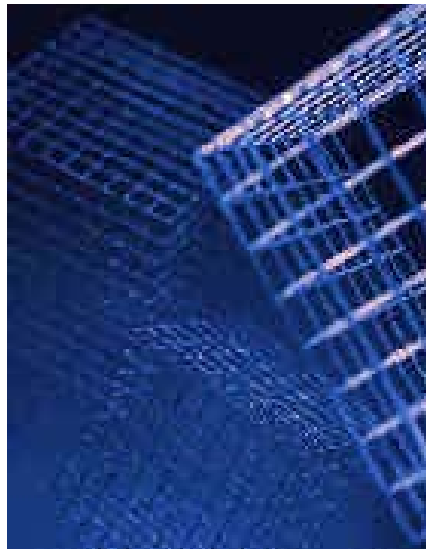


User Manual: SAP DB



Version 7.3








Copyright

© Copyright 2002 SAP AG.

Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.1 or any later version published by the Free Software Foundation.

For more information on the GNU Free Documentaton License see
<http://www.gnu.org/copyleft/fdl.html#SEC4>.

Icons

Icon	Meaning
	Caution
	Example
	Note
	Recommendation
	Syntax

Typographic Conventions

Type Style	Description
<i>Example text</i>	Words or characters that appear on the screen. These include field names, screen titles, pushbuttons as well as menu names, paths and options. Cross-references to other documentation
Example text	Emphasized words or phrases in body text, titles of graphics and tables
EXAMPLE TEXT	Names of elements in the system. These include report names, program names, transaction codes, table names, and individual key words of a programming language, when surrounded by body text, for example, SELECT and INCLUDE.
Example text	Screen output. This includes file and directory names and their paths, messages, names of variables and parameters, source code as well as names of installation, upgrade and database tools.
Example text	Exact user entry. These are words or characters that you enter in the system exactly as they appear in the documentation.
<Example text>	Variable user entry. Pointed brackets indicate that you replace these words and characters with appropriate entries.
EXAMPLE TEXT	Keys on the keyboard, for example, function keys (such as F2) or the ENTER key

User Manual: SAP DB	12
Architecture: SAP DB.....	12
Database Instance	13
Thread.....	14
User Kernel Thread (UKT)	14
Conv Scanner	14
Data Writer	15
Log Writer	15
Server Tasks	15
Timer Task.....	15
Trace Writer Task.....	15
User Task	16
Utility Task	16
Special Thread	16
Coordinator.....	16
Dev Thread.....	16
Requester	17
Temporary Dev Thread	17
Timer.....	17
Operating-System-Dependent Special Threads	17
Clock Thread	17
Console Thread	18
Cache.....	18
Catalog Cache	18
Converter Cache	18
Data Cache	19
File directory cache.....	19
Free Block Management (FBM) Cache	19
Log Cache.....	19
Devspace.....	19
System Devspace	20
Data Devspace.....	20
Log Devspace	20
Database Instance Type	21
SAP DB OLTP	21
liveCache	21
SAP DB Document Server.....	21
SAP DB OLAP	22
SAP DB E-Catalog.....	22

SAP DB Versions and Database Instance Types	22
Operating System Platform	23
Multiprocessor Configuration	23
User Concept	24
SAP DB User Classes	25
Database Manager Operator (DBM Operator)	25
User Authorizations	25
DBM Server Authorizations	26
Default Authorizations for the First DBM Operator	26
Operating System User Authorizations	27
Database User	27
Database User Classes	27
Database System Administrator (SYSDBA)	27
DBA/DOMAIN	28
RESOURCE	28
STANDARD	28
User Groups	28
The Role Concept	29
User Data as Options	29
Options (C/C++ Precompiler)	30
Required Options	32
User Data and XUSER	33
Using XUSER	33
XUSER Data	34
Generating XUSER Data in the Background	35
Security Concepts	36
Availability	36
Security Requirements	36
Restartability	37
Backup Strategy	37
Data Backup	38
Data Backup with Checkpoint	39
Data Backups without Checkpoint	39
Consistent Data Backup	39
Saving Data Backups	40
Log Backup	40
Automatic Log Backup	41
Interactive Log Backup	41
Saving Log Backups	41
SAP DB Tools	42

Architecture: SAP DB Tools	42
Architecture of the Database Manager.....	43
Architecture of the Replication Manager	44
SQL Studio Architecture	45
Architecture of the SAP DB Web Tools	46
X Server	47
DBM Server	47
REPM Server.....	47
Web Server	48
Database Manager	48
Database Manager GUI.....	48
Options (DBMGUI)	49
Database Manager CLI.....	49
Options (DBMCLI).....	50
DBM Server Commands	51
Web DBM.....	51
Replication Manager	52
Options (REPMCLI).....	53
REPM Server Commands	53
SQL Studio: Introduction	54
SQL Studio.....	55
Options (SQL Studio).....	55
Web SQL	55
Directory Structure: SAP DB for SAP Systems	57
Variables.....	57
Distribution of the SAP DB Directories on the Hard Disk.....	57
Security Requirements	58
Performance Requirements.....	58
Example Configuration	59
Various Database Systems	59
SAP DB Directories	59
Instance Data	60
Programs That Are Independent of the Database Software Version	61
Libraries for the Client Run-time Environment.....	61
Programs That Are Dependent on the Database Software Version	62
Client Tools	62
Example: SAP DB Directory Structure.....	62
Display SAP DB Directories.....	63
Define SAP DB Directories	63
Directory Structure: SAP DB for Open Source	64

Variables.....	64
Distribution of the SAP DB Directories on the Hard Disk.....	64
Security Requirements	65
Performance Requirements.....	65
Example Configuration	66
Various Database Systems	66
SAP DB Directories	66
Display SAP DB Directories.....	67
Define SAP DB Directories	68
Database Parameters	68
BACKUP_BLOCK_CNT	69
CAT_CACHE_SUPPLY	69
CONVERTER_CACHE	69
DATA_CACHE	70
DATE_TIME_FORMAT	70
DEADLOCK_DETECTION.....	70
DEFAULT_CODE.....	70
INSTANCE_TYPE.....	70
JOIN_MAXTAB_LEVEL9	70
JOIN_MAXTAB_LEVEL4	71
JOIN_SEARCH_LEVEL	71
KERNELDIAGSIZE	71
KERNELVERSION.....	72
LOG_BACKUP_TO_PIPE.....	72
LOG_IO_QUEUE	72
LOG_MODE	72
LOG_SEGMENT_SIZE	72
LRU_FOR_SCAN.....	73
MAXARCHIVELOGS.....	73
MAXBACKUPDEVS	73
MAXCPU	73
MAXDATADEVSPACES	74
MAXDATAPAGES.....	74
MAXLOCKS.....	74
MAXRGN_REQUEST	74
MAXSERVERTASKS	74
MAXUSERTASKS	74
MP_RGN_LOOP	75
OPTIM_BUILD_RESLT	75
OPTIM_FETCH_RESLT	75

OPTIM_KEY_INV_RATE	75
OPTIM_MAX_MERGE	75
OPTIM_ORDERBY_IDX	76
OPTIM_OR_DISTINCT	76
REQUEST_TIMEOUT	76
RESERVED_REDO_SIZE	76
RESTART_SHUTDOWN	76
RUNDIRECTORY	77
SEQUENCE_CACHE	77
SESSION_TIMEOUT	77
UTILITY_PROT_SIZE	77
_DATA_CACHE_RGNS	78
_EVENT_ALIVE_CYCLE	78
_MAXEVENTS	78
_MAX_MESSAGE_FILES	78
_ROW_RGNS	78
_TAB_RGNS	79
_TRANS_RGNS	79
_TREE_RGNS	79
_UNICODE	79
SAP DB as UNICODE Database	80
UNICODE	80
Installing a UNICODE-Enabled Database	80
Setting Database Parameter _UNICODE	81
Setting Code Attribute UNICODE	82
UNICODE and SQL	82
Example 1	83
UNICODE in Programming Languages	85
Example 2	87
Data Management Using B* Trees	90
Concepts	91
Primary Key	91
Secondary Key	91
B* Tree	91
Root/Index Page	92
Leaf Page	93
Table Access	93
Table ID	93
B* Trees for Tables	94
B* Trees for Table with LONG Columns	94

B* Trees for Tables with Secondary Key	95
B* Trees for Tables with LONG Columns and Secondary Key	95
Table Access Using B* Tree	96
Table Access (SELECT) Using B* Tree	96
Table Access (INSERT) Using B* Tree	98
Table Access (DELETE) Using B* Tree	99
Table Access (UPDATE) Using B* Tree.....	100
Changes in the B* Tree Structure	100
Non-Uniform Distributions of Data Pages.....	101
SAP DB for Users of Version 6.1	102
Requirements for a Database System	102
SAP DB Improvements Since 1997	102
SAP DB Tools.....	103
Comparison of SAP DB and Version 6.1	103
Technical Specification of SAP DB Version 7.3	104
Improvements in SAP DB Version 7.4	106
Terms	107
Automatic Log Backup	108
Backup History	109
Backup ID	109
Backup Medium.....	110
Cache	110
Catalog	110
Checkpoint.....	111
Data Backup	111
Data Devspace	112
Database Administrator	112
Database Catalog.....	112
Database Instance	113
Database Instance Type	113
Database Manager	114
Database Manager CLI	114
Database Manager GUI	115
Database Manager Operator (DBM Operator).....	115
Database Parameters	116
Database Session	117
Database System Administrator (SYSDBA).....	117
Database User.....	117
DBA/DOMAIN.....	117
DBM Server.....	118

DBMCLI	118
DBMGUI	118
Devspace.....	118
External Backup ID.....	119
External Backup Medium	119
External Backup Tool	119
Instance Type	119
Interactive Log Backup.....	119
Kernel	120
Language Support (MapChar Sets)	120
liveCache.....	120
Lock	121
Log Area	121
Log Backup.....	121
Log Devspace	122
Log Mode.....	122
Multiprocessor Configuration.....	123
Name of a Standard Backup Medium	123
Name of External Backup Medium.....	124
Parallel Backup Media.....	124
Redo Area	124
Replication Manager	124
REPM Server.....	125
RESOURCE	126
Run Directory.....	126
SAP DB Document Server	126
SAP DB E-Catalog	126
SAP DB OLAP.....	127
SAP DB OLTP	127
SAP DB Tools.....	127
SAP DB User Classes.....	127
Savepoint.....	127
Session.....	128
Single Backup Medium.....	128
SQL Studio	128
SQL Mode	128
System Devspace	129
Task.....	129
Terminal Support (Termchar Sets).....	129
Thread	129

Transaction.....	130
UNICODE.....	130
Users	131
User Data	131
Web DBM	131
Web SQL.....	132
Web Server.....	133
X Server.....	133
Documentation Overview.....	133



User Manual: SAP DB

This manual provides an overview of the database system SAP DB Version 7.3 and the tools contained therein.

[Architecture: SAP DB \[Page 12\]](#)

[User Concept \[Page 24\]](#)

[Security Concepts \[Page 36\]](#)

[SAP DB Tools \[Page 127\]](#)

[Directory Structure: SAP DB for SAP Systems \[Page 57\]](#)

[Directory Structure: SAP DB for Open Source \[Page 64\]](#)

[Database Parameters \[Page 116\]](#)

[SAP DB as a UNICODE Database \[Page 80\]](#)

[Data Management Using B* Trees \[Page 90\]](#)

[SAP DB for Users of Version 6.1 \[Page 102\]](#)

[Terms \[Page 107\]](#)

For more detailed information on the different SAP DB components, please consult the SAP DB documentation ([Documentation Overview \[Page 133\]](#)).



Architecture: SAP DB

You can find an overview of the main architecture features of the SAP DB relational database system in the **Fact Sheet** on the SAP DB homepage www.sapdb.org. Some aspects of the SAP DB architecture are described in more detail below.

Areas of Application

The SAP DB database system supports different application areas. The SAP DB [database instance \[Page 113\]](#) has different characteristics depending on the application area. The following [database instance types \[Page 113\]](#) exist:

- [SAP DB OLTP \[Page 127\]](#)
- [liveCache \[Page 120\]](#)
- [SAP DB Document Server \[Page 126\]](#)
- [SAP DB OLTP \[Page 127\]](#)
- [SAP DB E-Catalog \[Page 126\]](#)

Availability

SAP DB is available as a stand-alone system, but also supports all mySAP.com components. SAP DB can be used for various [operating system platforms \[Page 23\]](#).

Directory Structure

The information in the SAP DB database instance is stored in files or on raw devices. For information on the distribution of files, see [Directory Structure: SAP DB for Open Source \[Page 64\]](#) and [Directory Structure: SAP DB for SAP System \[Page 57\]](#).

Performance

A well-chosen system architecture ensures high performance of the database system. Therefore, when installing the database system, take account of the information in [Performance Requirements \[Page 65\]](#). If the relevant database parameters are set accordingly, SAP DB also supports [multiprocessor configuration \[Page 123\]](#).

Security Concepts

The architecture of the SAP DB database system ensures that [security concepts \[Page 36\]](#) are complied with.

Data Management Architecture

The SAP DB [data management with B* trees \[Page 90\]](#) ensures efficient data storage on disks and fast data access.

SAP DB Tools

A number of [SAP DB tools \[Page 127\]](#) are available for the SAP DB database system.

See also:

For further information on the SAP DB database system, visit the SAP DB homepage at www.sapdb.org.

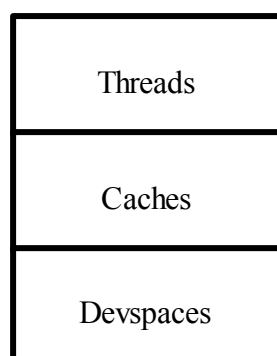


Database Instance

The SAP DB database can be installed and run on a computer in one mode (database instance) or several modes (database instances) ([Database Instance Type \[Page 113\]](#), **see also:** [SAP DB Versions and Database Instance Types \[Page 22\]](#)).

Each database instance consists of [threads \[Page 129\]](#), main memory structures ([caches \[Page 110\]](#)) and disk-based storage of the data on devspaces.

Database
instance



The following disk-based [devspaces \[Page 118\]](#) are available in every database instance for the physical storage of data:

- [System devspace \[Page 129\]](#)
- One or more [data devspaces \[Page 112\]](#)
- One or more [log devspaces \[Page 122\]](#)

Each database instance differentiates between the following areas for the logical storage of data:

- [Database catalog \[Page 112\]](#)

- [User data \[Page 131\]](#)



Thread

A whole series of operating system threads (often referred to as kernels) belong to a [database instance \[Page 113\]](#).

We differentiate between **UKTs** (user kernel threads) and **special threads**.

The required number of UKTs and of special threads depends on the number of [devspaces \[Page 118\]](#), the hardware configuration, and the database parameters.

- [User kernel thread \(UKT\) \[Page 14\]](#)
- [Special thread \[Page 16\]](#)
- [Operating-system dependent special thread \[Page 17\]](#)



User Kernel Thread (UKT)

A [database instance \[Page 113\]](#) contains a series of [threads \[Page 129\]](#). A user kernel thread (UKT) forms a subset of all [tasks \[Page 129\]](#) (for internal tasking).

The following types of tasks exist:

- [Conv Scanner \[Page 14\]](#)
- [Data Writer \[Page 15\]](#)
- [Log Writer \[Page 15\]](#)
- [Server Task \[Page 15\]](#)
- [Timer Task \[Page 15\]](#)
- [Trace Writer Task \[Page 15\]](#)
- [User Task \[Page 16\]](#)
- [Utility Task \[Page 16\]](#)



Conv Scanner

Normally, a sufficient number of addresses of empty pages is available in the [system devspace \[Page 129\]](#) and [converter cache \[Page 18\]](#). These addresses are passed to the dev threads.

At times when the [database instance \[Page 113\]](#) is very busy and an above-average number of new pages is being added to it, the number of addresses available in the relevant pool of the main memory (the Pno pool) may drop below a certain level. When this happens, the conv scanner (a special [user kernel thread \(UKT\) \[Page 14\]](#)) 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**.



Data Writer

Data writers are [user kernel threads \(UKT\) \[Page 14\]](#). Data writer tasks are responsible for writing data from the [data cache \[Page 19\]](#) to the [data devspaces \[Page 112\]](#). They become active when a [savepoint \[Page 127\]](#) or [checkpoint \[Page 111\]](#) is performed.

Savepoint writing takes a long time for a large data cache. The data writers 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 data writers is calculated by the system. It depends primarily on the data cache size and the number of data devspaces.



Log Writer

The log writer is a [user kernel thread \(UKT\) \[Page 14\]](#). The log writer is responsible for writing before and after images from the [log cache \[Page 19\]](#) to the [log devspace \[Page 122\]](#).



Server Tasks

A server task is a [user kernel thread \(UKT\) \[Page 14\]](#). The main purpose of server tasks is to parallelize database functions such as saving to a group of parallel media, restoring from a number of media in parallel, and index compilation.

When the database parameters are being configured, the number of server tasks is determined automatically from the number of [data devspaces \[Page 112\]](#) and the number of data backup devices in use. The maximum number of server tasks available is defined by the [MAXUSERTASKS \[Page 74\]](#) database parameter.



Timer Task

A timer task is a [user kernel thread \(UKT\) \[Page 14\]](#). The timer task handles timeout situations of all types.



Trace Writer Task

The database system allows the activation of a special protocol, the kernel trace, for diagnostic purposes. The trace writer task (a special [user kernel thread \(UKT\) \[Page 14\]](#)) is provided for this.



User Task

When he or it logs on to the [database instance \[Page 113\]](#), each [user \[Page 14\]](#) of the instance or each application is assigned precisely one fixed user task. The user task ensures the processing of SQL statements for the database session.

The number of user tasks available is defined by the [MAXUSERTASKS \[Page 74\]](#) database parameter.



Utility Task

A utility task is a [user kernel thread \(UKT\) \[Page 14\]](#). The utility task is reserved solely for managing the [database instance \[Page 113\]](#).

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 \[Page 108\]](#). This can be performed in parallel with other management actions.



Special Thread

A [database instance \[Page 113\]](#) has the following special [threads \[Page 129\]](#):

- [Coordinator \[Page 16\]](#)
- [Dev Threads \[Page 16\]](#)
- [Requester \[Page 17\]](#)
- [Temporary Dev Threads \[Page 17\]](#)
- [Timer \[Page 17\]](#)



Coordinator

The coordinator is a [special thread \[Page 16\]](#). The coordinator monitors all kernel threads in the [database instance \[Page 113\]](#).

When the database instance is started, the coordinator is the first active thread. It coordinates the starting processes of the other threads.

- If a thread fails while a UNIX operating system is running, the coordinator terminates all other threads.
- If a thread fails while a Windows NT/Windows 2000 operating system is running, an exception handler becomes responsible for terminating all the other threads in an orderly way.



Dev Thread

Dev threads are [special threads \[Page 16\]](#). Dev threads handle the read and write commands that read and write [tasks \[Page 129\]](#) ask to have performed.

The number of dev threads depends primarily on the number of [devspaces \[Page 118\]](#) in the [database instance \[Page 113\]](#). Under normal circumstances, two dev processes/threads are activated for the [system devspace \[Page 129\]](#) and for each [data devspace \[Page 112\]](#) and [log devspace \[Page 122\]](#). Only one dev thread is activated for writing the kernel trace if it is enabled.

The dev thread **dev0** plays a special role; **dev0** coordinates and monitors the dev threads.

- For example, if a mirrored log devspace fails in warm mode (bad devspace), dev0 ensures that the corresponding dev 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 threads are generated.

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



Requester

The requester is a [special thread \[Page 16\]](#). The requester receives both local communication requests (CONNECT) and requests from the network and assigns them to a [user kernel thread \(UKT\) \[Page 14\]](#).



Temporary Dev Thread

Temporary dev threads (`asdev<i>`) (which are [special threads \[Page 16\]](#)) are activated to read and write data for [data backups \[Page 111\]](#).



Timer

The timer is a [special thread \[Page 16\]](#). The timer monitors time for timeout control.



Operating-System-Dependent Special Threads

A [database instance \[Page 113\]](#) has the following operating-system-dependent special threads [\[Page 129\]](#):

- [Clock Thread \[Page 17\]](#)
- [Console Thread \[Page 18\]](#)



Clock Thread

The clock thread is an [operating-system dependent special thread \[Page 17\]](#). The clock thread is only used under Windows NT.

It computes internal times; for example, to determine the time needed to execute an SQL statement.



Console Thread

The console thread is an [operating-system dependent special thread \[Page 17\]](#).

	Windows NT/Windows 2000	UNIX
Requests from the XCONS console	The console thread processes requests from the XCONS console. The XCONS program communicates with the console thread for this purpose.	The XCONS console receives the necessary information directly from the shared memory of the threads.
knldiag file	The console thread collects all database instance [Page 113] messages from the other threads and logs these in a file called knldiag.	The knldiag file is created. Each thread writes information to this file.



Cache

Read and write operations to the [devspaces \[Page 118\]](#) of a [database instance \[Page 113\]](#) are buffered in order to save on disk accesses.

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

The database system recognizes the following caches:

- [Catalog Caches \[Page 18\]](#)
- [Converter Caches \[Page 18\]](#)
- [Data Caches \[Page 19\]](#)
- [File Directory Cache \[Page 19\]](#)
- [Free Block Management \(FBM\) Cache \[Page 19\]](#)
- [Log Cache \[Page 19\]](#)



Catalog Cache

The catalog [cache \[Page 110\]](#) of a [database instance \[Page 113\]](#) stores the catalog objects most recently used by a [database session \[Page 117\]](#) and the internal representation (execution plans) of the most recently executed SQL statements.

Data which is expelled from the catalog cache is moved for the time being to the [data cache \[Page 19\]](#).

A catalog cache exists once per database user session.



Converter Cache

The converter [cache \[Page 110\]](#) of a [database instance \[Page 113\]](#) contains the last read- or write-accessed pages of the [system devspace \[Page 129\]](#).

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%.



Data Cache

The data [cache \[Page 110\]](#) of a [database instance \[Page 113\]](#) contains the last read- or write-accessed pages of the [data devspace \[Page 112\]](#).

The data cache is shared by all simultaneously active users.

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.



File directory cache

The file directory [cache \[Page 110\]](#) is used by the [database instance \[Page 113\]](#) for internal organization. For example, the page addresses of the roots of the individual data [B*-trees \[Page 91\]](#) are administered in the file directory cache.



Free Block Management (FBM) Cache

The FBM [cache \[Page 110\]](#) of a [database instance \[Page 113\]](#) is used in the management of free disk blocks.



Log Cache

The log [cache \[Page 110\]](#) of a [database instance \[Page 113\]](#) consists of the rollback cache and the log queue.

Log entries are first written to the rollback cache. In the case of a COMMIT or if the page of the rollback cache is filled, the page is put in the log queue. The COMMIT is only confirmed once the [log writer \[Page 15\]](#) has written the pages from the log queue to the [log devspace \[Page 122\]](#).



Devspace

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

The drives used should have identical performance parameters (specifically access speeds) to ensure even distribution of data on the devspaces.

The database system has [system devspaces \[Page 129\]](#), [data devspaces \[Page 112\]](#) and [log devspaces \[Page 122\]](#).

Each [database instance \[Page 113\]](#) 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.

If necessary, you can add data or log devspaces to a database instance while the database is running. Paths for data and log devspaces can be changed.



System Devspace

Information on restarts and the mapping of logical page numbers to physical page addresses is managed in the system [devspace \[Page 118\]](#) of a [database instance \[Page 113\]](#).

Therefore, the size of the system devspace is directly related to the database size and is determined by the database system.



Data Devspace

User data (tables, indexes) and the database catalog are stored in the data [devspaces \[Page 118\]](#) of a [database instance \[Page 113\]](#). 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, a database-internal striping algorithm distributes the data belonging to a table evenly across all the data devspaces.



An assignment of tables to data devspaces is neither possible nor necessary.

You can configure one or more data devspaces when you install a new database instance. The disk storage space defined by all the data devspaces is the total size of the database. You can add new data devspaces while the database is in operation as long as the total size of the database does not then exceed the maximum size specified in the database parameter [MAXDATAPAGES \[Page 74\]](#).



Log Devspace

All modifications to the database contents are logged in the [devspace \[Page 118\]](#) of a [database instance \[Page 113\]](#) to ensure that the database can be restored if a media device fails.

The whole [log area \[Page 121\]](#) may comprise several log devspaces. When installing a new database instance you can configure the number of required log devspaces, and you can add new log devspaces during the running of the database.

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 \[Page 72\]](#) to `DUAL`).

During log backups, the content of the log area is copied to a file and the space it originally occupied is released. The backup files are numbered in sequence by the system numbers and are therefore also defined as version files.



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



Database Instance Type

The SAP DB database system supports different application areas. The SAP DB [database instance \[Page 113\]](#) has different characteristics depending on the application area. The following database instance types exist:

- [SAP DB OLTP \[Page 127\]](#)
- [liveCache \[Page 120\]](#)
- [SAP DB Document Server \[Page 126\]](#)
- [SAP DB OLAP \[Page 127\]](#)
- [SAP DB E-Catalog \[Page 126\]](#)

See also:

[SAP DB Versions and Database Instance Types \[Page 22\]](#)



SAP DB OLTP

SAP DB OLTP is a [database instance type \[Page 113\]](#) of the SAP DB database system. SAP DB is a relational database system that was developed for OLTP (Online Transaction Processing). The database system is optimized to process individual transactions fast in environments with a high number of users and large databases.



liveCache

liveCache is a [database instance type \[Page 113\]](#) of the SAP DB database system.

In Supply Chain Management, large volumes of data must be permanently available and changeable. For this reason, an addition has been made to the [SAP DB OLTP \[Page 127\]](#) relational database system to enable actual data structures and data flows (such as networks and relationships) to be mapped more easily and effectively. The product is called liveCache. The liveCache is object-oriented, and in contrast to SAP DB OLTP, stores its data in the main memory of the database system.



SAP DB Document Server

SAP DB Document Server is a [database instance type \[Page 113\]](#) of the SAP DB database system.

In today's information landscape, a large amount of data must be processed that does not have the typical format of a relational database, but is "unstructured" (such as videos, XML documents). SAP DB Document Server was developed on the basis of the [SAP DB OLTP \[Page 127\]](#) relational database system to ensure that as much of this unstructured data as possible can be processed outside the OLTP database. This improves the performance of the SAP DB OLTP database.

One application example is the SAP Content Server.



SAP DB OLAP

SAP DB OLAP is a [database instance type \[Page 113\]](#) of the SAP DB database system.

Online Analytical Processing (OLAP) technologies enable you to perform flexible analyses from a variety of business perspectives. It is based on a multi-dimensional data model that is achieved using relational database tables.

One application example is the Business Warehouse System. In contrast to [SAP DB OLTP \[Page 127\]](#) systems, a Business Warehouse System is configured so that large quantities of historical and operative data can be formatted with acceptable response times.



SAP DB E-Catalog

SAP DB E-Catalog is a [database instance type \[Page 113\]](#) of the SAP DB database system.

In Internet catalog applications, a small number of hits must be determined from a large number of product descriptions. To support this, the TREX search engine has been integrated in the [SAP DB OLTP \[Page 127\]](#) relational database system. With the TREX search engine, product descriptions (long texts) can be indexed, and the search terms looked for (exact search, phrase search, fuzzy search, linguistic search).

One application example is the BugsEye or eMerge product from Requisite.



SAP DB Versions and Database Instance Types

The following table contains a list of the SAP DB versions of the individual [database instance types \[Page 113\]](#).

SAP DB- Version	Database instance type				
	SAP DB OLTP	liveCache	SAP DB Document Server	SAP DB OLTP	SAP DB E-Catalog
6.2.10	✗				
7.1.03					
7.1.04		✗			
7.1.06		✗			

7.2.01					✕
7.2.02			✕		
7.2.03	✕		✕	✕	
7.2.04	✕	✕	✕		
7.2.05	✕	✕	✕	✕	
7.3.00	✕		✕	✕	✕
7.3.32					✕
7.4.01		✕			
7.4.02	✕*	✕			

* under construction



Operating System Platform

SAP DB supports the following operating system platforms:

- Compaq True64 Unix / Alpha
- IBM AIX / PowerPC
- SUN Solaris / SPARC
- HP-UX / HP-PA
- Linux / Intel
- Siemens Reliant Unix / MIPS
- Windows NT / Intel
- Windows NT / Intel



Multiprocessor Configuration

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

The aim here is to allow the maximum possible number of [database sessions \[Page 117\]](#) to be supported by the minimum possible number of operating system threads.

The configuration of the database system controls the degree of external/internal tasking via the two parameters [MAXUSERTASKS \[Page 74\]](#) and [MAXCPU \[Page 73\]](#).



On a computer, 4 processors are available for the [database instance \[Page 113\]](#). No more than 800 database sessions should be running simultaneously.

In this case, set MAXCPU to 4 and MAXUSERTASKS to 800.

The database instance can then utilize the four processors by establishing four operating system threads, each of which performs an internal tasking for up to 200 users.



User Concept

The SAP DB database system uses [SAP DB user classes \[Page 127\]](#) and supports different roles ([role concept \[Page 29\]](#)).

Setting Up a Database Session

A [database session \[Page 117\]](#) is started when the user logs on to the [database instance \[Page 113\]](#). In order to log on, certain user data needs to be transferred to the SAP DB component for identification purposes.

The following user data is needed to log on:

userid	User name
password	The user's password
database_name	The name of the SAP DB instance you want to work on
server_node	The name of the server node on which the database you called is running

In addition to this generally required user data, you can enter further data, which is then transferred when you log on.

You can also transfer the following user data to the SAP DB tools or the C/C++ Precompiler programs:

- Enter [user data as options \[Page 29\]](#) (note that the [required options \[Page 32\]](#) must always be entered for each tool used)
- Enter user data on the logon screen (for DBMGUI and SQL Studio)
- [User Data Using XUSER \[Page 33\]](#)

Entering incomplete or incorrect user data when calling the C/C++ Precompiler programs, application programs, or the SAP DB tools cause the program to terminate with an error message.



When you call the C/C++ Precompiler and application programs that were generated using the SAP DB programming interface, you can also transfer user data directly in the program itself. For this reason, the program contains a special modified rule for the transfer of user data. Basically, the logon options for the C/C++ Precompiler and application programs are the same as for the SAP DB tools.



SAP DB User Classes

The SAP DB database system differentiates between two main user classes:

- [Database Manager Operator \(DBM Operator\) \[Page 115\]](#)
- [Database Users \[Page 117\]](#)



Database Manager Operator (DBM Operator)

Users working with the database management tool [Database Manager \[Page 114\]](#) are known as Database Manager Operators. The [SAP DB user class \[Page 127\]](#) is the DBM operator.

Depending on what [user authorizations \[Page 25\]](#) the DBM operator has been given, a DBM operator is able to perform all kinds of Database Manager functions.

You create the **first** DBM operator when you register a [database instance \[Page 113\]](#). When you register a new database instance, the system asks you to create a DBM operator by entering a user name and password for the DBM operator. The DBM operator's password can be changed at a later date.

- This DBM operator is then responsible for managing and monitoring the database system and for running backups. The DBM operator is also authorized to perform all Database Manager functions, regardless of what operating mode the database instance is in.
- The DBM operator is also authorized to create additional DBM operators, and assign these all or some of their authorizations.
- The DBM operator can log on to the Database Manager more than once, which means the DBM operator can, for example, query operating parameters while functions that take a long time are still running.



To access a new database instance after registering it, this database instance must be registered in the Database Manager under the name and password of the DBM operator.

DBM operators are **not** [database users \[Page 117\]](#). You need to create database users in order to work on a database instance.



User Authorizations

The database system SAP DB makes a distinction between two types of authorizations for the [Database Manager operator \[Page 115\]](#) (DBM operator):

- [DBM Server Authorizations \[Page 26\]](#)
Depending on the DBM Server authorizations of the DBM operator, you can use the [Database Manager \[Page 114\]](#) to start and stop [database instances \[Page 113\]](#), backup the contents of database instances, to change [database parameters \[Page 116\]](#), and so on.
- [Operating System User Authorizations \[Page 27\]](#)
To execute operating system commands on remote computers using the Database Manager, the DBM operator must have operating system user authorizations on those computers.



DBM Server Authorizations

DBM Server authorizations are the [user authorizations \[Page 25\]](#) of the [Database Manager operator \[Page 115\]](#) (DBM operator). Depending on which DBM Server authorizations they have, DBM operators can execute certain commands on the [DBM Server \[Page 118\]](#).

An authorization may cover more than one command and one command may have more than one authorization assigned to it.

Use the [Database Manager \[Page 114\]](#) functionality to assign the relevant DBM server authorizations to a DBM operator.

See also:

Database Manager CLI: SAP DB 7.3 → Database Manager Operators (DBM Operators) → Operator Properties → DBM Operator Authorizations → [Authorizations for Using the DBM Server \[Extern\]](#). All the references in the following table relate to the relevant sections of this documentation.

DBM Server Authorization	Description
DBInfoRead [Extern]	Requesting status information
ExecLoad [Extern]	Running the LOAD program
SystemCmd [Extern]	Executing operating system commands
UserMgm [Extern]	Managing the DBM operator
DBFileRead [Extern]	Database file access (read only)
Backup [Extern]	Carrying out backups
InstallMgm [Extern]	Installation management
LoadSysTab [Extern]	Loading system tables
ParamCheckWrite [Extern]	Parameter access (checked write)
ParamFull [Extern]	Parameter access (read and write)
ParamRead [Extern]	Parameter access (read only)
DBStart [Extern]	Starting the database instance
DBStop [Extern]	Stopping the database instance
Recovery [Extern]	Restoring backups
AccessSQL [Extern]	Accessing SQL session
AccessUtility [Extern]	Accessing a utility session

[Default Authorizations for the First DBM Operator \[Page 26\]](#)



Default Authorizations for the First DBM Operator

The first [DBM operator \[Page 115\]](#) you create when you register a database instance contains all [DBM Server authorizations \[Page 26\]](#).



Operating System User Authorizations

The operating system user authorization is one of the [user authorizations \[Page 25\]](#) of the [Database Manager operator \[Page 115\]](#) (DBM operator). To execute operating system commands on remote computers using the [Database Manager \[Page 114\]](#), the DBM operator must have operating system user authorizations on those computers.

Use the Database Manager functionality to assign a DBM operator the user identification for the operating system.



Database User

Database users log on to a [database instance \[Page 113\]](#) and work with database objects such as tables, views, and indexes (the [SAP DB user class \[Page 127\]](#) is database user).

SAP DB differentiates between various [database user classes \[Page 27\]](#).

Database users can be grouped into [user groups \[Page 28\]](#).



Database User Classes

The database user class is used to define which database operations a [database user \[Page 117\]](#) may perform.

SAP DB differentiates between the following database user classes:

- [Database System Administrator \(SYSDBA\) \[Page 117\]](#)
- [DBA/DOMAIN \[Page 117\]](#)
- [RESOURCE \[Page 126\]](#)
- [STANDARD \[Page 28\]](#)



Database System Administrator (SYSDBA)

The database system administrator ([database user class \[Page 27\]](#) SYSDBA) is the first [database user \[Page 117\]](#) that is created when a new [database instance \[Page 113\]](#) is registered. Enter a user name and password for this user.

Each database instance has one single SYSDBA user.

The SYSDBA user's password can be changed once the database registration has finished.

The SYSDBA user is very important, especially when database instances are installed. The SYSDBA user is responsible for setting up the system and for creating other database users. The SYSDBA user is the owner of system tables. When system tables are uploaded, the upload tool logs on to the database instance as SYSDBA.

The SYSDBA is able to define data and database procedures. The SYSDBA can also grant other users privileges for these database objects.

Furthermore, the SYSDBA has all [DBM operator \[Page 115\]](#) authorizations and is able to carry out all Database Manager functions.



DBA/DOMAIN

DBA Users (Database Administrators)

DBA users ([database user class \[Page 27\]](#): DBA) are [database users \[Page 117\]](#). They are created by the [SYSDBA \[Page 117\]](#).

A DBA user is authorized to create [RESOURCE \[Page 126\]](#) and [STANDARD users \[Page 28\]](#). The DBA user can also define data and database procedures and grant other users all or some DBA user privileges for these database objects.

A DBA user can group users with identical access rights into [user groups \[Page 28\]](#).

DOMAIN User

The DOMAIN user is a special database user belonging to the DBA database user class. The DOMAIN user owns [database catalog \[Page 112\]](#) system tables.

Just like the SYSDBA user, the DOMAIN user is created when a [database instance \[Page 113\]](#) is registered. The DOMAIN user is created by the system under the pre-defined name DOMAIN. The password is the same as that of the SYSDBA user that was created earlier.

The DOMAIN user's password can be changed at a later date.



RESOURCE

[Database users \[Page 117\]](#) belonging to the [database user class \[Page 27\]](#) RESOURCE can be created by the [SYSDBA \[Page 117\]](#) and [DBA users \[Page 117\]](#).

RESOURCE users can define data and database procedures and grant other users privileges for these database objects.



STANDARD

[Database users \[Page 117\]](#) belonging to the [database user class \[Page 27\]](#) STANDARD only have access to data and database procedures that were defined by other users and for which they have privileges.

STANDARD users themselves can define view, synonyms, and temporary tables.



User Groups

[Database users \[Page 117\]](#) can be grouped into user groups.

A user group can either be assigned to the [database user class \[Page 27\]](#) [RESOURCE \[Page 126\]](#) or to the database user class [STANDARD \[Page 28\]](#). Database users can be defined as members of a user group.

All database objects defined by members of a certain user group can be identified by the user group name. The owner of objects such as these is the user group and not the individual user

that defined the objects. If a member of a user group creates objects, each member of that group can work with these objects as if they were the object owners.

Privileges can only be granted or removed from the user group as a whole and not from individual members of the group.



The Role Concept

The SAP DB database system supports different roles. A role is a group of privileges that you can assign to [database users \[Page 117\]](#), [user groups \[Page 28\]](#), or other roles by specifying a role name in the GRANT statement.

When the CREATE ROLE statement has been executed, the role is initially empty. The privileges must be assigned to the role using the GRANT statement.

That a role is available and the properties of that role are all registered in the catalog in the form of metadata. A user that creates a role becomes its owner.

Only database users belonging to database user class [DBA \[Page 117\]](#) are able to create roles.

The new role name cannot be the same as the name of any other role, a user, or a user group.

You can assign roles to a user or user group by using the ALTER USER or ALTER USERGROUP statement. The roles are active as soon as a [database session \[Page 117\]](#) is opened. Alternatively, you can activate roles during a session using the SET statement. If you activate a role during a session, the current session user has all the privileges assigned to that role.

If a password has been assigned to a role, users assigned to that role can only activate it by entering the password in the SET statement.



Roles are not active when executing data definition commands.

See also:

Reference Manual: SAP DB 7.2 and 7.3, section Basic elements → Names → [Role name \[Extern\]](#)



User Data as Options

User data and, in some cases, other data can be entered as options when calling the SAP DB tools, the C/C++ Precompiler, or the application programs.

Syntax

```
<dbmcli | repmgr | dbmgui | sqlsto | cpc <file_name>> [<options>] ...
```

<options>

SAP DB Component	
DBMCLI	DBMCLI options [Page 50]
REPMCLI	REPMCLI options [Page 53]
DBMGUI	DBMGUI options [Page 49]

SQL Studio	SQL Studio options [Page 55]
C/C++ Precompiler/application programs	C/C++ Precompiler options [Page 30]

If you enter all the required user data (in most cases, the options `-u`, `-d`, `-n`) and these entries are complete and correct, you will be logged on to the desired SAP DB tool or the C/C++ Precompiler.

For each tool, the user data required for logon to a database instance must be entered. For DBMGUI and SQL Studio, options are not necessarily required, but this data can also be entered in a logon screen. For other tools, there are [required options \[Page 32\]](#), which must always be entered.

The SAP DB tool DBMCLI and the C/C++ Precompiler support the XUSER concept ([User Data Using XUSER \[Page 33\]](#)).



Options (C/C++ Precompiler)

C/C++ Precompiler

You can enter the following options when you call up the C/C++ Precompiler:

	Option	Default
ansi c	-E ansi	
c++	-E cplus	
check nocheck	-H nocheck	-H check
check syntax	-H syntax	
comment	-o	
compatible	-C	
datetime eur	-D eur	-D internal
datetime iso	-D iso	
datetime jis	-D jis	
datetime usa	-D usa	
dblocale	-x <dblocale>	
extern	-e	
help	-h	
isolation-level	-I <isolation_level>	-I 10
list	-l	
margins	-m <m_begin>,<m_end>	-m 1,132
maxpacketsize	-b <max_packet_size>	-b 16000
nowarn	-w	
precom	-c	
profile	-R	
program	-P <program_name>	-P <file_name>
serverdb	-d <database_name>	
servernode	-n <server_node>	
silent	-s	
sqlmode internal	-S internal	-S internal
sqlmode ansi	-S ansi	

sqlmode ansiора	-S ansiора	
sqlmode db2	-S db2	
sqlmode oracle	-S oracle	
timeout	-t <timeout>	
trace file	-F <trace_file>	
trace long	-X	
trace short	-T	
unicode	-G unicode	
user	-u <userid>,<password>	
userkey	-U <user_key>	
version	-V	
32 Bit Support	-BIT32	64 Bit if available
SDK Version	-MmCC	7401
	M=Major m=Minor	
	C=Correctionlevel	



The following [user specifications as options \[Page 29\]](#) should be transferred: [database user \[Page 117\]](#) and the name of the [database instance \[Page 113\]](#)

```
cpc -u smith,geheim -d MK1 testfile
```

The C/C++ Precompiler is called and the user **smith** connects to the database instance **MK1** to compile the file **testfile.cpc**.

See also:

- Description of all options: *User Manual Precompiler*, section *The SAP DB Precompiler* → *SAP DB Precompiler Functions* → *Precompiler Options*
- *Installation Guide C/C++-Precompiler: SAP DB 7.3*, section *Installation Guide for Developers* → [Calling the C/C++-Precompiler \[Extern\]](#)

Application Programs

When an application program is called, you can enter the following options for the runtime environment:

	runtime_options
dblocale	-x <dblocale>
isolation-level	-l <isolation_level>
no select direct fast	-f
profile	-R
serverdb	-d <database_name>
servernode	-n <server_node>
timeout	-t <timeout>
trace alt	-Y <statement_count>
trace file	-F <trace_file>
trace long	-X
trace no date/time	-N
trace short	-T
trace time	-L <seconds>

user -u <userid>,<password>
 userkey -U <user_key>

See also:

- Description of all options: *User Manual Precompiler*, section *The SAP DB Precompiler* → *Functions of the SAP DB Runtime System* → *Runtime Options*
- *Installation Guide C/C++-Precompiler: SAP DB 7.3*, section *Installation Guide for Developers* → [Example for Executing a Precompiler Application \[Extern\]](#)



Required Options

If you do not enter all the required [user data as options \[Page 29\]](#), or if you were unable to log on to the [database instance \[Page 113\]](#) with that user data, the SAP DB tools will react as follows:

- [DBMGUI \[Page 115\]](#), [SQL Studio \[Page 54\]](#)
 Entry of options is not required. These tools are started even if options are not entered, or if options are incorrect or incomplete. DBMGUI and SQL Studio always display a logon screen. In the logon screen, enter the user data required to log on to the database instance.
- [DBMCLI \[Page 114\]](#)
 Only some of the options required to log on to the database instance were entered, or they were incorrect. DBMCLI starts and you are connected to the DBM Server. However, you are not logged on to the database instance. In this instance, the DBMCLI can only be used to call DBM Server commands that are not related to the database (for example, `help`, `dbm_version`).
 You need to connect to the database instance to use all the DBMCLI functions. Enter the option `-d <database_name>`, if the database instance is on the local computer, and enter the option `-n <server_node>` as well if the database instance is on a remote computer.
 Further user data (such as user name, ID) can be transferred as options (for example `-u`) or, if required, as DBM server commands (for example `user_logon`).



You should always start DBMCLI entering the options `-d`, `-u` and, if required, `-n` when calling the tool ([Options \(DBMCLI\) \[Page 50\]](#)).

- [REPMCLI \[Page 124\]](#)
 The minimum required option `-d <database_name>` for logging on to the database instance and the required option `-b <command_file>` for entering the command file were not entered at all, were incomplete, or incorrect. REPMCLI is not started, but displays a list of all possible options.
 Log on to the REPMCLI again by entering the required options.
 Further user data (such as user name, ID) can be transferred as options (for example `-u`) or, if required, as REPM server commands (for example `use_user`).



When you call the tool, you should always start the REPMCLI by entering the required options `-d` and `-b` ([Options \(REPMCLI\) \[Page 53\]](#)).



User Data and XUSER

Use

Using the tool XUSER, you can predefine and save user data ([using XUSER \[Page 33\]](#)). You can access your [XUSER data \[Page 34\]](#) when you call the [DBMCLI \[Page 114\]](#) or a precompiler program or when you start an SAP system. In this case, the user would use the data stored under a certain user key (USER_KEY).

- Windows NT: You can manage individual user data in this way if, for example, more than one database user is using the same Windows PC and each user is logging on under a different name.
- UNIX: You can manage individual user data in this way if, for example, more than one database user is using the same HOME directory and each user is logging on under a different name.

Procedure

1. The database administrator ([DBA operator \[Page 117\]](#)) sets up a [database user \[Page 117\]](#) using the relevant CREATE USER statement.
2. The database administrator informs the operating system user what this database user's data is.
3. The operating system user saves the database user's data using XUSER.

When the DBMCLI or the C/C++ Precompiler is now called using the option `-u <user_key>`, the system uses the data that was defined in `user_key` using XUSER. Enter the user key exactly as it was defined in XUSER, that is, your entries are case-sensitive.



Applications such as [REPMCLI \[Page 124\]](#) and [DBMGUI \[Page 115\]](#) and applications that use the ODBC interface (for example, [SQL Studio \[Page 54\]](#)) have no way of accessing XUSER data.



Using XUSER

User data for setting up a [database session \[Page 117\]](#) can be predefined and saved by entering a user key (USER_KEY) using the tool XUSER. When the [DBMCLI \[Page 114\]](#) or the C/C++ Precompiler is called or an SAP system is started, the system can access the required [XUSER data \[Page 34\]](#) by entering the relevant user key).

XUSER can manage up to 32 combinations of XUSER data per operating system user.

- Windows NT: XUSER data is stored in the database registry. You are not permitted to make "manual" changes in the registry because these may result in inconsistencies in the operating system.
- UNIX: XUSER data is stored in the file `.XUSER.62` (also referred to as the XUSER file) in the user's HOME directory.

Use

- XUSER data is generated using option `-b` ([Generating XUSER Data in the Background \[Page 35\]](#)).
- XUSER data is called using option `-u` ([User Data Using XUSER \[Page 33\]](#)).

- All XUSER data is deleted by entering the following command:
`xuser -b <empty_file>`
 The `empty_file` can only contain blanks or must be of length 0.



XUSER Data

Up to 32 combinations of XUSER data can be stored.

Parameter	Explanation
USERKEY	Name of the user key used to address this combination of XUSER data The first combination of parameters is called DEFAULT. This name cannot be changed. If you enter additional key names, these are case-sensitive.
USERID	User name If the user name contains small letters or special characters, it must appear within double quotation marks. Otherwise small letters will be converted to capitals.
PASSWORD	The user's password The password is not visible and must be entered twice for security reasons (Confirm Password). If the password contains small letters or special characters, it must appear within double quotation marks. Otherwise small letters will be converted to capitals.
SERVERDB	The name of the database instance you want to work on If you do not specify a database, the system uses the name in the environment variable SERVERDB. This entry is case-sensitive.
SERVERNODE	The name of the server node on which the database you called is running If you do not specify a server node, the local computer is selected. This entry is case-sensitive.
SQLMODE	This ensures that the SQL dialects of other manufacturers are compatible. The default value is INTERNAL. Other possible values are ANSI, DB2, and ORACLE. This parameter affects precompiler programs.
TIMEOUT	The length of time (in seconds) that can pass before the system terminates an inactive user session. If you want to use the default setting, enter -1. The default value is already set in the empty entry screen.
ISOLATION	The parameter ISOLATION LEVEL is for locks that affect the user (for application programs and precompilers only). If you want to use the default setting, enter -1. The default value is already set in the empty entry screen.

See also:

[Using XUSER \[Page 33\]](#)



Generating XUSER Data in the Background

[XUSER data \[Page 34\]](#) is generated in the background with the help of a file that you specify when you call XUSER ([Using XUSER \[Page 33\]](#)).

Syntax

```
xuser -b <file_name>
```

You are free to define your own file name. The file is made up of groups of eight lines:

If you use the option `-b <file_name>`, this will always generate new XUSER data. All existing XUSER data is overwritten.

Setting Up Each Group (Combining Parameters)

The file entries begin in the first column. There are no field descriptions and they contain the following values in the specified order:

```
USERKEY
USERID
PASSWORD
SERVERDB
SERVERNODE
SQLMODE
TIMEOUT
ISOLATION
```

These parameters are explained in more detail in the section [XUSER Data \[Page 34\]](#).



```
DEFAULT
meier
confidential
dbldial
sqldial
INTERNAL
-1
-1
```

If you do not want to enter any optional parameters, make sure the line is left blank.



```
home
meier
"Strictly_Confidential"
db2dial
sqldial

90
1
```

In the above example, the parameter SQLMODE was left blank.



Security Concepts

- You can achieve a high level of [availability \[Page 36\]](#) of the SAP DB database system by using fault-tolerant hardware and software.
- The [restartability \[Page 37\]](#) of the SAP DB database system allows the automatic correction of some errors when you start the database system.
- For backing up the datasets in line with the [backup strategy \[Page 37\]](#), SAP DB provides tools for carrying out [data backups \[Page 111\]](#) and [log backups \[Page 121\]](#).



Availability

To guarantee a high level of availability of the database system, see the [Security Requirements \[Page 65\]](#).

See also:

[Directory Structure: SAP DB for Open Source \[Page 64\]](#)

[Directory Structure: SAP DB for SAP Systems \[Page 57\]](#)



Security Requirements

Using the appropriate hardware, operating system, or database features can improve the downtime security of a [database instance \[Page 113\]](#).

If you want to ensure a high level of security, we recommend the use of RAID-5 or RAID-1 configurations for the [system devspace \[Page 129\]](#) and the [data devspace \[Page 112\]](#)s. A failure and the exchange of a disk does not impair database operation if the RAID system can carry out a restore process.

Every devspace category should be stored on a different disk.

- [Log Devspace \[Page 122\]](#)

In a productive system, the log devspaces should always be mirrored for security reasons. The DUAL [log mode \[Page 122\]](#) of the SAP DB database system can be used to this end. If the log entries are mirrored using other available hardware or software tools, then the SINGLE log mode can be used.

You should not use RAID-5 configurations for the log devspaces. RAID-5 systems do not allow full mirroring. Because the database instance writes log entries sequentially, this can lead to a loss in performance.

In a productive system, never use the DEMO log mode, even when you are using RAID systems.

- System Devspace, Data Devspaces

If you want to mirror the system devspace or the data devspaces, we recommend that you use RAID-1 configurations.

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.

UNIX: In a productive system, data devspaces and log devspaces should be used in conjunction with raw devices. In the event of a system crash, raw devices are extremely secure.



Restartability

If a database failure occurs (due to a power failure for example) and if the [devspaces \[Page 118\]](#) are fully functioning, the database system uses the log entries to restore the [database instance \[Page 113\]](#) to its last consistent state when the database instance is restarted.

This means that the effects which completed transactions had on the [data devspaces \[Page 112\]](#) are reproduced (rolled forward), and the effects that uncompleted transactions would have had are cancelled out (rolled back).

If there is an error in the data devspace (physical disk error), then all you have to do is to import the last complete [data backup \[Page 111\]](#) once the problem has been solved. If the [log area \[Page 121\]](#) still contains all the required data, the database system uses the log entries to restore the database instance to its last consistent state when the database instance is restarted.



Backup Strategy

Use

To ensure secure database system operation, you must back up your datasets regularly. You should carry out regular [data backups \[Page 111\]](#) and [log backups \[Page 121\]](#).

Data Backup

Note that, if data is restored, the older the data backup is, the more log entries have to be redone. Therefore, carry out data backups as often as possible.

- Carry out a complete [data backup without checkpoint \[Page 39\]](#) on every productive day.
- Carry out at least one complete [data backup with checkpoint \[Page 39\]](#) per week (if possible, over the weekend).
- If you cannot or do not want to perform a complete data backup every day, you should at least start an incremental data backup without checkpoint on every productive day.



If a complete data backup is active, an incremental data backup cannot be started.

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.

Automatic Log Backup

[Automatic log backup \[Page 108\]](#) (autosave log mechanism) should always be active. Complete and incremental data backups are also possible when automatic log backup is active.



Check at regular intervals that automatic log backup is active. Note that the autosave log mechanism is deactivated if, for example, the database system is shut down. When the database system is started, automatic log backup must be reactivated.

Interactive Log Backup

Back up the log entries on every productive day. If automatic log backup is deactivated, you must check at regular intervals that there is sufficient storage space in the [log area \[Page 121\]](#). If required, back up the log area immediately by starting an [interactive log backup \[Page 119\]](#). If insufficient storage space in the log area means that new log entries cannot be written, a database standstill results.

Version File Backup

For log backups, back up the version files to a medium of your choice at regular intervals. You can then delete the backed up version files and thereby ensure that there is always sufficient space in the directory for the version files.



Data Backup

In a complete data backup, all [data devspaces \[Page 112\]](#) are backed up to the [backup medium \[Page 110\]](#) you specified. In an incremental data backup it is only the pages which have been updated that are backed up.

You can carry out the following types of backups:

- Complete [data backup with checkpoint \[Page 39\]](#)
- Complete [data backup without checkpoint \[Page 39\]](#)
- Incremental data backup with checkpoint
- Incremental data backup without checkpoint

Data Backups in WARM Mode

In **WARM** mode, you can either perform complete or incremental data backups with or without checkpoint.

When you back up in warm mode, remember that the state of the [database instance \[Page 113\]](#) that is saved is always the state when the backup operation was started.

When you carry out a data backup with checkpoint, you get a [consistent data backup \[Page 39\]](#).

Data Backups in COLD Mode

Complete and incremental data backups (without Checkpoint) can also be carried out when the database is not running, that is in **COLD** mode.

Whether you get a [consistent data backup \[Page 39\]](#) in this case depends on how the database instance was shut down. If a [checkpoint \[Page 111\]](#) was requested on shutdown, and this could be performed correctly, you can generate a consistent data backup.

See also:

[Backup Strategy \[Page 37\]](#)

[Saving Data Backups \[Page 40\]](#)



If data is recovered, you may need the relevant [log backups \[Page 121\]](#) as well as the data backup.



Data Backup with Checkpoint

You can carry out complete and incremental [data backups \[Page 111\]](#) with [checkpoint \[Page 111\]](#).

Complete and incremental data backups **with** checkpoint are consistent in themselves ([consistent data backup \[Page 39\]](#)).

When making data backups of this kind, delays may be caused by the need to wait for the checkpoint.

Procedure

[Backup Strategy \[Page 37\]](#)

[Saving Data Backups \[Page 40\]](#)



Data Backups without Checkpoint

You can carry out complete and incremental [data backups \[Page 111\]](#) without [checkpoint \[Page 111\]](#).

A complete or incremental data backup without checkpoint is only consistent once the [log backups \[Page 121\]](#) carried out after the start of the data backup in question have been reloaded. You do not require consistent backups, however, for day-to-day database operation.

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

Procedure

[Backup Strategy \[Page 37\]](#)

[Saving Data Backups \[Page 40\]](#)



Consistent Data Backup

A [data backup \[Page 111\]](#) that is consistent in itself can be used to recover the [database instance \[Page 113\]](#). No further [log backups \[Page 121\]](#) need to be restored to recover the state of the database instance at the time of consistent data backup.

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



To determine whether a data backup is consistent, check the entry in the [backup history \[Page 109\]](#) in the *Log Required* column for the relevant data backup.

No: The data backup is consistent.

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

Procedure

[Backup Strategy \[Page 37\]](#)

[Saving Data Backups \[Page 40\]](#)



Saving Data Backups

You can find information on saving [data backups \[Page 111\]](#) in the following documentation:

- *Database Manager GUI: SAP DB 7.3*, Section [Backup \[Extern\]](#)
- *Database Manager CLI: SAP DB 7.3*, section *Calling the Database Manager CLI → DBM Server Commands → [Backing Up and Recovering Database Instances \[Extern\]](#)*
- *Computing Center Management System*, sections in *Database Management in CCMS → SAP DB - DBA in CCMS*



Log Backup

In a log backup, the contents of the [log area \[Page 121\]](#) are copied to version files. A version file is generated for each log segment.

The space originally occupied by the log segments freed up again after the log backup.

The names of the version files that are generated consist of the version file name entered during definition of the [backup media \[Page 110\]](#) and a sequence number assigned by the database system when the log segments are saved.

Need for Log Backup

- [Consistent data backup \[Page 39\]](#) (for example, a complete [data backup with checkpoint \[Page 39\]](#) in WARM mode is consistent in itself.)
If data is **recovered**, a consistent data backup is not sufficient to recover the current state of the [database instance \[Page 113\]](#) up to a certain point in time (for example, the time immediately before the disk failure occurred).
In order to recover the current state, the data backup and log entries that were written after the data backup must be restored to the database system.
- Inconsistent data backup (for example, a complete [data backup without checkpoint \[Page 39\]](#) in WARM mode is not consistent in itself.)
If data is **recovered**, you need the relevant log backups for the inconsistent data backup in order to restore a consistent database state.
However, the inconsistent data backup and related log backups are not sufficient to recover the current state of the database instance up to a certain point in time (for example, the time immediately before the disk failure occurred).
In order to recover the current state, the data backup, the related log backups, and the log entries that were written after the data backup must be restored to the database system.
- If insufficient storage space in the log area means that new log entries cannot be written, a database standstill results. For this reason, it is necessary to back up log entries regularly.

See also:

[Backup Strategy \[Page 37\]](#)

[Saving Log Backups \[Page 41\]](#)



Automatic Log Backup

Automatic [log backup \[Page 121\]](#) is recommended to ensure the safety of data in producing systems.

If automatic log backup is activated, a log segment is saved as soon as it has been filled. This log segment is then released again. This has the advantage that an overflow of the [log area \[Page 121\]](#) is almost impossible.

This mechanism is particularly recommended for all [database instances \[Page 113\]](#) in which extensive write and change-intensive transactions are carried out. In this way, constant monitoring of the usage level of the log area is not necessary.



While the automatic log backup is enabled, no other log backup can be performed, although a [data backup \[Page 111\]](#) can be performed.

You can perform log backups in `WARM` mode.

See also:

[Backup Strategy \[Page 37\]](#)

[Saving Log Backups \[Page 41\]](#)



Interactive Log Backup

By using the interactive [log backup \[Page 121\]](#) you can back up all the pages of the [log area \[Page 121\]](#) that have been saved in the log since the last log backup. A version file is also created for the last log segment, which may not have been filled.

Prerequisites

A full [data backup \[Page 111\]](#) of the current database instance has been created.



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

While an interactive log backup is running, no further backups can be performed.

See also:

[Backup Strategy \[Page 37\]](#)

[Saving Log Backups \[Page 41\]](#)



Saving Log Backups

Types of Log Backup

The following types of [log backups \[Page 121\]](#) are supported for the SAP DB database system:

- [Automatic Log Backup \[Page 108\]](#)

- [Interactive Log Backup \[Page 119\]](#)



You cannot perform log backups in the [log mode \[Page 122\]](#) DEMO (database parameter [LOG_MODE \[Page 72\]](#)).

Procedure

You can find information on saving log backups in the following documentation:

- *Database Manager GUI: SAP DB 7.3*, Section [Backup \[Extern\]](#)
- *Database Manager CLI: SAP DB 7.3*, section *Calling the Database Manager CLI → DBM Server Commands* → [Backing Up and Recovering Database Instances \[Extern\]](#)
- *Computing Center Management System*, sections in *Database Management in CCMS* → *SAP DB - DBA in CCMS*



SAP DB Tools

You can use the following SAP DB tools:

- [Database Manager \[Page 114\]](#)
- [Replication Manager \[Page 124\]](#)
- [SQL Studio \[Page 54\]](#)



Architecture: SAP DB Tools

The SAP DB tools [Database Manager \[Page 114\]](#) and [Replication Manager \[Page 124\]](#) each consist of a server part and a client part. The server part is responsible for the functions, and the user uses the client to access the tool.

The SAP DB tool [SQL Studio \[Page 54\]](#) has a graphical user interface and accesses the [database instance \[Page 113\]](#) through ODBC.

Web clients are also available for the Database Manager and SQL Studio.

Client/Server for the SAP DB Tools

	Client	Server
Database Manager	DBMGUI, DBMCLI Web DBM	DBM Server [Page 118]
Replication Manager	REPMCLI	REPM-Server [Page 125]
SQL Studio	SQL Studio Web SQL	Database Instance

The Database Manager and Replication Manager have a script interface (for example with Perl or Python). Further clients can therefore be defined via this script interface.

Prerequisite

Client	Windows NT/Windows 2000	UNIX
DBMCLI REPMCLI	The X Server [Page 133] must be active as a service on the database server.	The X server must be running as a background process on the database server.
DBMGUI	The X-server must be active as a service on the database server.	The X server must be running as a background process on the database server. The Database Manager GUI is installed on a Windows server. The database instance on the UNIX server is administered remotely.
Web DBM Web SQL	The Web services must be installed together with the Web Server [Page 133] on one server.	The Web services must be installed together with the Web Server on one server.

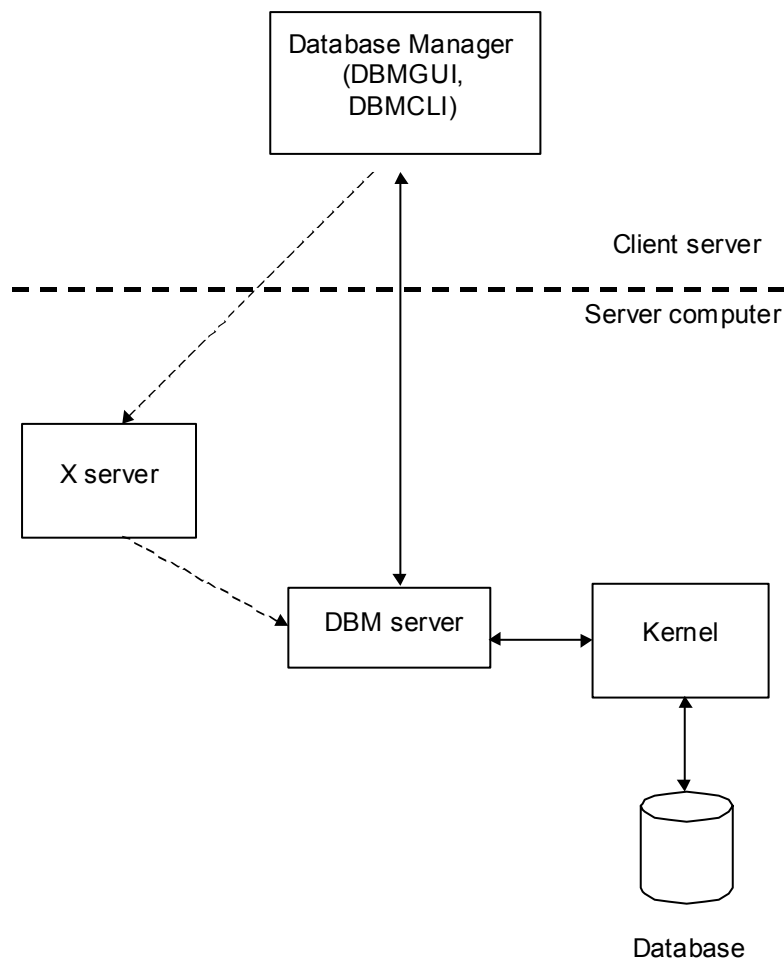
Architecture

- [Architecture of the Database Manager \[Page 43\]](#)
- [Architecture of the Replication Manager \[Page 44\]](#)
- [SQL Studio Architecture \[Page 45\]](#)
- [Architecture of the SAP DB Web Tools \[Page 46\]](#)



Architecture of the Database Manager

One example of the [architecture of the SAP DB tools \[Page 42\]](#) is the architecture of the [Database Manager \[Page 114\]](#). At the request of the client (DBMGUI or DBMCLI) of the Database Manager, the [X server \[Page 133\]](#) starts the [DBM-Server \[Page 118\]](#). After the DBM Server has been started, communication takes place directly between the client and the DBM Server.

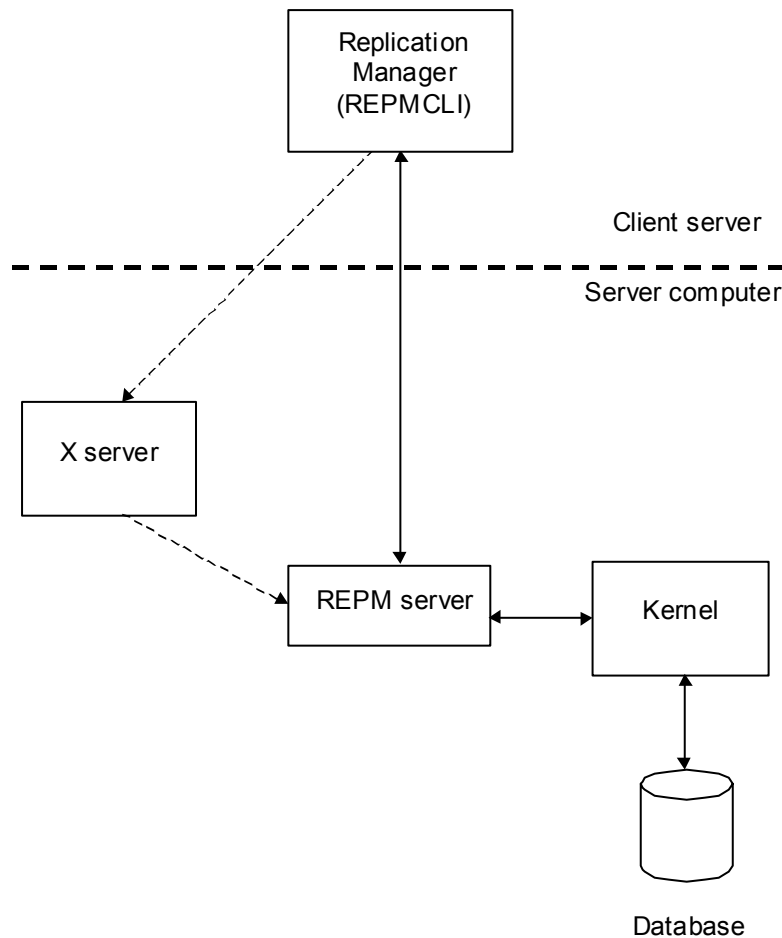


For more information about the architecture of the Web DBM, see [Architecture of the SAP DB Web Tools \[Page 46\]](#).



Architecture of the Replication Manager

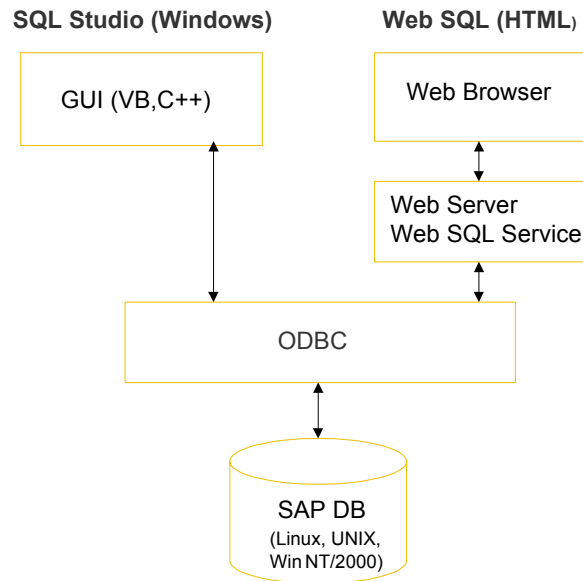
One example of the [architecture of the SAP DB tools \[Page 42\]](#) is the architecture of the Replication Manager. At the request of the client REPMCLI of the [Replication Manager \[Page 124\]](#), the [X-Server \[Page 133\]](#) starts the [REPM Server \[Page 125\]](#). After the REPM server has been started, communication takes place directly between the client and the REPM server.



SQL Studio Architecture

One example of the [architecture of the SAP DB tools \[Page 42\]](#) is the architecture of the SQL Studio. At the request of the client, [SQL Studio \[Page 128\]](#) GUI, a connection to the [database instance \[Page 113\]](#) is created using the ODBC interface. After this, communication can take place between the client and the database instance.

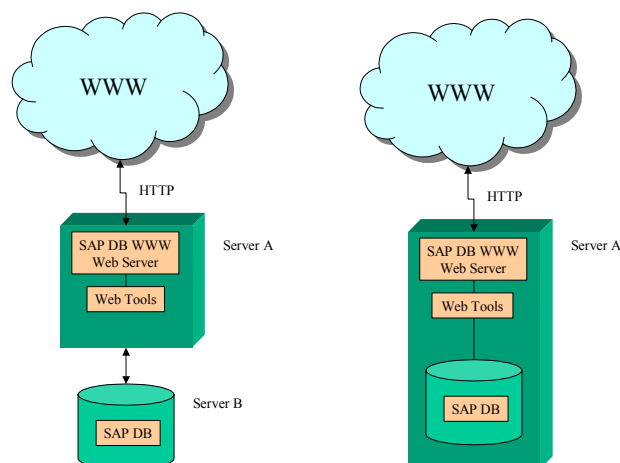
A connection to the client ([Web SQL \[Page 132\]](#) service integrated into the [Web Server \[Page 133\]](#) is created by calling the Web SQL in the Web Browser. The communication between this client and the database instance takes place through the ODBC interface (see also [Architecture of the SAP DB Web Tools \[Page 46\]](#)).



Architecture of the SAP DB Web Tools

One example of the [architecture of the SAP DB tools \[Page 42\]](#) is the architecture of the SAP DB Web tools. The SAP DB Web tools are implemented as a Web service and can be used with our own SAP DB Web Server as well as with current [Web servers \[Page 133\]](#) such as Apache or IIS.

The Web services are installed together with the Web Server on a server. The database system can be on a different server or on the same server as the Web server.



A connection to the client (Web DBM or Web SQL service integrated into the Web Server) is created by calling the SAP DB Web tool ([Web DBM \[Page 131\]](#) or [Web SQL \[Page 132\]](#)) in the Web Browser. The communication between the client and the [database instance \[Page 113\]](#) takes place through the [DBM Server \[Page 118\]](#) for the Web DBM tool, and through the ODBC interface for the Web SQL tool ([SQL Studio Architecture \[Page 45\]](#)).



X Server

The X Server is the communication instance of SAP DB software.

SAP DB client programs (such as database applications, DBMGUI, REPMCLI, and so on) first address the X Server when taking up a connection to a SAP DB Server program (database instance, DBM Server, REPM Server, and so on).

See also:

[Architecture of the Database Manager \[Page 43\]](#)

[Architecture of the Replication Manager \[Page 44\]](#)



DBM Server

The Database Manager Server (DBM Server) is the server part of the [Database Manager \[Page 114\]](#). It is installed by the server installation on the database server.

The DBM Server creates the connection to the [database instance \[Page 113\]](#) and can access its environment using operating system resources.

Client applications, such as the [Database Manager GUI \[Page 115\]](#), the [Database Manager CLI \[Page 114\]](#), or Web DBM, the program integrated into the [Web Server \[Page 133\]](#) create a connection to the DBM Server and exchange data with the DBM Server using a Request-Response mechanism.

See also:

[Architecture of the Database Manager \[Page 43\]](#)



REPM Server

The Replication Manager Server (REPM Server) is the server part of the [Replication Manager \[Page 124\]](#). The REPM Server must be installed on the server on which the external data (source files or target files) are stored.

The REPM Server creates the connection to the [database instance \[Page 113\]](#) and can access its environment using operating system resources.

The REPM Server can communicate remotely with the database instance and with the client.

See also:

[Architecture of the Replication Manager \[Page 44\]](#)

[Replication Manager: SAP DB \[Extern\]](#)



Web Server

The Web Server is the client part of the SAP DB Web tools [Web DBM \[Page 131\]](#) and [Web SQL \[Page 132\]](#). The server for the Web DBM tool is the [DBM Server \[Page 118\]](#), the server for the Web SQL tool is the [database instance \[Page 113\]](#).

Possible Web servers are the SAP DB Web Server or well-known Web servers such as Apache or IIS. The SAP DB Web server is installed during the installation of the SAP DB Web tools. Apache or IIS must be installed separately or configured. For more information, see the [Installation Guide Web Tools: SAP DB 7.3 \[Extern\]](#).

Web DBM and Web SQL are operating-system-independent. These tools are installed once, and can then be called from any browser. This enables occasional users as well as frequent users to make use of the Database Manager and SQL Studio functionalities quickly and easily in the network.

See also:

[Architecture of the SAP DB Web Tools \[Page 46\]](#)



Database Manager

The **Database Manager** is a tool for database administration.

The tasks of the Database Manager comprise creating, controlling, and monitoring [database instances \[Page 113\]](#) on the local computer or on remote computers. You can use the Database Manager to carry out backups and, if necessary, recoveries.

Architecture

The Database Manager consists of a server part and a client part.

Server/Client for the Database Manager

Server	Client
DBM Server [Page 118]	DBMGUI [Page 115] DBMCLI [Page 114] Web DBM [Page 131]

A script interface is available. The same functionality is offered for the Database Manager regardless of which client you use.

See also:

[Architecture of the Database Manager \[Page 43\]](#)

[Architecture of the SAP DB Web Tools \[Page 46\]](#)



Database Manager GUI

The [Database Manager \[Page 114\]](#) has a user-friendly graphical user interface, the Database Manager GUI (DBMGUI). If you want to use the Database Manager to monitor several SAP DB database instances, which may be on different computers, you should use the Database Manager GUI.

The Database Manager GUI can only be used on Windows operating systems. If you want to use the functionality of the Database Manager on other operating system platforms, you must use the [Database Manager CLI \[Page 114\]](#) or [Web DBM \[Page 131\]](#).

Call

You have the following options for calling the Database Manager GUI:

- Choose *Start* → *Programs* → *SAP DB* → *Database Manager*. Log on to the desired database instance.
- You can start the Database Manager GUI from the command line. In this case, you can transfer [options \[Page 49\]](#) to the DBMGUI program.

See also:

[Database Manager GUI: SAP DB 7.3 \[Extern\]](#) → [How the DBMGUI Works \[Extern\]](#)



Options (DBMGUI)

When you call the [Database Manager GUI \[Page 115\]](#) at command line level, you can specify options (*Database Manager GUI: SAP DB 7.3* → [Starting the DBMGUI \[Extern\]](#)). If you do not enter any options, you can make the entries needed for logging on to the database instance on the logon screen.



The following [user data as options \[Page 29\]](#) should be transferred: the [DBM operator \[Page 115\]](#) and the name of the [database instance \[Page 113\]](#)

```
dbmcli -u dbmmann,secret -d MK1
```

The Database Manager GUI is called and a database session is created for the DBM operator **dbmmann**, password **secret** with the registered database instance **MK1**.



Database Manager CLI

The [Database Manager \[Page 114\]](#) has a command-line oriented client, the Database Manager CLI (DBMCLI). The Database Manager CLI is operating-system-independent.

You can use the Database Manager CLI to carry out all Database Manager actions. You can also use the Database Manager CLI to schedule these actions in the background. You should use this option to automate Database Manager actions that have to be carried out regularly.

If you do not have a Windows operating system, you can only use Database Manager clients Database Manager CLI and [Web DBM \[Page 131\]](#) to manage database instances.

Call

```
dbmcli [<options>] [<command>]
```

You can transfer [options \[Page 50\]](#) and a maximum of one [DBM Server command \[Page 51\]](#) in a command line to the Database Manager CLI.

For a link to be established with a database instance on the local computer, you must enter at least the name of the database instance (option `-d <database_name>`) when calling the Database Manager CLI. If the required database instance is on a remote computer, you must also enter this computer name (option `-n <server_node>`).

You can open an interactive Database Manager CLI session if you do not enter any DBM Server commands other than the required options. You can then enter the required DBM server commands interactively.

You can write the required DBM Server commands to a separate file `<file_name>`. In this case, when you call the Database Manager CLI, you can enter option `-i <file_name>` in addition to the required options.

See also:

[Database Manager CLI: SAP DB 7.3 \[Extern\]](#) → [Functions of the Database Manager CLI \[Extern\]](#)



Options (DBMCLI)

A series of options can be transferred to the [Database Manager CLI \[Page 114\]](#) (*Database Manager CLI: SAP DB 7.3 → Calling the Database Manager CLI → Options when Calling the Database Manager CLI [Extern]*).

For a link to be established with the database instance on the local computer, you must enter at least option `-d <database_name>`. If you do not enter a command in the command line when calling the Database Manager CLI, an interactive Database Manager CLI session is opened.



The following [user data as options \[Page 29\]](#) should be transferred: [DBM operator \[Page 115\]](#) and the name of the [database instance \[Page 113\]](#) on the local computer:

```
dbmcli -u dbmmann,secret -d MK1
```

The Database Manager CLI is called and a database session is created for the DBM operator `dbmmann`, password `secret` with the registered database instance `MK1`. You can then enter the required DBM server commands interactively.



The name of the database instance on the local server should be transferred as an option. The database instance should be started.

```
dbmcli -d MK1
user_logon dbmmann,geheim
db_warm
exit
```

The Database Manager CLI is called and an interactive database session is created with registered database instance `MK1`. Details on the DBM operator `dbmmann`, password `secret` are made via the interactively entered DBM server command `user_logon`. Database instance `MK1` is started with DBM server command `db_warm`.



The name of the database instance on the local server should be transferred as an option. The database instance should be started. This action should be carried out in the background.

Create file `startMK1` with the following contents:

```
user_logon dbmann,geheim
db_warm
exit
```

Execute the command in the background

```
dbmcli -d MK1 -i startMK1
```

The Database Manager CLI is called and an interactive database session is created with registered database instance **MK1**. File **startMK1** is processed.



DBM Server Commands

DBM server commands can be transferred to the [Database Manager CLI \[Page 114\]](#) (*Database Manager CLI: SAP DB 7.3 → Calling the Database Manager CLI → [DBM Server Commands \[Extern\]](#)*).

Syntax

<command_name> [<parameters>]

A DBM Server command always consists of the command name and optional parameters that influence the execution of the command.



```
dbmcli -u dbmmann,secret -d MK1 param_getfull data_cache
```

The Database Manager CLI is called and a database session is created for the DBM operator **dbmmann**, password **secret** with the registered database instance **MK1**. DBM server command **param_getfull** is used to request all data for database parameter [DATA CACHE \[Page 70\]](#).



Web DBM

The [Database Manager \[Page 114\]](#) has a web-based client, the Web DBM. If you want to enable occasional users to use database administration functions, you should install the Web DBM once in the network. The Web DBM can then be called from all browsers.

If you do not have a Windows operating system, you can only use Database Manager clients [Database Manager CLI \[Page 114\]](#) and Web DBM to monitor database instances.

Web DBM provides you with basically the same functionality as the [Database Manager GUI \[Page 115\]](#), but it is operated as a web-based application.

Differences between Web DBM and DBMGUI

With the Database Manager GUI, you can monitor several database instances simultaneously. With the Web DBM, you can only monitor one database instance at a time. When you log on, you enter the database instance that you want to monitor. If you want to switch to another database instance, you must log off, and log on again entering the new database instance.

It is possible to open several browsers simultaneously. In this way, you can monitor a different database instance in each browser with the Web DBM.

In contrast to the Database Manager GUI, a timeout may occur when working with the Web DBM. This results in the connection to the database instance being broken. If this happens, you must log on again.

Set-up of the Web DBM

All administrative functions are listed on the left-hand side of the window in the Web DBM. The functions are divided up into the same groups as in the Database Manager GUI. The colored symbols (green dots and red squares) indicate whether an administrative function can

be used in the database instance in its present state. There is no menu bar or toolbar of the kind found in the DBMGUI.

Instead of a list of database instances registered in the DBMGUI, the Web DBM displays the detailed status of the database instance. The most important values displayed in the status have a link that takes you to the relevant function. The functions for starting and stopping the database instance are at the bottom of the status display.

The work area is located below the status display. The subsequent web page corresponding to the administrative function selected in the left-hand side of the window is displayed here.

The structure of these web page is similar to the corresponding screens in the DBMGUI. The bottom of these pages often contains a toolbar with the available administrative functions. If a new administrative function is selected in the Web DBM, and in a new page displayed in the work area as a result of this, the previous display and status is lost. To return to the previous display, you must reselect the relevant administrative function.

The header line contains a logoff link that enables you to end the current connection with a database instance.



Replication Manager

The Replication Manager provides developers and database administrators with a tool for loading data from external files into SAP DB databases and to unload data from SAP DB databases into external databases.

In addition to processing special load and unload commands, the Replication Manager can also execute all SQL statements. As a result, you can create command files that contain SQL statements for loading and unloading data. In particular, the DDL function for creating database catalog objects is indispensable in combination with the load command.

Architecture

The Replication Manager consists of a server part and a client part.

Server/Client for the Replication Manager

Server	Client
REPM Server [Page 125]	REPMCLI

A script interface is available. If you want to react to Replication Manager return codes, you must use this script interface.

See also: [Architecture of the Replication Manager \[Page 44\]](#)

Call

repmcli [<options>] -b <command_file>

The REPM Server is familiar with [options \[Page 53\]](#) and [REPM Server commands \[Page 53\]](#). The REPM Server commands must be stored in a command file (<command_file>). When you call the Replication Manager, you must enter this command file using -b <command_file>.

For a link to be established with a [database instance \[Page 113\]](#) on the local computer, you must enter at least the name of the database instance (option -d <database_name>) when calling the Replication Manager.

When creating the connection between the REPM Server and the database instance, the Replication Manager first uses the specified options. Next, it evaluates any commands about session creation that exist in the command file, and executes the commands for loading or unloading data and SQL statements.

See also:

[Replication Manager: SAP DB \[Extern\]](#) → [Current Functions of the Replication Manager \[Extern\]](#)



Options (REPMCLI)

A series of options can be transferred to the [Replication Manager \[Page 124\]](#) CLI (REPMCLI) (*Replication Manager: SAP DB* → [Options \[Extern\]](#)).

For a link to be established with the database instance on the local computer, you must enter at least option `-d <database_name>`. In addition, a command file containing the required [REPM server commands \[Page 53\]](#) must be entered (option `-b <command_file>`).



The following [user data as options \[Page 29\]](#) should be transferred: [database user \[Page 117\]](#) and the name of the [database instance \[Page 113\]](#)

```
repcli -u samplename,secret -d MK1 -b command.dat
```

The Replication Manager is called and a session is created for the database user **samplename**, password **secret**, with the database instance **MK1**. All further commands can be found in command file **command.dat**.



REPM Server Commands

As series of REPM server commands can be transferred to the [Replication Manager \[Page 124\]](#) CLI (REPMCLI) (*Replication Manager: SAP DB* → [REPM Server Commands \[Extern\]](#)) in a command file `<command_file>` (specification of option `-b <command_file>`).

An individual Replication Manager command consists of one or more keywords, and arguments and options.



The command file **command.dat** has the following contents:

```
DATALOAD TABLE cust
  cno      01-04
  lastname 06-12
  zip      14-18
  place    20-31
INFILE 'cust.data' FORMATTED
/
DATALOAD TABLE item
  ino      01-08
  descr    09-39          NULL IF POS 09-11 = '    '
  stock    40-43 INTEGER  NULL IF POS 40-43 INTEGER < '0'
  min_ord  44-45 INTEGER
  price    46-53 DECIMAL (2) NULL IF POS 1 <> 'X'
                                OR POS 46-53 DECIMAL < '0'
  weight   54-57 REAL
INFILE 'item.data' FORMATTED
/
```

```

USE USER sampleuser,secret serverdb MK2 on server2
/
SET TIMESTAMP ISO
/
SET MAXERRORCOUNT 100
/
FASTLOAD TABLE posting IF POS 4 >= '11.08.1999'
uno          1
cno          2
hno          3
from_dat     4
to_dat       5
price        6
INFILE 'master.data' DATE 'dd.mm.yyyy'

```



```
repmcli -u samplename,secret -d MK1 -b command.dat
```

The Replication Manager is called and a database session is created for the user **samplename**, password **secret**, with the database instance **MK1**. The REPM Server commands in the file **command.dat** are processed:

DATALOAD command: The tables **cust** and **item** are loaded.

USE USER command: A session is created for user **sampleuser**, password **secret**, with the database instance **MK2** on the server **server2**.

SET TIMESTAMP command: The format of the plain text values, in which TIMESTAMP columns are input and output is defined as ISO.

SET MAXERRORCOUNT command: The specified number of errors is ignored without terminating processing of the commands.

FASTLOAD command: The table **posting** is loaded.

See also:



SQL Studio: Introduction

With its user-friendly interface, SQL Studio allows easy access to data in an SAP DB database instance. You can create, execute, and manage any number of SQL statements. You can create, display, or change database catalog objects.

Architecture

The client part of the SQL Studio creates a connection to the [database instance \[Page 113\]](#) using the ODBC interface.

Server/Clients for SQL Studio

Server	Client
Database Instance	SQL Studio [Page 128] Web SQL [Page 132]

The same functionality is offered for the SQL Studio regardless of which client you use.

See also:

[SQL Studio Architecture \[Page 45\]](#)

[Architecture of the SAP DB Web Tools \[Page 46\]](#)



SQL Studio

The SQL Studio ([SQL Studio: Introduction \[Page 54\]](#)) has a user-friendly, graphical user interface.

The SQL Studio in the form described here can only be used on Windows operating systems. If you want to use the functionality of the SQL Studio on other operating system platforms, you must use [Web SQL \[Page 132\]](#).

Prerequisite

Check if the database instance is started.

Call

You have the following options for calling the graphical user interface of the SQL Studio:

- Choose *Start → Programs → SAP DB → SQL Studio*. Log on to the desired database instance.
- You can start SQL Studio from the command line. In this case, you can transfer [options \[Page 55\]](#) to the SQLSTO program.

See also:

[SQL Studio: SAP DB 7.3 \[Extern\]](#) → [Starting SQL Studio \[Extern\]](#)



Options (SQL Studio)

You can start [SQL Studio \[Page 54\]](#) (SQLSTO) from the command line. In this case, you can enter options (*SQL Studio: SAP DB 7.3 → [Starting SQL Studio \[Extern\]](#)*).

Procedure

1. Change to the directory, in which the program `sqlsto.exe` is stored.
2. Enter the following command:
`sqlsto [<options>]`



The following [user data as options \[Page 29\]](#) should be transferred: [database operator \[Page 117\]](#) and the name of the [database instance \[Page 113\]](#) on the local computer:

```
sqlsto -u samplename,secret -d MK1
```

SQL Studio is called and a database session is created for the database user `samplename`, password `secret`, with the database instance `MK1`.



Web SQL

The SQL Studio ([SQL Studio: Introduction \[Page 54\]](#)) has a web-based client, the Web SQL. If you want to enable occasional users to use SQL statements, you should install the Web SQL once in the network. The Web SQL can then be called from all browsers.

If you do not have a Windows operating system, you can only use the SQL Studio client Web SQL to send requests to the database instance.

Web SQL provides you with basically the same functionality as the [SQL Studio \[Page 128\]](#), but it is operated as a web-based application.

Prerequisite

Check if the database instance is started.

Call

Enter the following address in the browser:

http://<web_server>:<port>/websql

See *Web Tools Installation Guide: SAP DB 7.3* → [Using the SAP DB Web Tools \[Extern\]](#)

Logon dialog box

Enter the name of the database computer, the name of the database instance, the user name, and the user password.

Set-up of the Web SQL

Header

The header contains a link for logging off, among other things.

Stored SQL Studio Objects

Web SQL the same data as SQL Studio. The folders and SQL Studio objects stored in SQL Studio are visible in Web SQL. One exception is the SQL Studio 'Local Folder' for storing local SQL Studio objects.

Objects

Direct SQL objects can be read and written. In the case of the QueryByExample, VisualQuery and TableDefinition objects, the underlying SQL statement can be displayed, but these objects cannot be changed. Web SQL can create, delete, move, and rename folders and direct SQL objects itself.

Direct SQL Window

You can enter and execute SQL statements in this window. You can display SQL statements from all SQL Studio objects, and you can also store direct SQL objects. You can store newly created SQL statements.

Additional functions

Before executing each SQL statement, you can set the AutoCommit mode, the SQL mode, and the isolation level. You can display previously executed SQL statements using *Previous* and *Next*. A button is available for clearing the SQL window. You can execute several SQL statements. You must separate these from each other using a line with // or -- .

Result window

If SQL mode INTERNAL was selected for executing SQL statements, you can navigate flexibly through the result set.

If SQL mode INTERNAL was selected for executing SQL statements, you can zoom the column of data type LONG.

If one of the other SQL modes was selected for executing SQL statements, you can only navigate forwards in the result table.

If several SQL statements were executed, you can access the results and the messages relating to these SQL statements with a drop-down list.



Directory Structure: SAP DB for SAP Systems

Implementation Considerations

How the directories required for the SAP DB database system are distributed to the available hard disks has a significant impact on the security and performance of your database system. A good distribution of files on the hard disk must meet the following requirements:

- There must be enough free space to allow the database to expand
- The data must be stored securely.
- The hardware must meet the performance requirements.

[Distribution of the SAP DB Directories on the Hard Disk \[Page 57\]](#)

Conventions

[Variables \[Page 57\]](#)



Variables

The following table shows which variables are used in the explanation of the [Directory Structure: SAP DB for SAP Systems \[Page 57\]](#).

<version>	Version number of the SAP DB software
<database_name>	Name of database instance
<independent_data_path>	Data path independent of the version (IndepDataPath)
<independent_program_path>	Program path independent of the version (IndepProgPath)
<dependent_path>	Data path dependent on the version (InstRoot directory). This path specification must be explicit.



Distribution of the SAP DB Directories on the Hard Disk

The ideal [directory structure: SAP DB for SAP Systems \[Page 57\]](#) depends on the hardware configuration. There is no single solution or definition for the distribution of the files of an SAP DB [database instance \[Page 113\]](#).

When attempting to find the optimum data distribution for your database environment in a productive system, bear in mind the information in the following sections:

- [Security Requirements \[Page 65\]](#)
- [Performance Requirements \[Page 65\]](#)
- [Example Configuration \[Page 66\]](#)
- [Various Database Systems \[Page 66\]](#)

- [SAP DB Directories \[Page 59\]](#)



Security Requirements

Using the appropriate hardware, operating system, or database features can improve the downtime security of a [database instance \[Page 113\]](#).

If you want to ensure a high level of security, we recommend the use of RAID-5 or RAID-1 configurations for the [system devspace \[Page 129\]](#) and the [data devspace \[Page 112\]](#)s. A failure and the exchange of a disk does not impair database operation if the RAID system can carry out a restore process.

Every devspace category should be stored on a different disk.

- [Log Devspace \[Page 122\]](#)

In a productive system, the log devspaces should always be mirrored for security reasons. The DUAL [log mode \[Page 122\]](#) of the SAP DB database system can be used to this end. If the log entries are mirrored using other available hardware or software tools, then the SINGLE log mode can be used.

You should not use RAID-5 configurations for the log devspaces. RAID-5 systems do not allow full mirroring. Because the database instance writes log entries sequentially, this can lead to a loss in performance.

In a productive system, never use the DEMO log mode, even when you are using RAID systems.

- System Devspace, Data Devspaces

If you want to mirror the system devspace or the data devspaces, we recommend that you use RAID-1 configurations.

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.

UNIX: In a productive system, data devspaces and log devspaces should be used in conjunction with raw devices. In the event of a system crash, raw devices are extremely secure.



Performance Requirements

For performance reasons, each of the different types of [devspace \[Page 118\]](#) should be stored on a different disk. At the very least, [data devspace \[Page 112\]](#)s and [log-devspace \[Page 122\]](#)s should be stored on different disks. Because all changes to the database instance are logged in the [log areas \[Page 121\]](#), it is the log devspaces for a [database instance \[Page 113\]](#) that see the most write activity.

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.

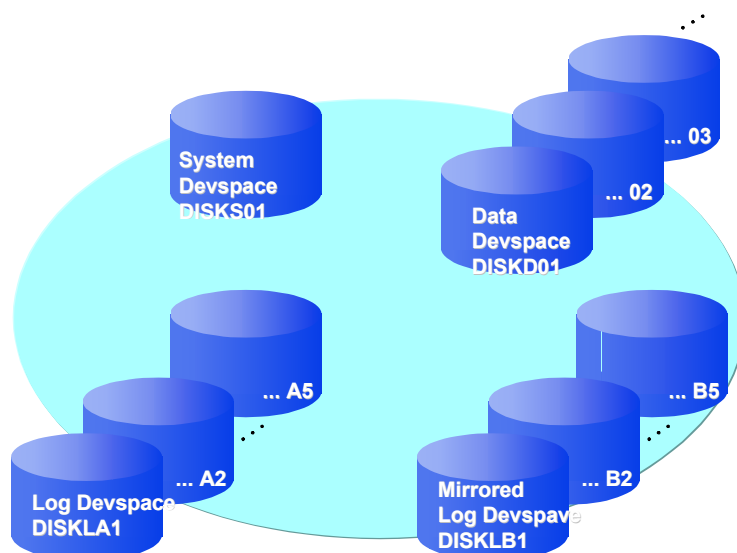
For performance reasons, the log devspaces must not be created on RAID-5 systems but only on dedicated disks or RAID-1 systems.

If swap or paging areas and log entries are kept on the same disk, performance will be negatively affected.

UNIX: Raw devices should be used for the data devspaces and log devspaces, because accessing to data in raw devices is generally quicker than accessing data in files.



Example Configuration



The [system devspace \[Page 129\]](#), the [data devspace \[Page 112\]](#)s and the [log devspace \[Page 122\]](#)s are stored on different disks. The log devspaces are mirrored.



Various Database Systems

For performance reasons, the SAP DB database system should not be installed on the same computer along with other database systems.

If you want to install several SAP DB [database instances \[Page 113\]](#), then every database instance should be installed on a separate computer. This will allow you to deal with possible performance problems more easily.



SAP DB Directories

The following table contains the [distribution of the SAP DB directories on the hard disk \[Page 57\]](#) for an SAP DB [database instance \[Page 113\]](#).

Conventions

[Variables \[Page 57\]](#)

SAP DB Directories

Directory name	Description
<independent_data_path>	IndepDataPath -Directory that, along with other data, contains the following instance data [Page 60] :

	<ul style="list-style-type: none"> • Configuration directory • Run directory [Page 126] of the database instance
<independent_data_path>/config	Configuration directory <ul style="list-style-type: none"> • Parameter files • Files for user authorization
<independent_data_path>/wrk/<database_name>	Run directory of the database instance <database_name> <ul style="list-style-type: none"> • Log files (knldiag) • Kernel trace files • Kernel dump files
<independent_program_path>	IndepProgPath -The directory that contains the programs that are independent of the database software version <ul style="list-style-type: none"> • Programs That Are Independent of the Database Software Version [Page 61] • Libraries for the Client Run-Time Environment [Page 61]
<dependent_path>	The directory (InstRoot) that contains the programs that are dependent on the database software version [Page 62]
/sapdb/<database_name>/data	Directory for devspace files <ul style="list-style-type: none"> • System Devspace [Page 129] • Data Devspaces [Page 112]
/sapdb/<database_name>/log	Directory for devspace files <ul style="list-style-type: none"> • Log Devspaces [Page 122] • Mirrored log devspaces
/sapdb/<database_name>/save	Directory for security files <ul style="list-style-type: none"> • Backups of the log entries or the data

See also:

[Example: SAP DB Directory Structure \[Page 62\]](#)

[Display SAP DB Directories \[Page 63\]](#)

[Define SAP DB Directories \[Page 63\]](#)



Instance Data

The instance data is the [run directories \[Page 126\]](#) required for the [database instance \[Page 113\]](#) and the configuration directory.

[SAP DB Directories \[Page 59\]](#)

[Variables \[Page 57\]](#)

The instance data is stored in the **IndepDataPath** directory: <independent_data_path>.

Run directory: <independent_data_path>/wrk/<database_name>

Configuration directory: <independent_data_path>/config



(UNIX)

/sapdb/data/wrk/LVC (Run directory for the LVC instance)
/sapdb/data/wrk/SDB (Run directory for the SDB instance)
/sapdb/data/wrk/P01 (Run directory for the P01 instance)
/sapdb/data/config (Configuration directory for all instances)



Programs That Are Independent of the Database Software Version

Programs that are independent of the database software version are only installed once for each computer, since they are needed for the database services that exist for each computer.

[SAP DB Directories \[Page 59\]](#)

[Variables \[Page 57\]](#)

The programs that are independent of the database software version are stored in the **IndepProgPath** directory: <independent_program_path>.

In this directory and its subdirectories, you will always find the programs for the most recently installed version of the database software.



(UNIX)

/sapdb/programs/bin/x_server
(version-independent program X_SERVER)
/sapdb/programs/pgm/dbmcli
(version-independent program DBMCLI)



Libraries for the Client Run-time Environment

The libraries required for the client run-time environment are installed once on each computer, but must be available in different versions.

[SAP DB Directories \[Page 59\]](#)

[Variables \[Page 57\]](#)

The libraries are stored in a subdirectory of the **IndepProgPath** directory:
<independent_program_path>/runtime/<version>



(UNIX)

/sapdb/programs/runtime/7240/lib



Programs That Are Dependent on the Database Software Version

The programs that are dependent on the database software version are installed once per [database instance \[Page 113\]](#). This means that the database software version of a database instance is independent of the database software versions of other database instances running on the same machine.

[SAP DB Directories \[Page 59\]](#)

[Variables \[Page 57\]](#)

The programs that are dependent on the database software version are situated in the directory `<dependent_path>`.



(UNIX)

`/sapdb/LVC/db` (InstRoot directory for the *liveCache* instance LVC)

`/sapdb/SDB/db` (InstRoot directory for the Content Server instance SDB)

`/sapdb/P01/db` (InstRoot directory for the database instance P01)



Client Tools

The client tools supported by SAP DB can be installed onto the database server or onto a computer of your choice.

The GUI clients are only installed once per computer. The settings for each user can be stored user-specifically.

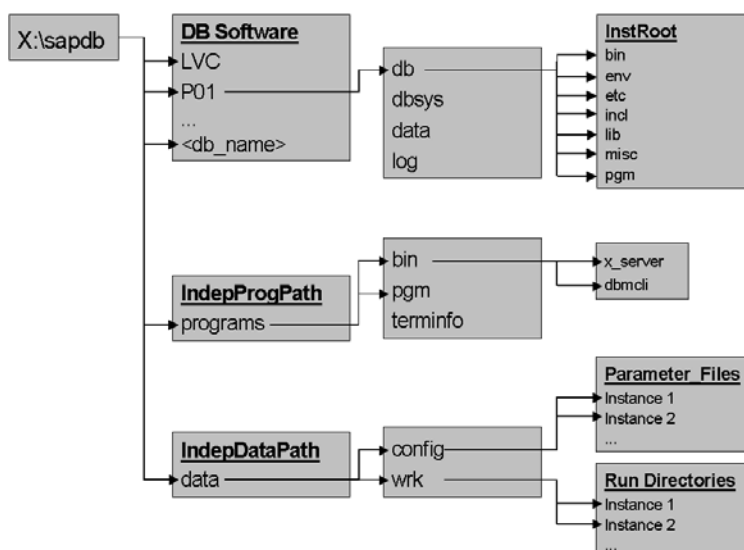
The directory for the client tools is freely-definable during the installation.

[SAP DB Directories \[Page 59\]](#)



Example: SAP DB Directory Structure

Example for the structure of [SAP DB directories \[Page 59\]](#)





Display SAP DB Directories

You can use the following DBMCLI commands to display the paths for the [SAP DB directories](#) [Page 59] **IndepDataPath**, **IndepProgPath** and **InstRoot**:

Conventions

[Variables](#) [Page 57]

Directory name	DBMCLI command
<independent_data_path>	dbm_getpath IndepDataPath
<independent_program_path>	dbm_getpath IndepProgPath
<dependent_path>	db_enum



Define SAP DB Directories

You can use the following DBMCLI commands to define the paths for the [SAP DB directories](#) [Page 59] **IndepDataPath**, **IndepProgPath** and **InstRoot**:

Conventions

[Variables](#) [Page 57]

Directory name	DBMCLI command
<independent_data_path>	dbm_putpath IndepDataPath <independent_data_path>
<independent_program_path>	dbm_putpath IndepProgPath <independent_program_path>
<dependent_path>	inst_reg <dependent_path>



Directory Structure: SAP DB for Open Source

Implementation Considerations

How the directories required for the SAP DB database system are distributed to the available hard disks has a significant impact on the security and performance of your database system. A good distribution of files on the hard disk must meet the following requirements:

- There must be enough free space to allow the database to expand
- The data must be stored securely.
- The hardware must meet the performance requirements.

[Distribution of the SAP DB Directories on the Hard Disk \[Page 64\]](#)

Conventions

[Variables \[Page 64\]](#)



Variables

The following table shows which variables are used in the explanation of the [Directory Structure: SAP DB for Open Source \[Page 64\]](#).

<version>	Version number of the SAP DB software
<database_name>	Name of database instance
<independent_data_path>	Data path independent of the database version (IndepDataPath) Linux For an RPM-based installation, you should define this directory as follows: <code>/var/opt/sapdb/indep_data</code>
<independent_program_path>	Program path independent of the database version (IndepProgPath) Linux For an RPM-based installation, you should define this directory as follows: <code>/var/opt/sapdb/indep_prog</code>
<dependent_path>	Data path dependent on the database version (InstRoot directory). This path specification must be explicit. Linux For an RPM-based installation, you should define this directory as follows: <code>/opt/sapdb/depend</code>



Distribution of the SAP DB Directories on the Hard Disk

The ideal [directory structure: SAP DB for Open Source \[Page 64\]](#) depends on the hardware configuration. There is no single solution or definition for the distribution of the files of an SAP DB [database instance \[Page 113\]](#).

When attempting to find the optimum data distribution for your database environment in a productive system, bear in mind the information in the following sections:

- [Security Requirements \[Page 65\]](#)
- [Performance Requirements \[Page 65\]](#)
- [Example Configuration \[Page 66\]](#)
- [Various Database Systems \[Page 66\]](#)
- [SAP DB Directories \[Page 66\]](#)



Security Requirements

Using the appropriate hardware, operating system, or database features can improve the downtime security of a [database instance \[Page 113\]](#).

If you want to ensure a high level of security, we recommend the use of RAID-5 or RAID-1 configurations for the [system devspace \[Page 129\]](#) and the [data devspace \[Page 112\]](#)s. A failure and the exchange of a disk does not impair database operation if the RAID system can carry out a restore process.

Every devspace category should be stored on a different disk.

- [Log Devspace \[Page 122\]](#)

In a productive system, the log devspaces should always be mirrored for security reasons. The DUAL [log mode \[Page 122\]](#) of the SAP DB database system can be used to this end. If the log entries are mirrored using other available hardware or software tools, then the SINGLE log mode can be used.

You should not use RAID-5 configurations for the log devspaces. RAID-5 systems do not allow full mirroring. Because the database instance writes log entries sequentially, this can lead to a loss in performance.

In a productive system, never use the DEMO log mode, even when you are using RAID systems.

- System Devspace, Data Devspaces

If you want to mirror the system devspace or the data devspaces, we recommend that you use RAID-1 configurations.

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.

UNIX: In a productive system, data devspaces and log devspaces should be used in conjunction with raw devices. In the event of a system crash, raw devices are extremely secure.



Performance Requirements

For performance reasons, each of the different types of [devspace \[Page 118\]](#) should be stored on a different disk. At the very least, [data devspace \[Page 112\]](#)s and [log-devspace \[Page 122\]](#)s should be stored on different disks. Because all changes to the database instance are logged in the [log areas \[Page 121\]](#), it is the log devspaces for a [database instance \[Page 113\]](#) that see the most write activity.

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,

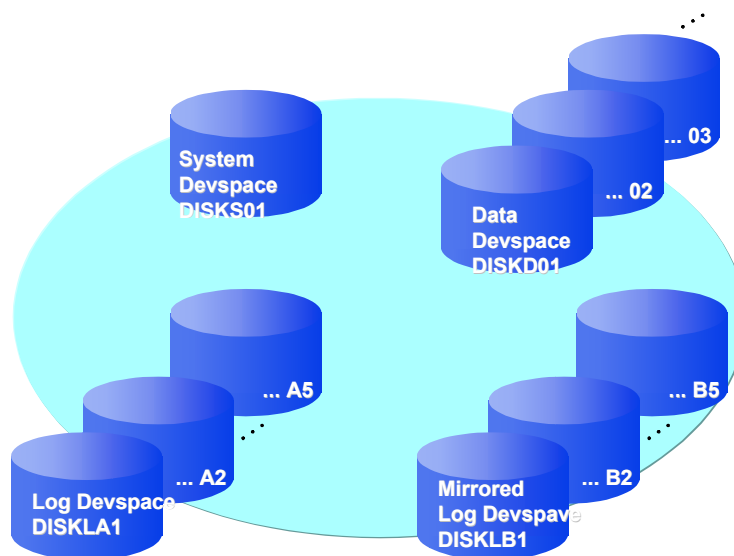
because some parallel mechanisms used by the database system depend on the number of configured data devspaces.

For performance reasons, the log devspaces must not be created on RAID-5 systems but only on dedicated disks or RAID-1 systems.

If swap or paging areas and log entries are kept on the same disk, performance will be negatively affected.

UNIX: Raw devices should be used for the data devspaces and log devspaces, because accessing to data in raw devices is generally quicker than accessing data in files.

Example Configuration



The [system devspace \[Page 129\]](#), the [data devspace \[Page 112\]](#)s and the [log devspace \[Page 122\]](#)s are stored on different disks. The log devspaces are mirrored.

Various Database Systems

For performance reasons, the SAP DB database system should not be installed on the same computer along with other database systems.

If you want to install several SAP DB [database instances \[Page 113\]](#), then every database instance should be installed on a separate computer. This will allow you to deal with possible performance problems more easily.

SAP DB Directories

The following table contains the [distribution of the SAP DB directories on the hard disk \[Page 64\]](#) for a SAP DB [database instance \[Page 113\]](#).

Conventions

[Variables \[Page 64\]](#)

SAP DB Directories

Directory name	Description
<independent_data_path>	IndepDataPath -Directory that, along with other data, contains the following instance data: <ul style="list-style-type: none"> Configuration directory Run directory [Page 126] of the database instance
<independent_data_path>/config	Configuration directory <ul style="list-style-type: none"> Parameter files Files for user authorization
<independent_data_path>/wrk/<database_name>	Run directory of the database instance <database_name> <ul style="list-style-type: none"> Log files (knldiag) Kernel trace files Kernel dump files
<independent_program_path>	IndepProgPath -The directory that contains the programs that are independent of the database software version Programs that are independent of the database software version are only installed once for each computer, since they are needed for the database services that exist for each computer. The libraries required for the client run-time environment are installed once on each computer, but must be available in different versions.
<dependent_path>	The directory (InstRoot) that contains the programs that are dependent on the database software version

The client tools supported by SAP DB can be installed onto the database server or onto a computer of your choice.

The GUI clients are only installed once per computer. The settings for each user can be stored user-specifically.

The directory for the client tools is freely-definable during the installation.

See also:

[Display SAP DB Directories \[Page 67\]](#)

[Define SAP DB Directories \[Page 68\]](#)



Display SAP DB Directories

You can use the following DBMCLI commands to display the paths for the [SAP DB directories \[Page 66\]](#) **IndepDataPath**, **IndepProgPath** and **InstRoot**:

Conventions

[Variables \[Page 64\]](#)

Directory name	DBMCLI command
<independent_data_path>	dbm_getpath IndepDataPath
<independent_program_path>	dbm_getpath IndepProgPath
<dependent_path>	db_enum



Define SAP DB Directories

You can use the following DBMCLI commands to define the paths for the [SAP DB directories \[Page 66\]](#) `IndepDataPath`, `IndepProgPath` and `InstRoot`:

Conventions

[Variables \[Page 64\]](#)

Directory name	DBMCLI command
<independent_data_path>	dbm_putpath IndepDataPath <independent_data_path>
<independent_program_path>	dbm_putpath IndepProgPath <independent_program_path>
<dependent_path>	inst_reg <dependent_path>



Database Parameters

To initialize the database parameters, 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 database parameters originally set to suit any specific requirements you may have.

Alternatively you can use the configuration of a [database instance \[Page 113\]](#) already present on the computer or the configuration from a [data backup \[Page 111\]](#). Even after this you can still adjust the database parameters to suit your own requirements.

General Database Parameters

DATA_CACHE [Page 70]	INSTANCE_TYPE [Page 70]	KERNELVERSION [Page 72]
LOG_MODE [Page 72]	LOG_SEGMENT_SIZE [Page 72]	MAXARCHIVELOGS [Page 73]
MAXBACKUPDEVS [Page 73]	MAXCPU [Page 73]	MAXDATADEVSPACES [Page 74]
MAXDATAPAGES [Page 74]	MAXLOCKS [Page 74]	MAXUSERTASKS [Page 74]
RESERVED_REDO_SIZE [Page 76]	RESTART_SHUTDOWN [Page 76]	RUNDIRECTORY [Page 77]

Special Database Parameters

BACKUP_BLOCK_CNT [Page 69]	CAT_CACHE_SUPPLY [Page 69]	CONVERTER_CACHE [Page 69]
DATE_TIME_FORMAT [Page 70]	DEADLOCK_DETECTION [Page 70]	DEFAULT_CODE [Page 70]
JOIN_MAXTAB_LEVEL4 [Page 71]	JOIN_MAXTAB_LEVEL4 [Page 70]	JOIN_SEARCH_LEVEL [Page 71]
KERNELDIAGSIZE [Page 71]	LOG_BACKUP_TO_PIPE [Page 72]	LOG_IO_QUEUE [Page 72]
LRU_FOR_SCAN [Page 73]	MAXRGN_REQUEST [Page 74]	MAXSERVERTASKS [Page 74]
MP_RGN_LOOP [Page 75]	OPTIM_BUILD_RESLT [Page 75]	OPTIM_FETCH_RESLT [Page 75]
OPTIM_KEY_INV_RATE [Page 75]	OPTIM_MAX_MERGE [Page 75]	OPTIM_ORDERBY_IDX [Page 76]
OPTIM_OR_DISTINCT [Page 76]	REQUEST_TIMEOUT [Page 76]	SEQUENCE_CACHE [Page 77]
SESSION_TIMEOUT [Page 77]	UTILITY_PROT_SIZE [Page 77]	_DATA_CACHE_RGNS [Page 78]
_EVENT_ALIVE_CYCLE [Page 78]	_MAXEVENTS [Page 78]	_MAX_MESSAGE_FILES [Page 78]
_ROW_RGNS [Page 78]	_TAB_RGNS [Page 79]	_TRANS_RGNS [Page 79]
_TREE_RGNS [Page 79]	_UNICODE [Page 79]	

**BACKUP_BLOCK_CNT**

The [database parameter \[Page 116\]](#) BACKUP_BLOCK_CNT denotes the block size in pages when data is backed up or restored.

**CAT_CACHE_SUPPLY**

The [database parameter \[Page 116\]](#) CAT_CACHE_SUPPLY denotes the memory size of the [catalog cache \[Page 18\]](#) in pages for all [user tasks \[Page 16\]](#).

**CONVERTER_CACHE**

The [database parameter \[Page 116\]](#) CONVERTER_CACHE denotes the size of the [converter cache \[Page 18\]](#) in pages.



DATA_CACHE

The [database parameter \[Page 116\]](#) DATA_CACHE denotes the size of the [data cache \[Page 19\]](#) in pages.



DATE_TIME_FORMAT

The [database parameter \[Page 116\]](#) DATE_TIME_FORMAT denotes the default format for displaying dates and times.



DEADLOCK_DETECTION

The [database parameter \[Page 116\]](#) DEADLOCK_DETECTION denotes 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 \[Page 76\]](#).

If DEADLOCK_DETECTION = 0, deadlock detection is disabled.



DEFAULT_CODE

If no other code attribute was specified, the [database parameter \[Page 116\]](#) DEFAULT_CODE denotes which code attribute was used to create columns of data type CHAR[ACTER], VARCHAR and LONG[VARCHAR] in the database instance.

See also:

Reference Manual: SAP DB 7.2 and 7.3 → Concepts → [Code Attribute \[Extern\]](#)



INSTANCE_TYPE

The [database parameter \[Page 116\]](#) INSTANCE_TYPE denotes the [database instance type \[Page 113\]](#) (SAP DB OLTP, liveCache, SAP DB OLAP, SAP DB Document Server, SAP DB E-Catalog).



JOIN_MAXTAB_LEVEL9

The [database parameter \[Page 116\]](#) JOIN_MAXTAB_LEVEL9 denotes the maximum number of tables allowed in a join when selecting the join sequence algorithm.

The default value is 4.

See also:

[JOIN_SEARCH_LEVEL \[Page 71\]](#)



JOIN_MAXTAB_LEVEL4

The [database parameter \[Page 116\]](#) JOIN_MAXTAB_LEVEL4 denotes the maximum number of tables allowed in a join when selecting the join sequence algorithm.

The default value is 16.

See also:

[JOIN_SEARCH_LEVEL \[Page 71\]](#)



JOIN_SEARCH_LEVEL

The [database parameter \[Page 116\]](#) JOIN_SEARCH_LEVEL determines the algorithm for the join sequence search. The level specified here determines how many resources and how much time the join sequence search takes.

- 9 (Join sequence search level 9): All possible join sequences are calculated.
- 4 (join sequence level 4): Various join sequences are calculated, depending on the query structure (transformer algorithm).
- 1 (join sequence search level 1): The simplest algorithm is used for the join sequence search (greedy algorithm).
- 0 (classified join sequence search level 0, level 0 is the default setting): The algorithm that is used for the join sequence search depends on the number of tables that were selected in join.
Joins with n number of tables, where $n \leq \text{JOIN_MAXTAB_LEVEL9 [Page 70]}$: Level 9 of the join sequence search is used.
Joins with n number of tables, where $\text{JOIN_MAXTAB_LEVEL9} < n \leq \text{JOIN_MAXTAB_LEVEL4 [Page 71]}$: Level 4 of the join sequence search is used.
Joins with n number of tables, where $\text{JOIN_MAXTAB_LEVEL4} < n$: Level 1 of the join sequence search is used.



JOIN_MAXTAB_LEVEL4 is set to 16.
JOIN_MAXTAB_LEVEL9 is set to 4.
JOIN_SEARCH_LEVEL is set to 0.
5 tables are used for a join.

Join sequence search level 4 (transformer algorithm) is used for this join because the database parameter $\text{JOIN_MAXTAB_LEVEL9} < 5 \leq \text{JOIN_MAXTAB_LEVEL4}$ is most appropriate.



KERNELDIAGSIZE

The [database parameter \[Page 116\]](#) KERNELDIAGSIZE denotes the size of log file of the [kernel \[Page 120\]](#) in pages.



KERNELVERSION

The [database parameter \[Page 116\]](#) KERNELVERSION denotes which database software version is being used.



LOG_BACKUP_TO_PIPE

The [database parameter \[Page 116\]](#) LOG_BACKUP_TO_PIPE determines whether or not a [log backup \[Page 121\]](#) is permitted in pipes.



LOG_IO_QUEUE

The [database parameter \[Page 116\]](#) LOG_IO_QUEUE denotes the size of the log queue in pages.



LOG_MODE

The [database parameter \[Page 116\]](#) LOG_MODE denotes [log mode \[Page 122\]](#) (SINGLE, DUAL, DEMO).

- SINGLE: Log entries are saved to a single [log devspace \[Page 122\]](#)
- DUAL: Log entries are saved to two (mirrored) log devspaces simultaneously
- DEMO: Log entries are not saved, but are overwritten by new entries



We recommend you always mirror the log devspaces. You can do this by either using the database software and setting the database parameter LOG_MODE to DUAL, or, if available, using hardware-based mirroring.

If your system allows hardware-based mirroring (RAID-1 systems), we advise you use it.



LOG_SEGMENT_SIZE

The [database parameter \[Page 116\]](#) LOG_SEGMENT_SIZE denotes the size of a log segment in the [log area \[Page 121\]](#) in pages.

The size specified here determines in what size parts the log entries are saved to during [interactive backups \[Page 119\]](#). It also determines the intervals at which log areas are [backed up automatically \[Page 108\]](#).

The size of the log segment depends on the size of the individual [log devspaces \[Page 122\]](#) and must not be greater than the sum of all log devspaces.

User-defined value	Log segment sizes used by the system
No definition of log segment size (LOG_SEGMENT_SIZE = 0)	A third of the total log area.

The log segment size in pages is smaller than or the same as 50% of the title log area.	The defined value for LOG_SEGMENT_SIZE.
The log segment > 50% of the total log area.	50% of the log area.



LRU_FOR_SCAN

The [database parameter \[Page 116\]](#) LRU_FOR_SCAN governs where data pages that have been scanned into the [data cache \[Page 19\]](#) are placed in the LRU list.

- **no** (default setting): The datapages scanned into the data cache are placed at the end of the LRU list. This means the datapages will be removed from the data cache promptly if new entries are made in the LRU list.
- **yes**: The datapages scanned into the data cache remain in the same place in the LRU list as they were before the scan.



MAXARCHIVELOGS

The [database parameter \[Page 116\]](#) MAXARCHIVELOGS denotes the maximum number of [log devspaces \[Page 122\]](#). This does not include mirrored log devspaces.



MAXBACKUPDEVS

The [database parameter \[Page 116\]](#) MAXBACKUPDEVS denotes the maximum number of files or tape devices that can be used in parallel.

You can speed up the process of backing up and restoring [data devspaces \[Page 112\]](#) by using more than one file or tape device in parallel.



MAXCPU

The [database parameter \[Page 116\]](#) MAXCPU denotes the maximum number of CPUs allowed on the [database instance \[Page 113\]](#). MAXCPU defines over how many CPUs the [user tasks \[Page 16\]](#) generating the main load are distributed.

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

This means you can define and also restrict the number of CPUs the database instance uses on multiprocessor computers ([Multiprocessor Configuration \[Page 123\]](#)). If you are using a single processor computer, set MAXCPU to 1.



MAXDATADEVSPACES

The [database parameter \[Page 116\]](#) MAXDATADEVSPACES denotes the maximum number of [data devspaces \[Page 112\]](#).



MAXDATAPAGES

The [database parameter \[Page 116\]](#) MAXDATAPAGES restricts the overall size (in pages) of all [devspaces \[Page 118\]](#).

You can only create another devspace when the value for MAXDATAPAGES is greater than the sum of the configured pages 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.

The database parameter MAXDATAPAGES is defined when the [database instance \[Page 113\]](#) is installed. Its default reserve is the same size as the largest possible devspace so a devspace of this size can be added if necessary.

If the reserve is filled to capacity once a devspace is added, the [Database Manager \[Page 114\]](#) increases the parameter MAXDATAPAGES by exactly one devspace at the next database instance restart.



MAXLOCKS

The [database parameter \[Page 116\]](#) MAXLOCKS determines the maximum number of lock list entries in which line or table locks of all users and their lock requests are stored.



MAXRGN_REQUEST

The [database parameter \[Page 116\]](#) MAXRGN_REQUEST denotes the maximum number of times a [task \[Page 129\]](#) can attempt to access a critical section. If this number is exceeded, the task lets another task access the CPU as long as it belongs to the same [user kernel thread \[Page 14\]](#).



MAXSERVERTASKS

The [database parameter \[Page 116\]](#) MAXSERVERTASKS denotes the maximum number of [server tasks \[Page 15\]](#) that are allowed when processing tasks.



MAXUSERTASKS

The [database parameter \[Page 116\]](#) MAXUSERTASKS denotes the maximum number of users that can work on a database at any one time ([user tasks \[Page 16\]](#), [database sessions \[Page 117\]](#)).

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.



MP_RGN_LOOP

The [database parameter \[Page 116\]](#) MP_RGN_LOOP denotes the maximum number of times a [task \[Page 129\]](#) may attempt to access a critical section that has been locked by another task. If this number is exceeded, the status of the task changes to "Waiting".



OPTIM_BUILD_RESLT

The [database parameter \[Page 116\]](#) OPTIM_BUILD_RESLT determines how the Optimizer's results are structured.

If the Optimizer estimates that the percentage of all pages of an index that need to be read in order to get results is greater than the figure defined in OPTIM_BUILD_RESLT, this index will not be used.



OPTIM_FETCH_RESLT

The [database parameter \[Page 116\]](#) OPTIM_FETCH_RESLT determines how the Optimizer is used.

If the Optimizer estimates that the percentage of all pages of an index that need to be read is greater than the figure defined in OPTIM_BUILD_RESLT and no results need to be built up, this index will not be used.



OPTIM_KEY_INV_RATE

The [database parameter \[Page 116\]](#) OPTIM_KEY_INV_RATE determines which optimizing algorithm should be used. This depends on whether it would be better to use the index or the primary key.

- Relationship1: The number of index pages that need to be read in relation to the size of the index
- Relationship2: The number of pages that need to be read using the primary key (primary data pages) in relation to the number of all primary data pages

If relationship 1 is OPTIM_KEY_INV_RATE percent smaller than relationship 2, the index will be used. If not, the data will be accessed directly using the primary key.



OPTIM_MAX_MERGE

The [database parameter \[Page 116\]](#) OPTIM_MAX_MERGE denotes how the optimizing algorithm for merging index lists changes.

If the number of pages of an index that need to be merged exceeds the value specified in `OPTIM_MAX_MERGE`, this index will not be used for an index merging strategy.



OPTIM_ORDERBY_IDX

The [database parameter \[Page 116\]](#) `OPTIM_ORDERBY_IDX` denotes how the optimization algorithm changes if you are using `ORDER BY` clauses.

If the `OPTIM_ORDERBY_IDX` values are quite high, the system will tend to use inversions to process SQL statements with `ORDER BY` clauses. If this is the case, the results do not need to be built up so the first results can be delivered quickly.

This parameter does not affect the join `SELECT` statements.



OPTIM_OR_DISTINCT

The [database parameter \[Page 116\]](#) `OPTIM_OR_DISTINCT` denotes how the optimization algorithm changes if you are using `OR` search conditions in SQL statements.

- Low `OPTIM_OR_DISTINCT` values: You should use low values for `OR` search conditions if you want `DISTINCT` hitlists where no lines are duplicated.
- High `OPTIM_OR_DISTINCT` values: You should use high values for `OR` search conditions if you do not want `DISTINCT` hitlists.



REQUEST_TIMEOUT

The [database parameter \[Page 116\]](#) `REQUEST_TIMEOUT` denotes the maximum amount of time (in seconds) you should have to wait for a lock to be lifted.

This database parameter limits the amount of time you have to wait until a lock is lifted by other users for all [database sessions \[Page 117\]](#).

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



RESERVED_REDO_SIZE

The [database parameter \[Page 116\]](#) `RESERVED_REDO_SIZE` denotes the size of the [area reserved for redo actions \[Page 124\]](#) in pages. If you set the `RESERVED_REDO_SIZE` value to 0, the database system sets the size of the redo area to half the size of the [log area \[Page 121\]](#).



RESTART_SHUTDOWN

The [database parameter \[Page 116\]](#) `RESTART_SHUTDOWN` determines whether [database instance \[Page 113\]](#) modes should be changed automatically or not.

The database parameter only takes effect if you are using a Windows operating system (NT, 2000, XP).

- **Manual:** All changes to the mode of the database instance must be initiated by the [database administrator \[Page 117\]](#).
- **Auto:** The following changes are made to the database instance automatically:
The Windows Service Manager switches the database instance straight to WARM mode when it starts.
A database instance in WARM mode will be shut down automatically if the operating system shuts down.



Since the shutdown of an operating system is subject to a time limit, it may interrupt the automatic shutdown of a database instance, particularly if you are working with large database instances where a high volume of data is being changed.

If you restart the database after the shutdown was interrupted, you must import log entries to restore the database instance. We therefore recommend you set the database parameter `RESTART_SHUTDOWN` to `manual`.



RUNDIRECTORY

The [database parameter \[Page 116\]](#) `RUNDIRECTORY` determines which directory is used as the [run directory \[Page 126\]](#) on the [database instance \[Page 113\]](#).



SEQUENCE_CACHE

The [database parameter \[Page 116\]](#) `SEQUENCE_CACHE` determines how big the sequence cache is in pages.



SESSION_TIMEOUT

The [database parameter \[Page 116\]](#) `SESSION_TIMEOUT` determines for how many seconds [database sessions \[Page 117\]](#) can remain inactive before being timed out.

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



UTILITY_PROT_SIZE

The [database parameter \[Page 116\]](#) `UTILITY_PROT_SIZE` denotes the size of the log file (`dbm.utl`) that was written in the [run directory \[Page 126\]](#) of the [database instance \[Page 113\]](#).



_DATA_CACHE_RGNS

The [database parameter \[Page 116\]](#) `_DATA_CACHE_RGNS` determines how many critical regions you can work with at once.



_EVENT_ALIVE_CYCLE

The [database parameter \[Page 116\]](#) `_EVENT_ALIVE_CYCLE` denotes the number of seconds an event cycle takes.



_MAXEVENTS

The [database parameter \[Page 116\]](#) `_MAXEVENTS` denotes the maximum number of events stored by the [kernel \[Page 120\]](#) in the cache for processing by the [Database Manager \[Page 114\]](#).



_MAX_MESSAGE_FILES

The [database parameter \[Page 116\]](#) `_MAX_MESSAGE_FILES` denotes the maximum number of trace files that can be opened at one time.



_ROW_RGNS

The [database parameter \[Page 116\]](#) `_ROW_RGNS` denotes the number of critical regions in which you can check lock collisions in rows.



_TAB_RGNS

The [database parameter \[Page 116\]](#) _TAB_RGNS denotes the number of critical regions in which you can check lock collisions in tables.



_TRANS_RGNS

The [database parameter \[Page 116\]](#) _TRANS_RGNS denotes the number of critical regions in which you can check lock collisions in [transactions \[Page 130\]](#) simultaneously.



_TREE_RGNS

Definition

The [database parameter \[Page 116\]](#) _TREE_RGNS denotes the number of critical regions in which you can check lock collisions in [B* trees \[Page 91\]](#) simultaneously.



_UNICODE

The [database parameter \[Page 116\]](#) _UNICODE determines whether user data and metadata for database objects is saved in [UNICODE \[Page 130\]](#).

This database parameter cannot be changed once the database system is installed.

See also:

[SAP DB as UNICODE Database \[Page 80\]](#) → [Installing a UNICODE-Enabled Database \[Page 80\]](#)



SAP DB as UNICODE Database

SAP DB can be used as a UNICODE database if the following versions or higher are used for the SAP DB components:

SAP DB Versions

SAP DB database software	7.3.00
C/C++ Precompiler, ODBC, SQL Studio, Web SQL	7.3.01

- [UNICODE \[Page 130\]](#)
- [Installing a UNICODE-Enabled Database \[Page 80\]](#)
- [UNICODE and SQL \[Page 82\]](#)
- [UNICODE in Programming Languages \[Page 85\]](#)



UNICODE

Data types such as CHAR ASCII and CHAR EBCDIC are mainly suited to English and central European languages. With other character sets, a code attribute is usually used for these data types. This code attribute uses a different presentation code to ASCII and EBCDIC, even for internal storage in the database system. This causes problems if you want to access these database systems using a different character set, or if you want to exchange data between database systems with different character sets.

You can avoid these problems by using internal character coding in accordance with UNICODE. Internally, the UNICODE data is stored in UTF-16/UCS-2 format. In UTF-16/UCS-2 format, all characters are two bytes long.

SAP DB is able to display various presentation codes in UNICODE format (UNICODE code in line with ISO 10646, page 1).

Metadata in UNICODE

The names of the database objects (such as table or column names) can be stored internally in UNICODE and can therefore then be displayed in the required presentation code in the database tools.

User data in UNICODE

SAP DB supports the code attribute UNICODE for the data types **CHAR[ACTER]**, **VARCHAR** and **LONG[VARCHAR]**.

See also:

[Installing a UNICODE-Enabled Database \[Page 80\]](#)

Reference Manual: SAP DB 7.2 and 7.3 → Concepts → [Code Attribute \[Extern\]](#)



Installing a UNICODE-Enabled Database

To make storage of [metadata \[Page 112\]](#) and [user data \[Page 131\]](#) in [UNICODE \[Page 130\]](#), proceed as follows:

Metadata in UNICODE

When installing the database instance, set the database parameter [_UNICODE \[Page 79\]](#) to YES.

User data in UNICODE

1. When installing the database instance, set the database parameter `_UNICODE` to YES.
2. Enter the code attribute UNICODE.



Please note that SAP DB UNICODE data is stored internally in UTF-16/UCS-2 format. As a result, double the space is required to store the UNICODE data in the database instance.

Procedure

[Setting Database Parameter _UNICODE \[Page 81\]](#)

[Setting Code Attribute UNICODE \[Page 82\]](#)



Setting Database Parameter _UNICODE

In order to [install a UNICODE-enabled database \[Page 80\]](#), the database parameter [_UNICODE \[Page 79\]](#) must be set to YES.



Please note that you cannot change database parameter `_UNICODE` once it has been set.

You can set the database parameter `_UNICODE` when you install the database instance with the Database Manager GUI or the Database Manager CLI.

Database Manager GUI

1. Start the Database Wizard.
2. Carry out the first four installation steps.
3. In step 5 (*Parameters*), choose *Extended*.
4. Set the `_UNICODE` parameter to YES.
5. Continue with the installation.

See also:

Database Manager GUI: SAP DB 7.3 → [Creating a New Database Instance \[Extern\]](#)

Database Manager CLI

A script with configuration information for the database instance contains, among other things, lines for definition of the database parameters. You must insert the following line at this point:

```
param_put _UNICODE YES
```

See also:

Database Manager CLI: SAP DB 7.3 → *Calling the Database Manager CLI* → *DBM Server Commands* → *Configuring Database Instances* → *Database Configuration Commands Overview* → [Parameter Value Change \[Extern\]](#)



Setting Code Attribute UNICODE

To be able to store the [user data \[Page 131\]](#) in [UNICODE \[Page 130\]](#), the database must be UNICODE-enabled and you must set the code attribute UNICODE for the required user data ([Installing a UNICODE-Enabled Database \[Page 80\]](#)). You can set the code attribute in the following ways:

- Database parameter [DEFAULT_CODE \[Page 70\]](#) is **not** set to value UNICODE.
In the column definition, make sure you enter **code attribute UNICODE**. User data is then only stored in UNICODE for these column values.
If you do not enter a code attribute, the code entered in the parameter DEFAULT_CODE (that is to say, not UNICODE) is used for these column values.
- Database parameter DEFAULT_CODE is set to value UNICODE.
In the column definition, enter code attribute UNICODE. User data is stored in UNICODE for these column values.
If you do not enter a code attribute, the user data is stored in UNICODE for these column values, because the parameter DEFAULT_CODE is set to UNICODE.



Database parameter DEFAULT_CODE is set to value UNICODE.

Column definition	Result
CHAR (n) UNICODE	UNICODE column
CHAR (n)	UNICODE column
CHAR (n) ASCII	ASCII column
CHAR (n) BYTE	Code-neutral, meaning that the column values are not converted by the database system

See also:

Reference Manual: SAP DB 7.2 and 7.3 → *Concepts* → [Code Attribute \[Extern\]](#)

For information on setting database parameters, see the following documentation:

- Database Manager GUI: SAP DB 7.3* → *Creating a New Database Instance* → [Displaying and Changing Current Database Parameters \[Extern\]](#)
- Database Manager CLI: SAP DB 7.3* → *Calling the Database Manager CLI* → *DBM Server Commands* → *Configuring Database Instances* → *Database Configuration Commands Overview* → [Parameter Value Change \[Extern\]](#)



UNICODE and SQL

UNICODE can be used for metadata, user data and in SQL statements if [installation of a UNICODE-enabled database \[Page 80\]](#) has been carried out.

UNICODE for metadata

If the database is UNICODE-enabled, all columns in the system tables that can be used to request the metadata have a data type with the code attribute UNICODE.

UNICODE for user data

To make user data UNICODE-enabled, you must [set the UNICODE code attribute \[Page 82\]](#) in a UNICODE-enabled database for the required user data. The [UNICODE \[Page 130\]](#) code

attribute can be used for the data types CHAR[ACTER] (n), VARCHAR (n) and LONG[VARCHAR]:

- CHAR[ACTER] (n) UNICODE
- VARCHAR (n) UNICODE
- LONG[VARCHAR] UNICODE

[Example 1 \[Page 83\]](#) illustrates the definition of Java class `TableDef`. Java class `TableDef` can be used to display the results of various column definitions.

Displaying the Column Definition of a Table

```
java TableDef <jdbcurl> <table_name>
```

Creating a temporary table using the determined column definitions, and displaying these column definitions

```
<command> ::= java TableDef <jdbcurl> <table_name>
```

```
<column_definition>
```

```
<jdbcurl> ::=
```

```
jdbc:sapdb:<database_name>?user=<userid>&password=<password>
```



```
java TableDef jdbc:sapdb:TST?user=TEST&password=TEST DUMMY a
varchar (20)
```

```
TABLE: DUMMY
```

```
A: VARCHARASCII (20)
```

UNICODE in SQL Statements

SQL statements can contain both UNICODE literals and UNICODE identifiers. The prerequisite for implementing these SQL statements is a UNICODE-enabled client (C/C++-Precompiler, JDBC, ODBC, SQL Studio or Web SQL).

The prerequisite for using UNICODE in the SQL Studio and Web SQL is that a UNICODE-enabled ODBC has been installed. SQL Studio and Web SQL are used on the operating systems NT/Windows 2000. These operating systems support UNICODE.



Example 1

The Java class `TableDef` can be used to display the results of various column definitions (see [UNICODE and SQL \[Page 82\]](#), section *UNICODE for user data*).

TableDef Definition

```
import java.sql.*;
/**
 *
 */
public class TableDef
{
    private Connection connection;
    /**
     * creates a new TableDef
     */
    public
    TableDef (
```

```

        String url)
throws SQLException, ClassNotFoundException
{
    // load class
    Class.forName ("com.sap.dbtech.jdbc.DriverSapDB");
    // create connection
    this.connection = DriverManager.getConnection(url,
        new java.util.Properties ());
    this.connection.setAutoCommit (false);
}
/**
 *
 */
protected void
createTable (
    String tableName,
    String createCommand)
throws SQLException
{
    String fullCommand = "CREATE TABLE " + tableName + " ("
        + createCommand + " )";
    Statement stmt = this.connection.createStatement ();
    stmt.execute (fullCommand);
}
/**
 *
 */
protected void
showTableDef (
    String tableName)
throws SQLException
{
    System.out.println ("Table: " + tableName); // #print
    DatabaseMetaData metaData = this.connection.getMetaData ();
    ResultSet tableColumns = metaData.getColumns(null,
        metaData.getUserName (), tableName, null);
    while (tableColumns.next ()) {
        String columnName = tableColumns.getString
("COLUMN_NAME");
        String typeName = tableColumns.getString ("TYPE_NAME");
        int colSize = tableColumns.getInt ("COLUMN_SIZE");
        System.out.println ("    " + columnName
            + ": " + typeName + " (" + colSize + ")"); // #print
    }
}
/**
 *
 */
protected void
close ()
{
    try {
        this.connection.rollback ();
        this.connection.close ();
    }
    catch (SQLException sqlExc) {
        // ignore
    }
}
/**
 *

```

```

    */
    static protected String
    join (
        String [] args,
        int startIndex)
    {
        if (startIndex >= args.length) {
            return null;
        }
        StringBuffer result = new StringBuffer ();
        for (int i = startIndex; i < args.length; ++i) {
            result.append(args [i]);
            result.append (' ');
        }
        return result.toString();
    }
    /**
     * used when called form the command line
     */
    public static void main (String [] args)
    throws ClassNotFoundException
    {
        String url = args [0];
        String tableName = args [1];
        String createCommand = join (args, 2);
        TableDef tableDef = null;

        try {
            tableDef = new TableDef (url);
            if (createCommand != null) {
                tableDef.createTable (tableName, createCommand);
            }
            tableDef.showTableDef (tableName);
        }
        catch (SQLException sqlExc) {
            System.out.println (sqlExc);
        }
        finally {
            if (tableDef != null) {
                tableDef.close ();
            }
        }
    }
}

```



UNICODE in Programming Languages

JDBC, ODBC and the C/C++ Precompiler support [UNICODE \[Page 130\]](#).

JDBC

Since Java works with UNICODE strings, it can read and write UNICODE columns.

If you also want to use UNICODE in SQL statements, you must set the `unicode` CONNECT-property to `true`. SQL statements are then transferred to the database instance in UTF-16/UCS-2 format. If the transfer package for the SQL statements is not large enough, you can increase its size using database parameter `_PACKET_SIZE`.

ODBC

UNICODE is supported in the ODBC driver.

Depending on your operating system, you must take account of the following factors:

Operating system	
Windows NT/Windows 2000	The ODBC driver only exports the UNICODE and/or Wide functions of the ODBC-API. ANSI functions are mapped to the relevant Wide functions by the <i>Windows Driver Manager</i> . This means that applications can use both the ANSI and the UNICODE functions of the ODBC-API.
UNIX/Linux	The use of the ODBC driver is currently not possible on platforms for which the standard UNICODE type <code>WCHAR_T</code> is defined with four bytes. The database and ODBC driver process UNICODE internally as values that are two bytes long. Both the ANSI and UNICODE variants of the ODBC-API are defined in the driver. Applications that do not require the functionality of a <i>Driver Manager</i> can be statically linked with the ODBC driver.

C/C++ Precompiler

During CONNECT, the C/C++ Precompiler checks whether the database is UNICODE-enabled ([Database parameter UNICODE \[Page 81\]](#) = YES). In UNICODE-enabled database, all SQL commands are transferred as UNICODE to the database instance.



```
EXEC SQL BEGIN DECLARE SECTION;

/* "SELECT tablename FROM domain.tables" encoded in UCS2
*/

SQLUCS2 sqlstmt[36] =
{0x0053, 0x0045, 0x004C, 0x0045,
0x0043,
0x0054, 0x0020, 0x0054, 0x0041,
0x0042,
0x004C, 0x0045, 0x004E, 0x0041,
0x004D,
0x0045, 0x0020, 0x0046, 0x0052,
0x004F,
0x004D, 0x0020, 0x0044, 0x004F,
0x004D,
0x0041, 0x0049, 0x004E, 0x002E,
0x0054,
0x0041, 0x0042, 0x004C, 0x0045,
0x0053,
0x0000};

SQLUCS2 resultstring [64];

EXEC SQL END DECLARE SECTION;

/* connect ... */
/* parse a unicode SQL statement and give it a statement
name */

EXEC SQL PREPARE stmt1 FROM :sqlstmt;
EXEC SQL DECLARE curs1 CURSOR FOR stmt1;
EXEC SQL OPEN curs1;

/* loop over resultset */
```

```

WHILE (sqlca.sqlcode != 100)
{
    EXEC SQL FETCH curs1 INTO :resultstring;
}
EXEC SQL CLOSE curs1;

```

For the complete example, see [Example 2 \[Page 87\]](#) (HelloUnicodeDB.cpc).

- The data type SQLUCS2 [] marks positive 16-bit wide whole numbers (unsigned short).
- Input and output variables can have data type CHAR [] or SQLUCS2 []. The values are converted into suitable data types internally, if required.
- SQL commands can have data type CHAR [] or SQLUCS2 [].

PYTHON, Perl

Python	UNICODE not supported	Planned for the end of 2001
Perl	UNICODE not supported	Planned for the end of 2001



Example 2

HelloUnicodeDB.cpc is a Precompiler program that uses UNICODE host variables (see [UNICODE in Programming Languages \[Page 85\]](#), Section *C/C++ Precompiler*).

HelloUnicodeDB.cpc Definition

```

/*****
**

module      : HelloUnicodeDB.cpc

-----
--

responsible : MarcoP

special area: demonstration program precompiler
description : demonstration program for unicode features of SAPDB
precompiler

The following steps are needed to execute the demo program:
1. customize the connect data in the embedded SQL program source
(line 43 - 46)

2. precompile/compile the embedded SQL program HelloUnicodeDB.cpc
   cpc [-u <userid>,<password> -d <database_name> -n <server_node>]
HelloUnicodeDB

3. linking the embedded SQL program HelloUnicodeDB.cpc
   cpclnk HelloUnicodeDB

last changed: 2000-04-30  17:17
see also    :

-----
--

```

```

copyright:      Copyright by SAP AG, 2001

*****
*/

/*=====
*
*   INCLUDES                                                    *
*=====
*/
#include <stdio.h>
#include <string.h>

/*=====
*
*   DEFINES                                                    *
*=====
*/

#define KNLIDNTFR 63      /* max. number of character for a unicode
kernel identifier*/

/* connect data */
#define USERID      "TEST"      /* username */
#define PASSWD      "TEST"      /* password */
#define SERVERNODE  "localhost" /* computer name */
#define SERVERDB    "TST"       /* name of database*/

/*=====
*
*   MACROS                                                    *
*=====
*/

/*=====
*
*   LOCAL CLASSES, STRUCTURES, TYPES, UNIONS ...            *
*=====
*/

/*=====
*
*   STATIC/INLINE FUNCTIONS (PROTOTYPES)                    *
*=====
*/

/* print UCS2 string as 7-bit Ascii, replace non-ascii characters
with '?'*/
static void printAs7bitAscii(SQLUCS2 *aUCS2Str, int length)
{
    int i;
    for (i = 0; i < length; i++) {
        if ( aUCS2Str[i] == 0x0000
            || aUCS2Str[i] == 0x0020 )
            return;
        if ( aUCS2Str[i] < 0x007f
            && aUCS2Str[i] > 0x0000)
            printf("%c",aUCS2Str[i]);
        else
            printf("?");
    }
}

```



```

}

/* format sql error for output */
static char *FormatSQLError(sqlcatype *sqlca)
{
    static char buffer[512];
#ifdef sql_oracle

    sprintf(buffer, "(%d):%.s", sqlca->sqlcode,
            sqlca->sqlerrml, sqlca->sqlerrmc);
#else
    sprintf(buffer, "(%d):%.s", sqlca->sqlcode,
            sqlca->sqlerrm.sqlerrml, sqlca->sqlerrm.sqlerrmc);
#endif
    return(buffer);
}

/*=====
 *  METHODS                                     *
 *=====
/

int main(int argc, char **argv)
{
    EXEC SQL BEGIN DECLARE SECTION;
    char *user          = USERID;
    char *pwd           = PASSWD;
    char *servernode    = SERVERNODE;
    char *serverdb      = SERVERDB;
    /* "SELECT TABLENAME FROM DOMAIN.TABLES " encoded in UCS2 */
    SQLUCS2 sqlstmt[36] = {0x0053, 0x0045, 0x004C, 0x0045, 0x0043,
0x0054, 0x0020, 0x0054, 0x0041, 0x0042,
                        0x004C, 0x0045, 0x004E, 0x0041, 0x004D,
0x0045, 0x0020, 0x0046, 0x0052, 0x004F,
                        0x004D, 0x0020, 0x0044, 0x004F, 0x004D,
0x0041, 0x0049, 0x004E, 0x002E, 0x0054,
                        0x0041, 0x0042, 0x004C, 0x0045, 0x0053,
0x0000};
    SQLUCS2 resultstring[64];
    EXEC SQL END DECLARE SECTION;

    /* set connect properties */
    EXEC SQL SET SERVERDB :serverdb ON :servernode;
    if (sqlca.sqlcode != 0 ) {
        printf("\n%s\n", FormatSQLError(&sqlca));
    }

    /* connect to database */
    EXEC SQL CONNECT :user IDENTIFIED BY :pwd;
    if (sqlca.sqlcode != 0 ) {
        printf("\n%s\n", FormatSQLError(&sqlca));
    }

    /* parse a unicode sql command and give it a statement name */
    EXEC SQL PREPARE stmt1 FROM :sqlstmt;
    if (sqlca.sqlcode != 0 ) {
        printf("\n%s\n", FormatSQLError(&sqlca));
    }

    /* declare a cursor for a prepared statement name */
    EXEC SQL DECLARE curs1 CURSOR FOR stmt1;

```

```

if (sqlca.sqlcode != 0 ) {
    printf("\n%s\n", FormatSQLError(&sqlca));
}

/* open the cursor "curs1" */
EXEC SQL OPEN curs1;
if (sqlca.sqlcode != 0 ) {
    printf("\n%s\n", FormatSQLError(&sqlca));
}

/* loop over resultset */
while (sqlca.sqlcode != 100)
{
    /* fetch a single row into a character hostvariable encoded in
    UCS2 */
    EXEC SQL FETCH curs1 INTO :resultstring;
    if (sqlca.sqlcode != 0 && sqlca.sqlcode != 100) {
        printf("\n%s\n", FormatSQLError(&sqlca));
    }
    else {
        printAs7bitAscii(resultstring, KNLIIDNTFR);
        printf("\n");
    }
}

/* close the cursor */
EXEC SQL CLOSE curs1;
if (sqlca.sqlcode != 0 ) {
    printf("\n%s\n", FormatSQLError(&sqlca));
}

/* commit and release the session */
EXEC SQL COMMIT WORK RELEASE;
if (sqlca.sqlcode != 0 ) {
    printf("\n%s\n", FormatSQLError(&sqlca));
}
}

/*=====
*
*   END OF CODE
*=====
*/

```



Data Management Using B* Trees

The SAP DB data management architecture ensures efficient data storage on disks and fast data access. The SAP DB database system performs automatic load balancing, thereby making reorganization unnecessary. The data areas (devspaces) can be extended online.

The developers of SAP DB have ensured that the structures required by users to implement an efficient I/O strategy are provided and that reorganization does not need to be carried out at all. For this reason, a special data management for the logical data pages has been created. This data management includes the following features:

- Sort data on SELECT([Table Access \(SELECT\) Using B* Tree \[Page 96\]](#))
- Sort data on INSERT([Table Access \(INSERT\) Using B*- Tree \[Page 98\]](#))

- UPDATE data in place ([Table Access \(UPDATE\) Using B* Tree \[Page 100\]](#))
- DELETE data in place ([Table Access \(DELETE\) Using B* Tree \[Page 99\]](#))

In addition to this special data management, the database system uses the following logical storage structures:

- Primary tables (with [primary keys \[Page 91\]](#) and [secondary keys \[Page 91\]](#))
- Secondary key tables (data of the secondary keys)
- [B* Tree \[Page 91\]](#)
There are build B* trees for the following tables ([B* Trees for Tables \[Page 94\]](#)): primary tables, secondary key tables
For tables with LONG columns (data type LONG also known as binary large objects (BLOBs)) there is a special data processing using B* trees for tables with LONG columns.



Concepts

[Primary Key \[Page 91\]](#)

[Secondary Key \[Page 91\]](#)

[B* Tree \[Page 91\]](#)

[Table ID \[Page 93\]](#)



Primary Key

Each SAP DB table has a primary key. The primary key is either defined by the [database user \[Page 117\]](#) or generated internally. A user-defined primary key can consist of multiple columns.



Secondary Key

Secondary keys can be defined for each table to optimize the data access via SQL statements. They can refer to any column combination and they help to prevent sequential scans over the table. Like the [primary key \[Page 91\]](#), the secondary key can consist of multiple columns.

A secondary key is often called an index (not to be confused with B* index).



B* Tree

All SAP DB data is stored in structures called B* trees. The B* tree method is far more efficient for accessing the rows of a table than other access methods (for example, sequential scans).

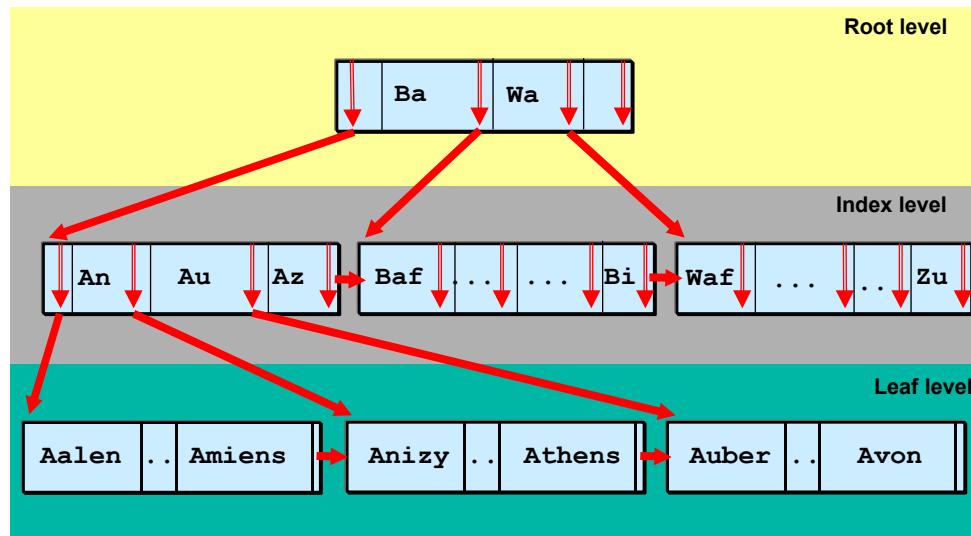
A B* tree spans several levels from the **root level** at the top, to the **index levels**, through to the lowest **leaf level** (one root level, n index levels (n>=1), one leaf level).

The storage units are called pages or data pages. SAP DB Version 7 supports a page size of 8 KB.

In SAP DB Version 7, all subsequent comparisons in a B* tree use ASCII code (each character is assigned a numerical value between 0 and 255). The comparisons are performed for all values on a character-by-character basis.

The sort criterion for building a B* tree is the [primary key \[Page 91\]](#).

B* Tree



See also:

[Root/Index Page \[Page 92\]](#)

[Leaf Page \[Page 93\]](#)

[Table Access \[Page 93\]](#)

[B* Trees for Tables \[Page 94\]](#)



Root/Index Page

Entries in root pages and index pages in a [B* tree \[Page 91\]](#) contain two sections.

- First section: This section contains an initial segment of the key fields of a table row. This section is called the **separator**. SAP DB uses only this part of the [primary key \[Page 91\]](#), which is required to distinguish the subsequent entries, thereby minimizing the storage space used in these pages.
The first separator in the leftmost page of each level (including the root level) contains only the pointer to the leftmost page of the next level down.
The average length of the separators depends on the structure and the selectivity of the primary key. If a very large number of key fields have to be evaluated to enable the entries (records) to be distinguished from each other, this results in longer separators.

- Second section: This section contains the **logical address** of a page at a lower index level or at the leaf level. The number of entries in this section depends on the length of the separators.



Leaf Page

Leaf pages in a [B* tree \[Page 91\]](#) contain the content of table rows. The number of entries in a leaf page depends on the total length of the table rows.

The leaf pages in the B* tree are sorted from left to right in ascending order. In other words, each leaf page contains only the table rows that fit its sort area.

The table rows are located in the first section (**data section**) of the leaf page. The rows in the data section are not sorted by default. The last section of the leaf page contains a **position list** with addresses that point to the corresponding table rows. The information in the position list is sorted.

The storage space required to store the entire table depends on the length of the separators and on the total length of table rows.



Table Access

The logical term **table** denotes possible primary data, including data in columns of type LONG and data of secondary key structures.

- As specified by its schema definition, a table has exactly one [B* tree \[Page 91\]](#) for the primary data, and exactly one B* tree for each [secondary key \[Page 91\]](#).
- If the table definition contains columns of data type LONG, an additional B* tree is created to accommodate all data of this type that does not exceed a specified length (data of type *short* LONG). Data items of type LONG that exceed this specific length are each stored in separate B* trees (data of type *long* LONG).

All searches and changes in the table are executed in the computer's main memory via SQL statements (SELECT, INSERT, UPDATE, DELETE,...), which means that they are carried out very quickly. The B* tree structure changes in all cases of INSERT, UPDATE and DELETE statements if there is not enough space in the target page for the new information or the fill level of a page falls below the predefined fill level.

See also:

[B* Trees for Tables \[Page 94\]](#)

[Table Access Using a B* Tree \[Page 96\]](#)



Table ID

The [database user \[Page 117\]](#) specifies a table by entering its name. The table is accessed internally using a table ID of a fixed length. The relationship between the table name and this table ID is stored in the database dictionary. In addition to this, SAP DB maintains a file directory in which the [root pages \[Page 92\]](#) of [B* trees \[Page 91\]](#) are assigned to Table IDs.

The Table IDs are stored in the file directory together with a type flag and other information. The type flag specifies whether the table contains primary data, [secondary key \[Page 91\]](#), or LONG data. The type flag specification enables one Table ID to be used for all the B* trees required for the logical structures of one table.



B* Trees for Tables

[B* trees \[Page 91\]](#) are set up for the following kinds of table:

- Tables with primary data
- Tables with [secondary keys \[Page 91\]](#)
- Tables with LONG columns

See also: [Table Access \[Page 93\]](#)

You can find some examples for setting up B* trees here:

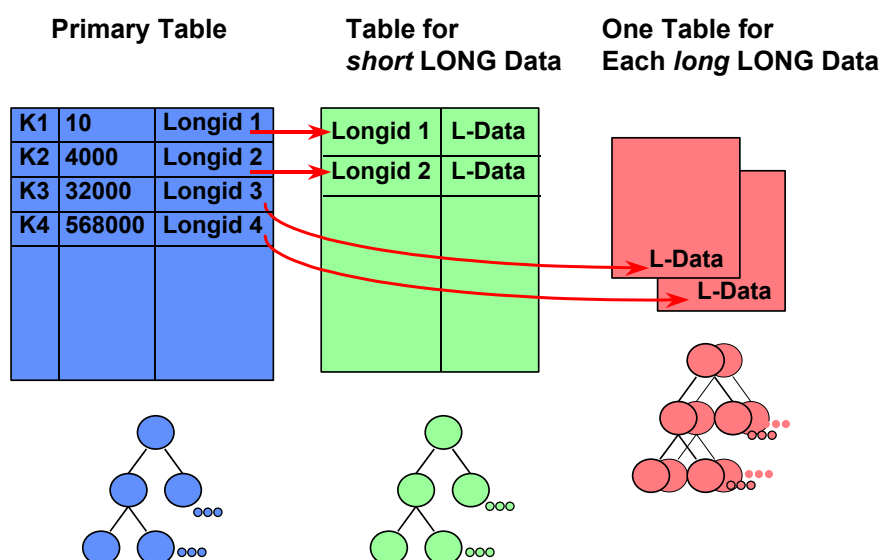
- [B* Trees for Tables with LONG Columns \[Page 94\]](#)
- [B* Trees for Tables with Secondary Keys \[Page 95\]](#)
- [B* Trees for Tables with LONG Columns and Secondary Keys \[Page 95\]](#)



B* Trees for Table with LONG Columns

An example of [B* trees for tables \[Page 94\]](#) is a [B* tree \[Page 91\]](#) structure in which the table has primary data and data of data type LONG.

B* Trees for Table with LONG Columns



The primary table contains a column of data type LONG. The numerical column defines the length of the LONG field for each line. There is one B* tree for the primary data, another for data of type *short* LONG and multiple B* trees for other data of type *long* LONG.

The primary table contains a fixed-length reference to the contents of the LONG field in each row. This reference points uniquely to the storage location of the content of the LONG field,

either as a key to a B* tree (*short* LONG) or as the [table ID \[Page 93\]](#) of one of the multiple B* trees (*long* LONG).



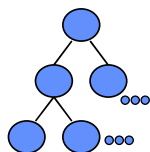
B* Trees for Tables with Secondary Key

An example of [B* trees for tables \[Page 94\]](#) is a [B* tree \[Page 91\]](#) structure in which the table has primary data and a [secondary key \[Page 91\]](#).

B* Trees for Tables with Secondary Key

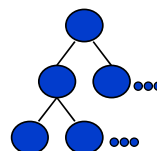
Primary Table

K1	10	10	
K2	20	10	
K3	10	10	
K4	20	40	
K5	10	10	
K6	20	40	
K7	40	30	



Secondary Key Table

1010	K1	K3	K5		
2010	K2				
2040	K4	K6			
4030	K7				



The table has a two-column multiple secondary key. There is one B* tree for the primary data and a second B* tree for the data of the secondary key table.



B* Trees for Tables with LONG Columns and Secondary Key

An example of [B* trees for tables \[Page 94\]](#) is a [B* tree \[Page 91\]](#) structure in which the table has primary data, a [secondary key \[Page 91\]](#), and data of data type LONG.

B* Trees for Tables with LONG Columns and Secondary Key

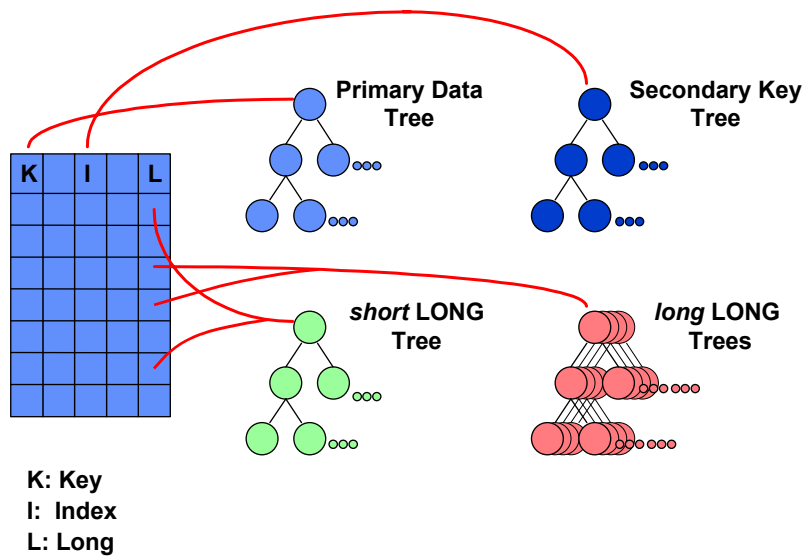


Table Access Using B* Tree

For all SELECT and DML (data manipulation language) statements, the database system uses the same search algorithm to determine the [leaf page \[Page 93\]](#) in which a table entry can be found or has to be changed.

Here you find some examples for table accessing using B* trees:

- [Table Access \(SELECT\) Using B* Tree \[Page 96\]](#)
- [Table Access \(INSERT\) Using B* Tree \[Page 98\]](#)
- [Table Access \(DELETE\) Using B* Tree \[Page 99\]](#)
- [Table Access \(UPDATE\) Using B* Tree \[Page 100\]](#)

See also:

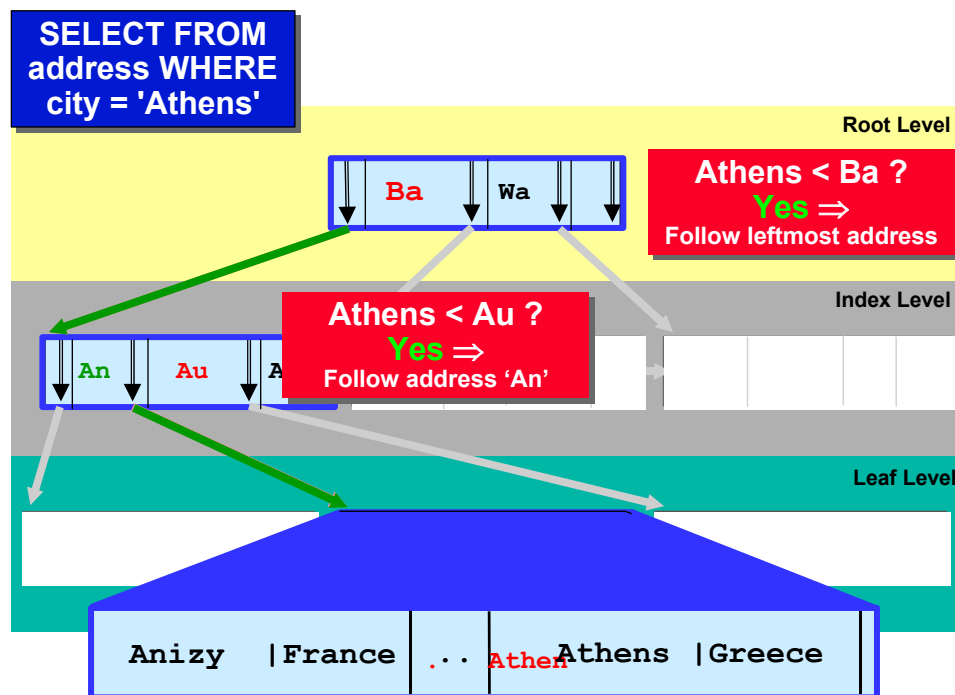
[Table Access \[Page 93\]](#)



Table Access (SELECT) Using B* Tree

[Table Access Using B* Tree \[Page 96\]](#) is illustrated using the example of a SELECT statement.

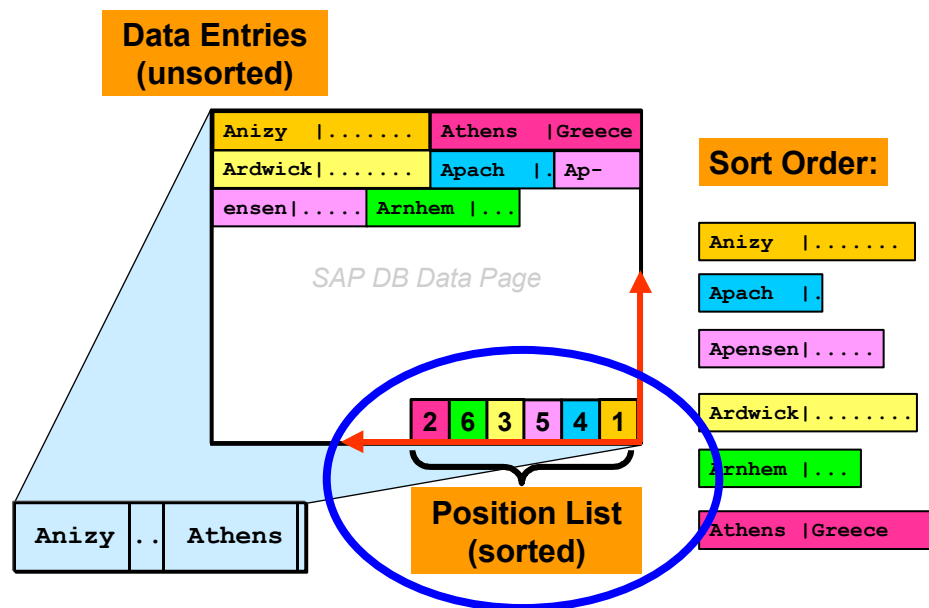
Table Access (SELECT) using B* Tree (1)



The *address* table is defined via the *city* [primary key \[Page 91\]](#) column. In the *address* table, the system is to search for an entry with value *Athens* for the *city* primary key field.

1. The search starts at the *root level* of the [B* tree \[Page 91\]](#). The database system compares the value *Athens* with the value of the first entry in the [root page \[Page 92\]](#), *Ba*. As the value *Athens* is lower than *Ba*, the relevant address information is evaluated. This points to an [index page \[Page 92\]](#).
2. The search continues at the *index level*. The value *Athens* is greater than the value of the first entry in the data page, *An*. The next value in the page is evaluated. As the value of *Athens* is smaller than the value of *Au*, the corresponding address information is evaluated. This points to a [leaf page \[Page 93\]](#).

Table Access (SELECT) using B* Tree (2)



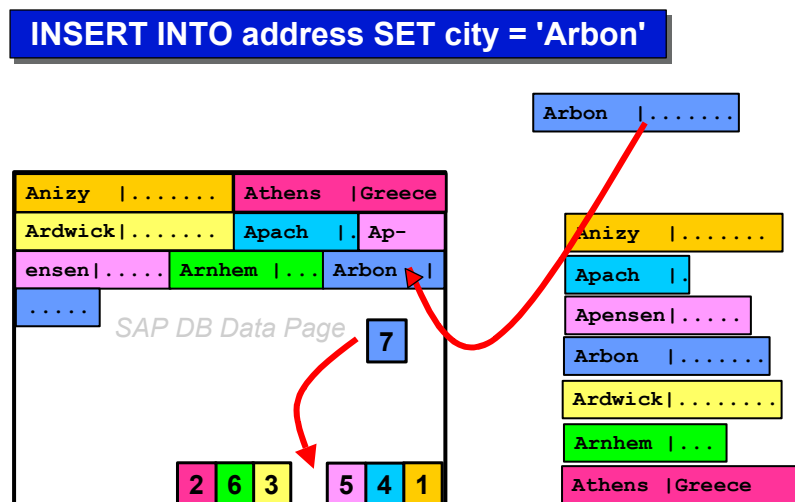
3. The search continues at the *leaf level*. In the position list of the leaf page a binary search algorithm is used. The corresponding address information for the table entry with the key value *Athens* is evaluated.
4. The database system scans the data section in the leaf page until it has found the corresponding table entries. The search is then complete.



Table Access (INSERT) Using B* Tree

[Table Access Using B* Tree \[Page 96\]](#) is illustrated using the example of a INSERT statement.

Table Access (INSERT) Using B* Tree



The *address* table is defined via the *city* [primary key \[Page 91\]](#) column. In the *address* table, insert an entry with the value *Arbon* for the *city* primary key field.

If there is enough space in the [leaf page \[Page 93\]](#) of the [B* tree \[Page 91\]](#) for the new entry, SAP DB inserts the entry at the end of the data section and updates the position list. The address of the new entry is written to the correct position in the position list. In the example above, this is position 4. Position 4 points to the new table entry number 7.

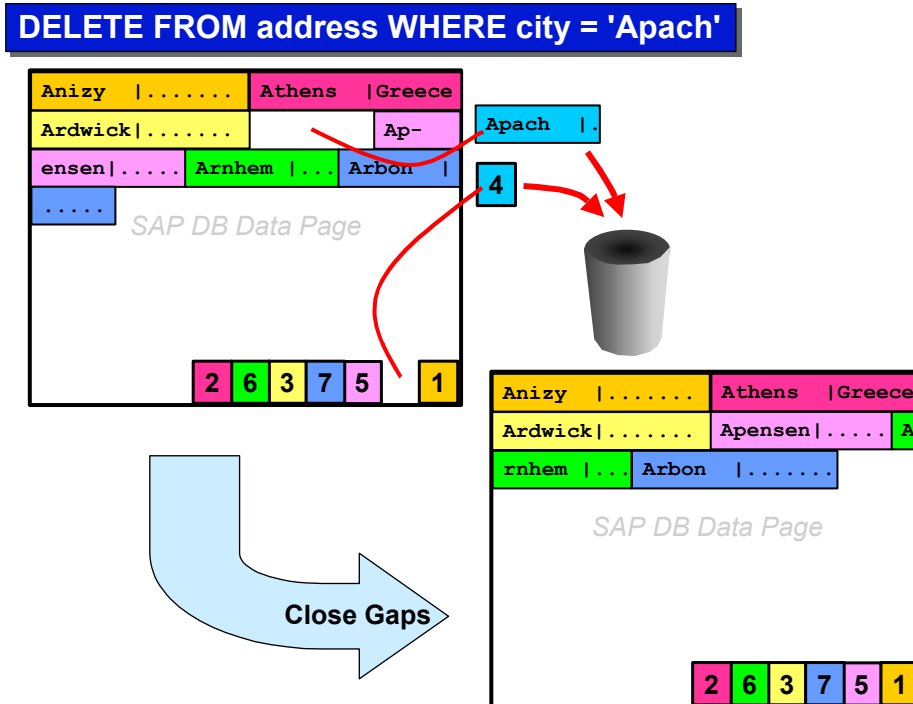
The position list and data section converge when data is inserted in a data page.



Table Access (DELETE) Using B* Tree

[Table Access Using B* Tree \[Page 96\]](#) is illustrated using the example of a DELETE statement.

Table Access (DELETE) Using B* Tree



The *address* table is defined via the *city* [primary key \[Page 91\]](#) column. In the *address* table, delete an entry with the value *Apach* for the *city* primary key field.

If an entry is deleted from the table, gaps appear in the data section and in the position list of the [leaf page \[Page 93\]](#) in the [B* tree \[Page 91\]](#). The database system closes these gaps and updates the position list to save storage space.



Table Access (UPDATE) Using B* Tree

[Table Access Using B* Tree \[Page 96\]](#) is illustrated using the example of an UPDATE statement.

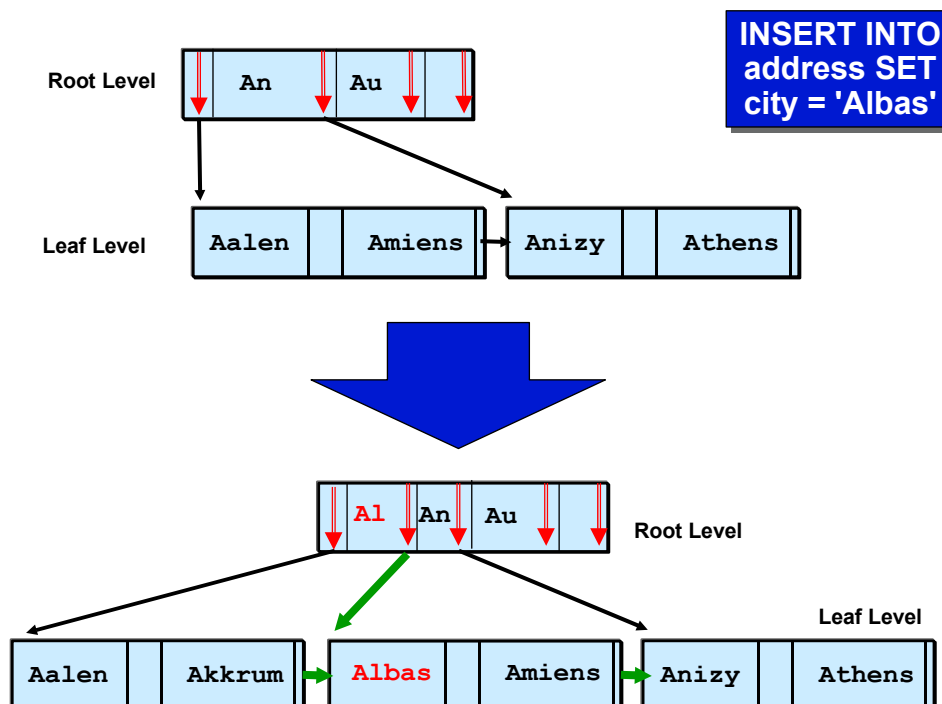
- If an UPDATE is performed and the [primary key \[Page 91\]](#) remains unchanged, only the contents of the table row are changed.
- If the data length changes, the positions of subsequent entries and their addresses in the position list of the [leaf page \[Page 93\]](#) in the [B* tree \[Page 91\]](#) are changed.
- If the key field changes, the UPDATE is executed as a DELETE ([Table Access \(DELETE\) Using B* Tree \[Page 99\]](#)) with a subsequent INSERT ([Table Access \(INSERT\) Using B* Tree \[Page 98\]](#)).



Changes in the B* Tree Structure

The [B* tree \[Page 91\]](#) structure changes in all cases of INSERT, UPDATE, and DELETE when there is either not enough space in the target page or the fill level falls below the predefined fill level.

Changes in the B* Tree Structure



The *address* table is defined via the *city* [primary key \[Page 91\]](#) column. In the *address* table, insert an entry with the value *Albas* for the *city* primary key field. The new table row is too large for the corresponding [leaf page \[Page 93\]](#) in the [B* tree \[Page 91\]](#).

- A new leaf page is created. The new row of the table is incorporated in this leaf page. In other words, the beginning of the sort area of the new leaf page is specified by this row of the table. The sort sequence of the individual leaf pages must be restored. To do so, it may be necessary to copy the table rows that belong in the new sort area from the leaf page with the next smallest sort area to the new leaf page, and then delete them from the previous leaf page. This procedure is called *page split*.
- Pointers to consecutive pages are updated whenever required.
- Address and separator information on the new page is also inserted in the corresponding [root or index page \[Page 92\]](#) one level higher. If the index page is too small to accommodate this information, a new index page is inserted at this index level as well. If the root page at the uppermost level is too small to accommodate a new entry, a new index level is created.

A chain of page splits can result in the B* tree being rebalanced.

See also:

[Non-Uniform Distribution of Data Pages \[Page 101\]](#)



Non-Uniform Distributions of Data Pages

[Changes in the B* tree structure \[Page 100\]](#) can lead to a non-uniform distribution of the data pages. If certain branches of the [B* tree \[Page 91\]](#) have more pages than others, data page distribution is not uniform. This can affect performance, because more accesses are generally required to find data.

Non-uniform distributions are detected by SAP DB when INSERT, UPDATE, or DELETE statements are performed, and the tree is rebalanced when the current operation is carried out. This means that the tree is constantly maintained for optimum operation. During this

procedure, page entries are moved to new locations and page pointers are redirected. As a result, data pages are used more efficiently.

Uniform distribution of data prevents individual data regions from overflowing. The only restriction on the size of tables is the storage space available in the database system.



SAP DB for Users of Version 6.1

Purpose

Development of the independent SAP DB software was begun in 1997 on the basis of version 6.1. Since then, SAP AG has significantly further developed and improved the SAP DB technology. The most important parts of the original software package have been completely redesigned and integrated into the SAP DB software. In this way, SAP DB meets the [requirements \[Page 102\]](#) for a modern database system.

- [SAP DB Improvements Since 1997 \[Page 102\]](#)
- [SAP DB tools \[Page 103\]](#)
- [Comparison of SAP DB and Version 6.1 \[Page 103\]](#)
- [Technical Specification of SAP DB Version 7.3 \[Page 104\]](#)
- [Improvements in SAP DB Version 7.4 \[Page 106\]](#)



Requirements for a Database System

- High performance
- High availability
- Easy to operate
- Low cost of ownership



SAP DB Improvements Since 1997

- Optimized checkpoint handling
- Parallel creation of indexes
- Parallel backup and restore
- Optimized locking and logging
- Better handling of “log full” and “database full” errors
- Interactive support for the restore process
- Reduced converter cache size (factor 10)
- Improved monitoring and diagnostics
- Unicode support (UTF-16)
- Support for the Linux operating system

- 64 bit UNIX versions as operating system platform
- New database tools



SAP DB Tools

Database Management

You can use the **Database Manager** to manage databases. You can use the following clients:

- DBMGUI (Windows NT, Windows 2000)
- DBMCLI (command line program)
- Web DBM (HTML/http)

You can make additional clients available using a script interface to Perl or Python.

Entering SQL Statements

You can use **SQL Studio** to enter SQL statements, such as data requests. You can use the following clients:

- SQL Studio (Windows NT, Windows 2000)
- Web SQL (HTML/http)

Unloading and Loading of Data

You can use the **Replication Manager** to unload and load data. You can use the following client:

- REPMCLI (command line program)

You can make additional clients available using a script interface to Perl or Python.



Comparison of SAP DB and Version 6.1

Comparison of Selected Maximum Values

	SAP DB 7.3	Version 6.1
Identifier length	32 characters	18 characters
Numeric precision	38 places	18 places
SQL statement length	>= 16 KB	8240 Bytes
Internal length of a table row	8088 Bytes	4047 Bytes
Number of columns per table (with KEY)	1024	255
Number of columns per table (without KEY)	1023	254
Number of key columns per table	512	127
Total of internal lengths of all key columns	1024 Bytes	255 Bytes
Number of results columns in a SELECT statement	1023	254
Number of join tables in a SELECT statement	64	16

Number of join conditions in a WHERE clause of a SELECT statement	128	64
Length of columns in an ORDER or a GROUP clause	1020 Bytes	250 Bytes
Number of parameters in an SQL statement	2000	300
Field length n for data type CHAR(n) Unicode/Non-Unicode	4000/8000 characters	-/4000 characters

Comparison of Functions

SAP DB 7.3	Version 6.1
No checkpoints – no wait situations. Checkpoints only at restart and shutdown of the database.	Checkpoints cause wait situations. Checkpoints at log and data backup, restart and shutdown, and at log segment completion.
No database downtime at database full. You can add devspaces without restarting. Application sessions remain open.	Database downtime at database full. You must restart the database after adding devspaces. You must restart applications.
No database downtime at log full. Wait for log save completion with running database. Application sessions remain open.	Database downtime at log full. You must backup the log. You must restart applications.
Parallel restore log. Restore log uses one read task and multiple redo tasks. Redo starts immediately. Multiple redo sessions.	Sequential restore log. Restore log copies all log entries from tape to the log device. Redo waits until copy completion. Single redo session.
You can automatically backup log in parallel to data backup.	Log and data backup are mutually exclusive.
Optimized lock list.	Locklist can contain up to 1000 entries.
Asynchronous mass DELETE and deletion of a table.	Synchronous mass DELETE and deletion of a table.



Technical Specification of SAP DB Version 7.3

Restrictions

	Maximum Value
Database size	32 TB (with 8 KB page size)
Number of files/devspaces per database	256
File/devspace size (data)	128 GB (depending on operating system limitations)
File/devspace size (log)	16 TB (depending on operating system limitations)

Number of sessions per user	8
SQL statement length	>= 16 KB (Default value 64 KB, defined by a system variable)
SQL character string length	Defined by a system variable
Identifier length	32 characters
Numeric precision	38 places
Number of tables	unlimited
Number of columns per table (with KEY)	1024
Number of columns per table (without KEY)	1023
Number of primary key columns per table	512
Number of columns in an index	16
Number of foreign key columns per table	16
Number of columns in an ORDER or a GROUP clause	128
Number of columns in a SELECT statement	1023
Number of columns in an INSERT statement	1024
Number of columns in a results table	1023
Number of join tables in a SELECT statement	64
Number of triggers per Basis table	3
Number of UNIQUE indexes per table	510
Number of indexes per table	510
Number of named indexes per table	255
Number of referring CONSTRAINT definitions (foreign key dependencies) per table	unlimited
Number of references per table	unlimited
Number of rows per table	limited by database size
Internal length of a table row	8088 Bytes
Total of internal lengths of all primary key columns	1024 Bytes
Total of internal lengths of all foreign key columns	1024 Bytes
Total of internal lengths of all columns belonging to an index	1024 Bytes
Internal length of a LONG column	2 GB
Length of columns in an ORDER or a GROUP clause	1020 Bytes
Nested trigger levels	unlimited
Nested subqueries	127
Number of join conditions in a WHERE clause of a SELECT statement	128
Number of correlated columns in an SQL statement	64
Number of correlated tables in an SQL statement	16
Number of parameters in an SQL statement	2000



Improvements in SAP DB Version 7.4

System devspace dropped

- Converter pages are spread over all data devspaces
- No need to configure system devspace size
- No need to extend system devspace to add data devspaces

Converter cache dropped

- The converter and data pages are held in the data cache
- Single configuration parameter for data cache size

Database size reduced

- High water mark for database fill rate is independent of the database size
- Number of data devspaces at restart time can be smaller than at backup time

Parallel converter queries

- Multi-region converter:
Converter queries can be made to multiple critical regions in parallel.

Improved log management

Separation of before and after images

- Before images are stored in transaction-specific UNDO files
- After images are stores in the log devspace

Checkpoints are obsolete

- Each database backup is consistent, even without the entries in the log devspace
- The database can be started when the log is missing

Parallel archiving of log entries

- You can configure the number of parallel log devspaces
- You can configure the number of parallel log wait queues

Parallel writing of before images

- Before images for transactions executed in parallel can be written in parallel, as the system stores these in transaction-specific UNDO files.

Improved backup

- Incremental data backups contain all changes since the last full data backup
- The restore process is therefore simplified. When restoring, you must only import the last full data backup and the **last** incremental data backup.



Terms

[Automatic log backup \[Page 108\]](#)
[Backup history \[Page 109\]](#)
[Backup ID \[Page 109\]](#)
[Backup medium \[Page 110\]](#)
[Cache \[Page 110\]](#)
[Catalog \[Page 110\]](#)
[Checkpoint \[Page 111\]](#)
[Data backup \[Page 111\]](#)
[Data devspace \[Page 112\]](#)
[Database administrator \[Page 112\]](#)
[Database catalog \[Page 112\]](#)
[Database instance \[Page 113\]](#)
[Database instance type \[Page 113\]](#)
[Database Manager \[Page 114\]](#)
[Database Manager CLI \[Page 114\]](#)
[Database Manager GUI \[Page 115\]](#)
[Database Manager operator \(DBM operator\) \[Page 115\]](#)
[Database parameters \[Page 116\]](#)
[Database session \[Page 117\]](#)
[Database system administrators \(SYSDBA\) \[Page 117\]](#)
[Database user \[Page 117\]](#)
[DBA/DOMAIN \[Page 117\]](#)
[DBM Server \[Page 118\]](#)
[DBMCLI \[Page 118\]](#)
[DBMGUI \[Page 118\]](#)
[Devspace \[Page 118\]](#)
[External backup ID \[Page 119\]](#)
[External backup medium \[Page 119\]](#)
[External backup tool \[Page 119\]](#)
[Instance type \[Page 119\]](#)
[Interactive log backup \[Page 119\]](#)
[Kernel \[Page 120\]](#)
[Language support \(Mapchar sets\) \[Page 120\]](#)
[liveCache \[Page 120\]](#)
[Lock \[Page 121\]](#)
[Log area \[Page 121\]](#)
[Log backup \[Page 121\]](#)
[Log devspace \[Page 122\]](#)

[Log mode \[Page 122\]](#)
[Multiprocessor configuration \[Page 123\]](#)
[Name of standard backup medium \[Page 123\]](#)
[Name of external backup medium \[Page 124\]](#)
[Parallel backup media \[Page 124\]](#)
[Redo area \[Page 124\]](#)
[Replication Manager \[Page 124\]](#)
[REPM Server \[Page 125\]](#)
[RESOURCE \[Page 126\]](#)
[Run directory \[Page 126\]](#)
[SAP DB Document Server \[Page 126\]](#)
[SAP DB E-Catalog \[Page 126\]](#)
[SAP DB OLAP \[Page 127\]](#)
[SAP DB OLTP \[Page 127\]](#)
[SAP DB tools \[Page 127\]](#)
[SAP DB user classes \[Page 127\]](#)
[Savepoint \[Page 127\]](#)
[Session \[Page 128\]](#)
[Single backup medium \[Page 128\]](#)
[SQL mode \[Page 128\]](#)
[SQL Studio \[Page 128\]](#)
[System devspace \[Page 129\]](#)
[Task \[Page 129\]](#)
[Terminal support \(Termchar sets\) \[Page 129\]](#)
[Thread \[Page 129\]](#)
[Transaction \[Page 130\]](#)
[UNICODE \[Page 130\]](#)
[User \[Page 131\]](#)
[User data \[Page 131\]](#)
[Web DBM \[Page 131\]](#)
[Web SQL \[Page 132\]](#)
[Web Server \[Page 133\]](#)
[X Server \[Page 133\]](#)



Automatic Log Backup

Automatic [log backup \[Page 121\]](#) is recommended to ensure the safety of data in producing systems.

If automatic log backup is activated, a log segment is saved as soon as it has been filled. This log segment is then released again. This has the advantage that an overflow of the [log area \[Page 121\]](#) is almost impossible.

This mechanism is particularly recommended for all [database instances \[Page 113\]](#) in which extensive write and change-intensive transactions are carried out. In this way, constant monitoring of the usage level of the log area is not necessary.



While the automatic log backup is enabled, no other log backup can be performed, although a [data backup \[Page 111\]](#) can be performed.

You can perform log backups in `WARM` mode.

See also:

[Backup Strategy \[Page 37\]](#)

[Saving Log Backups \[Page 41\]](#)



Backup History

Information on all actions that have been carried out that relate to saving and recovering the [database instance \[Page 113\]](#) are registered in chronological order in the backup history.

The system writes a backup history to log file `dbm.knl`. This file is in the database instance's [run directory \[Page 126\]](#).



Backup ID

The database system automatically gives each backup a backup ID to allow them to be identified. This ID uniquely identifies the backups done since the creation of the database instance. If external backup tools are used for the backup, the backup is given an [external backup ID \[Page 119\]](#).

The backup ID consists of the **type of backup** (complete or incremental [data backup \[Page 111\]](#) or [log backup \[Page 121\]](#)) and a **sequential number**.

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

After the backup, the backup ID is written to the log `dbm.knl`. You can display all the contents of this file in the [backup history \[Page 109\]](#).

In a restore, the ID 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 ID on the sticker of the tape or cassette at the end of the backup.

Formal Description of the Backup IDs

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

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

```
<sequence_no> ::= <number>
```



Backup Medium

A backup medium is assigned to every backup you carry out. Backup media include files, tapes, autoloaders, and pipes.

Backup media can be defined before or during the backup process and they are defined separately for each individual [database instance \[Page 113\]](#). You can define [single backup media \[Page 128\]](#) or combine media to form a group of [parallel backup media \[Page 124\]](#).

Recommended Backup Media

- The recommended backup media for [data backups \[Page 111\]](#) are tape and autoloader. Data may also be backed up to pipes or files.
- [Log backups \[Page 121\]](#) are always to files (version files).

Using files or version files as the backup medium only makes sense if they are written to tape or autoloader afterwards.

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



It is best always to copy the files to tape or autoloader. 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 backup media should be used.

Naming

When names are assigned, we differentiate between [standard backup media \[Page 123\]](#) and [backup media for external backup tools \[Page 124\]](#).



Cache

Read and write operations to the [devspaces \[Page 118\]](#) of a [database instance \[Page 113\]](#) are buffered in order to save on disk accesses.

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

The database system recognizes the following caches:

- [Catalog Caches \[Page 18\]](#)
- [Converter Caches \[Page 18\]](#)
- [Data Caches \[Page 19\]](#)
- [File Directory Cache \[Page 19\]](#)
- [Free Block Management \(FBM\) Cache \[Page 19\]](#)
- [Log Cache \[Page 19\]](#)



Catalog

See [database catalog \[Page 112\]](#)



Checkpoint

Unlike the [savepoint \[Page 127\]](#) the checkpoint is only executed on request and only when all the transactions have been completed. A checkpoint is implicitly requested during database backups or during a shutdown of the database instance.

Data Backup with Checkpoint

An example of when a checkpoint is explicitly requested is [data backup with checkpoint \[Page 39\]](#).

When you start a complete data backup with checkpoint, all the transactions currently running are completed and no further write transactions are permitted. All the amended pages held in the [data cache \[Page 19\]](#) are then written from the cache to disk to ensure that the status of the [data devspaces \[Page 112\]](#) 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 importing of [log backups \[Page 121\]](#).



Data Backup

In a complete data backup, all [data devspaces \[Page 112\]](#) are backed up to the [backup medium \[Page 110\]](#) you specified. In an incremental data backup it is only the pages which have been updated that are backed up.

You can carry out the following types of backups:

- Complete [data backup with checkpoint \[Page 39\]](#)
- Complete [data backup without checkpoint \[Page 39\]](#)
- Incremental data backup with checkpoint
- Incremental data backup without checkpoint

Data Backups in WARM Mode

In **WARM** mode, you can either perform complete or incremental data backups with or without checkpoint.

When you back up in warm mode, remember that the state of the [database instance \[Page 113\]](#) that is saved is always the state when the backup operation was started.

When you carry out a data backup with checkpoint, you get a [consistent data backup \[Page 39\]](#).

Data Backups in COLD Mode

Complete and incremental data backups (without Checkpoint) can also be carried out when the database is not running, that is in COLD mode.

Whether you get a [consistent data backup \[Page 39\]](#) in this case depends on how the database instance was shut down. If a [checkpoint \[Page 111\]](#) was requested on shutdown, and this could be performed correctly, you can generate a consistent data backup.

See also:

[Backup Strategy \[Page 37\]](#)

[Saving Data Backups \[Page 40\]](#)



If data is recovered, you may need the relevant [log backups \[Page 121\]](#) as well as the data backup.



Data Devspace

User data (tables, indexes) and the database catalog are stored in the data [devspaces \[Page 118\]](#) of a [database instance \[Page 113\]](#). 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, a database-internal striping algorithm distributes the data belonging to a table evenly across all the data devspaces.



An assignment of tables to data devspaces is neither possible nor necessary.

You can configure one or more data devspaces when you install a new database instance. The disk storage space defined by all the data devspaces is the total size of the database. You can add new data devspaces while the database is in operation as long as the total size of the database does not then exceed the maximum size specified in the database parameter [MAXDATAPAGES \[Page 74\]](#).



Database Administrator

A database administrator is a [database user \[Page 117\]](#) with database user class [DBA \[Page 117\]](#).



Database Catalog

The database catalog of a [database instance \[Page 113\]](#) consists of **metadata** in which definitions of database objects such as Basis tables, view tables, synonyms, value ranges, indexes, users, and user groups are stored.

The database catalog is made up of a number of sections. One section consists of the information about the installation of the database system and the metadata with the definitions of users and user groups. This section is not assigned to any user or user group.

For each user or user group, the catalog contains a section in which the metadata of the objects generated by this user or user group is stored. This includes metadata on Basis tables, view tables, and so on.

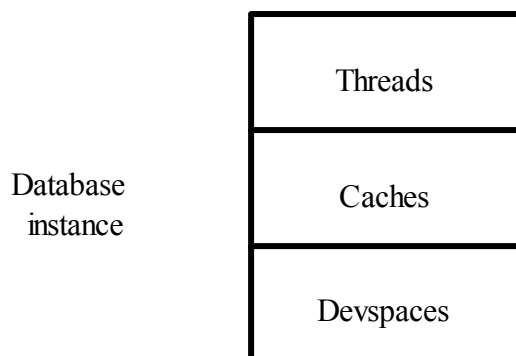
A user can only access the metadata of another user or another user group if authorized to do so.



Database Instance

The SAP DB database can be installed and run on a computer in one mode (database instance) or several modes (database instances) ([Database Instance Type \[Page 113\]](#), see also: [SAP DB Versions and Database Instance Types \[Page 22\]](#)).

Each database instance consists of [threads \[Page 129\]](#), main memory structures ([caches \[Page 110\]](#)) and disk-based storage of the data on devspaces.



The following disk-based [devspaces \[Page 118\]](#) are available in every database instance for the physical storage of data:

- [System devspace \[Page 129\]](#)
- One or more [data devspaces \[Page 112\]](#)
- One or more [log devspaces \[Page 122\]](#)

Each database instance differentiates between the following areas for the logical storage of data:

- [Database catalog \[Page 112\]](#)
- [User data \[Page 131\]](#)



Database Instance Type

The SAP DB database system supports different application areas. The SAP DB [database instance \[Page 113\]](#) has different characteristics depending on the application area. The following database instance types exist:

- [SAP DB OLTP \[Page 127\]](#)
- [liveCache \[Page 120\]](#)
- [SAP DB Document Server \[Page 126\]](#)
- [SAP DB OLAP \[Page 127\]](#)
- [SAP DB E-Catalog \[Page 126\]](#)

See also:

[SAP DB Versions and Database Instance Types \[Page 22\]](#)



Database Manager

The **Database Manager** is a tool for database administration.

The tasks of the Database Manager comprise creating, controlling, and monitoring [database instances \[Page 113\]](#) on the local computer or on remote computers. You can use the Database Manager to carry out backups and, if necessary, recoveries.

Architecture

The Database Manager consists of a server part and a client part.

Server/Client for the Database Manager

Server	Client
DBM Server [Page 118]	DBMGUI [Page 115] DBMCLI [Page 114] Web DBM [Page 131]

A script interface is available. The same functionality is offered for the Database Manager regardless of which client you use.

See also:

[Architecture of the Database Manager \[Page 43\]](#)

[Architecture of the SAP DB Web Tools \[Page 46\]](#)



Database Manager CLI

The [Database Manager \[Page 114\]](#) has a command-line oriented client, the Database Manager CLI (DBMCLI). The Database Manager CLI is operating-system-independent.

You can use the Database Manager CLI to carry out all Database Manager actions. You can also use the Database Manager CLI to schedule these actions in the background. You should use this option to automate Database Manager actions that have to be carried out regularly.

If you do not have a Windows operating system, you can only use Database Manager clients Database Manager CLI and [Web DBM \[Page 131\]](#) to manage database instances.

Call

```
dbmcli [<options>] [<command>]
```

You can transfer [options \[Page 50\]](#) and a maximum of one [DBM Server command \[Page 51\]](#) in a command line to the Database Manager CLI.

For a link to be established with a database instance on the local computer, you must enter at least the name of the database instance (option `-d <database_name>`) when calling the Database Manager CLI. If the required database instance is on a remote computer, you must also enter this computer name (option `-n <server_node>`).

You can open an interactive Database Manager CLI session if you do not enter any DBM Server commands other than the required options. You can then enter the required DBM server commands interactively.

You can write the required DBM Server commands to a separate file `<file_name>`. In this case, when you call the Database Manager CLI, you can enter option `-i <file_name>` in addition to the required options.

See also:

[Database Manager CLI: SAP DB 7.3 \[Extern\]](#) → [Functions of the Database Manager CLI \[Extern\]](#)



Database Manager GUI

The [Database Manager \[Page 114\]](#) has a user-friendly graphical user interface, the Database Manager GUI (DBMGUI). If you want to use the Database Manager to monitor several SAP DB database instances, which may be on different computers, you should use the Database Manager GUI.

The Database Manager GUI can only be used on Windows operating systems. If you want to use the functionality of the Database Manager on other operating system platforms, you must use the [Database Manager CLI \[Page 114\]](#) or [Web DBM \[Page 131\]](#).

Call

You have the following options for calling the Database Manager GUI:

- Choose *Start* → *Programs* → *SAP DB* → *Database Manager*. Log on to the desired database instance.
- You can start the Database Manager GUI from the command line. In this case, you can transfer [options \[Page 49\]](#) to the DBMGUI program.

See also:

[Database Manager GUI: SAP DB 7.3 \[Extern\]](#) → [How the DBMGUI Works \[Extern\]](#)



Database Manager Operator (DBM Operator)

Users working with the database management tool [Database Manager \[Page 114\]](#) are known as Database Manager Operators. The [SAP DB user class \[Page 127\]](#) is the DBM operator.

Depending on what [user authorizations \[Page 25\]](#) the DBM operator has been given, a DBM operator is able to perform all kinds of Database Manager functions.

You create the **first** DBM operator when you register a [database instance \[Page 113\]](#). When you register a new database instance, the system asks you to create a DBM operator by entering a user name and password for the DBM operator. The DBM operator's password can be changed at a later date.

- This DBM operator is then responsible for managing and monitoring the database system and for running backups. The DBM operator is also authorized to perform all Database Manager functions, regardless of what operating mode the database instance is in.
- The DBM operator is also authorized to create additional DBM operators, and assign these all or some of their authorizations.
- The DBM operator can log on to the Database Manager more than once, which means the DBM operator can, for example, query operating parameters while functions that take a long time are still running.



To access a new database instance after registering it, this database instance must be registered in the Database Manager under the name and password of the DBM operator.

DBM operators are **not** [database users \[Page 117\]](#). You need to create database users in order to work on a database instance.



Database Parameters

To initialize the database parameters, 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 database parameters originally set to suit any specific requirements you may have.

Alternatively you can use the configuration of a [database instance \[Page 113\]](#) already present on the computer or the configuration from a [data backup \[Page 111\]](#). Even after this you can still adjust the database parameters to suit your own requirements.

General Database Parameters

DATA_CACHE [Page 70]	INSTANCE_TYPE [Page 70]	KERNELVERSION [Page 72]
LOG_MODE [Page 72]	LOG_SEGMENT_SIZE [Page 72]	MAXARCHIVELOGS [Page 73]
MAXBACKUPDEVS [Page 73]	MAXCPU [Page 73]	MAXDATADEVSPACES [Page 74]
MAXDATAPAGES [Page 74]	MAXLOCKS [Page 74]	MAXUSERTASKS [Page 74]
RESERVED_REDO_SIZE [Page 76]	RESTART_SHUTDOWN [Page 76]	RUNDIRECTORY [Page 77]

Special Database Parameters

BACKUP_BLOCK_CNT [Page 69]	CAT_CACHE_SUPPLY [Page 69]	CONVERTER_CACHE [Page 69]
DATE_TIME_FORMAT [Page 70]	DEADLOCK_DETECTION [Page 70]	DEFAULT_CODE [Page 70]
JOIN_MAXTAB_LEVEL4 [Page 71]	JOIN_MAXTAB_LEVEL4 [Page 70]	JOIN_SEARCH_LEVEL [Page 71]
KERNELDIAGSIZE [Page 71]	LOG_BACKUP_TO_PIPE [Page 72]	LOG_IO_QUEUE [Page 72]
LRU_FOR_SCAN [Page 73]	MAXRGN_REQUEST [Page 74]	MAXSERVERTASKS [Page 74]
MP_RGN_LOOP [Page 75]	OPTIM_BUILD_RESLT [Page 75]	OPTIM_FETCH_RESLT [Page 75]
OPTIM_KEY_INV_RATE [Page 75]	OPTIM_MAX_MERGE [Page 75]	OPTIM_ORDERBY_IDX [Page 76]
OPTIM_OR_DISTINCT [Page 76]	REQUEST_TIMEOUT [Page 76]	SEQUENCE_CACHE [Page 77]
SESSION_TIMEOUT [Page 77]	UTILITY_PROT_SIZE [Page 77]	_DATA_CACHE_RGNS [Page 78]
_EVENT_ALIVE_CYCLE [Page 78]	_MAXEVENTS [Page 78]	_MAX_MESSAGE_FILES [Page 78]
_ROW_RGNS [Page 78]	_TAB_RGNS [Page 79]	_TRANS_RGNS [Page 79]
_TREE_RGNS [Page 79]	_UNICODE [Page 79]	



Database Session

When you create a user, a password is assigned to it. In order to work with a [database instance \[Page 113\]](#), users must enter a user name and password that the database system recognizes.

When a user enters a correct user name and password, they are granted access to the database instance. This opens a database session and automatically calls up the initial transaction.

You can only work with the database instance within a database session. Users must end each session explicitly.

If the user does not belong to a user group, the "Current User" is the user name they entered to gain access to the database instance. If, on the other hand, the user does belong to a user group, the "Current User" is name of the user group.

See also:

[User Concept \[Page 24\]](#)



Database System Administrator (SYSDBA)

The database system administrator ([database user class \[Page 27\]](#) SYSDBA) is the first [database user \[Page 117\]](#) that is created when a new [database instance \[Page 113\]](#) is registered. Enter a user name and password for this user.

Each database instance has one single SYSDBA user.

The SYSDBA user's password can be changed once the database registration has finished.

The SYSDBA user is very important, especially when database instances are installed. The SYSDBA user is responsible for setting up the system and for creating other database users. The SYSDBA user is the owner of system tables. When system tables are uploaded, the upload tool logs on to the database instance as SYSDBA.

The SYSDBA is able to define data and database procedures. The SYSDBA can also grant other users privileges for these database objects.

Furthermore, the SYSDBA has all [DBM operator \[Page 115\]](#) authorizations and is able to carry out all Database Manager functions.



Database User

Database users log on to a [database instance \[Page 113\]](#) and work with database objects such as tables, views, and indexes (the [SAP DB user class \[Page 127\]](#) is database user).

SAP DB differentiates between various [database user classes \[Page 27\]](#).

Database users can be grouped into [user groups \[Page 28\]](#).



DBA/DOMAIN

DBA Users (Database Administrators)

DBA users ([database user class \[Page 27\]](#): DBA) are [database users \[Page 117\]](#). They are created by the [SYSDBA \[Page 117\]](#).

A DBA user is authorized to create [RESOURCE \[Page 126\]](#) and [STANDARD users \[Page 28\]](#). The DBA user can also define data and database procedures and grant other users all or some DBA user privileges for these database objects.

A DBA user can group users with identical access rights into [user groups \[Page 28\]](#).

DOMAIN User

The DOMAIN user is a special database user belonging to the DBA database user class. The DOMAIN user owns [database catalog \[Page 112\]](#) system tables.

Just like the SYSDBA user, the DOMAIN user is created when a [database instance \[Page 113\]](#) is registered. The DOMAIN user is created by the system under the pre-defined name DOMAIN. The password is the same as that of the SYSDBA user that was created earlier.

The DOMAIN user's password can be changed at a later date.



DBM Server

The Database Manager Server (DBM Server) is the server part of the [Database Manager \[Page 114\]](#). It is installed by the server installation on the database server.

The DBM Server creates the connection to the [database instance \[Page 113\]](#) and can access its environment using operating system resources.

Client applications, such as the [Database Manager GUI \[Page 115\]](#), the [Database Manager CLI \[Page 114\]](#), or Web DBM, the program integrated into the [Web Server \[Page 133\]](#) create a connection to the DBM Server and exchange data with the DBM Server using a Request-Response mechanism.

See also:

[Architecture of the Database Manager \[Page 43\]](#)



DBMCLI

See [Database Manager CLI \[Page 114\]](#)



DBMGUI

See [Database Manager GUI \[Page 115\]](#)



Devspace

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

The drives used should have identical performance parameters (specifically access speeds) to ensure even distribution of data on the devspaces.

The database system has [system devspaces \[Page 129\]](#), [data devspaces \[Page 112\]](#) and [log devspaces \[Page 122\]](#).

Each [database instance \[Page 113\]](#) 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.

If necessary, you can add data or log devspaces to a database instance while the database is running. Paths for data and log devspaces can be changed.



External Backup ID

Every backup that is created on a [backup medium \[Page 110\]](#) using an external backup tool has a unique external backup ID. This ID is generated by the [DBM Server \[Page 118\]](#) or the backup tool during the backup. If a backup was created using a media group, every part of this backup has its own external backup ID.

The external backup ID is written to logs `dbm.knl` ([backup history \[Page 109\]](#)) and `dbm.ebf` during backup.

The external backup ID is used during a restore to request the desired backup. The external backup ID is displayed before the restore operation, and can be corrected if necessary. To do this, the [Database Manager \[Page 114\]](#) transfers the external backup ID to the DBM Server as a command parameter, and the DBM server forwards this to the external backup tool.



External Backup Medium

See [Name of an external backup mediums \[Page 124\]](#)



External Backup Tool

A detailed description of the use of external backup tools can be found in the following documentation: [External Backup Tools: SAP DB 7.2 and 7.3 \[Extern\]](#)



Instance Type

See [Database instance type \[Page 113\]](#)



Interactive Log Backup

By using the interactive [log backup \[Page 121\]](#) you can back up all the pages of the [log area \[Page 121\]](#) that have been saved in the log since the last log backup. A version file is also created for the last log segment, which may not have been filled.

Prerequisites

A full [data backup \[Page 111\]](#) of the current database instance has been created.



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

While an interactive log backup is running, no further backups can be performed.

See also:

[Backup Strategy \[Page 37\]](#)

[Saving Log Backups \[Page 41\]](#)



Kernel

The [threads \[Page 129\]](#) of a [database instance \[Page 113\]](#) are often referred to as the kernel.



Language Support (MapChar Sets)

A MapChar set is a character set that maps language-specific characters to the ASCII code or EBCDIC code. This character set is only of any significance for internal system functions.

MapChar sets were introduced to allow letters to be converted. MapChar sets enable country-specific letters to be mapped to one or two non-country-specific letters (for example, representation of "ü" as "ue").

When the database instance is configured, a conversion of these country-specific letters is implicitly allowed and stored under the name DEFAULTMAP. The DEFAULTMAP conversion definition can be changed.

Further MapChar sets can also be defined with the Database Manager.

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

With the Database Manager, you can create, display, change, or delete MapChar sets (*Database Manager GUI: SAP DB 7.3 → Creating a New Database Instance → [Defining MapChar Sets \[Extern\]](#)/Displaying, Changing and Deleting MapChar Sets [Extern]*).

See also:

Reference Manual: SAP DB 7.2 and 7.3, Sections [ASCII Code \[Extern\]](#), [EBCDIC Code \[Extern\]](#), [MAPCHAR\(x,n,i\) \[Extern\]](#)



liveCache

liveCache is a [database instance type \[Page 113\]](#) of the SAP DB database system.

In Supply Chain Management, large volumes of data must be permanently available and changeable. For this reason, an addition has been made to the [SAP DB OLTP \[Page 127\]](#) relational database system to enable actual data structures and data flows (such as networks and relationships) to be mapped more easily and effectively. The product is called liveCache. The liveCache is object-oriented, and in contrast to SAP DB OLTP, stores its data in the main memory of the database system.



Lock

The database system differentiates between SHARE locks and exclusive locks:

- **SHARE locks:** prevent the locked table or table line being changed by other users without excluding other readers.
- **Exclusive locks:** prevent the locked object being changed by and read by other users. However, the user who locked the object can make changes.

Tables and table lines within a [transaction \[Page 130\]](#) are locked by means of a lock operation mode, which can be defined during logon to the database system (CONNECT statement) or can be set via the LOCK option.



Log Area

The log area defined for the database system can cover several [log devspaces \[Page 122\]](#) and is divided into log segments. You configure the size of the individual log segment with the database parameter [LOG SEGMENT SIZE \[Page 72\]](#).



Log Backup

In a log backup, the contents of the [log area \[Page 121\]](#) are copied to version files. A version file is generated for each log segment.

The space originally occupied by the log segments freed up again after the log backup.

The names of the version files that are generated consist of the version file name entered during definition of the [backup media \[Page 110\]](#) and a sequence number assigned by the database system when the log segments are saved.

Need for Log Backup

- [Consistent data backup \[Page 39\]](#) (for example, a complete [data backup with checkpoint \[Page 39\]](#) in WARM mode is consistent in itself.)
If data is **recovered**, a consistent data backup is not sufficient to recover the current state of the [database instance \[Page 113\]](#) up to a certain point in time (for example, the time immediately before the disk failure occurred).
In order to recover the current state, the data backup and log entries that were written after the data backup must be restored to the database system.
- Inconsistent data backup (for example, a complete [data backup without checkpoint \[Page 39\]](#) in WARM mode is not consistent in itself.)
If data is **recovered**, you need the relevant log backups for the inconsistent data backup in order to restore a consistent database state.
However, the inconsistent data backup and related log backups are not sufficient to recover the current state of the database instance up to a certain point in time (for example, the time immediately before the disk failure occurred).
In order to recover the current state, the data backup, the related log backups, and the log entries that were written after the data backup must be restored to the database system.
- If insufficient storage space in the log area means that new log entries cannot be written, a database standstill results. For this reason, it is necessary to back up log entries regularly.

See also:

[Backup Strategy \[Page 37\]](#)

[Saving Log Backups \[Page 41\]](#)



Log Devspace

All modifications to the database contents are logged in the [devspace \[Page 118\]](#) of a [database instance \[Page 113\]](#) to ensure that the database can be restored if a media device fails.

The whole [log area \[Page 121\]](#) may comprise several log devspaces. When installing a new database instance you can configure the number of required log devspaces, and you can add new log devspaces during the running of the database.

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 \[Page 72\]](#) to DUAL).

During log backups, the content of the log area is copied to a file and the space it originally occupied is released. The backup files are numbered in sequence by the system numbers and are therefore also defined as version files.



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



Log Mode

Definition

You can specify how log entries are backed up using the database parameter [LOG_MODE \[Page 72\]](#):

- SINGLE: Log entries are saved to a single [log devspace \[Page 122\]](#)
- DUAL: Log entries are saved to two (mirrored) log devspaces simultaneously
- DEMO: Log entries are not saved, but are overwritten by new entries

SINGLE

You only configure one log devspace. This is useful if you only have one disk for the [log area \[Page 121\]](#).

The log devspace should be backed up periodically.

If a disk fault occurs whereby the contents of the log devspace are destroyed, the [data devspace \[Page 112\]](#) is no longer consistent with the transaction. You can, however, restore the [database instance \[Page 113\]](#) using a complete data backup and, if necessary, additional log backups. You cannot restore a log area that has not been backed up, nor any transactions that are incomplete.

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

DUAL

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

The advantage of this is that if a log devspace fails, it does not interrupt database operation, and once the defective log devspace has been repaired, it can be restored while the database is in operation.

DEMO

If you set the database parameter LOG_MODE to DEMO, the system writes and overwrites its log entries. As soon as the log area is full, the database system overwrites the previous log entries. This means that log entries are lost after a certain amount of time.

For this reason, you can only restore a database instance for which the LOG_MODE database parameter was set to DEMO using a consistent data backup.

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.



We recommend you always mirror the log devspaces. You can do this by either using the database software and setting the database parameter LOG_MODE to DUAL, or, if available, using hardware-based mirroring.

If your system allows hardware-based mirroring (RAID-1 systems), we advise you use it.



Multiprocessor Configuration

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

The aim here is to allow the maximum possible number of [database sessions \[Page 117\]](#) to be supported by the minimum possible number of operating system threads.

The configuration of the database system controls the degree of external/internal tasking via the two parameters [MAXUSERTASKS \[Page 74\]](#) and [MAXCPU \[Page 73\]](#).



On a computer, 4 processors are available for the [database instance \[Page 113\]](#). No more than 800 database sessions should be running simultaneously.

In this case, set MAXCPU to 4 and MAXUSERTASKS to 800.

The database instance can then utilize the four processors by establishing four operating system threads, each of which performs an internal tasking for up to 200 users.



Name of a Standard Backup Medium

The name of a standard [backup medium \[Page 110\]](#) comprises the path details and properties of the tape device/tapes, autoloader of files under a name that can be selected freely and which should relate to the practical situation.

See also:

[Name of external backup medium \[Page 124\]](#)



Name of External Backup Medium

If you are using an external backup tool, there are naming conventions that you should follow for the [backup media \[Page 110\]](#).

The following character strings at the beginning of a backup media name determine that external backup tools are used for backups to this medium or restores from this medium:

Character String	Backup Tool to Be Used
ADSM	ADSM/TSM (IBM/Tivoli)
BACK	Backint for Oracle Backint for SAP DB
NSR	NetWorker (Legato)

If the Database Manager is not to address any of the backup tools listed above, use backup media names that do not begin with any of the above reserved character strings.

See also:

[Name of standard backup medium \[Page 123\]](#)



Parallel Backup Media

You can carry out a parallel backup to several [backup media \[Page 110\]](#) and recover data from parallel backup media. For this purpose you define a group of parallel media that is identified by a single, grouped media name.

Parallel backup media are simultaneously written or read by the database instance. This increases data throughput - and thus the speed of backup or restore.



Redo Area

The redo area is a storage area reserved for log entries in the [data devspace \[Page 112\]](#).

When log entries are being imported, 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 or not it is worth recovering transactions because they have a rollback instruction at the end. Secondly, it allows log entries which do have to be imported to be dealt with simultaneously.

The redo area is used for restarting the [database instance \[Page 113\]](#). While the database is running it is not available for permanent data objects.



Replication Manager

The Replication Manager provides developers and database administrators with a tool for loading data from external files into SAP DB databases and to unload data from SAP DB databases into external databases.

In addition to processing special load and unload commands, the Replication Manager can also execute all SQL statements. As a result, you can create command files that contain SQL statements for loading and unloading data. In particular, the DDL function for creating database catalog objects is indispensable in combination with the load command.

Architecture

The Replication Manager consists of a server part and a client part.

Server/Client for the Replication Manager

Server	Client
REPM Server [Page 125]	REPMCLI

A script interface is available. If you want to react to Replication Manager return codes, you must use this script interface.

See also: [Architecture of the Replication Manager \[Page 44\]](#)

Call

repmdi [<options>] -b <command_file>

The REPM Server is familiar with [options \[Page 53\]](#) and [REPM Server commands \[Page 53\]](#). The REPM Server commands must be stored in a command file (<command_file>). When you call the Replication Manager, you must enter this command file using -b <command_file>.

For a link to be established with a [database instance \[Page 113\]](#) on the local computer, you must enter at least the name of the database instance (option -d <database_name>) when calling the Replication Manager.

When creating the connection between the REPM Server and the database instance, the Replication Manager first uses the specified options. Next, it evaluates any commands about session creation that exist in the command file, and executes the commands for loading or unloading data and SQL statements.

See also:

[Replication Manager: SAP DB \[Extern\]](#) → [Current Functions of the Replication Manager \[Extern\]](#)



REPM Server

The Replication Manager Server (REPM Server) is the server part of the [Replication Manager \[Page 124\]](#). The REPM Server must be installed on the server on which the external data (source files or target files) are stored.

The REPM Server creates the connection to the [database instance \[Page 113\]](#) and can access its environment using operating system resources.

The REPM Server can communicate remotely with the database instance and with the client.

See also:

[Architecture of the Replication Manager \[Page 44\]](#)

[Replication Manager: SAP DB \[Extern\]](#)



RESOURCE

[Database users \[Page 117\]](#) belonging to the [database user class \[Page 27\]](#) RESOURCE can be created by the [SYSDBA \[Page 117\]](#) and [DBA users \[Page 117\]](#).

RESOURCE users can define data and database procedures and grant other users privileges for these database objects.



Run Directory

The run directory is created when a new [database instance \[Page 113\]](#) is created by the database system (database parameter [RUNDIRECTORY \[Page 77\]](#)). The run directory is used to store log files in which all actions and errors are logged. All the relative path names relate to the run directory of the database instance.

See also:

[SAP DB Directories \[Page 66\]](#)



SAP DB Document Server

SAP DB Document Server is a [database instance type \[Page 113\]](#) of the SAP DB database system.

In today's information landscape, a large amount of data must be processed that does not have the typical format of a relational database, but is "unstructured" (such as videos, XML documents). SAP DB Document Server was developed on the basis of the [SAP DB OLTP \[Page 127\]](#) relational database system to ensure that as much of this unstructured data as possible can be processed outside the OLTP database. This improves the performance of the SAP DB OLTP database.

One application example is the SAP Content Server.



SAP DB E-Catalog

SAP DB E-Catalog is a [database instance type \[Page 113\]](#) of the SAP DB database system.

In Internet catalog applications, a small number of hits must be determined from a large number of product descriptions. To support this, the TREX search engine has been integrated in the [SAP DB OLTP \[Page 127\]](#) relational database system. With the TREX search engine, product descriptions (long texts) can be indexed, and the search terms looked for (exact search, phrase search, fuzzy search, linguistic search).

One application example is the BugsEye or eMerge product from Requisite.



SAP DB OLAP

SAP DB OLAP is a [database instance type \[Page 113\]](#) of the SAP DB database system.

Online Analytical Processing (OLAP) technologies enable you to perform flexible analyses from a variety of business perspectives. It is based on a multi-dimensional data model that is achieved using relational database tables.

One application example is the Business Warehouse System. In contrast to [SAP DB OLTP \[Page 127\]](#) systems, a Business Warehouse System is configured so that large quantities of historical and operative data can be formatted with acceptable response times.



SAP DB OLTP

SAP DB OLTP is a [database instance type \[Page 113\]](#) of the SAP DB database system. SAP DB is a relational database system that was developed for OLTP (Online Transaction Processing). The database system is optimized to process individual transactions fast in environments with a high number of users and large databases.



SAP DB Tools

You can use the following SAP DB tools:

- [Database Manager \[Page 114\]](#)
- [Replication Manager \[Page 124\]](#)
- [SQL Studio \[Page 54\]](#)



SAP DB User Classes

The SAP DB database system differentiates between two main user classes:

- [Database Manager Operator \(DBM Operator\) \[Page 115\]](#)
- [Database Users \[Page 117\]](#)



Savepoint

The system performs savepoints at regular intervals. It writes all changed pages held in the [data cache \[Page 19\]](#) to disk.

However, because the savepoint is carried out while the database is running, you cannot restart the [database instance \[Page 113\]](#) after the data restoring process until you have also restored the log entries and/or the [log backups \[Page 121\]](#) that were made since the savepoint.



Session

See [Database session \[Page 117\]](#)



Single Backup Medium

You have the option of defining single [backup media \[Page 110\]](#). The database system performs its backup operation to this medium. Where the capacity of the backup medium is too small for the backup that has started, the system asks for a succeeding medium.



SQL Studio

The SQL Studio ([SQL Studio: Introduction \[Page 54\]](#)) has a user-friendly, graphical user interface.

The SQL Studio in the form described here can only be used on Windows operating systems. If you want to use the functionality of the SQL Studio on other operating system platforms, you must use [Web SQL \[Page 132\]](#).

Prerequisite

Check if the database instance is started.

Call

You have the following options for calling the graphical user interface of the SQL Studio:

- Choose *Start* → *Programs* → *SAP DB* → *SQL Studio*. Log on to the desired database instance.
- You can start SQL Studio from the command line. In this case, you can transfer [options \[Page 55\]](#) to the SQLSTO program.

See also:

[SQL Studio: SAP DB 7.3 \[Extern\]](#) → [Starting SQL Studio \[Extern\]](#)



SQL Mode

The SAP DB database system is capable of executing correct database applications and applications that are written in accordance with one of the following definitions:

- INTERNAL (internal database system definition)
- ANSI standard (ANSI X3.135-1992, entry SQL)
- Definition DB2 Version 4
- Definition ORACLE7

The database system is capable of checking new applications to ascertain whether they correspond to one of the above definitions. In particular, this means that no extensions that go beyond the selected definition are regarded as correct. However, support of other SQL modes is restricted with regard to DDL statements.

When you log on to the database system, you can enter one of the above definitions or SQL mode INTERNAL (default value).



System Devspace

Information on restarts and the mapping of logical page numbers to physical page addresses is managed in the system [devspace \[Page 118\]](#) of a [database instance \[Page 113\]](#).

Therefore, the size of the system devspace is directly related to the database size and is determined by the database system.



Task

A task is executed by the [database instance \[Page 113\]](#). An overview of all database tasks can be found in [User kernel thread \(UKT\) \[Page 14\]](#).



Terminal Support (Termchar Sets)

A termchar set is a character set that maps terminal-specific codes to the ASCII code or EBCDIC code.

Within the database system, ASCII code, EBCDIC code and UNICODE code are used. As the ASCII code and EBCDIC code contain characters that have a different hexadecimal display on some terminals, you can use the TERMCHAR SET keywords to define terminal character sets that allow conversion between the terminal display of a character and the code used in the database system during input and output. In the case of CONNECT statements, one of the defined terminal character sets can be selected, and is then used for conversion in the database session. If a terminal character set is not selected, or an unsuitable one is selected, this may mean that characters that are contained in and are to be issued from the database system do not appear correctly on the terminal.

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

With the Database Manager, you can create, display, change, or delete termchar sets (*Database Manager GUI: SAP DB 7.3 → Creating a New Database Instance → [Defining Termchar Sets \[Extern\]](#)/[Displaying, Changing and Deleting Termchar Sets \[Extern\]](#)*).

See also:

Reference Manual: SAP DB 7.2 and 7.3, sections [ASCII Code \[Extern\]](#), [EBCDIC Code \[Extern\]](#), [CONNECT Statement \[Extern\]](#)



Thread

A whole series of operating system threads (often referred to as kernels) belong to a [database instance \[Page 113\]](#).

We differentiate between **UKTs** (user kernel threads) and **special threads**.

The required number of UKTs and of special threads depends on the number of [devspaces \[Page 118\]](#), the hardware configuration, and the database parameters.

- [User kernel thread \(UKT\) \[Page 14\]](#)
- [Special thread \[Page 16\]](#)

- [Operating-system dependent special thread \[Page 17\]](#)



Transaction

A transaction is a series of database operations that form a unit with regard to data backup and synchronization.

Transactions close with COMMIT or ROLLBACK:

- **COMMIT:** all the changes to data made within the transaction are retained.
- **ROLLBACK:** all changes made by the transaction on the database instance are reversed, including those that were closed with the statement SUBTRANS END (subtransactions).

Changes closed with a COMMIT can no longer be reversed with a ROLLBACK.

A new transaction is implicitly opened as a result of both a COMMIT and a ROLLBACK.

See also:

Reference Manual: SAP DB 7.2 and 7.3 → [Transactions \[Extern\]](#).



UNICODE

Data types such as CHAR ASCII and CHAR EBCDIC are mainly suited to English and central European languages. With other character sets, a code attribute is usually used for these data types. This code attribute uses a different presentation code to ASCII and EBCDIC, even for internal storage in the database system. This causes problems if you want to access these database systems using a different character set, or if you want to exchange data between database systems with different character sets.

You can avoid these problems by using internal character coding in accordance with UNICODE. Internally, the UNICODE data is stored in UTF-16/UCS-2 format. In UTF-16/UCS-2 format, all characters are two bytes long.

SAP DB is able to display various presentation codes in UNICODE format (UNICODE code in line with ISO 10646, page 1).

Metadata in UNICODE

The names of the database objects (such as table or column names) can be stored internally in UNICODE and can therefore then be displayed in the required presentation code in the database tools.

User data in UNICODE

SAP DB supports the code attribute UNICODE for the data types **CHAR[ACTER]**, **VARCHAR** and **LONG[VARCHAR]**.

See also:

[Installing a UNICODE-Enabled Database \[Page 80\]](#)

Reference Manual: SAP DB 7.2 and 7.3 → *Concepts* → [Code Attribute \[Extern\]](#)



Users

See [SAP DB user classes \[Page 127\]](#)



User Data

User data of a [database instance \[Page 113\]](#) consists of all the lines of all Basis tables.



Web DBM

The [Database Manager \[Page 114\]](#) has a web-based client, the Web DBM. If you want to enable occasional users to use database administration functions, you should install the Web DBM once in the network. The Web DBM can then be called from all browsers.

If you do not have a Windows operating system, you can only use Database Manager clients [Database Manager CLI \[Page 114\]](#) and Web DBM to monitor database instances.

Web DBM provides you with basically the same functionality as the [Database Manager GUI \[Page 115\]](#), but it is operated as a web-based application.

Differences between Web DBM and DBMGUI

With the Database Manager GUI, you can monitor several database instances simultaneously. With the Web DBM, you can only monitor one database instance at a time. When you log on, you enter the database instance that you want to monitor. If you want to switch to another database instance, you must log off, and log on again entering the new database instance.

It is possible to open several browsers simultaneously. In this way, you can monitor a different database instance in each browser with the Web DBM.

In contrast to the Database Manager GUI, a timeout may occur when working with the Web DBM. This results in the connection to the database instance being broken. If this happens, you must log on again.

Set-up of the Web DBM

All administrative functions are listed on the left-hand side of the window in the Web DBM. The functions are divided up into the same groups as in the Database Manager GUI. The colored symbols (green dots and red squares) indicate whether an administrative function can be used in the database instance in its present state. There is no menu bar or toolbar of the kind found in the DBMGUI.

Instead of a list of database instances registered in the DBMGUI, the Web DBM displays the detailed status of the database instance. The most important values displayed in the status have a link that takes you to the relevant function. The functions for starting and stopping the database instance are at the bottom of the status display.

The work area is located below the status display. The subsequent web page corresponding to the administrative function selected in the left-hand side of the window is displayed here.

The structure of these web page is similar to the corresponding screens in the DBMGUI. The bottom of these pages often contains a toolbar with the available administrative functions. If a new administrative function is selected in the Web DBM, and in a new page displayed in the work area as a result of this, the previous display and status is lost. To return to the previous display, you must reselect the relevant administrative function.

The header line contains a logoff link that enables you to end the current connection with a database instance.



Web SQL

The SQL Studio ([SQL Studio: Introduction \[Page 54\]](#)) has a web-based client, the Web SQL. If you want to enable occasional users to use SQL statements, you should install the Web SQL once in the network. The Web SQL can then be called from all browsers.

If you do not have a Windows operating system, you can only use the SQL Studio client Web SQL to send requests to the database instance.

Web SQL provides you with basically the same functionality as the [SQL Studio \[Page 128\]](#), but it is operated as a web-based application.

Prerequisite

Check if the database instance is started.

Call

Enter the following address in the browser:

http://<web_server>:<port>/websql

See *Web Tools Installation Guide: SAP DB 7.3* → [Using the SAP DB Web Tools \[Extern\]](#)

Logon dialog box

Enter the name of the database computer, the name of the database instance, the user name, and the user password.

Set-up of the Web SQL

Header

The header contains a link for logging off, among other things.

Stored SQL Studio Objects

Web SQL the same data as SQL Studio. The folders and SQL Studio objects stored in SQL Studio are visible in Web SQL. One exception is the SQL Studio 'Local Folder' for storing local SQL Studio objects.

Objects

Direct SQL objects can be read and written. In the case of the QueryByExample, VisualQuery and TableDefinition objects, the underlying SQL statement can be displayed, but these objects cannot be changed. Web SQL can create, delete, move, and rename folders and direct SQL objects itself.

Direct SQL Window

You can enter and execute SQL statements in this window. You can display SQL statements from all SQL Studio objects, and you can also store direct SQL objects. You can store newly created SQL statements.

Additional functions

Before executing each SQL statement, you can set the AutoCommit mode, the SQL mode, and the isolation level. You can display previously executed SQL statements using *Previous* and *Next*. A button is available for clearing the SQL window. You can execute several SQL statements. You must separate these from each other using a line with // or -- .

Result window

If SQL mode INTERNAL was selected for executing SQL statements, you can navigate flexibly through the result set.

If SQL mode INTERNAL was selected for executing SQL statements, you can zoom the column of data type LONG.

If one of the other SQL modes was selected for executing SQL statements, you can only navigate forwards in the result table.

If several SQL statements were executed, you can access the results and the messages relating to these SQL statements with a drop-down list.

Web Server

The Web Server is the client part of the SAP DB Web tools [Web DBM \[Page 131\]](#) and [Web SQL \[Page 132\]](#). The server for the Web DBM tool is the [DBM Server \[Page 118\]](#), the server for the Web SQL tool is the [database instance \[Page 113\]](#).

Possible Web servers are the SAP DB Web Server or well-known Web servers such as Apache or IIS. The SAP DB Web server is installed during the installation of the SAP DB Web tools. Apache or IIS must be installed separately or configured. For more information, see the [Installation Guide Web Tools: SAP DB 7.3 \[Extern\]](#).

Web DBM and Web SQL are operating-system-independent. These tools are installed once, and can then be called from any browser. This enables occasional users as well as frequent users to make use of the Database Manager and SQL Studio functionalities quickly and easily in the network.

See also:

[Architecture of the SAP DB Web Tools \[Page 46\]](#)

X Server

The X Server is the communication instance of SAP DB software.

SAP DB client programs (such as database applications, DBMGUI, REPMCLI, and so on) first address the X Server when taking up a connection to a SAP DB Server program (database instance, DBM Server, REPM Server, and so on).

See also:

[Architecture of the Database Manager \[Page 43\]](#)

[Architecture of the Replication Manager \[Page 44\]](#)

Documentation Overview

You will find the **SAP DB documentation** plus further important information on SAP DB on the SAP DB homepage at <http://www.sapdb.org>.

You will find the SAP DB documentation for **SAP solutions** in the external Help Portal (<http://help.sap.com>). Choose *SAP Web Application Server* → *SAP Web Application Server <release>* → *<Language>* → *mySAP Technology Components* → *SAP DB*.

Basis Knowledge

- *User Manual: SAP DB*
- *Reference Manual: SAP DB 7.2 and 7.3*

- *Optimizer: SAP DB 7.3*
- *Messages: SAP DB 7.2 and 7.3*
- *SAP High Availability*

SAP DB Tools

- *Database Manager GUI: SAP DB 7.3*
- *Database Manager CLI: SAP DB 7.3*
- *External Backup Tools: SAP DB 7.2 and 7.3*
- *SQL Studio: SAP DB 7.3*
- *Replication Manager: SAP DB*

CCMS

The Computing Center Management System (CCMS) with integrated database management for SAP DB is **only available for SAP Systems**.

You will find the CCMS documentation in the external Help Portal (<http://help.sap.com>). Choose *SAP R/3* → *SAP R/3 Release <release>* → *<Language>* → *Basis Components* → *Computing Center Management System* → *Computing Center Management System* → *Database Administration in CCMS* → *SAP DB - DBA in CCMS*.

Installation Guides

- *Installing and Upgrading the Database Server: SAP DB 7.3*
- *Web Tools Installation Guide: SAP DB 7.3*
- *C/C++ Precompiler Installation Guide: SAP DB 7.3*

You will find the Installation Guides, which are **only valid for SAP solutions**, at service.sap.com. Choose *Installation/Upgrade Guides* → *SAP Web Application Server/SAP R/3/SAP Basis*.

- *SAP Basis Installation on UNIX: SAP DB*
- *SAP Basis Installation on Windows NT: SAP DB*
- *R/3 Homogeneous System Copy (database system copy)*

Development

- *Development Environment: SAP DB*
- *Source Text for Tools: SAP DB*

Interfaces

- *User Manual ODBC: SAP DB*
- *User Manual Precompiler: SAP DB*