

Table of Contents

DATABASE MAINTENANCE TASKS	3
MYSQL MAINTENANCE TASKS	4
<i>Analyze Tables</i>	5
<i>Check Tables</i>	6
<i>Optimize Tables</i>	8
<i>Repair Tables</i>	9
<i>Flush</i>	10
ORACLE MAINTENANCE TASKS	11
<i>Table Maintenance Tasks</i>	12
Enable / Disable Table Lock	13
Enable / Disable Row Movement	14
Shrink Space	15
Move	16
Collect Statistics	17
Validate Structure	18
<i>View Maintenance Tasks</i>	19
<i>Function/Procedure Maintenance Tasks</i>	20
<i>Index Maintenance Tasks</i>	21
<i>Java Maintenance Tasks</i>	22
<i>Materialized View Maintenance Tasks</i>	23
<i>Materialized View Log Maintenance Tasks</i>	24
<i>Package Maintenance Tasks</i>	25
<i>Trigger Maintenance Tasks</i>	26
<i>Type Maintenance Tasks</i>	27
<i>XML Schema Maintenance Tasks</i>	28
<i>Tablespace Maintenance Tasks</i>	29
POSTGRESQL MAINTENANCE TASKS	30
<i>Analyze</i>	31
<i>Vacuum</i>	32
<i>Reindex</i>	34
SQLITE MAINTENANCE TASKS	36
<i>Database and Table Maintenance Tasks</i>	37
Analyze	38
Vacuum	39
Reindex	40
View Master Table	41

<i>Index Maintenance Tasks</i>	42
<i>Server Monitor (Available only in Full Version & only for MySQL, Oracle and PostgreSQL Servers)</i>	43
Process List	44
Variables	45
Status	46

Database Maintenance Tasks

Navicat provides a complete solution for database and table maintenance on MySQL, Oracle, PostgreSQL and SQLite servers.

- [MySQL Maintenance Tasks](#)
- [Oracle Maintenance Tasks](#)
- [PostgreSQL Maintenance Tasks](#)
- [SQLite Maintenance Tasks](#)
- [Server Monitor](#)

MySQL Maintenance Tasks

Navicat provides a complete solution for most of the native MySQL services, which are intended for database and table maintenance. To make your work with the server easier, Navicat also provides some graphical tools for working with the server as a whole.

[Analyze Tables](#)

Analyzes and stores the key distribution for the table.

[Check Tables](#)

Checks the database tables for errors.

[Optimize Tables](#)

Reclaims the unused space in tables and defragments the data files.

[Repair Tables](#)

Repairs database tables that are corrupted.

[Flush](#)

Clears the internal MySQL caches.

Analyze Tables

Analyze Table analyzes and stores the key distribution for the table. During the analysis, the table is locked with a read lock for MyISAM and BDB. For InnoDB the table is locked with a write lock. Currently, MySQL supports analyzing only for MyISAM, BDB, and InnoDB tables. For MyISAM tables, this statement is equivalent to using `myisamchk --analyze`.

Hint: Just simply right-click the table and select **Maintain Tables -> Analyze Tables**.

MySQL uses the stored key distribution to decide in which order tables should be joined when one does a join on something else than a constant.

Analyze Table returns a result set with the following columns:

Column	Value
Table	The table name.
Op	Always analyze.
Msg_type	One of status, error, info, or warning.
Msg_text	The message.

You can check the stored key distribution with the `SHOW INDEX` statement. If the table has not changed since the last **Analyze Table** statement, the table is not analyzed again.

Related topics:

[Check Tables](#)

[Optimize Tables](#)

[Repair Tables](#)

Check Tables

Check Table checks a table or tables for errors. Currently, MySQL supports checking only for MyISAM, InnoDB and ARCHIVE tables. For MyISAM tables, the key statistics are updated as well.

Hint: Just simply right-click the table and select **Maintain Tables -> Check Tables**.

Check Table returns a result set with the following columns:

Column	Value
Table	The table name.
Op	Always check.
Msg_type	One of status, error, info, or warning.
Msg_text	The message.

You might get many rows of information for each checked table. The last row has a *Msg_type* value of *status* and the *Msg_text* normally should be *OK*. If you do not get *OK*, or *Table is already up to date* you should normally run a repair of the table. *Table is already up to date* means that the storage engine for the table indicated that there was no need to check the table.

The other check options that can be given are shown in the following table:

Type	Meaning
QUICK	Don't scan the rows to check for wrong links.
FAST	Don't scan the rows to check for wrong links.
CHANGED	Only check tables which have been changed since last check or haven't been closed properly.
EXTENDED	Do a full key lookup for all keys for each row. This ensures that the table is 100 % consistent, but will take a long time!

Related topics:

[Analyze Tables](#)

[Optimize Tables](#)

[Repair Tables](#)

Optimize Tables

The main reason for optimizing your table is to reclaim unused space and to defragment the data file. You should optimize a table if you have deleted a large part of a table or if you have made many changes to a table with variable-length rows (tables that have VARCHAR, BLOB, or TEXT columns). Deleted records are maintained in a linked list and subsequent INSERT operations reuse old row positions.

Hint: Just simply right-click the table and select **Maintain Tables** -> **Optimize Tables**.

Currently, MySQL supports optimizing only for MyISAM, InnoDB and BDB tables.

For MyISAM tables, **Optimize Table** works as follows:

1. If the table has deleted or split rows, repair the table.
2. If the index pages are not sorted, sort them.
3. If the table's statistics are not up to date (and the repair could not be accomplished by sorting the index), update them.

Related topics:

[Analyze Tables](#)

[Check Tables](#)

[Repair Tables](#)

Repair Tables

Repair Table repairs a possibly corrupted table.

Hint: Just simply right-click the table and select **Maintain Tables -> Repair Tables**.

Repair Table returns a result set with the following columns:

Column	Value
Table	The table name.
Op	Always analyze.
Msg_type	One of status, error, info, or warning.
Msg_text	The message.

You might get many rows of information for each repaired table. The last row has a *Msg_type* value of *status* and *Msg_text* normally should be *OK*. If you do not get *OK*, you should try repairing the table with *myisamchk --safe-recover*. Repair Table does not yet implement all the options of *myisamchk*.) With *myisamchk --safe-recover*, you can also use options that Repair Table does not support, such as *--max-record-length*.

If **Quick** is given, Repair Table tries to repair only the index tree.

If you use **Extended**, MySQL creates the index row by row instead of creating one index at a time with sorting.

Related topics:

[Analyze Tables](#)

[Check Tables](#)

[Optimize Tables](#)

Flush

Flush clears or reloads various internal caches used by MySQL. To execute Flush, you must have the *Reload* privilege (see Server Security Management for MySQL Database).

Hint: Just simply right-click the connection and select **Flush**.

The following table illustrates the use of **Flush**:

- **Privileges**
Reloads the privileges from the grant tables in the MySQL database.
- **Hosts**
Empties the host cache tables. You should flush the host tables if some of your hosts change IP number or if you get the error message *Host 'host_name' is blocked*. When more than *max_connect_errors* errors occur in a row for a given host while connection to MySQL server, MySQL assumes something is wrong and blocks the host from further connection requests. Flushing the host tables allow the host to attempt to connect again.
- **Logs**
Closes and reopens all log files. If you have specified the update log file or a binary log file without an extension, the extension number of the log file will be incremented by one relative to the previous file. If you have used an extension in the file name, MySQL will close and reopen the update log file.
- **Status**
Resets most status variables to zero. This is something one should only use when debugging a query.
- **Tables**
Closes all open tables and forces all tables in use to be closed.

Oracle Maintenance Tasks

Navicat provides a complete solution for database object maintenance. To make your work with the server easier, Navicat provides some graphical tools for working with the server as a whole.

- [Tables](#)
- [Views](#)
- [Functions/Procedures](#)
- [Indexes](#)
- [Java](#)
- [Materialized Views](#)
- [Materialized View Logs](#)
- [Packages](#)
- [Triggers](#)
- [Types](#)
- [XML Schema](#)
- [Tablespaces](#)

Table Maintenance Tasks

Select the table for maintaining in the object pane. Right-click and select the **Maintain** from the popup menu.

[Enable / Disable Table Lock](#)

Allows / Prevents DDL operations on the table.

[Enable / Disable Row Movement](#)

Allows / Prevents the database to move a row.

[Shrink Space](#)

Shrinks space in a table.

[Move](#)

Relocates data of a nonpartitioned table or of a partition of a partitioned table into a new segment.

[Collect Statistics](#)

Collects table statistics.

[Validate Structure](#)

Verifies the integrity of the structure of a table.

Enable / Disable Table Lock

Table Lock locks a table to prevent DDL operations. Oracle Database permits DDL operations on a table only if the table can be locked during the operation. Such table locks are not required during DML operations.

Enable Table Lock

Choose Enable Table Lock to enable table locks, thereby allowing DDL operations on the table. All currently executing transactions must commit or roll back before Oracle Database enables the table lock.

Disable Table Lock

Choose Disable Table Lock to disable table locks, thereby preventing DDL operations on the table.

Enable / Disable Row Movement

Row Movement is the moving of rows in tables. It is possible for a row to move, for example, during table compression or an update operation on partitioned data.

Enable Row Movement

Choose Enable Row Movement to allow the database to move a row, thus changing the rowid.

Disable Row Movement

Choose Disable Row Movement if you want to prevent the database from moving a row, thus preventing a change of rowid.

Shrink Space

Shrink Space is to compact the table segment. This clause is valid only for segments in tablespaces with automatic segment management. By default, Oracle Database compacts the segment, adjusts the high water mark, and releases the recuperated space immediately.

Compacting the segment requires row movement. Therefore, you must enable row movement for the table you want to shrink before shrink space. Further, if your application has any rowid-based triggers, you should disable them before issuing this clause.

Move

Move relocates data of a nonpartitioned table or of a partition of a partitioned table into a new segment, optionally in a different tablespace, and optionally modify any of its storage attributes.

Collect Statistics

Collect Statistics analyzes the contents of tables. When you analyze a table, the database collects statistics about expressions occurring in any function-based indexes as well. Therefore, be sure to create function-based indexes on the table before analyzing the table.

Oracle Database collects the following statistics for a table. Statistics marked with an asterisk (*) are always computed exactly.

NUM_ROWS	Number of rows.
* BLOCKS	Number of data blocks below the high water mark - the number of data blocks that have been formatted to receive data, regardless whether they currently contain data or are empty.
* EMPTY_BLOCKS	Number of data blocks allocated to the table that have never been used.
AVG_SPACE	Average available free space in each data block in bytes.
CHAIN_COUNT	Number of chained rows.
AVG_ROW_LEN	Average row length, including the row overhead, in bytes.

Validate Structure

Validate Structure verifies the integrity of the structure of a table. The statistics collected by this clause are not used by the Oracle Database optimizer. If the structure is valid, no error is returned. However, if the structure is corrupt, an error message will be shown.

For a table, Oracle Database verifies the integrity of each of the data blocks and rows.

View Maintenance Tasks

Select the view for maintaining in the object pane. Right-click and select the **Maintain** from the popup menu.

Compile

To recompile the view specification or body.

Function/Procedure Maintenance Tasks

Select the function/procedure for maintaining in the object pane. Right-click and select the **Maintain** from the popup menu.

Compile

To recompile the function/procedure specification or body.

Compile for Debug

To recompile the function/procedure specification or body and instruct the PL/SQL compiler to generate and store the code for use by the PL/SQL debugger.

Index Maintenance Tasks

Select the index for maintaining in the object pane. Right-click and select the **Maintain** from the popup menu.

Rebuild

To re-create an existing index or one of its partitions or subpartitions. If the index is marked unusable, then a successful rebuild will mark it usable.

Make Unusable

To make the index unusable. An unusable index must be rebuilt, or dropped and re-created, before it can be used.

Coalesce

To instruct Oracle Database to merge the contents of index blocks where possible to free blocks for reuse.

Compute Statistics

To compute the statistics of the index.

Monitoring Usage

To begin monitoring the index. Oracle Database first clears existing information on index use, and then monitors the index for use until choosing No Monitoring Usage.

No Monitoring Usage

To terminate monitoring of the index.

Java Maintenance Tasks

Select the Java for maintaining in the object pane. Right-click and select the **Maintain** from the popup menu.

Compile or Resolve

To resolve the primary Java class schema object.

Set AuthID Current User

Set the invoker rights to AUTHID CURRENT_USER.

Set AuthID Definer

Set the invoker rights to AUTHID DEFINER.

Materialized View Maintenance Tasks

Select the materialized view for maintaining in the object pane. Right-click and select the **Maintain** from the popup menu.

Enable Row Movement

To enable row movement.

Shrink

To compact the materialized view segment. By default, Oracle Database compacts the segment, adjusts the high water mark, and releases the recuperated space immediately.

Compile

To explicitly revalidate a materialized view. If an object upon which the materialized view depends is dropped or altered, then the materialized view remains accessible, but it is invalid for query rewrite. You can choose this option to explicitly revalidate the materialized view to make it eligible for query rewrite.

Force Refresh

To perform a refresh.

Materialized View Log Maintenance Tasks

Navicat provides a complete solution for materialized view log maintenance.

Select the materialized view log for maintaining in the object pane. Right-click and select the **Maintain** from the popup menu.

Enable Row Movement

To enable row movement. Row movement indicates that rowids will change after the flashback occurs.

Disable Row Movement

To disable row movement.

Shrink Space

To compact the materialized view log segments. By default, Oracle Database compacts the segment, adjusts the high water mark, and releases the recuperated space immediately.

Package Maintenance Tasks

Select the package for maintaining in the object pane. Right-click and select the **Maintain** from the popup menu.

Compile

To recompile the package specification or body.

Compile Debug

To recompile the package specification or body and instruct the PL/SQL compiler to generate and store the code for use by the PL/SQL debugger.

Trigger Maintenance Tasks

Select the trigger for maintaining in the object pane. Right-click and select the **Maintain** from the popup menu.

Enable

To enable the trigger.

Disable

To disable the trigger.

Compile

To explicitly compile the trigger, whether it is valid or invalid. Explicit recompilation eliminates the need for implicit run-time recompilation and prevents associated run-time compilation errors and performance overhead.

Compile for Debug

To recompile the trigger and instruct the PL/SQL compiler to generate and store the code for use by the PL/SQL debugger.

Type Maintenance Tasks

Select the type for maintaining in the object pane. Right-click and select the **Maintain** from the popup menu.

Compile

To compile the type specification and body.

Compile Debug

To recompile the type specification or body and instruct the PL/SQL compiler to generate and store the code for use by the PL/SQL debugger.

XML Schema Maintenance Tasks

Select the XML Schema for maintaining in the object pane. Right-click and select the **Maintain** from the popup menu.

Compile

To re-compile an already registered XML schema. This is useful for bringing a schema in an invalid state to a valid state.

Purge

To remove the XML Schema completely from Oracle XML DB in Oracle 11g.

Tablespace Maintenance Tasks

Select the tablespace for maintaining in the object pane. Right-click and select the **Maintain** from the popup menu.

[Read Only]

To place the tablespace in transition read-only mode. In this state, existing transactions can complete (commit or roll back), but no further DML operations are allowed to the tablespace except for rollback of existing transactions that previously modified blocks in the tablespace.

Read Write

To indicate that write operations are allowed on a previously read-only tablespace.

Online

To take the tablespace online.

Offline

To take the tablespace offline.

Normal

To flush all blocks in all datafiles in the tablespace out of the system global area (SGA).

Temporary

Oracle Database performs a checkpoint for all online datafiles in the tablespace but does not ensure that all files can be written.

Immediate

Oracle Database does not ensure that tablespace files are available and does not perform a checkpoint.

Coalesce

To combine all contiguous free extents into larger contiguous extents for each datafile in the tablespace.

Shrink Space

To reduce the amount of space the tablespace is taking. This is valid only for temporary tablespaces in Oracle 11g.

PostgreSQL Maintenance Tasks

Navicat provides a complete solution for most of the native PostgreSQL services, which are intended for database and table maintenance. To make your work with the server easier, Navicat also provides some graphical tools for working with the server as a whole.

[Analyze](#)

Collects statistics about the contents of tables in the database.

[Vacuum](#)

Reclaims storage occupied by deleted tuples.

[Reindex](#)

Rebuilds an index using the data stored in the index's table.

Analyze

Analyze collects statistics about the contents of tables in the database, and stores the results in the system table *pg_statistic*. Subsequently, the query planner uses these statistics to help determine the most efficient execution plans for queries.

Analyze Database

Just simply right-click the database and select **Maintain Database -> Analyze Database....**

Analyze Table

Just simply right-click the table and select **Maintain Tables -> Analyze Tables....**

Parameters	
Verbose	Enables display of progress messages. (Default enabled in Navicat)

Outputs
When Verbose is specified, Analyze emits progress messages to indicate which table is currently being processed. Various statistics about the tables are printed as well.

Related topics:

[Vacuum](#)

[Reindex](#)

Vacuum

Vacuum reclaims storage occupied by deleted tuples. In normal PostgreSQL operation, tuples that are deleted or obsoleted by an update are not physically removed from their table; they remain present until a Vacuum is done. Therefore it's necessary to do Vacuum periodically, especially on frequently-updated tables.

Vacuum Database

Just simply right-click the database and select **Maintain Database -> Vacuum Database... -> desired option.**

Vacuum Table

Just simply right-click the table and select **Maintain Tables -> Vacuum Tables... -> desired option.**

Parameters	
Full	Selects "full" vacuum, which may reclaim more space, but takes much longer and exclusively locks the table.
Freeze	Selects aggressive "freezing" of tuples.
Analyze	Updates statistics used by the planner to determine the most efficient way to execute a query.
Verbose	Prints a detailed vacuum activity report for each table. (Default enabled in Navicat)

Outputs
When Verbose is specified, Vacuum emits progress messages to indicate which table is currently being processed. Various statistics about the tables are printed as well.

Related topics:

[Analyze](#)

[Reindex](#)

Reindex

Reindex rebuilds an index using the data stored in the index's table, replacing the old copy of the index. There are several scenarios in which to use Reindex:

- An index has become corrupted, and no longer contains valid data.
- An index has become "bloated", that it is contains many empty or nearly-empty pages.
- You have altered a storage parameter (such as fillfactor) for an index, and wish to ensure that the change has taken full effect.
- An index build with the CONCURRENTLY option failed, leaving an "invalid" index.

Reindex Database

Just simply right-click the database and select **Maintain Database -> Reindex Database....**

Reindex Table

Just simply right-click the table and select **Maintain Tables -> Reindex Tables....**

Related topics:

[Analyze](#)

[Vacuum](#)

SQLite Maintenance Tasks

Navicat provides a complete solution for most of the SQLite services, which are intended for database, table and index maintenance. To make your work with the server easier, Navicat also provides some graphical tools for working with the server as a whole.

- [Databases and Tables](#)
- [Indexes](#)

Database and Table Maintenance Tasks

To perform the following tasks, right-click and select the database or table for maintaining in the object pane.

[Analyze](#)

Collects statistics about the contents of indexes.

[Vacuum](#)

Cleans empty spaces from the database.

[Reindex](#)

Rebuilds an index using the data stored in the index's table.

[View Master Table](#)

Views all tables and indexes located in database.

Analyze

Analyze collects statistics about the indexes and stores the results in a special table in the database to help make better index selection. Subsequently, the query optimizer uses these statistics to help make better index choices.

Analyze Database

Just simply right-click the database and select **Maintain Database -> Analyze Database....**

Analyze Table

Just simply right-click the table and select **Maintain Tables -> Analyze Tables....**

Parameters	
Database	Analyzes all indexes in one database when the database name is specified.
Table	Analyzes all indexes in one table when the table name is specified.

Related topics:

[Vacuum](#)

[Reindex](#)

Vacuum

Vacuum cleans the empty spaces remained by dropping objects frequently from the database. It cleans the main database by copying its contents to a temporary database file and reloading the original database file from the copy. It only works on the main database. It is not possible to vacuum an attached database file.

Vacuum Database

Just simply right-click the database and select **Maintain Database -> Vacuum Database....**

Related topics:

[Analyze](#)

[Reindex](#)

Reindex

Reindex rebuilds an index using the data stored in the index's table, replacing the old copy of the index. Reindex is used when the definition of a collation sequence has changed.

Reindex Database

Just simply right-click the database and select **Maintain Database -> Reindex Database....**

Reindex Table

Just simply right-click the table and select **Maintain Tables -> Reindex Tables....**

Related topics:

[Analyze](#)

[Vacuum](#)

View Master Table

View Master Table allows viewing all tables and indexes located in an SQLite database. Every SQLite database has a special table named *SQLITE_MASTER* table that defines the schema for the database.

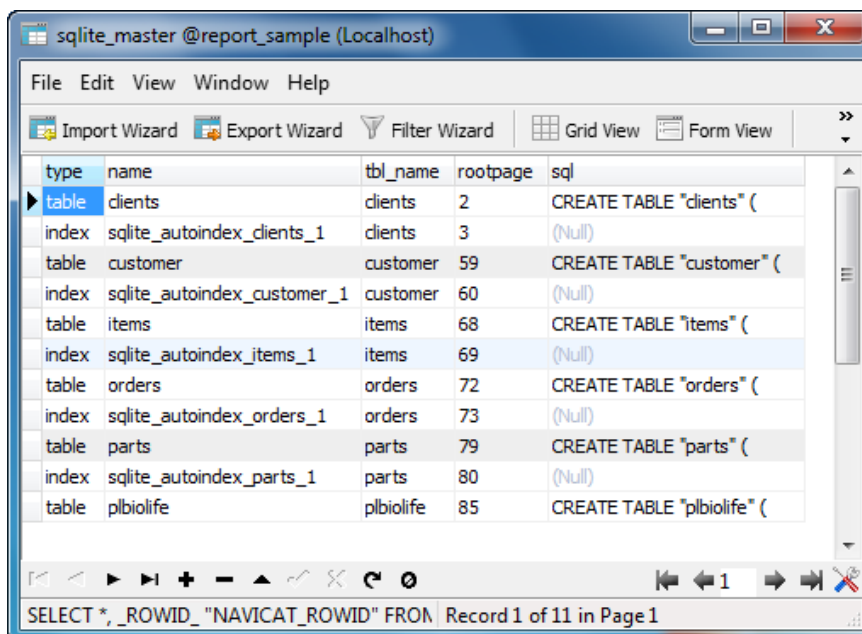
For tables, the **type** field will always be **table** and the **name** field will be the name of the table. For indexes, **type** is equal to **index**, **name** is the name of the index and **tbl_name** is the name of the table to which the index belongs. For both tables and indexes, the **sql** field is the text of the original *CREATE TABLE* or *CREATE INDEX* statement that created the table or index. For automatically created indexes (used to implement the PRIMARY KEY or UNIQUE constraints) the **sql** field is NULL.

The *SQLITE_MASTER* table is read-only. You cannot change this table using UPDATE, INSERT, or DELETE. The table is automatically updated by *CREATE TABLE*, *CREATE INDEX*, *DROP TABLE*, and *DROP INDEX* commands.

Temporary tables do not appear in the *SQLITE_MASTER* table. Temporary tables and their indexes and triggers occur in another special table named *SQLITE_TEMP_MASTER*. *SQLITE_TEMP_MASTER* works just like *SQLITE_MASTER* except that it is only visible to the application that created the temporary tables.

View Master Table

Just simply right-click the database and select **View Master Table**.




Index Maintenance Tasks

Select the index for maintaining in the object pane. Right-click and select the Maintain from the popup menu.

Reindex

To delete and recreate the index from scratch. This is useful when the definition of a collation sequence has changed.

Server Monitor (Available only in Full Version & only for MySQL, Oracle and PostgreSQL Servers)

Navicat provides **Server Monitor** to view the properties of the selected server. Just simply choose **Tools** ->  **Server Monitor** and select the target server type from the main menu.

- [Process List](#)
- [Variables](#)
- [Status](#)

Process List

Process List tab displays a list of processes from all the database servers selected in the check-list box.

To stop the selected process, just simply click the  **End Process** button.

Hint: The process list cannot be edited.

Auto refresh every seconds

If you want to take action on auto-refreshing the server in assigned seconds, just simply choose **View -> Set Auto Refresh Time** and enter an auto refresh value. To disable auto refresh feature, choose **View -> Auto Refresh**.

Note: Effect will take once you assign the value.

The process list provides the following information:

- Server name that is given while setting the connection.
- Process ID on the server.
- Serial number of the process. (Available only for Oracle Server)
- Current user who log in to the server.
- Host from which the user is connected.
- Database that the user is currently used. (Available only for MySQL and PostgreSQL Servers)
- Last command that was issued by the user.
- Time, state and info of the process.


Variables

Variables tab displays the list of all server variables and their values. The variables list is retrieved from the server(s) by issuing the SQL statement.

For MySQL Server: *SHOW VARIABLES*.

For Oracle Server: *SHOW ALL*.

For PostgreSQL Server: *SHOW ALL*.

Hint: To edit variable value in MySQL and Oracle servers, just simply click  or press Ctrl+Enter to open the editor for editing.

Hint: The value in PostgreSQL server cannot be edited here. (Those variables can be set using the *SET* statement, by editing the *postgresql.conf* configuration file.)

Status

Status tab displays the list of all server status of the server. The status list is retrieved from the MySQL server(s) by issuing the *SHOW STATUS* statement.

Hint: The status cannot be edited here.