

Database Manager DBMGUI (BC)



Release 4.6C



Copyright

© Copyright 2001 SAP AG. All rights reserved.

No part of this publication may be reproduced or transmitted in any form or for any purpose without the express permission of SAP AG. The information contained herein may be changed without prior notice.

Some software products marketed by SAP AG and its distributors contain proprietary software components of other software vendors.

Microsoft®, WINDOWS®, NT®, EXCEL®, Word®, PowerPoint® and SQL Server® are registered trademarks of Microsoft Corporation.

IBM®, DB2®, OS/2®, DB2/6000®, Parallel Sysplex®, MVS/ESA®, RS/6000®, AIX®, S/390®, AS/400®, OS/390®, and OS/400® are registered trademarks of IBM Corporation.

ORACLE® is a registered trademark of ORACLE Corporation.

INFORMIX®-OnLine for SAP and Informix® Dynamic Server™ are registered trademarks of Informix Software Incorporated.

UNIX®, X/Open®, OSF/1®, and Motif® are registered trademarks of the Open Group.






HTML, DHTML, XML, XHTML are trademarks or registered trademarks of W3C®, World Wide Web Consortium, Massachusetts Institute of Technology.

JAVA® is a registered trademark of Sun Microsystems, Inc.

JAVASCRIPT® is a registered trademark of Sun Microsystems, Inc., used under license for technology invented and implemented by Netscape.

SAP, SAP Logo, R/2, RIVA, R/3, ABAP, SAP ArchiveLink, SAP Business Workflow, WebFlow, SAP EarlyWatch, BAPI, SAPPHIRE, Management Cockpit, mySAP.com Logo and mySAP.com are trademarks or registered trademarks of SAP AG in Germany and in several other countries all over the world. All other products mentioned are trademarks or registered trademarks of their respective companies.

Icons

Icon	Meaning
	Caution
	Example
	Note
	Recommendation
	Syntax

Contents

Database Manager DBMGUI (BC)	8
Introduction	10
Overview	11
How the DBMGUI Works	12
DBM Server	13
Serverdb Architecture	14
System Devspace	15
Data devspace	16
Log devspace	17
Concepts	18
User Management	19
DBM User	20
Database Administrator (DBA)	21
DOMAIN User	22
Operator	23
Changing User Passwords	24
Assigning Rights as the Operating System User	25
Assigning Server Rights	26
Data Backup	27
Data Backups Without Checkpoint	29
Data Backups with Checkpoint	30
Log Backup	31
Automatic Log Backup	32
Interactive Log Backup	33
Restartability	34
Process Structure	35
User Kernel Process/User Kernel Thread	36
User Tasks	37
Server Tasks	38
Logwriter	39
Datawriter	40
Utility Task	41
Tracewriter Task	42
Conv Scanner	43
Timer Task	44
Special Processes/Threads	45
Requestor	46
Dev Processes/Threads	47
Coordinator	48
Timer	49
Temporary Dev Processes/Threads	50
Operating-System-Dependent Special Processes/Threads	51
Clock Thread	52
Console Process	53

Console Thread	54
Caches.....	55
Catalog Cache	56
Converter Cache	57
Free Blocks Management (FBM) Cache.....	58
File Directory	59
Data Cache	60
Log Cache.....	61
Savepoint	62
Checkpoint.....	63
Multiprocessor Configurations	64
Availability.....	65
Screen Areas of the Database Manager (DBMGUI)	66
Menu Bar.....	68
Icon Bar	69
Directory of Registered Database Instances	70
List of Registered Database Instances	71
The Menu List of the Current Database Instance	72
Output Area.....	73
Message Output Area	74
Name Window.....	75
New Database Instance Installation	76
Configuring the Database Parameters.....	78
Displaying and Changing Current Database Parameters	79
Database Parameters	80
General - General Database Parameters.....	81
MAXBACKUPDEVS	82
MAXCPU	83
MAXDATAPAGES.....	84
MAXLOCKS.....	85
MAXUSERTASKS	86
LOG_MODE	87
LOG_SEGMENT_SIZE	88
RESERVED_REDO_SIZE	89
RUNDIRECTORY.....	90
Extended - Special Database Parameters	91
DEADLOCK_DETECTION.....	93
REQUEST_TIMEOUT	94
SESSION_TIMEOUT	95
Devspaces.....	96
Devspace Configuration.....	97
Creating Devspaces.....	98
Devspace Change	99
Updating the System Tables.....	100
Terminal Support (Termchar Set)	101
Displaying, Changing and Deleting Termchar Sets.....	102
Defining Termchar Sets	103
Language Support (MapChar Sets).....	104

Displaying, Changing and Deleting MapChar Sets	105
Defining Mapchar Sets.....	106
Database Instance Administration	107
Activating and Deactivating Indices.....	108
Updating the Statistics Information	109
Checking the Consistency of the Devspaces.....	111
Database Instance Registration.....	112
Starting (Restarting) the Database Instance	113
Stopping the Database Instance (Shutdown).....	114
Monitoring the Mode	115
Backups.....	116
Backup Concepts	117
Backup Media	118
Backup Label	119
Managing the Backup Media	120
Defining Media	121
Defining a Single Backup Medium.....	122
Defining a Group of Parallel Media.....	123
Changing Media Definitions	125
Deleting Media Definitions	126
Backup Processes	127
Saving to a Single Medium	128
Saving to a Group of Parallel Media	130
Activating and Deactivating the Automatic Log Backup	131
Backups to Automatic Tape Loader.....	132
Backing Up to Manually Changed Media.....	133
Restore	134
Restore to the Last Complete Backup Made	135
Restore to a Backup from the Backup History	136
Restoring an Existing Database Instance	137
Restoring the Indices After a Database Restore.....	139
Restoring Without a Backup History	140
Recovering a Mirrored Devspace	141
Restore from Automatic Tape Loader.....	143
Continuing an Interrupted Restore	144
Requesting Information	145
Backup History	146
Cache Information	147
Data Devspaces.....	148
Read and Write Operations.....	149
Log Devspace and Logwriter	150
Locks and Lock Requests	151
User Sessions	152
Versions	153
Options for Diagnosing Problems	154
Diagnosis with Kernel Trace Function	155
Reading the Log Files of the Database Manager	156
Database Console Information	157

Use of the Command Line Version of the Program	158
Database Icon Legend	159
Updating the Database Software	160

Database Manager DBMGUI (BC)

The Database Manager consists of a server and a client part. The server part ([DBM server \[Page 13\]](#)) is responsible for functionality. The client part consists of a graphical user interface, called the Database manager (DBMGUI), and a command line version, called the Database Manager (DBMCLI).

See also:

R/3 Database Manager (DBMCLI)

Implementation Considerations

Guides containing information about the installation of the SAP DB database system with the R/3 System:

- *R/3 Installation on UNIX: SAP DB*
- *R/3 Installation on Windows NT: SAP DB*

You will find information on how to make a copy of your database system in the following:

- *R/3 Homogeneous System Copy*

Integration

Various DBA functions in the R/3 System's Computing Center Management System (CCMS) help you with your database management work. You can schedule data and log backups, check the consistency of the database system, and update the statistics needed by the Optimizer.

Further information is available in the following R/3 online documentation:

- *BC – Computing Center Management System; under SAP DB – DBA in the CCMS*

Further R/3 Online Documentation on the SAP DB Database System

- *BC - Messages: SAP DB*
- *BC - Reference Manual: SAP DB (7.2)*
- *BC - SQL Studio: SAP DB*
- *BC - Replication Manager: SAP DB*
- *BC - SAP High Availability*

Information on SAP DB / ADABAS for R/3 is also available in SAPNet:

- *UNIX Manual*
- *Windows NT Manual*
- *ODBC Manual*
- *QUERY*
- *LOAD*
- *C/C++ Precompiler*

You can use the **CONTROL** program to manage databases of type ADABAS for R/3 up to Version 6.2.10.

- *R/3 Database Management (CONTROL)*



Use the **Database Manager** program for databases of type SAP DB, Version 7.1.4 and later.

The **Database Manager (DBMGUI)** program is described here.

Introduction

Introduction

The **Database Manager (DBMGUI)** is a database management tool.

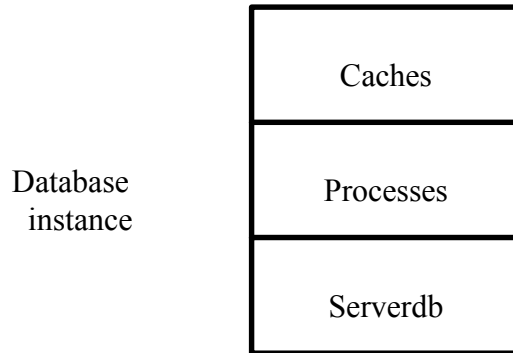
The Database Manager (DBMGUI) is used to control and monitor the database instance and the execution of backups.

The Database Manager (DBMGUI) allows you to access remote database servers.

Overview

One or more instances of the database can be installed and operated on a computer.

Each database instance consists of processes, main memory structures (caches), and a disk-based database.



[How the DBMGUI Works \[Page 12\]](#)

[Architecture of a Serverdb \[Page 14\]](#)

[Concepts \[Page 18\]](#)

[Installation of a New Database Instance \[Page 76\]](#)

[Administration of Database Instances \[Page 107\]](#)

How the DBMGUI Works

How the DBMGUI Works

The Database Manager (DBMGUI) is an easy-to-use tool that can manage any number of local or remote database instances.

The Graphical User Interface (GUI) of the Database Manager communicates with the [DBM Server \[Page 13\]](#) of the current database instance and enables users to use the DBM server functions.

DBM Server

The DBM server is the server part of the Database Manager. It is installed during the installation of the server on the database computer.

Client applications such as the [Database Manager \(DBMGUI\) \[Page 1\]](#) or the Database Manager (DBMGUI) form a link to the DBM server and exchange data with the DBM server using a request-reply mechanism.

Serverdb Architecture

Serverdb Architecture

A Serverdb consists of

- the [system devspace \[Page 15\]](#) and
- one or more [log devspaces \[Page 17\]](#) and
- one or more [data devspaces \[Page 16\]](#)

The term "devspace" denotes a physical disk or part of a physical disk. This can be a raw device or a file.

For security and performance reasons, each devspace type should be kept on a different disk. The log devspaces of a serverdb can also be mirrored to obtain a higher degree of [availability \[Page 65\]](#).

The disks used should present uniform performance data (especially access speeds) because this is the only way that equal usage of the devspaces can be achieved.

If necessary, a database instance can be expanded by additional data devspaces while the database is running.



The devspace usage level of a database instance is therefore a critical parameter of database operation and must be monitored.

If the data devspaces become full, database operation stops. Further data devspaces can be defined in this state to allow database operation to continue.

System Devspace

The restart information and the mapping of the logical page numbers to physical page addresses are administered in the system devspace.

The size of the system devspace therefore depends directly on the database size and is determined by the database kernel.

Data devspace

Data devspace

The user data (tables, indexes) and the SQL catalog are stored in the data devspaces. A table or an index needs one page (minimum); a table can use all the data devspaces (that is the whole database) (maximum).

A table increases or decreases in size automatically without administrative intervention.

As a rule, an ADABAS internal striping algorithm distributes the data belonging to a table evenly across all the data devspaces.



An assignment of tables to data devspaces is not possible nor is it necessary.

When [installing the database instance \[Page 76\]](#) you can [configure one or more data devspaces \[Page 97\]](#) and while the database is running you can also add new data devspaces ([Devspace Definition \[Page 98\]](#)). The disk storage space defined by all the data devspaces is the total size of the database.

Log devspace

What is recorded in a log devspace is all the changes in the contents of the database, to enable the contents to be [recovered \[Page 134\]](#) or restored after hardware faults.

The complete log can consist of a number of devspaces. You can define the number of log devspaces required when installing the database instance and can add new log devspaces even while the database is operating ([Devspace Configuration \[Page 97\]](#)).

To ensure that the data on the database is kept safe, you have the option of mirroring the log devspace(s) (set parameter [LOG_MODE \[Page 87\]](#) to DUAL).

In [log backups \[Page 31\]](#) the contents of the log devspace(s) is copied to a file and the space originally occupied by it is released for log data. The backup files are numbered by the system in sequence.



The selected size of the archive log devspace should therefore be sufficient for all the changes occurring between two backups to be recorded there.

Concepts

Concepts

[User Management \[Page 19\]](#)

[Data backups \[Page 27\]](#)

[Log backups \[Page 31\]](#)

[Restartability \[Page 34\]](#)

[Process structure \[Page 35\]](#)

[Caches \[Page 55\]](#)

[Savepoint \[Page 62\]](#)

[Checkpoint \[Page 63\]](#)

[Multiprocessor Configurations \[Page 64\]](#)

[Availability \[Page 65\]](#)

[The Database Manager \(DBMGUI\) screen \[Page 66\]](#)

User Management

The Database Manager uses various classes of users:

[DBM users \[Page 20\]](#)

[Database Administrators \(DBA\) \[Page 21\]](#)

[DOMAIN Users \[Page 22\]](#)

[Operator \[Page 23\]](#)

When you install the database instance, you are asked to create the DBM user and database administrator (DBA) by specifying a name and password. The system stores the DBA's password for the DOMAIN user.

The user "Operator" is defined by the system at the time of installation.

You can [change these users' passwords \[Page 24\]](#) at a later date.

DBM User

DBM User

The DBM user is the user identification defined when [installing a new database instance \[Page 76\]](#).

To be able to access this database instance later, you have to [register \[Page 112\]](#) it under the name and password of the DBM user in the Database Manager.

You can change DBM user's password (See: [Changing user passwords \[Page 24\]](#)).

The DBM user is responsible for controlling and monitoring the system and for backup operations. The DBM user is authorized to carry out all Database Manager (DBMCLI) functions, even when the database instance is in the `COLD` operational state.

A DBM user can also log on to the Database Manager (DBMGUI) more than once, so the DBM user can for example query operating parameters while functions that take a long time are still running.

Database Administrator (DBA)

The database administrator (DBA user) sets up the system and defines the users. The database administrator owns the system tables and is authorized to define other administrators.

The DBA user is called for particularly for installation work.

You can change DBA user's password (see: [Changing user passwords \[Page 24\]](#)).

DOMAIN User

DOMAIN User

The DOMAIN user is the owner of the catalog tables. The user is created during installation.

You can change the DOMAIN user's password (See: [Changing user passwords \[Page 24\]](#)).

Operator

The Operator is a database user with very limited powers. The operator is authorized only to make [backups \[Page 116\]](#).

You can change the operator's password (See: [Changing user passwords \[Page 24\]](#)).

Changing User Passwords

Changing User Passwords

There are different modes of the database instance in which the passwords for the various classes of user can be changed.

Users	Mode of database instance
DBM [Page 20]	COLD or OFFLINE
DBA [Page 21]	WARM
DOMAIN [Page 22]	WARM
Operator [Page 23]	WARM or COLD

Prerequisites

You have put the database instance into the appropriate mode ([Starting the Database Instance \[Page 113\]](#), [Stopping the Database Instance \[Page 114\]](#)).

1. Choose *Instance* → *Configuration* → *Users*.
2. Select the required user and then choose *Users* → *Edit*.
3. Select the tab page *General* and then choose *Change Password*.
4. Enter the old and new passwords and save your entries.

Assigning Rights as the Operating System User

Use

In order to allow a DBM user to administrate database instances remotely, he or she can also be granted rights as an operating system user for the servers in these database instances.

Prerequisites

You are an operating system user on the database computer and also have DBM user rights for the database instance on this computer.

Procedure

Choose *Instance* → *Configuration* → *Users*.

Select the DBM user and choose *Users* → *Edit*.

Under *System Account*, enter the name and password of the operating system user. Enter the password again.

Choose *OK*.

Result

The DBM user also has operating system user rights on the computer of the current database instance.

Assigning Server Rights

Assigning Server Rights

Use

Prerequisites

Procedure

Result

Data Backup

In the **WARM** and **COLD** database modes, the Database Manager (DBMGUI) provides the following kinds of interactive backup:

Complete [data backup with checkpoint \[Page 30\]](#)

Complete [data backup without checkpoint \[Page 29\]](#)

Incremental data backups with checkpoint

Incremental data backups without checkpoint



Before you initiate a manual backup to tape, you should always read the [backup label \[Page 119\]](#) of the backup that may already be on the tape.

Data Backups in **WARM** Mode

In **WARM** mode, you can either perform complete or incremental data backups.

When you back up in warm mode, remember that the database state which is saved is the state when the backup operation was started.

To allow the **latest** state of the database to be recovered, the [log backups \[Page 31\]](#) which were later in time than the start of the backup operation will also need to be retrieved, irrespective of whether or not the backup was carried out with a [checkpoint \[Page 63\]](#).

Data Backups in **COLD** Mode

Complete and incremental data backups can also be carried out when the database is not running, that is in **COLD** mode.

Consistency of Data Backup

You can create a consistent data backup once the database instance has been set to the cold, consistent operating mode with → *Shutdown* → *Cold (not Offline)*. This consistent backup can then be used to [restore \[Page 134\]](#) the database instance without having to import additional log backups.

Consistent backups are not required for daily database operation. They are only necessary if the database is to be **migrated** or **copied**.



To check whether a backup is consistent, select *Instance* → *Backup* → *History*.

Check the entry in the *Log Required* column for the backup concerned.

NO: The data backup is consistent.

YES: The log backups made at times later than the data backup will need to be retrieved after the data backup is retrieved.



You must perform a complete backup at sensible time intervals (e.g. at least weekly).

Data Backup

A tape containing a complete data backup should therefore not be overwritten immediately with the next backup. If you retain, say, the last four backup generations, it may be possible to use an older backup if a media failure occurs.

Data Backups Without Checkpoint

Using the Database Manager you can make both complete and incremental data backups without [checkpoint \[Page 63\]](#).

In an incremental data backup it is only the pages that have been updated that are backed up.

A complete or incremental [backup \[Page 116\]](#) of data made without checkpoint only becomes consistent when the [log backups \[Page 31\]](#) made at times later than the start of the data backup have been retrieved. Consistent backups, however, are not required for daily database operation.

Dispensing with the checkpoint increases the speed of backups, particularly with periodic data backups.

Data Backups with Checkpoint

Data Backups with Checkpoint

Using the [Database Manager \[Page 1\]](#) you can make both complete and incremental data backups with [checkpoint \[Page 63\]](#).

In an incremental data backup it is only the pages that have been updated that are backed up.

Complete and incremental data backups with checkpoint are inherently consistent. They are only recommended if the database is to be **migrated** or **copied**.

When making [backups \[Page 116\]](#) of this kind, delays may be caused by the need to wait for the checkpoint.

Log Backup

The Database Manager (DBMGUI) supports both [automatic log backup \[Page 32\]](#) and [interactive log backup \[Page 33\]](#).

Automatic Log Backup

Automatic Log Backup

Automatic log backup is recommended to ensure the safety of data in production systems (See [Enable Automatic Log Backup \[Page 131\]](#)).

With the automatic log backup on, a log segment is saved once it has been written in full. It is then released again. This has the advantage that a log overflow is almost impossible. You configure the size of the log segment with the database parameter [LOG SEGMENT SIZE \[Page 88\]](#).

When the automatic log backup is enabled, it generates version files whose names consist of the file name allotted when defining the medium ([Defining Medium \[Page 121\]](#)) and a number which the system assigns in sequence as the logs are being written.

This mechanism is particularly recommended for all database instances in which extensive write and change-intensive transactions are carried out. In this way, constant monitoring of the usage level of the log devspace is not necessary.



Rewind tapes, pipes, and external backup devices are not supported for the automatic backup of log segments.

While the automatic log backup is enabled or active, no other log backup can be performed, although a [data backup \[Page 27\]](#) can be performed.

Interactive Log Backup

By using the interactive log backup you can [back up \[Page 116\]](#) all the pages that have been written to the log devspace since the last [log backup \[Page 31\]](#). The log is backed up in segments. The size of a log segment is defined by the parameter [LOG_SEGMENT_SIZE \[Page 88\]](#).

The log is backed up to files.

Prerequisites

A [full data backup \[Page 27\]](#) of the current database instance has been created.



You can perform log backups in `WARM` or `COLD` mode.

You cannot execute log backups in `DEMO` [log mode \[Page 87\]](#).

Restartability

Restartability

If a database failure occurs (due to a power failure for example) and if the [devspaces \[Page 14\]](#) are still intact, the system uses the log to restore the database instance to its last consistent state when [the database instance is started \(restarted\) \[Page 113\]](#).

This means that the effects which completed transactions had on the [data devspaces \[Page 16\]](#) are reproduced (rolled forward), and the effects that uncompleted transactions would have had are canceled out (rolled back).

Even if a devspace fails (due to a physical fault with the disk), the database instance can be returned to its last consistent state once the problem has been rectified by simply importing the last full [data backup \[Page 27\]](#) (provided that the log still contains all of the necessary data).

Process Structure

A database instance consists of a set of UNIX processes or Windows NT threads.

Under UNIX, a process acts as a **UKP** (user kernel process) or as a **special process** with special tasks.

Under Windows NT, the term "thread" is used instead of process; accordingly, the term "**UKT**" (user kernel thread) or "**special thread**" is used as well.

The required number of UKPs/UKTs and of special processes/threads depends on the number of [devspaces \[Page 14\]](#), the hardware configuration, and the [database parameters \[Page 80\]](#).

[User Kernel Process/User Kernel Thread \(UKP/UKT\) \[Page 36\]](#)

[Special processes/threads \[Page 45\]](#)

[Operating-system-dependent special processes/threads \[Page 51\]](#)

User Kernel Process/User Kernel Thread

User Kernel Process/User Kernel Thread

A user kernel process/user kernel thread forms a subset of the complete task set (for internal tasking). The types of task which exists are as follows:

User tasks [Page 37]	Utility task [Page 41]
Server task [Page 38]	Tracewriter task [Page 42]
Log writer [Page 39]	Conv scanner [Page 43]
Data writer [Page 40]	Timer task [Page 44]

User Tasks

When logging on to the database instance, each [user \[Page 19\]](#) of the instance or each application is assigned precisely one fixed user task. The user task ensures the processing of SQL statements for the session.

The number of user tasks available is defined by the [MAXUSERTASKS \[Page 86\]](#) parameter.

Server Tasks

Server Tasks

The main purpose of server tasks is to parallelize database functions such as [backing up to a group of parallel media \[Page 130\]](#), [restoring a number of media in parallel \[Page 134\]](#), and index compilation.

When the database instance is being [configured \[Page 78\]](#), the number of server tasks is determined automatically from the number of [data devspaces \[Page 16\]](#) and the number of data backup devices in use.

Logwriter

The logwriter is responsible for writing before and after images to the [log devspace\(s\)](#) [Page 17].

Datawriter

Datawriter

Datawriter tasks are responsible for writing data from the [data cache \[Page 60\]](#) to the [data devspaces \[Page 16\]](#). They become active when a [savepoint \[Page 62\]](#) or [checkpoint \[Page 63\]](#) is performed.

Savepoint writing takes a long time for a large data cache. The datawriters also become active between the end of one and the start of the next savepoint, to write data asynchronously from the data cache to disk.

The number of datawriters is calculated by the system. It depends primarily on the data cache size and the number of data devspaces.

Utility Task

The utility task is reserved solely for managing the [database instance \[Page 11\]](#).

As there is only one utility task for each database instance, no parallel managing actions can be performed.

An exception to this rule is the [automatic log backup \[Page 32\]](#). This can be performed in parallel with other management actions.

Tracewriter Task

Tracewriter Task

The database system allows the activation of a special protocol, the **Vtrace**, for diagnostic purposes. The trace writer task is provided for this purpose.

Conv Scanner

Normally, a sufficient number of addresses is available in the [system devspace \[Page 15\]](#) and cache. These addresses are passed to the dev processes/threads.

At times when the database instance is very busy and an above-average number of new pages is being added to it, the number of addresses available in the main memory's Pno pool may drop below a certain level. When this happens, the conv scanner quickly supplies the addresses of fresh empty pages by asynchronously finding page addresses in the system devspace which have been cleared for reuse by **Delete** or **Drop**.

Timer Task

Timer Task

The timer task handles timeout situations of all types.

Special Processes/Threads

[Requestor \[Page 46\]](#)

[Dev Processes/Threads \[Page 47\]](#)

[Coordinator \[Page 48\]](#)

[Timer \[Page 49\]](#)

[Temporary Dev Processes/Threads \[Page 50\]](#)

Requestor

Requestor

The requestor receives both local communication requests (**CONNECT**) and requests from the network and assigns them to a [user kernel process/user kernel thread \[Page 36\]](#).

Dev Processes/Threads

Dev processes/threads handle the read and write commands that read and write tasks ask to have performed.

Their number depends primarily on the number of [devspaces \[Page 96\]](#) in the database installation. Under normal circumstances, two dev processes/threads are activated for the [system devspace \[Page 15\]](#) and for each [data devspace \[Page 16\]](#) and [log devspace \[Page 17\]](#). Only one dev process/thread is activated for writing the vtrace (if the vtrace is switched on).

The process/thread dev0 plays a special role; dev0 coordinates and monitors the dev processes/threads. For example, if a mirrored devspace fails in warm mode (bad devspace), dev0 ensures that the corresponding dev processes/threads are terminated. Database operation is not impaired in this case.

If the database is enlarged while running by adding another data devspace, dev0 ensures that new dev processes/threads are generated.

All the other dev processes/threads write data to or read it from the devspaces.

Coordinator

Coordinator

The coordinator process/thread monitors all the kernel processes/threads in the database.

At [startup of the database instance \[Page 113\]](#), the coordinator process/thread is the first process/thread to become active. It coordinates the starting processes of the other processes/threads.

If a process fails while a UNIX operating system is running, the coordinator stops all the other processes.

If a thread fails while a Windows NT operating system is running, an exception handler becomes responsible for terminating all the other threads in an orderly way.

Timer

The timer monitors time for timeout control.

Temporary Dev Processes/Threads

Temporary Dev Processes/Threads

Processes/threads (`asdev<i>`) are temporarily activated to read and write data for [data backups \[Page 27\]](#).

Operating-System-Dependent Special Processes/Threads

[Clock thread \[Page 52\]](#)

[Console process \[Page 53\]](#)

[Console thread \[Page 54\]](#)

Clock Thread

Clock Thread

The clock thread is only used under Windows NT.

It computes internal times; for example, to determine the time needed to execute a command.

Console Process

In UNIX systems the console process collects database messages from other processes and logs them in file `knldiag`.

This same file [knldiag \[Page 156\]](#) is also created under Windows NT. Each thread writes information to this file.

Console Thread

Console Thread

Under Windows NT, this special thread processes requests made by the X_CONS console. The X_CONS program communicates with the console thread for this purpose.

In UNIX systems, X_CONS receives the required information from the shared memory of the processes.

Caches

Read and write operations to the [devspaces \[Page 14\]](#) are buffered in order to save on disk accesses.

The pertinent main memory structures are called caches. They can be sized appropriately.

The database system recognizes the following caches:

[Catalog cache \[Page 56\]](#)

[Converter cache \[Page 57\]](#)

[Free block management \(FBM\) cache \[Page 58\]](#)

[File directory cache \[Page 59\]](#)

[Data cache \[Page 60\]](#)

[Log cache \[Page 61\]](#)

Catalog Cache

Catalog Cache

This contains the last catalog objects used by a database session and the internal representation (application plans) of the last executed commands.

Data which is expelled from the catalog cache is moved for the time being to the [data cache](#) [\[Page 60\]](#).

A catalog cache is created for each user session.

Converter Cache

The converter cache contains the last read- or write-accessed pages of the [system devspace](#) [\[Page 15\]](#).

The converter cache is shared by all simultaneously active users.

The hit rate, that is the relation between successful and unsuccessful accesses to the converter cache, is a crucial measure of performance. Successful access means that the required data was already available in the converter cache.

For the converter cache, you should strive for hit rates close to 100%.

Free Blocks Management (FBM) Cache

Free Blocks Management (FBM) Cache

The FBM cache is used in the management of free disk blocks.

File Directory

The database system uses the file directory cache for internal organization. For example, the page addresses of the roots of the individual data trees are administered in the file directory cache.

Data Cache

Data Cache

The data cache contains the last read- or write-accessed pages of the [data devspaces \[Page 16\]](#).

The data cache is shared by all simultaneously active [users \[Page 19\]](#).

The hit rate, that is the relation between successful and unsuccessful accesses to the data cache, is a crucial measure of performance. Successful access means that the required data was already available in the data cache.

Log Cache

The log cache comprises the rollback cache and the log queue.

First of all, log entries are written to the rollback cache. During the COMMIT or if a page of the rollback cache is full, it is transferred into the log queue. The COMMIT is only confirmed once the [Log Writer \[Page 39\]](#) has written the pages from the log queue to disk.

Savepoint

Savepoint

The system performs savepoints at regular intervals. It writes all changed pages held in the [data cache \[Page 60\]](#) to disk.

However, because the savepoint is carried out while the database is running, you cannot restart the [database instance \[Page 11\]](#) when [restoring \[Page 134\]](#) it until you have also retrieved the log entries and/or the [log backups \[Page 31\]](#) that were made since the savepoint.

Checkpoint

Unlike the [Savepoint \[Page 62\]](#) the checkpoint is only executed on request and only when all the transactions have been completed.

When you start a complete [data backup with checkpoint \[Page 30\]](#), all the transactions currently running are completed and no further write transactions are permitted. All the amended pages held in the [data cache \[Page 60\]](#) are then written from the cache to disk to ensure that the status of the [data devspaces \[Page 16\]](#) of the database remains consistent. Pending transactions (ones started after the checkpoint was requested) are not run until the checkpoint has been completed.

This produces a totally consistent data backup which you can use to restore the database to its full state at the time of the checkpoint without the need for any additional [log backups \[Page 31\]](#).

Multiprocessor Configurations

Multiprocessor Configurations

To allow multiprocessor configurations to be used to the best advantage, ADABAS supports external/internal tasking that can be configured.

The aim here is to allow the maximum possible number of database sessions to be supported by the minimum possible number of operating system processes.

The configuration of ADABAS controls the degree of external/internal tasking via the two parameters [MAXUSERTASKS \[Page 86\]](#) and [MAXCPU \[Page 83\]](#).



Basically, this means that if, for example, you want to have up to 800 user sessions on a computer equipped with 4 processors for the database, you must set MAXUSERTASKS to 800 and MAXCPU to 4. This then enables the database instance to fully utilize all four processors by setting up four operating system processes, each of which performs internal tasking for up to 200 users.

Availability

The availability of a database instance can be increased by using the appropriate hardware, operating system, or database features.

If you wish to ensure a high standard of availability, we recommend using RAID-5 configurations for the [data devspaces \[Page 16\]](#).

Then a failure and the exchange of a disk does not impair database operation.

When RAID-5 systems are used, the database instance should be configured with several data devspaces. Performance will be better with many data devspaces than with a single one, because some parallel mechanisms used by the database system depend on the number of configured data devspaces.

For performance reasons, the [log devspaces \[Page 17\]](#) must not be created on RAID-5 systems but on dedicated disks or RAID-1 systems.

When using fault-tolerant hardware, it is best to only use the same type of hardware when you want to extend the capacity. For example, RAID-5 systems should only be extended using RAID-5 systems and mirroring disks with mirroring disks.



If you expand fault-tolerant systems with conventional disks, the database instance needs to be [restored \[Page 134\]](#) to redistribute the data.

Irrespective of the hardware and operating system properties, the database system (up to Version 7.2) itself provides a mirror of the [system devspace \[Page 15\]](#) and all data devspaces. Database parameter `MIRRORED_DATA` controls the mirror, and it requires that a corresponding number of mirror devspaces are defined.

In addition, mirroring of the log devspaces can be specified with parameter [LOG MODE \[Page 87\]](#).

In a mirrored configuration, read operations alternate between the original and the mirrored devspace. Write operations take place to both devspaces.

Screen Areas of the Database Manager (DBMGUI)

Screen Areas of the Database Manager (DBMGUI)

The screen for the Database Manager (DBMGUI) contains menu bars and windows. The size of the window areas can be varied.

[Menu Bar \[Page 68\]](#)

Functions and screen indications for the current database instance are selected from the horizontal menu bar along the top edge of the screen. This menu bar changes dynamically to suit the functions selected.

[Icon Bar \[Page 69\]](#)

The icon bar contains icons you can use to call up functions shown on the menu bar for the current database instance.

[Directory of Registered Database Instances \[Page 70\]](#)

In this directory you can organize the registered database instances by creating files in a tree structure. To move files to different levels within a directory, select the required file and, keeping the left-hand mouse button pressed down, drag it to the desired location .

If you select a file, the database instances registered in it and all files below it are displayed as the list of registered database instances. The file at the highest level (*Databases*) contains all the registered database instances.

[List of Registered Database Instances \[Page 71\]](#)

This list shows the database instances which are registered in the file selected and in all files below it, and their current status.

You select the current database instance by marking a line, that is a database instance.

Depending on the status of the database instance selected, there are various functions it can perform and screen indications it can produce. Functions can be called up from either the menu bar at the top of the screen or the menu list at the left-hand edge.

Double clicking on the database instance in the Display opens this database instance's short description. To move registered database instances to other files, select the required database, keep the left-hand mouse button pressed down and drag it to the desired file within the Directory of Registered Database Instances.

[Menu List of the Current Database Instance \[Page 72\]](#)

The menu list at left-hand edge of the screen can be used to select functions for the current database instance in a similar way to the menu bar at the top of the screen.

[Display \[Page 73\]](#)

The display area is the central area on the screen. It shows the name of the current database instance, the entry templates for functions and the values displayed as screen indications.

When you select a menu option from the menu list of the current database instance, the appropriate items are shown in the output area.

Screen Areas of the Database Manager (DBMGUI)

[Message Display \[Page 74\]](#)

- The message display is used to output general messages and errors.

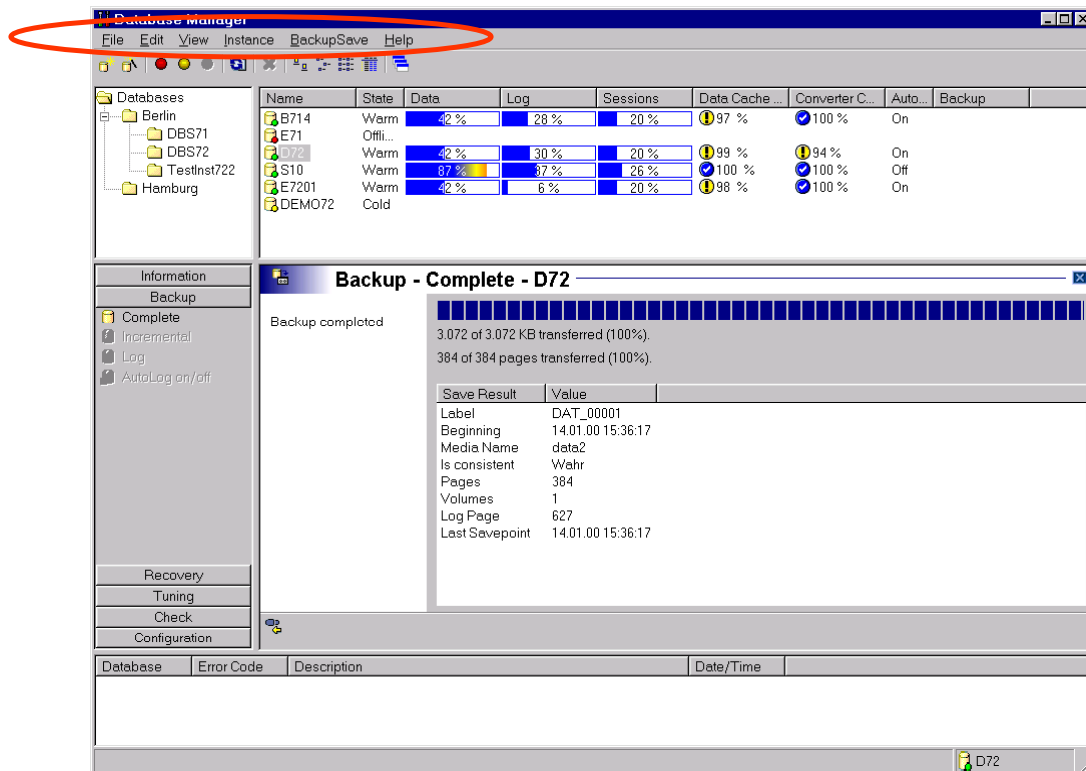
[Name Window \[Page 75\]](#)

The name window shows the name and operating status of the current database instance.

Menu Bar

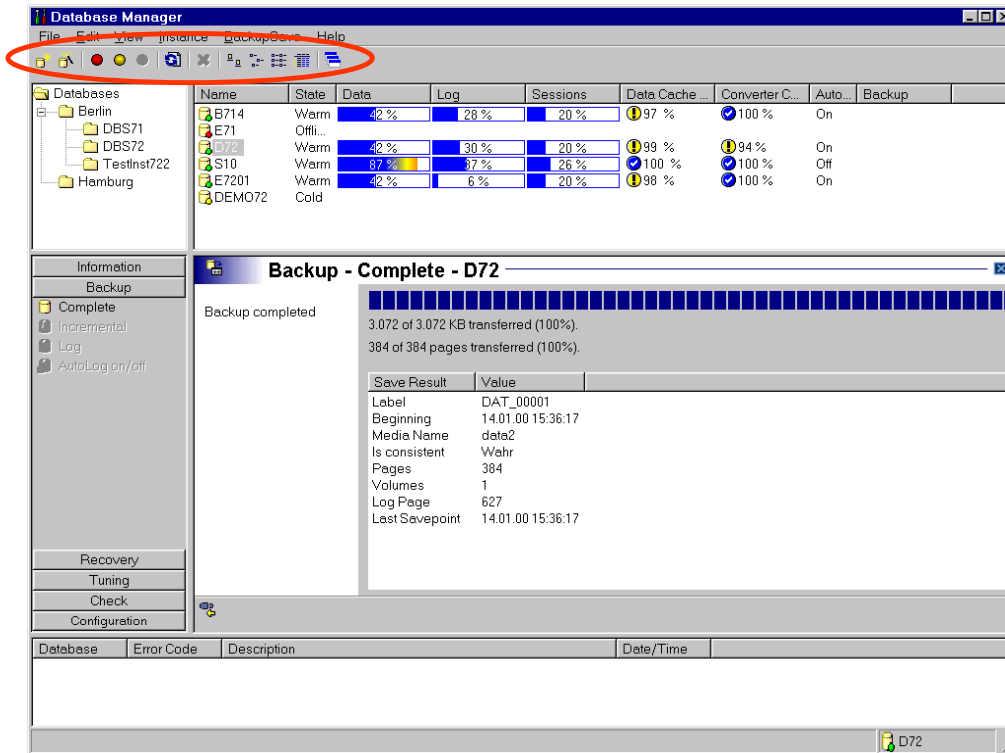
Menu Bar

The area on the screen shown in red below is the *menu bar*.



Icon Bar

The area on the screen shown in red below is the *icon bar*.



Directory of Registered Database Instances

Directory of Registered Database Instances

The area on the screen shown in red below is the *directory of registered database instances*.

Name	State	Data	Log	Sessions	Data Cache ...	Converter C...	Auto...	Backup
B714	Warm	22 %	28 %	20 %	97 %	100 %	On	
E71	Offli...							
D72	Warm	22 %	30 %	20 %	99 %	94 %	On	
S10	Warm	87 %	87 %	26 %	100 %	100 %	Off	
E7201	Warm	22 %	6 %	20 %	98 %	100 %	On	
DEMO72	Cold							

Backup - Complete - D72

Backup completed

3.072 of 3.072 KB transferred (100%).
384 of 384 pages transferred (100%).

Save Result	Value
Label	DAT_00001
Beginning	14.01.00 15:36:17
Media Name	data2
Is consistent	Wahr
Pages	304
Volumes	1
Log Page	627
Last Savepoint	14.01.00 15:36:17

Database	Error Code	Description	Date/Time
----------	------------	-------------	-----------

List of Registered Database Instances

List of Registered Database Instances

The area on the screen shown in red below is the *list of registered database instances*.

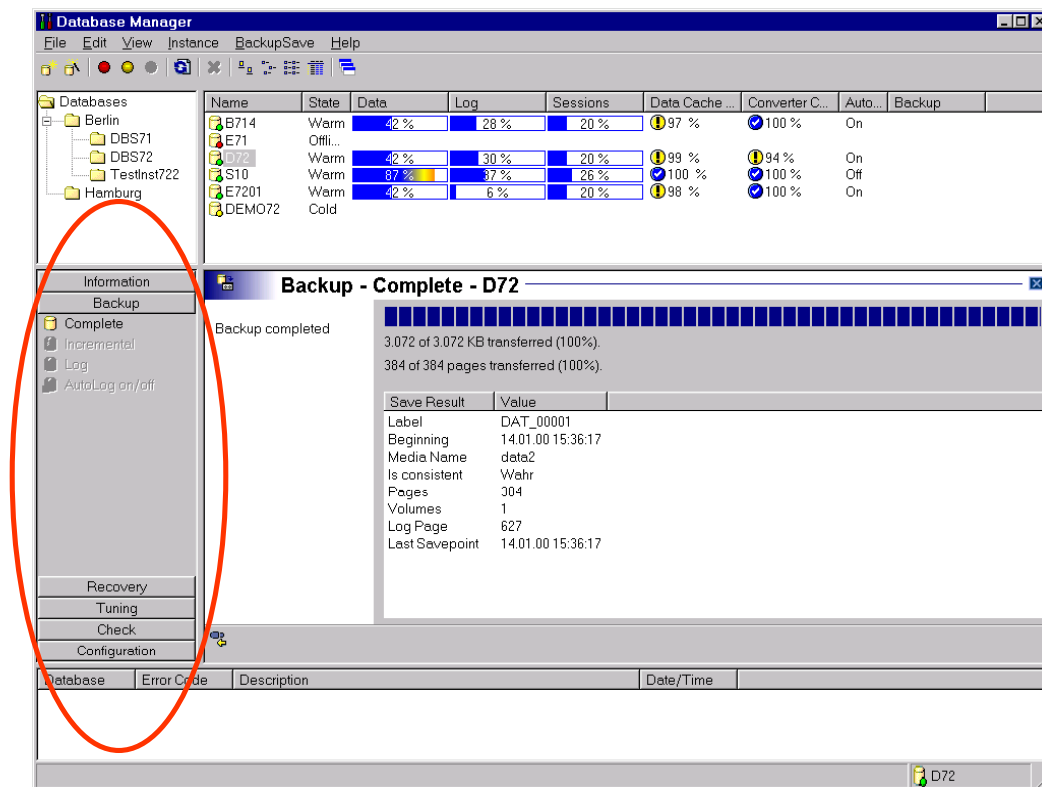
Name	State	Data	Log	Sessions	Data Cache	Converter C.	Auto	Backup
D714	Warm	42 %	28 %	20 %	97 %	100 %	On	
E71	Offli...							
D72	Warm	42 %	30 %	20 %	99 %	94 %	On	
S10	Warm	87 %	87 %	26 %	100 %	100 %	Off	
E7201	Warm	42 %	6 %	20 %	98 %	100 %	On	
DEMO72	Cold							

Save Result	Value
Label	DAT_00001
Beginning	14.01.00 15:36:17
Media Name	data2
Is consistent	Wahr
Pages	384
Volumes	1
Log Page	627
Last Savepoint	14.01.00 15:36:17

The Menu List of the Current Database Instance

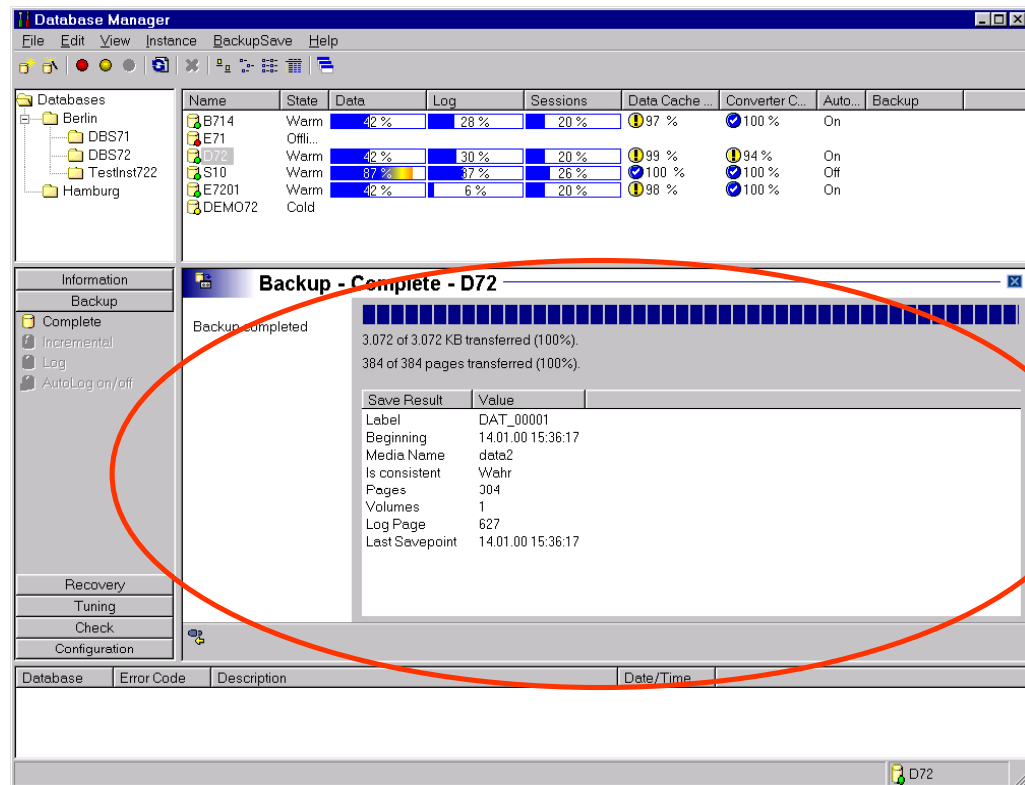
The Menu List of the Current Database Instance

The area on the screen shown in red below is the *menu list of the current database instance*.



Output Area

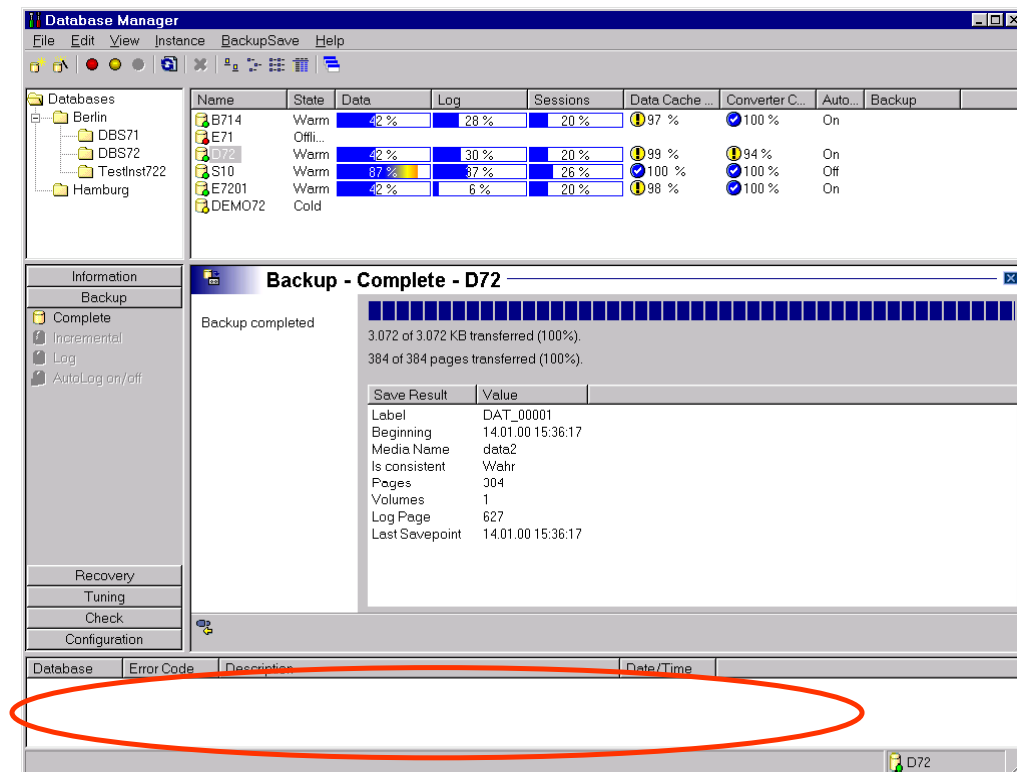
The area on the screen shown in red below is the *output area*.



Message Output Area

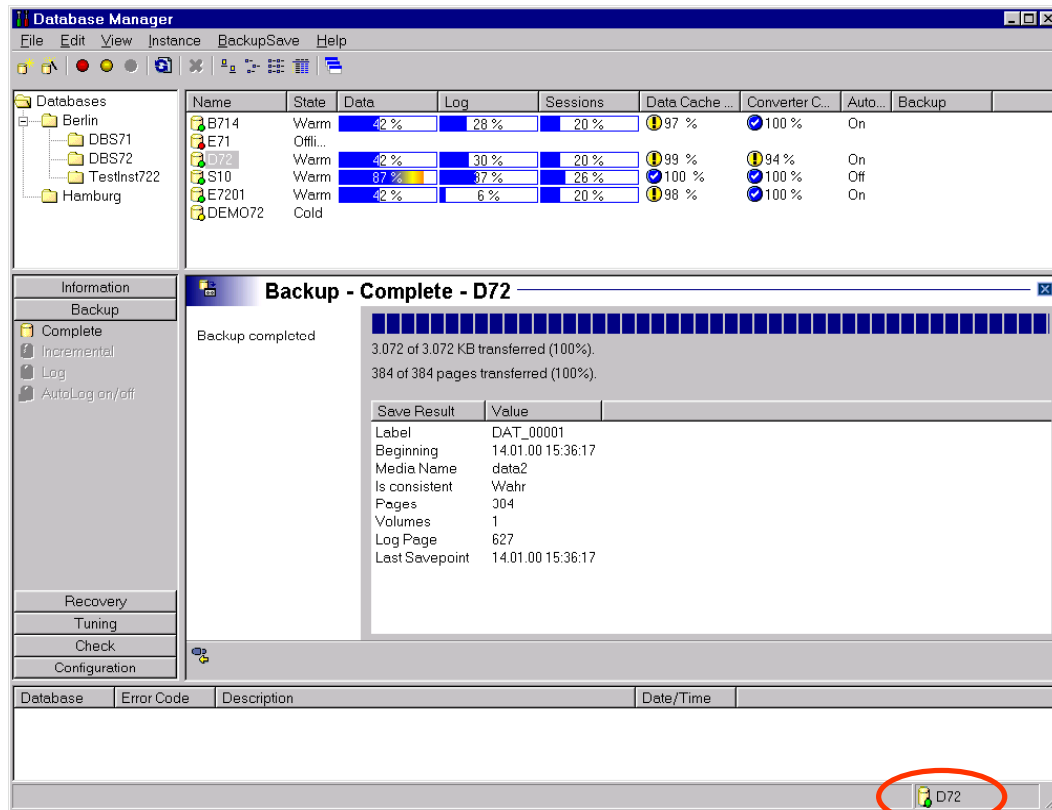
Message Output Area

The area on the screen shown in red below is the *message output area*.



Name Window

The area on the screen shown in red below is the *name window*.



New Database Instance Installation

New Database Instance Installation

Procedure

Select *Instance* → *Install* to open the Database Wizard. This installation wizard now guides you through the whole installation process.

1. Enter the name of the new database instance. This name must be unique.
If you want to install the database instance on a remote PC, enter the name of the PC as well as your user name and password for this PC.
Choose *Next*.
2. There may be more than one installation of the database software on the same server. The different installations may also be of different versions. Of the installations present on the server, select the one you want.
Choose *Next*.
3. Enter the users **DBM** and **DBA** and assign passwords for [DBM \[Page 20\]](#) and [DBA \[Page 21\]](#).
Choose *Next*.
4. Select one of the following options:

Initialize parameters with default values

When the parameter file for the database instance is created, the standard configuration stored when the database software was installed is used.

Choose an instance type for the database.

Use current parameters

You want to reinstall an existing database instance and copy its parameters.

Copy parameters from existing database

You want to use the configuration of a database instance on the server. Select the database instance whose parameter file is to be copied.

Restore parameters from a backup medium

You want to use the parameter file in a backup to install the new database instance.

5. Now adjust the parameters to your requirements. Select the required parameters and then choose *Edit*.

It now appears in the bottom part of the window. Next to the parameter you will see an explanation of what the parameter does and, in some cases, how it is calculated.

Enter the new value for the parameter in the *New Value* field and confirm your entry.

The new value appears in the *New Value* column. It is stored in the internal data structures and becomes effective once the database instance is restarted.

Choose *Next*.

When you exit the input screen, the parameters are checked following the rules stored on the server. You may be asked to make and confirm changes before you are permitted to exit the input screen.

6. Define the properties of the [log devspaces \[Page 17\]](#) and the [data devspaces \[Page 16\]](#).
When doing so, bear in mind the parameters set in the previous step.

New Database Instance Installation

Select a devspace and then choose *Edit*. Enter the size and ID (or absolute path) of the devspace and confirm your entries. Repeat for each devspace.

Choose *Next*.

7. Choose between the following options:

Install instance

The database instance is installed with the parameters specified under point 5.

Restore Instance

The database instance is re-initialized with the parameters under point 5. The database instance must then be [restored \[Page 134\]](#) in the Database Manager.

Choose *Next*.

8. Choose *Install* to start installing the new database instance or initialize the selected instance.

You can see how the generation or initialization of the database instance is progressing. If any errors or faults occur, the process is stopped immediately. However, the state that had been reached when the error or fault occurred is preserved.

Before you complete this step, the system prompts you to enter the name under which the database instance should be registered in the Database Manager. Confirm your entries.

9. Select *Close* to close the Database Wizard.
This takes you back to the [initial screen \[Page 66\]](#) of the Database Manager (DBMGUI).

Once a new database instance has been installed, its status is `WARM`.

After an existing database instance has been initialized, its status is `COLD` and can be restored straight away.

Configuring the Database Parameters

Configuring the Database Parameters

To set up the [database parameters \[Page 80\]](#) in the first instance, use is generally made of the default configuration which was stored when the database software was installed.

The configuration generated by the system will always be runnable. If necessary, you can still adjust the parameters originally set to suit any specific requirements you may have.

Alternatively you can use the configuration of a [database instance \[Page 14\]](#) already present on the server or the configuration from a [data backup \[Page 27\]](#). Even after this you can still adjust the database parameters to suit your own requirements ([Displaying and Changing Current Database Parameters \[Page 79\]](#)).

Displaying and Changing Current Database Parameters

Prerequisites

The database instance may be in any mode.

Procedure

1. Choose *Instance* → *Configuration* → *Parameters*
2. Choose
General to see the general database parameters or
Extended to see the special parameters.

Name	Name of database parameter
Value	Setting of database parameter currently being used in the kernel of the database instance
New value	No entry: The database parameter [Page 80] has not been changed since the database instance was last started. New value entered: The database parameter has been changed since the database instance was last started. However, no changes are saved to the parameter file and none of them become active until the next startup of the database instance (Restart [Page 113]).
Description	Help text on the selected parameter

3. To change a parameter setting, select the parameter.
4. Choose *Parameter* → *Edit*
It now appears in the bottom part of the window. Next to the parameter you will see an explanation of what the parameter does and, in some cases, how it is calculated.
5. Enter the new value for the parameter in the *New Value* field.
Confirm your entries.
The new value appears in the *New Value* column. It is stored in the internal data structures and becomes effective once the database instance is restarted.

Database Parameters

Database Parameters

For greater clarity, the database parameters for the user are divided into two groups and are listed in alphabetical order.

General [Page 81]	List of general database parameters
Extended [Page 91]	List of special database parameters

General - General Database Parameters

CONVERTER_CACHE	The size of the converter cache in pages
DATA_CACHE	The size of the data cache in pages
KERNELVERSION	Current database kernel variant
LOG_MODE [Page 87]	SINGLE: Log entries are saved to a log devspace DUAL: Log entries are saved to two (mirrored) log devspaces simultaneously DEMO: Log entries are overwritten cyclically
LOG_SEGMENT_SIZE [Page 88]	Size of log segment in pages
MAXARCHIVELOGS	Maximum number of log devspaces [Page 17] . This does not include mirrored log devspaces
MAXBACKUPDEVS [Page 82]	Maximum number of backup devices operating in parallel
MAXCPU [Page 83]	Maximum number of processors permitted
MAXDATADEVSPACES	Maximum number of data devspaces [Page 16]
MAXDATAPAGES [Page 84]	Total size of all devspaces together
MAXLOCKS [Page 85]	Maximum number of simultaneous line locks
MAXUSERTASKS [Page 86]	Maximum number of database users active at the same time (database sessions)
OBJ_LOG_SIZE	When the value of this database parameter is higher than 0, the database instance is operated as an object management system (OMS).
RESERVED_REDO_SIZE [Page 89]	Size (pages) of the memory area reserved for redo operations
RESTART_SHUTDOWN	Specification that determines how the database instance is started or shut down: AUTO (automatic) MANUAL (has to be started by database administrator) The parameter is only effective on Windows NT.
RUNDIRECTORY [Page 90]	Run directory for the database instance

MAXBACKUPDEVS**MAXBACKUPDEVS**

The [backup \[Page 116\]](#) and [recovery \[Page 134\]](#) of [data devspaces \[Page 16\]](#) can be speeded up by using a number of files or tape devices in parallel.

With this parameter, the user sets the maximum possible number of parallel files or tape devices.

MAXCPU

This parameter assigns the database instance a maximum number of CPUs.

The MAXCPU parameter defines over how many CPUs the [user tasks \[Page 37\]](#) generating the main load are distributed.

When defining MAXCPU, however, remember that operating system resources must also be available for other processes.

For [multiprocessor computers \[Page 64\]](#) you can define and limit the number of CPUs utilized by the database. For single-processor computers, enter a value of 1 for MAXCPU.

MAXDATAPAGES

MAXDATAPAGES

This parameter restricts the total size of all the [devspaces \[Page 14\]](#) combined.

You can only [create another devspace \[Page 98\]](#) when MAXDATAPAGES is greater than the sum of the configured sides of all current devspaces. This new devspace cannot be bigger than the difference between the total disk space configured with MAXDATAPAGES and the sum of all previous devspaces.

When the database instance is being configured, the Database Manager (DBMGUI) sets the [database parameter \[Page 80\]](#) MAXDATAPAGES with a reserve equal in size to the largest devspace so that a [data devspace \[Page 16\]](#) can be added in `WARM` mode. If the available space is exhausted by adding a devspace, the Database Manager (DBMGUI) increments parameter MAXDATAPAGES by exactly one devspace at the next [database instance restart \[Page 113\]](#).

MAXLOCKS

This parameter defines the maximum number of entries in the lock list in which the row and table locks held and requested are recorded for all users.

MAXUSERTASKS

MAXUSERTASKS

The MAXUSERTASKS parameter defines the maximum number of simultaneously active users (database sessions). Overconfiguration exceeding the actual requirement leads to increased demands on address space, especially shared memory.



Once the number of configured database sessions is reached, no other users can connect to the database instance concerned. The number of active sessions is therefore a critical parameter of database operation and must be monitored.

LOG_MODE

By defining this parameter you can decide how the log entries are backed up:

- singly (**SINGLE**)
- in mirrored form (**DUAL**)
- not at all (**DEMO**)

For security, you are recommended to mirror the log always. You can either do this using database software (by setting the parameter `LOG_MODE: DUAL`) or you can use hardware-based mirroring. If your operating system allows hardware-based mirroring (RAID-1 systems), we advise you to use it.

DUAL Option

The minimum configuration for a mirrored log is two disks, one for the log and one for the mirrored log.

The advantage of this configuration is that a failure of one of the log devspaces does not interrupt database operation, and once the defective devspace has been repaired, it can be restored while the database is operating.

SINGLE Option

You configure one log only. This is useful if only one disk is configured for the log area.

The log should be backed up periodically.

If a disk fault occurs, the contents of the [log devspaces \[Page 17\]](#) is corrupt and the data devspace is no longer consistent with the transaction. Most of the database can be restored with the help of a complete [data backup \[Page 27\]](#) and any further log backups which may be needed. What cannot be restored is any log content which had not been backed up yet and any transactions which had not yet been performed.

If you are using hardware-based RAID-5 or RAID-1 systems, you can enter **SINGLE** for the parameter `LOG_MODE` because these systems offer sufficient backups during operation. Accesses to the log devspace have a large impact on the performance of the overall system. For this reason, please ensure when you choose your RAID system that it has a high data throughput.

DEMO Option

The log is written if you select the option **DEMO**. Once it is full, the system does not simply stop; it overwrites the previous log entries. Please note that these log entries are then lost.

For this reason, database instances in which the log mode parameter is set to **DEMO** can only be restored with a consistent data backup ([Data Backup with Checkpoint \[Page 30\]](#)).

Because of the poor safety it offers in the event of a failure, you are advised to use this configuration only for test or demonstration databases.

LOG_SEGMENT_SIZE**LOG_SEGMENT_SIZE**

You use this parameter to define the log segment size in pages.

The log segment size defines the size of the segments in which the log is saved ([interactive log backup \[Page 33\]](#)) and also the threshold value which causes the logs to be backed up with the [automatic log backup \[Page 32\]](#).

The size of the log segments is independent of the size of the individual [log devspaces \[Page 17\]](#) but it must not exceed the size of the complete archive log.

User-defined value	Log segment size used by system
No definition of log segment size (LOG_SEGMENT_SIZE = 0)	One third of the overall log devspace
Log segment size in pages is not greater than 50% of the overall log devspace	The defined value for LOG_SEGMENT_SIZE
Log segment size is greater than 50% of the overall log devspace	50% of the log devspace

RESERVED_REDO_SIZE

This database parameter defines the size in pages of the reserved redo area in the [data devspace \[Page 16\]](#). If you enter 0, the system initializes the redo area at half the size of the log.

To enable the data devspace to hold all the requisite log entries, part of it is reserved for them. This reserved part is called the redo area and it is used for [restarting \[Page 113\]](#) the database instance. While the database is running it is not available for permanent data objects.

When log entries are being recovered, they are copied to the redo area of the data devspace. Firstly this allows the log entries to be read in advance to establish whether there is no point in recovering transactions because they have a rollback instruction at the end. Secondly it allows log entries which do have to be recovered to be dealt with in parallel.

RUNDIRECTORY

RUNDIRECTORY

The Run Directory is a directory created when the system was installed for the purpose of storing the [log files \[Page 156\]](#). All the relative path names relate to the Run Directory of the database instance.

Extended - Special Database Parameters

BACKUP_BLOCK_CNT	Block size (in pages) when saved and restored
CAT_CACHE_SUPPLY	Size of catalog cache memory for all user tasks (in pages)
DATE_TIME_FORMAT	System default for date and time formats
DEADLOCK_DETECTION	Maximum search level for deadlock detection.
DEFAULT_CODE	Character set used to display characters
KERNELDIAGSIZE	Size (in pages) of the database kernel's log file
KERNELTRACESIZE	Size (in pages) of the database kernel's trace file
LOG_IO_QUEUE	Size (in pages) of log I/O queue
LRU_FOR_SCAN	Performance of the scanner in the data cache
MAXRGN_REQUEST	Maximum number of times a task attempts to access an autonomous area
MAXSERVERTASKS	Maximum number of server tasks used for processing requests
MP_RGN_LOOP	Maximum number of times a different task attempts to get a region
OBJ_LOG_IO_QUEUE	Size (in pages) of OBJ_LOG_IO_QUEUE
OBJ_LOG_SEG_SIZE	Size (in pages) of an object log segment
OBJ_ROLLBACK_CACHE	Size (in pages) of OBJ_ROLLBACK_CACHE
OPTIM_BUILD_RESLT	Change in the optimizer algorithm's strategies for generating result sets
OPTIM_FETCH_RESLT	Change in the optimizer algorithm's strategies that do not generate result sets
OPTIM_KEY_INV_RATE	Change in the key range strategies of the optimizer algorithm
OPTIM_MAX_MERGE	Change in the optimizer algorithm's strategies for merging index lists
OPTIM_ORDERBY_IDX	Change in the optimizer algorithm of ORDER-BY conditions
OPTIM_OR_DISTINCT	Change in the optimizer algorithm of OR conditions
REQUEST_TIMEOUT [Page 94]	Maximum waiting time before a lock is released (in sec.)
ROLLBACK_CACHE	Size of the main memory area used for caching log pages to increase the speed of rollbacks.
SEQUENCE_CACHE	The size of the sequence cache in pages
SESSION_TIMEOUT [Page 95]	Maximum length of inactive periods during database sessions (in sec.)
UTILITY_PROT_SIZE	Size of log file control.utl written by utilities to the installation directory of the database instance

Extended - Special Database Parameters

_DATA_CACHE_RGNS	Number of autonomous areas that are being used in parallel
_EVENT_ALIVE_CYCLE	The time (in seconds) of an event cycle
_OMS_RGNS	Number of autonomous areas in which OMS class users can work in parallel
_MAXEVENTS	Maximum number of events cached by the kernel that must be processed by the Database Manager
_ROW_RGNS	Number of autonomous areas in which lock collisions in lines can be checked in parallel
_TAB_RGNS	Number of autonomous areas in which lock collisions in tables can be checked in parallel
_TRANS_RGNS	Number of autonomous areas in which lock collisions in transaction statuses can be checked in parallel
_TREE_RGNS	Number of autonomous areas in which lock collisions in B* trees can be checked in parallel

DEADLOCK_DETECTION

This parameter defines the maximum search level for deadlock detection.

Deadlocks that have not been detected by the deadlock detection up to the given search level of the specified value are only resolved by the [REQUEST_TIMEOUT \[Page 94\]](#).

If `DEADLOCK_DETECTION = 0`, deadlock detection is disabled.

REQUEST_TIMEOUT**REQUEST_TIMEOUT**

This parameter restricts the waiting time for a lock to be released by other users for all database sessions. The time is specified in seconds.

If a lock request cannot be satisfied within the time thus defined, a message is returned to the waiting database session.

SESSION_TIMEOUT

This parameter defines the maximum time of inactivity allowed for all database sessions. The time is specified in seconds.

If no SQL statement is issued within the specified time, the database session concerned is terminated (ROLLBACK WORK RELEASE statement).

Devspaces

Devspaces

The database system has [system devspaces \[Page 15\]](#), [data devspaces \[Page 16\]](#) and [log devspaces \[Page 17\]](#).

Each database instance has one system devspace and one or more log and data devspaces. You configure the maximum numbers of data and log devspaces that are possible when [installing the database instance \[Page 76\]](#) (see [Devspace configuration \[Page 97\]](#)).

Data and log devspaces can both be added while the database is running ([Defining Devspaces \[Page 98\]](#)).

Paths for data and log devspaces can be changed ([Changing Devspaces \[Page 99\]](#)).

Devspace Configuration

You configure the maximum number of devspaces that is possible by setting the appropriate [database parameter \[Page 80\]](#) when [installing the database Instance \[Page 76\]](#).

Devspace type	Parameter
Data devspace	MAXDATADEVSPACES
Log devspace	MAXARCHIVELOGS

You can change the settings of both parameters at a later date regardless of which mode the database instance is in. Like all parameter changes, these changes take effect when the database instance is [restarted \[Page 113\]](#).

Procedure

1. First, follow the procedure described in [Displaying and Changing Parameters \[Page 79\]](#).

2. Then switch the database instance to the `COLD` mode.

3. Restart the database.

The Database Manager (DBMGUI) switches the database instance first into `OFFLINE` mode and then into `WARM` mode and saves the new parameter setting in the parameter file. It reserves the amount of memory space called for by the setting plus a reserve equal in size to the largest devspace of this type. This reserve allows you to create an additional devspace of this type while the database is running when the number originally planned is too small.



How to define devspaces is described in [Defining Devspaces \[Page 98\]](#).

Creating Devspaces

Creating Devspaces

Devspace type	Parameter
Data devspace	MAXDATADEVSPACES
Log devspace	MAXARCHIVELOGS

Prerequisites

The database instance is in `WARM` mode.

The database parameter for the devspace type is equal at least to the total number of existing and planned devspaces.



Each time the database is started, the system checks that there is enough space available for at least one devspace of each type. If there is not, it automatically increase the setting of the appropriate database parameter by 1.

If you wish to enlarge the database instance by more than one devspace of a given type, adjust the relevant parameter accordingly ([Configuring Devspaces \[Page 97\]](#)). Later on, this enables you to add a devspace of the given type a number of times in succession with the database instance in `WARM` mode without having to stop it.

Procedure

1. Choose *Instance* → *Configuration* → *Data Devspace* or *Log Devspace*.
The system outputs all of the existing devspaces of the selected type and their data, as well as any devspaces still possible (in accordance with the parameter value).
The system devspace depends on the size of the data devspaces and is adjusted automatically by the system.

Name	Name of devspace
Size	Size of devspace
Type	File, raw device, or symbolic link
Location	Path to devspace
M. Type	File, raw device, or symbolic link (mirrored devspace)
M. Location	Path to mirrored devspace

2. Choose a devspace.
3. Choose *Devspaces* → *Edit*. Define the new devspace by entering the size, type, and path.
4. Save your work.

Devspace Change

Prerequisites

You have copied the log or data devspace to a new location and you wish to inform the Database Manager (DBMGUI) of the path to the new location.

The database instance is in `OFFLINE` mode.

Procedure

5. Choose *Instance* → *Configuration* → *Data Devspaces* or *Log Devspaces*.
You are shown all the devspaces of the selected type that exist at the moment.

Name	Name of devspace
Size	Size of devspace
Type	File, raw device, or symbolic link
Location	Path to devspace
M. Type	File, raw device, or symbolic link (mirrored devspace)
M. Location	Path to mirrored devspace

6. Choose *Devspaces* → *Edit*.
7. Select the log or data devspace which is going to be changed. It now appears in the bottom part of the window.
Under *Location* enter the new path to the devspace.
8. Save your work.
When you now [restart the database instance \[Page 113\]](#) the database system knows the new path in it.



The path to the system devspace on the other hand cannot be altered.

Updating the System Tables

Updating the System Tables

You can use the Database Manager (DBMGUI) to update the system tables when there is a change of version.

Procedure

1. Choose *Instance* → *Configuration* → *Upgrade System Tables*
You only have to specify the DBA and DOMAIN user names and passwords if they are not known to the DBMServer. When you enter the user data, it is stored in the DBMServer for future activities.
2. Choose *Start* to load the system tables.
The system tables are loaded and updated. At the end of the process, the log is displayed.



If an error occurs, the `ERROR` status appears. You can then look at the installation log with the file name [dbm.ins \[Page 156\]](#) for more information on the error.

Terminal Support (Termchar Set)

A Termchar set is a character set which shows on the screen terminal-specific characters based on the ASCII code to ISO 8859/1.2 or EBCDIC code to CCSID 500, codepage 500.

The database system is supplied with termchar sets GERMAN, IBM437_GER and USASCII_GER.

Database Manager (DBMGUI) options:

[Displaying, Changing and Deleting Termchar Sets \[Page 102\]](#)

[Defining New Termchar Sets \[Page 103\]](#)

Displaying, Changing and Deleting Termchar Sets

Displaying, Changing and Deleting Termchar Sets

1. Select *Instance* → *Configuration* → *Termchar Sets*.
Select the desired [termchar set \[Page 101\]](#).
2. Select *Termchar Sets* → *Edit*
You will now be shown the name of the terminal character set selected, the code type and the *Enabled* field.

If you check the *Enabled* field, the selected termchar set will become available to the database instance at the next [Restart \[Page 113\]](#). By setting the environment variables in the appropriate way you can now specify this termchar set as a default value for the current terminal.



This function is only relevant to the Precompiler and Replication Manager applications. For performance reasons, the number of termchar sets for which the *Enabled* field can be selected is restricted to 10.

3. Select the desired termchar set. You can now look at the suggested conversion for the characters or can change it.

Name	Name of termchar set (name must not be more than 18 bytes long)
Code type	The code on which the termchar set is based. The valid codes are the ASCII code to ISO 8859/1.2 or the EBCDIC code to CCSID 500, codepage 500.
Enabled	If you check this field, the termchar set selected will become available to the database instance at the next restart [Page 113] .
Internal	Original form (hexadecimal)
External	Target form (hexadecimal)
Comment	Target form as a printable character

4. Save your work.

Defining Termchar Sets

5. Select *Instance* → *Configuration* → *Termchar Sets*.
6. Choose *Termchar Sets* → *New*

Define the name of the new terminal character set.

Select the code type you wish to use for the conversion (ASCII code to ISO 8859/1.2 or EBCDIC code to CCSID 500, codepage 500).

Select whether the new termchar set is to be made available to the database instance.

Define the conversions for the characters required. Each one-byte character to be converted must be specified in its original form and then in the target form with a maximum length of two bytes.

Name	Name of termchar set (name must not be more than 18 bytes long)
Code type	The code on which the termchar set is based. The valid codes are the ASCII code to ISO 8859/1.2 or the EBCDIC code to CCSID 500, codepage 500.
Enabled	If you check this field, the termchar set selected will become available to the database instance at the next restart [Page 113] .
Internal	Original form (hexadecimal)
External	Target form (hexadecimal)
Comment	Target form as a printable character

7. Save your work.
The new or changed definition of the termchar set will not be saved to the parameter file until the next [restart of the database instance \[Page 113\]](#).



In German-speaking countries termchar sets are needed for displaying umlauts for example.

If you are using the character-oriented applications Precompiler or Replication Manager and want to display umlauts, select IBM437_GER and check the *Enabled* field. This enables the termchar set selected. It will be available to the database instance at the next restart. By setting the environment variables in the appropriate way you can specify this termchar set as a default value for your current terminal.

Language Support (MapChar Sets)

A mapchar set is a character set which shows on the screen language-specific characters based on the ASCII code to ISO 8859/1.2 or EBCDIC code to CCSID 500, codepage 500. It is only of any significance for functions internal to the system .

For example, the mapchar set is used by the MAPCHAR SQL function to sort fields containing diacritics.

The database system is always supplied with the mapchar set called DEFAULTMAP. Database Manager (DBMGUI) options:

[Displaying, Changing and Deleting Mapchar Sets \[Page 105\]](#)

[Defining Mapchar Sets \[Page 106\]](#)

Displaying, Changing and Deleting MapChar Sets

1. Choose *Instance* → *Configuration* → *Mapchar Sets*.
You are shown the mapchar sets which are already available. Select the mapchar set you want.
2. Choose *Mapchar Sets* → *Edit*.
You are shown the name of the mapchar set selected, the code used for it (ASCII code to ISO 8859/1.2 or EBCDIC code to CCSID 500, codepage 500) and the suggested conversions for the characters.
You can now look at these or change them. If you need to, change the code and overwrite or delete the characters.

Name	Name of mapchar set (name must not be more than 18 bytes long)
Code type	The code on which the mapchar set is based. The valid codes are the ASCII code to ISO 8859/1.2 or the EBCDIC code to CCSID 500, codepage 500.
Internal	Original form (hexadecimal)
External Hex	Target form (hexadecimal)
Ext. ASCII	Target form as a printable character

3. Save your work.

Defining Mapchar Sets

Defining Mapchar Sets

8. Choose *Instance* → *Configuration* → *Mapchar Sets*.
You are shown the mapchar sets which are already available.
9. Choose *Mapchar Sets* → *New*

Define the name of the new Mapchar set.

Select the type of code used for the source files (ASCII code to ISO 8859/1.2 or EBCDIC code to CCSID 500, codepage 500).

Define the conversions for the characters required. Each one-byte character to be converted must be specified in its original form and then in the target form with a maximum length of two bytes.

Name	Name of mapchar set (name must not be more than 18 bytes long)
Code type	The code on which the mapchar set is based. The valid codes are the ASCII code to ISO 8859/1.2 or the EBCDIC code to CCSID 500, codepage 500.
Internal	Original form (hexadecimal)
External Hex	Target form (hexadecimal)
Ext. ASCII	Target form as a printable character

10. Save your work.

Database Instance Administration

[Registration of Database Instances \[Page 112\]](#)

[Starting \(Restarting\) the Database Instance \[Page 113\]](#)

[Stop database instance \[Page 114\]](#)

[Operation Mode Monitoring \[Page 115\]](#)

[Backing up data and log entries \[Page 116\]](#)

[Data recovery \[Page 134\]](#)

[Requesting Information \[Page 145\]](#)

[Software Release Update \[Page 160\]](#)

Activating and Deactivating Indices

Activating and Deactivating Indices

Use

You can determine which of the existing indices of a database instance are used.

You can also activate or deactivate selected indices and thereby change the access times.

Prerequisites

The database instance is in `WARM` mode.

Procedure

Choose *Instance* → *Tuning* → *Index Use*.

You can choose which indices are to be displayed. To do so, enter appropriate search arguments under `Owner`, `Table Name` and `Index Name`.

<code>Owner</code>	Owner
<code>Table Name</code>	Name of the table
<code>Index Name</code>	Name of the index

You can also choose between the following options:

<code>Disabled indexes only</code>	Only indices that have been deactivated
<code>Unused indexes only</code>	Only indices that are not used

Then choose *Indexes* → *Select*. A list of the indices that match your criteria is then displayed.

Select the indices you want to activate in the list. If you want to activate all of the indices, choose *Indexes* → *Mark All*.

Select the indices you want to deactivate in the list.

Choose *Indexes* and the required action.

Updating the Statistics Information

Use

Various statistics are compiled for each database instance, such as the number of table entries, the size of the tables and indices, and the value distribution (various values) of indices and columns. This information is required by the SAP DB Optimizer to define the best strategy for executing complex SQL statements.

The statistics information is stored in the database catalog.

If the database size or the values it contains have changed considerably, you will have to update the statistics. This should be carried out roughly once a week.

The statistics values can be updated for certain tables, columns, or for all basic tables.

Prerequisites

The database instance is in `WARM` mode.

Procedure

Choose *Instance* → *Tuning* → *Optimizer Statistics*.

Updating statistics entries for all tables

If you then choose *OptimizerStatistics* → *Execute*, all of the statistics entries for **all** of the tables are updated.

Updating statistics entries for selected tables

You can also choose to update **selected** statistics. To do so, enter appropriate search arguments under *Owner*, *Table Name* and *Column Name*.

You can specify the size of sample by entering appropriate values in the *Estimate Rows* and *Estimate Percent* fields. The update is then only carried out across the number of rows or the specified percentage, based on the total number of database tables. This option, therefore, is faster.



The reliability of the random sample depends on the size of the table and the physical position of the data. Measurements have shown that random samples of between 1000 and 5000 rows in large tables have short response times and produce high-quality results.

Owner	Owner
Table Name	Table name
Column Name	Column name
Estimate Rows	Number of rows checked
Estimate Percent	Percentage of rows checked

Updating the Statistics Information

Then choose *OptimizerStatistics* → *Select*. A list of the tables that match your criteria is then displayed. From the *Update Statistics Date* and *Update Statistics Time*, you can tell when the statistics for this table were last updated.

If you want to update the statistics for **one of the tables** in the list, simply select the relevant entry. If you want to update the statistics for **all** of the tables, choose *OptimizerStatistics* → *Mark All*.

Choose *OptimizerStatistics* → *Execute* to update the statistics for the selected tables.

Checking the Consistency of the Devspaces

Use

To check the consistency of the internal data structures in the current database instance. If there are serious inconsistencies, the database must be recovered in the same way as after a disk failure.

In COLD operating mode, free storage pages wrongly recorded as used following an abnormal termination of database operation are released to the free space management.

The size of the serverdb determines how long this function will take.



We recommend that you execute this function before each [full data backup \[Page 27\]](#).

Prerequisites

The database instance is in WARM or COLD mode.

Procedure

Choose *Instance* → *Check* → *Database* and then *CheckDatabase* → *Verify* to start the consistency check.

Result

Database Instance Registration

Use

If [database instances \[Page 11\]](#) are to be managed by the Database Manager (DBMGUI), they have to be registered.

Prerequisites

If you want to access a remote database with the Database Manager, you have to start the **XServer** service on the relevant database server (`Status: started`).

Procedure

1. Choose *File* → *Register Database*
Enter the name of the server from which you wish to request a list of the database instances installed on it.
Click on *Enter* or the *exclamation mark (!)* next to the input field.
2. On the list of the installations present on the server which is displayed, select the one that has to be registered.
Choose *Register*.
3. Enter a name for the database you want to register and the [DBM user identification \[Page 20\]](#) of the database instance, which comprises the DBM user name and the DBM password.



You can choose whatever name you please as the registered name of the database instance. It only needs to be unambiguous within the Database Manager (DBMGUI). Generally speaking it should be the same as the installation name of the database instance. If this has already been assigned you must use a different registered name.

4. Choose *OK*.

Result

The database instance has been registered and from now on it will appear on the [list of registered databases \[Page 66\]](#).



To organize your [directory of registered database instances \[Page 66\]](#), select *File* → *New Folder* | *Delete* | *Rename*.

Starting (Restarting) the Database Instance

Prerequisites

You have selected the [database instance \[Page 11\]](#) you wish to start.

Procedure

Database Instance Transfer to WARM State

Choose *Instance* → *Restart* → *Warm* to start the database instance, that is to switch it to the operational state.

This operation can be performed from both `COLD` mode and the deactivated mode (`OFFLINE`). Irrespective of the mode it is currently in, the database instance is first switched off and then switched to `COLD` mode and finally to `WARM` mode.

In the event of a system crash, *Instance* → *Restart* → *Warm* ensures that all transactions terminated with COMMIT are taken into the database and that all uncommitted transactions are rolled back.

If *Instance* → *Restart* → *Warm* fails to set the database to `WARM` mode, you can look for the reason for the fault in the log file [knldiag \[Page 156\]](#).

Database Instance Transfer to COLD State

Select *Instance* → *Restart* → *Cold* to switch the database instance from switched-off mode (`OFFLINE`) to `COLD` mode.



Each time you start the database instance by choosing *Instance* → *Restart* → *Warm* or *Instance* → *Restart* → *Cold*, a copy of the console log file (file `knldiag` in the [Rundirectory \[Page 90\]](#) of the instance) is created with the name `knldiag.old`, and file `knldiag` is overwritten.

Stopping the Database Instance (Shutdown)

Stopping the Database Instance (Shutdown)

Prerequisites

You have selected the [database instance \[Page 11\]](#), you wish to stop.

The database instance is in `WARM` mode.

Procedure

Database Instance Transfer to COLD State	Function for Taking the Database Instance OFFLINE
<ol style="list-style-type: none">1. Choose <i>Instance</i> → <i>Shutdown</i> → <i>Cold</i> to switch the database instance from <code>WARM</code> to <code>COLD</code> mode.2. Select <i>OK</i>, <i>Quick</i> or <i>Cancel</i>.	<ol style="list-style-type: none">1. Choose <i>Instance</i> → <i>Shutdown</i> → <i>Offline</i> to shut the database instance down.2. Select <i>OK</i>, <i>Quick</i> or <i>Cancel</i>.



These commands fail if a [data backup \[Page 27\]](#) or a [log backup \[Page 33\]](#) is running or [automatic log backup \[Page 32\]](#) is active.

An error message tells you that the backup action has to be completed or automatic log backup disabled before the database instance can be shut down.

Options on Stopping Database Instance

<i>OK</i>	First the system waits until all transactions have been completed properly, then it stops the database instance.
<i>Quick</i>	All the current transactions and user connections are terminated immediately. For an overview of the uncompleted transactions in the database instance, choose <i>Instance</i> → <i>Information</i> → <i>Users</i> .
<i>Cancel</i>	Cancels the command.

Monitoring the Mode





Prerequisites

You have selected the [database instance \[Page 11\]](#) whose mode you would like to check.

Mode WARM | COLD | OFFLINE

There are two possible ways in which you can find out what mode the database instance is currently in.

- Read the mode the database concerned is in from the *Status* column in the [list of registered databases \[Page 66\]](#).
- Check what code symbol appears on the general database instance icon in the [name window \[Page 66\]](#) for the present database instance.

	General database instance icon
	Database instance is OFFLINE
	Database instance is COLD
	Database instance is WARM

Backups

Backups

You can carry out [Data backups \[Page 27\]](#) (full and incremental) and [Log backups \[Page 31\]](#) with the Database Manager.

Both types of backup can be activated either interactively using the [Database Manager \(DBMGUI\) \[Page 1\]](#) or by a batch call using the [Database Manager \(DBMCLI\) \[Ext.\]](#).

For log backups, we recommend that you enable [automatic log backup \[Page 32\]](#). However, you can also make your [log backups interactively \[Page 33\]](#) to file.

The backup function of the Database Manager can be set to the optimum automated timetable by using the DBA timetable of the Computing Center Management System (CCMS). Log files on the backups performed are generated in the R/3 System and these give you an accurate picture of what happened during the backups.

See also:

DBA Scheduling Calendar (SAP DB in the BC - Computing Center Management System Manual)

Backup Concepts

[Backup Media](#)

[\[Page 118\]](#)[Backup Label](#) [\[Page 119\]](#)

Backup Media

Backup Media

One medium is assigned to each backup action. This medium can be a file, a tape, or a pipe.

A distinction is made between standard backup media and media for external backup devices. The use of external backup devices is not supported by the current release of the Database Manager.

Backup media can be defined before or during the backup process and they are defined especially for each individual database instance.

[Single media can be defined \[Page 122\]](#) or media can be combined to form a [group of parallel media \[Page 123\]](#).

Parallel media are simultaneously written or read by the database server. This increases data throughput - and thus the speed of [backup \[Page 116\]](#) or [restore \[Page 134\]](#).

The media definitions comprise the path names and properties of the tape device and of the tapes or files under a name that can be selected freely and which should relate to the practical situation.

We generally recommend that you use tapes or cassettes as backup media. The Database Manager uses files to back up logs.



It is best always to copy the files to tape or cassette. Only separate media can be kept at other locations as a safeguard in the event of fire or similar hazards.

You must follow the manufacturer's recommendations on how frequently tapes and cassettes should be used.

Backup Label

The database system automatically gives each backup an identifying label (known as the medium or backup label). This label uniquely identifies the backups done since the [installation of the database \[Page 76\]](#).

The label consists of the **type of backup** (complete or incremental [data backup \[Page 27\]](#) or [log backup \[Page 31\]](#)) and a **sequential number**.

Full and incremental data backups are number sequentially together. Log backups are numbered in a separate sequence.

The label is written to log file [dbm.knl \[Page 156\]](#) after the [backup \[Page 116\]](#). The complete contents of this file are shown in the [backup history \[Page 146\]](#).

In a [restore \[Page 134\]](#) the label is shown first and must be confirmed before the operation can start.



If the backup medium is a tape or a cassette, you should write the backup label on the sticker of the tape or cassette at the end of the backup.

Formal Description of the Backup Labels

`<backup_id> ::= <save_type>_<sequence_no>`

`<save_kind> ::= DAT | PAG | LOG`

`<sequence_no> ::= <number>`

Managing the Backup Media

Managing the Backup Media

[Defining media \[Page 121\]](#)

[Changing media definitions \[Page 125\]](#)

[Deleting media definitions \[Page 126\]](#)

Defining Media

The medium recommended for [data backups \[Page 27\]](#) (full and incremental) is tape. Data and pages may also be backed up to pipes or files.

[Log backups \[Page 31\]](#) are always to version files.

Using files as the [backup medium \[Page 118\]](#) only makes sense if they are written to tape or cassette afterwards.

You should only use pipes if you are working with an external backup tool.

You have the option of defining [single backup media \[Page 122\]](#). The system performs its backup operation to this medium. Where the capacity of the medium is too small for the backup that has started, the system asks for a continuation medium.

If you wish to back up at a higher speed, you can back up to a number of media in parallel. For this purpose you define a [group of parallel media \[Page 123\]](#) which are identified by a single, grouped media name.

You can define a backup medium both before and during a backup.

Defining a Single Backup Medium

Defining a Single Backup Medium

1. Choose *Instance* → *Configuration* → *Backup Media*
2. Then choose *BackupMedia* → *New* → *Medium*.
3. Choose *General* and make the following entries:

Name :	Freely definable name to identify the medium
Location:	Define the location to which backups are to be made. With files and pipes, we recommend specifying the absolute path to the backup medium.
Device Type:	Select the type of backup medium: File, tape, or pipe. If you back up to tape under the UNIX operating system you need to set the device driver correctly. Use rewind tapes for backups. Under the Windows NT operating system, the system sets the device driver automatically. If you back up the log to a file, the Database Manager (DBMGUI) automatically generates version files. Log autosaves cannot be made to tape.
Backup Type:	Select the type of backup you want (data backup, page backup [Page 27] , interactive log backup [Page 33] or automatic log backup [Page 32])

4. Choose *Extended* and make the following entries:

Size:	The size of the backup medium (in KB)
Overwrite:	Check <i>Overwrite</i> to use the overwrite mode for the medium.
Autoloader:	Check <i>Autoloader</i> to use a tape device with autoloader facilities.
OS command:	You can enter any operating system commands appropriate for backup to tape now.

Defining a Group of Parallel Media

To allow you to [backup \[Page 116\]](#) at a higher speed, you have the option of backing up to a number of media in parallel. For this purpose you define a group of parallel media under a single medium name. A backup which is going to be made to a number of media in parallel should have this medium name assigned to it.

The maximum number of parallel media in a group is determined by the value set in the parameter [MAXBACKUPDEVS \[Page 82\]](#).

When you [restore \[Page 134\]](#), you can import the data from this type of backup in parallel. The value set in the parameter MAXBACKUPDEVS also determines from how many media you can import in parallel.

Procedure

1. Choose *BackupMedia → New → Media Group*.
In the Backup - Media Manager window, the system creates the media group New Group in the form of a directory whose media type is Parallel.
2. Give the media group any name. You can change this name at a later stage using *BackupMedia → Rename*.
3. Double click on the directory to open it.
4. Then define each individual medium in the media group:
Choose *BackupMedia → New → Medium*.
Select *General* in the Backup - Medium window and make the following entries:

Name:	Freely definable name to identify the medium
Location:	Define the location to which backups are to be made. With files and pipes, we recommend specifying the absolute path to the backup medium.
Device Type:	Select the type of backup medium: File, tape, or pipe. If you back up to tape under the UNIX operating system you need to set the device driver correctly. Use rewind tapes for backups. On the Windows NT operating system, the system sets the device driver automatically. If you back up the log to a file, the Database Manager (DBMGUI) automatically generates version files. Log autosaves cannot be made to tape.
Backup Type:	Select the type of backup you want (full data backup [Page 27] , incremental data backup [Page 27] , interactive log backup [Page 33] or automatic log backup [Page 32])

Select *Extended* in the Backup - Medium window and make the following entries:

Size:	Size of backup medium
Overwrite:	Check the Overwrite field when you want to use the overwrite mode for the medium (only possible with medium type File).

Defining a Group of Parallel Media

Autoloader:	Check <code>Autoloader</code> to use a tape device with autoloader facilities.
OS command:	You can enter any operating system commands appropriate for backup to tape now.

Changing Media Definitions

Procedure

1. Choose *Instance* → *Configuration* → *Backup Media*
A list of the backup media already defined is displayed.
2. Select the medium you want.
3. Follow the appropriate description below:

Change media name

Choose *BackupMedia* → *Rename* and make your changes.

Press the Enter key to save your changes.

Change the properties of a medium

Choose *BackupMedia* → *Edit* and make your changes.

Name:	Freely definable name to identify the medium
Location:	Define the location to which backups are to be made. With files and pipes, we recommend specifying the absolute path to the backup medium.
Device Type:	Select the type of backup medium: File, tape, or pipe. If you back up to tape under the UNIX operating system you need to set the device driver correctly. Use rewind tapes for backups. Under the Windows NT operating system, the system sets the device driver automatically. If you back up the log to a file, the Database Manager (DBMGUI) automatically generates version files. Log autosaves cannot be made to tape.
Backup Type:	Select the type of backup you want (data backup, page backup [Page 27] , interactive log backup [Page 33] or automatic log backup [Page 32])

Save your work.

Deleting Medium

Choose *BackupMedia* → *Delete* and confirm with *OK*.

Deleting Media Definitions

Deleting Media Definitions

1. Choose *Instance* → *Configuration* → *Backup Media*
You are shown a list of the media available.
2. Select the medium you wish to delete.
3. Select *BackupMedia* → *Delete* and then OK to delete the medium.

Backup Processes

Depending on the technical requirements, the following backup processes are possible:

- [Saving to a single medium \[Page 128\]](#)
- [Saving to a group of parallel media \[Page 130\]](#)

Where the capacity of the backup medium selected is too small for the backup, continuation media have to be used.

- [backups to automatic tape loader \[Page 132\]](#)
- [backups to manually changed media \[Page 133\]](#)

If the capacity of the medium is sufficient for the backup, no intervention of an operator is required during the [backup \[Page 116\]](#).

The Database Manager (DBMGUI) asks for further [backup media \[Page 118\]](#) when the first medium has been completely filled but the backup has not been completed. For it to do so, the media capacity required must be stated when [defining the medium \[Page 121\]](#), or the tape device must detect the end of the tape.

Saving to a Single Medium

Saving to a Single Medium



If possible, it is always preferable to [back up to a group of parallel media \[Page 130\]](#) rather than backing up to a single medium plus continuation media. Parallel backup of this kind is not however possible for [log backups \[Page 31\]](#).

Prerequisites

Only one backup device is available.

You have defined the backup medium to be used for the backup. Where you wish to [back up to an automatic tape loader \[Page 132\]](#), you must specify this explicitly when making your media definition.



Under normal circumstances, you should not backup database instances if they are in `WARM` mode.

You can also backup database instances in `COLD` mode, but only in exceptional circumstances. However, if you backup in `cold` mode, you no longer have the choice between backing up for `recovery` (full data backup without checkpoint) and backing up for `migration` (full data backup with checkpoint) because there is no real guarantee that the database instance was shutdown in the correct way (*Instance* → *Shutdown* → *Cold* → *OK*, **and not** → *Quick*).

Procedure

1. Choose *Instance* → *Backup* and the required backup type:
 - *Complete* (full backup)
 - *Incremental* (incremental backup)
 - *Log* ([interactive log backup \[Page 33\]](#))
 - *AutoLog* ([automatic log backup \[Page 32\]](#))

The list of the media defined for the selected backup type appears in the output display. If no media have been defined yet, you must now define them ([Defining a Single Backup Medium \[Page 122\]](#)).

2. Select the medium to be used for the backup.
3. Check the details of the medium. Change them if required ([Changing Media \[Page 125\]](#)).
4. In the [menu bar \[Page 66\]](#) at the top of the screen, choose *Backup* and the required backup type.
Specify whether the backup is for `recovery` ([data backup without checkpoint \[Page 29\]](#)) or `migration` ([data backup with checkpoint \[Page 30\]](#)) purposes.
5. If you have selected a **data backup or interactive log backup**, now choose *Backup* → *Start*. The backup action now starts.
If the capacity of the backup medium specified is not large enough for the backup and you are not using an automatic tape loader, the Database Manager DBMGUI asks you to specify and insert a new medium as soon as the previous medium has been filled ([Backups to Manually Changed Media \[Page 133\]](#)).

Saving to a Single Medium

Once the backup has been successfully completed, you see the message *Backup completed*.

If you want to activate the **automatic log backup**, choose *Backup* → *AutoLog on*.

Saving to a Group of Parallel Media

Saving to a Group of Parallel Media

To ensure maximum data throughput, the database instance can be backed up to a several different media at the same time. You can set the maximum number of tape devices which can be used at the same time using parameter [MAXBACKUPDEVS \[Page 82\]](#). The use of up to 32 tape devices allows you to reduce backup times considerably.

Parallel backing up is only possible for [data backups \[Page 27\]](#).



Under normal circumstances, you should not backup database instances if they are in `WARM` mode.

You can also backup database instances in `COLD` mode, but only in exceptional circumstances. However, if you backup in `cold` mode, you no longer have the choice between backing up for `recovery` (full data backup without checkpoint) and backing up for `migration` (full data backup with checkpoint) because there is no real guarantee that the database instance was shutdown in the correct way (*Instance* → *Shutdown* → *Cold* → *OK*, and not → *Quick*).

Procedure

6. Choose *Instance* → *Backup* and the selected backup type:
 - *Complete* (full data backup)
 - *Incremental* (incremental data backup)

A group of parallel media cannot be used for log backups.

A list of the media already defined for the selected backup type appears on the screen. If no media have been defined yet, you must now define them ([Defining a Group of Parallel Media \[Page 123\]](#)).

7. Select the medium to be used for the backup.
8. Check the details of the medium. Change them if required ([Changing Media \[Page 125\]](#)).
9. Select *BackupSave* → *Next Step* on the [menu bar \[Page 66\]](#) at the top of the screen. Specify whether the backup is for `recovery` ([data backup without checkpoint \[Page 29\]](#)) or `migration` ([data backup with checkpoint \[Page 30\]](#)) purposes.
10. If the capacity of the group of parallel media is not large enough for the backup, Database Manager DBMGUI asks you to specify and insert a new medium as soon as any of the media has been filled ([Backups to Manually Changed Media \[Page 133\]](#)).

Once the backup has been successfully completed, you see the message `Backup completed`.

Activating and Deactivating the Automatic Log Backup

Use

[Automatic log backup \[Page 32\]](#) is used in production systems to ensure data is secured.

Procedure

1. In the [menu bar \[Page 68\]](#) of the Database Manager (DBMGUI), choose *Instance* → *Backup* → *AutoLog on/off*.
A list of all of the media available for log backups is presented. If you have already used the automatic log backup function, the medium used is already selected. You can choose a different medium or create a new one ([Defining a Single Backup Medium \[Page 122\]](#)).
2. Then choose *Instance* → *Backup* → *AutoLog on* to activate the automatic log backup function. This function remains activated until you deactivate it by choosing *Instance* → *Backup* → *AutoLog off*.



Please note that the system switches the activated automatic log backup to *OFF* when you carry out an [interactive log backup \[Page 33\]](#) or [restart the database instance \[Page 113\]](#).

Backups to Automatic Tape Loader

Backups to Automatic Tape Loader

The Database Manager (DBMGUI) supports automatic tape loaders.

Interactive Procedure

If you want to use an automatic tape loader for [backups \[Page 116\]](#), select *Extended* when defining the medium ([Defining a Single Backup Medium \[Page 122\]](#), [Defining a Group of Parallel Media \[Page 123\]](#) or [Changing Media \[Page 125\]](#)) *Extended* and select the `Autoloader` option.

- On the Windows NT operating system, this defines the automatic loading of tapes.
- If you are using any other operating system, use the appropriate command for loading tapes. To do so, choose *Extended* and enter the operating system command in the *OS Command* field.

After inserting the tape cartridge in the autoloader, the first tape must be selected manually. When the end of this tape is reached, the autoloader will then take the next tape available.

Make sure there are enough tapes before you start.

At the end of the backup, the tape device shows the number of tapes written. The tapes should be marked with the [backup label \[Page 119\]](#) which was displayed and confirmed at the start of the backup.



Before a backup you must check that none of the tapes is write-protected because that would stop the entire backup process.

Backing Up to Manually Changed Media

Prerequisites

You have started a [backup to a single medium \[Page 128\]](#) or a [backup to a group of parallel media \[Page 130\]](#).

The capacity of the medium or media inserted is not sufficiently large for the [backup \[Page 116\]](#). The advancing bar showing the progress of the backup stops before reaching 100%.

A template appears asking you to insert the continuation medium.

Procedure

Continuation medium is tape:

1. Insert the next tape.
2. Make sure that the tape just written has been stored in a safe place and that the right tape has been inserted.
3. Choose *Start*.



We recommend you specify the exact capacity of the medium when defining the medium. Provided the tape device driver gives a reliable indication when the end of the tape is reached, the Database Manager (DBMGUI) reacts as described even when *Size* = 0.

If the end of the tape is not recognized correctly, the message *Writing Error* is displayed. Even when this happens you can continue as described in steps 1-3.

Continuation medium is file:

1. Under *New Location*, enter the name and full path of the file to which the backup should be made.
2. Choose *Backup* → *Continue*.

Repeat the procedure appropriate to the type of medium until the backup has been completed.

As long as the backup is being performed, no other backup can be started. A backup is considered to have been completed when the message *Backup completed* appears.

Restore

Restore

The Database Manager supports the restore or recovery of the database after hardware faults. The last consistent state of the database is restored or recovered. A prerequisite for this is that all the data specified in the [backup history \[Page 146\]](#) is available.

To obtain the highest possible throughput of data, full and incremental [data backups \[Page 27\]](#) can be retrieved from a number of media simultaneously. The number of parallel media in this case does not depend on the number of parallel media used to make the backup originally. Even a backup made to single medium plus continuation media can be restored in parallel.

You can set the maximum number of backup devices that can be retrieved from simultaneously with parameter [MAXBACKUPDEVS \[Page 82\]](#). The use of up to 32 tape devices allows you to reduce backup and restore times considerably.



[Log backups \[Page 31\]](#) cannot be restored from parallel media.

The Database Manager (DBMGUI) always suggests the quickest way of effecting a restore. Data backup(s) are retrieved at the outset in this case and these are followed firstly by incremental backups and then by log backups.

Restore to the Last Complete Backup Made

Prerequisites

The files [dbm.knl \[Page 156\]](#) and [dbm.mdf \[Page 156\]](#) in the run directory of the database instance have not been corrupted, which means the [Backup history \[Page 146\]](#) is complete.

The database instance is in `COLD` mode.

Procedure

1. Choose *Instance* → *Recovery* → *Database*.
2. Select the option *Restore last backup*.
In this screen, you can specify the time by which the restoration should be completed by selecting the option *Restore by a specific time*. The system default is the current time. If you do not change the time proposed by the system, the backups are restored completely and with all the changes they contain.
3. Choose *RecoveryDatabase* → *Next Step*.
The system displays the last complete backup made at the time you specified earlier (including any incremental data backups and log backups that have been made). You can now deselect in the list any incremental data backups that are not available or that have been damaged. The Database Manager (DBMGUI) then shows the appropriate log backups in place of the page backups. The system tells you when mount the data carrier.
4. Choose *RecoveryDatabase* → *Start*.
The complete data backup is restored.
If you then need to restore any incremental data backups, the Database Manager (DBMGUI) prompts you to confirm each one.
If you then need to make log backups, the Database Manager (DBMGUI) prompts you to confirm that you want to restore all displayed backups.
Choose *RecoveryDatabase* → *Restart* to restart the database instance.



When the consecutive backups are retrieved, the Database Manager prompts you to insert the next backup medium each time, unless you are [restoring from an automatic tape loader \[Page 143\]](#).

You can if you wish stop the restore process and resume it later on ([Continue Interrupted Restore \[Page 144\]](#)).

Restore to a Backup from the Backup History

Restore to a Backup from the Backup History

Prerequisites

The files [dbm.knl \[Page 156\]](#) and [dbm.mdf \[Page 156\]](#) in the run directory of the database instance have not been corrupted, which means the [Backup history \[Page 146\]](#) is complete.

The database instance is in `COLD` mode.

Procedure

1. Choose *Instance* → *Recovery* → *Database*.
2. Select the option *Restore a specified backup from history*.
In this screen, you can specify the time by which the restoration should be completed by selecting the option *Restore by a specific time*. The system default is the current time. If you do not change the time proposed by the system, the backups are restored completely and with all the changes they contain.
3. Choose *RecoveryDatabase* → *Next Step*.
You are now shown the entire [backup history \[Page 146\]](#).
4. Choose the required full data backup and then click *RecoveryDatabase* → *Next Step*.
The system displays the selected backup and the associated page and log backups. You can now deselect in the list any incremental data backups that are not available or that have been damaged. The Database Manager (DBMGUI) then shows the appropriate log backups in place of the page backups. The system tells you when mount the data carrier.
5. Choose *RecoveryDatabase* → *Start*.
The complete data backup is restored.
If you then need to restore any incremental data backups, the Database Manager (DBMGUI) prompts you to confirm each one.
If you then need to make log backups, the Database Manager (DBMGUI) prompts you to confirm that you want to restore all displayed backups.

Once the restore has finished, the database instance is in `OFFLINE` mode.



When the consecutive backups are retrieved, the Database Manager prompts you to insert the next backup medium each time, unless you are [restoring from an automatic tape loader \[Page 143\]](#).

You can if you wish stop the restore process and resume it later on ([Continue Interrupted Restore \[Page 144\]](#)).

Restoring an Existing Database Instance

Prerequisites

The files [dbm.knl \[Page 156\]](#) and [dbm.mdf \[Page 156\]](#) in the run directory of the database instance have not been corrupted, which means the [Backup history \[Page 146\]](#) is complete.

The database instance is in `COLD` mode.

Procedure

Select *Instance* → *Install* to launch the Database Wizard. This Database Wizard guides you through the entire procedure for restoring an existing database instance.

1. Specify under which name the existing database instance is registered in the Database Manager (DBMGUI).
2. Choose *Next*.
The Database Manager displays a message saying that a database instance with this name already exists.
3. Choose *Reinstall*. This particular database instance is then restored. The Database Manager uses the [parameters \[Page 80\]](#) of the old version of the database instance as the default values for the new version. All data from the old version will be lost.
4. Enter the users **DBM** and **DBA** and assign passwords for [DBM \[Page 20\]](#) and [DBA \[Page 21\]](#) and choose *Next*.
The Database Manager features a *Use current parameters* option for creating the initial parameter file.
5. Choose *Next*.
The system automatically proposes the parameter values from the old version of the database instance. You can adjust these values to your requirements. Select the required parameters and then choose *Edit*.
It now appears in the bottom part of the window. Next to the parameter you will see an explanation of what the parameter does and, in some cases, how it is calculated.

Enter the new value for the parameter in the `New Value` field.

Save your work.

The new value appears in the *New Value* column. It is stored in the internal data structures and becomes effective once the [database instance is restarted \[Page 113\]](#).

6. Choose *Next*.
When you exit the input screen, the parameters are checked following the rules stored on the DBM server. You may be asked to make and confirm changes before you are permitted to exit the input screen.
For the [log devspaces \[Page 17\]](#) and the [data devspaces \[Page 16\]](#), the system automatically proposes the configuration of the old version of the database instance. You can adjust these values to your requirements. Bear in mind the parameters set in the previous step. Select a devspace and then choose *Edit*. Enter the size and ID (or absolute path) of the devspace and confirm your entries. Repeat for each devspace.
7. Choose *Next* and *Restore instance*.
8. Select *Install*.
A new database instance with the parameters set in step 5 is installed (complete data backup

Restoring an Existing Database Instance

is used).

The system will prompt you to give the database instance a new registry name in the Database Manager (DBMGUI). This name must be unique throughout the Database Manager (DBMGUI).

9. Confirm your entries.

10. Select *Close* to close the Database Wizard.

This takes you back to the [initial screen \[Page 66\]](#) of the Database Manager (DBMGUI).

Once the restore procedure is complete, the database instance is in `WARM` mode.



When the consecutive backups are retrieved, the Database Manager prompts you to insert the next backup medium each time, unless you are [restoring from an automatic tape loader \[Page 143\]](#).

You can if you wish stop the restore process and resume it later on ([Continue Interrupted Restore \[Page 144\]](#)).

Restoring the Indices After a Database Restore

Use

For optimization reasons, the indices contained in the log are not restored with a database instance. Indices that have not been restored are marked `BAD` and can be restored explicitly by following the procedure described here.

Prerequisites

The database instance is in `WARM` mode.

Procedure

Choose *Instance* → *Recovery* → *Index*.

You can choose the tables for which the indices marked `BAD` are to be displayed. To do so, enter appropriate search arguments under `Owner`, `Table Name` and `Index Name`.

Owner	Owner
Table Name	Name of the table
Index Name	Name of the index

Then choose *Select*. A list of the indices that match your criteria is then displayed.

Select the indices you want to restore in the list. If you want to restore all of the indices, choose *Mark All*.

Choose *Execute* to restore the indices.



It is advisable to restore the indices during periods where the load on the system is lower.

Restoring Without a Backup History

Restoring Without a Backup History

Prerequisites

The database instance is in `COLD` mode.

Procedure

1. Choose *Instance* → *Recovery* → *Database*.
2. Select the option *Restore a medium*.
3. In this screen, you can specify the time by which a data medium should be restored by selecting the option *Restore by a specific time*. The system default is the current time. If you do not change the time proposed by the system, the backups are restored completely and with all the changes they contain.
4. Choose *RecoveryDatabase* → *Next Step*.
A list of all of the available media is presented.
5. Select the medium you wish to restore.
6. Choose *RecoveryDatabase* → *Next Step*.
The system tells you when mount the data carrier.
7. Choose *RecoveryDatabase* → *Start* to start the recovery process.
If you then want to restore other media without a backup history, proceed as outlined in step 2 onward.
8. Choose *RecoveryDatabase* → *Restart* to restart the database instance.



You can if you wish stop the restore process and resume it later on ([Continue Interrupted Restore \[Page 144\]](#)).

Recovering a Mirrored Devspace

Use

If your [devspaces \[Page 14\]](#) are mirrored, one of the devspace mirrors may fail. In this case, the database continues to operate but only the part of the devspace pair that is intact can be accessed. The faulty mirror devspace is marked as a `Bad Devspace`.

You use this function to recover a damaged mirror devspace. This can be a data, system, or log devspace.



To ensure the [availability \[Page 65\]](#) of your system, however, it is advisable to mirror log devspaces only. We recommend RAID 5 configurations to protect your data and system devspaces.

Recovering a damaged log devspace mirror

When you trigger the recovery of a log devspace, it is only reinitialized. In other words, the data is not copied from the intact devspace. Following this, both log devspaces are written to sequentially. The devspace originally marked `BAD` must be written to fully before the contents of both devspaces are identical again and both devspaces can be read.

The defective devspace remains marked `BAD` during the entire reintegration phase.

Prerequisites

The database instance is in `WARM` or `COLD` mode.

Procedure

Choose *Instance* → *Recovery* → *Devspaces*.

Select the `BAD` devspaces that are to be recovered.

Choose *RecoveryDevspaces* → *Reintegrate* to start initializing the mirror devspace.

Recovering a damaged data devspace mirror

When a damaged data devspace is recovered, the intact data devspace is copied to the repaired data devspace. Following this, both data devspaces operate normally again in mirrored mode.

Prerequisites

The database instance is in `WARM` mode.

Procedure

Choose *Instance* → *Recovery* → *Devspaces*.

Select the `BAD` devspaces that are to be recovered.

Choose *RecoverDevspaces* → *Reintegrate* to start copying the data to the mirror devspace.

Recovering a Mirrored Devspace

Restore from Automatic Tape Loader

Prerequisites

For the retrieval operation to be successful, all the tapes which form part of the backup must be present in the automatic tape loader.

Procedure

Use the same media definition as for backing up ([Defining a Single Backup Medium \[Page 122\]](#)). Follow the individual steps as described for the relevant recovery procedure:

- [Restore to the last complete backup made \[Page 135\]](#)
- [Restore to a backup from the backup history \[Page 136\]](#)
- [Restore existing database instance \[Page 137\]](#)
- [Restore without a backup history \[Page 140\]](#)

Continuing an Interrupted Restore

Continuing an Interrupted Restore

Prerequisites

The database instance is in COLD mode.

One of several possible restore procedures was started but was canceled before completion

- [Restore to the last backup made \[Page 135\]](#)
- [Restore to a backup from the backup history \[Page 136\]](#)
- [Restore from a single medium \[Page 140\]](#)

Procedure

1. Choose *Instance* → *Recovery* → *Database*.
2. Choose the option
Continue restoring pages/log
3. Choose *RecoveryDatabase* → *Next Step*.
This displays a list of all relevant log backups made for data backups that have already been restored.
4. Choose *RecoveryDatabase* → *Replace* to continue the recovery operation. You can follow the progress of the restore operation and its completion in the screen display.
5. *Close* closes the Restore Manager.

Requesting Information

The initial screen of the Database Manager contains an overview of the information and statistics on the important database functions in the current database instance.

By double-clicking on a database instance in the [list of registered database instances \[Page 71\]](#), you can display a more detailed overview of the statistics and important functions in the current database instance.

If you only want to view information on a certain area, choose *Instance* → *Information* and then the selected area. Information is available on the following topics:

[Backup history \[Page 146\]](#)

[Cache information \[Page 147\]](#)

[Data devspaces \[Page 148\]](#)

[Read and write operations \[Page 149\]](#)

[Log devspace and Logwriter \[Page 150\]](#)

[Locks and lock requests \[Page 151\]](#)

[User sessions \[Page 152\]](#)

[Versions \[Page 153\]](#)

Backup History

Backup History

The system writes the backup history to the file [dbm.knl \[Page 156\]](#). This file is stored in the [RUNDIRECTORY \[Page 90\]](#) of the database instance.

Recorded chronologically in the backup history is information on all the backup and restore actions which have been performed.

Procedure

Choose *Instance* → *Information* → *Backup History*

You will see the following items of information on each action:

Label	Backup Label
Action	The action (backup or recover) and the mode in which it took place.
Beginning	Time when the action began
End	Time when the action ended
Result	Result
Media Name	Name of medium
Log Required	Log backups required
Size (KB)	Size (KB)
Size (Pages)	Size (pages)
Volumes	Volumes
Next Log Page	Log page
From Page	First page
To Page	Last page
Last savepoint	Last savepoint
First commit	First commit (first completed transaction)
Last commit	Last commit (last completed transaction)
SysKey	System key



Choose *Instance* → *Information* → *Backup History* and then *BackupHistory*→*Columns* to define your own personal display variant for the backup history.

Cache Information

Select *Instance* → *Information* → *Caches*.

Name	Cache name [Page 55]
Accesses	Total number of accesses
Successful	Number of successful accesses
Unsuccessful	Number of unsuccessful accesses
Hit Rate (%)	Hit rate (number of hits as a percentage of the total number of accesses)

The hit rates for the caches show whether they were configured to be sufficiently large. The hit rates for the [data cache \[Page 60\]](#) and [converter cache \[Page 57\]](#) are particularly important.

The data cache hit rate should be at least 99% and the converter cache hit rate should be at least 98%.

Lower hit rates can occur for a short time, for example when tables are read for the first time, or when repeated table scans are being performed and the table does not fit into 10% of the data cache.

If the values are permanently lower than the recommended percentage, this is a sign of poor performance and the setting of the configuration [parameter \[Page 80\]](#) DATA_CACHE or CONVERTER_CACHE must be increased.

Data Devspaces

Data Devspaces

Choose *Instance* → *Information* → *Data*.

Name	Name of data devspace
Value	Current value of data devspace

The system presents the following data devspace information.

Total space (KB)	Total size of the data devspace in KB
Max. persistent space (KB)	Maximum persistent data devspace in KB
Used space (KB)	Used data devspace in KB
Used space (%)	Used data devspace as percentage
Free space (KB)	Free data devspace in KB
Free space (%)	Free data devspace as percentage
Updated perm pages	Updated permanent pages

Read and Write Operations

Choose *Instance* → *Information* → I/O.

The database server keeps statistics on the current number of both physical and logical read and write accesses since the [database instance was last started \[Page 113\]](#).

Name	Naming
Logical Reads	Number of logical read commands
Logical Writes	Number of logical write commands
Physical Reads	Number of physical read commands
Physical Writes	Number of physical write commands

The details are displayed for the following devspaces:

Catalog	Denotes the data devspace used by the data dictionary of the database system. Frequent writing to this area is a sign of modifications made to the database design.
Permanent Data	The proper data devspace for permanent data.
Temporary Data	A data devspace on the disk intended for temporary use; it is required, for example, for the generation of selected datasets.
Long Data	LONG columns
Leaf, Level 1, Level 2, Level 3	Data tree structure The database organizes the stored data as B* trees.

SUM shows the total number of various accesses.

Log Devspace and Logwriter

Log Devspace and Logwriter

Choose *Instance* → *Information* → *Log*.

You are provided with information on the state and size of the log devspace and the activities of the logwriter.

Log mode	Mode in which the log is operated
Max. size	Maximum storage space available in the log devspace
Save Segment size	Size of a segment
Used size (KB)	Used space in KB
Used size in %	Used space as percentage
Not saved (KB)	Size of log devspace not yet saved, in KB
Not saved (%)	Size of log devspace not yet saved, as percentage
Log since last data backup (KB)	Number of log pages written since the last data backup
Completed segments	Log segments filled
Savepoints written	Number of savepoints written since last log backup
Checkpoints written	Number of checkpoints written since last log backup
Physical reads	Read activities in the log devspace since the last restart of the database system
Physical writes	Write activities in the log devspace since the last restart of the database system
Queue size	Current size of log queue
Queue allocated	Current log queue allocation
Queue entries	Maximum size of log queue
Queue overflows	Number of log queue overflows
Group commits	Number of group commits
Waits for logwriter	Number of wait states for log write operations
Max. waits	Maximum number of wait states per log page
Avg. waits	Average number of wait states per log page
OMS Log used	Log devspace used by Object Management System (OMS)
OMS min. free pages	Minimum number of free pages for Object Management System (OMS)

Locks and Lock Requests

Select *Instance* → *Information* → *Locks*.

You are given information on the current locks and lock requests in the database server.

Lock List Statistics

The following information is displayed:

- average number of locks held and requested
- maximum number of locks held and requested
- number of collisions and escalations that have occurred
- number of current row or table locks.

Lock State

The following information is displayed:

- current locks
- imminent lock requests with detailed description
- relevant tables and rows
- type of lock
- lock holder and the lock holder's terminal

Share locks	These protect used data against modifications while being accessed. Other users can only read-access data which is locked in this way.
Exclusive locks	These prevent other users from accessing the same data. Not even read-access is possible. Therefore, if exclusive locks are held for a lengthy period, they can be disadvantageous for the other users. Locks are released again by a <i>Commit</i> or <i>Rollback</i> or by a <i>Lock Timeout</i> . Configurable parameters REQUEST_TIMEOUT and LOCK_TIMEOUT are used to resolve blocking and deadlock situations.
REQUEST_TIMEOUT	This indicates the time a task may wait for the setting of a requested lock before it is canceled.
LOCK_TIMEOUT	This indicates the time a task may hold a lock without activities before it can be rolled back automatically (ROLLBACK).

User Sessions

User Sessions

Choose *Instance* → *Information* → *Sessions*.

All database users currently connected to a database are shown.

The following information is displayed:

User ID	Name of user who has logged on
Terminal ID	A terminal identification specific to the particular operating system.
Task ID	Task number
Session ID	User's session number
Remote Host	Computer name within the network
Catalog Cache Size	Size of catalog cache available for user



In special cases, it may be necessary to stop a user process.

You can use *Select* → *Kill* to remove the user and stop the user session. The current transaction is rolled back.

Versions

Select *Instance* → *Backup* → *History*

You are shown which versions of the

- Database kernel
- Runtime environment
- Database Manager (DBMGUI)

are current.

Options for Diagnosing Problems

Options for Diagnosing Problems

In the event of errors or performance problems, the Database Manager (DBMGUI) offers you several different options for diagnosing the problem.

The DBMGUI provides the following functions:

[Kernel Trace Function \[Page 155\]](#)

[Database Manager \(DBMGUI\) Files \[Page 156\]](#)

[Database Console Information \[Page 157\]](#)

[Use of the Command Line Version of the Program \[Page 158\]](#)

Diagnosis with Kernel Trace Function

The Database Manager also gives you the option of switching on the kernel trace function if errors occur.



You should use this function when asked to do so by your SAP DB support organization. We would not recommend you switch on the kernel trace function unless it is really necessary, because it seriously affects the performance of the database instance.

The kernel trace documents all the reactions of the database kernel to database statements. This also means that it can be used not only to trace errors that occur when statements are processed but also to provide a more exact classification of inconsistencies caused, for example, by hardware errors. If the kernel trace is switched on, you should be running the database instance with the smallest load possible, and only those actions needed to reproduce the error should be performed on the database instance.

Switch off the kernel trace function as soon as the actions needed for the analysis have been logged.

Procedure

1. Choose *Instance* → *Check* → *Kernel Trace*.

In the list, select the kernel trace options you want by checking the relevant box.

2. Choose *KernelTrace* → *Selected Items ON* to activate the kernel trace function.

Select *KernelTrace* → *Selected Items OFF*, to switch off the kernel trace.

Reading the Log Files of the Database Manager

Reading the Log Files of the Database Manager

You can view a selection of log files that have been written by the Database Manager and stored in the [Run Directory \[Page 90\]](#) of the relevant database instance.

Before a Run Directory is defined at the time of installation, files are written to directory WRK, which is a subdirectory of the database's installation directory.

Procedure

Choose *Instance* → *Check* → *Files*.

Naming	File	Contents
Database Manager Protocol	dbm.prt	Command history of the write operations on the Database Manager
Database Manager Media	dbm.mmm	Details of currently defined backup media
Installation Protocol	dbm.ins	The log file of the last database instance installation.
Backup History	dbm.knl	History of backup and restore operations performed. The file is written from the database kernel.
Utility Statements	dbm.utl	The logs of the previous internal utility requests (save and restore requests, add devspace, update statistics). The file is written from the database kernel.
Backup Media History	dbm.mdf	History of media definitions
External Backup History	dbm.ebf	History of external backup labels from backups that were made using external tools
Database Messages	knldiag	Console log file
Database Messages (OLD)	knldiag.old	Copy of the previous version of the console protocol
Database Errors	knldiag.err	Contains the last error message written to file knldiag.
Database Trace	knltrace	With kernel trace enabled, all the activities of the database are logged in this file. The file is used for support purposes.

Database Console Information

The Database Manager (DBMGUI) enables you to see how much of the operating system resources the database system is using, how the database sessions have been distributed among the operating system processes/threads, and the status of the active database sessions. There are additional functions available, but these should be reserved for support employees and developers.

Prerequisites

The database instance is in `WARM` or `COLD` mode.

Procedure

1. Choose *Instance* → *Check* → *Server*.
2. Select the required information type and then choose *CheckServer* → *View*.

Use of the Command Line Version of the Program

Use of the Command Line Version of the Program

In the Database Manager (DBMGUI) program, you can also use the command line version of the program called the Database Manager (DBMCLI).







Choose *Instance* → *Command Line* .

Enter the desired DBMCLI command in the command field and select the menu entry *Command* → *Execute*.

See also:

R/3 Database Manager (DBMCLI)

Database Icon Legend

	Status of database instance is not known.
	Database instance is busy (e.g. is refreshing the mode [Page 115]).
	Server component of Database Manager (DBMGUI) has not started.
	Server component of Database Manager (DBMGUI) has started but has not yet made a connection to the database instance.
	Database instance is still registered with the Database Manager (DBMGUI) but is no longer present on the server.
	Database instance is already registered.

Updating the Database Software

Updating the Database Software

Procedure

1. Choose *Instance* → *Restart* → *Warm* to start the database instance.

The first time the Database Manager (DBMGUI) is started after there has been a change in the version of the database software, it automatically adjusts the database instance kernel parameters to the new version of the software.

The message `Migration of Param File` is displayed.

2. To update the system tables, choose *Instance* → *Configuration* → *Upgrade System Tables*.

Result

The Database Manager creates a copy of the parameter file each time a change is made. The serial number extension indicates the version, whereby 01 is the latest version:

UNIX: `/usr/spool/sql/config/<database_name>.<running_number>`

Windows NT/Windows 95: `%DBROOT%\CONFIG\<database_name>.<running_number>`

Should errors occur while loading the parameters, the operation can be undone by restoring the original parameter file. To do so, copy the file `<database_name>.<running_number>` (copy of the parameter file) to the `<database_name>` (active parameter file).



If a fault is found in the software when the database instance is [restarted \[Page 113\]](#) after the update, talk to Local Support.