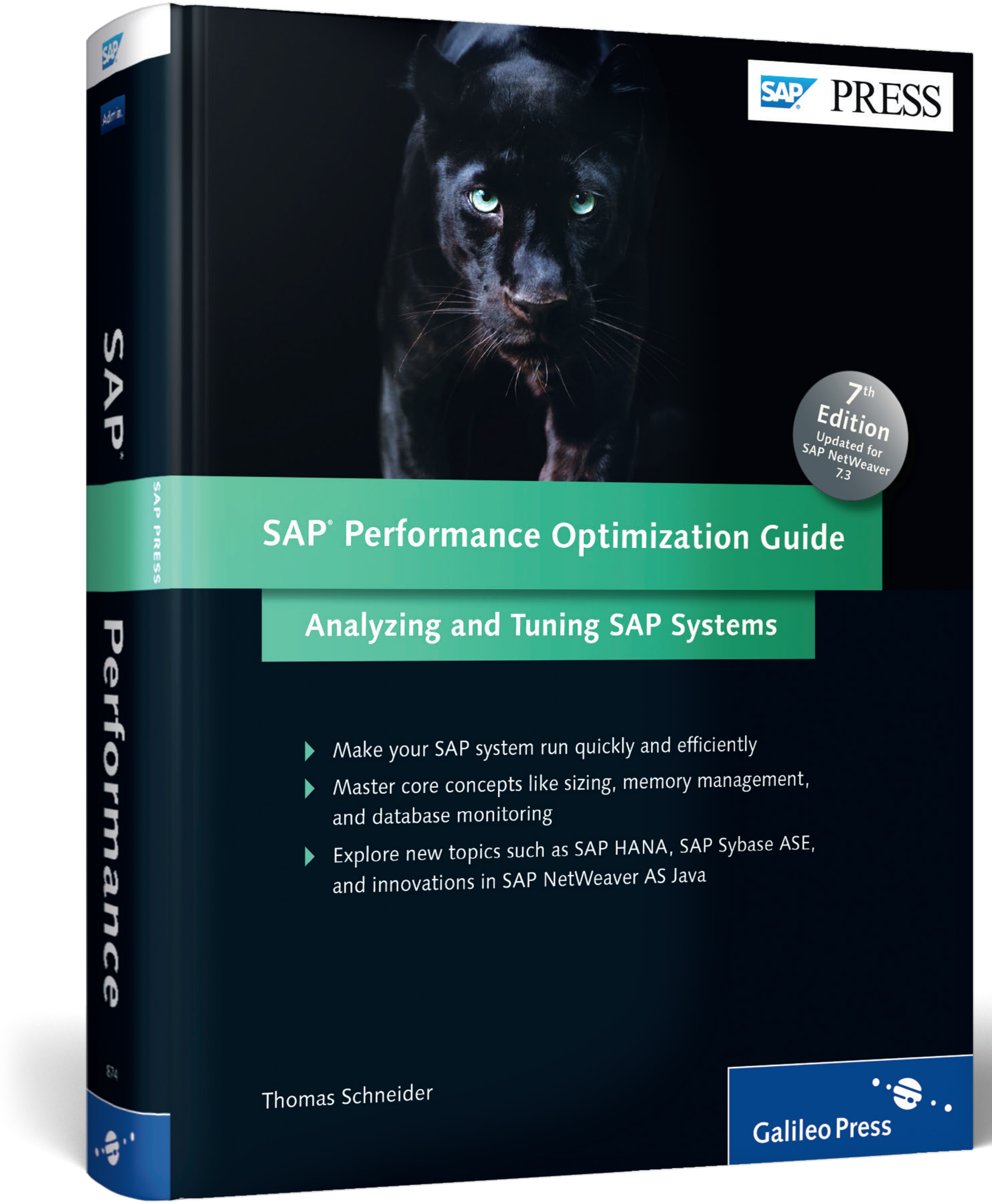
SAP PRESS First-hand knowledge.

Rheinwerk Publishing


SAP PRESS
7th Edition Updated for SAP NetWeaver 7.3
SAP® Performance Optimization Guide
Analyzing and Tuning SAP Systems
Make your SAP system run quickly and efficiently
Master core concepts like sizing, memory management, and database monitoring
Explore new topics such as SAP HANA, SAP Sybase ASE, and innovations in SAP NetWeaver AS Java
Thomas Schneider
Galileo Press
SAP Performance

Reading Sample

Dive into portions of two chapters in this reading sample. In Chapter 2, start analyzing the performance of your hardware, database, SAP memory configuration, and SAP work processes of SAP NetWeaver Application Server ABAP using simple instructions. In Chapter 9, first master the concept of locking for SAP and database systems. Then learn how you can monitor database locks and SAP enqueues.

"Introduction"

"Analysis of Hardware, Database, and ABAP Application Server"

"Locks"

Contents

Index

The Author

Thomas Schneider

SAP Performance Optimization Guide

837 Pages, 2013, $79.95
ISBN 978-1-59229-874-7

www.sap-press.com/3360

Contents

Introduction

Performance

Why is the performance of your business IT application important? Users will be motivated and work efficiently with an application only if response times are good. A slow system leads to downtime and frustration. If the situation deteriorates further, at worst, you no longer have the throughput necessary for running business processes. The results are overtime, delays in production, and financial loss. In contrast, the systematic, proactive optimization of performance considerably increases the value of your business application.

A data processing system's performance is defined as the system's ability to fulfill requirements in terms of response time and data throughput. The system might be required to achieve, for example, a throughput of 10,000 printed invoices in one hour or a response time of less than one second for the creation of a sales order. Good performance, however, is not an absolute characteristic of a business application. Rather, it should be viewed as always relative to the demands made on the application.

Proactive Performance Management

Performance optimization

In this book, *performance optimization* refers to a process that always includes five phases. The first two phases are *understanding the business processes* and *setting and quantifying performance goals*. These steps involve all participating parties—that is, technicians and application experts. Optimization can be successful only on the basis of these prerequisites. Phases three to five involve the *systematic monitoring*, *identification,* and *analysis* of problems, the *implementation of optimization measures*, and further analysis to *verify the success* of the introduced measures (see Figure 1). We advise against randomly tinkering with configuration parameters and similar impulsive tuning measures. The object of this book is to enable you to identify and analyze performance problems in order to deal with them effectively.

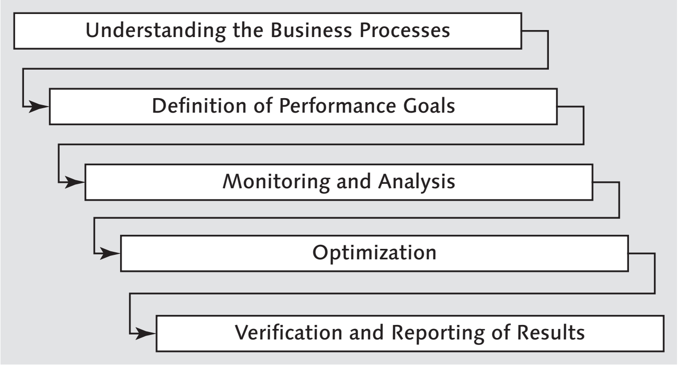

Understanding the Business Processes
Definition of Performance Goals
Monitoring and Analysis
Optimization
Verification and Reporting of Results

Figure 1 Performance Optimization in Five Phases

Technical optimization

From a technical point of view, a business IT application is made up of many different components. These include the logical components: processes such as services, threads, or work processes, as well as memory areas such as buffers and user contexts. There are also the physical components such as processors (CPU), main memory (RAM), hard disks, and network segments. Each of these components allows for maximum throughput and optimal response time. If the interplay among the components is not appropriately balanced or an individual component has reached its performance limit, wait situations that have a negative effect on throughput and response times can occur. In this book, *technical optimization* refers to the identification, analysis, and solution of these problems by tuning the components and distributing the system's load.

Application optimization

The second important task of performance optimization is preventing unnecessary load. Inefficient programs or their suboptimal use can weaken performance. The optimization of individual programs is referred to as *application optimization*.

The goal of optimization is to improve the system settings and applications to achieve the desired performance, based on existing hardware resources. If the existing resources are not sufficient, they must be extended according to the knowledge gained by analysis.

How much tuning is necessary?

How much effort is involved in the performance analysis and tuning of an SAP solution? The answer to this question depends largely on the size of the system. For a small or medium installation with no modifications to the SAP standard or customer developments, it is normally sufficient to do performance optimization just before and shortly after the start of production and after large-scale changes, such as upgrades, large data transfers or client transports, or when new SAP solutions or additional users are introduced into the system. Of course, it is also necessary to intervene when acute performance problems occur. The tuning potential, along with its inherent effort in analysis and optimization, increases proportionately with the size of the system. Experience has shown that many performance bottlenecks are caused by customer developments and modifications to the standard SAP software. The most common reason for this is insufficient testing, but problems can also arise as a result of time constraints or lack of experience on the part of the developer. The extreme case would be a large, constantly developing installation with several hundred users, complicated process chains, a dozen or more developers (often from different consulting firms, working on the system at different times and in different places), and outsourced system management. In such a system environment, it is absolutely necessary for a small group of administrators and developers to have an overview of the entire system and keep an eye on performance.

SAP's remote services offer help with performance analysis and tuning—namely, SAP GoingLive™ Check, which enables your system to make a smooth transition to production operation, and SAP EarlyWatch® Check, which monitors your system and suggests additional optimizations.

Proactive performance management

How does *proactive performance management* help you to achieve the objective of successfully running a business application? You should keep two influencing factors in mind if you're going to achieve this objective: the satisfaction of users and the costs of running the business application. Operating costs arise, on the one hand, from the cost of hardware (e.g., infrastructure, CPU, main memory, hard disks, and networks) and personnel (e.g., administration, maintenance, and error analysis). However, you should not overlook the costs that arise if an application is not available or does not achieve the required performance. In these cases, losses incurred in a few hours or days can exceed an average year's investment in proactive performance optimization. You must compare these costs to the costs of proactive performance management. Table 1 shows the value of proactive performance management with two concrete examples.

Proactive Measure	Effect on System	Immediate Value, Owing to Increased User Satisfaction	Immediate Value, Owing to Lower Operating Costs	Diminished Risk of Deterioration
Optimizing SQL statements	Reduction of database load	Faster response times for certain transactions	Stretching hardware investments (e.g., database server and memory system)	Avoiding overloading the database system
Proactive data management (e.g., data avoidance, archiving, and reorganization)	Reducing database growth Shorter times for maintenance work on the database (e.g., backup/recovery, upgrade, migration, and system copy)	Faster response times for certain transactions Shorter downtime during maintenance work	Stretching hardware investments Fewer personnel requirements for maintenance work	Maintaining manageable database size

Table 1 Examples of the Value of Proactive Performance Management

Current Developments

Challenges posed by the Internet and mobile devices

With the development of the Internet, smartphones, and tablet computers, there has been a paradigm shift in the world of business software: software is no longer aimed at the highly specialized employee on his PC, but to users of the Internet (externally, or internally as an intranet) or the user of a mobile device. With SAP R/3, the traditional strategy of process automation was based on highly specialized users who accessed their SAP Enterprise Resource Planning (ERP) system from fixed work centers via installed SAP graphical user interfaces (GUIs). The role of these specialized agents, who had to be trained to use the software, is becoming unnecessary in many cases. Instead, the end user can have direct access to the enterprise's SAP ERP systems via the Internet and mobile devices. Today, for example, the employees of many enterprises can enter their work and absent times, travel expenses, and so forth into the system themselves via the intranet, whereas previously, this would have been done by central users. Increasingly, customers order products directly via the Internet and no longer by means of letters, faxes, or telephone calls to sales centers.

User expectations concerning the usability and performance of an Internet or mobile application are disproportionately higher than the traditional employee's expectations regarding their SAP ERP system. The employee relies on his own SAP ERP system; if it normally helps to make day-to-day work easier, it is accepted, and minor errors or weak points in performance are tolerated. The Internet user is quite different: if applications offered over the Internet do not work easily and effectively, users can immediately switch to the competition and, for example, make their purchases there—the competition is only a mouse click away. In addition, the Internet does not finish work at five p.m.; an e-business solution on the Internet must be available and work efficiently 365 days a year, 24 hours a day. Users of mobile applications apply the usability and performance standards they are accustomed to with other mobile apps to an SAP application.

SAP HANA

With SAP HANA, SAP has succeeded in launching a product on the market that analysts today refer to as the most important innovation for years in the field of business software. The core element of the innovation is a *main memory database* around which additional services, such as an additional server referred to as *XS Engine*, are grouped. In this book, we tackle the SAP HANA database platform exclusively because practical experience is still lacking in relation to the additional services.

The most important argument in favor of introducing SAP HANA is performance. SAP HANA takes advantage of the availability of huge main memories and massive parallel processor architecture and consistently transforms them into performance by the most modern software architecture. You are perhaps wondering, do I even need a performance book if I have SAP HANA? Or does SAP HANA solve all performance problems? We are firmly convinced that the answer is *no* on both counts. On the one hand, even SAP HANA cannot perform miracles if your program reads complete database tables in the application server and—worse still—sends the data to the user's web browser. The lion's share of the runtime is then attributable to the application server, the network, and the browser. SAP HANA does not override the basic rules for efficient programming! A second *no* stems from the fact that every technical innovation is confronted with “greed” or, in other words, challenges. One of thise challenges is called *Big Data*.

Big Data

By Big Data, we mean the phenomenon whereby more and more data of interest to companies is created, but which up to now, could only be evaluated and made available insufficiently, if at all. Examples of such data include posts in social networks; logs of web accesses; transaction data of persons (for example, via mobile positioning or posts with location data in social networks) and products (which, for example, are covered via RFID chips); data from cameras, microphones, and other sensors; financial transactions; and stock market data, as well as consumption data in the energy sector. Companies are eager to collect, link, and evaluate this data and thus gain valuable insights into their customers, markets, and products. A production engineer in the consumer goods industry can be used as an example here. In the future, he will be able to use not only past production figures and current orders, but also the current trends of social networks, to plan more effectively.

Small “errors,” major impact

All of these data sources have in common that their quantity exceeds that of traditional business data (so-called master and transaction data) many times over. When transferred to our performance issue, this means that a non-performance-optimized system or program has a much greater impact than a “traditional” SAP Business Suite system. Thus, we conclude that performance know-how will continue to be a valuable asset in the future.

IT services

The demand for an open, flexible software architecture requires specialized, independently running software components that are linked via interfaces, which means a business process involves several software components. The constantly growing number of solutions and components presents an administrative challenge for data centers. The number of components has grown from the manageable SAP R/3 (with SAP instances, database, and hardware/operating system) to a constantly increasing range of technologies, including products that SAP does not produce but offers as a reseller.

Consequently, business process operators counteract this trend by integrating more and more service partners into the service and support processes. Outsourcing may involve only hardware (e.g., computer performance, hard disk memory, network resources, and so on), or it may also involve the application itself (i.e., *application service providing*, or ASP); for example, the services of an Internet product catalog can be completely allocated to a service provider instead of being operated by the catalog software in the enterprise. It is thus not only necessary to monitor hardware and software components, but monitoring must also go beyond company and component boundaries.

Overall, completely new requirements arise for administration and monitoring of SAP solutions—requirements that you cannot deal with using traditional concepts.

About This Book

Fundamentals of this book

The methods for performance analysis and optimization presented in this book reflect those initially used by experts in the *EarlyWatch service* and *GoingLive Check* and are included in the SAP Basis training courses ADM315 *Workload Analysis* and ADM490 *Optimization of ABAP Programs*. This is the seventh edition of this book, and with each new edition, we take the opportunity to thoroughly describe current trends in product development at SAP and, wherever relevant, to consider developments in the IT world in general.

In this edition, we have almost completely rewritten the Java topics and combined them in one chapter. Another new chapter is dedicated to the most important innovation from SAP for many years, SAP HANA, which initially focuses on the SAP HANA database platform. Other SAP HANA services are not yet dealt with due to a lack of practical experience. All other chapters have been revised and updated for this edition; for example, we describe the new ABAP load distribution concept (new to SAP NetWeaver 7.40) and have included the topic of background remote function calls (RFCs). We have also added a section on the SAP Sybase ASE database to Appendix A (Database Monitors).

Structure

Figure 2 presents the chapters of this book based on the five phases of performance optimization at a glance. **Chapter 1** of this book, Performance Management of an SAP Solution, is intended for both SAP administrators and SAP consultants, as well as application developers and SAP project leads. It discusses the following fundamental questions about performance analysis at a non-technical level:

- Which preventative measures must you take to guarantee the optimal performance of an SAP solution?
- What performance tuning measures should you take into consideration?
- Who is involved in the tuning process?

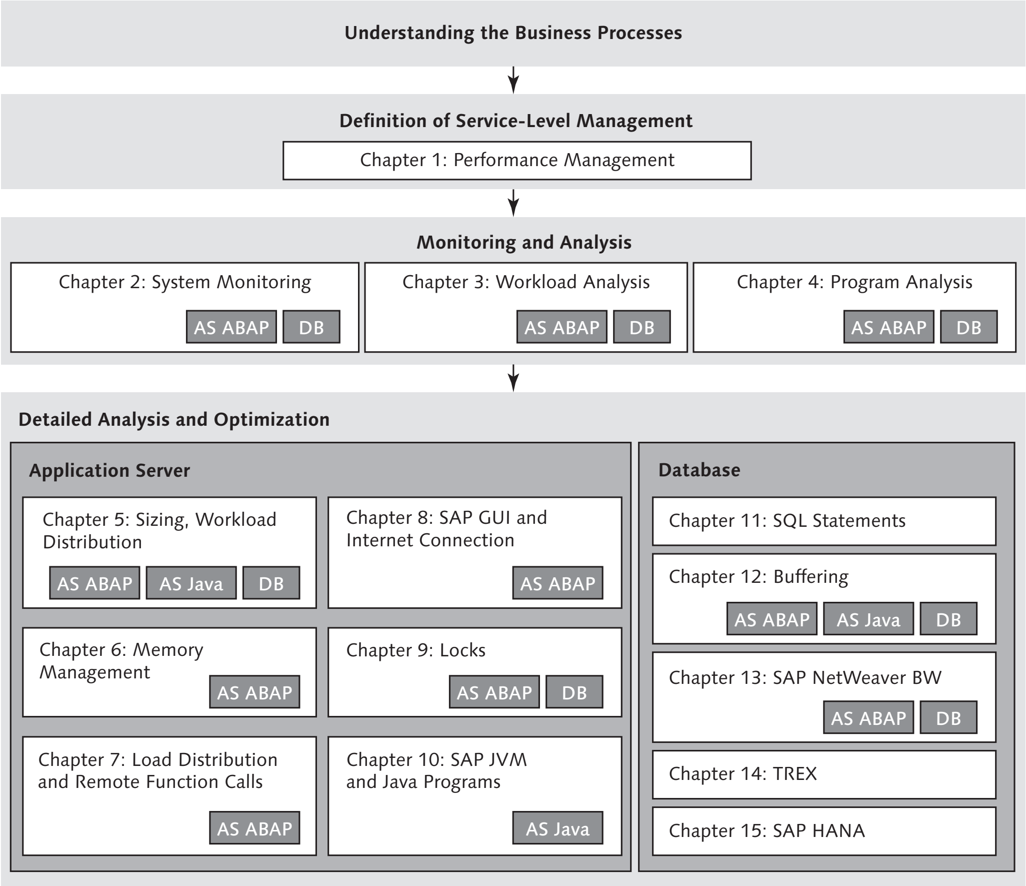

Understanding the Business Processes
Definition of Service-Level Management
Chapter 1: Performance Management
Monitoring and Analysis
Chapter 2: System Monitoring
AS ABAP
DB
Chapter 3: Workload Analysis
AS ABAP
DB
Chapter 4: Program Analysis
AS ABAP
DB
Detailed Analysis and Optimization
Application Server
Chapter 5: Sizing, Workload Distribution
AS ABAP
AS Java
DB
Chapter 8: SAP GUI and Internet Connection
AS ABAP
Chapter 6: Memory Management
AS ABAP
Chapter 9: Locks
AS ABAP
DB
Chapter 7: Load Distribution and Remote Function Calls
AS ABAP
Chapter 10: SAP JVM and Java Programs
AS Java
Database
Chapter 11: SQL Statements
Chapter 12: Buffering
AS ABAP
AS Java
DB
Chapter 13: SAP NetWeaver BW
AS ABAP
DB
Chapter 14: TREX
Chapter 15: SAP HANA

Figure 2 The Chapters of This Book

The service provided for the user frequently turns out to be a combination of a number of different services carried out by a network of partners. Parts are provided by many different, sometimes external, service providers. To master this complexity, many service providers and customers implement *service-level management* (SLM). SLM calls for a structured, proactive method to ensure an adequate service level for the IT application users, taking into account both cost efficiency and the customer's business objectives. In this book, we'll describe the tools and methods used to implement SLM for an SAP solution.

Chapters 2-4 present performance analysis based on SAP NetWeaver Application Server (AS) ABAP. After reading this chapter, you will be able to perform a systematic performance analysis for AS ABAP, including databases and operating systems.

In this book, we initially follow the bottom-up analysis strategy, starting in **Chapter 2**, Monitoring Hardware, Databases, and ABAP Application Server, with an examination of the operating system, database, SAP memory management, and SAP work processes. At the same time, we provide solution proposals that should enable the administrator or consultant to solve the most important performance problems. For small and medium-size installations, this level of tuning is often sufficient.

Then, **Chapter 3**, Workload Analysis, discusses the more complex workload analysis as an example of top-down analysis. In **Chapter 4**, Identifying Performance Problems in ABAP Programs, you will find methods for analyzing individual programs using tools such as single-record statistics, SQL trace, and ABAP runtime analysis, among others.

The remainder of the book, Chapters 5-15, presents information necessary for a more in-depth performance analysis. These chapters are intended for SAP consultants responsible for the efficient functioning of large systems who need to reach the full tuning potential of their systems. These chapters are independent units to a large extent, and you can read them in any order once you are familiar with the content of the first four chapters. Any dependencies are shown at the beginning of each chapter.

Chapters 5-10 deal with the topics that relate to the application server and the presentation server:

- **Chapter 5**, Hardware Sizing and System and Load Distribution: This is the guide to avoiding hardware bottlenecks on the one hand, and limiting the costs of unnecessary hardware on the other hand. Server consolidation—that is, the concentration of all services on a few powerful computers—has become an important IT market trend in recent years. We'll describe what you must take into account to use these technologies efficiently.

- **Chapter 6**, Memory Management: The configuration of the memory areas allocated by the SAP system has a considerable influence on performance.
- **Chapter 7**, Workload Distribution and Remote Function Calls: Optimal workload distribution of web, dialog, update, and background requests helps ensure efficient use of hardware and the avoidance of bottlenecks brought about by suboptimal configurations. Interface performance between software components also contributes greatly to the efficiency of the entire solution. E-business solutions that consisted solely of a monolithic R/3 system were rarely used, even in the past. Instead, open solutions that comprise several components connected to each other via interfaces are the standard.
- **Chapter 8**, SAP GUI and Internet Connection: Analysis and configuration recommendations demonstrate the optimization potential of linking GUIs (i.e., a classical SAP GUI or web browser) with the application. The chapter discusses performance aspects of SAP GUI controls, Internet Transaction Server (ITS), Business Server Pages, and Web Dynpro for ABAP in detail.
- **Chapter 9**, Locks: Database and SAP locks ensure data consistency. You can avoid bottlenecks in throughput with an optimized administration of locks (for example, with the ATP server or by buffering number ranges).
- **Chapter 10**, Optimizing Java Virtual Machine and Java Programs: This chapter contains the description of the tools with which you can perform the performance analysis of Java Virtual Machine (SAP JVM) and Java programs.

Chapter 11 introduces the series of chapters on database topics and is a prerequisite for Chapters 12-15:

- **Chapter 11**, Optimizing SQL Statements: Ineffective SQL statements make heavy demands on the database and can hamper the performance of the entire application. In this chapter, we present a detailed analysis of "expensive" SQL statements, as well as optimization options via database indexes and program optimization (i.e., "five golden rules").
- **Chapter 12**, SAP Buffering: Buffered tables on the application servers speed up access to frequently read data and help ease the load on the database.
- **Chapter 13**, Optimizing Queries to SAP NetWeaver Business Warehouse: SAP NetWeaver Business Warehouse queries are special SQL statements that usually process large quantities of data. Special optimization options exist for this type of queries.
- **Chapter 14**, Optimizing Search Queries Using TREX: You can use TREX for optimizing text-based and attribute-based search queries and SAP NetWeaver Business Warehouse queries (SAP NetWeaver BW Accelerator) instead of traditional database indexes and aggregation tables.
- **Chapter 15**, Optimization of Database Queries with SAP HANA: A separate chapter is dedicated to the new "child prodigy" from SAP, SAP HANA, which introduces the principles, tools, and methods for performance analysis and optimization.

Target groups

Knowledge of performance optimization of SAP systems and applications is highly beneficial for SAP administrators, SAP application managers, SAP developers, and SAP project leads, and these are the target groups of this book. Every chapter first provides an introduction that is followed by a short section, "When Should You Read This Chapter?," that specifies the target group of the chapter.

Prerequisites

This book assumes theoretical and practical knowledge of the administration of SAP components in areas that involve the specific implementation of recommendations. You should be familiar with the use of the *Computer Center Management System* (CCMS), in particular. SAP NetWeaver Application Server ABAP System Administration (see Appendix E, Sources of Information) should serve as good preparation. Parts of this book (for instance, Chapters 4, 9, 11, and 12) also assume familiarity with the ABAP programming language, the functioning of relational databases, and SQL.

Limits of this book

The book does not cover the following topics:

- **Hardware tuning and network tuning**
  Although this book helps you to identify bottlenecks in the CPU, main

memory, I/O, or network, a detailed analysis would require hardware or network provider tools. In view of the enormous number of products offered, we cannot include this subject (especially the tuning of hard disks).

- **Databases**
  In the CCMS, SAP offers tools that standardize most administrative and analysis tasks for different database systems. If you want to do more in-depth database tuning, however, you need to be familiar with the different database system architectures. It is not possible for this book to go into sufficient detail on the fine points of all database systems that can be used in conjunction with SAP solutions. However, this information is also unnecessary because reference material on tuning is available for all database systems. This book cannot replace these materials, nor does it endeavor to do so. Instead, the emphasis is on the SAP-specific context of database tuning and on explaining concepts common to all database systems. The specific examples used always refer to individual database systems. In Appendix A, you will find an overview of the most important monitors for analyzing database systems.
- **Application tuning**
  Many problems with performance can be solved only with detailed knowledge of the application and the individual SAP system modules. A change in customized settings often solves the problem. This book does not provide the know-how for tuning individual SAP system modules. However, it does provide you with analysis strategies so you can narrow performance problems down to certain applications and then consult the appropriate developer or consultant.

Release dependency

One question that was heatedly discussed prior to this book's publication is the extent to which release-dependent and time-dependent information, for example, menu paths, recommendations for configuration parameters, and guide values for performance counters, should be included. Factors such as a new version, patches (for the SAP system component, database, or operating system), or a new generation of computers, among others, could render previous information obsolete overnight. In the worst-case scenario, outdated recommendations could even have negative effects on performance. We are aware of this risk. Nevertheless, we have decided to include time-dependent information and rules in this book. This is the only way you can use this book as a reference for daily work in SAP administration. On the other hand, it is clear that this is not a book of fixed rules and regulations, and anyone who views performance optimization as mechanical rule following is mistaken. This book cannot replace direct analysis of the solution, the SAP Help Portal, or up-to-date SAP Notes on the SAP Service Marketplace. It aims only to support them.

SAP NetWeaver 7.30

All information on menu paths, references to performance monitor screens, and guideline values for performance counters refer to SAP NetWeaver 7.30, unless otherwise noted. At some points, we give a preview of SAP NetWeaver 7.40.

Notes on reading

In this book, you will find several orientation aids that are intended to facilitate your reading of its contents.

Highlighted information boxes include content that is worth knowing and useful, but which is also beyond the actual explanation. To enable you to immediately classify the information in the boxes, we have marked the boxes with symbols:

- The *Tips* marked with this symbol give you specific recommendations that can make your work easier. [«]
- In boxes that are marked with this symbol, you will find information about *additional topics* or important content that you should remember. [+]
- This symbol indicates *features that you should note*. It also *warns you* about common errors or problems that may occur. [!]
- *Examples* identified by this symbol indicate scenarios from practical experience and demonstrate the presented functions. [Ex]

www.sap-press.com

As for previous editions, we will provide updates and, if necessary, corrections to the book on the publisher's website (*www.sap-press.com*).

With the analysis of hardware resources, the database, and the work processes and memory areas of SAP NetWeaver AS ABAP, we are going bottom-up into performance analysis. Get an initial overview of the current situation in the system.

2 Analysis of Hardware, Database, and ABAP Application Server

This chapter provides the basic information on analyzing the performance of your hardware, database, SAP memory configuration, and SAP work processes of SAP NetWeaver Application Server ABAP. Procedure roadmaps at the end of each section summarize the most important analysis paths and clarify when to use the various monitors. The last section describes the central Alert Monitor, which integrates the performance indicators from all areas.

This chapter will provide simple recommendations to help you optimize each component, except where in-depth explanations are required (these are given in subsequent chapters). Unnecessary background information is intentionally kept to a minimum so that even application consultants or system administrators with limited experience in performance analysis can use this chapter to improve the performance of their system. For example, we describe monitoring and customizing SAP extended memory without explaining SAP extended memory in detail. You can find more detailed information in Chapters 5-15. Our experience suggests that you can solve many performance problems in the operating system, database, and SAP Basis by using simple instructions, without delving into technical details.

When Should You Read This Chapter?

You should read this chapter if you want to use your SAP system to technically monitor and optimize the performance of your SAP system, database, or operating system.

2.1 Basic Terms

The terms *computer*, *server*, *application server*, *SAP instance*, *database*, *database server,* and *database instance* are used in this book as follows:

Computer

A *computer* will always mean a physical machine with a CPU, main memory, IP address, and so on.

SAP application instance

An *SAP application instance*, also referred to as an *SAP instance*, is an administrative unit: it consists of a set of SAP work processes that are administered by a dispatcher. It also includes a set of SAP buffers located in the host computer's shared memory and accessed by the work processes. An SAP application instance can be an ABAP application instance (*SAP NetWeaver Application Server ABAP*, referred to as *AS ABAP*), or a Java application instance (*SAP NetWeaver Application Server Java*, referred to as *AS Java*). There can be multiple SAP instances on one computer. As a result, there will be multiple dispatchers and sets of buffers. An *application server* is a computer with one or more SAP instances.

Database

Every SAP system has only one database. The term *database* refers to a set of data that is organized into files, for example. The database can be thought of as the passive part of the database system.

The active part of the database system is the *database instance*, an administrative unit that allows access to the database. A database instance consists of database processes with a common set of buffers in the shared memory of a computer. A *database server* is a computer with one or more database instances. A computer can be both a database server and an application server if a database instance and an SAP instance run on it.

In the SAP environment, there is normally only one database instance for each database. Examples of database systems in which multiple database instances can access a database are DB2 and Oracle Parallel Server. This book does not cover the special features of these *parallel database systems*.

SAP system

We refer to SAP software components that are based on SAP Basis as *SAP systems*. These are SAP ERP, SAP NetWeaver BW, SAP APO, SAP SRM, and SAP NetWeaver Portal.

According to this terminology, an SAP ERP system can consist of one or two systems, depending on whether the Java and ABAP parts run on a joint system with one database (for example, SAP NetWeaver double stack) or on two systems with separate databases. This terminology also applies to SAP Solution Manager.

Server

SAP documentation and literature use the term *server* in both a hardware sense and a software sense. Therefore, the term can refer to a computer, for example, in the term *database server*, and to a logical service, such as in the terms *message server* and *ATP server*. Thus, we also use *ABAP server* or *Java server* as short forms for *SAP NetWeaver Application Server (AS) ABAP* or *Java*.

2.2 Hardware Monitoring

Operating system monitor

The operating system monitor analyzes hardware bottlenecks and operating system problems. To start the operating system monitor for the application server you are currently logged on to, select the following menu:

Tools • Administration • Monitor • Performance • Operating system • Local • Operating system monitor

Alternatively, you can use Transaction ST06. The main screen of the operating system monitor appears.

Call and availability

The operating system monitor was revised for SAP Basis version 7.10. Since the revision, Transactions OS06, OS07, and ST06 open a monitor that you can use to monitor both the local and remote computers. For versions prior to 7.10, the new transactions are available under Transactions OS06N, OS07N, and ST06N; with Transactions OS06, OS07, and ST06, you can still access the older transactions. All information discussed in this book is also available in the old transactions. You can view the detail analysis by clicking the Detail Analysis Menu button.

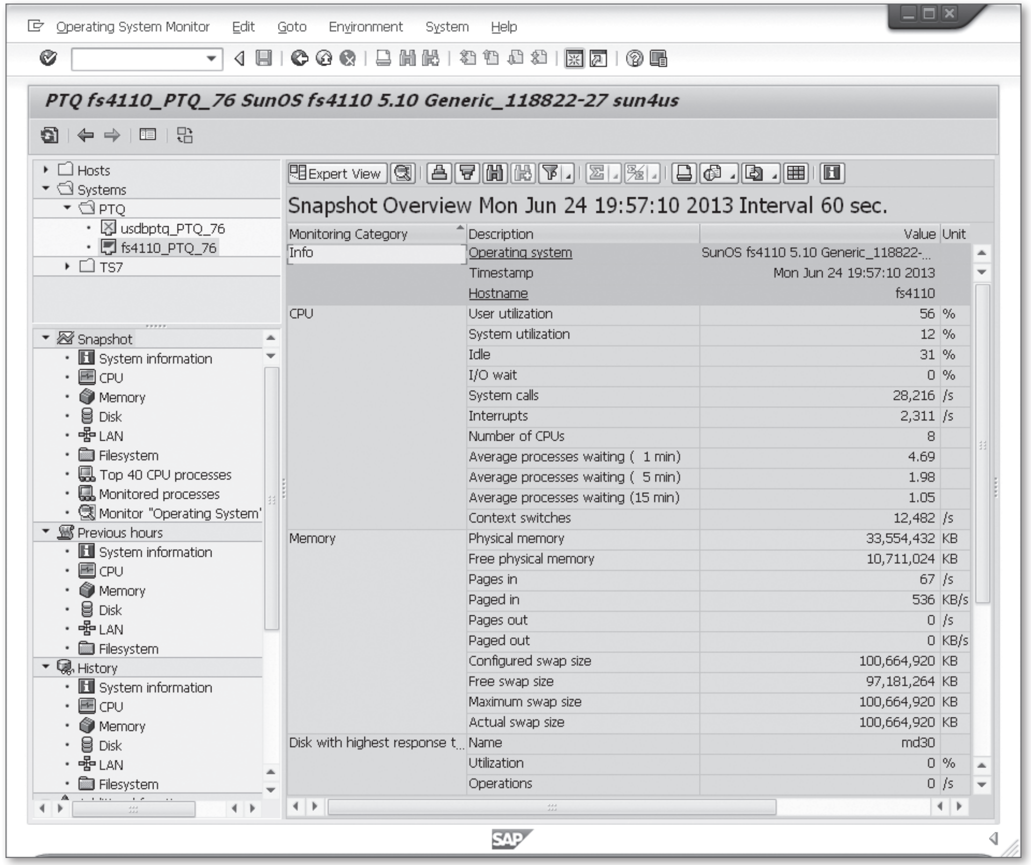
You can also call the operating system monitor from the server overview:

Tools • Administration • Monitor • System monitoring • Server (Transaction SM51)

Then, position the cursor on the desired application server and, in the menu, choose GoTo • Monitors • OS Monitor.

Structure

The operating system monitor screen is divided into three areas (see Figure 2.1). In the top-left window, you can view the list of computers that are monitored. There, you can select a computer that you want to analyze. In the lower-left window, you select the analysis data. The window on the right contains data on the selected computer and analysis.



PTQ fs4110_PTQ_76 SunOS fs4110 5.10 Generic_118822-27 sun4us

Snapshot Overview Mon Jun 24 19:57:10 2013 Interval 60 sec.

Monitoring Category	Description	Value	Unit
Info	Operating system	SunOS fs4110 5.10 Generic_118822-...	
	Timestamp	Mon Jun 24 19:57:10 2013	
	Hostname	fs4110	
CPU	User utilization	56	%
	System utilization	12	%
	Idle	31	%
	I/O wait	0	%
	System calls	28,216	/s
	Interrupts	2,311	/s
	Number of CPUs	8	
	Average processes waiting (1 min)	4.69	
	Average processes waiting (5 min)	1.98	
	Average processes waiting (15 min)	1.05	
	Context switches	12,482	/s
Memory	Physical memory	33,554,432	KB
	Free physical memory	10,711,024	KB
	Pages in	67	/s
	Paged in	536	KB/s
	Pages out	0	/s
	Paged out	0	KB/s
	Configured swap size	100,664,920	KB
	Free swap size	97,181,264	KB
	Maximum swap size	100,664,920	KB
	Actual swap size	100,664,920	KB
Disk with highest response t...	Name	md30	
	Utilization	0	%
	Operations	0	/s

Figure 2.1 Main Screen of the Operating System Monitor

By default, the selection list shows all computers on which SAP ABAP instances have been installed. Essentially, any computer can be integrated into the remote operating system monitor, provided a monitoring agent has been installed on the relevant computer. We strongly recommend installing monitoring agents on computers that run a stand-alone database, an SAP Java instance, or a TREX.

SAP Support

You should install this monitor even if you use a tool from a different vendor to monitor utilization of your computers. If you need support from SAP, an SAP expert can analyze the computers only via the SAP monitor.

2.2.1 Analysis of a Hardware Bottleneck (CPU and Main Memory)

You will find an overview of the most critical operating system and hardware data under Snapshot in the analysis selection of the operating system monitor (see Figure 2.1). All data is refreshed every 10 seconds by the auxiliary program `saposcol`. To update the data on the screen (after 10 seconds or longer), you need to click the corresponding button.

CPU load

Under the header CPU, you will find the fields User Utilization, System Utilization, and Idle. These values indicate the percentage of total CPU capacity currently being used by user processes (i.e., the SAP system, database, and other processes), the percentage being used by the operating system itself, and the percentage not being used. The Number of CPUs field indicates the number of CPU threads. Average Processes Waiting is the average number of work processes waiting for a free processor. This value is indicated as averaged over 1, 5, and 15 minutes. The other values in the CPU header are of less importance for the performance analysis. Table 2.1 provides an overview of the fields of the operating system monitor.

Field	Explanation
User Utilization	CPU workload caused by user processes (SAP system, database, etc.).
System Utilization	CPU workload caused by the operating system.
Idle	Idle CPU capacity. This value should be at least 20%, but ideally at least 35%.
Number of CPUs	Number of CPU threads.
Average Processes Waiting	Number of processes waiting for CPUs, averaged over 1, 5, or 15 minutes.
Physical Memory	Available physical main memory (RAM) in KB.

Table 2.1 Operating System Monitor Fields

[»] **Processors, Cores, and Threads**

As a description of the computer equipment, you can find the following specifications, for example: “2 processors, 8 cores, 16 threads, processor of manufacturer X with 2.93 GHz clock speed.” What does this information on the number of *processors*, *cores*, and *threads* mean for the SAP system?

The term *processor* refers to the *central processing unit* (CPU) of a computer, which is capable of executing programs. Here, a distinction is made between single-core processors and multicore processors. Multicore processors have multiple fully developed processing cores on one chip. The individual cores share only the bus; that is, they are considered full CPUs. Multithreaded CPUs have *one* CPU, but register as *multiple* CPUs on the operating system. As a result, various queues exist for these cores between which the core switches. To optimize this switching, each thread has its own register set, including a stack pointer and program counter, so you can switch among the threads without additional processor cycles. These hardware-based threads should not be confused with the threads that generate the application processes (that is, user or software threads). Within a process of the database or the ABAP, Java, or TREX server, you can generate multiple (software) threads that the operating system executes in time slices. The switch between the (software) threads is referred to as a *context switch*. Considering this, it can be assumed that additional (hardware) threads promote context switches among (software) threads and therefore support a better utilization of the existing core; however, the increase in performance doesn’t fully come up to an additional core.

Main memory utilization and swap space

The MEMORY header contains information on the available physical main memory (PHYSICAL MEMORY field) and values of the operating system paging.

Under the SWAP HEADER, the amount of currently allocated *swap space* is listed. The swap space must be greater than the total of the configured memory area.

[!] **Program Terminations Due to Memory Shortage**

If the sum of the physical memory and swap space is smaller than the total amount of memory required by the SAP system, database, and other programs, this may cause program terminations or even operating system failure. You should therefore ensure that there is sufficient swap space.

History: CPU and main memory

To display the CPU workload over the previous 24 hours, select the analysis PREVIOUS HOURS • CPU. The PREVIOUS HOURS CPU screen is displayed. The column headers are the same as in the fields under CPU in the operating system monitor initial screen, except that the values are for one hour. Similar overview is available for main memory usage (PREVIOUS HOURS • MEMORY) for the swap space, and so on.

When Is There a CPU or Main Memory Bottleneck?

Guideline values

The unused CPU capacity IDLE should normally average at least 20% per hour. This enables the system to accommodate temporary workload peaks. A reading of 35% idle CPU capacity is even better. For the paging rate, the following guideline values apply:

- For computers that contain a database, Java instance, or TREX, only very minor paging rates should occur; that is, they should be dimensioned in such a way that the available main memory corresponds to the configured memory areas.
- For computers that include only ABAP instances, you can tolerate moderate paging rates of up to 20% of the physical main memory per hour.

For operating systems that page continuously (for example, Microsoft Windows), the value indicated in the operating system monitor as the *paged-in rate* is the key statistic on paging performance. For other operating systems that page only when necessary (such as most UNIX derivatives), the key statistic is the *paged-out rate*. If the operating system monitor sometimes shows values that exceed these guideline values, this does not automatically mean you have a hardware bottleneck. Rather, you should use the workload monitor to check whether the high CPU workload or the paging rate is associated with poor response times. Corresponding analyses can be found in Chapter 3, Section 3.4.1, Analyzing General Performance Problems.

Main memory bottleneck

If you observe high paging rates on several computers, you should calculate the virtual main memory allocated by the SAP instances and the database (see Sections 2.4.3 Displaying Allocated Memory and 2.3.2 Analyzing the Database Buffer). Compare this with the available physical main memory. As a rule of thumb, there should be approximately 50% more virtual memory than physical memory.

In Microsoft Windows and Oracle Solaris operating systems, the analysis of the paging rate on the database server can lead to misinterpretation because in these operating systems, read/write operations (I/O) can sometimes be counted as paging. For more information on this issue, please refer to SAP Notes 124199 (Solaris) and 689818 (Windows).

Causes of Hardware Bottlenecks

If you detect a hardware bottleneck on one or more SAP system computers, it may be due to one or more of the following causes:

- **Incorrect load distribution**
  In a distributed system with multiple computers, if you discover a hardware bottleneck on at least one computer while other computers have unused resources, the workload is probably not optimally distributed. To improve performance, redistribute the SAP work processes and the user logons.

  It is extremely important that the database server has enough resources. A CPU or main memory bottleneck on the database server means the required data cannot be retrieved quickly from the database, which causes poor response times in the entire system.
- **CPU load of individual programs**
  In the operating system monitor (Transaction ST06), select the analysis Snapshot • Top CPU processes. The overview of the operating system processes is displayed. Here, you can see all currently active processes and their demands on resources.

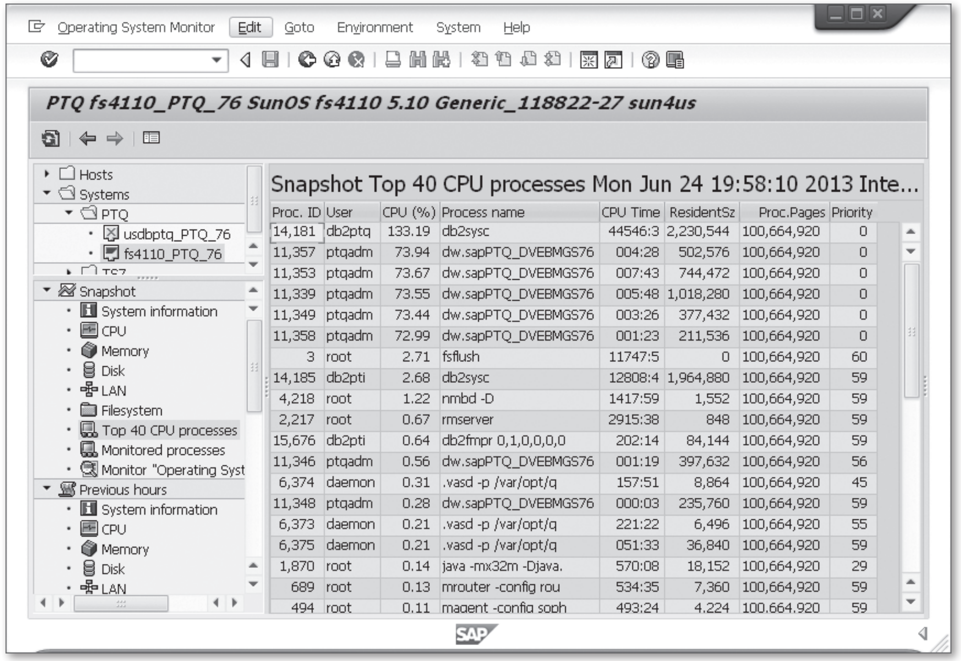
Overview of operating system processes

Figure 2.2 shows an overview of a system on which an ABAP instance and a DB2 database are installed. You can identify the following processes:

- `dw_<instance>`: SAP work process of the SAP ABAP instance on a UNIX operating system. On Windows operating systems, the name is `disp+work`.
- `db2sysc`: Database process of DB2 database. The processes of other databases normally carry the brand name (such as Oracle), which appears in the process or user name.

Operating system processes, which you can identify from the following elements of the name, are also part of SAP instances:

- `jstart` … : Server process of SAP Java instance.
- `TREX` … : TREX process. The server type is indicated in the process name, for instance, index server, preprocessor, and so on.
- `icman` … : Process of the Internet Communication Manager (ICM).
- `saposcol` … : Auxiliary program, which collects the data for the operating system monitor, for example.



PTQ fs4110_PTQ_76 SunOS fs4110 5.10 Generic_118822-27 sun4us

Snapshot Top 40 CPU processes Mon Jun 24 19:58:10 2013 Inte...

Proc. ID	User	CPU (%)	Process name	CPU Time	ResidentSz	Proc.Pages	Priority
14,181	db2ptq	133.19	db2sysc	44546:3	2,230,544	100,664,920	0
11,357	ptqadm	73.94	dw.sapPTQ_DVEBMGS76	004:28	502,576	100,664,920	0
11,353	ptqadm	73.67	dw.sapPTQ_DVEBMGS76	007:43	744,472	100,664,920	0
11,339	ptqadm	73.55	dw.sapPTQ_DVEBMGS76	005:48	1,018,280	100,664,920	0
11,349	ptqadm	73.44	dw.sapPTQ_DVEBMGS76	003:26	377,432	100,664,920	0
11,358	ptqadm	72.99	dw.sapPTQ_DVEBMGS76	001:23	211,536	100,664,920	0
3	root	2.71	fsflush	11747:5	0	100,664,920	60
14,185	db2pti	2.68	db2sysc	12808:4	1,964,880	100,664,920	59
4,218	root	1.22	nmbd -D	1417:59	1,552	100,664,920	59
2,217	root	0.67	rmserver	2915:38	848	100,664,920	59
15,676	db2pti	0.64	db2fmpr 0,1,0,0,0,0	202:14	84,144	100,664,920	59
11,346	ptqadm	0.56	dw.sapPTQ_DVEBMGS76	001:19	397,632	100,664,920	56
6,374	daemon	0.31	.vasd -p /var/opt/q	157:51	8,864	100,664,920	45
11,348	ptqadm	0.28	dw.sapPTQ_DVEBMGS76	000:03	235,760	100,664,920	59
6,373	daemon	0.21	.vasd -p /var/opt/q	221:22	6,496	100,664,920	55
6,375	daemon	0.21	.vasd -p /var/opt/q	051:33	36,840	100,664,920	59
1,870	root	0.14	java -mx32m -Djava.	570:08	18,152	100,664,920	29
689	root	0.13	mrouter -config rou	534:35	7,360	100,664,920	59
494	root	0.11	magent -config soph	493:24	4,224	100,664,920	59

Figure 2.2 Analysis of the Top CPU Processes in the Operating System Monitor

Analysis of the CPU load

To check whether individual processes are placing a heavy load on the CPU for long periods of time, refresh the monitor periodically and observe any changes in the value CPU (%). If the processes that place a heavy load on the CPU entail processes of SAP Basis or the database, the subsequent specified monitors provide further information on the processes' activities.

Start the monitor in a second mode, identify the process with the heavy CPU load using the process ID, which you can also find in the corresponding basis monitors, and check the monitors to determine which program or tables, queries, and so on are being processed by the process.

- **SAP work processes of ABAP instance**
  Open a new user session and call the local work process overview (see Section 2.5, Analyzing SAP Work Processes). From the work process overview, note the name of the ABAP program and the user corresponding to the process identifier (PID).
- **Server process of Java instance**
  Open the SAP Management Console (see Chapter 10, Section 10.3, SAP Management Console). Use a thread dump to obtain process-internal information.
- **TREX processes**
  Open the TREX administration tool (see Chapter 14, Section 14.2). You can find details on the TREX services in the Services monitor.
- **ICM**
  Open the ICM monitor (see Section 2.6, Analysis of the Internet Communication Manager [ICM]).
- **Database processes**
  Open the database process monitor in the Database Administration (DBA) Cockpit (see Section 2.3.3, Identifying Expensive SQL Statements) to identify the SQL statements that are being processed by the database.

Using the operating system monitor in conjunction with the monitors mentioned, you can fairly easily identify programs, transactions, SQL statements, or TREX queries that cause high CPU load.

External processes

A CPU bottleneck can be caused by *external processes*. In the operating system monitor, if you find external processes (that is, processes that are not part of the SAP system) with high CPU consumption that cause a CPU bottleneck, you should find out whether these processes are really necessary for your system or whether they can be switched off or moved to another computer. The following are examples of external processes: administrative software, virus scanners, backups, external systems, screen savers (!), and so on.

[Ex]

Identifying a CPU bottleneck

Suppose you notice a CPU bottleneck during times of peak user activity. The process overview in the operating system monitor reveals a single SAP work process that is causing a CPU load of 30% over several minutes. At the same time, the SAP work process overview shows a long-running background program. You should try to see if the background program could be run at a time when the dialog load is lighter.

Memory requirement of individual programs

To identify programs with high memory requirements that may be causing a main memory bottleneck, you can use a method similar to that previously described for CPU bottlenecks. See also Chapter 6, Memory Management.

Minimize file system cache

Operating systems normally administer their own *file system cache*. This cache is located in the main memory, where it competes for memory space with the SAP system and the database. If the cache is too large, it causes high paging rates, even though the physical main memory is more than large enough to accommodate both the SAP system and the database. SAP recommends reducing this cache to 7-10% of the physical memory.

UNIX

The operating system parameters for configuring the file system cache include `dbc_max_pct` for HP-UX, `ubc-maxpercent` for Digital UNIX, and `maxperm` for AIX.

Windows

To reduce the size of the file system cache for Microsoft Windows, call the network settings (symbol: Network) in the Control Panel of your Windows operating system. Select the tab Services, the service Server, and the Properties button. In the following screen, under the screen area Optimization, select the Maximize Throughput for Network Applications option, and confirm by clicking OK. You must reboot the computer to activate the file cache's new settings.

A main memory bottleneck creates excessive paging, which in turn requires more processor use and can lead to a CPU bottleneck. Removing the cause of excessive paging usually makes the CPU bottleneck disappear.

2.2.2 Identifying Read/Write (I/O) Problems

In the operating system monitor (Transaction ST06), you'll find in the analysis view SNAPSHOT • DISK, among other things, information on hard drive load and (if the operating system makes it available) information on the drives' wait and response times.

Hard drive monitor

By double-clicking a row in the hard drive monitor, you can display an overview of the average response times over the previous 24 hours for the selected hard drive. Table 2.2 lists the displayed fields and their significance.

Field	Explanation
DISK	Operating system name of the hard disk.
UTILIZATION (%)	Load on the hard disk (in %).
QUEUE LENGTH	Number of processes waiting for I/O operations.
WAIT TIME (MS)	Wait time (in ms).
SERVICE TIME (MS)	Service time (in ms).
TRANSFER (KBYTE/S)	Transfer rate (in Kb/second).
OPERATIONS (PER SEC)	Number of I/O operations (per second).
RESPONSE TIME (MS)	Average response times of the hard disk (in ms).

Table 2.2 Fields of the Hard Disk Monitor

I/O bottleneck

If you determine via these monitors that individual drives are heavily loaded (LOAD (%) > 50 %), a potential *I/O bottleneck* exists. However, you can gain only limited information about I/O problems from the SAP system. To perform a more detailed analysis, you need tools provided by the hardware manufacturer.

An I/O bottleneck is particularly critical if it is on the hard drive on which the operating system's paging file resides. The operating system monitor is recommended particularly for the database server. To prevent bottlenecks during read or write operations to the database, use the database performance monitor and the hard drive monitor. For further details on these problems, please see Section 2.3.4, Identifying Read/Write (I/O) Problems.

2.2.3 Further Analysis at the Operating System Level

Parameter changes

For UNIX operating systems, the SAP system logs all operating system parameter changes. The change log can be displayed via the following path in the operating system monitor: OTHER FUNCTIONS • PARAMETER CHANGES. Place the cursor over the name of a server and click the HISTORY OF FILE button. This log lets you determine whether the start of performance problems can be linked to the time when particular parameters were changed.

Network check

With the OTHER FUNCTIONS • LAN CHECK BY PING tool, you can carry out a quick test on the network. Select any database server, application server, or presentation server and test the network connection (for example, response times or whether there was any data loss). Although the analysis is incorrectly called LAN check, you can also address computers in WAN. You can find an example of an analysis with this tool in Chapter 8, Section 8.1.2, Analyzing and Optimizing the Performance of GUI Communication.

Summary

Performance problems may be indicated if at least one of the following items are true:

- The average idle CPU capacity is less than 20% every hour.
- More than 20% of the physical main memory is paged every hour.
- Utilization of individual hard drives is more than 50%.

Excessive utilization of the hard drives, particularly on the database server, can cause system-wide performance problems. To check whether the high CPU load or paging rate significantly damages response times in the SAP system or database, use the workload monitor (see Chapter 3, Section 3.4, Performing Workload Analyses).

Analyzing hardware bottleneck

Figure 2.3 and Figure 2.4 show the procedure for analyzing a hardware bottleneck: a common solution for resolving bottlenecks is to redistribute the workload (for example, move work processes). Possible causes of a CPU bottleneck include inefficient applications, which can usually be identified in the database process monitor and work process overview, and external processes that do not belong to an SAP instance or the database instance. You should always perform a complete performance analysis before deciding whether the existing hardware is sufficient for SAP system demands.

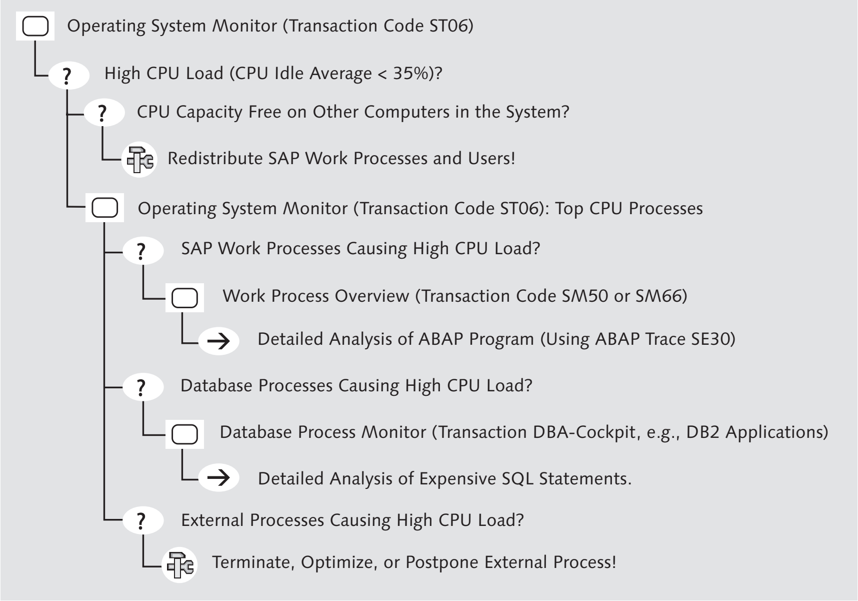

Operating System Monitor (Transaction Code ST06)
?
High CPU Load (CPU Idle Average < 35%)?
?
CPU Capacity Free on Other Computers in the System?
Redistribute SAP Work Processes and Users!
Operating System Monitor (Transaction Code ST06): Top CPU Processes
?
SAP Work Processes Causing High CPU Load?
Work Process Overview (Transaction Code SM50 or SM66)
Detailed Analysis of ABAP Program (Using ABAP Trace SE30)
?
Database Processes Causing High CPU Load?
Database Process Monitor (Transaction DBA-Cockpit, e.g., DB2 Applications)
Detailed Analysis of Expensive SQL Statements.
?
External Processes Causing High CPU Load?
Terminate, Optimize, or Postpone External Process!

Figure 2.3 Detailed Analysis of a Hardware Bottleneck (CPU)

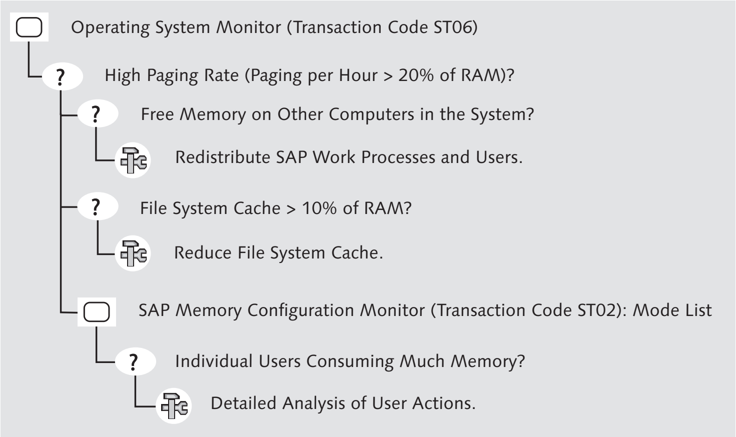

Operating System Monitor (Transaction Code ST06)
?
High Paging Rate (Paging per Hour > 20% of RAM)?
?
Free Memory on Other Computers in the System?
Redistribute SAP Work Processes and Users.
?
File System Cache > 10% of RAM?
Reduce File System Cache.
SAP Memory Configuration Monitor (Transaction Code ST02): Mode List
?
Individual Users Consuming Much Memory?
Detailed Analysis of User Actions.

Figure 2.4 Detailed Analysis of a Hardware Bottleneck (Main Memory)

The roadmaps in Figure 2.3 and Figure 2.4 show the procedure to follow in the event of a hardware bottleneck. They refer to monitors and analyses described later in this book. You will find similar roadmaps throughout this book.

2.3 Database Monitoring

SAP NetWeaver Application Server (AS) can currently be operated with eight different relational database systems, as well as with the main memory database SAP HANA. Even if these database systems all have a different architecture, performance problems can still occur independently of the database system used. To help customers analyze and tune their databases, SAP NetWeaver AS ABAP has its own database monitor with basic functions that work independently of the database system used.

In this section, we present the steps of database performance analysis based on an Oracle database. In Appendix A, you will find information and notes on the other databases, as well as more details on Oracle databases. Chapter 15, Optimizing Database Queries with SAP HANA, deals with the SAP HANA main memory database.

Basic functions of the database monitor

The most important functions that you require for performance monitoring of the database include the following:

- Overview of the database buffers' status
- Overview of the currently running database operations, particularly the SQL statements
- Statistics on the executed SQL statements and their costs
- Overview of the distribution of read and write accesses at the logical (*tablespaces*) and physical level (for example, *container* for DB2 for Linux, UNIX, and Windows or *datafiles* for Oracle) and thus about the hard drives
- Overview of the current lock situations

Locks on database tables or business objects are a prerequisite for consistent data. If locks are held too long, performance problems may occur in the system because users and processes have to wait for the locks to be released. This chapter presents the lock concepts and their monitoring in detail.

9 Locks

In an SAP system, many users can simultaneously read the contents of database tables. For changes to the dataset, however, you must ensure that only one user can change a particular table's content at a time. For this purpose, table content is locked during a change operation. The first section of this chapter introduces you to the concept of locking for SAP and database systems.

If locks remain in place for a long time, wait situations can occur, limiting the throughput of the SAP system. The second section of this chapter deals with the general performance aspects of using locks.

The SAP system uses special buffering techniques for availability checking with Available to Promise (ATP) logic and for document number assignment, which minimize the lock time and maximize the throughput. These techniques will be discussed in the third and fourth sections.

When Should You Read This Chapter?

You should read this chapter to help you do the following:

- Find out more about database locks and SAP enqueues
- Analyze system problems that are caused by database locks or enqueues

This chapter does not offer instruction on programming SAP transactions. Use ABAP textbooks or SAP Help for this.

9.1 Lock Concepts of Database System and SAP System

Data consistency

The consistency of the data in a database or SAP system is achieved by locks. The lock concepts of SAP and database systems have the same ultimate purpose of preserving data consistency, but they are based on different technologies and used in different situations. Locks that the database system manages are known as *database locks*, and locks that the SAP system manages are known as *SAP enqueues.*

[Ex]

Ordering a Computer

When you configure and order a new computer, you must check that all necessary components are available—for example, housing, CPU, main memory, hard drive, and so on. The "all-or-nothing" principle applies: when a component is sold out, the availability of the entire computer cannot be confirmed. Since you usually check the availability of the various components in succession, you want to be certain that other users do not access the already checked and approved components until the whole order has been finally confirmed or canceled.

9.1.1 Database Locks

Database locks are managed by the lock handler of a database instance. The locked entity is typically a row in a database table (special exceptions are explained at the end of Section 9.2.1, Database Locks). Database locks are set by modifying SQL statements (`UPDATE`, `INSERT`, or `DELETE`) and by the statement `SELECT FOR UPDATE`. Locks are held until the SQL statement `COMMIT` (database commit) finalizes all changes in the database and then removes the corresponding database locks. The time interval between two commits is called a *database transaction*. A program can undo the effects of all modifying SQL statements by executing a database rollback with the SQL statement `ROLLBACK`. In this case, all database locks are also removed.

[Ex]

Ordering a Computer (Continuation)

The aforementioned example of the availability check when ordering a computer using database locks is achieved in programming with the SQL statement `SELECT FOR UPDATE`: a particular item of travel data is read and locked with this statement. When this check has been successfully performed for all components, the data is then changed (with an `UPDATE` in the relevant table rows), and then the `COMMIT` command is used to finalize changes and release all locks. Once a lock has been set, other users can still read the affected data (a simple `SELECT` is still possible), but they cannot lock it. Neither an `UPDATE` nor a `SELECT FOR UPDATE` can be performed. Such a lock is known as *exclusive*.

After a transaction step, the SAP work process automatically triggers a database commit (or a database rollback). This removes all database locks. This means a database lock is not held during multiple transaction steps (via multiple input screens in the SAP system).

9.1.2 SAP Enqueues

To hold locks during multiple steps of an SAP transaction, use SAP enqueue administration. Work processes in the enqueue table, located in the main memory, manage SAP enqueues. To retain these enqueues even when an SAP instance is shut down, save them in a local file on the enqueue server.

SAP enqueue objects

An SAP enqueue locks a logical object. Therefore, an enqueue can lock rows from several different database tables if these rows form the basis of a single business document. An SAP enqueue can also lock one or more complete tables. SAP enqueue objects are created and modified in the ABAP dictionary (dictionary section Lock Objects). They are closely related to the concepts of *SAP transaction* and *SAP logical unit of work* (SAP LUW). Both of these are described extensively in the ABAP literature for dialog programming. Therefore, this chapter will not discuss the functions and uses of these techniques as part of ABAP programs. Rather, we will focus on aspects related to performance analysis. If you discover performance problems caused by the incorrect use of SAP enqueues, consult the responsible ABAP developer.

An SAP enqueue is a logical lock that acts within the SAP system. If a row in a database table is locked by an SAP enqueue, it can still be changed by an SQL statement executed from the database or by a customer-developed ABAP program that does not conform to SAP enqueue conventions. Therefore, SAP enqueues are valid only within the SAP system. Database locks, in contrast, resist all change attempts. They lock a table row "tight" for all database users and prevent changes by users outside the SAP system.

Function modules

For each object that can be held by an enqueue, there are two function modules: an enqueue module and a dequeue module. An SAP enqueue is set explicitly within an ABAP program by an enqueue function and is explicitly released by a dequeue function module. As a result, SAP enqueues can be held in place through multiple transaction steps. When an SAP transaction is completed, all SAP enqueues are automatically removed.

[Ex]

Ordering a Computer (Continuation)

Using our example of computer configuration and ordering, we will explain how SAP enqueue management works. A computer consists, for example, of the housing, CPU, main memory and hard drive. The individual components are edited on different input screens—that is, with several transaction steps—and are locked for editing by SAP enqueues. After determining the availability of each component, you can confirm the order for the entire computer. This concludes the dialog part of the transaction.

Under the protection of the enqueues, an update work process then transfers the changes to the database tables. When the update has been completed, the SAP LUW is finished, and the enqueues are unlocked.

An SAP LUW can also contain program modules that require a V2 update. An SAP enqueue is not used for this. You should not use modules that use this V2 update to process data that requires the protection of enqueues (also see Chapter 7, Section 7.1.8, Update).

Table 9.1 compares the main features of database locks and SAP enqueues.

	DB Locks	SAP Locks (Enqueues)
Locked object	Individual rows of a database table	Logical object (for example, a document) defined in the ABAP Dictionary
How object is locked	Implicitly, using modifying SQL statements (such as `UPDATE` and `SELECT FOR UPDATE`)	Explicitly, by the ABAP program calling an enqueue module
How lock is removed	Implicitly, with the SQL statement `COMMIT` or `ROLLBACK` Usually at the end of a transaction step	Explicitly, by calling a dequeue module Usually at the end of an SAP transaction
Maximum duration	One transaction step	Over multiple transaction steps
Result of lock conflicts	Wait situation (exclusive lock wait, also see section 9.2.1, Database Locks)	Program-specific, for example, the error message "Material X is locked"
How to monitor	Transaction DB01, Exclusive Lock Waits	Transaction SM12, Enqueue Monitor

Table 9.1 Features of Database Locks and SAP Enqueues

Table 9.1 Features of Database Locks and SAP Enqueues (Cont.)

9.2 Monitoring Locks

In this section, you will find information on how to monitor database locks and SAP enqueues.

9.2.1 Database Locks

Exclusive lock waits

What happens in the event of a lock conflict—that is, when a work process wants to lock an object that is already locked? With database locks, the second process waits until the lock has been removed. This wait situation is known as an *exclusive lock wait*. Most databases do not place a time limit on these locks. If a program fails to remove a lock, the wait situation can continue indefinitely.

This could become a major problem if the program fails to release a lock on critical SAP system data, such as the number range table `NRIV`. There is a danger that one work process after another will be waiting for this lock. If all work processes are waiting, no work process is available to allow you to intervene from within the SAP system. If you can identify the program holding the problem lock, you can terminate it through the operating system as a last alternative.

Checking lock wait situation

To monitor current lock wait situations, call the database-lock monitor (Transaction DB01), which you can start from the DBA Cockpit (Transaction DBACOCKPIT) by selecting Performance • Wait situations on Locks and Deadlocks or from the system-wide work process overview (Transaction SM66) by selecting GoTo • DB Locks.

For a description of this monitor and information on how to troubleshoot lock wait situations, see Chapter 2, Section 2.3.5, Other Database Checks. Lock wait situations increase database time and result in high database times in workload-monitor statistics. Some database systems explicitly monitor lock wait times, which you can view in the database performance monitor.

[Ex]

Lock Situation in the Database

With the following sample program, you can provoke a lock situation in the database:

```
REPORT zts_lock.
DATA: lv_text type natxt.
SELECT SINGLE FOR UPDATE text FROM T100 INTO lv_text WHERE
sprsl = 'DE' AND ARBGB = '00' AND msgnr = '001' .
BREAK-POINT.
```

To do so, proceed as follows:

1. Start the program in the ABAP Workbench (for instance, via Transaction SE38). After a few seconds, the debugger opens, and the program stops at the `BREAK-POINT` command. Before that, the program has set a database lock using the `SELECT SINGLE FOR UPDATE` command. Since the program waits in the debug mode, this database lock is not undone.
2. Open a second session and restart the program. An hourglass is displayed in the second session.
3. Again, you can restart the program in a third session, and the system again displays an hourglass.
4. Open another session and start the database lock monitor as previously described. The lock situation is displayed, and you can see which work process holds the lock and which one waits. Based on the work process overview (Transaction SM50) and the database process monitor, you can now analyze what happens in the process that holds the lock. In this example, the process overview displays the Stopped status and gives Debug as the reason.

5. Go to the debugger, where you continue the execution of the program in the first session. The program is terminated, and the database lock is undone due to an implicit commit or rollback of the database interface. As a result, the program can continue in the second session, which had to wait at the `SELECT SINGLE FOR UPDATE` command up to now. Within a very short period of time, it will thus reach the `BREAK-POINT` command and start the debugger.
6. Continue the program in the debugger for the second session and, if you started the program in further sessions, in these sessions to release the locks.

Typical problems

Basically, you should set programs to request locks as late as possible. It is preferable for a program to read and process data from the database before setting locks or making changes in the database. This is illustrated in Figure 9.1. The top part of the diagram shows how several changes are made during a database transaction and how, as a result, database locks are held for too long. The lower part of the diagram shows a more appropriate programming method: the transaction is programmed so that it collects the changes in an internal table and then transfers these changes to the database as a group at the end of the transaction. This reduces the lock time in a database.

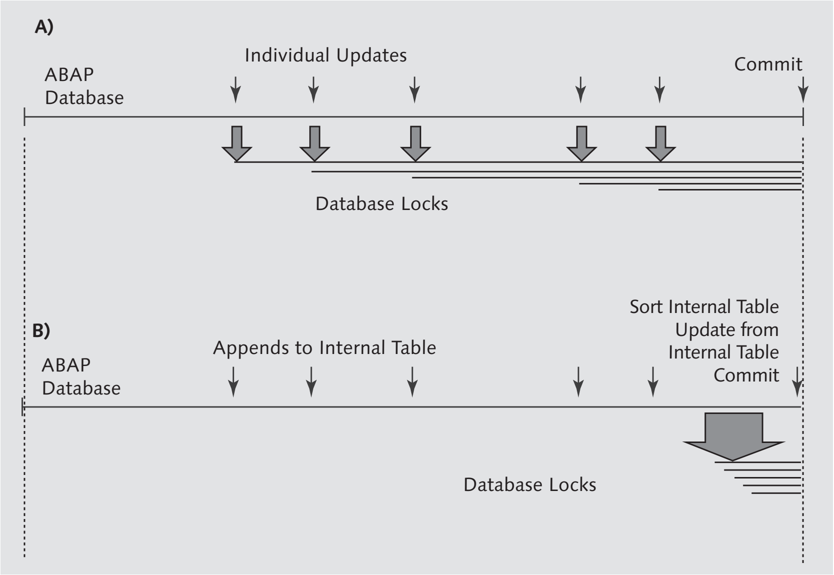


Figure 9.1 Locks Should Be Set as Late as Possible

Causes

Performance problems due to delays in releasing locks frequently occur when customers modify the programming of update modules. The separation of update modules from dialog modules is an attempt to reduce the number of locks needed in the dialog part of a transaction because changes to the database and the associated locks are mainly the task of the update modules. However, sometimes the update module is modified—for example, to supply a customer-developed interface with data. This modification may cause problems if the update module has already set locks and, for example, the modification generates expensive SQL statements. The locks cannot be released until the SQL statements are fully processed, and lengthy lock waits may result.

Another source of problems with locks is background programs that set locks and then run for several hours without initiating a database commit. If dialog transactions need to process the locked objects, they will be forced to wait until the background program finishes or initiates a database commit. To solve this problem, you should ensure that the background program either initiates a database commit at regular intervals (without sacrificing data consistency) or runs only when it will not interfere with dialog processing. Similar problems may occur when background jobs are run in parallel—that is, when a program is started several times simultaneously. Parallel processing is recommended only when the selection conditions of the programs do not lock the same data.

While you are working in the ABAP debugger, database commits are generally not initiated, and all locks stay in place until you are finished. You should therefore avoid using the debugger in a production SAP system.

[Ex]

Deadlocks

We'll now present an example of a situation known as a *deadlock*. Assume that work process one and work process two both want to lock a list of materials. Work process one locks material A, and work process two locks material B. Then, work process one tries to lock material B, and work process two tries to lock material A. Neither work process is successful because the materials already have locks on them. The work processes block each other. A deadlock is identified by the database instance and solved by sending an error message to one of the work processes. The corresponding ABAP program is terminated, and the error is logged in the SAP syslog.

You can avoid deadlocks with correct programming. In our example, the program should be changed so that its internal material list is sorted before any locks are set. Then the lock on material A will always be set before the lock on material B. Therefore, programs requiring the same materials are serialized and not deadlocked.

Deadlocks should occur very rarely. Frequent deadlocks indicate incorrect programming or configuration of the database instance.

Table locks

If, in some database systems (for example, DB2 and SAP MaxDB), a work process places single-row locks on more than 10% of the single rows in a table, the locks are automatically replaced by table locks. Here, the database decides that it is more efficient to lock the entire table for a work process than to maintain several locks on individual rows. Table locking has consequences for parallel processing in background jobs, where each program is intended to update a different part of the same table at the same time. It is not possible to schedule background jobs so that one updates the first half of the table and the other updates the second half because the database may decide to lock the table exclusively for one of the jobs. One program that is particularly affected by this is the period closing program in materials management.

[«]

Table Locks

There are database parameters you can use to specify when the database should convert single-row locks to a table lock.

Sometimes, the database locks entire tables for administrative reasons. This happens when indexes are created or when particular tables and indexes are analyzed—for example, during the Oracle analysis `VALIDATE STRUCTURE`. If these actions are performed during production operation, substantial performance problems may result.

9.2.2 SAP Enqueues

SAP enqueues are managed in the enqueue table located in the global main memory of the enqueue server. The work processes in the enqueue server directly access the enqueue table; the enqueue server also carries out lock operations for work processes from other application servers, which are communicated via the message service (❶ and ❷ in Figure 9.2).

The following abbreviations are used in Figure 9.2: DIA for dialog work process, ENQ for enqueue work process, and ENQ tab for enqueue table.

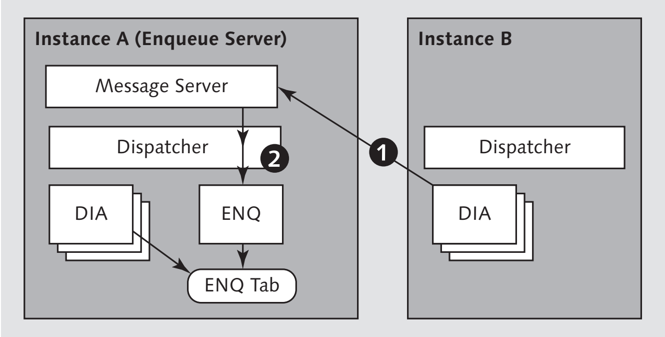


Figure 9.2 Communication for Setting and Removing SAP Enqueues

For work processes in the enqueue server, setting and releasing locks takes less than 1 millisecond; for work processes in other application servers, it takes less than 100 milliseconds.

Performance problems with SAP enqueues

If an SAP enqueue is requested but already held by another user, the attempt to set a lock is rejected and an error message is sent back to the ABAP program. The application developer has to decide how to deal with this error message with suitable programming. For programs in dialog mode, the error message is normally forwarded to the user, for example, with the message "Material X is locked by user Y." For background programs, you will normally attempt to set the lock again later. After a certain number of unsuccessful attempts, an error message is written to the program log.

If SAP enqueues are held for too long, performance problems can arise because after a failed attempt, the user will repeat the entry. Take, for example, a user who needs to process a material list, and to do, so needs to set 100 SAP enqueues. If the attempt to set lock number 99 fails, the program is interrupted with the message "Material number 99 is locked," and all of the previous system work is in vain and must be repeated. Therefore, rejected enqueue requests lead to higher system workload and restrict the throughput of transactions.

You can get an overview of all currently active SAP enqueues by using Transaction SM12, under the following menu:

Tools • Administration • Monitor • Lock Entries

Start the test programs under the following menu to diagnose errors:

Extras • Diagnose or Extras • Diagnose in VB

If errors are identified, check the SAP Service Marketplace for notes, or contact SAP directly.

Enqueue statistics

You can view statistics on the activity of the enqueue server with the menu option Extras • Statistics. The first three values show the number of enqueue requests, the number of rejected requests (unsuccessful because the lock requested was already held by another), and the number of errors that occurred during the processing of enqueue requests. The number of unsuccessful requests should not be more than 1% of the total number of enqueue requests. There should be no errors.

9.3 Number Range Buffering

With many database structures, it is necessary to be able to directly access individual database records. You do this with a unique key. *Number ranges* assign a serial number that forms the main part of this key. Examples of these numbers include order numbers and material master numbers. SAP number range management monitors the number status so that previously assigned numbers are not re-issued.

9.3.1 Fundamentals

A business object for which a partial key must be created using the number range is defined in the SAP system as a *number range object*. A number range contains a *number range interval* with a set of permitted characters. The number range interval is made up of numerical or alphanumeric characters and is limited by the From-Number and To-Number fields. You can assign one or more intervals to a number range.

Technical implementation

The current number level of a range, which is the number that is to be assigned next, is stored in the database table NRIV. If a program needs

Index

First-hand knowledge.

SAP PRESS
7th Edition
Updated for SAP NetWeaver 7.3
SAP Performance Optimization Guide
Analyzing and Tuning SAP Systems
Make your SAP system run quickly and efficiently
Master core concepts like sizing, memory management, and database monitoring
Explore new topics such as SAP HANA, SAP Sybase ASE, and innovations in SAP NetWeaver AS Java
Thomas Schneider
Galileo Press
SAP Performance

Dr. Thomas Schneider started his career at SAP AG in 1996. Among other things, he was in charge of the Center of Expertise for Performance in the Service & Support group and the responsible Support Alliance manager for key accounts. Since 2004 he has worked in the Research & Breakthrough Innovation group, where he first was responsible for IT Service & Application Management. Since 2009 he has focused on the Partner Development Infrastructure in the SAP Cloud organization as the responsible architect.

Thomas Schneider

SAP Performance Optimization Guide

837 Pages, 2013, $79.95
ISBN 978-1-59229-874-7

www.sap-press.com/3360