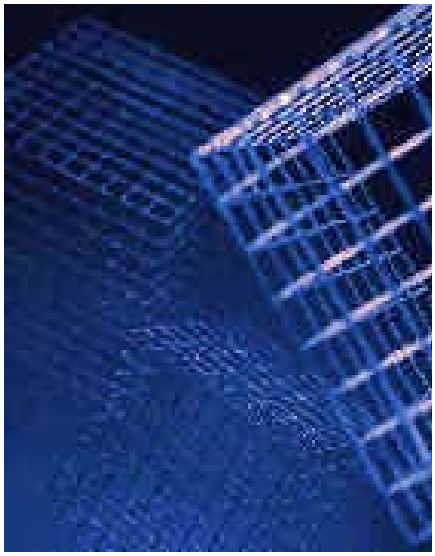
Database Manager CLI:
SAP DB



Version 7.3

Copyright

Icons

Icon	Meaning
	Caution
	Example
	Note
	Recommendation
	Syntax

Typographic Conventions

Type Style	Description
Example text	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

Database Manager CLI: SAP DB 7.3

The database management tool Database Manager can be called as a graphic user interface (Database Manager GUI [Extern]) or as a command-line oriented tool (Database Manager CLI).

The **Database Manager CLI**, Version 7.3, is described here. It can be used on all operating systems that are supported by the SAP DB database system.

The following functions can be executed with the Database Manager CLI:

To work with the Database Manager, a sound knowledge of database administration is required.

For general information on the SAP DB database system, see the User Manual: SAP DB [Extern] or visit the SAP DB Homepage *http://www.sapdb.org*.

Functions of the Database Manager CLI

The Database Manager CLI is an easy-to-use tool that can manage any number of local or remote database instances. These can be managed from the command line. The program is suitable for both interactive and background operation.

Calling up the Database Manager CLI is equivalent to opening a session. Once the commands have been processed, the session with the DBM Server [Page 12] is closed.

The Database Manager CLI is the client program which enables you connect to the DBM Server and exchange data with it. You can also call various functions using the Options when Calling the Database Manager CLI [Page 23] and DBM Server Commands [Page 34].

A request is directed to the DBM Server using a DBM Server command and the corresponding options. This then forwards the reply relating to this command and the

specified options. As well as information on the state of the database instance, the reply contains data in edited form (Reply Format [Page 12]).

DBM Server

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

Client applications like the Database Manager CLI [Page 1] or the Database Manager GUI form a link to the DBM Server and exchange data with the DBM Server using a request-reply mechanism.

Reply Format

Normally communication is in ASCII format. Only in special cases, such as binary data access, do the request and reply also contain binary data.

This ensures that the replies on the client side can be easily read. It also largely avoids dependence on the hardware architecture.

Successful Reply

On the successful reply to a request an OK message is given with the following format:

```
OK[,<description>]<NL>
[<answer>]
```

The character string `OK` is the first token of the reply. There is an optional explanatory text on the same line. The meta character `<NL>` stands for a line feed.

The reply data is provided after the line feed. The structure of the reply data depends on the concrete DBM Server command [Page 34].

Error Message

```
ERR[,<description>]<NL>
[<errcode>,<errdescription><NL>
[<subcode>,<subdescription><NL>]
[<extended description><NL>]]
```

An error always starts with the character string `ERR`. The same line may also contain a description.

> The number of the error `(<errcode>)` and a relevant text `(<errdescription>)` can be included on the next line (**See also:** *Messages: SAP DB 7.2 and 7.3*).

If the system has the error number and relevant text of a subcomponent (for example, runtime environment or operating system), these are transferred as `<subcode>` and `<subdescription>`.

Any additional information comes in the `<extended description>`. Special cases in which the `<extended description>` has a structured format are described with the relevant DBM Server commands.

Internal error of the DBM Server [Page 12]**:**

```
ERR
-24979,ERR_XPNOTFOUND_CN00 : parameter not found
```

Error in runtime environment:

```
ERR
-24994,ERR_RTE_CN00 : runtime environment error
3,cannot access PARAM file [32]
```

Error when analyzing the description file (see also: Configuring Database Instances [Page 78]**):**

```
ERR
-24978,ERR_XPSYNTAX_CN00 : xparam syntax error
16,XPERR_IVSEQ_CN21 : error in IF-ELSIF-ELSE sequence
D:\d628\usr\env\cserv.pcf(60) :     ELSE
```

Database Manager Operators (DBM Operators)

Operators that work with the database administration tool Database Manager are called Database Manager operators (DBM operators). Depending on their DBM operator authorizations [Page 14], they can start and stop databases, perform backups, change database parameters, and so on.

When registering a new database instance, you must define the **first** DBM operator.

During the registration of the new database instance, the system prompts you to define this operator by entering a name and password. You can change the operator's password again later.

The operator defined in this way is responsible for the control and monitoring of the database system and for performing backups. The operator is authorized to run **all** functions of the Database Manager in every operation status. This operator can create **additional** DBM operators and assign all or some of these authorizations to them.

The first operator can log on to the Database Manager more than once, and can therefore, for example, request additional operating parameters during long-running functions.

DBM operators are **not** database users. To be able to work with a database, you must create database users.

See also: *User Manual: SAP DB*

Operator Properties

You can assign operator properties to DBM operators [Page 13]. (See also: Change DBM Operator Data [Page 159].)

Properties that can be assigned to a DBM operator:

USERID, PASSWORD	Name and password for the identification of the operator.
SECONDPASSWORD	A second password used to allow other persons to work temporarily with an operator account, for example for support purposes. Afterwards the second password has to be changed. The original password need not be revealed or changed.
SYSTEMNAME, SYSTEMPASSWORD	Name and password of an operating system user that is to be used by the DBM Server [Page 12]
DBMUSER	The operator is authorized to use the DBM Server.
SQLUSER	The operator is registered in the database.
SQLUSERMODE	User mode in the database
DISABLED	The operator is locked.
SERVERRIGHTS [Page 14]	Operator rights in the DBM Server
GUIRIGHTS	Operator rights in the Database Manager GUI
COMMENT	More detailed description of the operator

DBM Operator Authorizations

A distinction is made between two groups of DBM operator authorizations:

- Authorizations for Using the DBM Server [Page 14] (SERVER_RIGHTS)
- Authorizations for the Database Manager GUI (GUI_RIGHTS).

The authorizations for the DBMGUI have not been defined yet.

Authorizations for Using the DBM Server

A server authorization is an authorization to execute certain DBM Server commands [Page 34].

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

Database Manager server authorizations:

Requesting Status Data [Page 15]	*DBInfoRead*
Executing the LOAD Program [Page 16]	*ExecLoad*
Executing Operating System Commands [Page 16]	*SystemCmd*
File Access (Read Only) [Page 17]	*DBFileRead*
Performing Backups [Page 17]	*Backup*

Installation Management [Page 18]	*InstallMgm*
Loading the System Tables [Page 19]	*LoadSysTab*
Parameter Access (Checked Write) [Page 19]	*ParamCheckWrite*
Parameter Access (Read and Write) [Page 19]	*ParamFull*
Parameter Access (Read Only) [Page 20]	*ParamRead*
Starting the Database Instance [Page 20]	*DBStart*
Stopping Database Instance [Page 21]	*DBStop*
Administration of DBM Operators [Page 21]	*UserMgm*
Restoring Backups [Page 21]	*Recovery*
Access to SQL Session [Page 22]	*AccessSQL*
Access to Utility Session [Page 22]	*AccessUtility*

Requesting Status Data: DBInfoRead

Operators who have been assigned the *DBInfoRead* DBM operator authorization [Page 14] can execute the following DBM Server commands [Page 34]:

Request the Automatic Log Backup Function [Page	`autolog_show`
Requesting the Media Data [Page 110]	`medium_get`
Requesting the Database Kernel Variant [Page 52]	`db_speed`
Database Instance Operating Mode Request [Page 53]	`db_state`
Request External Backup IDs [Page 131]	`backup_ext_ids_get`
Database Instance Information Request [Page 54]	`show`
Request Status Data for the Database Instance [Page 167]	`info`
Activate a Database Event [Page 175]	`event_set`
Media Definition File Change Date Request [Page 108]	`medium_date`
Backup History Change Date Request [Page 126]	`backup_history_date`
Terminating a Database Event Session [Page 176]	`event_release`
Terminating an SQL Session [Page 170]	`sql_release`
Scrolling Through the Contents of the Backup History [Page 126]	`backup_history_listnext`
Scroll Through the External Backup IDs [Page 131]	`backup_ext_ids_listnext`
Scroll in the Information on the Database Instance [Page 55]	`show_next`
Scroll in the Status Information of the Database Instance [Page 171]	`info_next`
Deactivating a Database Event [Page 176]	`event_delete`

Opening an SQL Session [Page 172]	`sql_connect`
Release the Memory Occupied by the External Backup IDs [Page 132]	`backup_ext_ids_forget`
Read Backup History [Page 127]	`backup_history_list`
Reading External Backup IDs [Page 132]	`backup_ext_ids_list`
List of all Defined Media [Page 111]	`medium_getall`
List of Activated Database Events [Page 177]	`event_list`
List the Information on the Database Instance [Page 58]	`show_list`
Open Backup History [Page 129]	`backup_history_open`
Close backup history [Page 130]	`backup_history_close`
Transferring an Existing Media Definition [Page 113]	`medium_migrate`
Wait for a Database Event [Page 178]	`event_wait`

Executing the LOAD program: ExecLoad

Operators who have been assigned the *ExecLoad* DBM operator authorization [Page 14] can execute the following DBM Server commands:

Loading the System Tables [Page 19]	`load_systab`
Load SAP-specific Tables [Page 57]	`load_r3tab`
LOAD Program Execution [Page 60]	`exec_xload`
Starting the PythonLOAD Program [Page 61]	`exec_load`

Executing Operating System Commands: SystemCmd

Operators who have been assigned the *SystemCmd* DBM operator authorization [Page 14] can execute the following DBM Server Commands [Page 34]:

Execute external program or command [Page	`exec_command`
Execute liveCache Initialization Script [Page 39]	`exec_lcinit`

Database File Access (Read Only): DBFileRead

Operators who have been assigned the *DBFileRead* DBM operator authorization [Page 14] can execute the following DBM Server commands [Page 34]:

Scrolling Through a Database File [Page 43]	`file_getnext`
Opening a Database File [Page 45]	`file_getfirst`
Compressing Diagnosis and Database Files [Page 46]	`diag_pack`
Backing Up a Database File [Page 48]	`file_backup`
Restoring a Database File [Page 48]	`file_restore`

Performing Backups: Backup

Operators who have been assigned the *Backup* DBM operator authorization [Page 14] can execute the following DBM Server commands [Page 34]:

Database Instance Current Information Request [Page	`db_restartinfo`
Request External Backup IDs [Page 131]	`backup_ext_ids_get`
Request the Automatic Log Backup Function [Page 115]	`autolog_show`
Requesting the Media Data [Page 110]	`medium_get`
Backup Information Request (OFFLINE) [Page 134]	`medium_label_offline`
Backup media request in WARM or COLD Status [Page 135]	`medium_label`
Requesting the Current Status of a Backup Check [Page 137]	`recover_state_check`
Current Backup Status Request [Page 115]	`backup_state`
Media Definition File Change Date Request [Page 108]	`medium_date`
Backup History Change Date Request [Page 126]	`backup_history_date`
Defining or Changing Backup Media [Page 108]	**medium_put**
Deactivating the Automatic Log Backup [Page 117]	`autolog_off`
Terminating a Service Session [Page 173]	`service_release`
Terminating the Automatic Log Backup [Page 114]	`autolog_cancel`
Terminating an Interrupted Backup [Page 118]	`backup_cancel`
Terminating an Interrupted Restore or Backup Check [Page 145]	`recover_cancel`
Terminating a Utility Session [Page 163]	`util_release`
Scrolling Through the Contents of the Backup History [Page 126]	`backup_history_listnext`

Scroll Through the External Backup IDs [Page 131]	`backup_ext_ids_listnext`
Activating the Automatic Log Backup [Page 118]	`autolog_on`
Opening a Service Session [Page 174]	`service_connect`
Opening a Utility Session [Page 164]	`util_connect`
Continue Parallel Restore or Backup Check [Page 145]	`recover_ignore`
Continue Backup Without Last Known Full Medium [Page 119]	`backup_ignore`
Release the Memory Occupied by the External Backup IDs [Page 132]	`backup_ext_ids_forget`
Reading External Backup IDs [Page 132]	`backup_ext_ids_list`
Read Backup History [Page 127]	`backup_history_list`
List of All Defined Media [Page 111]	`medium_getall`
Deleting a Medium [Page 113]	`medium_delete`
Open Backup History [Page 129]	`backup_history_open`
Backup Check [Page 139]	`recover_check`
Close Backup History [Page 130]	`backup_history_close`
Backing Up to a Succeeding Medium [Page 120]	`backup_replace`
Backing Up the Database Instance [Page 121]	`backup_start`
Transferring an Existing Media Definition [Page 113]	`medium_migrate`
Restore or Check a Backup with Succeeding Medium [Page 153]	`recover_replace`

Installation Management: InstallMgm

Users who have been assigned the *InstallMgm* DBM operator authorization [Page 14] can execute the following DBM Server commands:

Requesting a DBM Server Parameter [Page 77]	`dbm_configget`
Parameter Initialization for a New Database Instance [Page 97]	`param_init`
Parameter File Copy [Page 97]	`param_copy`
Deletion of Current Database Instance [Page 69]	`db_drop`
Parameter File Deletion [Page 99]	`param_rmfile`
Deleting the Registration of a Variant of the Current Database Instance [Page 70]	`db_unreg`
Registering a Variant of the Current Database Instance [Page 75]	`db_reg`
Setting a DBM Server Parameter [Page 78]	`dbm_configset`

Loading the System Tables: LoadSysTab

Operators who have been assigned the *LoadSysTab* DBM operator authorization [Page 14] can execute the following DBM Server commands [Page 34]:

Load the System Tables [Page 56]	`load_systab`
Load SAP-specific Tables [Page 57]	`load_r3tab`

Parameter Access (Checked Write): ParamCheckWrite

Operators that have been assigned the *ParamCheckWrite* DBM operator authorization [Page 14] can execute all of the commands listed under the ParamRead [Page 20] operator authorization. They can **additionally** execute the following DBM Server commands [Page 34]:

Parameter Value Change [Page 92]	`param_put`
Parameter Change Confirmation [Page 92]	`param_commitsession`
Correcting Parameters [Page 98]	`param_putconfirm`
Delete Devspace Parameters [Page 100]	`param_deldevspace`
Check All Parameters [Page 103]	`param_checkall`
Setting Devspace Parameters [Page 104]	`param_adddevspace`
Reset the Parameter File to a Previous Version [Page 106]	`param_restore`

Parameter Access (Read and Write): ParamFull

Operators that have been assigned the *ParamFull* DBM operator authorization [Page 14] can execute all of the commands listed under the ParamRead [Page 20] and ParamCheckWrite [Page 19] operator authorizations. They can **additionally** execute the following DBM Server commands [Page 34]:

Parameter File Value Direct Change [Page 95]	`param_directput`
Adding a Devspace [Page 96]	`db_adddevspace`
Parameter Deletion [Page 100]	`param_directdel`

Parameter Access (Read Only): ParamRead

Operators who have been assigned the *ParamRead* DBM operator authorization [Page 14] can execute the following DBM Server commands [Page 34]:

Terminating a Parameter Session [Page 81]	`param_abortsession`
Requesting All Data for a Parameter [Page 82]	`param_getfull`
Requesting the Data for Multiple Devspace Parameters [Page 87]	`param_getdevsall`
Requesting All Properties of a Parameter [Page 83]	`param_getproperties`
Requesting All Parameters of the Current Parameter File [Page 84]	`param_directgetall`
Request the Data for All Parameters [Page 84]	`param_extgetall`
Requesting Parameter Data [Page 87]	`param_extget`
Requesting the Current Parameter Value [Page 89]	`param_getvalue`
Requesting the Explanatory Text [Page 89]	`param_getexplain`
Requesting the Help Text [Page 90]	`param_gethelp`
Requesting the Parameter Type [Page 90]	`param_gettype`
Requesting the System Default [Page 91]	`param_getdefault`
Request the Data for a Devspace Parameter [Page 86]	`param_getdevspace`
Parameter File Value Request [Page 91]	`param_directget`
Scrolling Through the Parameter History [Page 94]	`param_gethistorynext`
Opening a Parameter Session [Page 95]	`param_startsession`
Parameter File List [Page 99]	`param_versions`
Opening the Parameter History [Page 101]	`param_gethistory`

Starting the Database Instance: DBStart

Operators who have been assigned the *DBStart* DBM operator authorization [Page 14] can execute the following DBM Server commands [Page 34]:

Starting the Database Instance [Page 59]	`db_start`
Database Instance Transfer to WARM State [Page 64]	`db_warm`

Stopping the Database Instance: DBStop

Operators who have been assigned the *DBStop* DBM operator authorization [Page 14] can execute the following DBM Server commands [Page 34]:

Restart Database Instance [Page 59]	`db_restart`
Stopping the Database Instance [Page 62]	`db_stop`
Database Instance Transfer to COLD State [Page 62]	`db_cold`
Taking the Database Instance OFFLINE [Page 63]	`db_offline`
Deleting Runtime Information After a Database Error [Page 58]	`db_clear`
Using the Database Console [Page 55]	`db_cons`

Administration of DBM Operators: UserMgm

Operators who have been assigned the *UserMgm* DBM operator authorization [Page 14] can execute the following DBM Server commands [Page 34]:

Request the Operator Data [Page 158]	`user_get`
Requesting the DBM Operator Authorizations [Page 156]	`user_getrights`
Create a DBM Operator [Page 159]	`user_create`
Change DBM Operator Data [Page 159]	`user_put`
Delete a DBM Operator [Page 160]	`user_delete`
List of DBM Operators [Page 160]	`user_getall`

Restoring Backups: Recovery

Operators that have been assigned the *Recovery* DBM operator authorization [Page 14] can execute all of the commands listed under the Backup [Page 17]. They can **additionally** execute the following DBM Server commands [Page 34]:

Current Recovery Status Request [Page 143]	`recover_state`
Database Instance Information Request [Page 54]	`db_restart_info`
Restoring a Database Instance [Page 147]	`recover_start`

Restoring the Parameter File from a Data Backup [Page 150]	`recover_config`
Recovering a Damaged Devspace [Page 151]	`recover_devspace`

Access to SQL Session: AccessSQL

Operators who have been assigned the *AccessSQL* DBM operator authorization [Page 14] can execute the following DBM Server [Page 34] commands:

Result Set Structure Request [Page 167]	`sql_info`
Update the Optimizer Statistics [Page 169]	`sql_updstat`
Update Optimizer Statistics with the XPU Program [Page 169]	`exec_xpu`
Scrolling in the Result Data [Page 170]	`sql_fetch`
Opening an SQL Session [Page 172]	`sql_connect`
Terminating an SQL Session [Page 170]	`sql_release`
SQL Statement Transfer [Page 172]	`sql_execute`
Recovering a Damaged Index [Page 155]	`sql_recreate_index`

Access to Utility Session: AccessUtility

Operators who have been assigned the *AccessUtility* DBM operator authorization [Page 14] can execute the following DBM Server commands [Page 34]:

New Database Instance Activation [Page 163]	`util_activate`
Terminating a Utility Session [Page 163]	`util_release`
Opening a Utility Session [Page 164]	`util_connect`
Adding a Devspace [Page 96]	`db_adddevspace`
Reading a Physical Database Page [Page 164]	`util_getpage`
Writing a Physical Database Page [Page 165]	`util_putpage`
Transferring a Utility Command [Page 166]	`util_execute`

Calling the Database Manager CLI

Use

Options [Page 23] and a maximum of one DBM Server command [Page 34] with its parameters can be transferred to the program Database Manager CLI.

If you use options with optional parameters, mark the beginning of the DBM Server command with the option –c [Page 29] (Indicator as DBM Server command: -c [Page 29]).
By doing so, you avoid the Database Manager CLI program interpreting the beginning of the DBM Server command as a parameter of the previously specified option.

Prerequisites

Check under *Control Panel* → *Services* that the service ***XServer*** has started *(status*: `started`*)*.

Syntax

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

Reply

See Reply Format [Page 12]

Options when Calling the Database Manager CLI

The following options can be used when calling the Database Manager CLI [Page 23]:

Logging on to the DBM Server [Page 24]	`-u | -U`
Logon to the XUSER Program [Page 25]	`-ux`
Display XUSER Data [Page 25]	`-ul`
Output File [Page 25]	`-o <file_name>`
Output of the Parameters for an XUSER Entry [Page 26]	`-up`
Input Script [Page 26]	`-i <file_name>`
Opening a Service Session [Page 26]	`-uSRV`
Opening an SQL Session [Page 27]	`-uSQL | -USQL`
Opening a Utility Session [Page 28]	`-uUTL | -UUTL`
Database Software Installation Directory [Page 28]	`-R <inst_path>`
Indicator as DBM Server Command [Page 29]	`-c`
Local Operation [Page 30]	`-s`
Deleting XUSER Data [Page 30]	`-ud`

Name of Database Instance [Page 31]	`-d <database_name>`
Name of the Log File [Page 31]	`-t <file_name>`
Name of the Database Server [Page 32]	`-n <server_node>`
Storing XUSER Data [Page 32]	`-us`
Version of DBM Server [Page 33]	`-V`
XUSER Key [Page 34]	`-uk`

```
dbmcli -uUTL samplename,secret -d MK1
```

This calls the Database Manager CLI and establishes a utility session for user **`samplename`**, password **`secret`**, with registered database instance **`MK1`**.

Logging on to the DBM Server: -u

Use

Option when Calling the Database Manager CLI [Page 23]

If you use **`-u`** without specifying **`<userid>,<password>`**, the Database Manager CLI prompts you to enter the operator name and password. In this way you can avoid the operator password is visible in the command line, command history, and the operating system process list.

If you do not specify a **`<user_key>`** when using **`-U`**, the Database Manager CLI uses the data under the **`DBMUSR`** XUSER key.

User details can be stored for various tasks with the help of the program XUSER and used for logging on.

Have a look at the significance of upper and lower case lettering on the meaning of the command.
`-u`: Explicit specification of the user with the **`<userid>`** and **`<password>`**
`-U`: Logon with an operator stored in the XUSER file.

See also:

User Manual: SAP DB

Prerequisites

You are using the option -d [Page 31] at the same time to specify a database instance.

Syntax

```
-u [<userid>,<password>]
```

or logon with XUSER: **`-U [<user_key>]`**

Logon to the XUSER Program: -ux

Use

Option when Calling the Database Manager CLI [Page 23]

You authorize yourself with the operator under the XUSER key `DEFAULT`.

Syntax

```
-ux <default_userid>,<password>
```

Display of XUSER Data: -ul

Use

Option when Calling the Database Manager CLI [Page 23]

You request the operator data stored in the XUSER program.

The system displays a table of the existing XUSER keys with the assigned operator names.

Prerequisites

You are also using option -ux [Page 25] (Logon to XUSER) at the same time.

Syntax

```
-ul
```

Output File: -o

Use

Option when Calling the Database Manager CLI [Page 23]

All details are written to the file specified as **<file_name>**.

Syntax

```
-o <file_name>
```

Output of the Parameters for an XUSER Entry: -up

Use

Option when calling the Database Manager CLI.

You store the connection parameters for an entry in the XUSER program.

See also: *User Manual: SAP DB*

Prerequisites

You are also using the option **-us** (Storage of XUSER data) at the same time.

Syntax

```
-up <param>=<value>; ...;
```

<param>	**SQLMODE | TIMEOUT | CACHELIMIT | ISOLATION | DBLOCALE**
<value>	for **SQLMODE:** **INTERNAL, ANSI, DB2, ORACLE,SAPR3** for **ISOLATION:** **0,1,2,3,10,15,20,30**

Input Script: -i

Use

Option when Calling the Database Manager CLI [Page 23]

If you specify the option **-i**, one or more DBM Server commands [Page 34] will be processed which are listed in the file **<file_name>**.

Syntax

```
-i <file_name>
```

Opening a Service Session: -uSRV

Use

Option when Calling the Database Manager CLI [Page 23]

Certain DBM Server commands [Page 34] require a service session before they can be executed.

If you specify this option this option, the DBM Server command for opening a service session [Page 174] is executed implicitly.

If you exit the Database Manager CLI (DBMCLI) and therefore the DBM Server [Page 12], the service kernel is also stopped.

Prerequisites

You are also using the option -d [Page 31] at the same time to specify a database instance and the option -u [Page 24] to log on to the DBM Server [Page 12].

Syntax

```
-uSRV
```

Opening an SQL Session: -uSQL

Use

Option when Calling the Database Manager CLI [Page 23]

You need an SQL session to execute certain DBM Server commands [Page 34].

If you specify this option this option, the DBM Server command for opening an SQL session [Page 172] is executed implicitly. An SQL session is established with the transferred operator data.

If no operator is specified with this option, the Database Manager CLI will use the data of the current DBM operator [Page 13].

Operator details can be stored for various tasks with the help of the program XUSER and used for logging on.

Have a look at the significance of upper and lower case lettering on the meaning of the command.
-uSQL: Explicit specification of the operator with the **<userid>** and **<password>**
-USQL: Logon with an operator stored in the XUSER file.

See also: *User Manual: SAP DB*

Prerequisites

You are also using the option -d [Page 31] at the same time to specify a database instance and the option -u [Page 24] to log on to the DBM Server [Page 12].

Syntax

```
-uSQL [<userid>,<password>]
```

or logon with XUSER: **-USQL [<user_key>]**

Opening a Utility Session: -uUTL

Use

Option when Calling the Database Manager CLI [Page 23]

A utility session is a prerequisite for the execution of certain DBM Server commands [Page 34] .

If you specify this option this option, the DBM Server command for opening a utility session [Page 164] is executed implicitly. A utility session is established with the transferred operator data.

If no operator is specified with **-uUTL**, the Database Manager CLI uses the data of the first DBM operator [Page 13].

If you do not specify a **<user_key>** when using **-UUTL**, the Database Manager CLI uses the data under the XUSER key **c**.

User details can be stored for various tasks with the help of the program XUSER and used for logging on.

Have a look at the significance of upper and lower case lettering on the meaning of the command.
-uUTL: Explicit specification of the user with the **<userid>** and **<password>**
-UUTL: Logon with an operator stored in the XUSER file.

See also: *User Manual: SAP DB*

Prerequisites

You are also using the option -d [Page 31] at the same time to specify a database instance and the option -u [Page 24] to log on to the DBM Server [Page 12].

Syntax

```
-uUTL <userid>,<password>
```

or logon with XUSER: **-UUTL [<user_key>]**

Database Software Installation Directory: -R

Use

Option when Calling the Database Manager CLI [Page 23]

On servers that contain several versions of the database system software it is necessary to address the correct version of the database. If you use the option -d [Page 31] (Name of the Database Instance) at the same time, option **-R** is ignored. Enter the path of the desired version under **<inst_path>**.

Syntax

```
-R <inst_path>
```

```
dbmcli -R "C:\Program Files\SAP DBTech\V72"
```

When you set up a session there is a check whether the specified version on the server is recorded in the list of registered versions. If it is not, the session connection is refused.

When calling the Database Manager CLI using option **-R**, a specified database version can only be connected if the database installation has been correctly registered [Page 75].

If you specify the option -d [Page 31] and the name of a database instance in addition to the option –R, option -R is ignored. In this case, the Database Manager CLI uses the version of the database software assigned to the specified database instance.

Indicator as DBM Server Command: -c

Use

Option when Calling the Database Manager CLI [Page 23]

Using the option **–c** to logically delimit an Database Manager CLI option with optional parameters and a DBM Server command from each other. The Database Manager CLI program interprets all specifications following **–c** as a DBM Server command.

Syntax

```
-c <DBM_server_command>
```

If you do not specify **-c** between the option with optional parameters and the DBM Server command, the beginning of the DBM Server command is interpreted as an operator name/password combination for the option –uSQL [Page 27]:

```
d:\v74>dbmcli -d a73 -u dbm,dbmp -uSQL sql_execute select *
from tables

ERR

-24988,ERR_SQL: sql error

-4008,Unknown user name/password combination
```

You mark the beginning of the DBM Server command with the option **–c.** The default value is used for the **–uSQL** option.

```
d:\v74>dbmcli -d a73 -u dbm,dbmp -uSQL –c sql_execute select
* from tables

OK

...
```

Local Operation: -s

Use

Option when Calling the Database Manager CLI [Page 23]

If you call the Database Manager CLI with this option, no communication takes places with a DBM Server. In this case, you use the internal DBM Server functions of the Database Manager CLI. In this way, you are also logged on to the operating system and can execute all commands that would otherwise require a logon to the operating system.

Syntax

```
-s
```

Deleting XUSER Data: -ud

Use

Option when Calling the Database Manager CLI [Page 23]

You delete user data from the program XUSER.

You can delete the following user data:

- User data for a particular combination of database instance and server
- User data for an XUSER key

See also: *User Manual: SAP DB*

Deleting User Data for a Particular Combination of Database Instance and Server

Syntax

```
-ud
```

You authorize yourself with the user currently stored in the XUSER program.

```
-d <database_instance> [-n <server_node>] -u
<userid>,<password> -ud
```

or

You authorize yourself with the operator under the XUSER key `DEFAULT`.

```
-d <database_instance> [-n <server_node>] -ux
<defaultuserid>,<password> -ud
```

Deleting User Data Using an XUSER Key

Syntax

-ud

You authorize yourself with the user currently stored in the XUSER program.

-uk <userkey> -u <userid>,<password> -ud

or

You authorize yourself with the operator under the XUSER key `DEFAULT`.

-uk <userkey> -ux <default_userid>,<password> -ud

Name of Database Instance: -d

Use

Option when Calling the Database Manager CLI [Page 23]

The name specified as **<database_name>** applies for the whole session. All database-specific DBM Server commands [Page 34] relate to this database instance.

When you set up a session there is a check whether the specified database instance on the server is recorded in the list of registered database instances. If it is not, the session connection is refused.

Prerequisites

The database instance specified in **<database_name>** exists (Database Instance Registration [Page 71]).

Syntax

-d <database_name>

Name of the Log File: -t

Use

Option when Calling the Database Manager CLI [Page 23]

All commands transferred to the DBM Server [Page 12] and the respective replies are logged in the specified file **<file_name>**. The existing content of the file is retained.

Syntax

```
-t <file_name>
```

Name of the Database Server: -n

Use

Option when opening a session with the Database Manager CLI.

The DBM Server program on the database server specified with **<server_node>** is addressed.

Prerequisites

Program XServer is active on node **<server_node>** (*Status*: `Started`).

See also: *User Manual: SAP DB*

Syntax

```
-n <server_node>
```

Storing XUSER Data: -us

Use

Option when Calling the Database Manager CLI [Page 23]

You store user data in the XUSER program.

You can store this user data as follows:

- As a particular combination of database instance and database server
- As an XUSER key

See also: *User Manual: SAP DB*

Storing User Data for a Particular Combination of Database Instance and Server

Prerequisites

You use the options -d [Page 31] (Name of the Database Instance) and, optionally, -n [Page 32] (Name of the Database Server) at the same time.

You authorize yourself for the XUSER program with the currently stored user or the user under the XUSER key `DEFAULT`.

Syntax

-us <new_userid>,<new_password>

You authorize yourself with the user currently stored in the XUSER program

```
-d <database_instance> [-n <server_node>] [-u
<userid>,<password>] -us <new_userid>,<new_password>
```

or

You authorize yourself with the operator under the XUSER key `DEFAULT`

```
-d <database_instance> [-n <server_node>] [-ux
<default_userid>,<password>] -us <new_userid>,<new_password>
```

Storing as an XUSER key

Prerequisites

You use the option -uk [Page 34] (XUSER key) at the same time.

You authorize yourself for the XUSER program with the currently stored user or the user under the XUSER key `DEFAULT`.

Syntax

```
-us <new_userid>,<new_password>
```

You authorize yourself with the user currently stored in the XUSER program.

```
-uk <userkey> [-u <userid>,<password>] -us
<new_userid>,<new_password>
```

or

You authorize yourself with the operator under the XUSER key `DEFAULT`.

```
-uk <user_key> [-ux <default_userid>,<password>] -us
<new_userid>,<new_password>
```

Version of DBM Server: -V

Use

Option when Calling the Database Manager CLI [Page 23]

You request the version of the DBM Server [Page 12]. The Database Manager CLI displays the version number of the relevant DBM Server.

Syntax

```
-v
```

XUSER Key: -uk

Use

Option when Calling the Database Manager CLI [Page 23]

You specify the XUSER key under which the user data is to be stored in the XUSER program.

Syntax

```
-uk <user_key>
```

DBM Server Commands

Syntax

```
<command_name> [<parameters>]
```

A DBM Server command is always made up of the command name and optional parameters affecting its execution.

In a request the commands are transferred to the DBM Server [Page 12] as an ASCII character string.

A DBM Server command can also contain line feeds.

All commands during a DBM Server session relate to the database instance whose name was specified with the option -d [Page 31] when calling the Database Manager CLI [Page 23].

The Database Manager CLI provides DBM Server commands for the following functions:

Version Information Request [Page 35]

DBM Operator Logons [Page 36]

Executing External Programs or Commands [Page 38]

Execute liveCache Initialization Script [Page 39]

Terminating a DBM Server Session [Page 40]

File Access [Page 40]

Database Trace Functions [Page 49]

Commands for Database Operation [Page 51]

Installation and Registration Management [Page 64]

Configuration of the DBM Server [Page 77]

Configuring Database Instances [Page 78]

Listing the DBM Server Commands [Page 106]

Backing up and Restoring Database Instances [Page 107]

Administration of DBM Operators [Page 156]

Accessing the Database Kernel [Page 161]

Version Information Request

Use

You request the version of the DBM Server [Page 12].

Prerequisites

You do not need to log on to the DBM Server [Page 12] to execute this DBM Server command [Page 34].

You also require no DBM operator authorization [Page 14].

Syntax

dbm_version

Reply

```
OK
VERSION     = <version><NL>
BUILD       = <build_number><NL>
OS          = <os><NL>
INSTROOT    = <inst_path><NL>
LOGON       = <logon_state><NL>
CODE        = <code><NL>
SWAP        = <swap><NL>
UNICODE     = (YES|NO)<NL>
INSTANCE    = (OLTP|LVC|CS|BW|unknown)<NL>
SYSNAME     = <os><NL>
```

Values of the reply lines

<version>	Version of the DBM Server
<build_number>	Identification number of the DBM Server program file
<os>	Operating system of the DBM Server
<rundir>	Installation directory of the version-dependent components
<logon_state>	Display whether a logon to the OS has taken place: `True` – logon taken place `False` – no logon
<code>	Output of the character set used on the DBM Server: ASCII or EBCDIC
<swap>	Output of the swap type used on the DBM Server (internal representation of numeric values): `no` - hihi-hilo-lohi-lolo `full` - lolo-lohi-hilo-hihi `half` - lohi-lolo-hihi-hilo
<os>	Name of the operating system

In the event of errors, see Reply Format [Page 12].

DBM Operator Logons

Use

The execution of DBM Server commands can be tied to three prerequisites:

- Logon to the Database Manager [Page 36]
- DBM Operator Authorization [Page 14]
- Operating System Logon [Page 36]

You can find out which of these prerequisites must be fulfilled for the execution of a particular DBM Server command from the descriptions of the individual DBM Server commands.

Logon to the Database Manager

Use

The DBM operator [Page 13] specified in **<userid>** logs on to the DBM Server [Page 12] and becomes a new active operator.

For local communication, the operator is already logged on to the operating system.

For remote communication, the operator is logged on to the operating system of the remote server if an operating system user (SYSTEMNAME, SYSTEMPASSWORD) has already been created there for the operator (Operating System Logon [Page 36]).

Prerequisites

You do not need to log on to the DBM Server [Page 12] to execute this DBM Server command [Page 34].

Syntax

```
user_logon <userid>,<password>
```

Reply

The system outputs an OK message.

In the event of errors, see Reply Format [Page 12]

Operating System Logon

Use

This logon is needed for executing the following DBM Server commands [Page 34].

Database Instance Registration [Page 71]	`db_create`
Registering a Version of the Database Software [Page	`inst_reg`

76]	
Deletion of Current Database Instance [Page 69]	`inst_unreg`

Prerequisites

You do not need to log on to the DBM Server [Page 12] to log on to the operating system.

Whether or not you require a DBM operator authorization [Page 14], depends on whether you want to access the remote server once or more than once.

Access to the local server

If you are working locally as a DBM operator [Page 13], you also have the operating system authorization.

Access to a remote server

A DBM operator can access a remote server. The command used for the remote access depends on whether the DBM operator wants to access this server once or permanently.

- Accessing a Remote Server Repeatedly [Page 37] (`user_put`)
- Accessing a Remote Server Once [Page 38] (`user_system`)

Accessing a Remote Server Repeatedly

Use

To enable a DBM operator [Page 13] to access a remote server permanently, an operating system user account must be created on this server for the operator.

Prerequisites

You have the DBM operator authorization User Mgm [Page 21].

Procedure

1. Log on to the Database Manager CLI (Logon to the Database Manager [Page 36]).
2. Use the `user_put` command (Change DBM Operator Data [Page 159]) to create an operating system user account (SYSTEMNAME, SYSTEMPASSWORD) for the DBM operator.
   Whenever the DBM operator logs on the Database Manager, the operating system user is also logged on.

Syntax

```
user_put <userid> <SYSTEMNAME>=<value> <SYSTEMPASSWORD>=<value>
```

Reply

The system outputs an OK message.

In the event of errors, see Reply Format [Page 12].

Accessing a Remote Server Once

Use

To access a remote server once, a DBM operator [Page 13] logs on to the operating system using a DBM Server command [Page 34] in session mode [Page 180] or script mode [Page 179].

Prerequisites

No special DBM operator authorization [Page 14] is required for this command.

Procedure

1. Log on to the Database Manager CLI as a DBM operator (Logon to the Database Manager [Page 36]).
2. Log on the operating system using the command **user_system**.

Syntax

```
user_system <userid>,<password>
```

Reply

The system outputs an OK message.

In the event of errors, see Reply Format [Page 12].

Executing External Programs or Commands

Use

You are logged on to the Database Manager CLI and start external programs or commands from this program on the server on which DBM Server is running.

Prerequisites

You have the DBM operator authorization SystemCmd [Page 16].

When you called the Database Manager CLI you logged on with a DBM operator [Page 13] that is registered as an operating system user [Page 36].

Syntax

```
exec_command <command>
```

Successful Reply

The output from the program or command is supplied by the Database Manager CLI.

```
OK<NL>
0,<err_description><NL>
<pgm_code>,<command><NL>
<command_output_lines><NL>
```

Error Message

```
ERR<NL>
<err_code>,<err_description><NL>
<pgm_code>,<command><NL>
<command_output_lines><NL>
```

Values for the individual fields of the reply

<err_code>	Error number of DBM Server. If successful the error number is 0
<err_description>	Description of the error
<pgm_code>	Return value of the executed program
<command>	Command line executed by the DBM Server
<command_output_lines>	Output text of the program
<NL>	Line feed

Execute liveCache Initialization Script

This command is only relevant in connection with SAP applications.

Use

You are logged on to the Database Manager CLI and start the liveCache initialization script stored on the database server from the Database Manager.

You can specify parameters with this DBM Server command that are passed to the liveCache initialization script by the Database Manager. The output from the liveCache initialization script is supplied by the Database Manager CLI.

Prerequisites

You have the DBM operator authorization SystemCmd [Page 16].

Syntax

```
exec_lcinit [<lcinitparams>]
```

Successful Reply

```
OK<NL>
0,<err_description><NL>
<pgmcode>,<command><NL>
<command_output_lines><NL>
```

Error Message

```
ERR<NL>
<err_code>,<err_description><NL>
<pgm_code>,<command><NL>
<command_output_lines><NL>
```

Values for the individual fields of the reply

<err_code>	Error number of DBM Server. If successful the error number is 0
<err_description>	Description of the error
<pgm_code>	Return value of the executed program
<command>	Command line executed by the DBM Server
<command_output_lines>	Output text of the program
<NL>	Line feed

Terminating a DBM Server Session

Use

This command ends the session with the DBM Server [Page 12].

Prerequisites

You do not need to log on to the DBM Server [Page 12] to execute this DBM Server command [Page 34].

Syntax

```
release | bye | exit | quit
```

Reply

The system outputs an OK message.

In the event of errors, see Reply Format [Page 12].

File Access

Use

You can access files that are connected to database instances or versions of the database software using the commands for file access.

Prerequisites

Note the DBM operator authorizations [Page 14] for each of the DBM Server commands [Page 34].

File Access Commands: Overview [Page 41]

File Access Commands: Overview

Requesting the Diagnosis History [Page 41]	**diag_histlist**
Editing a Database File [Page 42]	**file_operation**
Scrolling Through a Database File [Page 43]	**file_getnext**
List Database Files [Page 44]	**file_getlist**
Opening a Database File [Page 45]	**file_getfirst**
Compressing Diagnosis and Database Files [Page 46]	**diag_pack**
Backing Up a Database File [Page 48]	**file_backup**
Restoring a Database File [Page 48]	**file_restore**

Requesting the Diagnosis History

Use

You request the available diagnosis history.

If a database instance was not properly stopped, the system backs up certain database files when restarting the database instance.

These files are required for the diagnosis of problems that occurred previously. The files are backed up by the system in a diagnosis backup and stored in the directory `DIAGHISTORY` in the run directory of the database instance (Database Manager CLI default value). You can define this path using the parameter `DIAG_HISTORY_PATH`.

If you run the command without specifying the option **<YYYYMMDDHHMMSS>**, the system displays a list of all available diagnosis backups. The backups are listed with their time stamps and the paths under which they are stored. This means that you can also access the diagnosis backup using operating system resources.

If you enter the command again, and specify a time stamp for the option **<YYYYMMDDHHMMSS>**, the system displays the IDs `<file_id>` for the files belonging to the diagnosis backup. Using the `<file_id>`, you can display the contents of a file (Opening a Database File [Page 45]).

Prerequisites

You have the DBM operator authorization DBFileRead [Page 17].

Syntax

```
diag_histlist [<YYYYMMDDHHMMSS>]
```

Command without option specified

```
dbmcli -d DB -u dbm,dbmp diag_histlist
```

```
OK
20010706140709
d:\sapdb\usr\wrk\DB\DIAGHISTORY\A74_20010706_14-07-09
20010706162223
d:\sapdb\usr\wrk\DB\DIAGHISTORY\A74_20010706_16-22-23
```

Command with option specified

```
dbmcli -d DB -u dbm,dbmp diag_histlist 20010706162223
```

```
OK
DIAGHIST#20010706_16-22-23\knldiag
DIAGHIST#20010706_16-22-23\knldump
DIAGHIST#20010706_16-22-23\knltrace
DIAGHIST#20010706_16-22-23\rtedump
```

Reply

The result of the request depends on whether you specify the option **<YYYYMMDDHHMMSS>** (time stamp).

After Command Entry Without Option

If you do not specify a time stamp, the system displays a list of all available diagnosis backups.

```
OK<NL>
<YYYYMMDDHHMMSS> <directory><NL>
<YYYYMMDDHHMMSS> <directory><NL>
...
```

After Command Entry With Option

If you specify a time stamp, the system displays a list of the files that belong to the diagnosis backup.

```
OK<NL>
<file_id><NL>
<file_id><NL>
...
```

In the event of errors, see Reply Format [Page 12].

Editing a Database File

Use

You delete part of the content of database files or complete database files.

This command is only permissible for certain database files, such as log files. If you use the command on a file for which it is not permissible, you will receive the error `-24996 ERR_PARAM - wrong parameters.`

You specify the operation that is to be performed with the parameter **OP**.

Prerequisites

You have the DBM operator authorization DBFileRead [Page 17].

Syntax

```
file_operation <file_id> OP=DELETE [DATE=<yyyymmddhhmmss>] |
OP=SHRINK DATE=<yyyymmddhhmmss>|LINE=<n>
```

<file_id>	Database file(s) [Page 181]
OP=DELETE	The file(s) is/are deleted. You can specify that all files that have not been changed since the specified date, **DATE=<yyyymmddhhmmss>**, are to be deleted.
OP=SHRINK	Part of the content of the file are deleted. If you additionally specify the parameter **DATE=<yyyymmddhhmmss>**, all lines that were written before the specified date are deleted. If, instead, you specify the paramtere **LINE=<n>**, the contents of the file are reduced to **<n>** lines. The **<n>** lines that were written most recently are retained.

Reply

The system outputs an OK message.

In the event of errors, see Reply Format [Page 12].

Scrolling Through a Database File

Use

You request output of the rest of the content of an opened database file. Use the **<file_handle>** output by the system when you were opening the database file [Page 45].

Prerequisites

You have opened a database file. Keyword `CONTINUE` in the reply shows that you have not yet read the whole file.

You have the DBM operator authorization DBFileRead [Page 17].

Syntax

```
file_getnext <file_id> <file_handle>
```

Options for <file_id>

see: Database Files [Page 181]

Reply

Output in ASCII Mode

```
OK<NL>
[CONTINUE|END]<NL>
<file_length>:20<data_length>:20<NL>
<data><NL>
<data><NL>
...
```

Output in Binary Mode

```
OK<NL>
[CONTINUE|END]<NL>
<file_length><data_length><NL>
<data><NL>
```

Values for the individual fields of the reply

END	Contents of the file have been completely transferred, file is automatically closed
CONTINUE	File has additional entries that have not been transferred due to the limited size of the reply memory. Call these by entering the command used above again
<file_length>	File length
<data_length>	Data length
<data>	Data value

The total length of file is not determined again and is therefore set to 0.

List Database Files

Use

You request a list of the database files of the current database instance.

You determine the scope of the displayed file list using the parameter **<list_level>**.

Possible values for **<list_level>**:

0	Default value of the Database Manager CLI, display the most important files
1	Display all files

Prerequisites

You used option -d [Page 31] to log on as a user (Logging on to the DBM Server [Page 24]).

You have the DBM operator authorization DBFileRead [Page 17].

Syntax

```
file_getlist <list_level>
```

Only those files are listed that are actually available. The list may therefore vary with the state of the database instance.

Reply

The system outputs an OK message. Then it lists the database files currently available.

```
OK
key_name,mode,size,date,time,comment,file_name
<file_id> (ASCII|BINARY) <file_lengthe> <date> <time> <comment>
<file_name>
<file_id> (ASCII|BINARY) <file_length> <date> <time> <comment>
<file_name>
...
```

Values for the individual fields of the reply

file_id	Key identifying name of a database file
file_length	Size of database file
date	Date last changed
time	Time last changed
comment	Additional information
file_name	File Name

Opening a Database File

Use

You open the database file specified under **<file_id>** and transfer the first block. Using the **<file_id>**, the DBM Server determines the actual name of the file and whether it is a binary or text file.

File access using this command is limited to the database files [Page 181]. For this reason this command does not necessitate logging on to the operating system.

You can request the files that are currently available and their <file_id> with the command **file_getlist** (List Database Files [Page 44]).

Prerequisites

You have the DBM operator authorization DBFileRead [Page 17].

Syntax

```
file_getfirst <file_id>
```

Options for <file_id>

see: Database Files [Page 181]

Reply

Output in ASCII Mode

```
OK<NL>
<file_handle><NL>
[CONTINUE|END]<NL>
<file_length>:20<data_length>:20<NL>
<data><NL>
<data><NL>
...
```

Output in Binary Mode

```
OK<NL>
file_handle><NL>
[CONTINUE|END]<NL>
<file_length><data_length><NL>
<data><NL>
```

Values for the individual fields of the reply

<file_handle>	Numeric value that can be used for subsequent access to this file.
END	The contents of the file have been transferred in full. The file is closed automatically.
CONTINUE	The file contains further entries that were not transferred due to the limited storage available for replies. Interrogate this data by entering the above command or close the file.
<file_length>	File length
<data_length>	Data length
<data>	Data

In ASCII mode the data is read from the file line by line. Each line is extended by a line feed (0x0D,0x0A) in the output area.

On a UNIX server, the line feed only consists of one character. This is why the value initially output for the file length may be smaller than the data length.

In the event of errors, see Reply Format [Page 12].

Compressing Diagnosis and Database Files

Use

To be able to comprehensively diagnose a database problem, Support require a significant number of database instance files.

Using this command, you can collect all of the required files in a compressed archive file.

Default values of the Database Manager CLI for the archive file:

File Name	`diagpack.tgz`
Directory	Run directory of the database instance
`<file_id>`	`DIAGTGZ`

You can define the file name and path yourself using the parameter **`file=<archive>`**.

You can specify which classes of files should be included in the package using the parameter **`<class=class_spec>`**. If you specify multiple classes, separate them using commas.

If you specify the class `hist`, you must additionally specify the time stamp of the diagnosis backup for the parameter **`<time_stamp>`**.

When opening the file, note that it is stored in binary format (Opening a Database File [Page 45]).

Prerequisites

You have the DBM operator authorization DBFileRead [Page 17].

Syntax

`diag_pack [file=<archive>] [class=<class_spec> | <class_spec>,...] [date=<time_stamp>]`

`<archive>`	Name and path of the archive file to be created
`<class_spec>`	Class of files that is to be included in the package The default value of the Database Manager CLI is the class `protocol` Separate multiple class specifications by commas.
`<time_stamp>`	Additional parameter when specifying the class `hist`, Time stamp of the diagnosis backup Determine this by requesting the diagnosis history [Page 41]

Options for <class_spec>

Class	Description
`protocol`	General log files
`backup`	Log files from backups and restores
`config`	Configuration files
`lvc`	Additional log files for a database of instance liveCache

For more information about the contents of database files, see: Database Files [Page 181]

Backing Up a Database File

Use

With this command, you can create a backup copy of the database file specified in **<file_id>**.

Normally, you do not have to create backup copies of internal DBM Server [Page 12] files because the DBM Server creates backups automatically every time it write-accesses configuration files.

Prerequisites

You have the DBM operator authorization DBFileRead [Page 17].

Syntax

```
file_backup <file_id>
```

Options for <file_id>

see: Database Files [Page 181]

Reply

The system outputs an OK message.

In the event of errors, see Reply Format [Page 12].

Restoring a Database File

Use

Use this command to copy the backup copy of the database file specified using **<file_id>** to its original position. If a file already exists, it will be overwritten.

The DBM Server uses this function internally to access the backup copy of a file if was unable to access the file normally.

Normally, you do not have to copy backups of internal DBM Server [Page 12] files to their original position manually because the DBM Server will automatically use the backups if it is unable to access the original file.

Prerequisites

You have the DBM operator authorization DBFileRead [Page 17].

Syntax

```
file_restore <file_id>
```

Options for <file_id>

see: Database Files [Page 181]

Reply

There is an OK message after the command has been executed successfully.

In the event of errors, see Reply Format [Page 12]

Database Trace Functions

Use

Use this function only when told to do so by support. The database trace is not required for normal database operation.

You log selected activities of the database kernel in detail using the database trace. You can activate, deactivate, and evaluate the database trace using the commands described below.

Prerequisites

Note the DBM operator authorizations [Page 14] for each of the DBM Server commands [Page 34].

Database Trace Commands Overview [Page 49]

Database Trace Commands Overview

Requesting Options for the Text Version of the Database Trace [Page 50]	`trace_protopt`
Creating the Text Version of the Database Trace [Page 50]	`trace_prot`

Requesting Options for the Text Version of the

Database Trace

Use

You can specify various options when creating the text version of the database trace. You query which options are permissible for the text version of the current database trace using this command. You receive a list with the names and description of the valid options.

Prerequisites

You have the DBM operator authorization DBInfoRead [Page 15] or DBFileRead. [Page 17]

Syntax

trace_protopt

```
dbmcli -d DB -u dbm,dbm trace_protopt
```

```
OK
name,option
a       Order Interface (AK)
b       Record Interface (BD)
k       Show Message Block (KB)
m       Message Block
e       No Entrypos Output
s       Strategy
t       Time
x       Switch Output (Slow Kernel)
```

Reply

```
OK<NL>
name,option<NL>
<name>,<option><NL>
<name>,<option><NL>
...
```

Creating the Text Version of the Database Trace

Use

Use this command only when told to do so by support. The database trace is not required for normal database operation.

You transfer the binary version of the database trace to the text version. You can use all values and combinations of values for **<option>** options that you received as a reply when Requesting the Options for the Text Version of the Database Trace [Page 50].

The binary version of the database trace has the file name `knltrace` and is stored in the run directory of the database instance (see also: Database Files [Page 181]).

The `<file_id>` (see also: Opening a Database File [Page 45]) for openring this text version is `KNLTRCPRT`.

Prerequisites

You have the DBM operator authorization DBInfoRead [Page 15] or DBFileRead. [Page 17]

You have created the database trace (Activating the Database Trace [Extern]) using the Database Manager GUI or relevant utility command.

Syntax

```
trace_prot <option> | <option>...
```

Create the text version of the database trace for the areas `a - Order Interface and b - Record Interface:`

```
dbmcli -d DB -u dbm,dbmp trace_prot ab
OK
```

Reply

The system outputs an OK message.

In the event of errors, see Reply Format [Page 12].

Functions for Database Operations

Use

DBM Server commands [Page 34] are provided for use in the day-to-day operation of the database instance. You can use these commands to request general information on the status of the database instance and transfer the instance to the required operating status.

Prerequisites

Note the DBM operator authorizations [Page 14] for each of the DBM Server commands [Page 34].

Overview of the Commands for Database Operation [Page 51]

Overview of the Commands for Database Operation

Requesting the Database Kernel Variant [Page 52]	`db_speed`
Database Instance Operating Mode Request [Page 53]	`db_state`
Database Instance Information Request [Page 54]	`show`
Scroll in the Information on the Database Instance	`show_next`

[Page 55]	
Load the System Tables [Page 56]	**load_systab**
Load SAP-Specific Tables [Page 57]	**load_r3tab**
List the Information on the Database Instance [Page 58]	**show_list**
Restart Database Instance [Page 59]	**db_restart**
Starting the Database Instance [Page 59]	**db_start**
Starting the LOAD Program [Page 60]	**exec_xload**
Starting the PythonLOAD Program [Page 61]	**exec_load**
Stopping the Database Instance [Page 62]	**db_stop**
Database Instance Transfer to COLD State [Page 62]	**db_cold**
Taking the Database OFFLINE [Page 63]	**db_offline**
Database Instance Transfer to WARM State [Page 64]	**db_warm**

Requesting the Database Kernel Variant

Use

You request the kernel variant of the current database instance.

Prerequisites

You have the DBM operator authorization DBInfoRead [Page 15].

The database instance is in `WARM` or `COLD` operation status.

Syntax

db_speed

Reply

```
OK<NL>
Speed<NL>
(FAST|QUICK|SLOW|UNKNOWN|)
```

FAST	Fastest database kernel variant, with minimum logging and runtime checks.
QUICK	Variant of the database kernel with various logs and runtime checks.
SLOW	Variant of the database kernel with extensive logs and runtime checks.
UNKNOWN	The kernel variant cannot be determined.

If the database is in the OFFLINE mode, the system answers this command with an error message.

Database Instance Operating Mode Request

Use

You request the operation status of the current database instance.

Prerequisites

You have the DBM operator authorization DBInfoRead [Page 15].

Syntax

db_state

Reply

OK<NL>
State<NL>
(WARM|COLD|OFFLINE|UNKNOWN|STOPPED INCORRECTLY)

Mode	Description
WARM	The database is ready.
COLD	The database has been shutdown. Only special administrative work can be done.
OFFLINE	The database has not been started.
STOPPED INCORRECTLY	With UNIX operating system only: The database stopped after an error.
UNKNOWN	The state cannot be determined.

Database Instance Information Request

Use

You request database instance information, for example about statistics, the mode, or tasks. The **show** command obtains and outputs information from program x_cons.

Prerequisites

You have the DBM operator authorization DBInfoRead [Page 15].

Syntax

show <command>

Possible values for <command>

IO	Input/output activities in regular database operation
AIO	Input/output activities for data backup
STORAGE	Configuration and current status of the various storage areas
TASKS	List of tasks
ACTIVE [DW|SV|US]	Active tasks for [Datawriter-Task|Server-Task|User-Task]
RUNNABLE [DW|SV|US]	Runnable tasks (waiting for CPU)
T_C [DW|SV|US|Tx]	Task-specific information on [Datawriter_Task | Server Task | User Task | Task No.x}
VERSIONS	Current variant of database kernel and runtime environment
REGIONS	Information about the critical regions used in the kernel
STATE	Mode of database instance
RTE	Runtime environment
QUEUES	Queues of runnable tasks (waiting for CPU)
SUSPENDS	Information about suspend states anywhere in the overall system
SLEEP	The CPU load from user kernel threads measured by the database kernel.
THRD_TIMES	Information from the system about the CPU load from user kernel threads (Windows NT operating system only)
ALL	All of the information provided by the database console

Reply

```
OK<NL>
[CONTINUE]<NL>
<info_record><NL>
<info_record><NL>
```

CONTINUE	More result data ready
<info_record>	Information

Using the Database Console

Use

You use the database console to analyze and influence the runtime behavior of the database instance.

As parameters, you use a command for the database console, **<cons_cmd>**, and additional parameters for the database console **<cons_param>**.

Only use the database console when told to do by Support.

Prerequisites

You have the DBM operator authorization DBStop [Page 21].

The database instance is in `WARM` or `COLD` operation status.

Syntax

```
db_cons <cons_cmd> [<cons_param>]
```

Reply

You receive the database console reply.

```
OK<NL>
<console_output>
```

In the event of errors, see Reply Format [Page 12].

Scroll in the Information on the Database Instance

Use

You can output further information on the database instance.

Prerequisites

You have the DBM operator authorization DBInfoRead [Page 15].

You have entered a command to request database instance information [Page 54]. The keyword `CONTINUE` in the reply indicates that further information is available; this information has not yet been output because of the limited size of the reply memory.

Syntax

```
show_next
```

Reply

```
OK<NL>
[CONTINUE | END]<NL>
<info_record><NL>
<info_record><NL>
```

Values in the individual fields of the reply

END	The requested information on the database instance has been transferred in full. The file is closed automatically.
CONTINUE	More result data is available but was not transferred due to the limited size of the reply memory. You can call up this data by entering the **show_next** command again.
<info_record>	Information

Load the System Tables

Use

System tables are internal tables for the system that users do not define or change.

You have to reload the system tables after a change of database software version.

You do not need to reload system tables when restoring a database instance.

In order to load the system tables, the DBM Server [Page 12] must know the name and password of the database administrator as well as the password of the DOMAIN user. If the DBM Server does not already have this information, enter the database administrator's user and password under **-u <sysdba>,<pwd>**, and the DOMAIN user's password under **-ud <domain_password>**.

See also: *User Manual: SAP DB*

Prerequisites

You have the DBM operator authorization ExecLoad [Page 16] or LoadSysTab [Page 19].

Syntax

```
load_systab [-u <sysdba>,<pwd>] [-ud <domain_password>]
```

Reply

```
OK<NL>
<errcode>,<err_description><NL>
<pgmcode>,<command><NL>
<xload_output_lines><NL>
<xload_protocol_lines><NL>
```

Values for the individual fields of the reply

<errcode>	Error number of DBM Server. If successful the error number is 0

<err_description>	Description of the error
<pgmcode>	Return value of the executed program
<command>	Command line executed by the DBM Server
<xload_output_lines>	Output text of the program
<xload_protocol_lines>	Lines of the LOAD log
<NL>	Line feed

Load SAP-Specific Tables

Use

The SAP-specific tables contain information for the SAP application on the status of the database instance.

You maintain the contents of the SAP-specific tables. Normally, these tables are maintained by the SAP application on a regular basis.

Prerequisites

You have the DBM operator authorization ExecLoad [Page 16] or LoadSysTab [Page 19].

Syntax

```
load_r3tab
```

Reply

```
OK<NL>
<errcode>,<err_description><NL>
<pgmcode>,<command><NL>
<xload_output_lines><NL>
<xload_protocol_lines><NL>
```

Values for the individual fields of the reply

<errcode>	Error number of DBM Server. If successful the error number is 0
<err_description>	Description of the error
<pgmcode>	Return value of the executed program
<command>	Command line executed by the DBM Server
<xload_output_lines>	Output text of the program
<xload_protocol_lines>	Lines of the LOAD log
<NL>	Line feed

List the Information on the Database Instance

Use

You can call up a list containing all of the options for a database instance information request [Page 54].

Prerequisites

You have the DBM operator authorization DBInfoRead [Page 15].

Syntax

```
show_list
```

Reply

You receive a list of the keywords **<keyword>** that you can use to call up detailed information on the database instance.

Use **show <keyword>** to call up the information you require.

In the event of errors, see Reply Format [Page 12].

Deleting Runtime Information After a Database Error

Use

If runtime information is still present in the operating system after a database error, it is not possible to restart the database instance.

Using this command, you remove this runtime information from the operating system and therefore allow the restart of the database instance.

Prerequisites

You have the DBM operator authorization DBStop [Page 21].

Syntax

```
db_clear
```

Reply

The system outputs an OK message.

In the event of errors, see Reply Format [Page 12].

Restart Database Instance

Use

Regardless of the current state of the database instance, you can transfer it initially to the `OFFLINE` mode and then to the `WARM` mode.

Prerequisites

You have the DBM operator authorization DBStop [Page 21].

Syntax

```
db_restart [<option>]
```

Options

You can use the following options **(<option>)** with the command:

-f | -fast [Page 73]	When transferring from `OFFLINE` to `COLD` mode under Windows NT, the *fast* variant of the database kernel is started.
-q | -quick [Page 73]	When transferring from `OFFLINE` to `COLD` mode under Windows NT, the *quick* variant of the database kernel is started.
-s | -slow [Page 74]	When transferring from `OFFLINE` to `COLD` mode under Windows NT, the *slow* variant of the database kernel is started.
-d | -dump	A dump is generated on transferring from `COLD` to `OFFLINE` mode.
-i | -immediate	The transfer from WARM to COLD mode is made without the current transaction being completed first.

Reply

The system outputs an OK message.

In the event of errors, see Reply Format [Page 12].

Starting the Database Instance

Use

You can transfer the database instance from the `OFFLINE` state to `COLD` (Database Instance Transfer to COLD State [Page 62]).

Prerequisites

You have the DBM operator authorization DBStart [Page 20].

Under the Windows NT operating system, this database instance must be registered as a service under the name `SAP DB: <database_name>`.

Syntax

```
db_start [<option>]
```

Options

Options **(<options>)** that determine the kernel variant with which the database instance will run:

-f | -fast [Page 73]

-q | –quick [Page 73]

-s | –slow [Page 74]

If you do not specify an option the system makes a service entry for the fastest available variant of the database kernel.

The **-q | -quick** and **-s | -slow** options should only be used in case of error and in consultation with Support.

Reply

The system outputs an OK message.

In the event of errors, see Reply Format [Page 12].

Starting the LOAD Program

Use

With this command, the LOAD program is executed on the current database instance with the user specified in **<userid> <password>**. Specify a LOAD script file **<file>**. LOAD options in double quotation marks are transferred uninterpreted to the LOAD program. When specifying a relative file name, this is used relative to the directory `<dependent_path>\env`.
(**See also:** Database Manual LOAD)

Prerequisites

You have the DBM operator authorization ExecLoad [Page 16].

Syntax

```
exec_xload <userid> <password> <file> ["<option>"]
```

Reply

```
OK<NL>
<errcode>,<errdescription><NL>
<pgmcode>,<command><NL>
<xload_output_lines><NL>
<xload_protocol_lines><NL>
```

Values for the individual fields of the reply

<errcode>	Error number of DBM Server. If successful the error number is 0
<errdescription>	Description of the error
<pgmcode>	Return value of the executed program
<command>	Command line executed by the DBM Server
<xload_output_lines>	Output text of the program
<xload_protocol_lines>	Lines of the LOAD log
<NL>	Line feed

Starting the PythonLOAD Program

Use

PythonLOAD is an environment programmed in the script language Python that allows load operations to be executed. The system displays information about the execution status of the program as a reply.

With this command, the PythonLOAD program is executed on the current database instance with the user specified in **<userid> <password>**. Specify a PythonLOAD script file **<file>**. If you specify a relative file name, it is used relative to the directory <dependent_path>\env.

PythonLOAD options in double quotation marks are transferred uninterpreted to the PythonLOAD program.

Prerequisites

You have the DBM operator authorization ExecLoad [Page 16].

Syntax

```
exec_load <userid> <password> <file> ["<option>"]
```

Reply

```
OK<NL>
<errcode>,<errdescription><NL>
<pgmcode>,<command><NL>
<pythonload_output_lines><NL>
```

Values for the individual fields of the reply

<errcode>	Error number of DBM Server. If successful the error number is 0
<errdescription>	Description of the error
<pgmcode>	Return value of the executed program
<command>	Command executed by the DBM Server

<pythonload_output_lines>	Output text of the program
<NL>	Line feed

Stopping the Database Instance

Use

With this command the database instance is transferred from COLD or WARM mode (emergency shutdown) to OFFLINE mode.

This command should only be used if the database instance is in COLD state.

Prerequisites

You have the DBM operator authorization DBStop [Page 21].

Syntax

```
db_stop [<option>]
```

Option

If you specify the value **-d | -dump** as an **<option>**, a dump is created when the database instance transfers from COLD to OFFLINE mode.

Reply

The system outputs an OK message.

In the event of errors, see Reply Format [Page 12].

Database Instance Transfer to COLD State

Use

The database instance is transferred to COLD mode.

If the database is in WARM mode, it is first transferred to OFFLINE operation status and only then to operation status COLD.

Prerequisites

You have the DBM operator authorization DBStop [Page 21].

Syntax

```
db_cold [<option>]
```

Options

Options **(<options>)** you can use with the command:

-i | -immediate	SHUTDOWN QUICK used for transfer from WARM to COLD
-no	The database instance is transferred directly from operation status WARM to the operation status COLD (no temporary OFFLINE operation status)
-f | -fast [Page 73]	When transferring from OFFLINE to COLD mode under Windows NT, the *fast* variant of the database kernel is started.
-q | -quick [Page 73]	When transferring from OFFLINE to COLD mode under Windows NT, the *quick* variant of the database kernel is started.
-s | -slow [Page 74]	When transferring from OFFLINE to COLD mode under Windows NT, the *slow* variant of the database kernel is started.

Reply

The system outputs an OK message.

In the event of errors, see Reply Format [Page 12].

Taking the Database Instance OFFLINE

Use

The database instance is transferred to OFFLINE mode.

This means that a database instance in WARM mode is initially transferred into COLD and subsequently into OFFLINE mode.

Prerequisites

You have the DBM operator authorization DBStop [Page 21].

Syntax

db_offline [<option>]

Options

Options **(<options>)** you can use with the command:

-d | -dump	A dump is generated on transferring from COLD to OFFLINE mode.
-i | -immediate	SHUTDOWN QUICK used for transfer from WARM to COLD

Reply

The system outputs an OK message.

In the event of errors, see Reply Format [Page 12].

Database Instance Transfer to WARM State

Use

Regardless of the current state of the database instance it is transferred to `WARM`.

Prerequisites

You have the DBM operator authorization DBStart [Page 20].

Syntax

```
db_warm [<option>]
```

Options

Options **(<options>)** you can use with the command:

-f | -fast [Page 73]	When transferring from `OFFLINE` to `COLD` mode under Windows NT, the *fast* variant of the database kernel is started.
-q | -quick [Page 73]	When transferring from `OFFLINE` to `COLD` mode under Windows NT, the *quick* variant of the database kernel is started.
-s | -slow [Page 74]	When transferring from `OFFLINE` to `COLD` mode under Windows NT, the *slow* variant of the database kernel is started.

Reply

The system outputs an OK message.

In the event of errors, see Reply Format [Page 12].

Installation and Registration Management

Use

Information on the versions of the database software and the generated database instances is managed in Registration Management.

DBM Server Commands [Page 34] are provided for maintaining the installation(s) of the database software and for registering and deleting database instances. Many of these commands, however, do not need to be called explicitly during normal database operation, but are called by installation programs of the database software or by the database application.

Installation and Registration Management Commands [Page 65]

Prerequisites

Note the DBM operator authorizations [Page 14] for each of the DBM Server commands [Page 34].

Installation and Registration Management Commands

Request the Version-Independent Directories [Page 65]	`dbm_getpath`
Change Software Version of Current Database Instance [Page 66]	`db_reg -R`
Define the Version-Independent Directories [Page 66]	`dbm_setpath`
List All Registered Database Instances [Page 67]	`db_enum`
List All Registered Versions of Database Software [Page 68]	`inst_enum`
Deletion of Current Database Instance [Page 69]	`db_drop`
Deleting the Registration of a Variant of the Current Database Instance [Page 70]	`db_unreg`
Deleting the Registration of a Version of the Database Software [Page 70]	`inst_unreg`
Database Instance Registration [Page 71]	`db_create`
Registering a Variant of the Current Database Instance [Page 75]	`db_reg`
Registering a Version of the Database Software [Page 76]	`inst_reg`

Request the Version-Independent Directories

Use

You request the version-independent directories that are configured on the database server. If more than one version of the database software is installed on your computer, version-independent data or programs are stored and found in these directories.

Prerequisites

You do not need to log on to the DBM Server [Page 12] to execute this DBM Server command [Page 34]. You also require no DBM operator authorization [Page 14].

Syntax

```
dbm_getpath [IndepDataPath | IndepProgPath]
```

`IndepDataPath`	Directory for version-independent data
`IndepProgPath`	Directory for version-independent programs

Change Software Version of the Current Database Instance

Use

You change the version of the database software for the current database instance.

You can only switch between versions of the SAP DB software that **do not** require migration of the database instance.

Prerequisites

The database instance is in `OFFLINE` operation status (Taking the Database Instance OFFLINE [Page 63]).

You have called the Database Manager CLI with the -d [Page 31] option.

Syntax

```
db_reg -R
```

Procedure

1. End all sessions with the DBM Server, particularly all sessions in the Database Manager GUI.
2. Run the command **`db_register -R`**
3. If you are working with the Database Manager CLI in session mode [Page 180], terminate the Database Manager CLI now.

Result

For every subsequent access to this database instance, the Database Manager program uses the new version of the database software.

Define the Version-Independent Directories

Use

You define the version-independent directories.
If more than one version of the database software is installed on your computer, version-independent data or programs are stored and found in these directories.

Prerequisites

You do not need to log on to the DBM Server [Page 12] to execute this DBM Server command [Page 34].

You also require no DBM operator authorization [Page 14].

Syntax

dbm_setpath [IndepDataPath | IndepProgPath] <path>

IndepDataPath	Directory for version-independent data
IndepProgPath	Directory for version-independent programs
<path>	Path

Reply

The system outputs an OK message.

In the event of errors, see Reply Format [Page 12].

List All Registered Database Instances

Use

You can call up the list of all database instances registered on the server.

If you specify the **[-s]** option, the service database instances for each version of the database software are also listed (Kernel Access Through a Service Session [Page 173]).

Prerequisites

You have called the Database Manager CLI using the -n [Page 32] option.

You do not need to log on to the DBM Server [Page 12] to execute this DBM Server command [Page 34].

You also require no DBM operator authorization [Page 14].

Syntax

db_enum [-s]

Reply

The system outputs an OK message. Then a table is output showing all of the registered database instances.

```
OK<NL>
<database_name><TAB><inst_root><TAB><version>[<TAB><description>]<NL>
<database_name><TAB><inst_root><TAB><version>[<TAB><description>]<NL>
...
```

Reply values

<database_name>	Name of database instance
<inst_root>	Installation path

<version>	Software installation version
<description>	More information about the database instance

```
MK1       C:\Program Files\SAP DBTech\V72"     7.2.1.0
   fast   offline
MK1       C:\Program Files\SAP DBTech\V73"     7.3.0.14
   slow   running
```

List All Registered Versions of the Database Software

Use

You can call up the list of all registered versions of the database software on the server.

Prerequisites

You do not need to log on to the DBM Server [Page 12] to execute this DBM Server command [Page 34].

You also require no DBM operator authorization [Page 14].

Syntax

inst_enum

Reply

The system outputs an OK message. The system outputs a table of the registered versions <version> and their installation paths <inst_root>.

```
OK<NL>
<version><TAB><inst_root><NL>
<version><TAB><inst_root><NL>
...
```

```
OK
7.2.1.0  "C:\Program Files\SAP DBTech\V72"
7,30,00,14     "C:\Program Files\SAP DBTech\V73"
```

Deletion of Current Database Instance

Use

With this command, you delete the database instance registration [Page 71].

If you specify the option **<WITHOUTFILES>** for this command, the database files on the DBM Server [Page 12] are retained. If you do not specify this option, the files for the database instance are also deleted.

Prerequisites

The database instance is in `OFFLINE` operation status (Taking the Database Instance OFFLINE [Page 63]).

You have the DBM operator authorization InstallMgm [Page 18].

Syntax

```
db_drop [<WITHOUTFILES>]
```

Reply

The system outputs an OK message.

In the event of errors, see Reply Format [Page 12].

db_drop Command: Example

1. Stop the database instance:

dbmcli -d [Page 31] **<database_name> <userid>,< password>** db_stop [Page 62]

or

dbmcli -d <database_name> -u [Page 24] **<userid>,< password>** db_offline [Page 63]

2. Deleting the current database instance of the database server:

dbmcli -d <database_name> -u <userid>,< password> db_drop [Page 69]

This deletes the database instance's files.

Deleting the Registration of a Variant of the Current Database Instance

Use

You can register several variants of one database instance (Registering a Variant of the Current Database Instance [Page 75]). The difference between these variants is that they use different variants of the database kernel.

With this command, you remove the specified variant of the current database instance from the server. Once you have done this, you will not be able to start this variant of the database instance again. The deletion procedure has no effect on the current operation status of the database instance. A database instance in `WARM` or `COLD` mode remains active and is thus recognized by the system.

All parameters listed here apply to the Windows NT operating system only. This command is ignored under the UNIX operating system.

Prerequisites

The database instance is in `OFFLINE` mode.

You have the DBM operator authorization InstallMgm [Page 18].

Syntax

```
db_unreg <option>
```

Options

You use the following options (<option>) to specify which database instance entry variant is to be deleted:

-f | -fast [Page 73]

-q | -quick [Page 73]

-s | -slow [Page 74]

The **`-q | -quick`** and **`-s | -slow`** options should only be used in case of error and in consultation with Support.

```
db_unreg MK1 -f
```

The registration of the `fast` variant of database instance `MK1` is deleted.

Reply

The system outputs an OK message.

In the event of errors, see Reply Format [Page 12].

Deleting the Registration of a Version of the

Database Software

Use

You remove an entry from the list of registered versions of the database software. Components of a version that require this entry are then no longer operable.

Specify the installation path **<dependent_path>** as a parameter.

Prerequisites

You have called the Database Manager CLI with -R [Page 28] option.

You are logged on to the operating system (Operating System Logon [Page 36]).

You do not require any DBM operator authorization.

Syntax

```
inst_unreg <dependent_path>
```

Reply

The system outputs an OK message.

In the event of errors, see Reply Format [Page 12].

Database Instance Registration

Use

You register a new database instance using an operating system user.

Prerequisites

The following prerequisites must be fulfilled in accordance with the operating system on the server on which the database instance is to be created:

Communication	Windows NT	UNIX
local	Operating system user has administration rights for database server * (Logon to the operating system is implicitly given)	Logon to the operating system using option **-s** (Local Operation [Page 30]) when Calling the Database Manager CLI [Page 23]
remote	Operating system user has administration rights for database server and right *log on as batch job* * Log on to the operating system using **db_create** and option **<os_user>,<password>**	Log on to the operating system using **db_create** and option **<os_user>,<password>**

If several versions of the database software are available on the database server, you must specify the path of the version to which this database instance is to be assigned when registering the database instance. To do this, use the option **-R** (Installations Directory of the Database Instance [Page 28]) when calling the Database Manager CLI.

*(You can find information about maintaining operating system users in your operating system documentation.)

Syntax

```
db_create [<option>] <database_name> <userid>,<password>
[<os_user>,<password>]
```

<database_name>	Name of database instance, maximum length 8 characters
<userid>,<password>	DBM Operator This operator is stored on the database server when creating the database instance. Subsequent access to the database instance is possible only with this operator.
[<os_user>,<password>]	Operating System User If you want to install the database instance on a remote server, you must also specify the operating system user for this server. If the logon to the operating system fails, the database instance cannot be installed.

From now on, to access this database instance on you need the **<userid>** and **<password>** specified at registration.

Options

Options **(<option>)** that determine which database instance kernel variant is registered:

-f | -fast [Page 73]

-a | –auto [Page 73]

The options -q | –quick [Page 73] and -s | –slow [Page 74] should only be used in case of error and in consultation with Support.

If you do not specify an option the system makes a service entry for the fastest available variant of the database kernel.

All parameters listed here apply to the Windows NT operating system only.

Under the UNIX operating system these parameters are ignored.

Creating a local database instance under Windows NT:
```
db_create MK1 samplename,secret
```

Creating a remote database instance under Windows NT:
```
db_create MK1 samplename,secret winuser,win
```

Reply

There is an OK message after the command has been executed successfully.

In the event of errors, see Reply Format [Page 12].

Example: How to Create a Database Instance [Page 74]

Kernel Variant -a | -auto

`-a	-auto`	If you use this option, under Windows NT the service entry *Start-up-type* in the Control Panel is set to `automatic`. Under the UNIX operating system this parameter is ignored.

Kernel Variant -f | -fast

`-f	-fast`	Default value of the Database Manager CLI Fastest database kernel variant, with minimum logging and runtime checks. The specification of this option has the following effect: **Windows NT** During database instance registration [Page 71], the service entry of the *fast* variant of the database kernel is used. The kernel variant *fast* is started when the command to start the database instance [Page 59] is run. **UNIX** The command to register the database instance has no effect. The specification of the option `-f	-fast` therefore only makes sense when starting the database instance.

Kernel Variant -q | -quick

`-q	-quick`	Variant of the database kernel with various logs and runtime checks. The specification of this option has the following effect: **Windows NT** During database instance registration [Page 71], the service entry of the *quick* variant of the database kernel is used. The kernel variant *quick* is started when the command to start the database instance [Page 59] is run. **UNIX** The command to register the database instance has no effect. The specification of the option `-q	-quick` therefore only makes sense when starting the database instance.

This option should only be used in case of error and in consultation with Support.

Kernel Variant -s | -slow

`-s	-slow`	Variant of the database kernel with extensive logs and runtime checks. The specification of this option has the following effect: **Windows NT** During database instance registration [Page 71], the service entry of the *slow* variant of the database kernel is used. The kernel variant *slow* is started when the command to start the database instance [Page 59] is run. **UNIX** The command to register the database instance has no effect. The specification of the option `-s	-slow` therefore only makes sense when starting the database instance.

This option should only be used in case of error and in consultation with Support.

db_create Command: Example

You only need a few DBMCLI commands to create a new database instance using the Database Manager CLI.

First create a new database instance and define the first user:

```
dbmcli db_create <database_name> <userid>,<password>
```

You can also use the following variations for this operation:

- If you have a choice between several versions of the database software, use the option `-R <inst_root>` to specify your choice.

- If you want to install a new database instance on a remote server, use the option **-n <server_node>**. You also need authorization for the operating system.

  ```
  dbmcli -n <server_node> db_create <database_name>
  <userid>,<password> <os_user>,<password>
  ```

- On UNIX platforms there is no difference between remote and local communication. Therefore authorization is always required. When you are creating a new database instance locally, you can avoid communication and therefore the authorization otherwise required by using the option **-s**:

  ```
  dbmcli -s db_create <database_name> <userid>,< password>
  ```

Use a DBMCLI script for the rest of the installation:

```
dbmcli -d <database_name> -u <userid>,<password> -i <script_file>
```

Example of a script:

```
param_startsession [Page 95]
param_init [Page 97]
param_put [Page 92] MAXUSERTASKS 5
param_checkall [Page 103]
param_commitsession [Page 92]
param_adddevspace [Page 104] 1 SYS sys_001 F
param_adddevspace 1 LOG LOG_001 F 2000
param_adddevspace 1 DATA DAT_001 F 10000
db_cold [Page 62]
util_connect [Page 164]
util_execute [Page 166] INIT CONFIG
util_activate [Page 163] <sysdba>,<password>
load_systab [Page 56] -ud <domain_password>
```

Obviously you will have to adjust the individual parameter values.

Registering a Variant of the Current Database Instance

Use

You register a variant of a database instance. If you do not specify any options, the system makes a service entry for the fastest available variant of the database kernel.

The **-f | -fast**, **-q | -quick**, and **-s | -slow** options should only be used in case of error and in consultation with Support.

The options **-f | -fast**, **-q | -quick** and **-s | -slow** allow you to register a database instance that has already been registered in an additional variant. If the same variant is registered more than once, the existing entry is overwritten with the same data.

These options are only valid for the Windows NT operating system.

Under the UNIX operating system these parameters are ignored.

Prerequisites

You have the DBM operator authorization InstallMgm [Page 18].

The database instance is in `OFFLINE` mode.

You have called the Database Manager CLI with the -d [Page 31] option.

Syntax

```
db_reg [<option>]
```

Options

-f | -fast [Page 73]

-q | –quick [Page 73]

-s | –slow [Page 74]

see also: Deleting the Registration of a Variant of the Current Database Instance [Page 70]

Registering a Version of the Database Software

Use

You can enter the version of the current DBM Server [Page 12] in the list of registered versions of the database software.

The version is registered once only, upon installation of the relevant database software. The installation must be registered for this software to function correctly.

Prerequisites

You have called the Database Manager CLI with -R [Page 28] option.

You are logged on to the operating system (Operating System Logon [Page 36]).

Syntax

```
inst_reg [-k <key>][-c]
```

Options

You must specify a unique key **`-k <key>`** if you have installed the same version of the database software more than once.

If only the client programs of the database software have been installed on a computer, register the client with the **`-c`** option.

Reply

The system outputs an OK message.

In the event of errors, see Reply Format [Page 12].

Configuration of the DBM Server

Use

You can use the following DBM Server commands [Page 34] to configure the DBM Server [Page 12]:

Requesting a DBM Server Parameter [Page 77]	**dbm_configget**
Setting a DBM Server Parameter [Page 78]	**dbm_configset**

Prerequisites

You have the DBM operator authorization InstallMgm [Page 18].

Requesting a DBM Server Parameter

Use

You request the value of the DBM Server [Page 12] parameter identified as **<parameter_name>**. You can specify that the value is stored unencrypted in the DBM Server with the option **-raw**.

If the system outputs a sequence of letters and digits instead of a value, the value is encrypted.

Note that parameter names are case sensitive when working with the Database Manager CLI.

Prerequisites

You have the DBM operator authorization InstallMgm [Page 18].

Syntax

dbm_configget [-raw] <parameter_name>

Request the encrypted value **TEST** with the **-raw** option.

```
dbmcli -d DB -u dbm,dbm dbm_configget -raw
OK
c92ee241db0bf5632bc029bcf98f186b440bb548c0f056eb
```

Reply

```
OK<NL>
<value><NL>
```

<value> represents the value of the requested parameter.

In the event of errors, see Reply Format [Page 12].

Setting a DBM Server Parameter

Use

You set the value specified for the parameter specified as **<parameter_name>** in the configuration of the DBM Server [Page 12]. The new parameter value is registered by the system, and is transferred to the parameter file [Page 78] as the effective value when the database instance is restarted.

You can specify that the value is to be stored unencrypted in the DBM Server with the option **-raw**.

Maximum number of characters for encrypted values: 18
A longer value is shortened to 18 characters by the Database Manager CLI.

Maximum number of characters for unencrypted values: 512

Prerequisites

You have the DBM operator authorization InstallMgm [Page 18].

Syntax

```
dbm_configset <parameter_name> <value>
```

Reply

The system outputs an OK message.

In the event of errors, see Reply Format [Page 12].

Configuring Database Instances

Use

The configuration parameters of a database instance are stored in the **description file** and the **parameter file**.

The default values, value areas, calculation rules, and parameter properties the DBM Server [Page 12] needs to process parameters correctly are all stored in the description file.

Th description file and a parameter file that has standard values assigned to it are installed with the database instance. With the commands for the configuration of database instances you change the entries in the parameter file of the database instance.

Prerequisites

For some of the commands for configuring database instances you require a parameter session [Page 95].

Note the DBM operator authorizations [Page 14] for each of the DBM Server commands [Page 34].

Procedure

To do this, when you set up the session with the DBM Server, you use the -d option to specify the name of the database instance [Page 31] whose parameters you want to process. On entering the **param_*** commands, a search is made in the parameter file belonging to this database instance for the value of the respective parameters.

If you do not specify a name or you specify the name of a non-existent database, the required parameter cannot be found.

Database Instance Configuration Commands Overview [Page 80]

Parameter Properties

Property	Explanation	Possible values
CASESENSITIVE	Upper/lower case distinctions are relevant (for example, file names under UNIX)	**YES | NO**
CHANGE	Parameter can be changed	**YES | NO**
CLEAR	In a new installation, the parameter must not be copied from another database instance (examples: database instance name, devspaces)	**YES | NO**
DEVSPACE	Type of parameter (**Yes**: devspace parameter, **NO**: kernel parameter)	**YES | NO**
DISPLAYNAME	Character string (may include spaces) displayed in place of the parameter name in the DBMGUI	
DYNAMIC	Parameter has a position indicator in the description file (for example: DATADEV_?) In the parameter file such parameters become, for example, DATADEV_00, DATADEV_01, and so on)	**YES | NO**
GROUP	Assignment to a group for display in the Database Manager GUI **GENERAL: General database parameter**, visible **EXTENDED: Special database parameter**, visible **SUPPORT:** Parameter list specified only for support, visible **NO:** The parameter is not accessible using the Database Manager GUI **INFO:** The parameter contains special information for users	**GENERAL** **| EXTENDED** **| SUPPORT** **| NO** **| INFO**
INSTANCES	Instance type for which the parameter is relevant No entry: Parameter is generally valid	**LVC | OLTP | BW | CS**
INTERN	The value is only in the description file, not in the parameter file (Configuring Database Instances [Page 78])	**YES | NO**
LASTKNOWNGOOD	Valid value of the parameter with which the database instance was last started	
MANDATORY	Mandatory parameter	**YES | NO**
MAX	Maximum parameter value	

MIN	Minimum parameter value	
MODIFY	You can still change the parameter after the database instance has been generated (for example: RUNDIRECTORY: **NO**)	**YES | NO**
OVERRIDE	You can overwrite the DBM Server's proposed value	**YES | NO | HIGHER**
VALUESET	Permitted value set for a parameter The individual values must be separated by spaces or tabs. Permitted string constants in character string parameters must be set off in quotation marks.	

Database Configuration Commands Overview

Parameter File List [Page 99]	`param_versions`
Parameter File Deletion [Page 99]	`param_rmfile`
Parameter Deletion [Page 100]	`param_directdel`
Delete Devspace Parameters [Page 100]	`param_deldevspace`
Opening the Parameter History [Page 101]	`param_gethistory`
Check All Parameters [Page 103]	`param_checkall`
Setting Devspace Parameters [Page 104]	`param_adddevspace`
Reset the Parameter File to a Previous Version [Page 106]	`param_restore`

The commands of the class `param_direct*` access the parameter file of the database instance directly. These commands do not require a parameter session and they **cannot** be rolled back.

Terminating a Parameter Session

Use

You terminate the parameter session. All of the changes [Page 92] and corrections [Page 98] made to parameters in this session are rejected. The internal data structures in which the contents of the description file and the parameter file were stored are removed (see also: Configuring Database Instances [Page 78]).

If you change values without a parameter session, that is, directly in the parameter file (Make direct changes with param_directput [Page 95] and delete directly with param_directdel [Page 100]), the changes are immediately written to the parameter file, that is, they cannot be rejected.

Prerequisites

You have opened a parameter session [Page 95].

You have the DBM operator authorization ParamRead [Page 20].

Syntax

```
param_abortsession
```

Reply

The system outputs an OK message.

In the event of errors, see Reply Format [Page 12]

Requesting All Data for a Parameter

Use

You request all of the data for the parameter identified as **<parameter_name>**.

Prerequisites

You have the DBM operator authorization ParamRead [Page 20].

Syntax

param_getfull <parameter_name>

Example [Page 82]

Reply

The system outputs an OK message followed by all of the data for the parameter concerned.

```
OK<NL>
<type><NL>
<default><NL>
<value><NL>
<property> <value><NL>
<property> <value><NL>
...
HELP
<helpline><NL>
<helpline><NL>
...
EXPLAIN
<explainline><NL>
<explainline><NL>
...
```

Reply values

<type>	Data type of the parameter
<default>	System default value
<value>	Current value, taken from the parameter file at the start of the session
<property>	Parameter property [Page 79]
<helpline>	Help text
<explainline>	Explanatory text

In the event of errors, see Reply Format [Page 12].

param_getfull Command: Example

Request the data for the parameter RUNDIRECTORY

```
dbmcli -d DB -u dbm,dbm param_getfull RUNDIRECTORY
```

```
OK
c64
d:\sapdb\usr\wrk\DB
CHANGE          YES
INTERN          NO
MANDATORY       YES
CLEAR           YES
DYNAMIC         NO
CASESENSITIVE   YES
OVERRIDE        NO
DEVSPACE        NO
MODIFY          YES
GROUP           GENERAL
DISPLAYNAME
VALUESET
MAX
MIN
INSTANCES
LASTKNOWNGOOD   d:\sapdb\usr\wrk\DB
HELP
Path where context and diagnosis information is stored
EXPLAIN
Path where context and diagnosis information is stored for this
database instance.
(char(64))
```

Requesting All Properties of a Parameter

Use

You request all properties [Page 79] of the parameter identified as **<parameter_name>**. All properties are listed with their values.

Prerequisites

You have the DBM operator authorization ParamRead [Page 20].

Syntax

param_getproperties <parameter_name>

Reply

```
OK<NL>
<property> <value><NL>
<property> <value><NL>
...
```

In the event of errors, see Reply Format [Page 12].

Requesting All Parameters of the Current Parameter File

Use

You request all parameters of the current parameter file with their values. One parameter with name and value is output per line.

The **param_directgetall** command accesses the database instance parameter file directly. It does not require a parameter session and so cannot be rolled back.

Prerequisites

You have the DBM operator authorization ParamRead [Page 20].

Syntax

```
param_directgetall
```

Reply

```
OK<NL>
<parameter_name> <value><NL>
<parameter_name> <value><NL>
...
```

In the event of errors, see Reply Format [Page 12].

Requesting the Data for All Parameters

Use

You request all the parameters from the description file and the parameter file (see also: Configuring Database Instances [Page 78]).

If the reply is successful, the name of each parameter <parameter_name>, its data type <type> and its current value <value> will be output. This is the value, which is taken over from the parameter file at the start of the session. If the parameter is not in it, the default value from the description file is displayed.

Prerequisites

You have the DBM operator authorization ParamRead [Page 20].

Syntax

```
param_extgetall [<property>=<value>]
```

<property> Option

By specifying properties [Page 79] you can limit what is displayed. However, each property can only be specified once. If the same property is entered more than once, the value of the last entry is used.

A parameter must correspond to the specified value in all specified properties (AND link). Several values for one property separated by commas are treated as OR-linked.

Property	Possible values
CASESENSITIVE	**YES | NO**
CHANGE	**YES | NO**
CLEAR	**YES | NO**
DEVSPACE	**YES | NO**
DISPLAYNAME	Character string (may include spaces) displayed in place of the parameter name in the DBMGUI
DYNAMIC	**YES | NO**
GROUP	**GENERAL | EXTENDED | SUPPORT | NO**
INFO	**YES | NO**
INSTANCES	Instance types for which the parameter is relevant No entry: Parameter is generally valid
INTERN	**YES | NO**
MANDATORY	**YES | NO**
MAX	Maximum parameter value
MIN	Minimum parameter value
MODIFY	**YES | NO**
OVERRIDE	**YES | NO | HIGHER**
VALUESET	Permitted parameter value set

param_getextall CHANGE=YES GROUP=GENERAL

This command finds all parameters that are changeable **and** assigned to the *General - General database parameters* group.

param_getextall GROUP=GENERAL,EXTENDED

This command finds all parameters that are assigned to the *General - General database parameters* group **or** the *Extended - Special database parameters* group.

Reply

```
OK<NL>
<parameter_name> <type> <value><NL>
<parameter_name> <type> <value><NL>
...
```

In the event of errors, see Reply Format [Page 12].

Requesting the Data for a Devspace Parameter

Use

You request the data for a devspace.

Prerequisites

You have the DBM operator authorization ParamRead [Page 20].

Syntax

```
param_getdevspace <dev_no> <dev_mode>
```

Options

<dev_no>	Number of devspace
<dev_mode>	Type of devspace

Option <dev_mode>

SYS	System devspace
MSYS	Mirrored system devspace
DATA	Data devspace; the number of the data devspace is assigned by the system as a four-digit number with leading zeros when setting up the devspace
MDATA	Mirrored data devspace
LOG	Log devspace; the number of the log devspace is assigned by the system as a three-digit number with leading zeros when setting up the devspace
MLOG	Mirrored log devspace (LOG_MODE=DUAL)

```
dbmcli -d DB -u dbm,dbm param_getdespace 1 DATA

OK
DAT_001
F
10000
```

Reply

The system displays the data for the devspace

```
OK<NL>
<devname>
<devtype>
<devsize>
```

Values for the Fields of the Reply

<devname>	Name of devspace/file
<devtype>	Type of devspace (such as file, raw device)
<devsize>	Size of devspace

In the event of errors, see Reply Format [Page 12].

Requesting Parameter Data

Use

You request the name, data type, and current value of the parameter specified in **<parameter_name>**. The parameter help file and the current database instance parameter file are searched (see also: Configuring Database Instances [Page 78]).

Prerequisites

You have the DBM operator authorization ParamRead [Page 20].

Syntax

```
param_extget <parameter_name>
```

Reply

```
OK<NL>
<parameter_name> <type> <value><NL>
```

<parameter_name**>**	Name of parameter
<type>	Data type of the parameter
<value>	Current value This is the value, which is taken over from the parameter file at the start of the session. If the parameter is not in it, the default value from the description file is displayed here.

In the event of errors, see Reply Format [Page 12].

Requesting the Data for Multiple Devspace Parameters

Use

You request the data for multiple or all devspaces.

Prerequisites

You have the DBM operator authorization ParamRead [Page 20].

Syntax

```
param_getdevsall [<dev_mode>]
```

Options

<dev_mode>	Type of devspace

Option <dev_mode>

SYS	System devspace
MSYS	Mirrored system devspace
DATA	Data devspace; the number of the data devspace is assigned by the system as a four-digit number with leading zeros when setting up the devspace
MDATA	Mirrored data devspace
LOG	Log devspace; the number of the log devspace is assigned by the system as a three-digit number with leading zeros when setting up the devspace
MLOG	Mirrored log devspace (`LOG_MODE=DUAL`)

Requesting the data for all devspaces of the type `DATA`

```
dbmcli -d DB -u dbm,dbm param_getdevsall DATA
```

```
OK
MAXDATADEVSPACES    3
DATADEV_0001        10000       F   DAT_001
DATADEV_0002        11000       F   DAT_002
```

Reply

The system displays the data for the devspace

```
OK<NL>
<paramname> <value>
<paramname> <value>
...
<devspace> <devsize> <devtype> <devname>
<devspace> <devsize> <devtype> <devname>
...
```

Values for the Fields of the Reply

`<paramname>`	Name of a parameter relevant to a devspace
`<value>`	Value of the parameter
`<devspace>`	Devspace identifier
`<devname>`	Name of devspace/file
`<devtype>`	Type of devspace (such as file, raw device)
`<devsize>`	Size of devspace

In the event of errors, see Reply Format [Page 12].

Requesting the Current Parameter Value

Use

You request the current value of the parameter identified as **<parameter_name>**.

The system outputs the value of the parameter as saved in the parameter file the last time the database instance was started.

Prerequisites

You have the DBM operator authorization ParamRead [Page 20].

Syntax

```
param_getvalue <parameter_name>
```

Reply

```
OK<NL>
<value>
```

In the event of errors, see Reply Format [Page 12].

Requesting the Explanatory Text

Use

You request the explanatory text associated with the parameter identified as **<parameter_name>**.

The system outputs an OK message followed by the explanatory text <explanation line>. If no explanatory text is stored, the system does not display a result line.

Prerequisites

You have the DBM operator authorization ParamRead [Page 20].

Syntax

```
param_getexplain <parameter_name>
```

Reply

```
OK<NL>
<explanation line><NL>
<explanation line><NL>
...
```

In the event of errors, see Reply Format [Page 12].

Requesting the Help Text

Use

You request the help text associated with the parameter identified as **<parameter_name>**. The system outputs an OK message followed by the help text <helpline>.

If no help text is stored, the system does not display a result line.

Prerequisites

You have the DBM operator authorization ParamRead [Page 20].

Syntax

param_gethelp <parameter_name>

Reply

```
OK<NL>
<helpline><NL>
<helpline><NL>
...
```

In the event of errors, see Reply Format [Page 12].

Requesting the Parameter Type

Use

You request the type of the parameter identified as **<parameter_name>**.

There is an OK message after the command has been executed successfully. This is followed by the data type <type> of the specified parameter. The following output values for **<**type**>** are possible:

```
int2 | int4 | c8 | c18 | c24 | c40 | c64
```

Prerequisites

You have the DBM operator authorization ParamRead [Page 20].

Syntax

param_gettype <parameter_name>

Reply

```
OK<NL>
<type>
```

In the event of errors, see Reply Format [Page 12]

Requesting the System Default

Use

You request the system default value of the parameter identified as **<parameter_name>**.

The system outputs an OK message and a line showing the system <default> value. The line is blank if the parameter is not present.

Prerequisites

You have the DBM operator authorization ParamRead [Page 20].

Syntax

param_getdefault <parameter_name>

Reply

```
OK<NL>
<default>
```

In the event of errors, see Reply Format [Page 12].

Parameter File Value Request

Use

You request the value of parameter **<parameter_name>** from the parameter file [Page 78].

After a successful search an OK message is displayed. The name of the parameter <parameter_name> and its value <value> are displayed in the following line. The message output if the specified parameter is not in the parameter file is not found.

The **param_directget** command accesses the database instance parameter file directly. It does not require a parameter session and so cannot be rolled back.

Prerequisites

You have the DBM operator authorization ParamRead [Page 20].

Syntax

param_directget <parameter_name>

Reply

```
OK<NL>
<parameter_name> <value><NL>
```

In the event of errors, see Reply Format [Page 12].

Parameter Value Change

Use

You use this command to change the value of a parameter in the internal data structures and store it in the DBM Server [Page 12].

In contrast to the function for a parameter file direct change [Page 95], the parameter is not written straight to the parameter file and you can still reject it by not confirming the parameter change or by terminating the parameter session [Page 81].

The change of this parameter is rejected, as the entered value is not permissible or the parameter characteristics make the change impermissible.

Prerequisites

You have opened a parameter session [Page 95].

You have the DBM operator authorization ParamCheckWrite [Page 19].

Syntax

```
param_put <parameter_name> <user_value>
```

Reply

The system outputs an OK message.

In the event of errors, see Reply Format [Page 12].

Parameter Change Confirmation

Use

You confirm all changes to the parameters and transfer them to the parameter file. Before storing them the check status of all parameters is verified.

If you specify the **NOCLOSE** option, the parameter session remains open (Opening a Parameter Session [Page 95]) after the parameters have been stored. If you do not specify this option, the parameter session is closed after the parameters have been stored.

The changed values in the parameter file are effective when the database instance is restarted.

Prerequisites

You have changed parameter values in a parameter session [Page 95].

You have the DBM operator authorization ParamCheckWrite [Page 19].

Syntax

```
param_commitsession [NOCLOSE]
```

Successful Reply

The system outputs an OK message.

Error Message

Execution of the command is refused:

```
ERR
14,ERR_XPCHECK_CN00 : param check failure/request
<identifier>      <checkstatus>
<user_value>
<computed_value>
```

Values for the individual fields of the reply

<checkstatus>	The following check status may occur: Mandatory: An obligatory parameter has been assigned a blank value. Constraint: a constraint could not be fulfilled. Request: the user input deviates both from the value transferred to the system and confirmed by it earlier, as well as from the system default value.
<user_value>	Value that the user entered with **param_put**.
<computed_value>	Default value calculated by the system

Check status request is output only if the user entry varies from the value of the parameter in the parameter file.

Continuation of Procedure

If the check status is a request, you can correct the error as follows:

Specify the valid value with **param_putconfirm** (Correcting Parameters [Page 98]). Otherwise the system default <computed_value> is used this for parameter for all further calculations of other parameters.

In the following cases the new parameter is refused:

- The parameter has the value NOBODY for the CHANGE property [Page 79]
- - The parameter has the value NO for the property MODIFY and the parameter file was already checked by the kernel (parameter __PARAM_CHANGED___ exists)
- You are dealing with a devspace parameter (DEVSPACE property = YES) and there are changed kernel parameters in the parameter file (DEVSPACE property= NO), that have not yet been checked by the kernel (Exception: You are creating a new database instance)
- The parameter is a kernel parameter and devspace parameters have already been changed, but not yet checked by the kernel.

Scrolling through the Parameter History

Use

You can call up the rest of the contents of the parameter history.

```
If you a reply of only:

OK
<header_line><NL>
```

then all specified entries have already been displayed.

If you have set selection criteria in the options on opening the parameter history [Page 101], then these also apply while you are scrolling through the parameter history.

Prerequisites

You have opened the parameter history.

You have the DBM operator authorization ParamRead [Page 20].

Syntax

param_gethistorynext

Reply

```
OK
<header_line><NL>
<param_line><NL>
<param_line><NL>
...
```

Values for the individual fields of the reply

<header_line>	contains the names of the fields displayed S stands for STATE and describes the parameter's change state
<param_line>	contains the values for the fields described in the <header_line>, the values are in the same column position as the field description in the <header_line> Field S for STATE can contain any of the following entries: c: changed d: deleted a: active

Parameter File Value Direct Change

Use

With this command you change the parameters of the database instance. Unlike when you use the parameter change [Page 92] function, the value **<value>** for the file identified as **<key_name>** is written straight to the parameter file. An entry is only made if the parameter is already contained in the parameter file.

In the following cases the parameter change is refused:

- The parameter has the value NOBODY for the CHANGE property [Page 79]
- The parameter has the value NO for the property MODIFY and the parameter file was already checked by the kernel (parameter __PARAM_CHANGED___ exists)
- You are dealing with a devspace parameter (DEVSPACE property = YES) and there are changed kernel parameters in the parameter file (DEVSPACE property= NO), that have not yet been checked by the kernel
  (Exception: You are creating a new database instance)
- The parameter is a kernel parameter and devspace parameters have already been changed, but not yet checked by the kernel.

The entered value for the parameter is not checked for consistency. Therefore unqualified changes to parameters are also possible using the command **param_directput**. These can then **not** be rejected by param_abortsession [Page 81] because the changes are written directly to the parameter file.

Prerequisites

You have the DBM operator authorization ParamFull [Page 19].

Syntax

```
param_directput <key_name> <value>
```

Reply

The system outputs an OK message.

In the event of errors, see Reply Format [Page 12].

Opening a Parameter Session

Use

You open a parameter session to check or change the parameters of a database instance.

The DBM Server loads the description file and parameter file of the database instance (see also: Configuring Database Instances [Page 78]) and stores the contents in internal data structures. All access operations to this internal data take place within a parameter session.

In case of syntax error an error message is displayed which contains the file name, line number and line text.

Prerequisites

You have the DBM operator authorization ParamRead [Page 20].

Syntax

```
param_startsession
```

Reply

The system outputs an OK message.

In case of syntax error an error message is displayed which contains the file name, line number and line text.

Adding a Devspace

Use

You define additional data or log devspaces to extend the disk space for the database instance.

Prerequisites

You have the DBM operator authorization ParamFull [Page 19] or AccessUtility [Page 22].

Syntax

```
db_adddevspace <mode> <master_name> <master_type> <size>
[<mirror_name> <mirror_type>]
```

<mode>	Type of devspace that is to be added. Possible values are: **DATA** (Data devspace) **LOG** (Log devspace)
<master_name>	Name of device/file
<master_type>	Type of devspace that is to be added. Possible values are: **F** (File) **R** (Raw device) **L** (Link)
<size>	Size of the devspace to be added in pages
<mirror_name>	Name of the mirror device (only relevant if parameter **MIRRORED_DATA = YES** and **LOG_MODE = DUAL**)
<mirror_type>	Type of the mirror device (only relevant if parameter **MIRRORED_DATA = YES** and **LOG_MODE = DUAL**)

Reply

The system outputs an OK message.

In the event of errors, see Reply Format [Page 12].

Parameter Initialization for a New Database Instance

Use

You initialize all the parameters needed to generate a new database instance.

This procedure uses the name of the database instance [Page 31] from the DBM Server session. No parameter file can yet exist for this database instance.

All parameters are assigned their default values from the description file (see also: Configurating Database Instances [Page 78]). Then a complete computation run is carried out. The values from the calculation run overwrite the default values.

You get no message about non-fulfilled constraints or missing obligatory values. The initialization takes place only internally in the DBM Server. The parameters in the parameter file are not backed up until parameter change confirmation [Page 92].

The following parameters are assigned values by the system:

- `SERVERDB` is assigned to the database name
- `RUNDIRECTORY` contains a directory name `<work_path>\<serverdb>`
- `KERNEL_VERSION` is initialized with the version label stored in the DBM Server

Use this command to check the new parameter values in the parameter file for the correct syntax.

Prerequisites

You have opened a parameter session [Page 95].

You have the DBM operator authorization InstallMgm [Page 18].

Syntax

```
param_init
```

Reply

The system outputs an OK message.

In the event of errors, see Reply Format [Page 12]

Parameter File Copy

Use

You copy the parameter file of an existing database instance as the initial configuration for a new database instance.

You can only copy a source parameter file if the new database instance does not yet have one.

No changes made during the current parameter session up to this point are transferred into the parameter file.

Parameters that cannot be changed because of their properties [Page 79] retain their system default values copied from the description file. This applies to the following parameters:

- `SERVERDB` is assigned to the database name
- `RUNDIRECTORY` contains a directory name `<work_path>\<serverdb>`
- `KERNELVERSION` is initialized with the version label stored in the DBM Server

Specify the name of the source database in **`<source_db>`**.

(**See also:** Configuring Database Instances [Page 78])

Prerequisites

You have opened a parameter session [Page 95].

You have the DBM operator authorization InstallMgm [Page 18].

Syntax

`param_copy <source_db>`

Reply

The system outputs an OK message.

In the event of errors, see Reply Format [Page 12]

Correcting Parameters

Use

You can correct parameter values. In doing so, you have the choice of assigning either the user input **`<user_value>`** or the system default value **`<computed_value>`** **to** the parameter **`<parameter_name>`** registered with `request`.

The value is stored in the parameter manager and becomes effective once the database instance is restarted. Parameters that you have used this command to confirm will only be queried if both the current user input and the system default value do not match the confirmed value.

Prerequisites

You have the DBM operator authorization ParamCheckWrite [Page 19].

You have opened a parameter session [Page 95] and checked parameters.

The check status `request` for a parameter was indicated.

Syntax

`param_putconfirm <parameter_name> <user_value>|<computed_value>`

Reply

The system outputs an OK message.

In the event of errors, see Reply Format [Page 12].

Parameter File List

Use

You request a list of all available parameter files.

Prerequisites

You have the DBM operator authorization ParamRead [Page 20].

Syntax

```
param_versions
```

Reply

```
OK<NL>
<file><NL>
<file><NL>
...
```

`<file>`	Name of the parameter file

In the event of errors, see Reply Format [Page 12].

Parameter File Deletion

Use

You can delete the parameter file.

Use this command with great care.

With the command **param_restore** (Reset the Parameter File to a Previous Version [Page 106]), you can restore the version of the parameter file that was valid before the deletion.

Prerequisites

You have the DBM operator authorization InstallMgm [Page 18].

Syntax

```
param_rmfile
```

Reply

The system outputs an OK message.

In the event of errors, see Reply Format [Page 12].

Parameter Deletion

Use

You can delete the parameter specified as **<parameter_name>** from the parameter file.

The changed values are transferred to the parameter file and are effective when the database instance is restarted.

Use this command with great care.

The **param_directdel** command accesses the database instance parameter file directly. It does not require a parameter session and so **cannot** be rolled back.

Prerequisites

You have the DBM operator authorization ParamFull [Page 19].

Syntax

```
param_directdel <parameter_name>
```

Reply

The system outputs an OK message.

In the event of errors, see Reply Format [Page 12].

Delete Devspace Parameters

Use

You delete the data for the specified devspace from the parameter file.

Use this command with great care.

Inconsistencies in the devspace settings can lead to damage to the database instance.

Prerequisites

You have the DBM operator authorization ParamCheckWrite [Page 19].

Syntax

```
param_deldevspace <dev_no> <dev_mode>
```

Options

<dev_no>	Number of devspace
<dev_mode>	Type of devspace

Option <dev_mode>

SYS	System devspace
MSYS	Mirrored system devspace
DATA	Data devspace; the number of the data devspace is assigned by the system as a four-digit number with leading zeros when setting up the devspace
MDATA	Mirrored data devspace
LOG	Log devspace; the number of the log devspace is assigned by the system as a three-digit number with leading zeros when setting up the devspace
MLOG	Mirrored log devspace (`LOG_MODE=DUAL`)

Reply

The system outputs an OK message.

In the event of errors, see Reply Format [Page 12]

Opening the Parameter History

Use

Open the parameter history and transfer the first block of the contents. The entries are output in reverse chronological sequence, that is, starting with the most recent and ending with the oldest.

You can limit the output by specifying options. If you enter several selection criteria, then only those parameter changes that correspond to all the criteria are output.

If the parameter history entries have been incompletely transferred, you can call up the rest of the contents using the command for scrolling through the parameter history [Page 94].

Prerequisites

You have the DBM operator authorization ParamRead [Page 20].

Syntax

```
param_gethistory [fields=<fieldlist>] [name=<paramname>]
[date=<yyyymmdd>] [group=<grouplist>] [state=<statelist>]
```

Example [Page 103]

Options

<fieldlist>	You can limit the quantity of the parameter history fields supplied by the DBM Server by specifying a field list. Enter the fields that you want output here using a comma as a separator. The fields you select will always be output in the following sequence: Possible entries: **DATE** Date of the parameter change (YYYYMMDD) **TIME** Time when parameter was changed (00HHMMSS) **NAME** Name of the parameter **NEWVALUE** New value of the parameter **OLDVALUE** Old value of the parameter **STATE** The change status of the parameter **GROUP** Group to which the parameter is assigned If you do not specify a field list, then all fields will be output.
<paramname>	You can display the history of one specific parameter by specifying the parameter name. If you do not specify a parameter name, then the entries for all parameters in the parameter history will be displayed.
<yyyymmdd>	If you specify a date, then only those parameter changes that have taken place since this date will be output. If you do not specify a date, then all entries in the parameter history will be supplied.
<grouplist>	You can limit the parameters for which the changes are output by specifying a parameter group. Use a comma to separate entries. Possible entries are: **GENERAL, EXTENDED, SUPPORT** If you do not specify a parameter group, then the parameter changes for all groups will be output.
<statelist>	You can specify the change state to determine whether active, changed or deleted parameters should be output. Use a comma to separate entries. Possible entries are: **A** (active), **C** (changed), **D** (deleted) if you do not specify a state, then all parameter changes will be output.

Reply

```
OK
<headerline><NL>
<paramline><NL>
<paramline><NL>
...
```

Values for the individual fields of the reply

<headerline>	contains the names of the fields displayed S stands for STATE and describes the parameter's change status
<paramline>	contains the values for the fields described in the <headerline>, the values are in the same column position as the field description in the <headerline> Field S for STATE (parameter change state) can contain any of the following entries: c: changed

	d: deleted a: active

param_gethistory Command: Example

Open the parameter history for the parameter _MAXDEVSPACES

```
dbmcli -d DB -u dbm,dbmp param_gethistory name=_MAXDEVSPACES
```

```
OK
DATE     TIME     NAME           NEWVALUE     OLDVALUE        S GROUP
20010613 00152700 _MAXDEVSPACES  5            4               A SUPPORT
20010613 00152700 _MAXDEVSPACES  4            3               C SUPPORT
20010613 00152659 _MAXDEVSPACES  3                            C SUPPORT
```

Check All Parameters

Use

You analyze the whole set of all parameters.

Prerequisites

You have opened a parameter session [Page 95].

You have the DBM operator authorization ParamCheckWrite [Page 19].

Syntax

```
param_checkall [<mode>]
```

Option <mode>

With the parameter **<mode>** you control the behavior of the parameter manager in parameters with the check status request.

Possible values of **<mode>**:

No entry	The system default values are used for all parameters (necessary for background processing).
GENERAL	Only the parameters from the group General - General Database Parameters are requested during the check status request.
EXTENDED	Only the parameters from the groups General - General Database Parameters and Extended - Special Database Parameters are checked.
SUPPORT	This option is only relevant for support. All parameters (from the groups General, Extended and Support) are checked.

Reply

If an error occurs the parameter check is canceled with an error message.

```
ERR
14,ERR_XPCHECK_CN00 : param check failure/request
<parameter_name>      <check_status>
<user_value>
<computed_value>
```

<parameter_name>	Name of parameter
<check_status>	The following check statuses may occur: Mandatory – An obligatory parameter has been assigned a blank value. Constraint – a constraint could not be fulfilled. Request – the user input deviates both from the value transferred to the system and confirmed by it earlier, as well as from the system default value.
<user_value>	Value that the user entered with **param_put**.
<computed_value>	Default value calculated by the system

For open requests the parameter changes cannot be confirmed (parameter change confirmation [Page 92]).

Continuation of Procedure

Correct the parameter in question using **param_putconfirm** (correcting parameters [Page 98]) or **param_put** (parameter value change [Page 92])and carry out a new check with the command **param_checkall [<mode>]**.

The new parameter value is registered by the system, and is transferred to the parameter file as the effective value when the database instance is restarted.

Setting Devspace Parameters

Use

It is preferable that you extend an existing database instance using the command **db_adddevspace** (Adding a Devspace [Page 96]).

Use the command **param_adddevspace** described here only when asked to do so by Support. Inconsistencies in the devspace settings can lead to damage to the database instance.

Use this command to enter the devspace parameters in the parameter file. The new parameter value is registered by the system, and is transferred to the parameter file [Page 78] as the effective value when the database instance is restarted.

You set devspace parameters in two steps:

1. Change the entries for the parameters **in the parameter file**

2. Transfer the corresponding information **to the database kernel** using a utility command (Transferring a Utility Command [Page 166])

If this utility command fails, you must also remove the entries in the parameter file.

No check is made of whether there is sufficient disk capacity available to add devspace.

No check is made of whether the database instance is working in mirrored (DUAL) mode for the relevant devspace.

Prerequisites

You have the DBM operator authorization ParamFull [Page 19].

Syntax

```
param_adddevspace <dev_no> <dev_mode> <dev_name> <dev_type>
<dev_size>
```

Options

<dev_no>	Number of devspace
<dev_mode>	Type of devspace
<dev_name>	Name of devspace/file
<dev_type>	Type of devspace (such as file, raw device)
<dev_size>	Size of devspace

Option <dev_mode>

SYS	System devspace
MSYS	Mirrored system devspace
DATA	Data devspace; the number of the data devspace is assigned by the system as a four-digit number with leading zeros when setting up the devspace
MDATA	Mirrored data devspace
LOG	Log devspace; the number of the log devspace is assigned by the system as a three-digit number with leading zeros when setting up the devspace
MLOG	Mirrored log devspace (`LOG_MODE=DUAL`)

Reply

The system outputs an OK message.

In the event of errors, see Reply Format [Page 12].

Reset the Parameter File to a Previous Version

Use

The parameter file is reset to the version specified by **<version_number>**.

This parameter file will be used when the database instance is next started.

Prerequisites

You have the DBM operator authorization ParamCheckWrite [Page 19].

Syntax

```
param_restore <version_number>
```

Reply

The system outputs an OK message.

In the event of errors, see Reply Format [Page 12].

Listing the DBM Server Commands

Use

You request a list of all DBM Server commands [Page 34] with a short syntax description.

The list of displayed commands can be restricted via the parameter **<command>**. Only those commands are displayed which start with the specified character string.

Prerequisites

You do not need to log on to the DBM Server [Page 12] to execute this DBM Server command [Page 34].

Syntax

```
help [-obsolete] <command>
```

```
d:\v73>dbmcli help autosave

OK
autosave_cancel          (obsolete version of
autolog_cancel)
autosave_off             (obsolete version of autolog_off)
autosave_on              (obsolete version of autolog_on)
autosave_show            (obsolete version of autolog_show)
```

Option

If you specify the **-obsolete** option, only the obsolete commands will be displayed. The current command name is indicated in each case.

If you then enter the **help <command>**, the full syntax and available options for the command are displayed.

```
d:\v73>dbmcli help -obsolete

OK
autosave_cancel          (obsolete version of
autolog_cancel)
autosave_off             (obsolete version of autolog_off)
autosave_on              (obsolete version of autolog_on)
autosave_show            (obsolete version of autolog_show)
backup_save_cancel       (obsolete version of autolog_on)
backup_save_ignore     (obsolete version of backup_ignore)
.....
```

Reply

```
OK<NL>
<commandname> <description><NL>
<commandname> <description><NL>
...
```

In the event of errors, see Reply Format [Page 12].

Backing Up and Recovering Database Instances

Use

You can use the Database Manager CLI to carry out data backups [Extern] (both complete and incremental) and log backups [Extern] and to import these backups if hardware errors occur, thereby restoring the database to a consistent state.

Prerequisites

Note the DBM operator authorizations [Page 14] for each of the DBM Server commands [Page 34].

Commands for Backup Media

Use

A backup medium is assigned to every backup you carry out. Backup media include files, tapes, and pipes.

The path specifications and properties of the tape device, tapes, or files are grouped together in the media definition under a freely selectable, realistic name. Under these names, the media can be re-used for all possible backup operations.

Prerequisites

Note the DBM operator authorizations [Page 14] for each of the DBM Server commands [Page 34].

Syntax

Requesting the Media Data [Page 110]	**medium_get**
Media Definition File Change Date Request [Page 108]	**medium_date**
Defining or Changing Backup Media [Page 108]	**medium_put**
List of All Defined Media [Page 111]	**medium_getall**
Deleting a Medium [Page 113]	**medium_delete**
Transferring an Existing Media Definition [Page 113]	**medium_migrate**

Media Definition File Change Date Request

Use

You request the date on which the current media definition file was last changed.

Prerequisites

You have the DBM operator authorizations Backup [Page 17] and DBInfoRead [Page 15].

Syntax

medium_date

Reply

```
OK<NL>
<YYYYMMDDHHMMSS><NL>
```

In the event of errors, see Reply Format [Page 12].

Defining or Changing Backup Media

Use

You create a new backup medium or change the data for a backup medium that already exists. To do this, you must make the following specifications:

<name>	The medium specified in **<name>** is created or updated. The name of the medium may comprise a group name and a member name. These are separated by an oblique. <name> ::= [<group_name>/]<member_name> This type of group is used with parallel backup or restore operations. If you want to perform a backup or a restore using an external backup tool such as ADSM/TSM, NetWorker, Backint for Oracle, or Backint for SAP DB, ensure the name of the medium starts with ADSM, NSR, BACK, or BACK. In this case, you can only enter **PIPE** under **<type>** (Backing Up with External Backup Tools [Page 124]).
<location>	The name of the device/file with which it can be addressed through operating system functions.
<type>	Type of medium; possible values are: **TAPE** **FILE** **NO-REWIND** **PIPE** **AUTOLOADER** **UNKNOWN**
<backup_type>	The type of backup for which the medium is to be used: **DATA** (complete data backup), **PAGES** (incremental data backup), or **LOG** (log backup)
<size>	Maximum number of pages which can be written to the medium. This is necessary to change the tape correctly, for example. If the backup media is of sufficient size, a zero can be specified.
<block_size>	Number of pages which are transferred when accessing the medium (default: 8).
<overwrite>	This option is only relevant for the medium type **FILE** and describes the behavior if the file is already present. Permitted values are: **NO** **YES**
<autoldr>	Specifies whether the device changes the medium automatically: **YES** **NO**
<oscmd>	Operating system command to be executed before backing up to a succeeding medium [Page 120] or restoring a succeeding medium [Page 153].

Prerequisites

You have the DBM operator authorization Backup [Page 17].

Syntax

```
medium_put <name> <location> <type> <backup_type> <size> <block_size>
<overwrite> <autoldr> <oscmd>
```

Examples: Creating Backup Media [Page 110]

Reply

```
OK<NL>
<name> <type> <location> <size> <overwrite> <block_size><NL>
```

In the event of errors, see Reply Format [Page 12].

medium_put Command: Examples

Defining a File as a Medium for an Interactive Log Backup:

```
dbmcli -d <database_name> -u <userid>,<password> medium_put logsave
/usr/sapdb/log FILE LOG
```

Defining a File as a Medium for an Automatic Log Backup:

```
dbmcli -d <database_name> -u <userid>,<password> medium_put autosave
/usr/sapdb/auto FILE AUTO
```

Defining a File as an Overwritable Medium for a Complete Data Backup:

```
dbmcli -d <database_name> -u <userid>,<password> medium_put completeF
/usr/sapdb/complete FILE DATA 0 0 YES
```

Defining a Tape Device as a Medium for a Complete Data Backup

```
dbmcli -d <database_name> -u <userid>,<password> medium_put completeT
/dev/rft0 TAPE DATA 64000 8 NO
```

Defining a File as a Medium for an Incremental Data Backup:

```
dbmcli -d <database_name> -u <userid>,<password> medium_put incrF
/usr/sapdb/incr FILE PAGES
```

Requesting the Media Data

Use

You request the media data for the backup medium specified in **<name>**.

Prerequisites

You have the DBM operator authorization Backup [Page 17] or DBInfoRead [Page 15].

Syntax

```
medium_get <name>
```

Reply

```
OK<NL>
<name> <location> <type> <backup_type> <size> <block_size>
<overwrite> <autoldr> <oscmd> <date_created> <date_modified> <NL>
```

The fields of the one-line output are separated by tabs.

Values for the individual fields of the reply

<name>	Name to identify the medium. The name of the medium may comprise a group name and a member name. These are separated by an oblique. **<name> ::= [<group_name>/]<member_name>** This type of group is used with parallel backup or restore operations.
<location>	The name of the device/file with which it can be addressed through operating system functions.
<type>	Type of medium; possible values are: **TAPE** **FILE** **NO-REWIND** **PIPE** **AUTOLOADER** **UNKNOWN**
<backup_type>	The type of backup for which the medium is to be used: **DATA, PAGES**, or **LOG**
<size>	Maximum number of pages which can be written to the medium. This is necessary to change the tape correctly, for example. If the media are of sufficient size, a zero can be specified.
<block_size>	Number of pages which are transferred when accessing the medium (default: 8).
<overwrite>	This option is only relevant for the medium type FILE and describes the behavior if the file is already present. Permitted values are: **NO** **YES** **VERSION**
<autoldr>	Specifies whether the device changes the medium automatically: **YES** **NO**
<**oscmd**>	Operating system command to be executed before backing up to a succeeding medium [Page 120] or restoring a succeeding medium [Page 153].
<date_created>	Date on which the medium was created
<date_modified>	Date on which the medium was modified

In the event of errors, see Reply Format [Page 12].

List of All Defined Media

Use

You can call up a list of all defined media.

Prerequisites

You have the DBM operator authorization Backup [Page 17] or DBInfoRead [Page 15].

Syntax

```
medium_getall
```

Reply

```
OK<NL>
<name> <location> <type> <backup_type> <size> <block_size>
<overwrite> <autoldr> <oscmd> <date_created> <date_modified> <NL>

<name> <location> <type> <backup_type> <size> <block_size>
<overwrite> <autoldr> <oscmd> <date_created> <date_modified> <NL>

...
```

Values for the individual fields of the reply

<name>	Name to identify the medium. The name of the medium may comprise a group name and a member name. These are separated by an oblique. **<name> ::= [<grpname>/]<membername>** This type of group is used with parallel backup or restore operations.
<location>	The name of the device/file used for operating system functions.
<type>	Type of medium. Possible values are: **TAPE** **FILE** **NO-REWIND** **PIPE** **AUTOLOADER** **UNKNOWN**
<backup_type>	The type of backup for which the medium is to be used: **DATA, PAGES**, or **LOG**
<size>	Maximum number of pages which can be written to the medium. This is necessary to change the tape correctly, for example. If the media are of sufficient size, a zero can be specified.
<block_size>	Number of pages which are transferred when accessing the medium (default: 8).
<overwrite>	This option is only relevant for the medium type `FILE` and describes the behavior if the file is already present. Permissible values are: **NO** **YES** **VERSION**
<autoldr>	Specifies whether the device changes the medium automatically: **YES** **NO**
<oscmd>	Operating system command to be executed before backing up to a succeeding medium [Page 120] or restoring a succeeding medium [Page 153].

In the event of errors, see Reply Format [Page 12]

Deleting a Medium

Use

You delete the medium specified as **<name>**.

Prerequisites

You have the DBM operator authorization Backup [Page 17].

Syntax

```
medium_delete <name>
```

Reply

The system outputs an OK message.

In the event of errors, see Reply Format [Page 12].

Transferring an Existing Media Definition

Use

You can transfer existing media definitions from the media definition file `control.med` (database version 6.2) to the new media definition file `dbm.mmm` (from database version 7.2.4.2).

see also: Database Files [Page 181]

The media definition from the old file is not transferred if there is an identical media name in the old and new media definition files.

As the old media definitions did not contain a backup type, you should set this using the command **medium_put** (Defining or Changing Backup Media [Page 108]) Otherwise you must specify this when starting the backup process.

Prerequisites

You have the DBM operator authorization Backup [Page 17] or DBInfoRead [Page 15].

Syntax

```
medium_migrate
```

Reply

The system outputs an OK message.

In the event of errors, see Reply Format [Page 12].

Commands for Backups

Use

You can carry out data backups (full and incremental) and log backups with the Database Manager.

For log backups, we recommend that you enable automatic log backup. The option of interactive log backups is also available to you however.

Prerequisites

Note the DBM operator authorizations [Page 14] for each of the DBM Server commands [Page 34].

The prerequisite for executing all other backup commands is to execute the command to backup the database instance [Page 121] first.

Syntax

Terminating the Automatic Log Backup [Page 114]	`autolog_cancel`
Request the Automatic Log Backup Function [Page 115]	`autolog_show`
Current Backup Status Request [Page 115]	`backup_state`
Deactivating the Automatic Log Backup [Page 117]	`autolog_off`
Terminating an Interrupted Backup [Page 118]	`backup_cancel`
Activating the Automatic Log Backup [Page 118]	`autolog_on`
Continue Backup Without Last Known Full Medium [Page 119]	`backup_ignore`
Backing Up to a Succeeding Medium [Page 120]	`backup_replace`
Backing Up the Database Instance [Page 121]	`backup_start`
Backing Up with External Backup Tools [Page 124]	`---`

Terminating the Automatic Log Backup

Use

You cancel an automatic log backup while it is running. This deactivates the *automatic log backup* function. Use this command to prevent technical problems with the backup media, for example.

If you want the log to be backed up automatically again later, then you have to explicitly enable the log backup again (Activating the Automatic Log Backup [Page 118]).

Prerequisites

You have the DBM operator authorization Backup [Page 17].

Syntax

```
autolog_cancel
```

Reply

The system outputs an OK message.

In the event of errors, see Reply Format [Page 12].

Request the Automatic Log Backup Function

Use

You determine whether the automatic log backup function is enabled or disabled.

Prerequisites

You have the DBM operator authorization DBFileRead [Page 17] or DBInfoRead. [Page 15].

Syntax

```
autolog_show
```

```
dbmcli -d <database_name> -u <userid>,<password>
autolog_show
```

```
OK<NL>
AUTOLOG IS OFF <NL>
```

Reply

```
OK<NL>
AUTOLOG IS (OFF | ON)<NL>
```

`AUTOLOG IS ON`	Automatic log backup is enabled
`AUTOLOG IS OFF`	Automatic log backup is disabled

In the event of errors, see Reply Format [Page 12].

Current Backup Status Request

Use

You request the current status of the backup operation.

You can use this request during a database instance backup.

Prerequisites

You have the DBM operator authorization Backup [Page 17].

You have started backing up the database instance [Page 121].

Syntax

backup_state

Reply

```
OK<NL>
Return code               <value><NL>
Date                     [<value>]<NL>
Time                     [<value>]<NL>
Server                   [<value>]<NL>
Database                 [<value>]<NL>
Kernel Version           [<value>]<NL>
Pages Transferred        [<value>]<NL>
Pages Left               [<value>]<NL>
Volume Count             [<value>]<NL>
Medianame                [<value>]<NL>
Location                 [<value>]<NL>
Errortext                [<value>]<NL>
Label                    [<value>]<NL>
Is Consistent            [<value>]<NL>
First LOG Page           [<value>]<NL>
Last LOG Page            [<value>]<NL>
DB Stamp 1 Date          [<value>]<NL>
DB Stamp 1 Time          [<value>]<NL>
DB Stamp 2 Date          [<value>]<NL>
DB Stamp 2 Time          [<value>]<NL>
Page Count               [<value>]<NL>
Devices Used             [<value>]<NL>
Database ID              [<value>]<NL>
Max Used Data Page       [<value>]<NL>
```

Values for the individual fields of the reply

Date	Date
Time	Time
Server	Name of the database server
Database	Name of the database
Kernel Version	Database kernel version
Pages Transferred	Number of pages transferred
Pages Left	Number of pages left
Volumes	Number of backup volumes used
Medianame	Name of backup medium
Location	File or device name
Errortext	Error message text
Label	Backup Label
Is Consistent	Only for data backup: backup is internally consistent
First LOG Page	For data backup: first page of log backup to be read

	For log backup: first page saved in log
`Last LOG Page`	Only for log backup: last page saved in log
`DB Stamp 1 Date` `DB Stamp 1 Time`	Time stamp for first page of log backup
`DB Stamp 2 Date` `DB Stamp 2 Time`	Time stamp for last page of log backup
`Page Count`	Total number of pages saved
`Devices Used`	Number of backup devices used
`Database ID`	Database ID used to identify data and log backups that belong together
`Max Used Data Page`	Maximum number of pages used (indication of minimum database size when backup is imported)

In the event of errors, see Reply Format [Page 12].

Deactivating the Automatic Log Backup

Use

You disable the automatic log backup.

Prerequisites

You have the DBM operator authorization Backup [Page 17].

Syntax

`autolog_off`

```
dbmcli -d <database_name> -u <userid>,<password> autolog_off
OK
```

Reply

The system outputs an OK message.

In the event of errors, see Reply Format [Page 12].

Terminating an Interrupted Backup

Use

You can exit an interrupted backup process completely.

Prerequisites

You have the DBM operator authorization Backup [Page 17].

You have begun backing up the database instance [Page 121]. The backup operation was interrupted.

Syntax

backup_cancel

Reply

In the reply to this command only the field `Return code` is assigned a value.

```
OK<NL>
Return code                <value>
```

In the event of errors, see Reply Format [Page 12].

Activating the Automatic Log Backup

Use

You enable the automatic log backup. For **<medium>**, specify the backup medium on which the log is to be automatically backed up.

Prerequisites

You have the DBM operator authorization Backup [Page 17].

Syntax

autolog_on <medium>

dbmcli -d <database_name> -u <userid>,<password> autolog_on

```
OK
```

Reply

The system outputs an OK message.

In the event of errors, see Reply Format [Page 12].

Continue Backup Without Last Known Full Medium

Use

Use this command to continue a backup without the last medium that was reported full.

Prerequisites

You have the DBM operator authorization Backup [Page 17].

You have started backing up the database instance [Page 121]. The backup operation was interrupted because the backup medium was reported full.

Syntax

backup_ignore

Reply

The reply to this command provides information about the backup supplied by the kernel.

This information is only given when the backup has been ended or interrupted. This command may therefore take a long time to execute.

```
OK<NL>
Return code               <value><NL>
Date                     [<value>]<NL>
Time                     [<value>]<NL>
Server                   [<value>]<NL>
Database                 [<value>]<NL>
Kernel Version           [<value>]<NL>
Pages Transferred        [<value>]<NL>
Pages Left               [<value>]<NL>
Volumn Count             [<value>]<NL>
Medianame                [<value>]<NL>
Location                 [<value>]<NL>
Errortext                [<value>]<NL>
Label                    [<value>]<NL>
Is Consistent            [<value>]<NL>
First LOG Page           [<value>]<NL>
Last LOG Page            [<value>]<NL>
DB Stamp 1 Date          [<value>]<NL>
DB Stamp 1 Time          [<value>]<NL>
DB Stamp 2 Date          [<value>]<NL>
DB Stamp 2 Time          [<value>]<NL>
Page Count               [<value>]<NL>
Devices Used             [<value>]<NL>
```

In particular, analyze the `Return code` reply field, which contains a numeric value supplied by the kernel.

See also: Messages: SAP DB 7.2 and 7.3

If it is clear from the message number that the backup was only interrupted, the session must be continued with the continuation command: (backup_replace [Page 120], backup_ignore [Page 119]), or terminated with backup_cancel [Page 118].

In the event of errors, see Reply Format [Page 12].

Backing Up to a Succeeding Medium

Use

You define a new backup medium. The backup continues to the medium you defined.

Prerequisites

You have the DBM operator authorization Backup [Page 17].

You have started a backup. The backup operation was interrupted because the backup medium was reported as full.

Syntax

backup_replace <medium>

Reply

The reply to this command provides information about the backup supplied by the kernel.

This information is only given when the backup has been ended or interrupted. This command may therefore take a long time to execute.

```
OK<NL>
Return code               <value><NL>
Date                     [<value>]<NL>
Time                     [<value>]<NL>
Server                   [<value>]<NL>
Database                 [<value>]<NL>
Kernel Version           [<value>]<NL>
Pages Transferred        [<value>]<NL>
Pages Left               [<value>]<NL>
Volumn Count             [<value>]<NL>
Medianame                [<value>]<NL>
Location                 [<value>]<NL>
Errortext                [<value>]<NL>
Label                    [<value>]<NL>
Is Consistent            [<value>]<NL>
First LOG Page           [<value>]<NL>
Last LOG Page            [<value>]<NL>
DB Stamp 1 Date          [<value>]<NL>
DB Stamp 1 Time          [<value>]<NL>
DB Stamp 2 Date          [<value>]<NL>
DB Stamp 2 Time          [<value>]<NL>
Page Count               [<value>]<NL>
Devices Used             [<value>]<NL>
Database ID              [<value>]<NL>
Max Used Data Page       [<value>]<NL>
```

In particular, analyze the `Return code` reply field, which contains a numeric value supplied by the kernel.

See also: *Messages: SAP DB 7.2 and 7.3*

If it is clear from the message number that the backup was only interrupted, the session must be continued with the continuation command: (backup_replace [Page 120], backup_ignore [Page 119]), or terminated with backup_cancel [Page 118].

Values for the individual fields of the reply

Date	Date
Time	Time
Server	Name of the database server
Database	Name of the database
Kernel Version	Database kernel version
Pages Transferred	Number of pages transferred
Pages Left	Number of pages left
Volumes	Number of backup volumes used
Medianame	Name of backup medium
Location	File or device name
Errortext	Error message text
Label	Backup Label
Is Consistent	Only for data backup: backup is internally consistent
First LOG Page	For data backup: first page of log backup to be read For log backup: first page saved in log
Last LOG Page	Only for log backup: last page saved in log
DB Stamp 1 Date DB Stamp 1 Time	Time stamp for first page of log backup
DB Stamp 2 Date DB Stamp 2 Time	Time stamp for last page of log backup
Page Count	Total number of pages saved
Devices Used	Number of backup devices used
Database ID	Database ID used to identify data and log backups that belong together
Max Used Data Page	Maximum number of pages used (indication of minimum database size when backup is imported)

In the event of errors, see Reply Format [Page 12].

Backing Up the Database Instance

Use

You save the database instance.

Prerequisites

You have the DBM operator authorization Backup [Page 17].

You have opened a utility session [Page 164].

Syntax

backup_start <medium> [<backup_for>] [<type>] [AUTOIGNORE]

<medium>	Name of the backup media, For a backup on several parallel media specify the name of the media group. This must be defined first (Defining or Changing a Backup Medium [Page 108]).
<backup_for>	**RECOVERY | MIGRATION** Indicate whether the backup shall be made with checkpoint (for migration) or without checkpoint (for recovery). If nothing is specified, the backup is made without checkpoint.
<type>	Type of Backup: **DATA** (Complete data backup), **PAGES** (Incremental data backup) or **LOG** (Log backup)
AUTOIGNORE	The backup is to be automatically continued without the last medium reported as full (Continue Backup Without Last Known Full Medium [Page 119])

If you are using media that is already available, it is vital you follow the current media naming conventions (see Defining or Chaning Backup Media [Page 108]) to avoid error messages.

Reply

The reply to this command provides information about the backup supplied by the kernel.

This information is only given when the backup has been ended or interrupted. This command may therefore take a long time to execute.

```
OK<NL>
Return code               <value><NL>
Date                     [<value>]<NL>
Time                     [<value>]<NL>
Server                   [<value>]<NL>
Database                 [<value>]<NL>
Kernel Version           [<value>]<NL>
Pages Transferred        [<value>]<NL>
Pages Left               [<value>]<NL>
Volumes                  [<value>]<NL>
Medianame                [<value>]<NL>
Location                 [<value>]<NL>
Error text               [<value>]<NL>
Label                    [<value>]<NL>
Is Consistent            [<value>]<NL>
First LOG Page           [<value>]<NL>
Last LOG Page            [<value>]<NL>
DB Stamp 1 Date          [<value>]<NL>
DB Stamp 1 Time          [<value>]<NL>
DB Stamp 2 Date          [<value>]<NL>
DB Stamp 2 Time          [<value>]<NL>
Page Count               [<value>]<NL>
Devices Used             [<value>]<NL>
Database ID              [<value>]<NL>
Max Used Data Page       [<value>]<NL>
```

Values for the individual fields of the reply

Date	Date
Time	Time
Server	Name of the database server
Database	Name of the database
Kernel Version	Database kernel version
Pages Transferred	Number of pages transferred
Pages Left	Number of pages left
Volumes	Number of backup volumes used
Medianame	Name of backup medium
Location	File or device name
Errortext	Error message text
Label	Backup Label
Is Consistent	Only for data backup: backup is internally consistent
First LOG Page	For data backup: first page of log backup to be read For log backup: first page saved in log
Last LOG Page	Only for log backup: last page saved in log
DB Stamp 1 Date DB Stamp 1 Time	Time stamp for first page of log backup
DB Stamp 2 Date DB Stamp 2 Time	Time stamp for last page of log backup
Page Count	Total number of pages saved
Devices Used	Number of backup devices used
Database ID	Database ID used to identify data and log backups that belong together
Max Used Data Page	Maximum number of pages used (indication of minimum database size when backup is imported)

Reply in the Event of an Error

If an error occurs while you are using an external backup tool, you will receive a reply in the following format:

```
ERR<NL>
<errcode>, <err_description><NL>
[<extended_description><NL>]
```

<errcode>	Error message number **See also:** *Messages: SAP DB 7.2 and 7.3*
<err_description>	Description of the error
<extended_description>	Cause of error

The following errors may occur:

<errcode>	<err_description>	Explanation
-24927	ERR_TOOLCHK: the external backup tool was not found	The external backup tool could not be found or has been installed incorrectly.
-24926	ERR_MEDIUMCHK: the medium cannot be used with an external backup tool	The medium specified cannot be used with the backup tool the medium name refers to (Defining and Changing Backup Media [Page 108]).
-24925	ERR_PREPARE: prepare of the backup operation failed	The preparations necessary to use the backup tool were not made correctly.
-24924	ERR_DBREQ: cannot start database kernel request	The database instance was unable to start the backup.
-24923	ERR_TOOLREQ: cannot start external backup tool correctly	The backup tool could not be started correctly.
-24922	ERR_OPCHK: cannot check state of backup operation	Unable to check the status of the database instance or the backup tool.
-24921	ERR_POSTOP: cannot finish backup operation correctly	Although the backup was successful, the post-processing steps required could not be performed.
-24920	ERR_BACKUPOP: backup operation was unsuccessful	The backup failed due to a problem with the database or the backup tool.
-24919	ERR_CLEANUP: cannot clean up correctly after backup operation	Although the backup was successful, the temporary system resources that were used could not be freed up again.

If it is clear from the message number that the backup was only interrupted, the session must be continued with the continuation command: (backup_replace [Page 120], backup_ignore [Page 119]), or terminated with backup_cancel [Page 118].

Backing Up with External Backup Tools

The Database Manager CLI currently supports the use of the following external backup tools:

- ADSM/TSM (IBM(Tivoli)
- Backint for Oracle
- Backint for SAP DB
- NetWorker (Legato)

If you want to use an external backup tool that is not included on this list, please contact Support.

Using the Backup Tools

Start the backup operation directly from the Database Manager CLI. The naming conventions for backup media enable the program to recognize the external backup tool and start it. The pipes for transferring the backup data are created implicitly by the DBM Server or the database in the course of a backup operation. They must exist available beforehand.

Prerequisites

You have the DBM operator authorization Backup [Page 17].

The database instance is in `COLD` or `WARM` operation status. You have defined the media in accordance with the naming conventions for external backup media (Defining or Changing Backup Media [Page 108]).

Procedure

Proceed as described in Backing Up the Database Instance [Page 121]

Saving with ADSM:

```
dbmcli -u dbm,dbm -uUTL -d mydb backup_start ADSMData
recovery data
```

Commands for the Backup History

Use

The system writes a backup history to file `dbm.knl`. This file is stored in the run directory of the database instance (see also: database files [Page 181]).

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

The Database Manager CLI provides you with a number of DBM Server commands [Page 34], with which you can work with the backup history.

Prerequisites

You have the DBM operator authorization Backup [Page 17] or DBInfoRead [Page 15].

Syntax

Backup History Change Date Request [Page 126]	**backup_history_date**
Scrolling Through the Contents of the Backup History [Page 126]	**backup_history_listnext**
Read Backup History [Page 127]	**backup_history_list**
Open Backup History [Page 129]	**backup_history_open**
Close Backup History [Page 130]	**backup_history_close**

Backup History Change Date Request

Use

You can request the date on which the current backup history was last changed.

Prerequisites

You have the DBM operator authorization Backup [Page 17] or DBInfoRead [Page 15].

Syntax

backup_history_date

Reply

```
OK<NL>
<YYYYMMDDHHMMSS><NL>
```

In the event of errors, see Reply Format [Page 12].

Scrolling Through the Contents of the Backup History

Use

You read more of the backup history [Extern]. Options you specified for reading the backup history remain active.

Prerequisites

You have the DBM operator authorization Backup [Page 17] or DBInfoRead [Page 15].

You have opened the backup history [Page 129] and executed the command to read the backup history [Page 129]. Keyword `CONTINUE` in the reply shows that you have not yet read the whole file.

Syntax

backup_history_listnext

Reply

```
OK<NL>
[END|CONTINUE]<NL>
<history_line><NL>
[<media_line><NL>]
[<external_backup_id-line><NL>]
<history_line><NL>
[<media_line><NL>]
[<external_backup_id-line><NL>]
....
```

Values for the individual fields of the reply

END	The contents of the backup history have been transferred in full. The file is closed automatically.
CONTINUE	The backup history contains further entries that were not transferred due to the limited storage available for replies. You can interrogate this data by entering the above command or close the backup history [Page 130].
<history_line>	Information on the completed backups. The individual columns are separated by |. A separate line is output for each backup.
<media_line>	Information on the backup medium [Extern] used for the backup The line is prefixed with an M: and is followed by columns separated by |.
<external_backup_id_line>	If the backup was created using an external backup tool, information on the external backup ID is provided here. You have to specify this information when restoring a backup [Page 147] from an external backup tool. The line is prefixed with E: which, in turn, is followed by columns separated with |.

In the event of errors, see Reply Format [Page 12].

Read Backup History

Use

The system displays the content of the backup history [Extern]. You can restrict or extend what is displayed using options.

Prerequisites

You have the DBM operator authorization Backup [Page 17] or DBInfoRead [Page 15].

You have opened the backup history [Page 129].

Syntax

backup_history_list <options>

Options

-c <columns>	With option **-c** you can restrict the number columns. The keywords for the individual columns are: **KEY** **LABEL** **ACTION** **STAMP1** **STAMP2** **START** **STOP** **FIRSTLOG** **LASTLOG** **LOG** **MEDIA**

	PAGES **VOLUMES** **RC** **ERROR** If several columns are specified, you separate them by commas.
-k <key>	Only the line of the backup history that contains the keyword specified under **key** is displayed. This option cannot be used with the **-r** option.
-l <label>	Only the lines of the backup history that contain the specified backup ID [Extern] are displayed. This option cannot be used with the **-r** option.
-a <action>	Only the lines of the backup history that correspond to the specified backup type are output. This option cannot be used with the **-r** option.
-r	All data backups [Extern] that were required to recover the database instance are displayed. If the log devspace [Extern] is intact, only the data backups that match the available log are displayed.
-r LAST	Beginning with the last complete data backup, all backups that were required to recover the database instance are displayed.
-r <key>	Beginning with the complete data backup specified in **<key>**, all backups that were required to recover the database instance are displayed.
-u <yyyymmddhhmmss>	Only those lines in the backup history that contain data that was saved in the database instance before the specified time are displayed. This option cannot be used with the **-r** option.
-m	For each line that relates to a backup, information is supplied on the backup medium [Extern] used.
-e	For each line that relates to a backup, information is supplied on the relevant external backup ID [Extern].

Reply

```
OK<NL>
(END|CONTINUE)<NL>
<history_line><NL>
[<media_line><NL>]
[<external_backup_id-line><NL>]
<history_line><NL>
[<media_line><NL>]
[<external_backup_id-line><NL>]
....
```

Values for the individual fields of the reply

END	The contents of the backup history have been transferred in full. The file is closed automatically.
CONTINUE	The backup history contains further entries that were not transferred due to the limited storage available for replies. You can interrogate this data

	by entering the above command or close the backup history [Page 130].
<history_line>	Information on the completed backups. The individual columns are separated by |. A separate line is output for each backup.
<media_line>	Information on the medium used for the backup The line is prefixed with an M: and is followed by columns separated by |.
<external_backup_id_line>	If the backup was created using an external backup tool, information on the external backup ID is provided here. You have to specify this information when restoring a backup [Page 147] from an external backup tool. The line is prefixed with E: which, in turn, is followed by columns separated with |.

In the event of errors, see Reply Format [Page 12].

Open Backup History

Use

With this command you open the current backup history. You can add information here on the backup tools used.

To view the content, choose **backup_history_list** (Read Backup History [Page 127]) or **backup_history_listnext** (Scroll Through the Contents of the Backup History [Page 126]).

Prerequisites

You have the DBM operator authorization Backup [Page 17] or DBInfoRead [Page 15].

Syntax

backup_history_open <option>

Option

-e	If you specify the **-e** option, the availability of the respective backups in the external standard tools is added to the backup history information.

Reply

The system outputs an OK message.

In the event of errors, see Reply Format [Page 12].

Close Backup History

Use

You close a backup history that had been opened [Page 129]

Prerequisites

You have the DBM operator authorization Backup [Page 17] or DBInfoRead [Page 15].

Syntax

```
backup_history_close
```

Reply

The system outputs an OK message.

In the event of errors, see Reply Format [Page 12].

Commands for External Backup IDs

Use

Backups created with external backup tools can be identified uniquely by means of an external backup ID [Extern]. This ID is logged in the backup history [Extern] . It can be called up using the commands for the backup history [Page 125] and must be entered when restoring using external backup tools [Page 152].

To enable restores without a backup history, the external backup IDs of the backups currently in the backup tool can, however, also be determined using the DBM Server commands below.

Prerequisites

You have the DBM operator authorization Backup [Page 17] or DBInfoRead [Page 15].

Syntax

Requesting External Backup IDs [Page 131]	backup_ext_ids_get
Scrolling in the External Backup Ids [Page 131]	backup_ext_ids_listnext
Releasing the Memory Occupied by the External Backup IDs [Page 132]	backup_ext_ids_forget
Reading External Backup IDs [Page 132]	backup_ext_ids_list

Requesting External Backup IDs

Use

You request the external backup IDs [Extern] of backups for a database instance [Extern] that are currently registered in the backup tool specified by **<medium>**.

Prerequisites

You have the DBM operator authorization Backup [Page 17] or DBInfoRead [Page 15].

Syntax

backup_ext_ids_get <medium> [<dbname>] [<node>]

<medium>	Name of the backup medium [Extern]; this must be defined beforehand (Defining or Changing Backup Media [Page 108])
<dbname>	Name of the database instance from which the backups were created
<node>	Name of the host on which the backup was created

Reply

The system outputs an OK message.

Use the command **backup_ext_ids_list** (see Reading External Backup IDs [Page 132]) to display the external backup IDs.

In the event of errors, see Reply Format [Page 12].

Scrolling in the External Backup IDs

Use

You display the external backup IDs [Extern] requested by a backup tool which were not transferred because of the limited storage space for replies. This information comprises the availability status, the external backup ID, the backup type, and the creation date/time for each backup.

Prerequisites

You have the DBM operator authorization Backup [Page 17] or DBInfoRead [Page 15].

You have requested the external backup Ids of a backup tool (Requesting External Backup IDs [Page 131]) and executed the command for reading the external backup IDs [Page 132]. Keyword CONTINUE in the reply shows that you have not yet read the whole file.

Syntax

```
backup_ext_ids_listnext
```

Reply

```
OK<NL>
(END|CONTINUE)<NL>
[<external_backup_id_line><NL>]
```

```
[<external_backup_id_line><NL>]
....
```

Values for the individual fields of the reply

END	All of the requested information has been transferred in full.
CONTINUE	Information on further backups is available. This was not transferred because of the limited storage available for replies. You can call up this information by entering the above command again or release the occupied memory [Page 132].
<external_backup_id_line>	A line containing information on one of the backups still registered in the backup tool. The individual columns are separated by a |.

In the event of errors, see Reply Format [Page 12].

Releasing the Memory Occupied by the External Backup IDs

Use

You release the memory in which the external backup IDs [Extern] requested by a backup tool are stored.

Prerequisites

You have the DBM operator authorization Backup [Page 17] or DBInfoRead [Page 15].

Syntax

backup_ext_ids_forget

Reply

The system outputs an OK message.

In the event of errors, see Reply Format [Page 12].

Reading External Backup IDs

Use

You display the information requested by a backup tool. This information comprises the availability status, the external backup ID, the backup type, and the creation date/time for each backup.

Prerequisites

You have the DBM operator authorization Backup [Page 17] or DBInfoRead [Page 15].

You have requested a backup tool's external backup IDs (Requesting External Backup IDs [Page 131]).

Syntax

```
backup_ext_ids_list
```

Reply

```
OK<NL>
(END|CONTINUE)<NL>
[<external_backup_id_line><NL>]
[<external_backup_id_line><NL>]
....
```

Values for the individual fields of the reply

END	All of the information requested by the backup tool was transferred in full.
CONTINUE	Information on further backups is available. This was not transferred because of the limited storage available for replies. You can scroll [Page 131] through this information or release the memory occupied by it [Page 132].
<external_backup_id_line>	A line containing information on one of the backups registered in the backup tool. The individual columns are separated by a |.

In the event of errors, see Reply Format [Page 12].

Commands for Backup Information

Use

Whatever operational state the database instance is in, you can call up the information about backups stored on the backup media. You can use various DBM Server commands [Page 34] to do this.

Prerequisites

You have the DBM operator authorization Backup [Page 17].

Syntax

Backup Information Request (OFFLINE Operation) [Page 134]	**medium_label_offline**
Backup Information Request in WARM or COLD Status [Page 135]	**medium_label**
Requesting the Current Status of a Backup Check [Page 137]	**recover_state_check**
Backup Check [Page 139]	**recover_check**

Backup Information Request (OFFLINE Operation)

Use

You request the information about the backup that is stored on a medium. Access to the backup medium is direct so this function is possible when the database is `OFFLINE`. If the database instance is in `WARM` or `COLD` mode, you use the command for requesting the backup information in WARM or COLD status [Page 135].

Prerequisites

You have the DBM operator authorization Backup [Page 17].

Syntax

```
medium_labeloffline <medium>
```

Reply

```
OK<NL>
Return code               <value><NL>
Date                     [<value>]<NL>
Time                     [<value>]<NL>
Server                   [<value>]<NL>
Database                 [<value>]<NL>
Kernel Version           [<value>]<NL>
Pages Transferred        [<value>]<NL>
Pages Left               [<value>]<NL>
Volume Count             [<value>]<NL>
Medianame                [<value>]<NL>
Location                 [<value>]<NL>
Errortext                [<value>]<NL>
Label                    [<value>]<NL>
Is Consistent            [<value>]<NL>
First LOG Page           [<value>]<NL>
Last LOG Page            [<value>]<NL>
DB Stamp 1 Date          [<value>]<NL>
DB Stamp 1 Time          [<value>]<NL>
DB Stamp 2 Date          [<value>]<NL>
DB Stamp 2 Time          [<value>]<NL>
Page Count               [<value>]<NL>
Devices Used             [<value>]<NL>
Database ID              [<value>]<NL>
Max Used Data Page       [<value>]<NL>
```

In particular, analyze the `Return code` reply field, which contains a numeric value supplied by the kernel.

See also: *Messages: SAP DB 7.2 and 7.3*

Values for the individual fields of the reply

`Date`	Date
`Time`	Time
`Server`	Name of the database server
`Database`	Name of the database
`Kernel Version`	Database kernel version

`Pages Transferred`	Number of pages transferred
`Pages Left`	Number of pages left
`Volumes`	Number of backup volumes used
`Medianame`	Name of backup medium
`Location`	File or device name
`Errortext`	Error message text
`Label`	Backup Label
`Is Consistent`	Only for data backup: backup is internally consistent
`First LOG Page`	For data backup: first page of log backup to be read For log backup: first page saved in log
`Last LOG Page`	Only for log backup: last page saved in log
`DB Stamp 1 Date` `DB Stamp 1 Time`	Time stamp for first page of log backup
`DB Stamp 2 Date` `DB Stamp 2 Time`	Time stamp for last page of log backup
`Page Count`	Total number of pages saved
`Devices Used`	Number of backup devices used
`Database ID`	Database ID used to identify data and log backups that belong together
`Max Used Data Page`	Maximum number of pages used (indication of minimum database size when backup is imported)

In the event of errors, see Reply Format [Page 12].

Backup Information Request in WARM or COLD Status

Use

You request the information about the backup that is stored on a medium.

Prerequisites

You have the DBM operator authorization Backup [Page 17].

You have started a utility session (Opening a Utility Session [Page 164]). The database instance is in `WARM` or `COLD` operation status.

Syntax

`medium_label <medium>`

You use the **`<medium>`** parameter to specify which backup medium the information is to be read in from.

Only the name of the media group has to be specified for parallel media. You must first define your media group (Defining or Changing Backup Media [Page 108]).

Reply

```
OK<NL>
Return code               <value><NL>
Date                     [<value>]<NL>
Time                     [<value>]<NL>
Server                   [<value>]<NL>
Database                 [<value>]<NL>
Kernel Version           [<value>]<NL>
Pages Transferred        [<value>]<NL>
Pages Left               [<value>]<NL>
Volumes                  [<value>]<NL>
Medianame                [<value>]<NL>
Location                 [<value>]<NL>
Errortext                [<value>]<NL>
Label                    [<value>]<NL>
Is Consistent            [<value>]<NL>
First LOG Page           [<value>]<NL>
Last LOG Page            [<value>]<NL>
DB Stamp 1 Date          [<value>]<NL>
DB Stamp 1 Time          [<value>]<NL>
DB Stamp 2 Date          [<value>]<NL>
DB Stamp 2 Time          [<value>]<NL>
Page Count               [<value>]<NL>
Devices Used             [<value>]<NL>
Database ID              [<value>]<NL>
Max Used Data Page       [<value>]<NL>
```

 In particular, analyze the `Return code` reply field, which contains a numeric value supplied by the kernel.

See also: *Messages: SAP DB 7.2 and 7.3*

Values for the individual fields of the reply

`Date`	Date
`Time`	Time
`Server`	Name of the database server
`Database`	Name of the database
`Kernel Version`	Database kernel version
`Pages Transferred`	Number of pages transferred
`Pages Left`	Number of pages left
`Volumes`	Number of backup volumes used
`Medianame`	Name of backup medium
`Location`	File or device name
`Errortext`	Error message text
`Label`	Backup Label
`Is Consistent`	Only for data backup: backup is internally consistent
`First LOG Page`	For data backup: first page of log backup to be read

	For log backup: first page saved in log
`Last LOG Page`	Only for log backup: last page saved in log
`DB Stamp 1 Date` `DB Stamp 1 Time`	Time stamp for first page of log backup
`DB Stamp 2 Date` `DB Stamp 2 Time`	Time stamp for last page of log backup
`Page Count`	Total number of pages saved
`Devices Used`	Number of backup devices used
`Database ID`	Database ID used to identify data and log backups that belong together
`Max Used Data Page`	Maximum number of pages used (indication of minimum database size when backup is imported)

In the event of errors, see Reply Format [Page 12].

Requesting the Current Status of a Backup Check

Use

You request information about the current status of a backup check. You can run this command while a backup is being checked.

Prerequisites

You have the DBM operator authorization Backup [Page 17].

You have opened a service session (Opening a Service Session [Page 174]).

Syntax

```
recover_state_check
```

Reply

```
OK<NL>
Return code               <value><NL>
Date                     [<value>]<NL>
Time                     [<value>]<NL>
Server                   [<value>]<NL>
Database                 [<value>]<NL>
Kernel Version           [<value>]<NL>
Pages Transferred        [<value>]<NL>
Pages Left               [<value>]<NL>
Volumes                  [<value>]<NL>
Medianame                [<value>]<NL>
Location                 [<value>]<NL>
Error text               [<value>]<NL>
Label                    [<value>]<NL>
Is Consistent            [<value>]<NL>
First LOG Page           [<value>]<NL>
Last LOG Page            [<value>]<NL>
```

```
DB Stamp 1 Date          [<value>]<NL>
DB Stamp 1 Time          [<value>]<NL>
DB Stamp 2 Date          [<value>]<NL>
DB Stamp 2 Time          [<value>]<NL>
Page Count               [<value>]<NL>
Devices Used             [<value>]<NL>
Database ID              [<value>]<NL>
Max Used Data Page       [<value>]<NL>
```

Values for the individual fields of the reply

`Date`	Date
`Time`	Time
`Server`	Name of the database server
`Database`	Name of the database
`Kernel Version`	Database kernel version
`Pages Transferred`	Number of pages transferred
`Pages Left`	Number of pages left
`Volumes`	Number of backup volumes used
`Medianame`	Name of backup medium
`Location`	File or device name
`Errortext`	Error message text
`Label`	Backup Label
`Is Consistent`	Only for data backup: backup is internally consistent
`First LOG Page`	For data backup: first page of log backup to be read For log backup: first page saved in log
`Last LOG Page`	Only for log backup: last page saved in log
`DB Stamp 1 Date` `DB Stamp 1 Time`	Time stamp for first page of log backup
`DB Stamp 2 Date` `DB Stamp 2 Time`	Time stamp for last page of log backup
`Page Count`	Total number of pages saved
`Devices Used`	Number of backup devices used
`Database ID`	Database ID used to identify data and log backups that belong together
`Max Used Data Page`	Highest page number assigned (indication of minimum database size when backup is imported)

In the event of errors, see Reply Format [Page 12].

Backup Check

Use

You check that the backup is consistent and complete. This command is executed independently of the current database instance functions but can adversely impact database instance performance temporarily.

Prerequisites

You have the DBM operator authorization Backup [Page 17].

You have opened a service session [Page 174].

Syntax

```
recover_check <medium> <type> [ExternalBackupID <external_backup_ID>]
[<nnn>] [UNTIL <date> <time>]
```

<medium>	Name of the backup medium [Extern]; this must be defined beforehand (Defining or Changing Backup Media [Page 108])
<type>	Type of Backup: **DATA** (Complete data backup [Extern]), **PAGES** (Incremental data backup) or **LOG** (Log Backup [Extern])

Options

ExternalBackupID <external_backup_ID>	To check a backup created with an external backup tool, enter a backup ID name that the external tool will recognize.
<nnn>	Actual backup version on the backup media that is being checked; relevant only for media of type **FILE**.
UNTIL <date> <time>	For log backups you can enter an exact time up to which the log backups are to be checked.

The backup specified is just checked, not restored.

Reply

```
OK<NL>
Returncode                <value><NL>
Date                      [<value>]<NL>
Time                      [<value>]<NL>
Server                    [<value>]<NL>
Database                  [<value>]<NL>
Kernel Version            [<value>]<NL>
Pages Transferred         [<value>]<NL>
Pages Left                [<value>]<NL>
Volume Count              [<value>]<NL>
Mediumname                [<value>]<NL>
Location                  [<value>]<NL>
Errortext                 [<value>]<NL>
Label                     [<value>]<NL>
Is Consistent             [<value>]<NL>
First LOG Page            [<value>]<NL>
Last LOG Page             [<value>]<NL>
DB Stamp 1 Date           [<value>]<NL>
DB Stamp 1 Time           [<value>]<NL>
DB Stamp 2 Date           [<value>]<NL>
DB Stamp 2 Time           [<value>]<NL>
```

```
Page Count              [<value>]<NL>
Devices Used            [<value>]<NL>
Database ID             [<value>]<NL>
Max Used Data Page      [<value>]<NL>
```

Values for the individual fields of the reply

Date	Date
Time	Time
Server	Name of the database server
Database	Name of the database
Kernel Version	Database kernel version
Pages Transferred	Number of pages transferred
Pages Left	Number of pages left
Volumes	Number of backup volumes used
Medianame	Name of backup medium
Location	File or device name
Errortext	Error message text
Label	Backup Label [Extern]
Is Consistent	Only for data backup: backup is internally consistent
First LOG Page	For data backup: first page of log backup to be read For log backup: first page saved in log
Last LOG Page	Only for log backup: last page saved in log
DB Stamp 1 Date DB Stamp 1 Time	Time stamp for first page of log backup
DB Stamp 2 Date DB Stamp 2 Time	Time stamp for last page of log backup
Page Count	Total number of pages saved
Devices Used	Number of backup devices used
Database ID	Database ID used to identify data and log backups that belong together
Max Used Data Page	Maximum number of pages used (indication of minimum database size when backup is imported)

Reply in the Event of an Error

If an error occurs during the process, you will receive a reply in the following format:

```
ERR<NL>
<errcode>, <err_description><NL>
[<extended_description><NL>]
```

<errcode>	Error message number **See also**: *Messages: SAP DB 7.2 and 7.3*
<err_description>	Description of the error
<extended_description>	Cause of error

The following errors may occur:

<errcode>	<err_description>	Explanation
-24927	ERR_TOOLCHK: the external backup tool was not found	The external backup tool could not be found or has been installed incorrectly.
-24926	ERR_MEDIUMCHK: the medium cannot be used with an external backup tool	The medium specified cannot be used with the backup tool the medium name refers to (Defining and Changing Media [Page 108]).
-24925	ERR_PREPARE: prepare of the backup operation failed	The preparations necessary to use the backup tool were not made correctly.
-24924	ERR_DBREQ: cannot start database kernel request	The database instance was unable to start the check.
-24923	ERR_TOOLREQ: cannot start external backup tool correctly	The backup tool could not be started correctly.
-24922	ERR_OPCHK: cannot check state of backup operation	Unable to check the status of the database instance or the backup tool.
-24921	ERR_POSTOP: cannot finish backup operation correctly	Although the backup was successful, the post-processing steps required could not be performed.
-24920	ERR_BACKUPOP: backup operation was unsuccessful	The check failed due to a problem with the database or the backup tool.
-24919	ERR_CLEANUP: cannot clean up correctly after backup operation	Although the check was successful, the temporary system resources that were used could not be freed up again.

Commands for Restoring

Use

The Database Manager supports the restore of the database instance after hardware faults. The last state of the database is restored. A prerequisite for this is that all the data in the backup history is available.

To obtain the highest possible throughput of data, full and incremental data backups can be imported from a number of backup media simultaneously. The number of parallel media used for the restore does not depend on the number of parallel media used to make the backup originally. Even a backup made to single medium plus continuation media can be imported in parallel.

You define the maximum number of media that can be imported concurrently using the parameter MAX_BACKUP_DEVS. The use of up to 32 tape devices allows you to reduce backup and restore times considerably.

Prerequisites

Note the DBM operator authorizations [Page 14] for each of the DBM Server commands [Page 34].

The prerequisite for executing all other restore commands is to first execute the command for restoring a database instance [Page 147].

Syntax

Database Instance Current Information Request [Page 142]	**db_restartinfo**
Current Recovery Status Request [Page 143]	**recover_state**
Terminating an Interrupted Restore or Backup Check [Page 145]	**recover_cancel**
Continue Parallel Restore or Backup Check [Page 145]	**recover_ignore**
Restoring a Database Instance [Page 147]	**recover_start**
Restoring the Parameter File from a Data Backup [Page 150]	**recover_config**
Recovering a Damaged Devspace [Page 151]	**recover_devspace**
Restoring with External Backup Tools [Page 152]	**---**
Restore or Check a Backup with Succeeding Medium [Page 153]	**recover_replace**
Restoring a Damaged Index [Page 155]	**sql_recreate_index**

Database Instance Current Information Request

Use

You request the current status of a database instance after importing a backup. You receive information on the state of the log and the ability of the database instance to restart.

Prerequisites

You have the DBM operator authorization Recovery [Page 21].

You have executed the command for restoring a database instance [Page 147].

Syntax

db_restartinfo

```
dbmcli -d DB -u dbm,dbm db_restartinfo

OK
Used LOG Page           10636
First LOG Page          8640
Restartable             1
Id Restart Record       P46643:DB_20010629_115558
```

```
        Id LOG Info             P46643:DB_20010629_115558
        Consistent              1
```

Reply

```
OK<NL>
Used LOG Page           <number><NL>
First LOG Page          <number><NL>
Restartable             (1|0)<NL>
Id Restart Record       <id><NL>
Id LOG Info             <id><NL>
Consistent              (1|0)<NL>
```

Values for the individual fields of the reply

`Used LOG Page`	Next log page to be used after restarting
`First LOG Page`	Oldest available log page
`Restartable`	Restartable (1 = yes, 0 = no)
`Id Restart Record`	Identifier of the restart record for the database instance
`Id LOG Info`	Identifier for the database instance log
`Consistent`	Consistency of the database instance, that is, can be restarted without log (`1` = yes, `0` = no)

In the event of errors, see Reply Format [Page 12].

Current Recovery Status Request

Use

You request information about the current status of the restore. You can do this while a backup is being restored.

Prerequisites

You have the DBM operator authorization Recovery [Page 21].

You have executed the command for restoring a database instance [Page 147]. The operation was interrupted.

Syntax

```
recover_state
```

Reply

```
OK<NL>
Return code              <value><NL>
Date                    [<value>]<NL>
Time                    [<value>]<NL>
Server                  [<value>]<NL>
Database                [<value>]<NL>
Kernel Version          [<value>]<NL>
Pages Transferred       [<value>]<NL>
```

```
Pages Left              [<value>]<NL>
Volumes                 [<value>]<NL>
Medianame               [<value>]<NL>
Location                [<value>]<NL>
Error text              [<value>]<NL>
Label                   [<value>]<NL>
Is Consistent           [<value>]<NL>
First LOG Page          [<value>]<NL>
Last LOG Page           [<value>]<NL>
DB Stamp 1 Date         [<value>]<NL>
DB Stamp 1 Time         [<value>]<NL>
DB Stamp 2 Date         [<value>]<NL>
DB Stamp 2 Time         [<value>]<NL>
Page Count              [<value>]<NL>
Devices Used            [<value>]<NL>
Database ID             [<value>]<NL>
Max Used Data Page      [<value>]<NL>
```

Values for the individual fields of the reply

`Date`	Date
`Time`	Time
`Server`	Name of the database server
`Database`	Name of the database
`Kernel Version`	Database kernel version
`Pages Transferred`	Number of pages transferred
`Pages Left`	Number of pages left
`Volumes`	Number of backup volumes used
`Medianame`	Name of backup medium
`Location`	File or device name
`Errortext`	Error message text
`Label`	Backup Label
`Is Consistent`	Only for data backup: backup is internally consistent
`First LOG Page`	For data backup: first page of log backup to be read For log backup: first page saved in log
`Last LOG Page`	Only for log backup: last page saved in log
`DB Stamp 1 Date` `DB Stamp 1 Time`	Time stamp for first page of log backup
`DB Stamp 2 Date` `DB Stamp 2 Time`	Time stamp for last page of log backup
`Page Count`	Total number of pages saved
`Devices Used`	Number of backup devices used
`Database ID`	Database ID used to identify data and log backups that belong together
`Max Used Data Page`	Highest page number assigned (indication of minimum database size when backup is imported)

In the event of errors, see Reply Format [Page 12].

Terminating an Interrupted Restore or a Backup Check

Use

You definitively terminate an interrupted restore or a backup check.

Prerequisites

You have the DBM operator authorization Backup [Page 17].

You have executed the command for restoring a database instance [Page 147]. The operation was interrupted.

Syntax

```
recover_cancel
```

Reply

In the reply to this command only the field `Return code` is assigned a value.

```
OK<NL>
Return code                <value>
```

See also: *Messages: SAP DB 7.2 and 7.3*

In the event of errors, see Reply Format [Page 12].

Continue Parallel Restore or Backup Check

Use

You continue reading in from the other backup media. As a reply to this command, you receive the information through the import of the backup. The output is, however, only given when the backup has been completely read in or is interrupted. This command may therefore take a long time to execute.

Prerequisites

You have the DBM operator authorization Backup [Page 17].

You have started restoring a database instance [Page 147] by reading data from several media simultaneously. The procedure was interrupted because at least one medium no longer has a succeeding medium.

Syntax

```
recover_ignore
```

Reply

```
OK<NL>
Return code               <value><NL>
Date                     [<value>]<NL>
Time                     [<value>]<NL>
Server                   [<value>]<NL>
Database                 [<value>]<NL>
Kernel Version           [<value>]<NL>
Pages Transferred        [<value>]<NL>
Pages Left               [<value>]<NL>
Volumes                  [<value>]<NL>
Medianame                [<value>]<NL>
Location                 [<value>]<NL>
Errortext                [<value>]<NL>
Label                    [<value>]<NL>
Is Consistent            [<value>]<NL>
First LOG Page           [<value>]<NL>
Last LOG Page            [<value>]<NL>
DB Stamp 1 Date          [<value>]<NL>
DB Stamp 1 Time          [<value>]<NL>
DB Stamp 2 Date          [<value>]<NL>
DB Stamp 2 Time          [<value>]<NL>
Page Count               [<value>]<NL>
Devices Used             [<value>]<NL>
Database ID              [<value>]<NL>
Max Used Data Page       [<value>]<NL>
```

 In particular, analyze the `Return code` reply field, which contains a numeric value supplied by the kernel.

See also: *Messages: SAP DB 7.2 and 7.3*

If it is clear from the message number that the recovery was only interrupted, the session must be continued with the continuation command (**`recover_ignore`**) or terminated (recover_cancel [Page 145]).

Values for the individual fields of the reply

`Date`	Date
`Time`	Time
`Server`	Name of the database server
`Database`	Name of the database
`Kernel Version`	Database kernel version
`Pages Transferred`	Number of pages transferred
`Pages Left`	Number of pages left
`Volumes`	Number of backup volumes used
`Medianame`	Name of backup medium
`Location`	File or device name
`Errortext`	Error message text
`Label`	Backup Label
`Is Consistent`	Only for data backup: backup is internally consistent
`First LOG Page`	For data backup: first page of log backup to be read

	For log backup: first page saved in log
`Last LOG Page`	Only for log backup: last page saved in log
`DB Stamp 1 Date` `DB Stamp 1 Time`	Time stamp for first page of log backup
`DB Stamp 2 Date` `DB Stamp 2 Time`	Time stamp for last page of log backup
`Page Count`	Total number of pages saved
`Devices Used`	Number of backup devices used
`Database ID`	Database ID used to identify data and log backups that belong together
`Max Used Data Page`	Maximum number of pages used (indication of minimum database size when backup is imported)

In the event of errors, see Reply Format [Page 12].

Restoring a Database Instance

Use

Use this command to read a backup. As a reply to this command, you receive the information through the import of the backup. The output is, however, only given when the backup has been completely read in or is interrupted. This command may therefore take a long time to execute.

Prerequisites

You have the DBM operator authorization Recovery [Page 21].

Syntax

`recover_start <medium> <type> [ExternalBackupID <external_backup_ID>] [<nnn>] [UNTIL <date> <time>] [AUTOIGNORE]`

`<medium>`	Medium from which the backup is to be read. When restoring a backup made on several media at the same time, you must enter the name of the backup media group here.
`<type>`	Type of backup to be read: **`DATA`** (Complete data backup), **`PAGES`** (Incremental data backup) or **`LOG`** (Log backup)
`ExternalBackupID <external_backup_ID>`	To restore a backup created with an external backup tool, enter a backup ID name that the external tool will recognize.
`<nnn>`	Actual backup version on the backup carrier that is to be read; relevant only for media of type **`FILE`**.
`UNTIL <date> <time>`	For log backups you can enter an exact time up to which the log backups are to be read.
`AUTOIGNORE`	In the cases of a parallel restore, the process is automatically continued by the system (Continue Parallel Restore or Backup

	Check [Page 145]).

It is vital you follow the current media naming conventions (see Defining or Changing Backup Media [Page 108]) to avoid error messages.

Reply

```
OK<NL>
Return code                <value><NL>
Date                      [<value>]<NL>
Time                      [<value>]<NL>
Server                    [<value>]<NL>
Database                  [<value>]<NL>
Kernel Version            [<value>]<NL>
Pages Transferred         [<value>]<NL>
Pages Left                [<value>]<NL>
Volumes                   [<value>]<NL>
Medianame                 [<value>]<NL>
Location                  [<value>]<NL>
Error text                [<value>]<NL>
Label                     [<value>]<NL>
Is Consistent             [<value>]<NL>
First LOG Page            [<value>]<NL>
Last LOG Page             [<value>]<NL>
DB Stamp 1 Date           [<value>]<NL>
DB Stamp 1 Time           [<value>]<NL>
DB Stamp 2 Date           [<value>]<NL>
DB Stamp 2 Time           [<value>]<NL>
Page Count                [<value>]<NL>
Devices Used              [<value>]<NL>
Database ID               [<value>]<NL>
Max Used Data Page        [<value>]<NL>
```

Values for the individual fields of the reply

Date	Date
Time	Time
Server	Name of the database server
Database	Name of the database
Kernel Version	Database kernel version
Pages Transferred	Number of pages transferred
Pages Left	Number of pages left
Volumes	Number of backup volumes used
Medianame	Name of backup medium
Location	File or device name
Errortext	Error message text
Label	Backup Label
Is Consistent	Only for data backup: backup is internally consistent
First LOG Page	For data backup: first page of log backup to be read For log backup: first page saved in log

Last LOG Page	Only for log backup: last page saved in log
DB Stamp 1 Date DB Stamp 1 Time	Time stamp for first page of log backup
DB Stamp 2 Date DB Stamp 2 Time	Time stamp for last page of log backup
Page Count	Total number of pages saved
Devices Used	Number of backup devices used
Database ID	Database ID used to identify data and log backups that belong together
Max Used Data Page	Maximum number of pages used (indication of minimum database size when backup is imported)

Reply in the Event of an Error

If an error occurs while you are using an external backup tool, you will receive a reply in the following format:

```
ERR<NL>
<errcode>, <err_description><NL>
[<extended_description><NL>]
```

<errcode>	Error message number **See also:** *Messages: SAP DB 7.2 and 7.3*
<err_description>	Description of the error
<extended_description>	Cause of error

The following errors may occur:

<errcode>	**<err_description>**	**Explanation**
-24927	ERR_TOOLCHK: the external backup tool was not found	The external backup tool could not be found or has been installed incorrectly.
-24926	ERR_MEDIUMCHK: the medium cannot be used with an external backup tool	The medium specified cannot be used with the backup tool the medium name refers to (Defining and Changing Backup Media [Page 108]).
-24925	ERR_PREPARE: prepare of the backup operation failed	The preparations necessary to use the backup tool were not made correctly.
-24924	ERR_DBREQ: cannot start database kernel request	The database instance was unable to start the restore operation.
-24923	ERR_TOOLREQ: cannot start external backup tool correctly	The backup tool could not be started correctly.
-24922	ERR_OPCHK: cannot check state of backup operation	Unable to check the status of the database instance or the backup tool.
-24921	ERR_POSTOP: cannot finish backup operation correctly	Although the recovery was successful, the post-processing steps required could not be performed.
-24920	ERR_BACKUPOP: backup operation was unsuccessful	The recovery failed due to a problem with the database or the backup tool.

-24919	ERR_CLEANUP: cannot clean up correctly after backup operation	Although the recovery was successful, the temporary system resources used could not be freed up again.

Restoring the Parameter File from a Data Backup

Use

You recover the parameter file from the data backup stored on a medium.

All parameters except for SERVERDB are taken over. The parameter SERVERDB is converted to the value of the current database instance.

The parameter file is stored under the name of the current database instance.

Prerequisites

You have the DBM operator authorization Recovery [Page 21].

You have opened a service session [Page 174].

You have executed the command for restoring a database instance [Page 147].

Syntax

```
recover_config <medium>
```

Reply

```
OK<NL>
Return code               <value><NL>
Date                     [<value>]<NL>
Time                     [<value>]<NL>
Server                   [<value>]<NL>
Database                 [<value>]<NL>
Kernel Version           [<value>]<NL>
Pages Transferred        [<value>]<NL>
Pages Left               [<value>]<NL>
Volumn Count             [<value>]<NL>
Medianame                [<value>]<NL>
Location                 [<value>]<NL>
Error text               [<value>]<NL>
Label                    [<value>]<NL>
Is Consistent            [<value>]<NL>
First LOG Page           [<value>]<NL>
Last LOG Page            [<value>]<NL>
DB Stamp 1 Date          [<value>]<NL>
DB Stamp 1 Time          [<value>]<NL>
DB Stamp 2 Date          [<value>]<NL>
DB Stamp 2 Time          [<value>]<NL>
Page Count               [<value>]<NL>
Devices Used             [<value>]<NL>
Database ID              [<value>]<NL>
Max Used Data Page       [<value>]<NL>
```

Values for the individual fields of the reply

Date	Date
Time	Time
Server	Name of the database server
Database	Name of the database
Kernel Version	Database kernel version
Pages Transferred	Number of pages transferred
Pages Left	Number of pages left
Volumes	Number of backup volumes used
Medianame	Name of backup medium
Location	File or device name
Errortext	Error message text
Label	Backup Label
Is Consistent	Only for data backup: backup is internally consistent
First LOG Page	For data backup: first page of log backup to be read For log backup: first page saved in log
Last LOG Page	Only for log backup: last page saved in log
DB Stamp 1 Date DB Stamp 1 Time	Time stamp for first page of log backup
DB Stamp 2 Date DB Stamp 2 Time	Time stamp for last page of log backup
Page Count	Total number of pages saved
Devices Used	Number of backup devices used
Database ID	Database ID used to identify data and log backups that belong together
Max Used Data Page	Maximum number of pages used (indication of minimum database size when backup is imported)

In the event of errors, see Reply Format [Page 12].

Recovering a Damaged Devspace

Use

If you are running the log spaces of a database instance in mirrored form, and a log devspace fails while the database is running, there are functions available that enable you to recover the failed devspace.

Entries that were written to the intact devspace only while the damaged devspace was out of action are not copied.

Note that the mirrored log devspace is only secure once the recovered devspace is operating again. You should therefore backup the log entries that only exist once using a log backup (Backing Up the Database Instance [Page 121]).

Prerequisites

You have the DBM operator authorization Recovery [Page 21].

You are operation the database instance with mirrored log devspaces. (`LOG_MODE: DUAL`).

You have opened a utility session [Page 164].

You have executed the command for restoring a database instance [Page 147].

Syntax

```
recover_devspace <dev_no> [<dev_mode>]
```

<dev_no>	Number of devspace
<dev_mode>	Here, **LOG** is the only possible entry.

This command returns the damaged devspace of a devspace pair to operation.

Reply

The system outputs an OK message.

In the event of errors, see Reply Format [Page 12].

Restoring with External Backup Tools

If you want to use an external backup tool that is not included on this list, please contact Support.

The Database Manager CLI supports the use of the following external backup tools:

- ADSM/TSM (IBM/Tivoli)
- Backint for Oracle
- Backint SAP DB
- NetWorker (Legato)

Using the Backup Tools

Start the operation for restoring the database instance [Page 147] from the Database manager CLI. The naming conventions for external backup media [Extern] enable the program to recognize the external backup tool and start it.

Procedure

First determine the external backup ID [Extern] for the required backup (Requesting External Backup IDs [Page 131], Reading External Backup IDs [Page 132]). The Database Manager uses this ID to identify the backup that is to be restored.

The number of media in a group of parallel media [Extern] must correspond to the number of media used to create the backup.

Proceed as described in Restoring the Database Instance [Page 147]

Restoring using NetWorker:

```
dbmcli -u dbm,dbm -uUTL -d mydb recover_start NSRPages pages
ExternalBackupID 9025
```

Restoring or Checking a Backup with Succeeding Medium

Use

You specify the next medium from which reading in the backup is to continue. As a reply to this command, you receive the information through the import of the backup. The output is, however, only given when the backup has been completely read in or is interrupted. This command may therefore take a long time to execute.

Prerequisites

You have the DBM operator authorization Backup [Page 17].

You have executed the command for restoring a database instance [Page 147]. The procedure was interrupted because the backup has finished reading in one backup medium [Extern].

Syntax

recover_replace <medium> [ExternalBackupID <ext_backup_ID>] [<loc> [<nnn>]]

<medium>	Medium from which the backup is to be read. When restoring a backup made on several media at the same time, you must enter the name of the group of parallel backup media [Extern] here.

Options

ExternalBackupID <external_backup_ID>	To restore a backup created with an external backup tool, enter a backup ID name that the external tool will recognize.
<loc>	Name of the device/the file in which the backup to be restored is stored. Operating system functions must be able to address the device/file using **<loc>.**
<nnn>	Actual backup version on the backup carrier that is being checked; relevant only for media of type **FILE**.

It is vital you follow the current media naming conventions (see Creating or Changing a Medium [Page 108]) to avoid error messages.

Reply

```
OK<NL>
Returncode                <value><NL>
Date                      [<value>]<NL>
Time                      [<value>]<NL>
Server                    [<value>]<NL>
Database                  [<value>]<NL>
Kernel Version            [<value>]<NL>
Pages Transferred         [<value>]<NL>
Pages Left                [<value>]<NL>
Volume Count              [<value>]<NL>
Medianame                 [<value>]<NL>
Location                  [<value>]<NL>
Errortext                 [<value>]<NL>
Label                     [<value>]<NL>
Is Consistent             [<value>]<NL>
First LOG Page            [<value>]<NL>
Last LOG Page             [<value>]<NL>
DB Stamp 1 Date           [<value>]<NL>
DB Stamp 1 Time           [<value>]<NL>
DB Stamp 2 Date           [<value>]<NL>
DB Stamp 2 Time           [<value>]<NL>
Page Count                [<value>]<NL>
Devices Used              [<value>]<NL>
Database ID               [<value>]<NL>
Max Used Data Page        [<value>]<NL>
```

In particular, analyze the `Returncode` reply field, which contains a numeric value supplied by the kernel.

See also: *Messages: SAP DB 7.2 and 7.3*

If it is clear from the message number that the recovery was only interrupted, the session must be continued with the continuation command (**`recover_replace`**) or terminated (recover_cancel [Page 145]).

Values for the individual fields of the reply

`Date`	Date
`Time`	Time
`Server`	Name of the database server
`Database`	Name of the database
`Kernel Version`	Database kernel version
`Pages Transferred`	Number of pages transferred
`Pages Left`	Number of pages left
`Volumes`	Number of backup volumes used
`Medianame`	Name of backup medium
`Location`	File or device name
`Errortext`	Error message text
`Label`	Backup Label [Extern]
`Is Consistent`	Only for data backup [Extern]: backup is internally consistent

`First LOG Page`	For data backup: first page of log backup [Extern] to be read For log backup: first page saved in log
`Last LOG Page`	Only for log backup: last page saved in log
`DB Stamp 1 Date` `DB Stamp 1 Time`	Time stamp for first page of log backup
`DB Stamp 2 Date` `DB Stamp 2 Time`	Time stamp for last page of log backup
`Page Count`	Total number of pages saved
`Devices Used`	Number of backup devices used
`Database ID`	Database ID used to identify data and log backups that belong together
`Max Used Data Page`	Maximum number of pages used (indication of minimum database size when backup is imported)

In the event of errors, see Reply Format [Page 12].

Recovering a Damaged Index

Use

If you do not specify an options, you recover **all** damaged indexes with this command. If you specify data for a certain index, only the selected index is recovered:

To save time, indexes are not maintained when a backup is imported. These must be recovered explicitly. You recover damaged indexes or indexes that are no longer up to date.

The database software ignores damaged indexes. For this reason, they cannot cause errors, but do impair the performance of the database instance.

Prerequisites

You have the DBM operator authorization AccessSQL [Page 22].

You have opened an SQL session [Page 172].

You have executed the command for restoring a database instance [Page 147].

Syntax

```
sql_recreate_index [<scheme>.<table>.<index>]
```

Options

`<scheme>`	Scheme
`<table>`	Table name
`<index>`	Index name or field name

Reply

The system outputs an OK message.

In the event of errors, see Reply Format [Page 12].

Administration of DBM Operators

Use

The Database Manager CLI provides a number of DBM Server commands [Page 34] for transferring user data to the DBM Server [Page 12] or requesting it from the DBM Server.

Requesting the DBM Operator Authorizations [Page 156]	**user_getrights**
Request the Operator Data [Page 158]	**user_get**
Logon to the Database Manager [Page 36]	**user_logon**
Create a DBM Operator [Page 159]	**user_create**
Change DBM Operator Data [Page 159]	**user_put**
Delete a DBM Operator [Page 160]	**user_delete**
List of DBM Operators [Page 160]	**user_getall**

Prerequisites

You have the DBM operator authorization UserMgm [Page 21].

Requesting the DBM Operator Authorizations

Use

You call the list of DBM operator authorizations [Page 14] for the operator specified in **<userid>**.

You specify which group of DBM operator authorizations you want to retrieve with the option **<right_class>**.

Operator authorizations for the DBM Server	SERVERRIGHTS
Authorizations for the Database Manager GUI	GUIRIGHTS

Prerequisites

You have the DBM operator authorization UserMgm [Page 21].

Syntax

user_getrights <userid> <right_class>

Example [Page 157]

Reply

```
OK<NL>
UserMgm                 <flag>  User management
```

```
InstallMgm              <flag>  Install management
SystemCmd               <flag>  System commands
FileFull                <flag>  File full access
FileRead                <flag>  File read access
DBFileRead              <flag>  Database file read access
AccessUtility           <flag>  Utility session access
BackupRestore           <flag>  Backup restore actions
BackupSave              <flag>  Backup save actions
ParamFull               <flag>  Parameter full access
ParamCheckWrite         <flag>  Parameter check-write access
ParamRead               <flag>  Parameter read access
AccessSQL               <flag>  SQL session access
ExecLoad                <flag>  Can execute load
LoadSysTab              <flag>  Can load system tables
DBStop                  <flag>  Can stop database
DBStart                 <flag>  Can start database
DBInfoRead              <flag>  Info functions access
```

A plus symbol under `<flag>` indicates that the relevant authorization is assigned to the operator. A minus symbol under `<flag>` indicates that the relevant authorization is not assigned to the operator.

In the event of errors, see Reply Format [Page 12].

user_getrights Command: Example

Request the rights of the user with the name DBM, password DBM for using the DBM Server (see also: DBM operator authorizations [Page 14])

```
dbmcli -d DB -u dbm,dbm  user_getrights DBM Serverrights
```

```
OK
DBInfoRead              +       Request status data
ExecLoad                +       Execute the LOAD program
SystemCmd               +       Execute operating system commands
UserMgm                 +       User management
FileFull                -       File access (read and write)
FileRead                -       File access (read only)
DBFileRead              +       Database file access (read only)
Backup                  +       Saving backups
InstallMgm              +       Installation management
LoadSysTab              +       Load the system tables
ParamCheckWrite         +       Parameter access (checked write)
ParamFull               +       Parameter access (read and write)
ParamRead               +       Parameter access (read only)
DBStart                 +       Start database instance
DBStop                  +       Stop database instance
Recovery                +       Restoring backups
AccessSQL               +       Access to SQL session
AccessUtility           +       Access to utility session
```

Result of the Request

The DBM operator [Page 13] DBM has all DBM operator authorizations [Page 14] for the DBM Server [Page 12], with the exception of *FileFull* and *FileRead*.

Request the Operator Data

Use

You call the data for the DBM operator [Page 13] specified in **<userid>**.

The DBM operator authorizations [Page 14] assigned to the operator are listed in the <right_list> output by the Database Manager CLI, separated by commas.

Prerequisites

You have the DBM operator authorization UserMgm [Page 21].

Syntax

user_get <userid>

Request for the data for the DBM operator with the name DBM, password DBM

dbmcli -d DB -u dbm,dbm user_get dbm

```
OK
SERVERRIGHTS=DBInfoRead,ExecLoad,SystemCmd,UserMgm,DBFileRea
d,Backup,...
GUIRIGHTS=
SECONDPASSWORD=NO
DBMUSER=YES
SQLUSER=NO
SQLUSERMODE=
DISABLED=NO
SYSTEMNAME=
COMMENT=
```

Reply

```
OK<NL>
SERVERRIGHTS=<right_list><NL>
GUIRIGHTS=<right_list><NL>
SECONDPASSWORD=[YES|NO]<NL>
DBMUSER=[YES|NO]<NL>
SQLUSER=[YES|NO]<NL>
SQLUSERMODE=[DBA|STANDARD|RESOURCE]<NL>
SYSTEMNAME=<username><NL>
COMMENT=<comment><NL>
```

In the event of errors, see Reply Format [Page 12].

Create a DBM Operator

Use

You create a DBM operator [Page 13] having the *operator name* and *password* specified in **<userid>** and **<password>**.

For the moment this operator has no DBM operator authorizations [Page 14]. You must explicitly assign these to the operator (Change DBM Operator Data [Page 159]).

Alternatively, you can also specify a **<template_user>**, from which all DBM operator authorizations, but not operator name and operator password are then copied.

Prerequisites

You have the DBM operator authorization UserMgm [Page 21].

Syntax

```
user_create <userid>,<password> [<template_user>]
```

Reply

The system outputs an OK message.

In the event of errors, see Reply Format [Page 12]

Change DBM Operator Data

Use

You can modify the data of the DBM operator [Page 13] specified in **<userid>**.

The new values are specified with **<property>=<value>**.

In **<property>** you can specify all operator properties [Page 13] except the name. The name of a operator must not be modified.

When specifying server authorizations [Page 14] and GUI authorizations, you can indicate several authorizations separated by commas. Put a plus sign in front of the right to assign it and a minus sign to revoke it.

Prerequisites

You have the DBM operator authorization UserMgm [Page 21].

Syntax

```
user_put <userid> <property>=<value> [, <property>=<value>, ...]
```

Reply

The system outputs an OK message.

```
user_put samplename SERVERRIGHTS=+StartDB, -StopDB
```

```
OK
```

In the event of errors, see Reply Format [Page 12].

Delete a DBM Operator

Use

You delete the DBM operator [Page 13] specified in **<userid>**.

You cannot delete the DBM operator currently logged on, or the **first DBM operator** that was created when creating the database instance.

Prerequisites

You have the DBM operator authorization UserMgm [Page 21].

Syntax

user_delete <userid>

Reply

The system outputs an OK message.

In the event of errors, see Reply Format [Page 12]

List of DBM Operators

Use

The system lists all DBM operators [Page 13] registered for the current database instance.

Prerequisites

You have the DBM operator authorization UserMgm [Page 21].

Syntax

user_getall

Display all DBM operators for the database instance:

dbmcli -d DB -u dbm,dbm user_getall

```
OK
dbm
DBA
domain
```

Reply

```
OK<NL>
<userid><NL>
<userid><NL>
...
```

In the event of errors, see Reply Format [Page 12]

Accessing the Database Kernel

Use

This group of commands is for communicating with the database kernel.

Prerequisites

Note the DBM operator authorizations [Page 14] for each of the DBM Server commands [Page 34].

To use these commands you must generate and work in a utility session [Page 164], an SQL session [Page 172], or a service session [Page 174]. For more information see the documentation about the particular commands.

Overview of Commands for Accessing the Database Kernel: [Page 161]

- Access Through Utility Session [Page 162]
- Access Through SQL Session [Page 166]
- Access Through Service Session [Page 173]
- Access to Database Events [Page 174]

Overview of Commands for Accessing the Database Kernel

Result Set Structure Request [Page 167]	**sql_info**
Request Status Data for the Database Instance [Page 167]	**info**
New Database Instance Activation [Page 163]	**util_activate**
Activate a Database Event [Page 175]	**event_set**
Update the Optimizer Statistics [Page 169]	**sql_updstat**
Terminating a Database Event Session [Page 176]	**event_release**
Terminating a Service Session [Page 173]	**service_release**
Terminating an SQL Session [Page 170]	**sql_release**

Terminating a Utility Session [Page 163]	`util_release`
Scroll in the Result Data [Page 170]	`sql_fetch`
Scroll in the Status Information of the Database Instance [Page 171]	`info_next`
Deactivating a Database Event [Page 176]	`event_delete`
Opening a Service Session [Page 174]	`service_connect`
Opening an SQL Session [Page 172]	`sql_connect`
Opening a Utility Session [Page 164]	`util_connect`
Reading a Physical Database Page [Page 164]	`util_getpage`
List of Activated Database Events [Page 177]	`event_list`
Writing a Physical Database Page [Page 165]	`util_putpage`
SQL Statement Transfer [Page 172]	`sql_execute`
Transferring a Utility Command [Page 166]	`util_execute`
Wait for a Database Event [Page 178]	`event_wait`
Recovering a Damaged Index [Page 155]	`sql_recreate_index`

Kernel Access Through a Utility Session

Use

Commands you can use in a utility session with the database instance:

New Database Instance Activation [Page 163]	`util_activate`
Terminating a Utility Session [Page 163]	`util_release`
Opening a Utility Session [Page 164]	`util_connect`
Reading a Physical Database Page [Page 164]	`util_getpage`
Writing a Physical Database Page [Page 165]	`util_putpage`
Transferring a Utility Command [Page 166]	`util_execute`

Prerequisites

You have the DBM operator authorization AccessUtility [Page 22].

New Database Instance Activation

Use

You run a newly registered database instance for the first time and create the first **database** user (Database Instance Registration [Page 71]). This user is the database administrator with the name *SYSDBA*.

See also: *User Manual: SAP DB*

Prerequisites

You have the DBM operator authorization AccessUtility [Page 22].

Syntax

```
util_activate <sysdba>,<password>
```

Reply

The system outputs an OK message.

In the event of errors, see Reply Format [Page 12].

Terminating a Utility Session

Use

With this command you end the active utility session. All assigned resources are released.

Prerequisites

You have the DBM operator authorization AccessUtility [Page 22] or Backup [Page 17].

You have opened a utility session [Page 164].

Syntax

```
util_release
```

Reply

The system outputs an OK message.

In the event of errors, see Reply Format [Page 12].

Opening a Utility Session

Use

You open a session with the database instance with which to use the Utility task of the database kernel. This task is reserved solely for the administration of the database.

Only one utility task exists for each database instance. Administrative operations cannot therefore be performed in parallel.

If you have not specified an operator, the DBM operator [Page 13] registered on the database server will be logged on.

Prerequisites

You have the DBM operator authorization AccessUtility [Page 22] or Backup [Page 17].

You are working in script mode [Page 179] or session mode [Page 180].

Syntax

```
util_connect
```

Reply

The system outputs an OK message.

In the event of errors, see Reply Format [Page 12].

Reading a Physical Database Page

Use

A physical page is called from the devspace specified. Enter the number of the devspace required in the parameter **<dev_no>,** and its type in **<dev_mode>**. The system displays the requested page, and before the `<page_data>` also the length of the page `<page_size>` in bytes.

If you only specify one page number **<page_no>**, the corresponding logical page is output.

This command is only useful when using client applications.

Prerequisites

You have the DBM operator authorization AccessUtility [Page 22].

The database is in `COLD` mode.

You have opened a utility session [Page 164].

Syntax

```
util_getpage [<dev_mode> <dev_no>] <page_no>
```

Option <dev_mode>

SYS	System devspace

MSYS	Mirrored system devspace
DATA	Data devspace
MDATA	Mirrored data devspace
LOG	Log devspace
MLOG	Mirrored log devspace (`LOG_MODE=DUAL`)
NAME	When using this option, do not specify a number for the following parameter **<dev_no>**; instead enter the name of the devspace **util_getpage [NAME <dev_name>] <page_no>**

Reply

```
OK<NL>
<page_size><NL>
<page_data>
```

Writing a Physical Database Page

Use

The page is written to the devspace specified.

Enter the number of the devspace required in the parameter **<dev_no>**, and its type in **<dev_mode>**.

If you only specify one page number **<page_no>**, this is the logical page that is written to the devspace.

You can define the length of the database page **(<page_data>)** using the parameter **<page_size>**.

This command is only useful when using client applications.

Prerequisites

You have the DBM operator authorization AccessUtility [Page 22].

The database is in `COLD` mode.

You have opened a utility session [Page 164].

Syntax

util_putpage [<dev_mode> <dev_no>] <page_no>
<page_size><NL><page_data>

Option <dev_mode>

SYS	System devspace
MSYS	Mirrored system devspace
DATA	Data devspace

MDATA	Mirrored data devspace
LOG	Log devspace
MLOG	Mirrored log devspace (`LOG_MODE=DUAL`)
NAME	When using this option, do not enter a number for the following parameter **<dev_no>**; enter the name of the devspace instead: **util_putpage [NAME <dev_name>] <page_no>**

Reply

The system outputs an OK message.

In the event of errors, see Reply Format [Page 12].

Transferring a Utility Command

Use

You transfer the utility command specified to the database kernel. Once the command has been executed successfully, no result data are output, just an OK message.

Prerequisites

You have the DBM operator authorization AccessUtility [Page 22].

You have opened a utility session [Page 164].

Syntax

```
util_execute <statement>
```

Reply

The system outputs an OK message.

If an error occurs, the system terminates the utility session and outputs an error message number. The system sets up the session again with the current parameters.

See also: *Messages: SAP DB 7.2 and 7.3*

Kernel Access Through an SQL Session

Use

Commands you can use in an SQL session with the database kernel:

Result Set Structure Request [Page 167]	**sql_info**
Request Status Data for the Database Instance [Page 167]	**info**
Update the Optimizer Statistics [Page 169]	**sql_updstat**

Terminating an SQL Session [Page 170]	**sql_release**
Scrolling in the Result Data [Page 170]	**sql_fetch**
Scroll in the Status Information of the Database Instance [Page 171]	**info_next**
Opening an SQL Session [Page 172]	**sql_connect**
SQL Statement Transfer [Page 172]	**sql_execute**

Prerequisites

Note the DBM operator authorizations [Page 14] for each of the DBM Server commands [Page 34].

Result Set Structure Request

Use

You request the structure of the result set for an SQL statement.

Prerequisites

You have the DBM operator authorization AccessSQL [Page 22].

You have opened an SQL session [Page 172].

Syntax

sql_info <statement>

sql_info select username,connectmode,templimit, user_id from users

```
OK
USERNAME;CONNECTMODE;TEMPLIMIT;USER_ID
```

Reply

The names of all columns are output separated by semicolons.

```
OK<NL>
<col_1>;<col_2>;...;<col_n><NL>q
```

Request Status Data for the Database Instance

Use

You request the information given under **<info_id>**.

Prerequisites

You have the DBM operator authorization DBInfoRead [Page 15].

You have opened an SQL session [Page 172].

Syntax

```
info <info_id>
```

Option <info_id>

BADDEVSPACES	Damaged devspaces
BADIDX	Damaged indexes
CACHES	Cache accesses
CLSCON	liveCache classes container (only for liveCache instance type)
COLUMNS	Columns of database instance tables
DATA	Data devspaces
DEVSPACES	Devspaces
DISABLEDIDX	Deactivated indexes
EVENTS	Active events
INDEXES	Database instance indexes
INFOS	Available information
IO	Read and write operations
LOCKS	Locks
LOG	Log status
OMSCACHES	liveCache caches (only for liveCache instance type)
OMSMON	liveCache monitors (only for liveCache instance type)
PARAMS	Parameters
STATE	A short description of the status of the database instance.
SYSDDCONFIG	Internal configuration
SYSDDMONITOR	Internal monitors
SYSDDSRVDB	Internal database status
TABLES	Database instance tables
UNUSEDIDX	Unused indexes
UPDSTAT	Statistics can be updated for the Optimizer
UPDSTATW	Statistics must be updated for the Optimizer
USERS	Logged on database users
VERSION	Database program version

Reply

```
OK<NL>
(END|CONTINUE)<NL>
<descriptionrecord><NL>
<inforecord><NL>
<inforecord><NL>
```

END	Result set output complete
CONTINUE	More result data ready

The description line <descriptionrecord> contains the field names of the following information lines <inforecord>.

Update the Optimizer Statistics

Use

You update the statistics for the Optimizer.

These statistics enable the Optimizer to choose the best strategy for retrieving data. Obsolete statistics impair the performance of the database instance.

You can update all the statistics or restrict the update function to a set of statistics using a table specification **<tablespec>**.

Prerequisites

You have the DBM operator authorization AccessSQL [Page 22].

You have opened an SQL session [Page 172].

Syntax

```
sql_updstat [<tablespec>]
```

Reply

The system outputs an OK message.

In the event of errors, see Reply Format [Page 12].

Update Optimizer Statistics with the XPU Program

Use

The optimizer is part of the database instance and ensures the fastest possible processing of database accesses. To do this, it accesses the statistics about the contents of database tables.

You update the statistics about the contents of the tables of the database instance using the XPU program.

You use the parameters for the XPU program as parameters.

Prerequisites

You have the DBM operator authorization AccessSQL [Page 22].

The database instance is in WARM mode.

Syntax

```
exec_xpu <xpu_params>
```

Reply

You receive the reply of the XPU program.

```
OK<NL>
<xpu_output>
```

In the event of errors, see Reply Format [Page 12].

Terminating an SQL Session

Use

You exit the active SQL session. All assigned resources are released.

Prerequisites

You have the DBM operator authorization AccessSQL [Page 22] or DBInfoRead [Page 15].

You have opened an SQL session [Page 172].

Syntax

```
sql_release
```

Reply

The system outputs an OK message.

In the event of errors, see Reply Format [Page 12].

Scrolling in the Result Data

Use

You can output additional result data.

Prerequisites

You have the DBM operator authorization AccessSQL [Page 22].

You have opened an SQL session [Page 172] and transferred an SQL statement [Page 172]. In the response, `CONTINUE` indicated that additional result data was available.

Syntax

```
sql_fetch
```

Reply

The fields in a record are separated by semicolons.

Character strings are output in single quotation marks.

```
OK<NL>
(END|CONTINUE)<NL>
<record><NL>
<record><NL>
```

Values for the individual fields of the reply

END	The contents of the file have been transferred in full. The file is closed automatically.
CONTINUE	The file contains further entries that were not transferred due to the limited storage available for replies. You can interrogate these by entering the above command again.

Scroll in the Status Information of the Database Instance

Use

You can output further information on the state of the database instance.

Prerequisites

You have the DBM operator authorization DBInfoRead [Page 15].

When you request status information for the database instance [Page 167] or scroll in the status information for the database instance, the system informs you that you can request more information using the keyword CONTINUE.

Syntax

```
info_next
```

Reply

```
OK<NL>
(END|CONTINUE)<NL>
<description_record><NL>
<info_record><NL>
<info_record><NL>
```

Values for the individual fields of the reply

END	Result set output complete
CONTINUE	More result data ready
<description_record>	Name of the fields of the subsequent information lines
<info_record>	Information

Opening an SQL Session

Use

You create an SQL session with the database kernel for the specified operator. The operator of an SQL session shares the performance of the database instance with up to ***n*** additional operators.

If you have not specified an operator, the DBM operator [Page 13] registered on the server will be logged on.

Prerequisites

You have the DBM operator authorization DBInfoRead [Page 15] or AccessSQL [Page 22].

You are working in script mode [Page 179] or session mode [Page 180].

Syntax

```
sql_connect [<userid>,<password>]
```

Reply

The system outputs an OK message.

In the event of errors, see Reply Format [Page 12].

SQL Statement Transfer

Use

You transfer the statement specified to the database kernel. If you specify a **SELECT** statement, the results are output: The system outputs an OK message for other statements.

Prerequisites

You have the DBM operator authorization AccessSQL [Page 22].

You have opened an SQL session [Page 172].

Syntax

```
sql_execute <statement>
```

```
sql_execute select username,connectmode,templimit,user_id
from users

OK
END
'PUBLIC';'SINGLE';(null);3
'DBA';'SINGLE';(null);10
'DOMAIN';'SINGLE';(null);11
```

```
'SYS';'SINGLE';(null);12
'CONTROL';'MULTIPLE';(null);0
```

Reply

```
OK<NL>
(END|CONTINUE)<NL>
<record><NL>
<record><NL>
```

The fields in an answer record are separated by semicolons.

Character strings are output in single quotation marks.

END	Result set output complete
Continue	More result data ready

In the event of errors, see Reply Format [Page 12].

Kernel Access Through a Service Session

Use

To use the DBM Server commands for backup checks [Page 139] and restoring parameter files from the data backup [Page 150], you must have a service session running.

You can use the following DBM Server commands to open and terminate a service session:

Opening a Service Session [Page 174]	**service_connect**
Terminating a Service Session [Page 173]	**service_release**

Prerequisites

You have the DBM operator authorization Backup [Page 17].

Terminating a Service Session

Use

This command ends the active service session. All assigned resources are released. This command is executed implicitly when the session is exited with the DBM Server [Page 12].

Prerequisites

You have the DBM operator authorization Backup [Page 17].

You have opened a service session [Page 174].

Syntax

```
service_release
```

Reply

The system outputs an OK message.

In the event of errors, see Reply Format [Page 12].

Opening a Service Session

Use

You start the appropriate service kernel for the version of the database software and open a session with the utility task.

The service kernel is a specially configured database instance without a dataset of its own. You can use it to carry out tasks for other database instances, such as backup checks [Page 139], or restoring the parameter file from a data backup [Page 150].

Prerequisites

You have the DBM operator authorization Backup [Page 17].

You are working in script mode [Page 179] or session mode [Page 180].

Syntax

```
service_connect
```

Reply

The system outputs an OK message.

In the event of errors, see Reply Format [Page 12]

Access to Database Events

Use

You can use the following DBM Server commands [Page 34] to access events of the database kernel:

Activate a Database Event [Page 175]	`event_set`
Terminating a Database Event Session [Page 176]	`event_release`
Deactivating a Database Event [Page 176]	`event_delete`
List of Activated Database Events [Page 177]	`event_list`
Wait for a Database Event [Page 178]	`event_wait`

Prerequisites

You have the DBM operator authorization DBInfoRead [Page 15].

Activate a Database Event

Use

You activate a database event. The Database Manager CLI informs you of the occurrence of the activated event, in accordance with the parameters that have been set, if you call it using the command to wait for a database event [Page 178].

Prerequisites

You have the DBM operator authorization DBInfoRead [Page 15].

Syntax

```
event_set <name> [<prio>] [[<value>] <value>]
```

You activate the event `BACKUP_PAGES` with the value `1000`.

```
dbmcli -d DB -u dbm,dbm -uUTL -c event_set BACKUP_PAGES 1000
```

```
OK
```

The Database Manager CLI will, from now on, report the backup of 1000 pages during the running backup process, if you call it using the command to wait for a database event [Page 178].

Options

<name>	Name of the event
<prio>	Priority of the event **NIL**: No priority defined **MEDIUM**: Medium priority **HIGH**: High priority
<value>	Additional value whose meaning depends on the event

Reply

The system outputs an OK message.

In the event of errors, see Reply Format [Page 12].

Terminating a Database Event Session

Use

When waiting for a database event [Page 178], a database instance session is implicitly created. You end that session with this command.

Prerequisites

You have the DBM operator authorization DBInfoRead [Page 15].

Syntax

```
event_release
```

Reply

The system outputs an OK message.

In the event of errors, see Reply Format [Page 12].

Deactivating a Database Event

Use

You deactivate a database event. The occurrence of the event will now no longer be reported by the Database Manager CLI.

Prerequisites

You have the DBM operator authorization DBInfoRead [Page 15].

Syntax

```
event_delete <name> [<prio>] [[<value>] <value>]
```

You deactivate the event `BACKUP_PAGES`, for which the value `1000 was defined`.

```
dbmcli -d DB -u dbm,dbm -uUTL -c event_delete BACKUP_PAGES
1000
```

```
OK
```

Options

<name>	Name of the event
<prio>	Priority of the event **NIL**: No priority defined **MEDIUM**: Medium priority **HIGH**: High priority
<value>	Additional value whose meaning depends on the event

Reply

The system outputs an OK message.

In the event of errors, see Reply Format [Page 12].

List of Activated Database Events

Use

You request a list of activated database events.

Prerequisites

You have the DBM operator authorization DBInfoRead [Page 15].

Syntax

event_list

```
dbmcli -d DB -u dbm,dbm event_list
```

```
OK
Name             State    Prio     1st  2nd   Description
BACKUP_PAGES     OFF      NIL      0          Next <n> pages
moved
DB_ABOVE_LIMIT   ON       MEDIUM   90         Database above
limit
DB_ABOVE_LIMIT   ON       HIGH     95         Database above
limit
DB_ABOVE_LIMIT   ON       HIGH     96         Database above
limit
DB_ABOVE_LIMIT   ON       HIGH     97         Database above
limit
DB_ABOVE_LIMIT   ON       HIGH     98         Database above
limit
DB_ABOVE_LIMIT   ON       HIGH     99         Database above
limit
DB_BELOW_LIMIT   OFF      NIL      0          Database below
limit
DW_COMPRESS      OFF      NIL                 Compress device
starts
DW_DISPLACE      OFF      NIL                 Displace device
starts
LOG_ABOVE_LIMIT OFF       NIL      0          Log above limit
```

Reply

```
OK<NL>
Name        State    Prio     1st      2nd      Description<NL>
<name>      <state> <prio>   <value> <value> <description><NL>
<name>      <state> <prio>   <value> <value> <description><NL>
...
```

Values in the individual fields of the reply

<name>	Name of the event
<state>	Status of the event ON: Event is activated OFF: Event is not activated The database kernel reports the occurrence of an activated event while waiting for a database event [Page 178].
<prio>	Priority of the event NIL: No priority defined MEDIUM: Medium priority HIGH: High priority
<value>	Additional value whose meaning depends on the event
<description>	Description of the event

In the event of errors, see Reply Format [Page 12].

Wait for a Database Event

Use

You request notification from the Database Manager CLI about the occurrence of an activated event and the data associated with it.

Once you have received the notification, evaluate it, and wait for the next notification from the Database Manager CLI of the occurrence of the activated event by entering the command again.

You only receive a reply to this command if the activated event occurs.

Use this command through a Database Manager programming interface.

If you are to use the command with the Database Manager CLI, work in session mode [Page 180].

Prerequisites

You have the DBM operator authorization DBInfoRead [Page 15].

You have activated a database event (Activate a Database Event [Page 175]).

Syntax

event_wait

```
dbmcli -d DB -u dbm,dbm
dbmcli on DB>event_wait
```

```
OK
NAME=BACKUP_PAGES
PRIORITY=LOW
DATE=20010712
TIME=00151725
VALUE1=99
COUNT=188
DESCRIPTION=Next <n> pages moved
```

You have received the notification about the occurrence of the event BACKUP_PAGES.

Operating Modes

The program Database Manager (DBMCLI) provides three operating modes:

- Command mode [Page 179]
- Script mode [Page 179]
- Session mode [Page 180]

All modes are implicitly activated by the way the Database Manager CLI is called.

Command Mode

The Database Manager CLI works in command mode if a DBM Server command [Page 34] is specified when calling.

A session with the DBM Server [Page 12] is then set up and the command executed.

The session (and thus also the DBM Server) and the program Database Manager CLI are then terminated.

Script Mode

Syntax

dbmcli -i [Page 26]

Use this command to activate the script mode.

A script with DBM Server commands [Page 34] is transferred to the Database Manager CLI; these are processed in one session with the DBM Server [Page 12].

Use the following input methods for this:

Local System Call: ! [Page 181]
Comment: # [Page 180]
Linking Lines: / [Page 181]
Linking Lines: < [Page 181]

Session Mode

The Database Manager CLI works in the session mode if no DBM Server command [Page 34] and no input script are specified.

Within a session of the Database Manager CLI you can execute several commands.

Use the following input methods for this:

Local System Call: ! [Page 181]
Comment: # [Page 180]
Linking Lines: / [Page 181]
Linking Lines: < [Page 181]

You terminate the session mode by entering **`release`**.

Input Options

The following input options are available for the script mode [Page 179] and the session mode [Page 180]:

Local System Call: ! [Page 181]
Comment: # [Page 180]
Linking Lines: / [Page 181]
Linking Lines: < [Page 181]

Comment:

A command which starts with a number character is ignored.

Local System Call: !

The command to be entered is preceded by an exclamation mark.

The command is not transferred to the DBM Server [Page 12] but is executed in the local command interpreter of the operating system.

Linking Lines: /

Prerequisites

You are working in session mode [Page 180] or script mode [Page 179].

If a multiple-row DBM Server Command [Page 34] ends with an oblique, the system recognises that the command is not complete. It is not executed when a line feed is inserted.

Only the entry of a line feed without a preceding oblique indicates to the system that the command is complete. It is then transferred to the DBM Server [Page 12].

Linking Lines: <

Using the special character less than (<), DBM Server commands [Page 34] consisting of several lines can be transferred to the DBM Server [Page 12].

Syntax

```
'<'<stop-token> <command-begin>
```

The character string <command-begin> is linked to the continuation lines until a line is found which starts with **<stop-token>**.

All lines are separated from one another by a line feed.

The line with **<stop-token>** is not incorporated into the command.

Database Files

The following database files exist in connection with SAP DB database instances:

File Content	File Name	Directory	DBM Server <file_id>	File Class (see also:

			(see also: **Opening a Datbase File [Page 45]**)	Compressing Diagnosis and Database Files [Page 46])
Database messages	knldiag Specification of the file name using datbase parameter _KERNELDIAGFILE	<independent_data_path>/wrk/<database_name>	KNLDIAG	protocol
Database error messages	knldiag.err Specification of the file name (not the file type) using database parameter _KERNELDIAGFILE	<independent_data_path>/wrk/<database_name>	KNLDIAGERR	protocol
Old database messages	knldiag.old Specification of the file name (not the file type) using database parameter _KERNELDIAGFILE	<independent_data_path>/wrk/<database_name>	KNLDIAGOLD	protocol
DBM Server log	dbm.prt	<independent_data_path>/wrk/<database_name>	DBMPRT	protocol
Binary version of the database trace	knltrace Specification of the file name (not the file type) using database parameter _KERNELTRACEFILE	<independent_data_path>/wrk/<database_name>	KNLTRC	dump
Text version of the database trace (see also: Creating the Text Version of the Database Trace [Page 50])	<dbname>.prt	<independent_data_path>/wrk/<database_name>	KNLTRCPRT	protocol
Log of the utility command	dbm.utl Specification of the file name and file type using database parameter _UTILITY_PROTFILE	<independent_data_path>/wrk/<database_name>	UTLPRT	protocol
History of backups and restores	dbm.knl Specification of the file name and the file type using database parameter _BACKUP_HISTFILE	<independent_data_path>/wrk/<database_name>	BACKHIST	backup

History of the media used for backups	dbm.mdf Specification of the file name and the file type using database parameter _BACKUP_MED_DEF	<independent_data_path>/wrk/<database_name>	BACKMDF	backup
History of backups with external backup tools	dbm.ebf	<independent_data_path>/wrk/<database_name>	BACKEBF	backup
Log of last backup with an external backup tool	dbm.ebp	<independent_data_path>/wrk/<database_name>	BACKEBP	backup
Log of loading the system tables	dbm.ins	<independent_data_path>/wrk/<database_name>	INSTPRT	protocol
Log of database administration activities	dbahist.prt	<independent_data_path>/wrk/<database_name>/dbahist	DBAHIST	protocol
Database Manager backup media	dbm.mmm	<independent_data_path>/wrk/<database_name>	DBMMDF	config
Diagnosis package (see also: Compressing Diagnosis and Database Files [Page 46])	diagpkg.tgz	<independent_data_path>/wrk/<database_name>	DIAGTGZ	
Memory extract of the database instance	knldump Specification of the file name and the file type using database parameter _KERNELDUMPFILE	<independent_data_path>/wrk/<database_name>	KNLDUMP	dump
Memory extract of the database instance runtime environment	rtedump Specification of the file name and the file type using database parameter _RTEDUMPFILE	<independent_data_path>/wrk/<database_name>	RTEDUMP	dump
Log of database events (see also: Accessing the Database Kernel [Page 161])	knldiag.evt Specification of the file name and the file type using database parameter _EVENTFILE	<independent_data_path>/wrk/<database_name>	KNLEVT	protocol
Log of the last liveCache initialization (see also:	lcinit.log	<dependent_data_path>	LCINIT	lvc

Execute liveCache Initialization Script [Page 39])				
Log of all liveCache initializations	lcinit.his	<dependent_data_path>	LCINITHIS	lvc
Log of liveCache activities	lc<token>	<independent_data_path>/wrk/<database_name>	LCTRC#<token>	lvc
DBM Server configuration	dbm.cfg	<independent_data_path>/wrk/<database_name>		config
Database instance parameters	<dbname>	<independent_data_path>/config		config
History of database instance parameters	<dbname>.pah	<independent_data_path>/config	DBMPAHI	protocol
Memory extract of the SQL analysis of the database instance	AK.buf, AK.dmp, AK.stm	<independent_data_path>/wrk/<database_name>		dump
Logs of the XServer program	xserver.prt, xserver.old, vserver.prot	<independent_data_path>/wrk		protocol
Script for the initialization of liveCache	lcinit or lcinit.cmd	<dependent_data_path>/sap		lvc

For more information about the SAP DB directory structure, **see also:** *User Manual: SAP DB*