculate the capacity requirements for the query component and the maintenance component of a workload. Generally, one or the other is more CPU-intensive, and therefore, is the component of the workload the system must be sized to handle. This chapter offered a way to determine the impact growing both the user population and the data will have on either component.

Chapter 7. Sizing methodologies for new applications

In the prior chapters we discussed capacity planning for growing Business Intelligence applications that have been in production for a period of time. Another aspect of workload growth is the addition of *new* BI applications. For the purpose of this chapter, new applications have any of the following characteristics:

- New business users with new processing requirements against existing data structures in the data warehouse
- New BI data structures which may be accessed by either existing business users or new user communities
- Existing applications and users where usage information has not been captured

Estimating resource requirements for new BI applications with respect to CPU and DASD is important in order to determine the hardware resources required to implement the application. However, it can be difficult to estimate the necessary resources when there is no production measurement data. In the following sections, we describe how to gather data and develop realistic estimates.

7.1 Fact gathering and iterative estimation

Ideally, with new applications and users, the data to estimate capacity should be gathered throughout all phases of the application development cycle. This allows you to initially develop high-level estimates, then refine them as you move through the various phases of the development cycle.

Worksheet 1, described earlier in Chapter 3, provides a framework for initial sizing discussions with the business community and the development group responsible for implementing the new BI application. This information also will need to be reviewed with the DBAs and the DB2 systems programmer. These people must provide some of the information upon which to base a capacity estimate and will participate in determining whether the existing system can absorb the new workload.

As with any capacity planning effort, determining resource requirements is an iterative process. Each phase of the application development cycle can provide a higher level of understanding about the business usage and associated resource consumption, which can then be used to refine the sizing estimate. In BI applications, there is often a great deal of change over time to

the amounts of data, the types of queries, and the number of users that will be accessing the system. Therefore, it is critical to repeat this process as this application approaches production.

7.2 Planning methodologies

In this section we discuss the methods available to size new BI applications, when to use each technique, and what data each method requires.

To estimate the resource requirements for a Business Intelligence workload, the options are:

1. Rules of thumb (ROT) extrapolation from:
 - BI industry rules, or
 - Customized query profiles, or
 - Formulas and guidelines
2. Modeling via DB2 Estimator
3. Proofs of concept to quantify hardware requirements for extrapolation

The appropriate technique to estimate the BI workload is selected based upon a number of factors, including the accuracy and level of confidence in the projected application workload, the available DB2 and MVS capacity planning skills, the time allotted to determine the hardware resource requirements, and the financial exposure associated with obtaining additional hardware for the new application.

Each of these methodologies is discussed in the following sections of this chapter.

7.3 Method 1: Rule of thumb extrapolation for the query workload

The first method involves using application rules of thumb (ROT) to size a new workload. This is a quick and simple method to estimate hardware resource requirements for new BI applications. As the name implies, it is based upon rules of thumb that are developed from hardware and DBMS vendors' performance studies of BI workloads and customer experiences. Rules of thumb may also be developed using one's own company BI workload profiles, as described in earlier chapters.

ROT extrapolation is most appropriate during the *requirements* phase of the application development cycle, when the business community is only able to provide general data volume and usage information for the new application. Minimum technical skills are needed to use this method. However, we must offer a few words of caution regarding the use of ROT extrapolation (credit is given to *DB2 for OS/390 Capacity Planning*, SG24-2244):

A friendly word of warning

Rules of thumb can seriously damage your health!

Always attempt to verify these guidelines in your own environment. Rules of thumb can gain almost mythological status, at which point they are potentially more harmful than useful. To put it another way, "There are no Golden Rules in this game."

As indicated by this advice, rules of thumb may or may not match your operating environment. Relying on this method can lead to significant inaccuracies in estimating your system requirements.

7.3.1 Industry rules of thumb

Industry rules of thumb for sizing a BI query workload are often derived from CPU and I/O scan rates of large data volumes. These measures are very dependent upon specific vendor hardware and DBMS release combinations, used in conjunction with query workload assumptions. These conditions may or may not correspond to your actual application. Hence, the resulting hardware estimate may vary significantly from what you will actually need. However, when you have little quantitative information from the business community, a wide margin of error must be factored into any method you choose to use.

To develop system resource estimates, a company with little prior experience with BI applications may work with the hardware and software suppliers in order to use rule of thumb extrapolation. Raw data volumes and the size of the user population are the minimum that would be needed for this methodology. Possibly, a better estimate may be jointly developed if you can also provide:

- Number of projected concurrently executing queries
- A query profile for the application
- Typical data volume accessed per query, and/or
- The desired single query elapsed time

If you choose to use industry rules of thumb, it is important to get the most current values from the appropriate vendors, as they often change significantly with each new release of hardware and software products. Never rely on a ROT older than six months, or anything provided within a publication. These will most likely be out-of-date values.

7.3.2 Rules of thumb: custom query profiles

In an experienced BI installation, Query Profiles, as described earlier in this book, may be developed to assist in the initial sizing of a new BI application. Segmentation based on either query complexity (that is, trivial, small, medium, large, and extra large) or business user type (that is, operational reporting user, analytic user, or mining user), along with the associated CPU consumption per segmentation type, is the basis for this methodology.

To the extent that the new user population or new application can be mapped to the profiles, a sizing estimate itself could be as simple as multiplying the new query/user counts times the associated CPU resource. However, we want to reiterate that the prerequisite profiles likely require an investment of your skilled MVS and DB2 personnel working together to produce them. Having established these profiles allows future sizings to be done simply and quickly.

The custom query profiles have the following advantages:

- They reflect the hardware performance characteristics of your environment, not an optimized benchmark system, which could be unrealistic as your ongoing operating environment.
- They reflect the BI usage of your company. Often new applications will be similar to existing applications in the types of work done within the application, i.e. operational reporting, statistical analysis, and data mining.

7.3.2.1 Creating the custom query profile

Query Type	Data Scanned	Elapsed Time	CPU Time	% Query Workload
Trivial	< 20 KB (100 rows)	subsecond	subsecond	
Small	< 2 MB (10,000rows)	1-2 secs	.1 secs	
Medium	<500 MB (2.5 million rows)	2 min	15 secs	
Large	<100 GB	2 hours	1 hours	
XLarge	500 GB	10 hours	5 hours	
Totals	N/A	N/A	N/A	100%

User population _____ Active queries _____ DASD to raw data _____

Figure 17. Custom query profile - Table 6-1

Figure 17 illustrates a sample custom query profile that a company would complete. This table is an example of the Query Profile Table (6-1) in Worksheet 6 in the Appendix A, "Sample worksheets" on page 147. The queries of a workload are characterized by the amount of data scanned, the elapsed time, and the CPU time for each type of query. This information can be easily obtained from DB2 Performance Monitor reports. Table 6-1 is a subset as well as an extension of Worksheet 3 Tables 3-3, 3-4, and 3-5 as described earlier in Chapter 5, "Establishing current system usage" on page 65 and found in Appendix A, "Sample worksheets" on page 147.

DBAs are generally the best source of this information you need when developing this profile. The measured query workload of an existing application can be used as a model for the new application, or the information for Table 6-1 can be newly estimated by the DBA, the application analyst, and the business user department that will use the application. The columns are:

Data Scanned

An estimate of how much data typically is processed by a query in the query segment. This may be derived from Table 3-5 for each query segment. It is

the $((\text{Total Getpages per Day})/(\text{Avg Total/Day})) * 4\text{KB}/\text{page}$ to obtain Avg KB/query. If an estimate of rows is preferred, the Avg KB/query may be divided by the average row length which can be derived from the DB2 catalog.

Elapsed time

The target elapsed time for a query in the query segment. This is obtained directly from Table 3-4, Avg ET.

CPU time

The target CPU time for a query in the segment. This is obtained directly from Table 3-4, Avg CPU.

% Query workload

The percentage of this query segment of the total query workload. This is obtained directly from Table 3-5, % of daily total.

User population

The total number of people who have access to the BI applications from Worksheet 3, User Profile.

Active queries

The average peak queries in the system. This may be obtained directly from Table 3-3, Avg peak.

DASD to raw data

The ratio of DASD to raw data used to support the BI application in DB2. This may be derived from Worksheet 3, Data profile and is calculated as (GB of used DASD)/(GB of Raw Data).

7.3.2.2 Estimating hardware requirements

To estimate the CPU and DASD hardware needed for the new application, Table 6-2 in Appendix A, "Sample worksheets" on page 147 may be used. Table 6-2 is an extension of 6-1 described above. The additional fields are:

Projected user population

An estimate of the total number of people who will have access to the new BI application.

Projected raw data

An estimate of the total raw data to be stored for the new application.

Projected active peak queries

An estimate of the maximum number of active queries in the system. This may be obtained directly from Table 6-1 or estimated as 1% to 2% of the projected user population as described in 2.6, "User population" on page 29.

Projected DASD

The amount of additional DASD to support the new BI application. This is calculated as (DASD to raw data) * (Projected raw data). The DB2 DASD to raw data ratio for BI data stores typically averages 1.5 to 2.5, with the lower end for implementations using DB2 data compression, little updating of existing data once loaded into the data store, and few indices/table. Two is a reasonable ratio to use if your own BI data profile is not available.

% new workload

The percentage of this type of query in the new query workload. For new workloads in which the user interaction with the application has been defined, the application analyst with the business group would complete the last column, providing estimates of the percentage each query segment type that would contribute to the new workload. Or, if the query application flow has not yet been defined, they may be obtained from Table 6-1, % Query workload.

Projected CPU

For each query segment type, this is calculated as (% of new workload) * (Projected active queries) * (CPU time).

7.3.3 Query ROT technique comparison

Why go through the effort to create a custom query profile, instead of using the simpler ROT approach? The fact is that early in the planning cycle either method is acceptable, and if the Query Profiling is already done, they each take approximately the same amount of time.

The effort the capacity planning staff must put into the Custom Query method may result in a more accurate and reliable estimate due to extrapolation from your operating environment and BI application flows. However, to insure this accuracy, the typical Query Profiles for Business Intelligence workloads must be maintained over time. As users mature, their usage patterns change, and this historical view of usage should be captured and maintained as a model of how the business users at your company develop. It would be reasonable to assume that new users would have the same developmental profile over time.

Rule-of-thumb estimation results in adequate sizings when the potential for a sizable margin of error is factored into the acquisition process. This tends to

be a fast, easy to use approach to developing a sizing estimate; however, these ROTs change quickly, and numbers quoted six months or longer should not be used. Additionally, if a new version of either the hardware or software product is being proposed, there will be no true ROT to use until it can be exercised with actual business workloads. In these situations, the margin of error should be increased.

Finally, both methods are subject to error due to inaccuracies of business user estimates. Some things never change.

7.3.4 Method 2: formulas/guidelines for batch workloads

The next method specifically addresses batch or maintenance workloads, and is not applicable to sizing query workloads, which can be estimated using the other methods in this chapter. Some customers have found the maintenance workload to have substantially higher resource requirements than the query workload, especially in the early rollout of the BI application when few users are accessing the system or when the BI data must be maintained nightly. This method is often used in conjunction with one of the other methods, to determine the overall hardware needed to support a new BI application.

In Chapter 3, we look at the process that consists of extracting data from source systems, transforming it to a consistent format, then adding the data into the data store. This sequential batch process must be completed within the available batch windows, varying from a few hours each night to as much as a period of days, and sometimes requiring more computing capacity than the query component.

The batch workload often does not have the dynamic, volatile nature characteristic of query workloads. The batch component tends to be predictable, often consisting of traditional COBOL or other fourth-generation language (4GL) generated or interpretive code programs, permitting the use of formulas and guidelines that have evolved over many years of DB2 application implementation. Additional information may be found in the redbook *DB2 for OS/390 Capacity Planning*, SG24-2244, Chapter 3.

In recent years, the ETL batch workload may also be processed via SQL scripts. All logic is accomplished via SQL only, similar to SPUFI or DSNTEP2 programs. Many, many rows of data, sometimes millions of rows, are processed per SQL statement. For this type of batch processing, we do not recommend the use of this method. Instead, Method 3 - DB2 Estimator, which is described later in this chapter, may be used.

This method can be used during the design phase of the ETL process when input volumes and corresponding DB2 table access have been defined. Assumptions and characteristics of the batch processing are:

- 2-3 way table joins in the SELECT statements with matching index access
- Few rows scanned/qualified/returned per SQL DML statement
- The DB2 processing and 3GL/4GL generated business logic code are approximately equal in CPU costs

7.3.4.1 Table 6-3: Batch ROT Sizing Estimate

To estimate the batch/maintenance workload for a new application, you need to determine the key tasks to be completed: the volume of data being extracted, the processing work that must be performed to standardize and transform that data, and finally, the processing requirements to add the data to the data store. Each aspect of this processing must be estimated to establish the CPU capacity requirements for the workload.

When estimating this work, it is important to understand the frequency of the maintenance workload, and plan to the peak for this workload. When discussing this workload with the application analysts and business users, it is critical to identify these key characteristics of the workload, particularly what will be the expected normal processing vs. peak update volumes. This is then mapped against the batch window during which this processing must be completed, to develop a complete picture of the workload's needs. In some cases, this maintenance workload can have substantially higher resource requirements than the query workload.

To aid in developing the batch capacity requirements, use Table 6-3 from Worksheet 6. As part of this table, rules-of-thumb for processing SQL statements and associated application logic are used. The hardware/software-based ROTs listed across row 2 of the table vary with the versions of hardware and software used to implement the solution. They are derived from DB2 pathlength estimates and many years of customer design reviews. The values provided in this book apply to current levels of hardware and software, but should be validated in your environment. Your DBAs or vendors would be appropriate sources for this type of information.

These ROTs are expressed in milliseconds per SQL call. This is the CPU time in milliseconds needed for each SQL call plus an additional percentage for associated business logic.

To simplify the use of this table, it is also being provided as a download file for 1-2-3, at Web site:

<http://www.redbooks.ibm.com/SG245689>

To help identify the ROTs and fields you need to complete, they are color-coded. The editable ROTs are yellow, and your batch programs with SQL statements quantities are green in the spreadsheet, medium grey here in Figure 30. These fields contain unique data for your application and operating environment. The following discussion highlights how to use the table/spreadsheet.

For this table, you must first identify each job that is in the critical path batch job stream for the projected application. As an example, the key jobs are listed in the first column of the table in Figure 18 on page 113. To estimate the capacity for this work, you next need to identify the expected number of records to be processed during each job that is listed in this table for the extraction and transformation steps. This is best estimated by the application designer who will be responsible for identifying the data to be extracted from operational sources and the transformation processes required to standardize this data.

Batch ROT Performance Estimates							
RoTs (ms/SQL call)			1	2	20		
Critical Path Program/Plan	# Input Records	# SQL Calls/ Input Record	Estimated CPU Minutes	Estimated I/O Minutes Minimum	Estimated I/O Minutes Maximum	Elapsed time Minutes Minimum	Elapsed time Minutes Maximum
Extract 1	250,000	2	8	17	167	25	175
Extract 2	1,000	5	0	0	2	0	2
:			0	0	0	0	0
Transform 1	1,000,000	5	83	167	1,667	250	1,750
Transform 2			0	0	0	0	0
:			0	0	0	0	0
Add/Update 1	1,000,000	3	50	100	1,000	150	1,050
Insert	1,000,000	1	17	33	333	50	350
:			0	0	0	0	0
			0	0	0	0	0
Total Time			159	317	3,168	476	3,327
Batch Window (hours)		5					
Batch Safety Factor		40%					
Parallel jobstreams needed						3	19
Engines needed			1			2	1
Utilities - DB2 Estimator preferred							
ROTs	LOAD	250	MB/minute, one index, inline stats				
	# Input Records	DB2 Rowlength	Elapsed Time Minutes				
LOAD 1	1,000,000	250	1.0				
COPY 1			0				

Figure 18. Batch Rule of Thumb: Performance Estimate - Table 6-3

The number of input records to be processed and the average number of SQL calls required per input record must be supplied. Consult the DBA to validate the ROTs listed in the second row of this table/spreadsheet.

The next step is calculating the remaining values in the table. The easiest approach will be to walk through the table, discussing the calculations for each column, across the table. The spreadsheet will automatically complete these remaining cells.

Estimated CPU Minutes

In the fourth column, starting in the row for Extract1, the estimated CPU minutes is calculated by multiplying the ROT of estimated CPU ms/SQL call (row2), by the number of input records (row3-col2), by the number of SQL calls/input record (row3-col3). Finally, since the ROT is in milliseconds, it is necessary to divide by the number of milliseconds in a minute (60,000).

Estimated I/O Minutes (Minimum/Maximum)

The fifth and sixth columns in the table list both the minimum and maximum value for the I/O time. The minimum time assumes DB2 sequential prefetch I/O is occurring; DB2 is pre-staging the data in 32 page blocks before the application program requests it, little I/O wait occurs. The maximum time assumes all synchronous I/O is occurring, and therefore, the program must wait for the data to be retrieved. This is calculated using the ROT for each column listed in row 2, multiplying by the number of input records in column 2, and by the #SQL calls/input record in column 3. Finally, divide by 60,000.

Elapsed Time Minutes (Minimum/Maximum)

Once the time needed to complete the I/O is determined, add the minimum time to execute the I/O (col5), to CPU minutes (col4) necessary for Extract1, and record that value in the minimum Elapsed time column. The minimum value is additive, rather than the maximum(CPU time, I/O time) to allow for some skip sequential processing with I/O wait. Repeat this for the maximum value (col6+col4). This gives you the range of elapsed time necessary to process each job.

7.3.4.2 Load vs. insert processing

The final step in the ETL process is loading the data into the DB2 database. This process can be completed either with SQL DML statements, or by using the DB2 load utility. The load utility is the fastest method of getting the data into the database, however, it simply reads a sequential input file and adds the records to the table. Conditional processing accessing other rows in the database in order to validate the add or update must be done with a SQL program. Once you have identified the method of adding data to the relational database, complete another line in Table 6-3 or the spreadsheet for the load or insert process.

Inserts of data

Inserts are more complex, and therefore will take more processing power to execute than load utilities. Since inserts are a DML statement, it has the same calculation as the SQL statements for the transform/extraction, and are found in the upper portion of the table/spreadsheet. Follow the same calculations as described in the previous section.

Loads of data

The load estimation is based on some hardware ROTs expressed in load rates, the amount of data that can be loaded per unit of time by a single CP or engine (for example, the number of Megabytes loaded per second per CP). This ROT value can vary with platform, architecture, and disk subsystems as well as the version of the Load utility. The ROT may be validated through your own experience with the DB2 Load utility. Log No is assumed in this ROT.

Note: The IMAGE COPY utility may be executed after the LOAD, but no time is allotted to this, since it can usually be run concurrent with query processing.

If you do not have experience with DB2 or would like a more accurate estimate for a specific system configuration, the DB2 Estimator described later in this chapter may be used. For load utility estimation, it is quick and easy to use. Otherwise, ask the vendor for an approximation of what the platform is capable of achieving. The spreadsheet found on the Web site has the load rate for the platform at 250 MB/minute. This may or may not apply to your environment.

7.3.4.3 Final calculations

Totaling the last two columns will give a reasonable estimate of the CPU and elapsed time necessary for the extraction, transformation and insert steps of the ETL process. This needs to be compared to the available time.

Batch window

This field has the hours the system will be available for batch processing. It is developed jointly by the application analysts, the operations staff, and DBAs in order to factor in existing job schedules that must precede the processing of the BI data, i.e. transmission of input data from other sites or prior operational processing before extracts can be initiated, and the time at which the cleansed data must be available for query access.

Batch safety factor

This field is the portion of the batch window typically reserved for delays due to late transmissions and job recoveries for a batch job stream. It should be input as a decimal fraction.

Parallel Job streams needed (Minimum/Maximum)

Batch elapsed times are a combination of CPU and I/O time, with the CPU sometimes waiting for I/O to complete before the program can proceed. In order to meet the batch window requirements, a calculation based upon estimated elapsed time is needed. The parallel job stream row computes the number of concurrent jobs/program needed for completion within the factored

batch window (batch window*(1-minus safety factor). It is the total of the elapsed times of jobs in the critical path divided by the factored batch window. This is calculated for the minimum and maximum elapsed times to provide the range of job parallelism that must be designed into the batch jobs. The absolute numbers are not as important as knowing whether or not a flexible application architecture must be implemented to support varying degrees of job parallelism.

Engines Needed

This row is the range of CPs needed for the CPU component of the critical path job stream. It is a range due to the variability between I/O wait and CPU busy for batch processing. Three values corresponding to CPs needed for no I/O wait, mostly asynchronous I/O wait, and all synchronous I/O wait are calculated:

- No I/O wait

Assuming the jobs are executed sequentially with no application parallelism and zero I/O wait, the minimum number of engines is calculated as the Total CPU minutes / ((60*Number of hours in the batch window)*(1-Batch Safety Factor))

- With I/O wait

If parallel job streams are needed per the previous parallel job stream calculation, ratios of the amount of I/O ms to CPU ms are also needed to determine how many engines are needed to support the number of parallel job streams.

These ratios are computed by dividing the ROTs contained in row 2 of the above table: the asynchronous I/O to CPU ratio is the I/O service/wait time ms ROT (above estimated minimum I/O minutes) divided by CPU ms per SQL call and the synchronous I/O to CPU ratio is the I/O service/wait time ms ROT (above estimated maximum I/O minutes) divided by CPU ms per SQL call. To compute the engines needed with I/O wait:

- Mostly asynchronous I/O wait: #parallel job streams for estimated minimum elapsed is divided by the asynchronous I/O to CPU ratio to derive the engines needed
- All synchronous I/O wait: #parallel job streams for estimated maximum elapsed is divided by the synchronous I/O to CPU ratio to derive the engines needed

The number of engines needed is the maximum of the three values.

The spreadsheet automates the calculations for this batch sizing. The above text outlines the calculations necessary to determine the CPU needed to process the batch ETL workload.

7.3.5 Initial load estimates

A unique aspect to a new Business Intelligence application that is being brought into production is that it requires an initial load of data into the database. The volume of data to be transformed and loaded is often several years' worth of data. Without this large initial amount of data, the application may not be of value to the users.

When establishing the initial data requirements of the business users, it is important to ask them how much historical data is required for this initial load of the database: one year, two years? This data must be accumulated over time, for it to be loaded. Will they require weekly backups of data? Daily backups? Understanding the data needs of the users is necessary to make sure you gather and load the necessary information.

To calculate the load time for the initial load, use the utilities segment of the table/spreadsheet. If the data must be transformed, use that section as well. This processing can take days, and it is important to estimate these one time requirements, so that the necessary resources can be planned for.

An easy way to calculate the initial load is to use the DB2 Estimator tool. This tool makes it easy to estimate, as explained in the following section. Additionally, *DB2 Performance Topics*, SG24-2213, has calculations for this task.

7.4 Method 3: Modeling with DB2 Estimator

DB2 Estimator is a predictive modeling tool that estimates the CPU cost and elapsed time associated with the execution of specific SQL DML statements and DB2 utilities. It also provides estimates of the DASD space required. This is a free feature of DB2 for OS/390 Versions 5 and 6, which can be downloaded from the Web site:

<http://www.ibm.com/software/download>

The program runs on your desktop PC or laptop.

Modeling using DB2 Estimator to estimate resource consumption of an SQL DML statement will provide far more precision than the previously discussed methods. However this tool provides valid estimates only if accurate, detailed information is used as input.

This includes:

- Information related to the characteristics of the statement being modeled
- A detailed understanding of the application workload
- A good knowledge of DB2 SQL processing
- The details of the physical database design

However, with BI applications, this type of accurate information may not be available for the query workload until you are close to implementation or even after. The quality of the results obtained with this tool relies on the accuracy of the input data.

In contrast to ROT extrapolation, DB2 Estimator is better to use during the last steps of the design phase and during the code/test phase of the BI application development cycle. More accurate DDL, SQL DML, and data distribution statistics are usually available at this time, which can be easily loaded into DB2 Estimator from the DB2 system. This information can be used to develop the CPU cost and elapsed time for various workloads of the Business Intelligence environment.

DB2 Estimator can be used to understand the elapsed time for a query, as well as the impact the SQL statement has on hardware resource consumption. It is also possible to estimate the effect increased data volumes may have on the query execution time. This is particularly helpful when there are significant differences between data volumes in the development environment versus the production environment. Often for BI applications, the large projected data volumes for the production system can be orders of magnitude greater than the data stored and accessed in the test environment, making extrapolation of elapsed time and CPU consumption from test to production difficult.

DB2 Estimator is also able to estimate DASD requirements, as well as DB2 CPU cost and elapsed time for the batch data maintenance workloads before the application program is written. It can assist in the evaluation of the physical database or application design to determine if the performance will meet the necessary windows. By inputting table definitions into the tool, you can obtain estimates of the amount of DASD space required, as well as the costs associated with running utilities against those tables.

To summarize, DB2 Estimator can be very effective during the design and construction of a Business Intelligence application. With accurate input, it can provide valuable information for estimating the query and maintenance workloads for a new Business Intelligence application. Additional information

about using DB2 Estimator can be found in *DB2 or OS/390 Capacity Planning*, SG24-2244.

7.4.1 Method 4: Proofs of Concept (POC)

The final option available to size a new application is to conduct a proofs of concept (POC). Proofs of concept can address a number of issues to increase the success of the project. Assessing hardware and software integration requirements, defining and measuring end-user interface ease of use, or simulating production volumes for performance satisfaction are potential objectives. However, significant human resource and equipment may be required to fulfill these objectives, as well a very good understanding of the data design and usage. Consequently, the method is best used late in the design phase or is sometimes merged with either the construction phase or systems test phase of the BI application development cycle.

7.4.2 Functional Proofs of Concept

Proofs of concept generally consist of at least a small, test implementation of the hardware and software components to ensure that the products will work together to satisfy business usability requirements. While performing such a test, it is often possible to gather application usage data that can be used to size the production environment. While the business users may not have access to large volumes of data, understanding the types of queries that may be submitted by the business community is valuable. Using this information with your BI vendors may increase the accuracy of your sizing projection.

Issues often minimized in sizing new BI applications are: determining if there is adequate network bandwidth for the workload, or estimating the size of the application server needed for it. A proofs of concept or pilot environment is usually demonstrated with a minimal number of users. It likely will not stress test the network connections that users rely on to access the data store, nor expose the server requirements needed to support concurrent application processing of the data before results can be displayed to the users. These requirements should also be evaluated to ensure that there will be adequate resources.

Many POCs only test online query workloads, rather than batch maintenance type of work. Once you have captured the SQL statements and their corresponding performance characteristics from the RMF and DB2 monitor data, you can combine it with other qualitative information from the business users to estimate the future resource requirements. It is also important to determine if there are other queries that would typically be executed as part of this new workload, that were not included in the POC test.

All of this information can then be used to either:

- Develop the Custom Query Profile and Sizing Tables described earlier in this chapter, or
- Input into the DB2 Estimator to develop a hardware estimate if data volumes were not sufficiently large for simple hand extrapolation.

In either case, the result will be a better estimate of the resource required to handle this new application, rather than just using rules of thumb.

7.4.3 Performance Proofs of Concept

Other proofs of concept can include assessing the performance of the system as part of the project objectives. Measuring scalability of the system, both hardware and software, for both query volume growth and data growth are typical objectives, as well as functional integration. These POCs require a noticeable investment in hardware, software, and people. However, they are much more likely to provide an accurate sizing if representative data and usage volumes are exercised in the proofs of concept.

As an alternative to customers conducting a proofs of concept at their own location, IBM offers some unique capabilities for customers. IBM has established Business Intelligence Proofs of Concept centers that can aid customers in conducting these tests, including the Business Solution Centers located in the United States and Europe, as well as the Teraplex Centers. The Teraplex Centers have been specifically put in place to address large terabyte Business Intelligence proofs of concepts for customers. Your IBM sales representative will be able to assist you in getting in touch with these centers, to determine if they can help you with your efforts.

The results of a number of studies conducted at these centers have been published in a series of redbooks including *Data Warehousing with DB2 for OS/390*, SG24-2249 and *Building VLDB for BI Applications on OS/390: Case Study Experiences*, SG24-5609.

In summary, this method is particularly useful when an application requires a large hardware investment. In this case, POCs may assist with the validation of the query usage assumptions and associated resource estimates used to price the solution, so that the business users can have an accurate picture of the cost of the system to weigh against the expected business value of the solution.

7.5 Summary

There are a number of methods to estimate the capacity requirements for both the query and batch component of the work. We discussed using industry rules of thumb, as well as query profiles, DB2 Estimator and proofs of concept for determining the processing requirements for this workload. Formulas for calculating the batch component were also reviewed, along with the table and spreadsheet to aid in this calculation.

Which method to use is dependent upon the accuracy of the input data, the time and skill available to produce the estimate, the required accuracy of the estimate, as well as the potential financial investment.

Note: Keep in mind that the accuracy of your estimate can be no better than the accuracy of your input.

Technique	Skill Requirements	Input Accuracy	Effort	Estimate Accuracy
Industry RoTs	Low	Low	Little	Low/Medium
DB2 RoTs	Low	Low/Medium	Little	Medium
DB2 Estimator	Medium/High	Medium/High	Medium/High	Medium/High
Proof of Concepts	High	High	High	High

Figure 19. Summary of new application sizing methods

Chapter 8. Estimating model size for IBM DB2 OLAP Server

When capacity planning for OLAP, we need to estimate the machine requirements for:

- CPU
- Memory
- DASD

Our goal is a machine specification that has sufficient power and storage to build cubes and deploy them to their analysts.

Unlike many other server technologies, where we focus mainly on the base data or the number of users we need to support, the key consideration for OLAP is the time and space required to calculate and store the OLAP cubes. For an OLAP cube to be useful, it must be regularly updated with new data, and then recalculated, before being made available to the analysts. As data flows into the data warehouse at the end of each day, week, or month, we refresh the cube with a new set of data. This operation must be accomplished within a certain time period. Typically, we know how long the batch window is, and how much of that time we can allocate to building OLAP cubes, so we can plan to meet these requirements.

In this chapter we discuss the factors that influence model size:

- Inflation
- Sparsity
- Block density

With an understanding of these factors, we then develop a method for estimating model size that can give us quality estimates based on actual input data, but allows us to avoid building the complete model.

8.1 Inflation

DB2 OLAP Server is a multidimensional OLAP technology. This class of OLAP tools offers incredibly fast and consistent query response times. To achieve such quick results, we build an OLAP hypercube, thereby minimizing the work the server needs to carry out in order to supply data to the analysts.

Unlike the base tables in a data warehouse, OLAP hypercubes contain aggregates. It is these aggregates that result in cubes exploding to many times the size of the input data.

Consider a simple two-dimensional model in which we load monthly data, as shown in Table 4, for sales, cost of goods, and expenses. If we lay out these members as a grid, we have a 3 X 12 grid of 36 cells

Table 4. Example of two-dimensional model

	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
Sales	100	120	170	200	200	200	190	180	185	185	220	250
Cost of goods	75	80	110	140	140	140	120	120	125	130	160	190
Expenses	10	10	10	15	15	15	15	15	15	15	20	20

To add value to this simple model, we add some simple aggregates:

- We will rollup the months into quarters, and the quarters into years.
- We will calculate:
 - Margin (sales less cost of goods)
 - Profit (margin less expenses)
 - Profit and margin as a percentage of sales

Although our model is simple, we have added significant value. Analysts can now track margin and profit, and some key percentages, and we can see how the business is performing at the year, quarter and monthly levels.

We have added 5 members to the existing 12 in the time dimension, approximately a 42% increase in the total number of members.

We have added 4 members to the existing 3 members of the measures dimension, which is approximately a 133% increase in the number of members.

But, if we look at what has happened to our simple grid, it is now 7 X17. That's 119 cells, so our simple cube has inflated from 36 cells to 119 cells, an increase of approximately 300%.

Table 5. Example model with aggregates added

	J a n	F e b	M a r	Q 1	A p r	M a y	J u n	Q 2	J u l	A u g	S e p	Q 3	O c t	N o v	D e c	Q 4	Y r
Sales	I	I	I	G	I	I	I	G	I	I	I	G	I	I	I	G	G
Cost of goods	I	I	I	G	I	I	I	G	I	I	I	G	I	I	I	G	G
Margin	G	G	G	D	G	G	G	D	G	G	G	D	G	G	G	D	D
Expenses	I	I	I	G	I	I	I	G	I	I	I	G	I	I	I	G	G
Profit	G	G	G	D	G	G	G	D	G	G	G	D	G	G	G	D	D
Profit %	G	G	G	D	G	G	G	D	G	G	G	D	G	G	G	D	D
Margin %	G	G	G	D	G	G	G	D	G	G	G	D	G	G	G	D	D

Note the following in Table 5:

- I shows input cells.
- G shows cells calculated (generated) from input values.
- D shows cells derived from generated values.

In Table 5, we have 36 cells (I) that we loaded with input data values. We have added 63 cells (G), where we calculated a value from several input values (such as Q1 sales) or from an input value and a calculated value (such as profit for March). We calculated 20 cells (D) based on derived members in both dimensions, such as profit for Q1.

The 63 additional cells that represent the intersection of input members and derived members are the cells that contribute most to inflation. When you consider this effect across 5, 6, or 7 dimensions, you begin to understand why our OLAP cubes are much larger than the input data. This effect is known as *inflation*. Erik Thomsen suggests that inflation can be calculated as follows:

1. Calculate the ratio of total members to input members for each dimension of the cube.
2. Average the ratio.
3. Raise the average ratio to the power of the number of dimensions.

In the example we used above, we find that the averaged ratio is 1.85. If we take this ratio and apply it to a 5 dimensional model, we get an inflation factor of 21, and for 8 dimensions we get an inflation factor of 137. Therefore, for an

8 dimensional model, every megabyte of data we load generates 136 megabytes of data when we calculate the cube.

Although the inflation factor cannot be used to estimate cube size, it can be used to quickly test your design, as soon as you understand the number of dimensions and the number of members in each dimension. We calculated the inflation factor for a number of cubes, and found an average inflation factor of 3.3, the highest being 4.6.

8.2 Sparsity

Our inflation factor can give us an early indication of a bad design, but it does not take into account the sparsity of the data. Today's OLAP cubes are very sparse. As we build larger models that hold more detailed data, they will become even sparser.

Why is enterprise data so sparse? To answer this question, let's consider a simple example. Widgets Inc., sells Real Widgets. For large retailers it creates branded versions, labeled for the retailer. If we look at the sales figures for November 1999, we might see data like that shown in Table 6.

Table 6. Example widgets model that exhibits sparsity

	Real Widgets	Acme Branded	Gonzales Branded	Robinson Branded
Acme retail		200,000		
Gonzales retail			1,000,000	
Robinson retail	250,000			250,000

Acme, Gonzales, and Robinson retail are not purchasing widgets branded for their competitors. Some retailers may stock both generic Real Widgets, and their own brand. Many smaller retailers will only stock Real Widgets. The matrix across many products and customers will be very sparse.

Now consider a real example such as a bookstore. They may stock many thousands of titles, but each customer will only purchase a very small number of titles each year. Even if we rollup titles into categories before loading the OLAP cube, we will still have a very sparse matrix. Think of the categories of books available at your local bookstore, and imagine the matrix for *your* purchases in the last year. This is *sparsity*.

Sparsity makes an enormous difference to the efficiency with which DB2 OLAP Server can store the data. Using a patented algorithm, DB2 OLAP Server views the dimensions of a model as either sparse or dense.

For combinations of members in the sparse dimensions for which one or more data points exist in the combinations of members in the dense dimensions, DB2 OLAP Server stores an index entry, and a block to hold the data values.

In our Widgets example:

- Product and Customer are our sparse dimensions
- Measures is our dense dimension

Therefore, DB2 OLAP Server will create index entries and blocks to store the sales figures for the combinations:

- Acme Retail -> Acme Branded
- Gonzales Retail -> Gonzales Branded
- Robinson Retail -> Real Widgets
- Robinson Retail -> Robinson Branded

Although there are 12 possible combinations of Product and Customer, we only manage the four combinations for which we have data, so the percentage of potential combinations we store is 33 percent. This ratio, which is usually referred to as the percentage of maximum blocks existing, is key to understanding how many data blocks DB2 OLAP Server must store, how large the final cube will be, and how long it will take to calculate.

We recently carried out some tests using small data sets with varying data densities. The results are shown in Figure 20 on page 128. With the percentage of maximum blocks varying between 0.037 percent and 3.455 percent, we saw storage requirements increase 60 times, and calculation times increase by a factor of 162.

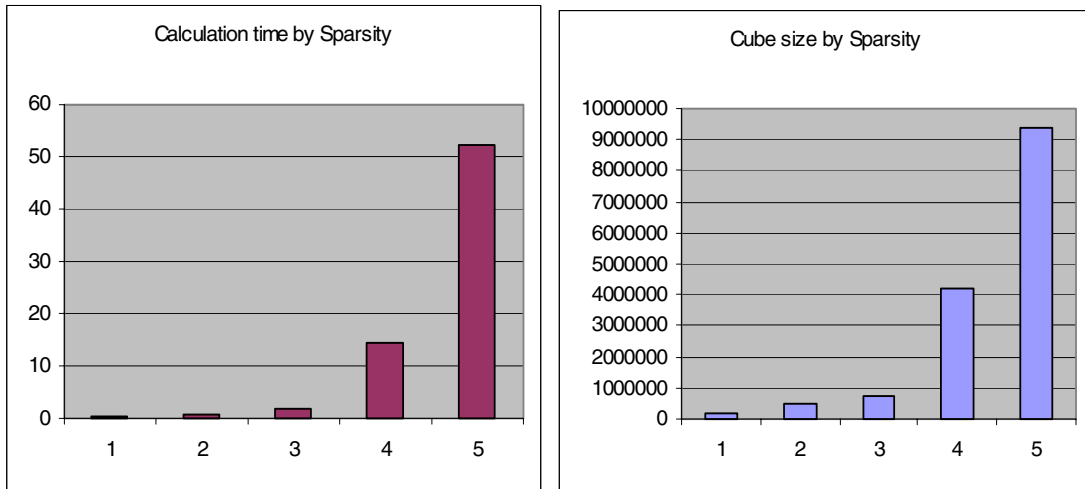


Figure 20. Cube storage requirement and calculation time versus sparsity

Unfortunately, estimating sparsity is very difficult. As E. F. Codd states in his paper *Providing OLAP (On-Line Analytical Processing) to User-Analysts: An IT Mandate*:

...if the enterprise data is sparse, and has certain distribution characteristics, the resulting physical schema might be one hundred times the size of the enterprise data input. But, given the same size data set, and the same degree of sparseness, but with different data distribution, the size of the resulting physical schema might be only two and one half times the size of the enterprise data input, as in the case of the perfectly dense example. Or we could experience anything in between these two extremes. Eyeballing the data in order to form an educated guess is as hopeless as is using conventional statistical analysis tools to obtain cross-tabs of the data.

For many of today's OLAP models, the percentage of maximum blocks existing will be as low as 0.2 percent.

8.3 Block density

The final factor to consider when estimating model size is the character of the dense blocks that DB2 OLAP Server stores. As we saw in the previous section, we store one block for each combination of sparse members where one or more values exist in the dense member combinations. In our simple example, we had just one measure to store from our single dense dimension.

Real models will usually have a number of measures, and more than one dense dimension. *Time* and *Scenario* are frequently the other dense dimensions.

When deciding which dimensions to tag as dense, we must consider the size of the resulting block. We can calculate block size by multiplying together the number of stored members in each dense dimension, and then multiplying by 8. This gives us block size in bytes. We should aim for a block size that is between 8 KB and 100 KB.

When DB2 OLAP Server stores the blocks, it carries out compression. Although these blocks are dense, relative to the other combinations of dimensions, data densities in these blocks are still low, often between 5% and 10%. Compression removes the empty cells, resulting in much less I/O and high storage efficiencies. The ratio of compressed block size to full block size is known as the *compression ratio*. Ratios between 0.4 and 0.8 are common.

8.4 Gathering information

In order to begin the estimating process, we must first know how many cubes we plan to build.

For each of the cubes, we must consider the number of dimensions and the number of members in each dimension. If we can determine the ratio of input members to total members in each dimension, then we can check the inflation factor.

Next, we decide which dimensions are dense and which are sparse. Using the member counts for the dense dimensions, we can check that our block size is within the acceptable range.

Now we must concern ourselves with the characteristics of the data. How sparse is the index, and what storage efficiency can we expect for the dense blocks? This is by far the hardest part of the process, and the resulting numbers have enormous impact on the estimates.

Because these factors are very difficult to estimate, and critical to the process, we strongly suggest that the values be determined by working with the data, either by building a prototype or by conducting a proof of concept (POC).

8.4.1 Proof of concept

Proof of concept is by far the most reliable method of gathering the information we need. Typically, this method will result in a good design; and estimates that are accurate to within 20%, even though we do not build the complete model.

Our goal is to build sufficient subsets of the model to allow us to understand:

- How large the DB2 OLAP Server blocks will be
- How effectively the blocks are compressed when we store them on disk, the compression ratio
- How many blocks the final model will contain, which is the percentage of maximum blocks existing, an indication of the sparsity of the DB2 OLAP Server index
- The time it takes to generate and write calculated data

In order to gather the key metrics, we need to understand the blocks and the index.

Blocks are defined by dense dimensions, and the index is defined by the sparse dimensions, so we should be able to investigate these two factors independently.

Figure 21 on page 131 illustrates how we can define slices through the cube.

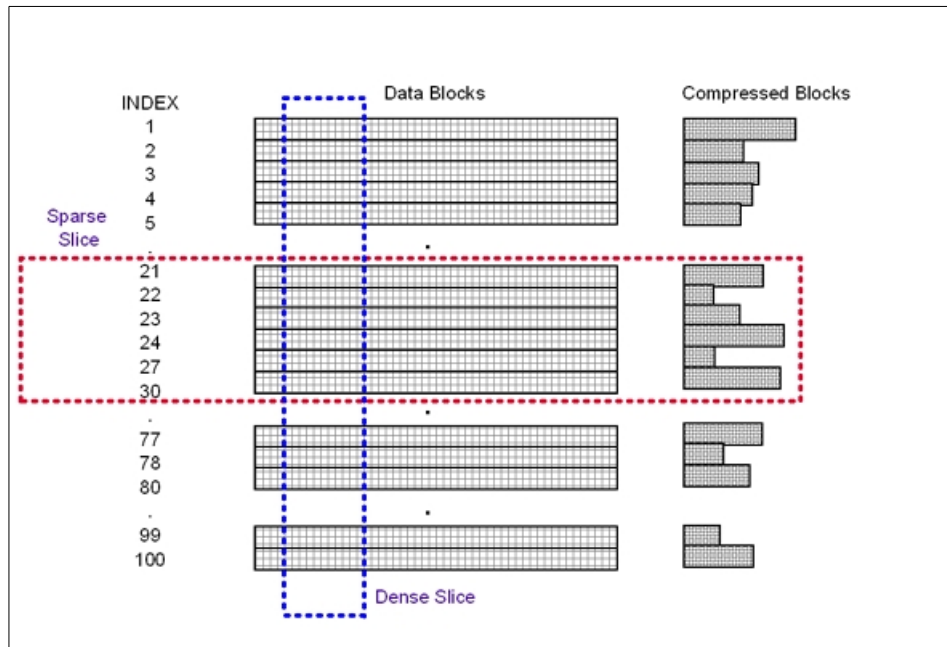


Figure 21. Way to define slices through the cube

The sparse slice cuts through the index, but complete blocks are processed and compressed. From the sparse slice we can determine the size of the data blocks, and the compression ratio.

The dense slice cuts through the data blocks, but the complete index is processed. From the dense slice we can determine the sparsity of the DB2 OLAP Server index.

We build two slices through the cube. In the first case, we subset the dense dimensions, and measure the percentage of maximum blocks existing, our ratio across the sparse dimensions. In the second case, we subset the sparse dimensions and measure the compression ratio across the dense dimensions.

In both cases, we note the size of the cube before and after the calculation, and the time taken to calculate the cube. From these figures, we can estimate the calculation rate for the hardware platform, and use this rate to estimate how long it will take to build the final cube.

We have built all dimensions, and improved our understanding of the model, but because we load and calculate subsets, we do not have to wait for the

entire model to build. This method gives us accurate metrics in a fraction of the time it would take to build the complete model, and it is far more reliable than sampling.

We applied this technique to the model illustrated in Table 7.

Table 7. Example proof of concept model

Dimension	Dimension type	Members	Stored Members
Time	Dense	86	32
Measures	Dense	15	6
Scenario	Dense	16	2
Channel	Sparse	6	5
Product	Sparse	154	132
Customer	Sparse	1001	1000

For our dense slice, we loaded and calculated one fiscal quarter, reducing the number of stored members in the time dimension from 32 to 4. When the calculation is complete, the percentage of maximum blocks existing can be found on the statistics tab of the database information dialog in Application Manager. With this subset, the percentage of maximum blocks was reported as 3.46%, the same as the full model.

Next, we built a sparse slice through the product dimension. We took one family of 5 products, reducing the number of stored members in the product dimension from 132 to 5. When the calculation is complete, the compression ratio can be found on the statistics tab of the database information dialog in Application Manager. When we loaded and calculated this subset, we got a compression ratio of 0.11. The full model has a compression ratio of 0.10.

For the hardware platform we used, we found the OLAP engine was generating 3 GB of calculated data per hour.

As long as you choose the dense and sparse slices carefully, this technique will yield excellent results. If you are dealing with data that is extremely seasonal, you may need to build subsets for off-season and peak periods, and factor the different metrics into your estimate.

8.5 Using the information

Now that we have gathered the required information, we can use it to generate estimates for CPU, memory, and disk storage, and to verify that we can build the cubes within the batch window. As an example, Figure 22 shows the characteristics of a sample cube to estimate system requirements.

Cube 1
Number of dimensions – 7
• Member counts
• Dimension 1 – Dense – 34 members
• Dimension 2 – Dense – 42 members
• Dimension 3 – Dense – 3 members
• Dimension 4 – Sparse – 12 members
• Dimension 5 – Sparse – 134 members
• Dimension 6 – Sparse – 167 members
• Dimension 7 – Sparse – 1277 members
Percentage of maximum blocks existing 0.2%
Compression Ratio 0.6
Calculation rate 3 GB per hour

Figure 22. Sample cube for hardware estimate

First, we calculate the number of blocks the cube will contain by multiplying together the member counts for the sparse dimensions, and then multiplying by the percent of maximum blocks existing:

$$(12 \times 134 \times 167 \times 1277) \times 0.2\% = 685,841 \text{ -- (1)}$$

Next, we calculate stored block size by multiplying together the member counts for the dense dimensions, multiplying by 8 bytes per value, and then applying the compression ratio:

$$((34 \times 42 \times 3) \times 8) \times 0.6 = 20.6 \text{ KB}$$

Multiplying these values, we get an estimate of disk storage requirements:

$$685,841 \times 20.6 = 14.1 \text{ GB disk storage}$$

We get maximum parallelism when each cube is allocated its own storage devices. We must take this into account when totaling the disk requirements. If we have four cubes, each estimated at 3 GB, we should plan for 4 devices of 4 GB each, rather than fewer, larger devices.

Today's DB2 OLAP Server engine architecture is multi-threaded to support many concurrent users, but cube calculation is largely single-threaded. This means that the calculation process for one cube will drive one CPU. Adding additional CPUs will not reduce the calculation time. To leverage SMP machines, we build a number of cubes in parallel, allocating one CPU to each cube.

Since we have a single cube, we require:

1 cube = 1 processor

Running on today's processors, with high-speed storage devices, we expect DB2 OLAP Server to generate between 1 GB/hour and 3 GB/hour of data. Calculation rates as high as 7 GB/hour have been recorded on an S80, the high end RS/6000, but we will base our initial estimate on the 3 GB/hour calculation rate observed when building the dense and sparse slices. We estimate the batch time required for the cube as:

$14.1 / 3 = 4.7$ hours batch time

We calculate cache memory for the index at (1) x 80 bytes per entry:

$685,841 \times 80 = 55$ MB

Allowing for index cache, three times this amount for block caching, and 2 MB overhead, we estimate memory for this cube at:

$(55 \times 4) + 2 = 222$ MB memory

Based on these estimates, and allowing for error accordingly, our machine specification for this cube will be:

- 1 processor
- 250 MB of memory
- 16 GB of disk storage
- A batch window of 5 hours

8.6 Summary

In order to estimate OLAP cube size, we need a good understanding of the model, the characteristics of the data, and knowledge of the way the data is

processed by the OLAP engine. Together, these factors influence inflation, sparsity and block density, which all contribute to final model size, and the time it will take to build the model.

Our estimating technique is based upon two slices through the complete cube. Both slices result in a cube which represents a subset of the final model. Because we control the slices to subset through the cube once through the sparse dimensions, and once through the dense dimensions, we understand which metrics to gather from which slice, and how to use them as the basis of a final estimate.

Using this technique, we can accurately estimate final model size and batch time without building the complete model.

Chapter 9. Sizing Intelligent Miner for Data for performance

The focus of this chapter is to enable you to create a sound methodology as a continual process for sizing and planning your Intelligent Miner (IM) application environment. This includes giving you the ability to measure the right data and correlate Intelligent Miner application transaction events to the measured data.

Three aspects need to be considered:

- CPU
- Memory
- DASD

9.1 Factors influencing Intelligent Miner performance

The following major factors influence the performance of the Intelligent Miner server.

9.1.1 Location of input data

Remote data forces the mining processes to be I/O-bound and increases dependence on network performance and locks. Local data removes network overhead.

9.1.2 Data volume and variables

The quality of the input data is directly correlated to the quality of the mining results. The number of attributes of the input data is directly proportional to the distortion of the data mining results and the performance of the mining process. It is also known that most mining techniques will scale linearly with the amount of data you use.

9.1.3 Mining techniques

Performance also depends on the mining techniques that you use, although it is not the real deciding factor. Business requirements usually constrain the choice of mining techniques. You should estimate the running time by first mining on a *sampling* of data.

9.1.4 Parallelism

We can discern two different kinds of parallelism associated with data mining: within the server, or within the mining technique.

9.1.4.1 Server parallelism

Server parallelism means availability of multiple processors that can run any of the tasks executed on the server. The operating system distributes tasks over processors, meaning that if you execute several mining runs in parallel, or a database engine together with the mining engine, you will automatically gain from the mining configuration.

9.1.4.2 Mining parallelism

Some mining techniques lend themselves to parallel execution on multiple servers or multiple subsystems within the same server. This may drastically reduce the run time, sometimes linearly with the number of processors. However, planning, preparation, and tuning of the overall architecture to achieve mining parallelism may require a considerable amount of time.

You should consider exploiting this feature only if you have large amounts of data available, or if the mining process itself is time constrained. For example, you might have to evaluate a large amount of customers each day, for which a batch window of only one hour is available. In this case, you will gain the most from the application of parallelism with repeatable applications because the relative effort of tuning is reduced.

9.1.5 Database design

Data mining techniques will generally execute table scans on single tables, so the main bottleneck is I/O throughput. Internal database parallelism will help deliver the data faster to the requesting mining technique. As a general rule, the better the database design, the faster the response of the mining algorithms.

9.2 Sample response for a selected system configuration

In this section we will use a demo S/390 server as a sample for a possible data mining configuration server. We will document the time elapsed for each mining run using several data samples and data mining techniques.

Since mining techniques scale linearly with the amount of input data, it has been found that mining on a sample of the customer's data will give a clear idea of the response time of the actual mining runs. It is also useful to compare the actual system configuration/resources to the following system configuration, in order to estimate the elapsed time for the mining runs on the system that you use.

The following measures were taken on a 9672-R63 (G2) processor, running in LPAR mode using 3 CPUs and 256 MB of storage.

9.2.1 Clustering

We used the sample data for clustering as input. The number of input columns used is 10 and the number of input rows is approximately 450.

As shown in Figure 23, using the default system settings we were able to cluster the mentioned input data into 10 clusters in 46 seconds.

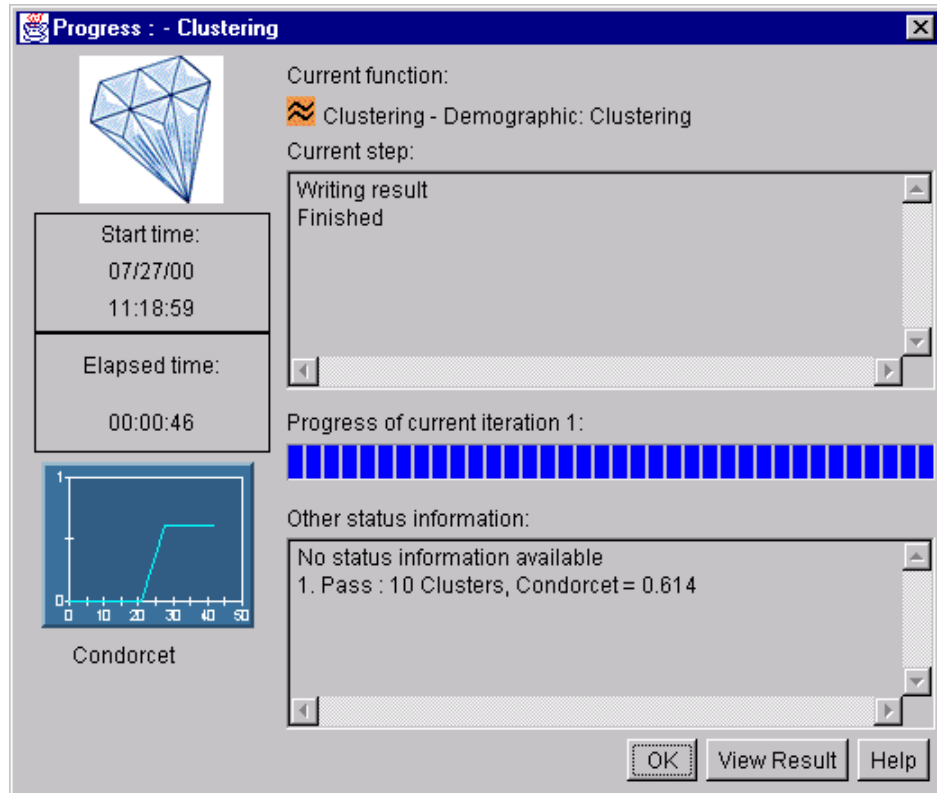


Figure 23. Result of clustering

9.2.2 Sequential patterns

We used the sample data for Sequential Patterns as input. The number of input columns used is 5 and the number of input rows is approximately 800.

The input columns are: Customer, Date, Item, Store and Transaction.

As shown in Figure 24, using the default system settings we were able to create the sequential patterns for the mentioned input data in 18 seconds.

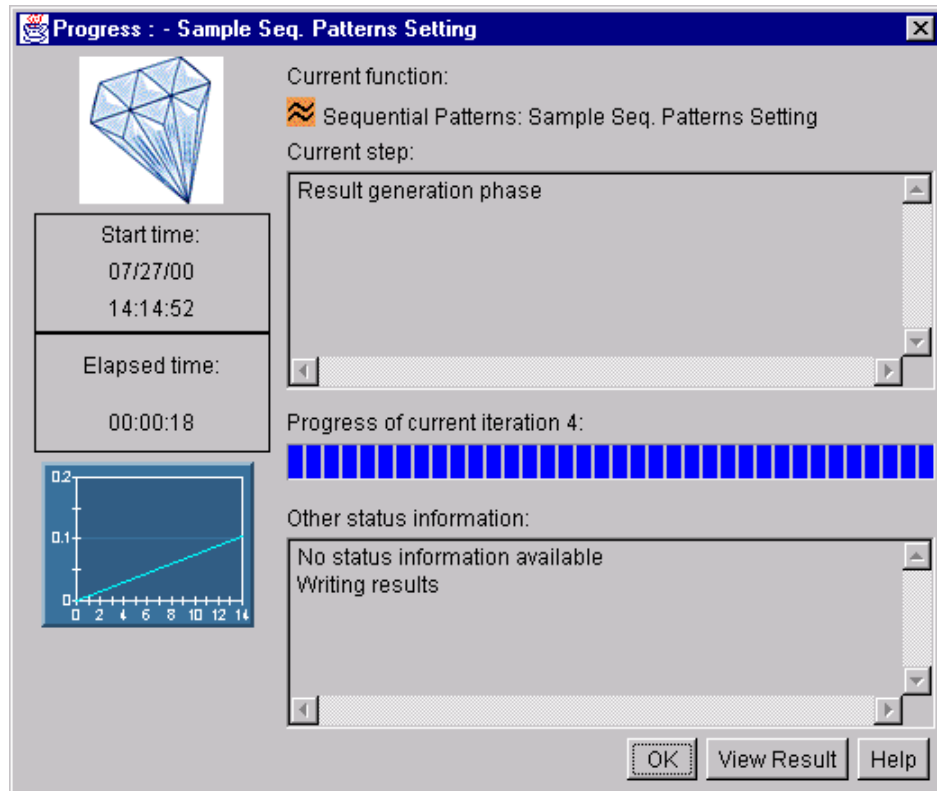


Figure 24. Result of sequential pattern

9.2.3 Associations

We used the sample retail store transaction data. The number of input rows used is approximately 1150. The input transaction data included Customer, Store and Date.

We used the settings shown in Figure 25 on page 141 for the Association run. The resulting rules that matched the requested support and confidence factors are shown in Figure 26 on page 142.

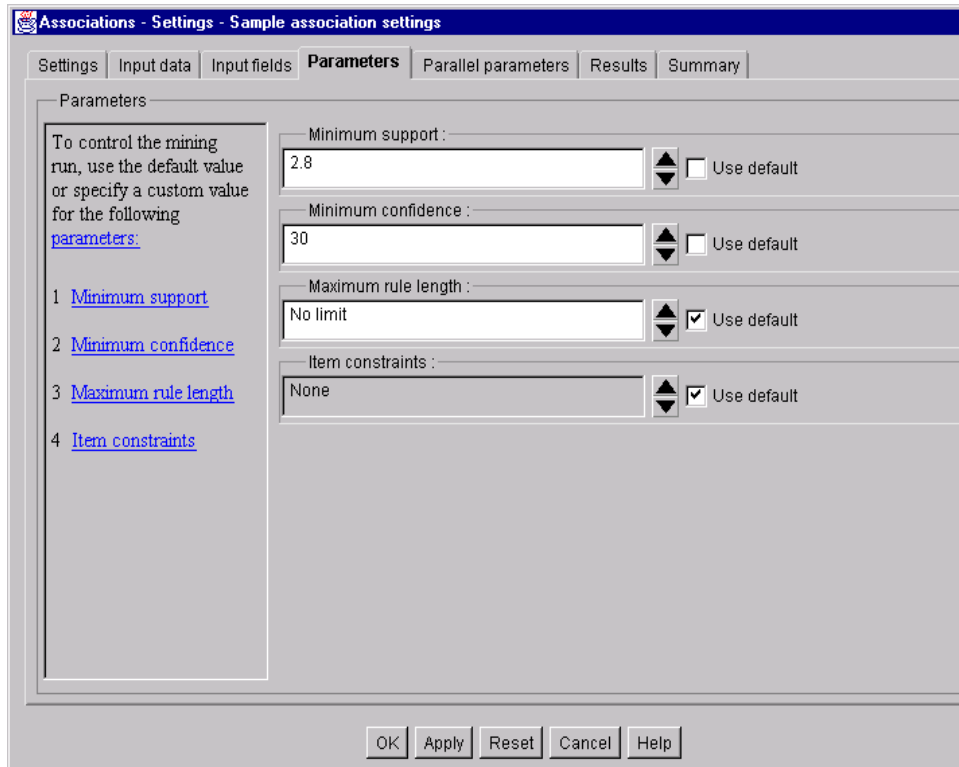


Figure 25. Settings for association

Figure 26 on page 142 shows part of the resulting rules. However, for this example 39 rules were produced in 22 seconds using the same S/390 server.

Support(%)	Confidence(%)	Type	Lift	Rule Body	Rule Head
2.9963	80.0000	+	21.3600	[Mineral water]+[Soap A]	=... [Antifreeze]
2.9963	100.0000	+	19.0700	[Mineral water]+[Antifreeze]	=... [Soap A]
2.9963	80.0000	+	15.2600	[Antifreeze]	=... [Soap A]
2.9963	57.1400	+	15.2600	[Soap A]	=... [Antifreeze]
2.9963	66.6700	+	12.7100	[Mineral water]+[Lemonade]	=... [Soap A]
3.3708	64.2900	+	10.7300	[Soap A]	=... [Toilet paper]
3.3708	56.2500	+	10.7300	[Toilet paper]	=... [Soap A]
2.9963	100.0000	+	8.6100	[Antifreeze]+[Soap A]	=... [Mineral water]
2.9963	100.0000	+	8.6100	[Lemonade]+[Soap A]	=... [Mineral water]
2.9963	80.0000	+	8.5400	[Gouda Cheese]	=... [Crackers]
2.9963	32.0000	+	8.5400	[Crackers]	=... [Gouda Cheese]
2.9963	53.3300	+	8.3800	[B-Beer]	=... [Crisps]
2.9963	47.0600	+	8.3800	[Crisps]	=... [B-Beer]
3.3708	60.0000	+	7.6300	[Orange juice]	=... [Brandy]
3.3708	42.8600	+	7.6300	[Brandy]	=... [Orange juice]
2.9963	80.0000	+	6.8900	[Antifreeze]	=... [Mineral water]
3.7453	71.4300	+	6.1500	[Soap A]	=... [Mineral water]
3.7453	32.2600	+	6.1500	[Mineral water]	=... [Soap A]
3.7453	66.6700	+	5.7400	[Apple juice]	=... [Mineral water]
3.7453	32.2600	+	5.7400	[Mineral water]	=... [Apple juice]
3.7453	100.0000	+	5.4500	[Detergent]	=... [Lemonade]
2.9963	61.5400	+	5.3000	[Colour slide film]	=... [Mineral water]
2.9963	80.0000	+	4.3600	[Mineral water]+[Soap A]	=... [Lemonade]
2.9963	80.0000	+	4.3600	[Antifreeze]	=... [Lemonade]
2.9963	33.3300	+	3.5600	[178]	=... [Disp. nappies P]
2.9963	32.0000	+	3.5600	[Disp. nappies P]	=... [178]
3.3708	60.0000	+	3.2700	[Apple juice]	=... [Lemonade]
2.9963	57.1400	+	3.1100	[Soap A]	=... [Lemonade]

Rules (t/r/u/s): 39/0/0/39

Figure 26. Rules detail list

9.2.4 Statistical Analyses: Linear Regression

In the following example we used the Linear Regression function as an example of the Statistical Analysis functions of the Intelligent Miner for Data.

We used the sample car models database. The number of input rows (cars, in this example) used is approximately 60, and there are 26 input columns, including all attributes of the input car models.

Figure 27 shows the output value fitting curve of the Horsepower attribute. All curves are produced at the same time. The user switches between curves by using the drop-down list shown.

This system configuration needed 20 seconds to produce this output for the input data. The confidence level used is 0.95.

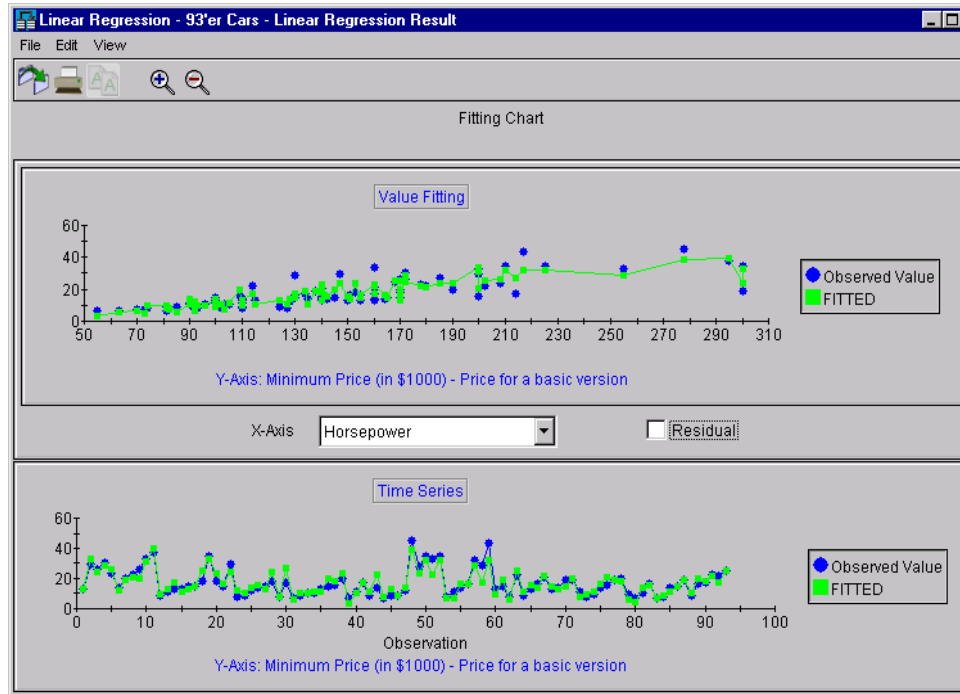


Figure 27. Result of linear regression

9.3 Intelligent Miner performance recommendations

In previous sections we discussed several factors that influence the performance of IBM's Intelligent Miner server. In this section we offer recommendations for improving the response time of various mining algorithms:

- We recommend that the location of mining input data be *local to the mining server* to prevent the mining process from becoming I/O-bound. If this is not feasible, the data can be stored in a lookaside buffer or local cache on the mining server, so that only the first access will be I/O-bound. Each subsequent data access then occurs locally on the server. This

requires sufficient local storage (up to the amount of the original data) and will increase performance for mining techniques that perform multiple passes over input data.

- Some experts advocate a sampling strategy that extracts a reliable, statistically representative sample from the full detail data. Mining a representative sampling instead of the whole volume drastically reduces the processing time required to get crucial business information, in addition to minimizing the resources such as the required disk space. The result, of course, is a less costly process that yields faster results of equal accuracy. On the other hand, details in data such as local effects can be lost due to sampling. It is now possible to execute mining techniques against amounts of data in the terabytes range on a multiple node mainframes.

Sampling can be used as a preparatory step to find the main effects, while the actual mining run uses the maximum amount of data available.

Local effects in your data might increase the model's complexity, or cause certain limits to be reached, such as the amount of data that fits in local memory. In such cases, the actual mining run time can be much higher than you expect. A good rule of thumb is to calculate the amount of data in two or more consecutive runs with different sample sizes, and then add about 25% extra time, to stay on the safe side.

- Try to avoid joining tables during the mining run, unless one of the tables is very small.
- A database should be well indexed in order to achieve optimum performance. Some of the mining techniques will benefit from indexing, for example, Associations Discovery and Sequential Patterns. Both will automatically create an "order by" clause within the SQL statement, and as a consequence, an (ascending) index on the appropriate columns will help speed up the run.
- IM for Data V6.1 is a client/server application that depends on the correct performance of its prerequisite products:
 - OS/390 UNIX System Services for program execution
 - IBM Language Environment for MVS for program execution
 - TCP/IP for client/server communication
 - DB2 for database support

All these products must be correctly installed and configured, and must be at the latest service level available at installation time of IM for Data V6.1.

9.4 Data considerations for data mining

There are no special considerations for maintaining your data mining source and data, other than those considerations applicable to any other database or data warehouse. Just keep in mind that, since you will make decisions based on information derived from this data, it is very important that your data represents the actual status in the real world.

Avoid data inflation

However, it is very important to monitor the mining data against inflation. Consider an example of a sales database: the sales database usually contains a “Date” column. However, depending on the type of analysis, you may need to consider adding a “Quarter” column in order to be able to analyze sales by quarter or to perform clustering based on the Quarter dimension. As a result, this could cause *data inflation*, which is a significant increase in the size of input data.

Output data considerations

Unlike many other server technologies, where the focus is on the number of users accessing the server, the Intelligent Miner server focuses on the *location* and *complexity* of input and output data. In particular, output data (or result data) is an important factor to consider when you design for data mining.

Kinds of output data

There are two kinds of output data:

- The model generated by the data mining process
- Data resulting from applying the model to the data that is available

Though the location of the output data is determined by the business requirement or the group of the business people requiring access to the mining results, it is extremely important to release any bottlenecks in writing output data. A rule of thumb is to follow the general database performance rules (if using a database) or system performance recommendations (if using flat files).

Appendix A. Sample worksheets

This appendix contains six sample worksheets that can be used for the capacity planning for BI applications.

The worksheets included, along with references to the chapters where they are discussed, are as follows:

1. Worksheet 1: Application profile; see 5.3 on page 70.
2. Worksheet 2: System characterization; see 5.4 on page 71.
3. Worksheet 3: Workload characterization; see 5.5 on page 73.
4. Worksheet 4: Maintenance requirements; see 5.6 on page 82.
5. Worksheet 5: Growth projection; see Chapter 6, "Growing existing workloads" on page 87.
6. Worksheet 6: New application, see Chapter 7, "Sizing methodologies for new applications" on page 103.

Worksheet 1: Application profile

System Profile:

Workload ID: _____ System ID: _____

Time Period of Analysis: _____

Application Profile:

What departments in the company are the top 3 primary users of this application?

For each department, complete the following information:

End User Information:

How many users have a log on to this system? _____

How many users are actively querying the system on a normal day: _____

What are the user IDs or identifying characteristics of these users? _____

Are generic IDs used? _____

What is the naming convention for IDs? _____

How are the users accessing the data:

End User product being used: _____

Is DB2 Connect installed on the end-user workstations? _____

Who installed the End User software on the systems? _____

If so, has the Set Client Info API been used? _____

What are the most important user IDs accessing the system? _____

How are they physically connected to the system? _____

Application Server Information:

Is there a logical Application Server function in this implementation? _____

What system is the Application Server installed on? _____

If S/390: What is the name of this application server code? _____

Who installed it? _____

What is the STC name for this product? _____

Is it in a unique performance Group /Reporting Group:

Service Class (Perf Group): _____ Reporting Group: _____

Critical Processing Windows for Business Unit:

When are the key processing times for this data? _____

Week of Year: _____ Time period: _____ Concurrent users: _____

Day of week: _____ Time period: _____ Concurrent users: _____

Day of month: _____ Time period: _____ Concurrent users: _____

What is the most important information derived from this system? _____

When is this information needed? _____

Worksheet 2: System characterization

System profile:

Workload ID: _____ System ID: _____

Time period of Analysis: _____

Processor Model: _____ #CPs: _____ Memory: _____ Channels: _____

System utilization:

CPU Workload:

This Table reflects either the QUERY or MAINTENANCE workload: _____

Is this a repeatable pattern of high utilization? _____

What is the peak day of the week? _____ Peak hour of the day? _____

Throughout the year, determine the Peak week for CPU Utilization: _____

Table 2-1: Peak CPU utilization profile

Week of year: _____ Query/Maintenance: _____

Time	Monday	Tuesday	Wednes.	Thursday	Friday	Saturday	Sunday
00:00							
01:00							
02:00							
03:00							
04:00							
05:00							
06:00							
07:00							
08:00							
09:00							
10:00							
11:00							
12:00							
13:00							

Time	Monday	Tuesday	Wednes.	Thursday	Friday	Saturday	Sunday
14:00							
15:00							
16:00							
17:00							
18:00							
19:00							
20:00							
21:00							
22:00							
23:00							

Daily Profile: Peak day workload

Peak day CPU utilization profile:

Day of week: _____ Date: _____

Query or maintenance workload: _____

Table 2-2: Peak day CPU utilization profile

Day	00:00-00:15	00:15-00:30	00:30-00:45	00:45-00:59	AVG for Hour
00:00					
01:00					
02:00					
03:00					
04:00					
05:00					
06:00					
07:00					
08:00					
09:00					
10:00					
11:00					
12:00					
13:00					
14:00					
15:00					
16:00					
17:00					
18:00					
19:00					
20:00					
21:00					
22:00					
23:00					

Worksheet 3: Workload characterization

Date of analysis: _____

Complete a copy of this sheet for each of the key work periods analyzed in Worksheet 2.

Workload ID: _____ System ID: _____

Data profile:

GB of raw data: _____ GB of used DASD: _____

User profile:

Total user population: _____

Query profile:

Table 3-1: Query segmentation table

Query type	Time limit	Max CPU (sec.)	Min CPU (sec.)	Avg CPU (sec.)	% of query workload CPU usage	% of total CPU
Trivial	< 5 sec.					
Small	< 5 min.					
Medium	< 30 min.					
Large	< 1 hour					
Xlarge	> 1 hour					

% query workload: _____

% of total system workload: _____

Additional information:

The following tables gather additional information that assists in understanding the workload. While the capacity planning tasks can be done without this additional information, it may be helpful to know the more detailed characteristics of the workload.

Table 3-2: Disk usage table

	Raw data	Index	Summary tables	DB2 work area	Batch work area	System	Reserve/Other
GBs							

Table 3-3: Disk subsystems physically installed

Disk Subsystem	Number	Number of channels	Usable disk space (GB)

Query profile:

Table 3-4: Query average/peak table

	Avg. no. /day	Avg. concurrent	Peak no./day	Peak concurrent
No. of queries				

Table 3-5: Detailed query segmentation

Category	Avg Total/day	% of daily total	Avg ET (sec)	Min ET (sec)	Max ET (sec)	Avg CPU (sec)	Min CPU (sec)	Max CPU (sec)	Max Get-page	Min Get-page	Total Get-page
Trivial < 5 s											
Small < 5 m											
Med. <30m											
Large < 1 h											
Xlarge >1 h											
Totals	0	100%									

Worksheet 4: Maintenance requirements

Date of analysis: _____

The following analysis will be done for the peak days/weeks of maintenance processing identified in Worksheet 2.

Table 4-1: Operational profile

	Hours/day	Hours/week
Operational		
Maintenance		
Updates		

Where:

Operational: Available for query processing

Maintenance: Available for backups, archives, reorgs, and so forth

Updates: Available for adding or updating data into the database

Table 4-2: Critical path for batch jobs

Job name	Start time	Stop time	Elapsed time	CPU Seconds
Stream				

Worksheet 5: Growth projection

Date of analysis: _____

System Profile:

Workload ID: _____ System ID: _____

Data Growth:

1. Is the historical data currently stored growing? (Are you increasing the number of years of data stored?) _____

Will there be a corresponding growth in the number of queries? _____

Will the existing queries access more of the historical data, changing the query profile? _____

2. Is data growing due to business acquisition or new application being added to data store?

What is the initial amount of data being added (date)? _____

What is ongoing growth of this data (for how long) _____

What is the nightly update volume for this data? _____

3. Massive update of the source data.

What is the amount of source data to be updated and when? _____

How often will a massive update occur? _____

4. Calculate the data growth rate for this group of users:

(GBs being added/GBs currently installed=xx %)

User Growth:

1. User growth:

A. Is the entire population of users increasing by a factor? _____

B. Is a subset (business unit) of users increasing by a factor? _____

Identify the name of the subset: _____

C. Is a subset of users using a particular tool to access data increasing by a factor? _____

Identify the name of the subset: _____

2. Is this factor to be applied to the entire population or group? Yes/No _____

If no, do you have a breakdown of users by:

Operational users: will grow by _____

Analytical users: will grow by _____

Mining type users: will grow by _____

3. Determine the capture ratio for the workload:

Avg. CPU Utilization _____% No. Engines _____

Interval Time _____

Total APPL% for all Query Workloads _____

CR= _____

(details to calculate Capture Ratio (CR) in Appendix D, "Capture ratios" on page 171).

4. Complete the following table for each subset of the population that is growing:

Table 5-1: Data growth

Query type	Growth impact	% Data growth	Data growth factor

Table 5-2: Overall capacity projection

(Note: % total CPU column values should be adjusted with capture ratio)

Query type	% of total CPU(CR applied)	User Growth factor	New CPU reqmt (user growth)	Data Growth factor	% Current CPU resource	Additional Capacity
Trivial						
Small						
Medium						
Large						
X-large						

Total: _____

Batch processing:

Refer to worksheet 4 for batch processing data:

Total Elapsed time for Job stream x Data growth being planned for = Time for batch processing

Worksheet 6: New application

Date of analysis: _____

System Profile:

Workload ID: _____ System ID: _____

Time of period of analysis: _____

Complete a copy of this sheet for each new BI workload

Query workload:

Table 6-1: Custom query profile

Query type	Data scanned	Elapsed time	CPU time	% Query workload
Trivial				
Small				
Medium				
Large				
Xlarge				
Total	N/A	N/A	N/A	100%

User population: _____ Active queries: _____ DASD of raw data: _____

Table 6-2: Query sizing projection

Iteration: _____

Projected user population: _____ Projected raw data: _____

Projected active peak queries: _____ Projected DASD: _____

Query type	Data scanned	Elapsed time	CPU time	Projected % of new workload	Projected CPU
Trivial					
Small					
Medium					
Large					
Xlarge					
Total	N/A	N/A	N/A	100%	

Batch workload:

A. Initial load:

Number of months or years of data for initial load: _____

Input records/month: _____ Amount (GBs) of raw data/month: _____

Table 6-3: Maintenance processing

Batch ROT Performance Estimates							
RoTs (ms/SQL call)			1	2	20		
Critical Path Program/Plan	# Input Records	# SQL Calls / Input Record	Estimated CPU Minutes	Estimated I/O Minutes Minimum	Estimated I/O Minutes Maximum	Elapsed time Minutes Minimum	Elapsed time Minutes Maximum
Extract 1							
Extract 2							
:							
Transform 1							
Transform 2							
:							
Add/Update 1							
Insert							
:							
Total Time							
Batch Window (hours)							
Batch Safety Factor							
Parallel jobstreams needed							
Engines needed							
Utilities - DB2 Estimator preferred							
ROTs	LOAD	250	MB/minute, one index, inline stats				
	# Input Records	DB2 Rowlength	Elapsed Time Minutes				
LOAD 1							
COPY 1							

Appendix B. Data models

Data modeling is an integral part of every data warehouse design. Here is a brief description of the most common types of data models.

B.1 Overview

In relational database management, *normalization* is a process which breaks down data into record groups for efficient processing. There are six stages of normalization. The third stage, third normal form (3NF), is usually found to be sufficient. In 3NF, data is identified only by a key field, and every column of the table is only dependent on the key. Each column is independent of every other column except the key. The advantage of 3NF is reduced data redundancy and therefore less chance of inconsistencies of data elements across different tables. *Denormalization* avoids joins of very large tables by repeating certain pieces of data in several tables. SQL would involve joins of fewer tables, generally requiring fewer data pages to be accessed and thus reducing CPU resources.

There are basically three types of data models discussed in this appendix: entity-relationship, star schema, and snowflake models. Figure 28 on page 163 illustrates these models.

B.2 Entity-relationship (ER)

An entity-relationship logical design is data-centric in nature. That is to say, the database design reflects the nature of the data to be stored in the database, as opposed to reflecting the anticipated usage of that data. Because an entity-relational design is not usage-specific, it can be used for a variety of application types: OLTP and batch, as well as business intelligence. This same usage flexibility makes an entity-relational design appropriate for a data warehouse that must support a wide range of query types and business objectives.

B.3 Star schema

The star schema logical design, unlike the entity-relational model, is specifically geared towards decision support applications. The design is intended to support very efficient access to information in support of a predefined set of business requirements. A star schema is generally not suitable for general-purpose query applications.

A star schema consists of a central *fact* table surrounded by *dimension* tables, and is frequently referred to as a multidimensional model. A fact table holds the numerical measures and key figures of a business subject area. The fact table relates these measures to a number of dimension tables which contain the information to establish a context for the recorded facts. Figure 28 shows examples of a fact table and dimension tables. Although the original concept was to have up to five dimensions as a star has five points, many stars today have more or fewer than five dimensions.

The information in the star usually meets the following guidelines:

- A fact table contains numerical elements.
- The fact table also usually has a small number of columns but a large number of rows, sometimes in the billions.
- The primary key of each dimension table is a foreign key of the fact table.
- A dimension table contains textual elements.
- A column in one dimension table should not appear in any other dimension table.
- The dimension tables tend to have a small number of rows and columns.

Some tools and packaged application software products assume the use of a star schema database design.

B.4 Snowflake schema

The snowflake model is a further normalized version of the star schema. When a dimension table contains data that is not always necessary for queries, too much data may be picked up each time a dimension table is accessed. To eliminate access to this data, it is kept in a separate table off the dimension, thereby making the star resemble a snowflake.

The key advantage of a snowflake design is improved query performance, because less data is retrieved and joins involve smaller, normalized tables rather than larger, denormalized tables. The snowflake schema also increases flexibility because of normalization, and can possibly lower the granularity of the dimensions.

The disadvantage of a snowflake design is that it increases both the number of tables a user must deal with and the complexities of some queries. For this reason, many experts suggest refraining from using the snowflake schema. Having entity attributes in multiple tables, the same amount of information is available whether a single table or multiple tables are used.

BI Logical Data Models

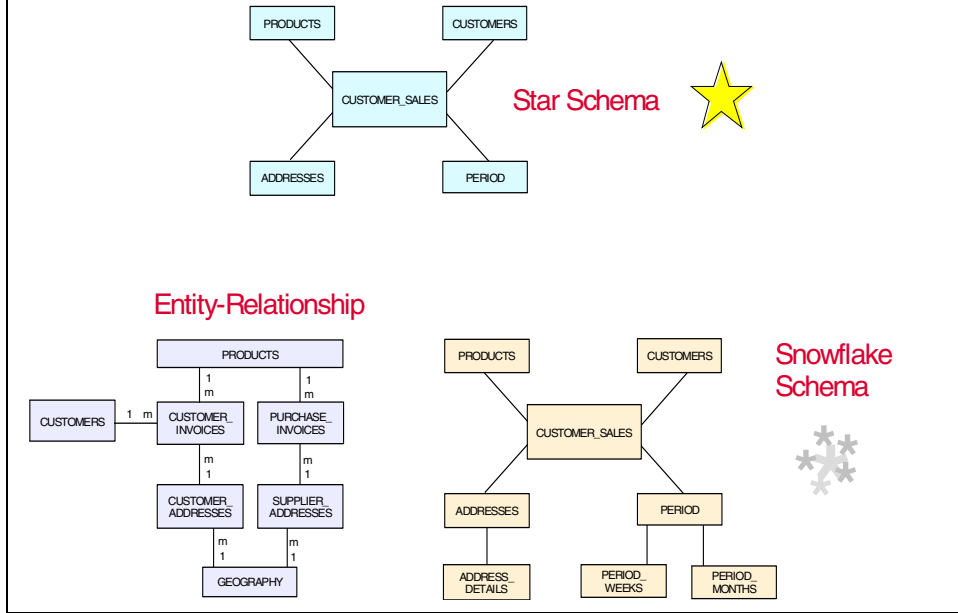


Figure 28. Data models

Appendix C. Data Propagator Relational (DB2 DPropR)

DPropR is the central product of IBM's replication solution. It enables you to create and run powerful homogeneous replication systems. This means that you can propagate any DB2 database located on any platform to any other DB2 database.

In this appendix we provide an overview of DPropR v5 and a methodology for capacity planning when you extend your replication system. For detailed information about this product and the complete IBM data replication solution, refer to the following documentation:

- DPROPR Planning and Design Guide
- DB2 Replication Guide and Reference

C.1 Architecture and components

DPROPR consists of three components: Capture, Apply, and Administration.

The *Capture* component reads the source database log activity, recognizes database updates of interest as specified by the user, and stores information related to these updates in local DB2 tables. Each insert/update/delete to a source table is stored as an insert to a *changed data* (CD) table. There is one changed data table for each source table. DB2 commit points for units of works are stored in a global table called a *unit of work* (UOW) table.

Another important process done by the Capture is the *pruning* process. During pruning, Capture will delete the data no longer needed in the CD and in the UOW tables.

The *Apply* components fetch the data stored by Capture in the changed data tables, store the fetched rows into one or more *spill files*, and finally make the changes to the copy tables in the data warehouse environment. There are two basic configurations:

- Pull design: The Apply program runs at the target server.
- Push design: The Apply program runs at the source server.

The *Administration* component is used to define sources and targets for a replication scenario, to set the schedule for updating the targets, and to specify the enhancements to the target data.

The three components operate independently and asynchronously to minimize the impact of replication in the operational environment. The only

interface between the different components is a set of relational tables, the DPropR *control tables*. The administration component feeds these control tables when you define the replication sources and the replication targets. The Capture and Apply programs use the information of the control tables to capture updated data and then replicate it into the target tables, in a proper format, and at the appropriate interval.

The three components of DPropR are shown in Figure 29.

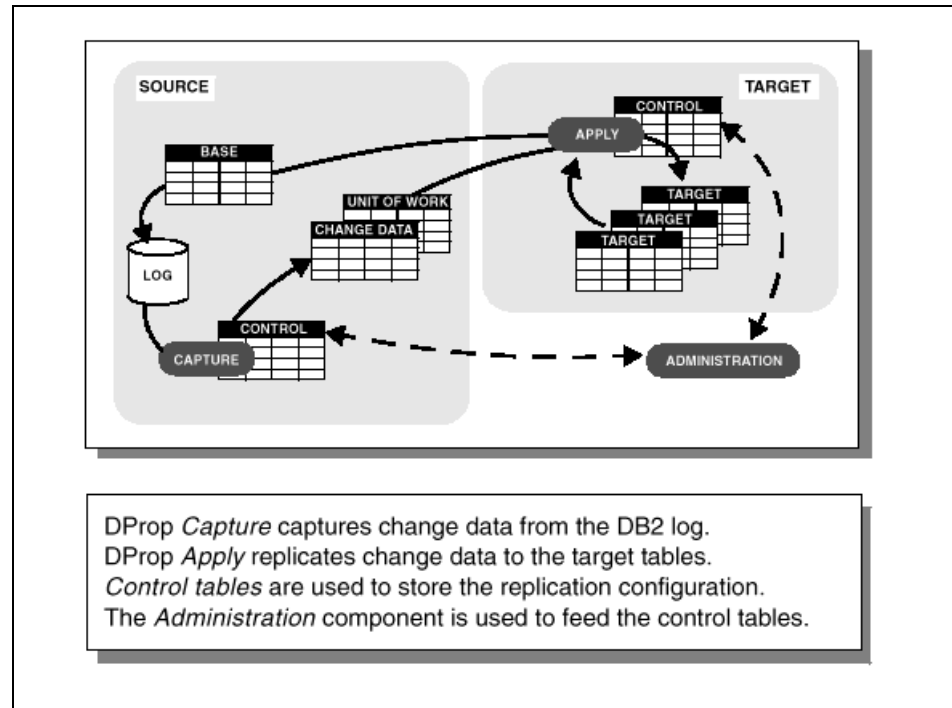


Figure 29. DPropR components

C.2 Storage requirements

DB2 DPropR needs the following storage requirements.

C.2.1 Log impact

DB2 source tables selected for replication *must have* the DATA CAPTURE CHANGES attribute. With this attribute, the before/after value of all the columns are logged. This is why the source log space increases.

As a rule of thumb, you can estimate that the log space needed for the replication source tables after you have set the `DATA CAPTURE CHANGES` attribute will be three times larger than the original log space needed for these tables. Therefore, each time you add a new table to be replicated, you have to consider the impact on the log size.

You also need to consider the increase in log space needed because the staging tables and the UOW tables are DB2 tables, and so they are also logged.

C.2.2 New tables: CD tables and UOW tables

This storage is based on the source updating applications and the replication requirements.

Space requirements for the CD tables are highly dependent upon when you prune the data. If you do not change your pruning schedule, the CD table growth will only depend on the following:

- Growth in the number of updates/inserts/deletes in the source applications as a result of business growth
- The addition of new columns to be replicated
- The addition of new tables
- The addition of new applications to be replicated to your target system

If you change your pruning schedule by holding the data on CD tables for a longer length of time, you will need to re-estimate the storage needed for CD tables.

Space requirements for UOW tables grow with the number of commits issued on transactions that update tables subject to replication.

C.2.3 Target tables

As a capacity planner, and once you have your replication system running, you must do capacity planning for the target tables in the same way for every DB2 database. For more detailed information about capacity planning in DB2 applications, refer to *DB2 for OS/390 Capacity Planning*.

In a replication system, you also have to consider:

- The growth of the source table of each target table: how are your target tables growing as a result of the source tables growth?

- The length of each row: how will your target tables be affected when you add new columns to your replication system?
- The addition of new tables.
- The addition of new applications to be replicated to your target system.

C.2.4 Apply spill file

The Apply component uses work files called spill files when it fetches data from either the source tables or the staging tables. As a general rule, the spill file increases only as a result of growth in the source data. You have to consider an increase of the spill file in the following situations.

- There many updates to replicate, caused by:
 - Unexpected failures in the replication architecture, so you have to do the propagations of several days at the same time.
 - Massive updating of data in the source tables. For example, in the source environment all the rows in the customer table get changed, because of having a new type of customer.
- The addition of new tables.
- The addition of new applications to be replicated to your target system.

C.3 Capacity requirements

DB2 DPropR needs the following capacity requirements.

C.3.1 Capture

Capture is run as a single task per DB2 subsystem or data sharing group.

The CPU requirements for the Capture programs are generally low. It does not impact the updating application of the operational environment and requires a minimum of CPU capacity. Capture for MVS is usually scheduled at a lower priority than the source applications. The Capture workload increases as a result of increasing the inserts/updates/deletes in the operational environment. You can extrapolate this at the same time you do capacity planning for the operational system.

To obtain the CPU cycle for the Capture workload you can use RMF reports and DB2PM.

C.3.2 Apply

The Apply program CPU requirements can vary greatly. The main factors that affect CPU are the currency and the future requirements of the target system; the more frequent the propagation, the higher the overhead per propagated row and the greater the CPU consumption.

Apply requires CPU on both the source and the target systems, so you must consider both workloads when you define the profile of your system.

Apply is a single task, as Capture is, but it is possible to run multiple Apply task at the same time, thereby increasing the performance of the propagation.

C.3.3 Administration

In general, the impact of the Administration component is not significant in terms of the local CPU.

C.3.4 Network capacity

The apply task will require network capacity if the target sever and the source server are not the same database or subsystem. In general, the capacity required depends on the volume of data to be applied, the timing window available in which to apply the data, the desired currency of the target data and the bandwidth installed. For more efficient use of the network capacity, Apply is usually installed at the target server so that it can pull the data from the data server. You have to consider if you will have enough timing window for the currency data that you need and your current bandwidth, when, as a result of the data growth, the volume of the data to be applied increase.

C.3.5 Throughput capacity

The following factors influence the throughput possible with DPropR:

- Network bandwidth and tuning.
- Source and target machine CPU capacity availability.
- Database tuning and contention factors.
- Change volume and workload mix.
- Frequency of Apply program.
- Number of Apply programs used.
- Apply program pull versus push design.

C.4 DPropR workload

DPropR covers the ETL process in the data warehouse environment. We consider DPropR as one of the maintenance workload that we have identified in Chapter 3, “Data maintenance” on page 37.

You need to identify your Capture workload and Apply workload running in the source system, and your Apply workload running in the target system (we assume a pull design). Both of them can be analyzed using RMF or DB2PM reports. Use the worksheet 4: Maintenance Requirements for identify the profile workload of the DPropR components, as it is explained in Chapter 4, “Characterization of existing workloads” on page 51 and Chapter 5, “Establishing current system usage” on page 65. You have to do it the same way as for the rest of the maintenance workload.

Once you have your current DPropR workload, you need to estimate the future workload as a result of the data growth. In the previous section of this appendix we have explained which are the effects of the data growth in the DPropR components. With the data growth estimation you need to use the Worksheet 5: Growth Projections, as it is explained in Chapter 6, “Growing existing workloads” on page 87.

We consider DPropR components as part of the maintenance workload, so you analyze his profile and growth the same way that you do for all the maintenance workloads.

Appendix D. Capture ratios

To effectively plan for the capacity of a workload, you need to determine the total processing requirements for the workload. CPU consumption is reported at a number of levels in various reports available through the Resource Measurement Facility of OS/390 (RMF). Each report for CPU activity has a slightly different view of the system, and therefore reports a slightly different value. One source is the RMF Workload Activity Report, where the CPU resource is reported for specific workloads in the system and does not include the processor time used in “service” address spaces such as JES or VTAM. It does not include system address spaces such as the Master address space or the many monitors and other services that exist in the system.

Another source of the CPU consumption of a workload is at the operating system level, for all the work consumed within that image. That is reported in the CPU Activity Report for the system, both in basic mode and LPAR mode. Additionally, there is some amount of CPU activity in this report, which is uncaptured in the workload reports.

A capture ratio attempts to equitably adjust the time directly attributed to the business unit work (performance groups within an RMF Workload Activity Report), to include uncaptured system time. With the capture ratio, you can then determine the expected utilization of a workload at the system level. This is the hardware capacity that must be installed to handle the work. For a more detailed discussion on capture ratios and performance groups, refer to *RMF Performance Management Guide*.

To calculate the capture ratio, you need to gather system utilizations from a number of system reports. In the S/390 environment, the RMF reports information related to system usage. There are a number of levels at which system resource utilization is reported by RMF.

The operating system utilization provides the total CPU resources consumed by the system image. This data is reported in the RMF CPU Activity Report. It is listed as the percentage of utilization of all CPUs allocated to that system image. The utilization you use for capacity planning depends on whether the system is executing in a logical partition (LPAR) or not.

When the system is *not* in an LPAR, the utilization is listed in the MVS BUSY TIME PERC column on the TOTAL/AVERAGE line. This is the percentage of the interval time that the processors were busy executing instructions. It includes the total CPU time in the Workload report, and the non-captured

CPU time above it. This was changed a few years ago from a model that reported this value in Wait Time. The DB2 Capacity Planning redbook refers to this older definition of the field.

When in LPAR mode, the LPAR Busy Time Perc field in the CPU Activity Report provides the percentage of the actual CPU, not based on the amount of CPU allocated via the weighting factor. This would be the best value to use for calculating the Capture Ratio, as it is based on the same CPU consumption as the other reports that will be used for this calculation.

Another report to consult when determining the CPU consumption of an LPAR is the Partition Data Report. This report takes into account the LPAR Management processing requirements when reporting the capacity of a workload. The last column on the right of the report, titled Average Physical Processor Utilization- Total, lists all the processing cycles consumed by this workload, expressed as a percentage of the total capacity of the installed system. This value would indicate all the processing cycles required to handle the workload, and should be compared to the above report. These numbers should be similar.

In addition to the overall system utilization reported at the CPU/LPAR level, the application utilization is reported by performance groups in the RMF Workload Activity Report. For a discussion on performance groups, or Service Classes as they are now called, refer to *RMF Performance Management Guide*. Once the work is grouped into performance groups/service classes, the CPU cycles directly consumed by a specific service class of work is reported in the RMF Workload Activity Report in the SERVICE RATES column on the APPL% row for each workload. This is the percentage utilization of a single CPU and could therefore exceed 100% if the system has more than one engine. This value tells you how much CPU time was captured for the workload in a given performance group or domain. This information is used to calculate capture ratios. The workload activity report sums the CPU utilization for all reported workloads, and reports this in the PGN=ALL. All these various reports are used when calculating the capture ratio.

The system RMF Workload Activity reports will capture most (often 90% in BI environments) of the total CPU time used to run the workload. The ratio between the captured time for all the workloads as reported in the RMF Workload Activity Report PGN=ALL, to the total CPU time consumed as reported in the CPU Activity Report (MVS Busy field for basic mode, LPAR Busy for LPAR mode), is the capture ratio.

A number of capacity planning books address different ways to distribute the uncaptured times of a system. See *RMF Report Analysis* for more information. In many cases, customers run their business intelligence workloads in dedicated system images, rather than mixed with production OLTP workloads. Since the query workload uses all the same system resources during a measuring period, it is valid to use the method outlined in section 1.6.4.1 of *RMF Report Analysis*, and distribute all the uncaptured CPU time as well as all the system services (USS, JES, VTAM, DB2 address spaces) across all the query work, proportionally. This is not a case where the workload in a system is diverse and uses distinctly different system services, such as when CICS, batch, and TSO are mixed into a single system image.

The system reporting function can be set up to report different groups of queries in different performance groups in the RMF Workload Activity Report. To determine the capture ratio for this homogeneous workload environment, only those performance groups that represent query workload are used in the calculations in this discussion.

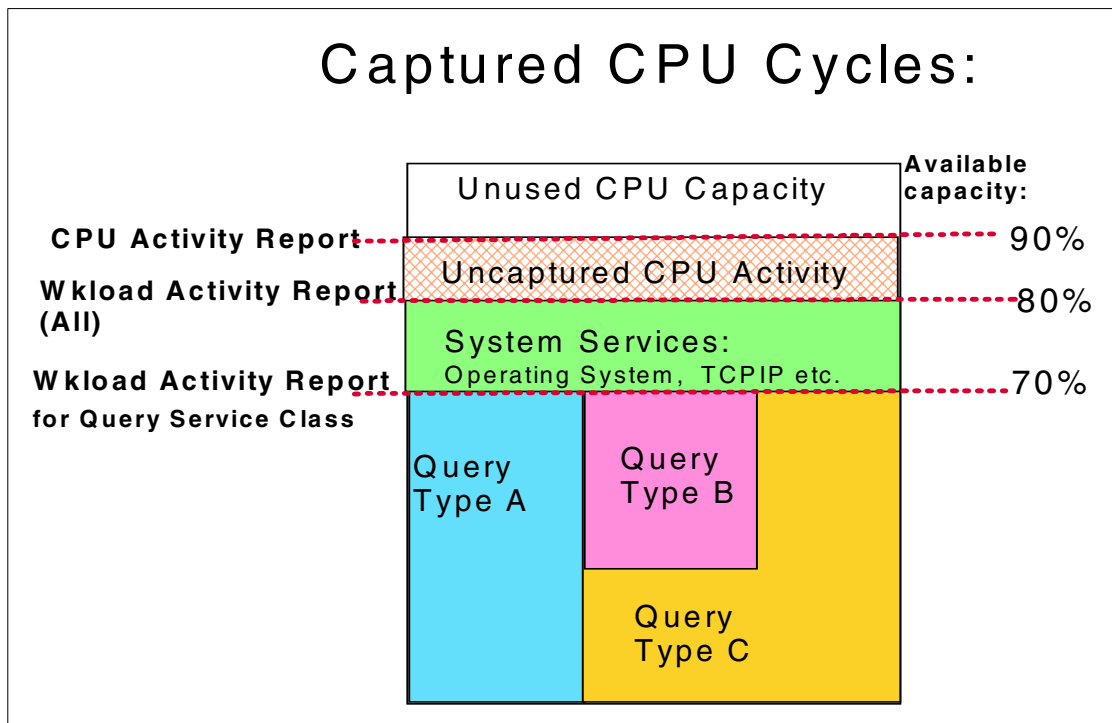


Figure 30. Captured CPU time

In Figure 30, the performance groups in the Workload Activity Report for the queries (types A, B, and C) account for 70% of the capacity of the system. The remaining performance groups in the Workload Activity Report represent any other work in the system (performance groups/service classes that report the CPU cycles used by other system-related functions) whose resource utilization is not attributable to any single application or query. In Figure 30, this accounts for another 10% of the available system capacity. Finally, there are used CPU cycles that are not captured in RMF Workload Activity Report. This is the uncaptured CPU time, which is the difference between the sum of the CPU seconds reported in the total RMF Workload Report PGN=ALL, and the CPU Activity Report.

To simplify calculating the system capture ratio when you have a homogeneous workload, you can combine the uncaptured CPU cycles that are consumed by the system with the CPU cycles used by the system address spaces, and prorate this across the user application workload. This is done using the methodology to calculate capture ratio described in the *RMF Performance Management Guide* and repeated in Figure 31 on page 175. In this approach, only the query workload performance groups in the Workload Activity Report are combined to represent the user workload in APPL%. This approach groups the system services address spaces reported in the Workload Activity Reports into uncaptured CPU time. A more complex formula for calculating Capture Ratio is presented in the *RMF Performance Management Guide*.

The redbook *DB2 for Capacity Planning*, addresses capture ratio in much more detail. It discusses developing capture ratios at a system level, an application level, and the transaction level. It is up to the individual account to determine the best process for their location. If WLM is set up to segment the query workload, it is possible to develop capture ratios at the individual query type level. However, practically speaking, the approach shown in Figure 31 is appropriate when you have a system where the work has similar processing characteristics, such as all running through the DB2 subsystem.

When the business intelligence application is running in an operating system image with other workloads, it would make sense to follow the steps in the *RMF Performance Management Guide* to develop an application capture ratio.

C a l c u l a t i n g C a p t u r e R a t i o :

$$\text{c a p t u r e r a t i o} = \frac{\text{A P P L \%}}{\text{M V S B u s y \%} \times \# \text{ C P s}}$$

Figure 31. Calculating capture ratio

To develop the capture ratio, use the formula in Figure 31 and information from the appropriate fields in the RMF reports. For APPL%, sum the APPL% value for the performance groups/service classes that pertain only to the query workload. The system address space APPL% will then be incorporated into the uncaptured CPU time. If you are running in LPAR mode, the calculation has to be performed using LPAR BUSY (instead of MVS Busy %), and the number of logical processors. After developing the capture ratio, it is used to uplift the CPU time for the individual workloads, thereby more closely reflecting the CPU usage for a given type of query. Using capture ratios will yield more accurate capacity projections when we return to our worksheets in Appendix A, "Sample worksheets" on page 147.

Appendix E. Using the additional material

This redbook also contains Web material. See the appropriate section below for instructions on using or downloading each type of material.

E.1 Locating the additional material on the Internet

The 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/SG245689>

Alternatively, you can go to the IBM Redbooks Web site at:

<http://www.redbooks.ibm.com/>

Select the **Additional materials** and open the directory that corresponds with the redbook form number.

You can also download the DSTATS utility from the IBM Redbook Web site at:

<ftp://www.redbooks.ibm.com/redbooks/dstats/statistics.zip>

Appendix F. Special notices

This publication is intended to help OS/390 capacity planner address the unique characteristics of business intelligence environment. The information in this publication is not intended as the specification of any programming interfaces that are provided by DB2 UDB for OS/390 and OS/390 V2.7. See the PUBLICATIONS section of the IBM Programming Announcement for DB2 UDB for OS/390 and OS/390 V2.7 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 information about non-IBM ("vendor") products in this manual has been supplied by the vendor and IBM assumes no responsibility for its accuracy or completeness. The use of this information or the implementation of any of these techniques is a customer responsibility and depends on the customer's ability to evaluate and integrate them into the customer's operational environment. While each item may have

been reviewed by IBM for accuracy in a specific situation, there is no guarantee that the same or similar results will be obtained elsewhere. Customers attempting to adapt these techniques to their own environments do so at their own risk.


Any pointers in this publication to external Web sites are provided for convenience only and do not in any manner serve as an endorsement of these Web sites.


Any performance data contained in this document was determined in a controlled environment, and therefore, the results that may be obtained in other operating environments may vary significantly. Users of this document should verify the applicable data for their specific environment.

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

Reference to PTF numbers that have not been released through the normal distribution process does not imply general availability. The purpose of including these reference numbers is to alert IBM customers to specific information relative to the implementation of the PTF when it becomes available to each customer according to the normal IBM PTF distribution process.

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

e (logo)® 
IBM ®

Redbooks
Redbooks Logo 

Data Propagator Relational
DB2 PM
IBM ®
QMF
SMF

DB2
Enterprise Storage System
Intelligent Miner for Data
RMF

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 Kjøbenhavns 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 G. Related publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this redbook.

G.1 IBM Redbooks

For information on ordering these publications see “How to get IBM Redbooks” on page 185.

- *OS/390 MVS Parallel Sysplex Capacity Planning*, SG24-4680
- *DB2 for OS/390 V5 Performance Topics*, SG24-2213
- *DB2 for OS/390 Capacity Planning*, SG24-2244
- *Data Warehousing for DB2 for OS/390*, SG24-2249
- *Building VLDB for BI Applications on OS/390: Case Study Experiences*, SG24-5609
- *The Role of S/390 in Business Intelligence*, SG24-5625
- *S/390 Business Intelligence Architecture Presentation Guide*, SG24-5641
- *OS/390 Workload Manager Implementation and Exploitation*, SG24-5326
- *DPROPR Planning and Design Guide*, SG24-4771

G.2 IBM Redbooks collections

Redbooks are also available on the following CD-ROMs. Click the CD-ROMs button at <http://www.redbooks.ibm.com/> for information about all the CD-ROMs offered, updates and formats.

CD-ROM Title	Collection Kit Number
System/390 Redbooks Collection	SK2T-2177
Networking and Systems Management Redbooks Collection	SK2T-6022
Transaction Processing and Data Management Redbooks Collection	SK2T-8038
Lotus Redbooks Collection	SK2T-8039
Tivoli Redbooks Collection	SK2T-8044
AS/400 Redbooks Collection	SK2T-2849
Netfinity Hardware and Software Redbooks Collection	SK2T-8046
RS/6000 Redbooks Collection (BkMgr)	SK2T-8040
RS/6000 Redbooks Collection (PDF Format)	SK2T-8043
Application Development Redbooks Collection	SK2T-8037

G.3 Other resources

These publications are also relevant as further information sources:

- *RMF Performance Management Guide*, SC28-1951
- *RMF Report Analysis*, SC28-1950
- *RMF Users Guide*, SC28-1949
- *DB2 Performance Monitor for OS/390 Batch User's Guide*, SC26-9167
- *DB2 Performance Monitor for OS/390 Report Reference Volume I*, SC26-9164, and *Volume II*, SC26-9165
- *DB2 UDB Administration API Reference*, SC09-2947
- *DB2 Connect User's Guide*, SC09-2954
- *DB2 Replication Guide and Reference*, SC26-9920

G.4 Referenced Web sites

These Web sites are also relevant as further information sources:

- <http://www.ibm.com/software/download/>
IBM free software download Web site
- <http://www.s390.ibm.com/wlm>
IBM S/390 WLM home page
- <http://www.tpc.org/>
Transaction processing performance council home page

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:

	e-mail address
In United States or Canada	pubscan@us.ibm.com
Outside North America	Contact information is in the "How to Order" section at this site: http://www.elink.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.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.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

First name Last name

Company

Address

City Postal code Country

Telephone number Telefax number VAT number

Invoice to customer number _____

Credit card number _____

Credit card expiration date Card issued to Signature

We accept American Express, Diners, Eurocard, Master Card, and Visa. Payment by credit card not available in all countries. Signature mandatory for credit card payment.

Index

Symbols

% Query workload 108

Numerics

1-tiered implementation 7

2-tiered implementation 7

3-tiered implementation 7

A

ad hoc queries 29

aggregation 43

Analytical users 93

API

Set Accounting String API 61

application logic layer 7

Application profile 69, 70, 148

application server 52

Application Server Information 70

Archiving data

advantages 42

method 43

Associations 140

Automatic Summary Tables 46

B

Backup 40

balanced computer system 9

bandwidth 39

batch processing window 47

batch window 47

batch workload

factors 46

Sizing Estimate 111

BI workload

resource requirements 104

Block density 123, 128

Blocks 130

Business Intelligence

architecture 2

data stores 2

infrastructure 8

logical processing layers 7

processing 3

Business Intelligence (BI) 1

Business Intelligence system

purpose 1

C

Campaign Management Systems (CMS) 2

capacity planner 13

capacity planning 9, 16

checklist 35

critical elements 12

ETL issues 39

information needed 13

methods 15

system resources 12

tools 15

capacity planning team 13

capture ratio 88, 171

cardinality 28

Channels 72

Checklist 35

cleansing 38

clustering 139

compression ratio 129

concurrency 23

concurrency level 29

CPU

consumption 89

cycles 33

parallelism 20

time 65, 108

Critical Path 84, 155

Critical Processing Windows 70, 149

Custom query profiles 106, 158

Customer Relationship Management (CRM) 2

D

Daily Profile 151

data administration 5

data analysis 5

data analyst 14

data collection 4

Data growth 49, 156

consideration 90

data inflation 145

data layer 7

data maintenance 4

data mart 2

- data mining 5
- Data Mining users 93
- data model
 - Entity-Relationship (ER) 31, 161
 - Snow Flake 31
 - snowflake model 162
 - star schema 31, 161
- data population 37
- Data profile 73, 153
- data server 51
- data volume 30
- data warehouse 2, 19
- database administrator 14
- DB2 104
 - Catalog 27
 - DDF thread 36
 - getpages 79
 - inline statistics 41
 - instrumentation data 67
 - LOAD RESUME REPLACE PART 45
 - LOAD utility 41, 45
 - read through locks 24
 - REBUILD INDEX 41
 - REORG UNLOAD EXTERNAL 45
 - REORG utility 41
 - RUNSTATS utility 27, 41
 - SYSCOLDIST 28
 - trace facility 67
 - uncommitted reads 24
 - work area 76
- DB2 Connect 29, 61
- DB2 Estimator 104
- DB2 OLAP Server 5, 123
- DB2 Optimizer 20, 27
- DB2 parallelism
 - goal 21
- DB2 Performance Monitor 66
- DB2 PM
 - #OCCURS 78
 - Accounting 68
 - Accounting Report 78
 - Batch commands 78
 - Batch report 67
 - INCLUDE/EXCLUDE 78
 - Online Monitor 67
 - ORDER 78
 - REDUCE 78
 - REQLOC 78
 - Statistics report 68
- DB2 Statistics 27
- DB2 systems programmer 14
- DBMS 8
- decision support 9
- degree of concurrency 23
- degree of parallelism 20, 23
- dense dimension 129
- dense slice 131
- DPropR
 - Apply components 165
 - capacity requirements 168
 - Capture component 165
 - storage requirements 166
 - workload 170
- DSTATS utility 28, 177

E

- Elapsed time 108
- End User Information 70
- Enterprise Storage Server (ESS) 23
- entity-relationship 161
- ESS
 - Flashcopy 40
- ETL
 - capacity planning issues 39
- ETL process 37, 111
- Extract 37

G

- Growth Projections 69

H

- hash partitioning 22

I

- I/O activity 23
- I/O bandwidth 36, 89
- I/O devices 66
- I/O parallelism 20
- I/O subsystem queue (IOSQ) 23
- IFCID 67
- Index 76
- Inflation 123, 145
- inflation factor 126
- initial load 117, 159
- instrumentation data 67
- Intelligent Miner 5, 137

K

killer query 26

L

latent demand 24
Linear Regression 142
Load 38
locking 23
LPAR 171

M

maintenance processing 63
 profile 69
 requirements 82
maintenance workload 40
 Influencing factors 46
maximum blocks 128
Memory 72
Metadata 5
mining
 data considerations 145
 database design 138
 output data 145
 parallelism 137
 performance 143
 Sampling 144
 Statistical Analysis 142
mining techniques 137
Multidimensional OLAP 123

N

Network specialists 15
normalization 161

O

OLAP
 compression 129
 cube size 134
 system requirements 133
 tools 123
OLTP environments 23
On Line Analytical Processing (OLAP) 5
Online Transaction Processing (OLTP) 19
operational data store (ODS) 2
Operational profile 84, 155
Operational users 93
operations staff 15

P

Parallel access volumes (PAV) 23
Parallel Job stream 115
parallel processing 20
Parallel Sysplex 20
parallelism
 I/O parallelism 20
 server parallelism 138
Partitioning 21
 hash partitioning 22
 range partitioning 21
partitioning key 22
performance data 67
Physical disk used 58
platform specialist 15
power users 30
presentation logic layer 7
Processor Model 71
proofs of concept 104, 119
 performance 120

Q

Query and Reporting 5
query characteristics 19
Query concurrency 24, 25
Query identification 26
Query Management Facility (QMF) 68
Query monopoly 26
query parallelism 19, 20
 CPU parallelism 20
 I/O parallelism 20
 Sysplex parallelism 20
Query profile 60, 62, 73, 153
Query segmentation 73, 79
 by business unit 61
 by type 59
Query sizing 158
Query workload 158

R

range partitioning 21, 22
raw data 3, 58, 76, 108
Recovery 41
relative I/O content (RIOCI) 88
reorganization 41
resource consumption 19
Resource Monitoring Facility (RMF) 66
resource sizing 16

resource utilization 89
response time 24
Risk Management 2
RMF
 CPU Activity Report 66, 72, 174
 Summary Report 66
 Workload Activity Report 66, 172
Rules of thumb (ROT) 104
RUNSTATS utility 27, 41

S

S/390
 system measurement tools 12
sampling 137
sampling strategy 144
segment 44
Sequential Patterns 139
Service Level Agreements (SLAs) 10, 19, 57
Service units 66
sizing 16
SMF records 65
Snapshot 40
snowflake model 162
sparse slice 131
sparsity 123, 126
star schema 161
Storage 65
storage specialist 14
Structured Query Language (SQL) 5, 7
summary tables 32, 45, 76
Supplier Relationship Management (SRM) 2
Supply Chain Management (SCM) 2
Sysplex parallelism 20
System capacity 12
system data 3
System Profile 69, 71
system resources 65
System utilization 150
Systems Management Facility (SMF) 65
systems programmer for OS/390 14

T

tools specialists 14
Transaction systems 10
transfer window 47
Transformation 38

U

Usage pattern 26
user growth 93, 156
 projections 94
user population 30, 108
User profile 59, 73, 153

W

WLM
 goal mode 36
 home page 26, 184
 issues 25
 Period Aging 36
 service definition 25
Workload Characterization 69
Workload ID 69
Workload Manager (WLM) 25
workload profile 68
worksheet 64
Worksheet 1 70, 148
Worksheet 2 71, 150
Worksheet 3 73, 153
Worksheet 4 82, 155
Worksheet 5 94, 156
Worksheet 6 107, 158

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	SG24-5689-00
Redbook Title	Capacity Planning for Business Intelligence Applications: Approaches and Methodologies
Review	
What other subjects would you like to see IBM Redbooks address?	
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.	<input type="checkbox"/> Please do not use the information collected here for future marketing or promotional contacts or other communications beyond the scope of this transaction.
Questions about IBM's privacy policy?	The following link explains how we protect your personal information. ibm.com/privacy/yourprivacy/



Capacity Planning for BI Applications: Approaches and Methodologies

(0.2"spine)
0.17" <-> 0.473"
90 <-> 249 pages



Capacity Planning for Business Intelligence Applications: Approaches and Methodologies



Understand the characteristics of resource consumption of BI applications

Businesses of all sizes are recognizing the powerful effect that data analysis can have on decision-making, resulting in the increasing deployment of Business Intelligence (BI) applications and solutions. This IBM Redbook contains approaches and methodologies that help you do capacity planning for the system resources that support BI applications accessing DB2 on OS/390.

Learn capacity planning methodologies for the S/390 BI infrastructure

Written for S/390 capacity planners, it includes the following:

- A discussion of the fundamental concepts of BI environments
- A structured methodology for doing a capacity plan for the S/390 BI hardware and software infrastructure
- A detailed description of how experienced capacity planners can modify traditional S/390 capacity planning techniques to address the unique characteristics of BI environments
- A set of worksheets to help you and your team members develop a S/390 BI infrastructure capacity plan

Use our capacity planning worksheets

This redbook addresses capacity planning for the S/390 Business Intelligence infrastructure, which includes processors, memory, I/O subsystems and DASD; it does not address capacity planning of network or non-S/390 hardware platforms.

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

SG24-5689-00

ISBN 0738418994