FAQ • 03/2017

How do you access an SQL database in WinCC Runtime Advanced using a script?

SIMATIC WinCC Advanced, SIMATIC WinCC Runtime Advanced

https://support.industry.siemens.com/cs/ww/en/view/61883659

This entry is from the Siemens Industry Online Support. The general terms of use (http://www.siemens.com/terms_of_use) apply.

Security information

Siemens provides products and solutions with industrial security functions that support the secure operation of plants, systems, machines and networks.
In order to protect plants, systems, machines and networks against cyber threats, it is necessary to implement – and continuously maintain – a holistic, state-of-the-art industrial security concept. Siemens' products and solutions only form one element of such a concept.
Customer is responsible to prevent unauthorized access to its plants, systems, machines and networks. Systems, machines and components should only be connected to the enterprise network or the internet if and to the extent necessary and with appropriate security measures (e.g. use of firewalls and network segmentation) in place.
Additionally, Siemens' guidance on appropriate security measures should be taken into account. For more information about industrial security, please visit http://www.siemens.com/industrialsecurity.

Siemens' products and solutions undergo continuous development to make them more secure. Siemens strongly recommends to apply product updates as soon as available and to always use the latest product versions. Use of product versions that are no longer supported, and failure to apply latest updates may increase customer's exposure to cyber threats.
To stay informed about product updates, subscribe to the Siemens Industrial Security RSS Feed under http://www.siemens.com/industrialsecurity.

Go to the following link to download this document.

https://support.industry.siemens.com/cs/ww/en/view/61883659

Question

How do you access an SQL database in WinCC Runtime Advanced using a script?

Answer

Follow the instructions and notes listed in this document for a detailed answer to the above question.

The application is applicable exclusively for PC Runtime systems.

Table of contents

1 General Information 4

1.1 Notes on This Sample Configuration 4
1.2 Why Should You Archive Tags Using Scripts? 4
1.3 Note on Further Reading 4
1.4 Creating a Data Source 4
1.5 Development Environment 5

2 Scripts for Processing an SQL Database 6

2.1 Script Structure 6
2.2 Creating/Deleting an SQL Database 8
2.2.1 Creating an SQL Database 8
2.2.2 Deleting an SQL Database 8
2.3 Editing Tables and Data Records 9
2.3.1 Creating a New Table 11
2.3.2 Writing a New Data Record to a Table 11
2.3.3 Reading Data Records from a Table 12
2.3.4 Editing Data Records in a Table 12
2.3.5 Deleting Data Records from a Table 13
2.3.6 Reading All Data Records from a Table 14
2.3.7 Copying a Table 15
2.3.8 Deleting Tables 15

3 Using the Configured Screens 16

3.1 Requirements 16
3.2 Start Screen 16
3.3 Creating/Deleting a Database 17
3.4 Adding a New SQL Table 18
3.5 Adding a New Data Record to an SQL Table 19
3.6 Reading a Data Record from the SQL Table 20
3.7 Editing a Data Record from the SQL Table 21
3.8 Deleting a Data Record from the SQL Table 22
3.9 Reading All the Data Records from the SQL Table 23
3.10 Copying the SQL Table 24
3.11 Deleting an SQL Table 25

1 General Information

1.1 Notes on This Sample Configuration

A sample project is attached to this FAQ response. The sample project includes the scripts described in chapter **2**.

To underline the functions of the scripts, we have created a screen for each script by means of which you can execute the configured function. Refer here to chapter **3**.

1.2 Why Should You Archive Tags Using Scripts?

This FAQ response describes how to log tags in an SQL database using scripts.

The advantage here is that the tags can be stored, read out and subsequently edited in the form of a table, for example. Furthermore, you can also log tags of the "STRING" type.

1.3 Note on Further Reading

Information about SQL commands and about accessing SQL databases is available in various literature and in the internet.

- Website in English: http://www.sqlcommands.net/
- Website in German: http://ffm.junetz.de/members/reeg/DSP/node10.html

1.4 Creating a Data Source

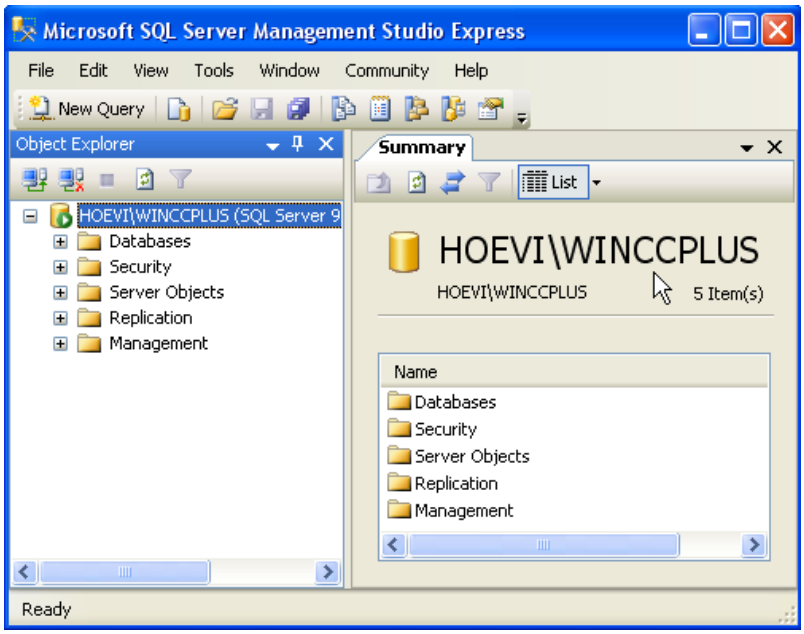
In order to create access to an ODBC data source, you must create a data source. Data sources can be created in the Windows Control Panel under "Administrative Tools > Data Sources (ODBC)".

How to create a Microsoft SQL database is described in Entry ID 61886098.

The entry includes the principal configuration steps.

The data source "**Database_1**" is used in this example; it is connected to the SQL server "Computer name**WINCCPLUS**".

Figure 1-1



Note The archiving is not linked to a specific SQL instance. Instead of "...\WINCCPLUS" you can also use "...\WINCCPLUSMIG", for example. Furthermore, you can also create your own instance and archive the data there. You need "SQL Server Management Studio" to create a new instance.

1.5 Development Environment

The project and the screenshots were created with the software and hardware components below.

- SIMATIC WinCC Advanced V14
- SIMATIC WinCC Runtime Advanced V14
- Microsoft SQL Server 2014
- Microsoft SQL Server Management Studio 11.0

2 Scripts for Processing an SQL Database

The sections below describe the scripts in the sample configuration.

The following functions can be executed using the scripts.

- Creating a new database.
- Deleting a database.
- Creating a table in a database.
- Creating a data record in a table.
- Reading a data record from a table.
- Editing a data record in a table.
- Deleting a data record in a table.
- Reading out the data records from a complete table.
- Copying a table.
- Deleting a table.

Note Open the attached configuration to understand better the sections below.

2.1 Script Structure

Essentially the scripts have the same structure. Below are descriptions of the lines that are identical in all the scripts.

Line 13:
The "**On Error Resume Next**" statement is required in case a runtime error occurs in the script. As soon as an error occurs the next line that contains the error routine is executed automatically.

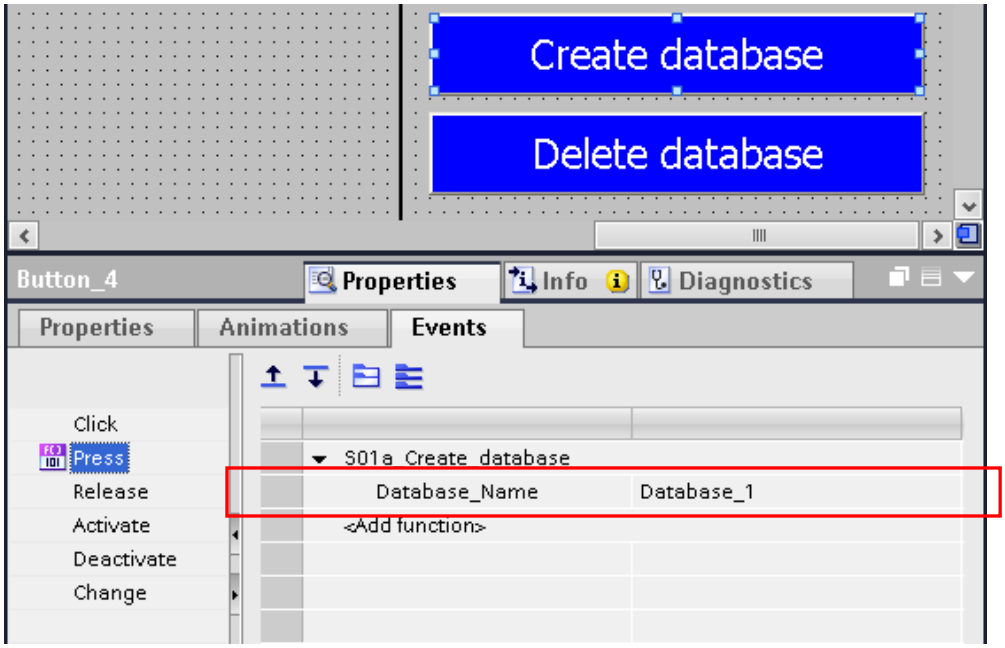
Lines 15 to **16:**
The "**ADODB.Connection**" object is required in order to establish a connection to an SQL data source. "**ADODB.Recordset**" can be used to create, edit or delete databases or tables, for example.

Line 19:
The **Provider** and **Name** of the data source must be known in order to open it.
In this example: Provider: MSDASQL
Name (DSN): Database_Name *)

The connection to the data source is established using the "**Open**" method of the "**conn**" object tag.

*) The data source name (DSN) is transferred to the scripts in this example as "**Parameter**". This means: The data source name is transferred to the script only when the script is called (see figure below).

Figure 2-1



The Data Source Name (DSN) links the configuration parameters for the communication with a specific database. Refer also to the information in section **1.4**.

Lines 22 to **27:**
Should a runtime error occur during connection, the following error routine is processed in which a system message is output and the script is ended immediately.

2.2 Creating/Deleting an SQL Database

2.2.1 Creating an SQL Database

Script "S01a_Create_database"

A new database is created with the script.

Description of the commands used

Lines **30** to **34**:
The SQL command "**CREATE DATABASE**" and the name of the database are used to create a new database employing the "**Execute(SQL Table)**" method of the "**conn**" object tag.

Lines **36** to **45**:
Error routine: If, for example, the name of the database already exists, a relevant error message is output.

Line **47**:
The "**Close**" method is used to disconnect the connection to the data source.

2.2.2 Deleting an SQL Database

Script "S01b_Delete_database"

A database is deleted with the script.

Description of the commands used

Lines **30** to **34**:
The SQL command "**DROP DATABASE**" and the name of the database are used to delete a database employing the "**Execute(SQL Table)**" method of the "**conn**" object tag.

Lines **36** to **45**:
Error routine: If, for example, the name of the database does not exist, a relevant error message is output.

Line **47**:
The "**Close**" method is used to disconnect the connection to the data source.

2.3 Editing Tables and Data Records

General information

You can create tables in an SQL database. You can create the table below by means of the attached project.

The column headers (columns 1 to 3) and the associated values in the cells can be predefined through the configured screens.

The column header for data record numbers cannot be edited. You can specify a data record number by means of the cells.

Table 2-1

No. *) (data record no.)	Column 1	Column 2	Column 3
1	Data type Char (30)	Data type SMALLINT	Data type SMALLINT
2	Auto 1	123	456
X	…	…	…

SQL instruction "SELECT * FROM"

All the data records from the table in the database are initially selected using the SQL instruction "**SELECT * FROM**" and the name of the table (see yellow entries in the table below).

Table 2-2

No. (data record no.)	Column 1	Column 2	Column 3
1	Value 10	Value 20	Value 30
2	Value 11	Value 21	Value 31
3	Value 12	Value 22	Value 32

SQL instruction "WHERE"

Using the command extension "**WHERE**" and specifying a column / data record no., **only** the specified data record (line) in the table is selected (see yellow entries in the table below).

A "new" data record thus corresponds to a new entry in a table line.

Table 2-3

No. (data record no.)	Column 1	Column 2	Column 3
1	Value 10	Value 20	Value 30
2	Value 11	Value 21	Value 31
3	Value 12	Value 22	Value 32

2.3.1 Creating a New Table

Script "S02_Create_new_table"

A new table is created in the database with the script.

Description of the commands used

Lines **30** to **37**:
The SQL instruction "**CREATE TABLE**" and the corresponding syntax (table structure) are used to create an SQL table.

The function is executed with the "**Execute(SQL Table)**" method of the "**conn**" object tag.

Lines **39** to **48**:
Error routine: If, for example, the name of the table already exists, a relevant error message is output.

Line **50**:
The "**Close**" method is used to disconnect the connection to the data source.

2.3.2 Writing a New Data Record to a Table

Script "S03_Write_data_record_into_a_table"

A new data record is added to a table with the script.

Description of the commands used

Lines **30** to **35**:
The SQL command "**SELECT * FROM**" is used in conjunction with the name of the table and the extension "**WHERE**" to select the appropriate data record.

The function is executed with the "**Execute(SQL Table)**" method of the "**conn**" object tag.

Lines **37** to **46**:
Error routine: If, for example, the name of the table does not exist, a relevant error message is output.

Lines **48** to **60**:
The instructions (rst.**EOF** and rst.**BOF**) check whether the specified data record already exists ("EOF" = End of File and "BOF" = Begin of File). If the data record already exists, a system message is output. Otherwise the specified data record is written to the table.

The SQL command "**INSERT INTO**" and the name of the SQL table are required in order to add a data record. After the "**VALUES**" extension all the parameters (tags) that are to be entered in the table are executed. The individual parameters are separated by a comma.

The function is executed with the "**Execute(SQL Table)**" method of the "**conn**" object tag.

Line **62**:
The "**Close**" method is used to disconnect the connection to the data source.

2.3.3 Reading Data Records from a Table

Script "S04_Read_data_record_from_a_table"

A data record is read from a table with the script.

Description of the commands used

Lines **30** to **35**:
The SQL command "**SELECT * FROM**" is used in conjunction with the name of the table and the extension "**WHERE**" to select the appropriate data record.

The function is executed with the "**Execute(SQL Table)**" method of the "**conn**" object tag.

Lines **37** to **46**:
Error routine: If, for example, the name of the table does not exist, a relevant error message is output.

Lines **48** to **65**:
The instructions (rst.**EOF** and rst.**BOF**) check whether the data record to be called already exists ("EOF" = End of File and "BOF" = Begin of File). If the data record does not exist, a system message is output. Otherwise the selected data record is output by means of the stored tag.

Line **67**:
The "**Close**" method is used to disconnect the connection to the data source.

2.3.4 Editing Data Records in a Table

Script "S05_Edit_data_record"

A data record can be edited in a table with the script.

Description of the commands used

Lines **30** to **35**:
The SQL command "**SELECT * FROM**" is used in conjunction with the name of the table and the extension "**WHERE**" to select the appropriate data record.

The function is executed with the "**Execute(SQL Table)**" method of the "**conn**" object tag.

Lines **37** to **46**:
Error routine: If, for example, the name of the table does not exist, a relevant error message is output.

Lines **48** to **61**:
The instructions (rst.**EOF** and rst.**BOF**) check whether the data record to be called already exists ("EOF" = End of File and "BOF" = Begin of File). If the data record does not exist, a system message is output. Otherwise the selected data record is output by means of the stored tag.

Lines **63** to **67**:
The specified values are assigned to the existing data record using the SQL instruction "**UPDATE**" and the name of the SQL table with the command extension "**Set**".

The function is executed with the "**Execute(SQL Table)**" method of the "**conn**" object tag.

Lines **69** to **78**:
Error routine: If, for example, the name of the table does not exist, a relevant error message is output.

Line **80**:
The "**Close**" method is used to disconnect the connection to the data source.

2.3.5 Deleting Data Records from a Table

Script "S06_Delete_data_record"

A data record in a table can be deleted with the script.

Description of the commands used

Lines **30** to **34**:
The SQL command "**DELETE FROM**" is used in conjunction with the name of the table and the extension "**WHERE**" to select the appropriate data record.

The function is executed with the "**Execute(SQL Table)**" method of the "**conn**" object tag.

Lines **36** to **45**:
Error routine: If, for example, the name of the table does not exist, a relevant error message is output.

Line **47**:
Upon completion of the function a system message is output. **Note:**
No check is made before execution of the function as to whether or not the selected data record is available.

Line **49**:
The "**Close**" method is used to disconnect the connection to the data source.

Note:
There is no safety query for confirmation that you really wish to delete this data record. The data record is deleted irretrievably.

2.3.6 Reading All Data Records from a Table

Script "S07_Show_all_entries_of_a_table"

All the data records can be read from a table with the script.

Description of the commands used

Lines **30** to **34**:
The SQL command "**SELECT * FROM**" is used in conjunction with the name of the table to select all the data records in the table.

The function is executed with the "**Execute(SQL Table)**" method of the "**conn**" object tag.

Lines **36** to **40**:
The SQL command "**SELECT * FROM**" is used in conjunction with the name of the table and the extension "**ORDER BY**" to sort the table in ascending order.

The function is executed with the "**Execute(SQL Table)**" method of the "**conn**" object tag.

Lines **42** to **51**:
Error routine: If, for example, the name of the table does not exist, a relevant error message is output.

Lines **53** to **107**:
The instructions (rst.**EOF** and rst.**BOF**) check whether the data record to be called already exists ("EOF" = End of File and "BOF" = Begin of File). If the data record does not exist, a system message is output.

The number of entries in the table is entered with the "**Do...Loop Until**" loop and the "**MoveNext**" instruction. The pointer is then reset to the first entry in the table with "**MoveFirst**" of the "**rst**" object tag.

Only a maximum of six entries in the table are shown simultaneously in this example. The table section can be moved here using the arrow buttons. The value of the "**Tab**" tag may vary between zero and the number of entries minus the number of data records to be displayed. When the table extract is moved, the pointer for the SQL table must also be moved. This is done with the "**MoveNext**" instruction.

Line **110**:
The "**Close**" method is used to disconnect the connection to the data source.

2.3.7 Copying a Table

Script "S08_Copy_table"

A table can be copied with the script.

Description of the commands used

Lines **30** to **34**:
The SQL command "**SELECT * INTO**" and the extension "**FROM**" are used to copy all the data records of the selected table to the new table.

The function is executed with the "**Execute(SQL Table)**" method of the "**conn**" object tag.

Lines **36** to **45**:
Error routine: If, for example, the name of the table does not exist, a relevant error message is output.

Line **47**:
The "**Close**" method is used to disconnect the connection to the data source.

2.3.8 Deleting Tables

Script "S09_Delete_table"

A table can be deleted with the script.

Description of the commands used

Lines **30** to **34**:
The SQL command "**DROP TABLE**" is used in conjunction with the name of the table to delete a table from the database.

The function is executed with the "**Execute(SQL Table)**" method of the "**conn**" object tag.

Lines **36** to **45**:
Error routine: If, for example, the name of the table does not exist, a relevant error message is output.

Line **47**:
The "**Close**" method is used to disconnect the connection to the data source.

Note:
There is no safety query for confirmation that you really wish to delete this table. **All** the data of the table is deleted irretrievably.

3 Using the Configured Screens

3.1 Requirements

To test the sample configuration you need an established connection to an SQL server.

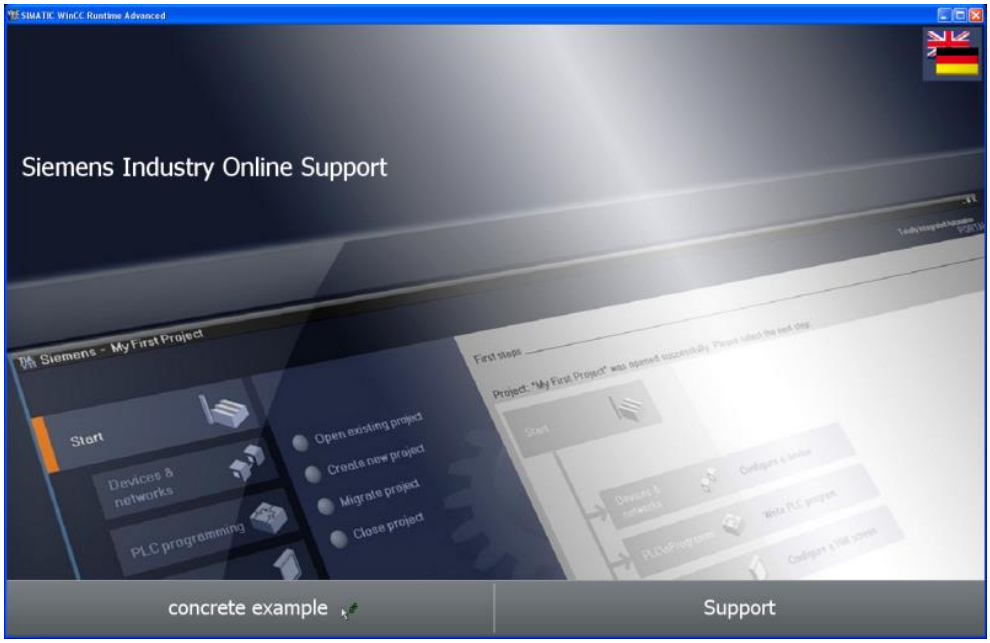
Refer here to section **1.4**.

The "**WINCCPLUS**" server is used as SQL server.
The data source name (DSN) is "**Database_1**".

3.2 Start Screen

After start-up of the sample project, the following page opens. Use this to call the configured screens with the "Sample Project > Topic A" button.

Figure 3-1



3.3 Creating/Deleting a Database

By means of this screen you can create a new database or delete a database.

Name of the database

Specify the name of the database by means of the IO field. Close the entry dialog with the "ENTER" button.

Create a database

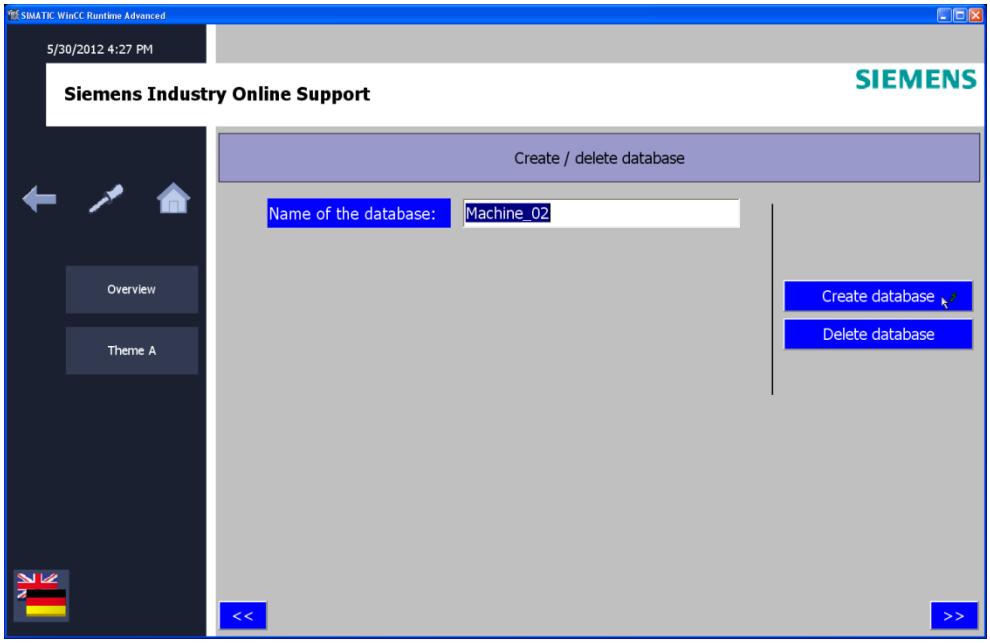
You create a new database with the specified name using the "**Create database**" button.

Delete a database

You delete a database with the specified name using the "**Delete database**" button.

Note Make sure that the database name does not include any spaces.

Figure 3-2



3.4 Adding a New SQL Table

A new table is created in a database with the screen.

Name of the database and the table

Specify the name of the database and the name of the table using the IO fields. Close the entry dialog with the "ENTER" button.

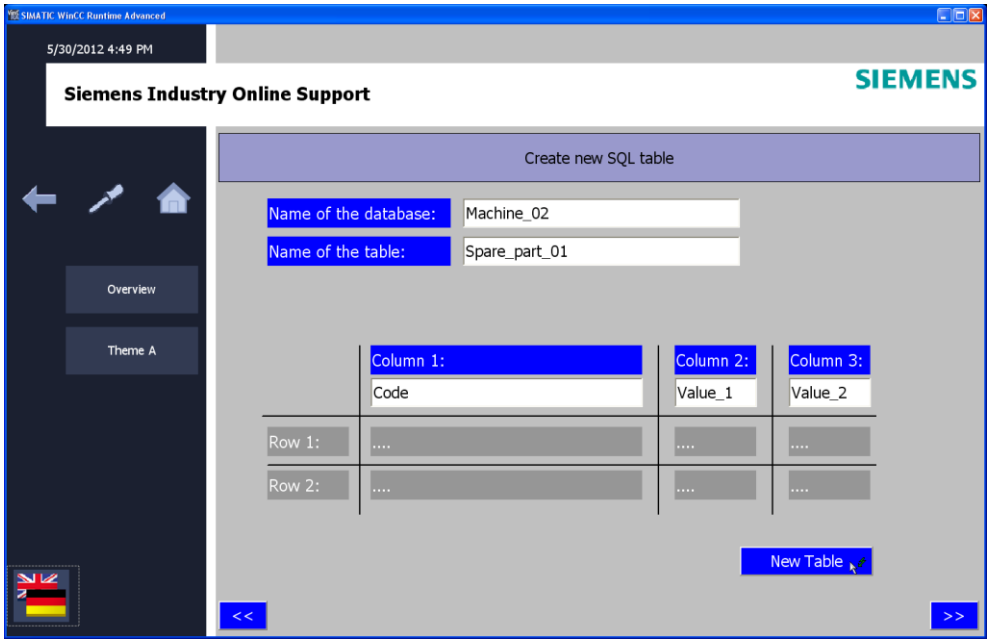
Table designation

Under "Column 1" etc. you enter the texts for the column headers.

You use the "**New table**" button to create a new table with the specified data in the database.

Note Make sure that the names do not include any spaces.

Figure 3-3



3.5 Adding a New Data Record to an SQL Table

A new table is created in a database with the screen.

Name of the database and the table

Specify the name of the database and the name of the table using the IO fields.
Close the entry dialog with the "ENTER" button.

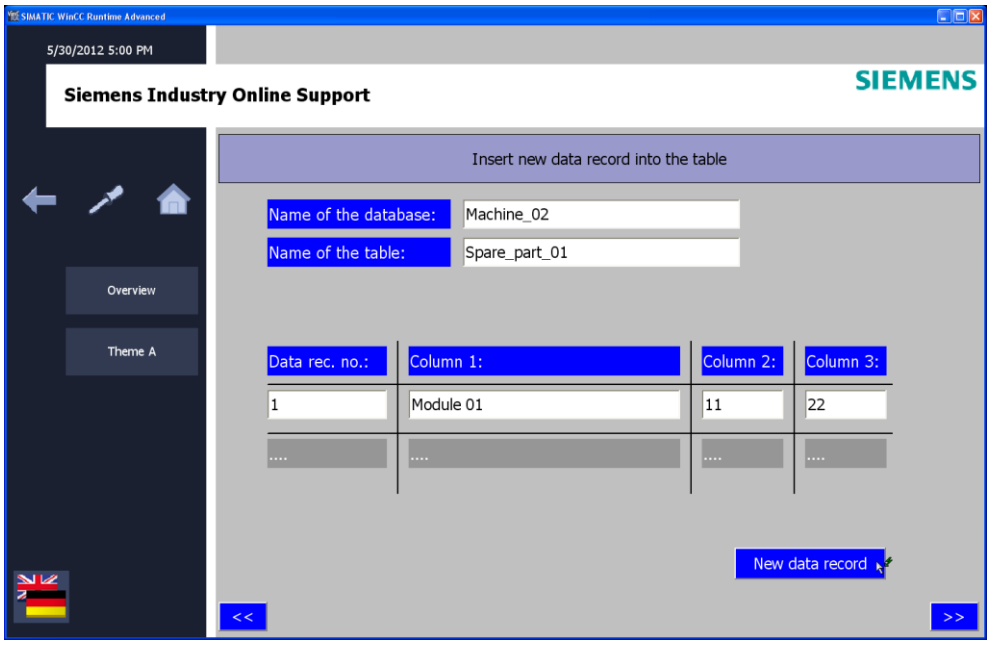
Enter the data record

"**Data record no.:**" column
Here you enter the number of the data record.

"**Column 1 to Column 3**" columns
Here you enter the values for the data record. In "Column 1" you can specify a string length of up to 30 characters.

You use the "**New data record**" button to insert the specified data record into the selected table.
If the specified data record number already exists, a system message is output. In this case you change the data record number.

Figure 3-4



3.6 Reading a Data Record from the SQL Table

You can use the screen to read a data record from a table.

Name of the database and the table

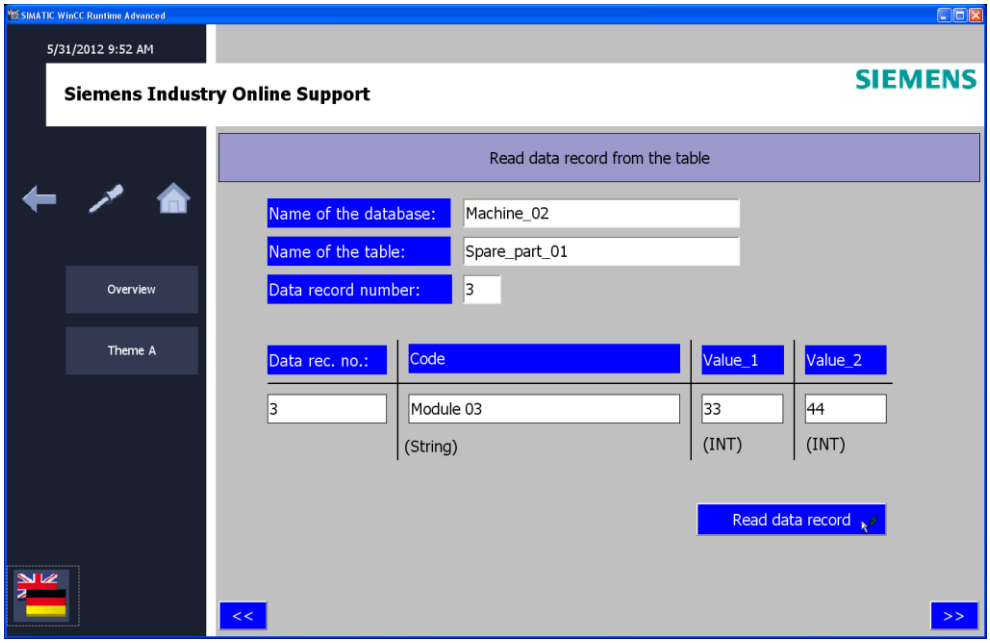
Specify the name of the database and the name of the table using the IO fields. Close the entry dialog with the "ENTER" button.

Edit data record

In the IO field you enter the data record number to be read out.

You use the "**Read data record**" button to read out the specified data record from the selected table.
If the specified data record number does not exist, a system message is output. In this case you change the data record number.

Figure 3-5



3.7 Editing a Data Record from the SQL Table

You can use the screen to edit a data record from a table.

Name of the database and the table

Specify the name of the database and the name of the table using the IO fields.
Close the entry dialog with the "ENTER" button.

Enter the data record

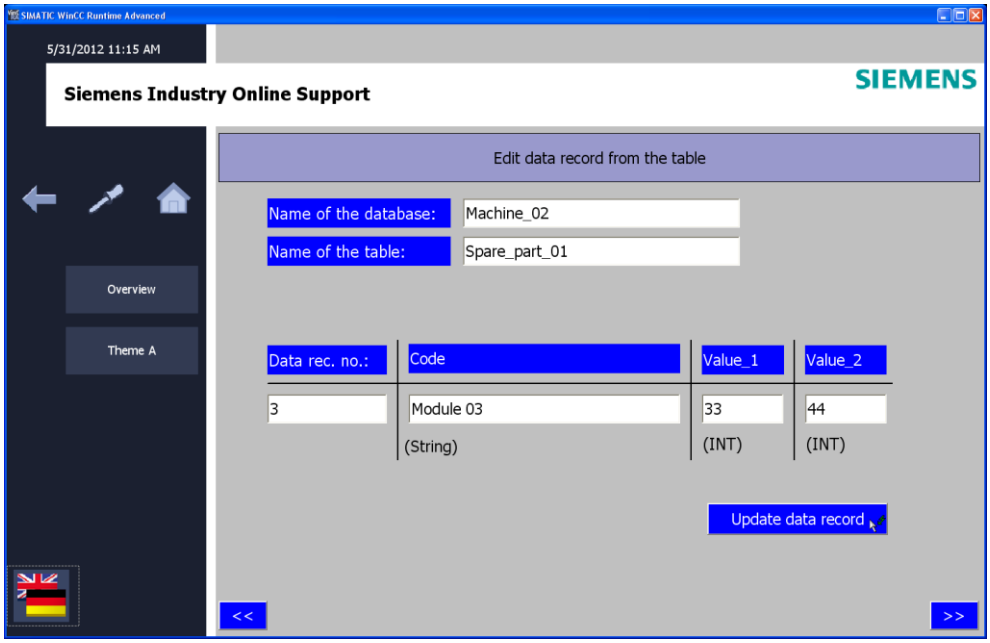
"**Data record no.:**" column
Here you enter the number of the data record to be changed.

"**Column 1 to Column 3**" columns
Here you enter the values for the data record. In "Column 1" you can specify a string length of up to 30 characters.

You use the "**Update data record**" button to update the specified data record in the selected table.
If the specified data record number does not exist, a system message is output. In this case you change the data record number.

Note To change the existing data record you first execute the "Read data record from the SQL table" function. The values read out are displayed automatically on this page and can be edited there.

Figure 3-6



3.8 Deleting a Data Record from the SQL Table

You can use the screen to delete a data record from a table.

Name of the database and the table

Specify the name of the database and the name of the table using the IO fields.
Close the entry dialog with the "ENTER" button.

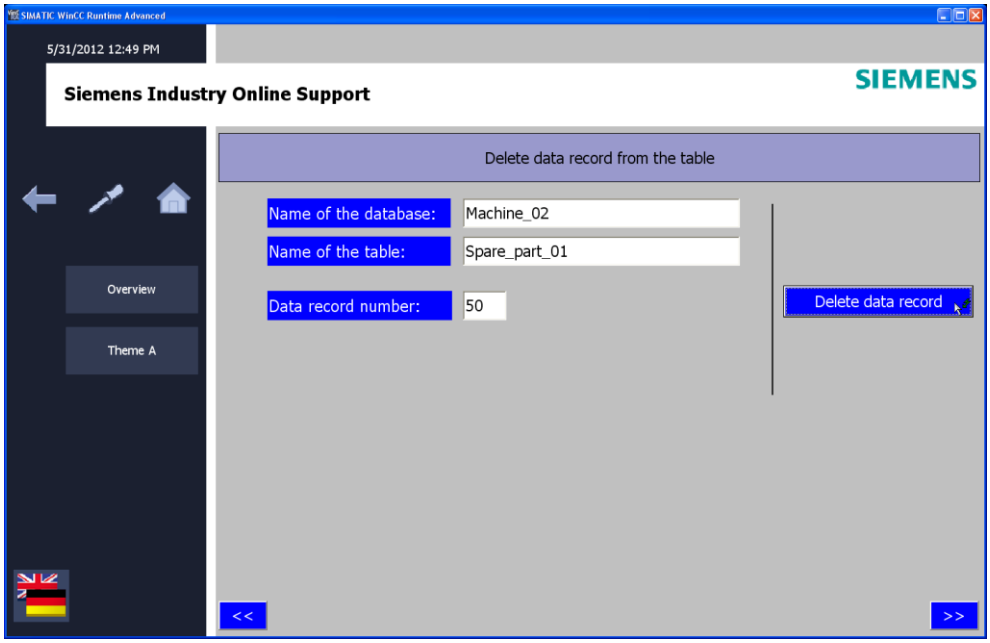
Edit data record

In the IO field you enter the data record number to be deleted.

You use the "**Delete data record**" button to delete the specified data record from the selected table.
If the specified data record number does not exist, a system message is output. In this case you change the data record number.

Note There is no safety query. The execution of this function deletes the specified data record irretrievably.

Figure 3-7



3.9 Reading All the Data Records from the SQL Table

You can use the screen to read all the data records from a table.

Name of the database and the table

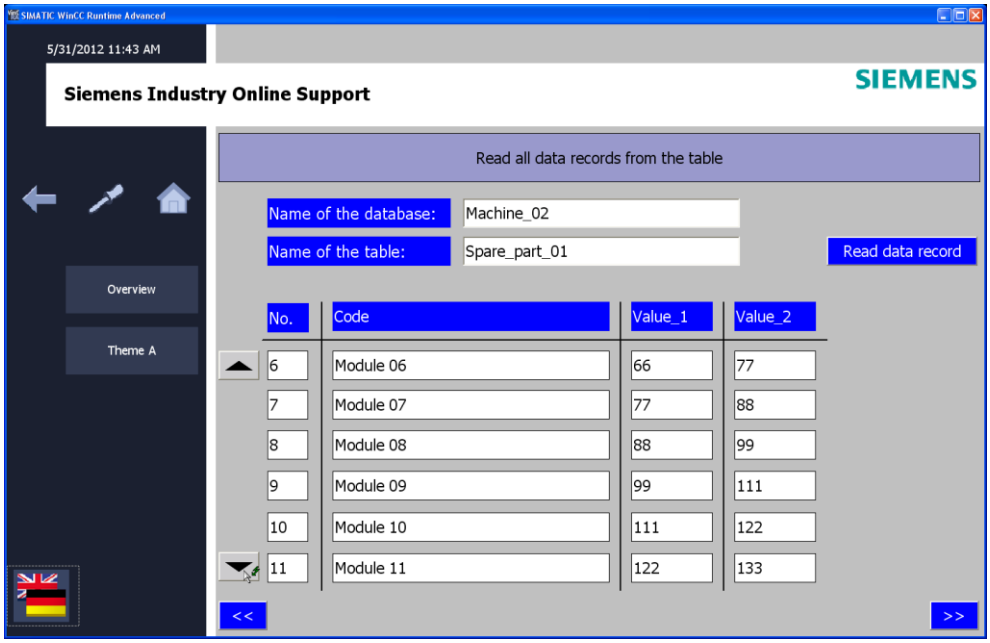
Specify the name of the database and the name of the table using the IO fields. Close the entry dialog with the "ENTER" button.

Read out the data record

You use the "**Read data record**" button to read out all the data records from the selected table.
If the specified data record number does not exist, a system message is output. In this case you change the data record number.

You can use the arrow keys to scroll in the table.

Figure 3-8



3.10 Copying the SQL Table

You can use the screen to copy a database table.

Name of the database and the table

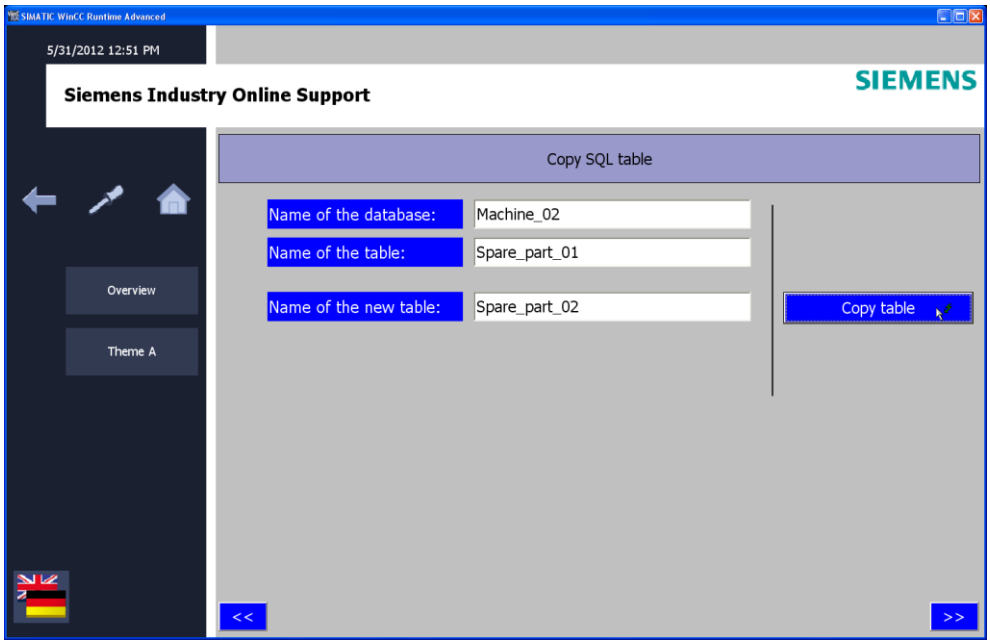
Specify the name of the database and the name of the table using the IO fields. Close the entry dialog with the "ENTER" button.

New table name

Enter a new table name in the IO field.

You use the "**Copy table**" button to copy the table and assign the specified name to it.

Figure 3-9



3.11 Deleting an SQL Table

You can use the screen to delete a database table.

Name of the database and the table

Specify the name of the database and the name of the table using the IO fields. Close the entry dialog with the "ENTER" button.

Delete table

You delete the table with the "**Delete table**" button.

Note There is no safety query. The execution of this function deletes the table irretrievably.

Figure 3-10