Connecting a S7-1500 to a SQL Database

TIA Portal V16 / S7-1500 / Microsoft SQL / TDS

https://support.industry.siemens.com/cs/ww/en/view/109779336

Siemens Industry Online Support

Legal information

Use of application examples

Application examples illustrate the solution of automation tasks through an interaction of several components in the form of text, graphics and/or software modules. The application examples are a free service by Siemens AG and/or a subsidiary of Siemens AG (“Siemens”). They are non-binding and make no claim to completeness or functionality regarding configuration and equipment. The application examples merely offer help with typical tasks; they do not constitute customer-specific solutions. You yourself are responsible for the proper and safe operation of the products in accordance with applicable regulations and must also check the function of the respective application example and customize it for your system.

Siemens grants you the non-exclusive, non-sublicensable and non-transferable right to have the application examples used by technically trained personnel. Any change to the application examples is your responsibility. Sharing the application examples with third parties or copying the application examples or excerpts thereof is permitted only in combination with your own products. The application examples are not required to undergo the customary tests and quality inspections of a chargeable product; they may have functional and performance defects as well as errors. It is your responsibility to use them in such a manner that any malfunctions that may occur do not result in property damage or injury to persons.

Disclaimer of liability

Siemens shall not assume any liability, for any legal reason whatsoever, including, without limitation, liability for the usability, availability, completeness and freedom from defects of the application examples as well as for related information, configuration and performance data and any damage caused thereby. This shall not apply in cases of mandatory liability, for example under the German Product Liability Act, or in cases of intent, gross negligence, or culpable loss of life, bodily injury or damage to health, non-compliance with a guarantee, fraudulent non-disclosure of a defect, or culpable breach of material contractual obligations. Claims for damages arising from a breach of material contractual obligations shall however be limited to the foreseeable damage typical of the type of agreement, unless liability arises from intent or gross negligence or is based on loss of life, bodily injury or damage to health. The foregoing provisions do not imply any change in the burden of proof to your detriment. You shall indemnify Siemens against existing or future claims of third parties in this connection except where Siemens is mandatorily liable.

By using the application examples you acknowledge that Siemens cannot be held liable for any damage beyond the liability provisions described.

Other information

Siemens reserves the right to make changes to the application examples at any time without notice. In case of discrepancies between the suggestions in the application examples and other Siemens publications such as catalogs, the content of the other documentation shall have precedence.

The Siemens terms of use (https://support.industry.siemens.com) shall also 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 constitute one element of such a concept.

Customers are responsible for preventing unauthorized access to their plants, systems, machines and networks. Such systems, machines and components should only be connected to an enterprise network or the internet if and to the extent such a connection is necessary and only when appropriate security measures (e.g. firewalls and/or network segmentation) are in place.

For additional information on industrial security measures that may be implemented, please visit https://www.siemens.com/industrialsecurity.

Siemens’ products and solutions undergo continuous development to make them more secure. Siemens strongly recommends that product updates are applied as soon as they are available and that the latest product versions are used. Use of product versions that are no longer supported, and failure to apply the latest updates may increase customer’s exposure to cyber threats.

To stay informed about product updates, subscribe to the Siemens Industrial Security RSS Feed at: http://www.siemens.com/industrialsecurity.

Table of contents

1 Introduction

1.1 Overview

Initial situation

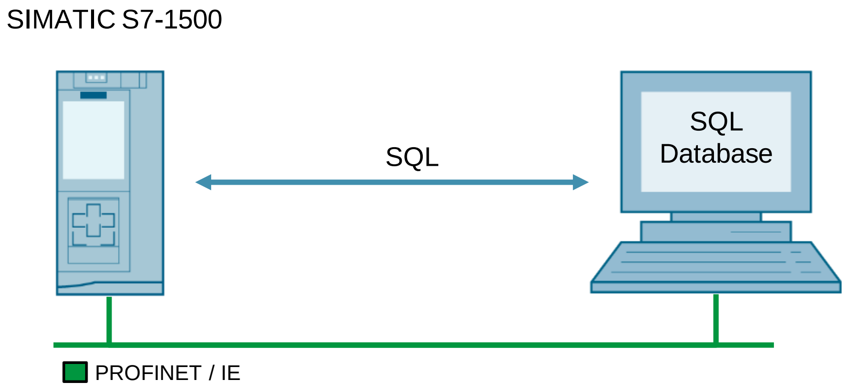
The **T**abular **D**ata **S**tream protocol (TDS) gives you the ability to establish a direct connection with a Microsoft SQL server. Using TDS, you can log in to an SQL server database and transmit SQL instructions. In this way it is possible to read data from the database, or send them to the database for storage.

This application example demonstrates how a SIMATIC S7-1500 establishes a connection to a Microsoft SQL server via TDS and sends data to a database using "Open User Communication blocks" (TCON, TSEND, TRCV and TDISCON).

Overview of the application example

The figure below provides an overview of the application example:

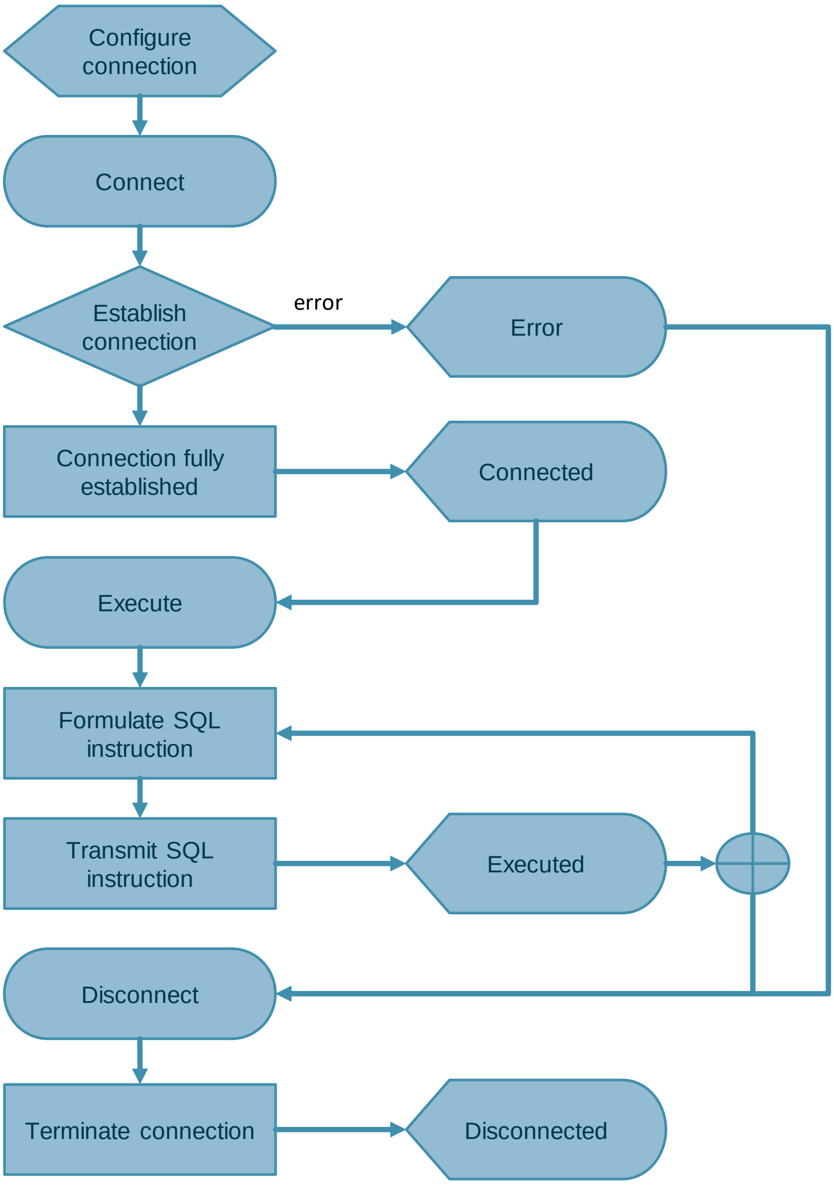
Figure 1-1



1.2 Principle of operation

The following figure shows the principle of operation, structure and states of the function block "LSql_Microsoft".

Figure 1-2



1.3 Components Used

The following hardware and software components were used to create this application example:

Table 1-1

Components	Quantity	Item number	Note
STEP 7 Professional V16 Update 2	1	6ES7822-1AA06-0YA7	Engineering system
SIMATIC S7-1500 CPU 1511-1 PN	1	6ES7511-1AK01-0AB0	Alternatively, you can use any other SIMATIC S7-1500 or ET 200SP CPU of firmware V2.5 or later.

This application example consists of the following components:

Table 1-2

Components	File name	Note
Project	109779336_SQL_S7_1500_CODE_V20.zip	This zipped file contains the STEP 7 project of the application example for the S7-1500.
Documentation	109779336_SQL_S7_1500_DOC_en_V20.pdf	This document

2 Engineering

2.1 Interface description

Function description

The function block "LSql_Microsoft" emulates the TDS protocol on the basis of "Open User Communication blocks". It facilitates the following actions:

- Logging in to a Microsoft SQL server database (enable, connSettings)
- Transmitting SQL instructions (sqlCommand, executeSqlCommand)
- Receipt of read Data (data, dataReceived)

Internally, the block works with helper blocks. They are not explained here in more detail.

Block interface

The following figure shows the interfaces of the function block "LSql_Microsoft" and the associated data types.

Abbildung 2-1: LSql_Microsoft

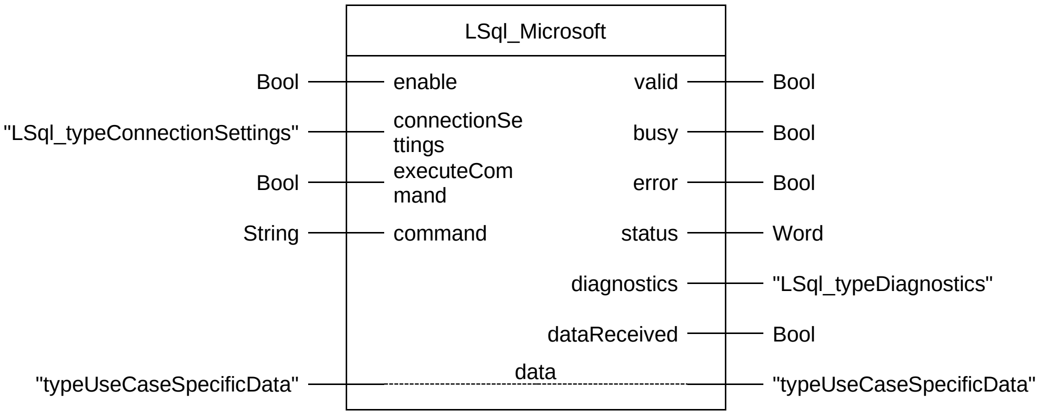


Tabelle 2-1: Parameter of LSql_Microsoft

Name	P-Type	Data Type	Comment
enable	IN	Bool	TRUE: Enable functionality of FB
connectionSettings	IN	"LSql_typeConnectionSettings"	used for logging on to database
executeCommand	IN	Bool	TRUE: execute sql command once
command	IN	String	SQL command to execute when executeSqlBatch = TRUE
valid	OUT	Bool	TRUE: Valid set of output values available at the FB
busy	OUT	Bool	TRUE: FB is not finished and new output values can be expected
error	OUT	Bool	TRUE: An error occurred during the execution of the FB
status	OUT	Word	16#0000 - 16#7FFF: Status of the FB, 16#8000 - 16#FFFF: Error identification
diagnostics	OUT	"LSql_typeDiagnostics"	Diagnostics information of FB (optional)

Name	P-Type	Data Type	Comment
dataReceived	OUT	Bool	TRUE: new data was received
data	IN_OUT	"typeUseCaseSpecificData"	response recieved from SQL server

Tabelle 2-2: Parameter of LSql_typeLoginInformation

Name	Type	Value	Comment
hostName	String	DEF_VAL	optional: name of local host
userName	String	DEF_VAL	mandatory: username for login to database
password	String	DEF_VAL	mandatory: password for login to database
appName	String	DEF_VAL	optional: name of app connecting to database
serverName	String	DEF_VAL	mandatory: servername of database
libraryName	String	DEF_VAL	optional: name of ui
local	String	DEF_VAL	optional: language used in ui
databaseName	String	DEF_VAL	mandatory: database to read or write from
sspi	String	DEF_VAL	optional / not supported: encryption via sspi
attachDbfile	String	DEF_VAL	optional: filename to add at transfer
changePassword	String	DEF_VAL	optional: new password if old one should be changed

Tabelle 2-3: Parameter of typeUseCaseSpecificData

Name	Type	Value	Comment
header	"LSql_typeTDSPacketHeader"	DEF_VAL	TDS Protocol header
tokenColumnMetaData	Array[0..37] of Byte	DEF_VAL	information about the columns returned in the tokenRows - depends on user data
tokenRows	Array[0..14] of "typeUseCaseSpecificTokenRow"	DEF_VAL	Rows returned for the SQL Batch - structure depends on user data!
bytes	Array[0..867] of Byte	DEF_VAL	[No comment]

Tabelle 2-4: Parameter of LSql_typeDiagnostics

Name	Type	Value	Comment
status	Word	DEF_VAL	Status of the Block or error identification when error occurred
subfunctionStatus	Word	DEF_VAL	Status or return value of called FB's, FC's and system blocks
stateNumber	DInt	DEF_VAL	State in the state machine of the block where the error occurred

Tabelle 2-5: Parameter of LSql_typeConnectionSettings

Name	Type	Value	Comment
interfaceSettings	TCON_IP_v4	DEF_VAL	[No comment]
loginInformation	"LSql_typeLoginInformation"	DEF_VAL	[No comment]

2.2 Integration into the user project

Requirements

The following requirements apply to the use of the application example:

- S7-1500 firmware V2.5 or later
- Microsoft SQL server is fully configured
- S7-1500 and Microsoft SQL server are in the same subnet.

NOTE This block is also functional with an S7-1200 from firmware V4.4 or higher.

Restrictions

The following restrictions apply for this application example:

- The application example only works with the tested hardware and software versions.
- Using Open User Communication with an S7-1500, a maximum of 65536 bytes per command can be sent or received.
- Using Open User Communication with an S7-1200, a maximum of 8192 bytes per command can be sent or received.
- The block "LSql_Microsoft" may be called no more than once per Microsoft SQL server.

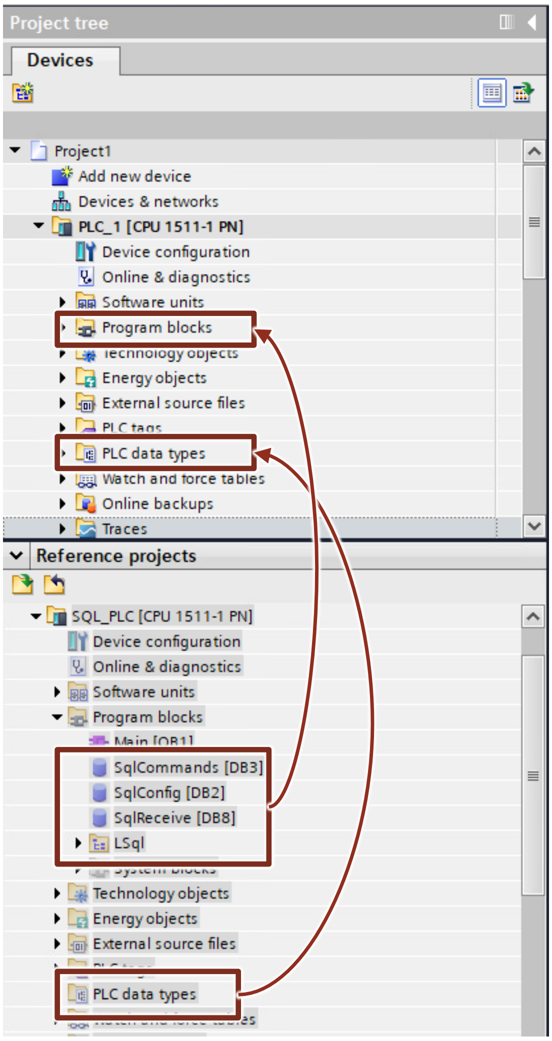
Integrating project components into the user project

Copy the following project components into your TIA Portal project:

- "LSql" block folder
- Data block "SqlConfig"
- Data block "SqlCommand"
- Data block "SqlReceive"
- PLC data types folder

Here, working in the "Reference projects" view is recommended.

Figure 2-2



Project tree

Devices

Project1
Add new device
Devices & networks
PLC_1 [CPU 1511-1 PN]
Device configuration
Online & diagnostics
Software units
Program blocks
Technology objects
Energy objects
External source files
PLC tags
PLC data types
Watch and force tables
Online backups
Traces

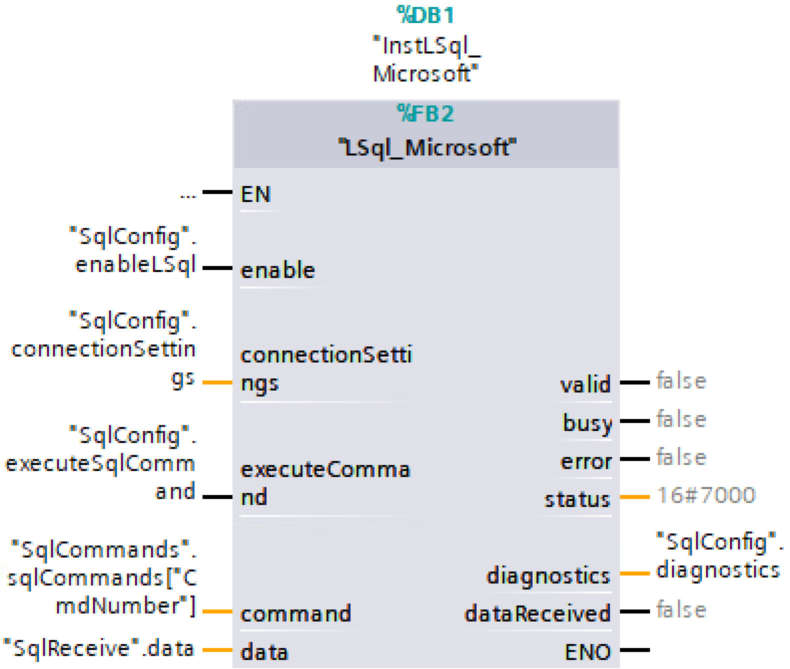
Reference projects

SQL_PLC [CPU 1511-1 PN]
Device configuration
Online & diagnostics
Software units
Program blocks
Main [OB1]
SqlCommands [DB3]
SqlConfig [DB2]
SqlReceive [DB8]
LSql
System blocks
Technology objects
Energy objects
External source files
PLC tags
PLC data types
Watch and force tables

Interconnecting the parameters of the "LSql_Microsoft" block

Call the block "LSql_Microsoft" in a cyclic block, e.g. "Main [OB1]" and interconnect the inputs and outputs as seen in the following figure.

Figure 2-3


%DB1
"InstLSql_Microsoft"
%FB2
"LSql_Microsoft"
... EN
"SqlConfig".enableLSql enable
"SqlConfig".connectionSettings connectionSettings
valid false
busy false
error false
status 16#7000
"SqlConfig".executeSqlCommand executeCommand
"SqlCommands".sqlCommands["CmdNumber"] command
diagnostics "SqlConfig".diagnostics
dataReceived false
"SqlReceive".data data
ENO

NOTE

The wiring of the parameters described here is an essential requirement for operating the block. In this application example, data are transmitted from the controller to storage in the SQL server. It is not necessary to use an archive to do this.

The connection settings and login information must be cleared on the SQL server side in order to establish the connection. Further information on this topic can be found in chapter Settings in Microsoft SQL Server Express.

NOTE

The parameter "data" must be interconnected with an "Array of Byte" when using a S7-1200. A structure is not permitted.

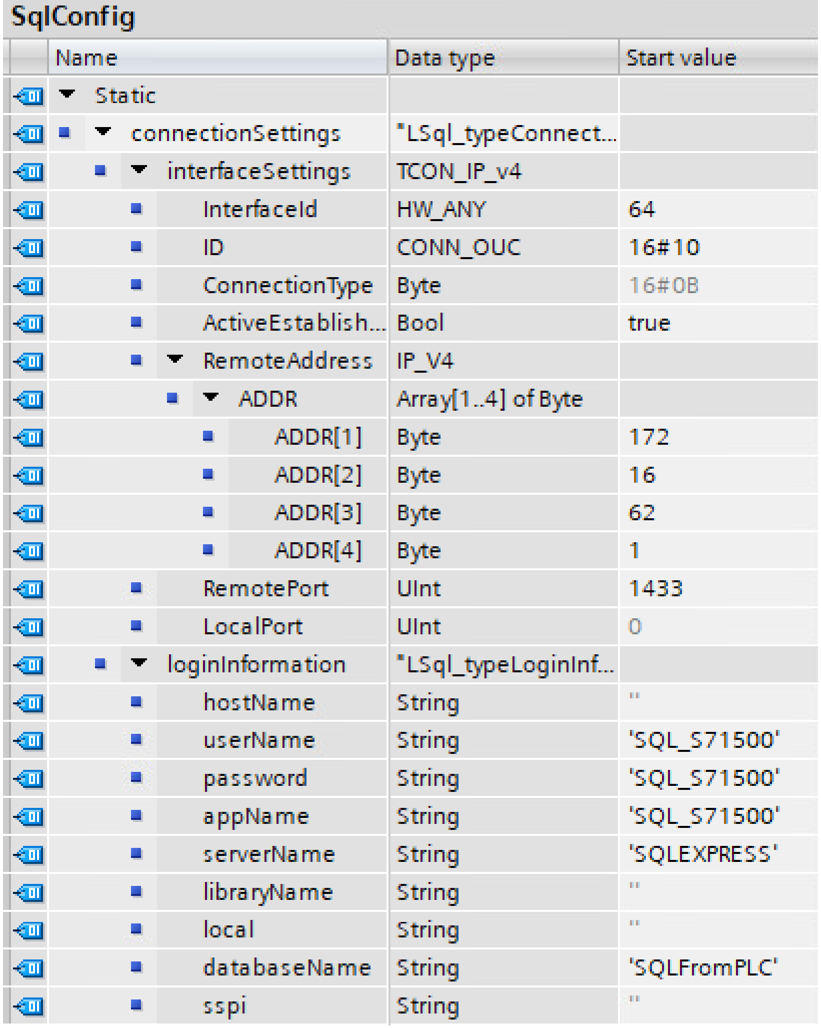
2.3 Operation of LSql_Microsoft

The block LSql_Microsoft is controlled via the inputs "enable" and "executeSqlCommand".

Establishing and terminating connection

The "enable" input controls the process of establishing and terminating a connection to the SQL server. "enable" must have the value "TRUE" as long as SQL instructions must be transmitted to the SQL server. If "enable" is set to "FALSE" then the connection to the SQL server is terminated.

In order to successfully establish a connection, the following parameters must be set in the data block "SqlConfig". The unfilled parameters are optional.



SqlConfig

Name	Data type	Start value
Static		
connectionSettings	"LSql_typeConnect...	
interfaceSettings	TCON_IP_v4	
InterfaceId	HW_ANY	64
ID	CONN_OUC	16#10
ConnectionType	Byte	16#0B
ActiveEstablish...	Bool	true
RemoteAddress	IP_V4	
ADDR	Array[1..4] of Byte	
ADDR[1]	Byte	172
ADDR[2]	Byte	16
ADDR[3]	Byte	62
ADDR[4]	Byte	1
RemotePort	UInt	1433
LocalPort	UInt	0
loginInformation	"LSql_typeLoginInf...	
hostName	String	''
userName	String	'SQL_S71500'
password	String	'SQL_S71500'
appName	String	'SQL_S71500'
serverName	String	'SQLEXPRESS'
libraryName	String	''
local	String	''
databaseName	String	'SQLFromPLC'
sspi	String	''

Table 2-6

	Parameter	Note
1.	IP address and port of the SQL server	Default port for Microsoft SQL server is 1433.
2.	SQL server login information	See Settings in Microsoft SQL Server Express
3.	Name of SQL server	In this application example: SQLEXPRESS
4.	Name of the database of the SQL server	An SQL server can contain multiple databases. Use this parameter to specify which database you wish to connect to.

Structure of the example database

The structure of the example database "SQLFromPLC" can be seen in the table below.

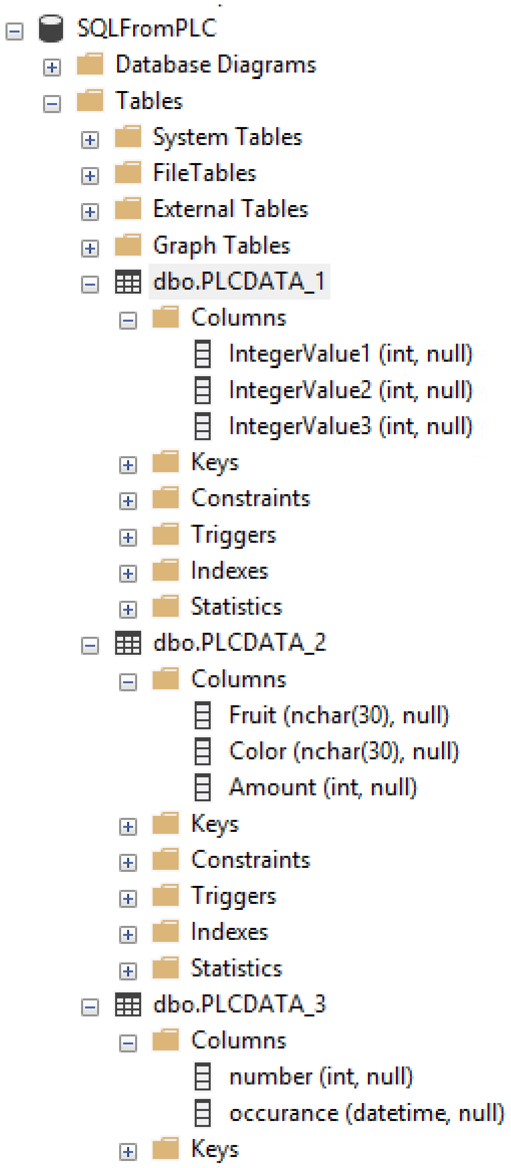
Table 2-7

Table	Line	Data type
PLCDATA_1	IntegerValue1	Int
	IntegerValue2	Int
	IntegerValue3	Int
PLCDATA_2	Fruit	Nchar(30)
	Color	Nchar(30)
	Amount	Int
PLCDATA_3	number	Int
	occurrence	datetime

The structure has been chosen to demonstrate how numbers, strings and time stamps are saved.

The following figure shows you the database in SQL Server Management Studio.

Figure 2-4



- SQLFromPLC
  - Database Diagrams
  - Tables
    - System Tables
    - FileTables
    - External Tables
    - Graph Tables
    - dbo.PLCDATA_1
      - Columns
        - IntegerValue1 (int, null)
        - IntegerValue2 (int, null)
        - IntegerValue3 (int, null)
      - Keys
      - Constraints
      - Triggers
      - Indexes
      - Statistics
    - dbo.PLCDATA_2
      - Columns
        - Fruit (nchar(30), null)
        - Color (nchar(30), null)
        - Amount (int, null)
      - Keys
      - Constraints
      - Triggers
      - Indexes
      - Statistics
    - dbo.PLCDATA_3
      - Columns
        - number (int, null)
        - occurance (datetime, null)
      - Keys

Transmitting SQL instructions

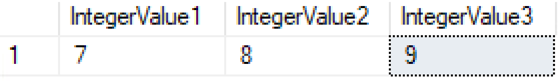
You will formulate and store an SQL instruction at the sqlCommand input. Once the controller has established a connection to the SQL server, you can transmit the SQL instruction to the SQL server with a positive edge at the executeSqlCommand input.

NOTE The example supports only the standard ASCII encoding.

You can add a new line to this database table with "**insert into PLCDATA_1**". "**Values (5,6,7)**" specifies the value that should be entered in the first (5), second (6) and third (7) column. The complex command is "**insert into PLCDATA_1 Values (5,6,7)**". The SQL instruction is transmitted to the database with a positive edge at the input "executeSqlCommand".

The following figure shows the contents of the table "PLCDATA_1" after this SQL instruction is executed.

Figure 2-5

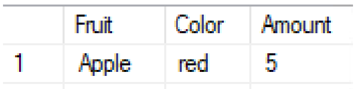


	IntegerValue1	IntegerValue2	IntegerValue3
1	7	8	9

You can add a new line to this database table with "**insert into PLCDATA_2**". "**Values ('Apple','red',5)**" specifies the value that should be entered in the first ('Apple'), second ('red') and third (5) column. The complex command is "**insert into PLCDATA_2 Values ('Apple','red',5)**". The SQL instruction is transmitted to the database with a positive edge at the input "executeSqlCommand".

The following figure shows the contents of the table "PLCDATA_2" after this SQL instruction is executed.

Figure 2-6

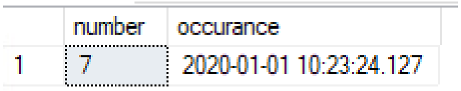


	Fruit	Color	Amount
1	Apple	red	5

You can add a new line to this database table with "**insert into PLCDATA_3**" "**Values (7, $'2020-01-01 10:23:24.125$')**" specifies the value that should be entered in the first (7) and second ('2020-01-01 10:23:24.125') column. The $ signs are necessary to send the " ' " character. The complex command is "**insert into PLCDATA_2 Values (7, $'2020-01-01 10:23:24.125$')**". The SQL instruction is transmitted to the database with a positive edge at the input "executeSqlCommand".

The following figure shows the contents of the table "PLCDATA_3" after this SQL instruction is executed.

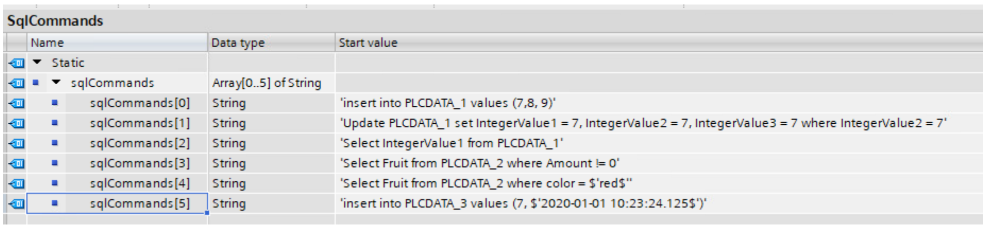
Figure 2-7



	number	occurance
1	7	2020-01-01 10:23:24.127

You can find additional example commands in the "SqlCommand" data block, as seen in the figure below.

Figure 2-8



SqlCommands

Name	Data type	Start value
▼ Static		
▼ sqlCommands	Array[0..5] of String	
sqlCommands[0]	String	'insert into PLCDATA_1 values (7,8, 9)'
sqlCommands[1]	String	'Update PLCDATA_1 set IntegerValue1 = 7, IntegerValue2 = 7, IntegerValue3 = 7 where IntegerValue2 = 7'
sqlCommands[2]	String	'Select IntegerValue1 from PLCDATA_1'
sqlCommands[3]	String	'Select Fruit from PLCDATA_2 where Amount != 0'
sqlCommands[4]	String	'Select Fruit from PLCDATA_2 where color = $'red$''
sqlCommands[5]	String	'insert into PLCDATA_3 values (7, $'2020-01-01 10:23:24.125$')'

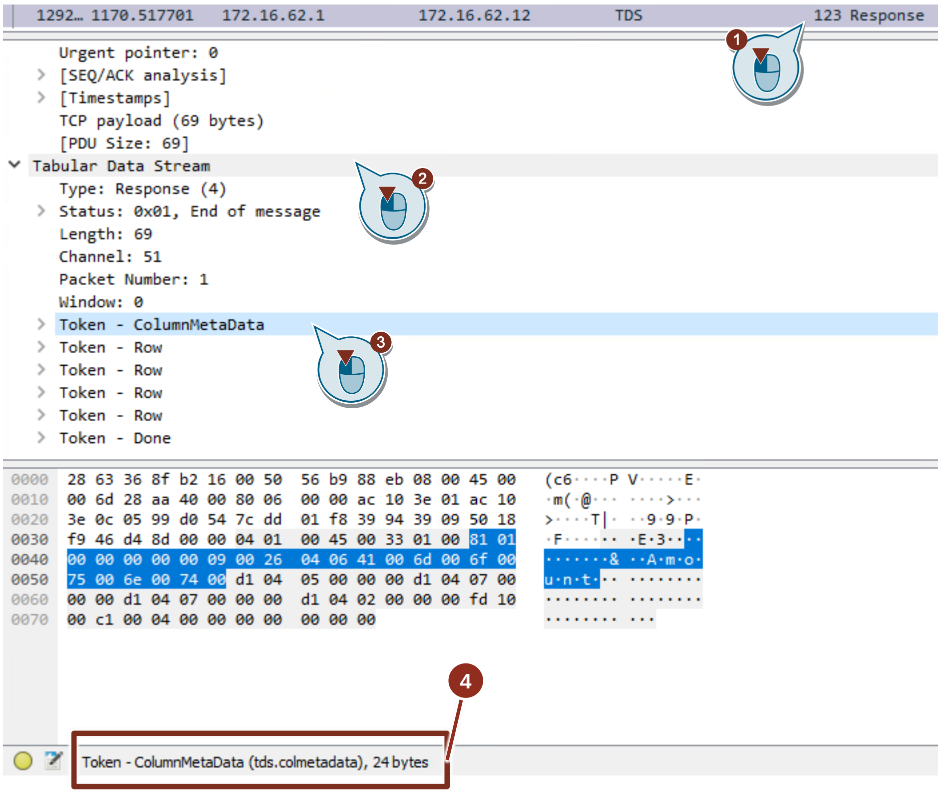
The "select" instruction

Using the "select" instruction, you can read data out of a database table and perform other operations on them in your controller. Below, we use an example to show how this instruction works and which modifications you will need to make for your query.

The example uses the following query:
"**Select Amount from PLCDATA_2 where Fruit = $'Apple$'**".

The data queried are stored in the "SqlReceive" data block in the data type structure "**typeUseCaseSpecificData**". You must modify these data types individually for each query. You can do this as follows:

1. Start a Wireshark recording to find the packet length of the metadata.
2. Run the query "**Select Amount from PLCDATA_2 where Fruit = $'Apple$'**".
3. Stop the Wireshark recording and search for the response frame from the SQL server using the filter "tds".
4. Select the frame and click on the line "Token – ColumnMetaData".



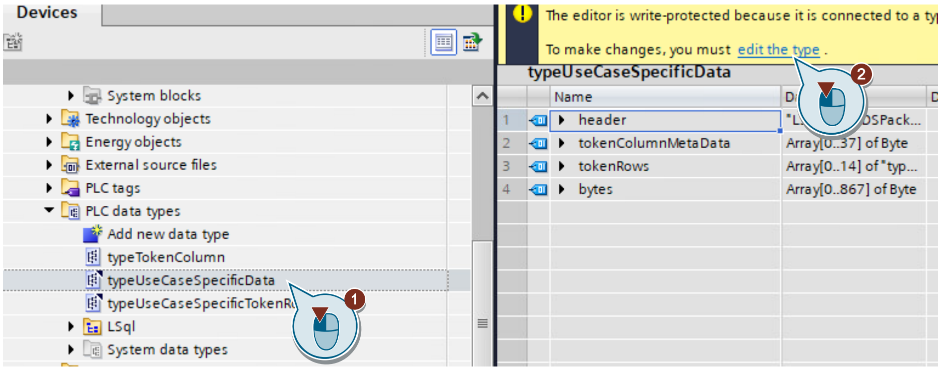
```
1292… 1170.517701   172.16.62.1          172.16.62.12          TDS                123 Response

      Urgent pointer: 0
    > [SEQ/ACK analysis]
    > [Timestamps]
      TCP payload (69 bytes)
      [PDU Size: 69]
  v Tabular Data Stream
      Type: Response (4)
    > Status: 0x01, End of message
      Length: 69
      Channel: 51
      Packet Number: 1
      Window: 0
    > Token - ColumnMetaData
    > Token - Row
    > Token - Row
    > Token - Row
    > Token - Row
    > Token - Done

0000  28 63 36 8f b2 16 00 50  56 b9 88 eb 08 00 45 00   (c6····P V·····E·
0010  00 6d 28 aa 40 00 80 06  00 00 ac 10 3e 01 ac 10   ·m(·@··· ····>···
0020  3e 0c 05 99 d0 54 7c dd  01 f8 39 94 39 09 50 18   >····T|· ··9·9·P·
0030  f9 46 d4 8d 00 00 04 01  00 45 00 33 01 00 81 01   ·F······ ·E·3····
0040  00 00 00 00 00 09 00 26  04 06 41 00 6d 00 6f 00   ·······& ··A·m·o·
0050  75 00 6e 00 74 00 d1 04  05 00 00 00 d1 04 07 00   u·n·t··· ········
0060  00 00 d1 04 07 00 00 00  d1 04 02 00 00 00 fd 10   ········ ········
0070  00 c1 00 04 00 00 00 00  00 00 00                  ········ ···

Token - ColumnMetaData (tds.colmetadata), 24 bytes
```

5. Now adjust the length of the data type structure "**typeUseCaseSpecificData**". To do this, you must edit the type of the example. This type can then be saved as a new version.



Devices

- System blocks
- Technology objects
- Energy objects
- External source files
- PLC tags
- PLC data types
  - Add new data type
  - typeTokenColumn
  - typeUseCaseSpecificData
  - typeUseCaseSpecificTokenR
  - LSql
  - System data types

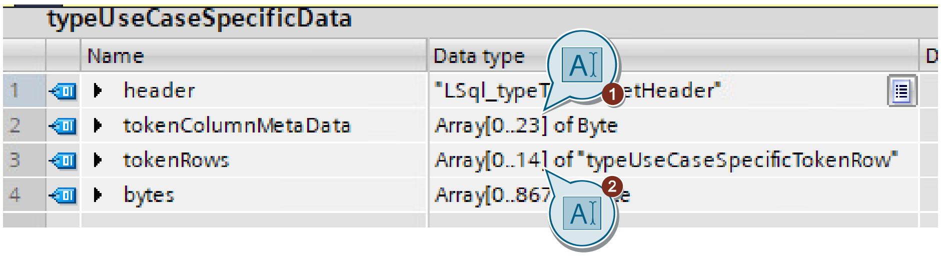
The editor is write-protected because it is connected to a ty

To make changes, you must edit the type .

typeUseCaseSpecificData

	Name	D
1	header	"L… OSPack…
2	tokenColumnMetaData	Array[0..37] of Byte
3	tokenRows	Array[0..14] of "typ…
4	bytes	Array[0..867] of Byte

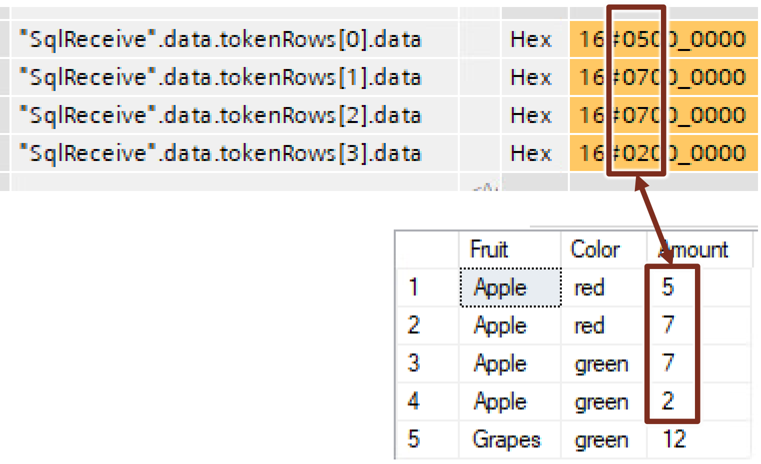
6. Now, for the array "tokenColumnMetaData", enter the length that was shown in Wireshark. In the example, it is "24 Bytes". If you expect more than 15 results, enlarge the array "tokenRows" to the appropriate size. If necessary, adapt the contained data type of the element "tokenRows.data" if you expect a string, for example.



typeUseCaseSpecificData

	Name	Data type
1	header	"LSql_typeT…etHeader"
2	tokenColumnMetaData	Array[0..23] of Byte
3	tokenRows	Array[0..14] of "typeUseCaseSpecificTokenRow"
4	bytes	Array[0..867…e

7. Download the changes to your controller and restart the query. You will now receive the queried data in the individual elements of the array "tokenRows[x].data". You can see the result of the example query in the figure below.



"SqlReceive".data.tokenRows[0].data	Hex	16#0500_0000
"SqlReceive".data.tokenRows[1].data	Hex	16#0700_0000
"SqlReceive".data.tokenRows[2].data	Hex	16#0700_0000
"SqlReceive".data.tokenRows[3].data	Hex	16#0200_0000

	Fruit	Color	Amount
1	Apple	red	5
2	Apple	red	7
3	Apple	green	7
4	Apple	green	2
5	Grapes	green	12

Long queries with Stored Procedures

Complex select queries can quickly exceed the 254 character limit. If you want to use queries longer than 254 characters, you have the option to call a "stored procedure". This is the most performant way to perform a long query of the database.

A stored procedure is a function that processes the stored queries and then returns the result to the user.

1. Create a "Stored Procedure" with any name (e. g. get_all_placed_orders).
2. Execute the query "**execute get_all_placed_orders**".
3. You will get back the result of the procedure.

2.4 Error handling

If the "LSql_Microsoft" has an error, inspect the output parameters "error", "status" and "diagnostics". "Error - TRUE" signals that an error occurred during processing. The "status" provides unambiguous information on the status of the block. "diagnostics" gives you detailed status and diagnostic information from subfunctions which the "LSql_Microsoft" block uses internally.

Per the status concept of the SIMATIC programming style guide used, the parameters "error" and the highest-value bit (MSB) of "status" (bit 15) are identical. The remaining bits are used for an error code which points unambiguously to the cause. The error codes are also stored as constants in the local data of the block.

Status and error message HEX Codes

The following table describes the most important aspects of the status and error messages from the block "LSql_Microsoft".

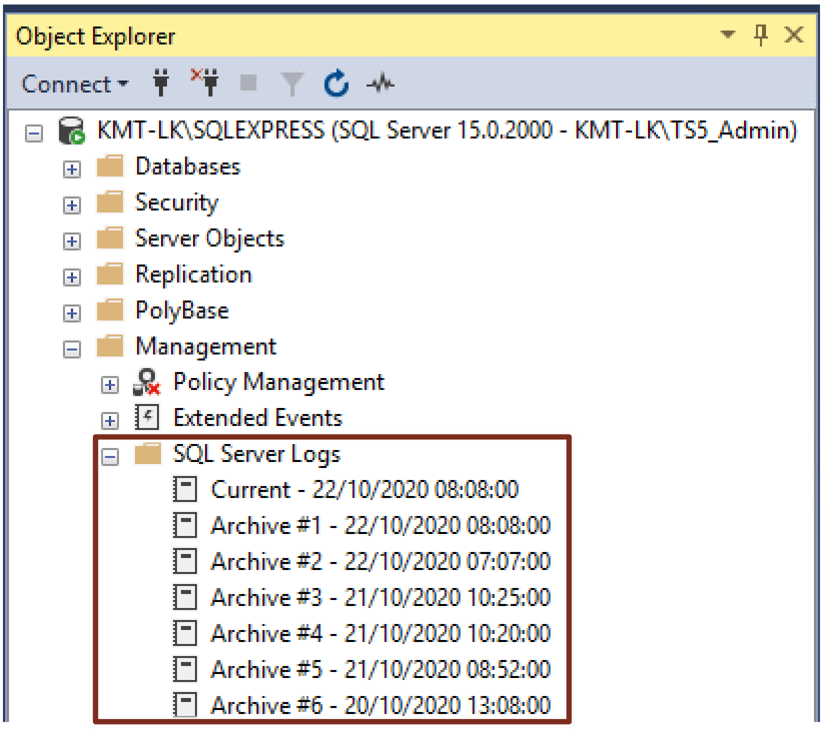
Table 2-8

Constant name/error message	Value	Meaning
STATUS_NO_CALL	7000	No job being currently processed
STATUS_FIRST_CALL	7001	First call after incoming new job (rising edge at "enable")
STATUS_SUBSEQUENT_CALL	7002	Subsequent call during active processing without further details
STATUS_TDISCON_SUCCESSFUL	7011	status from TDISCON called successfully
ERR_IN_BLOCK_OPERATION	8001	Error: Wrong operation of the function block
ERR_TDISCON_REMOTE_TERMINATION	80A3	Error: TDISCON, The connection is already terminated or doesn't exist
ERR_PARAMETERIZATION	8200	Error during parameterization
ERR_PROCESSING_EXTERN	8400	Error when processing from outside (e. g. wrong I/O signals)
ERR_UNDEFINED_STATE	8600	Error due to an undefined state in state machine
ERR_DISCONNECT	8601	Error when calling TDISCON
ERR_CONNECT	8602	Error when calling TCON
ERR_PRELOGIN	8603	Error while executing prelogin
ERR_LOGIN	8604	Error while executing login
ERR_TRCV	8605	Error when calling TRCV
ERR_ARCHIVE	8606	Error: archiving in db
ERR_PRELOGIN_DATA	8607	Error when converting prelogin data to word
ERR_LOGIN_DATA	8608	Error when converting login data to word
ERR_SQLBATCH_DATA	8609	Error when converting SQL command data to word

Status and error messages from the SQL server

You can perform a detailed error analysis directly in "SQL Server Management Studio". To do this, go to the folder "Management > SQL Server Logs". There you can find the server logs where, among other things, error messages are stored.

Figure 2-9



Object Explorer

Connect

- KMT-LK\SQLEXPRESS (SQL Server 15.0.2000 - KMT-LK\TS5_Admin)
  - Databases
  - Security
  - Server Objects
  - Replication
  - PolyBase
  - Management
    - Policy Management
    - Extended Events
    - SQL Server Logs
      - Current - 22/10/2020 08:08:00
      - Archive #1 - 22/10/2020 08:08:00
      - Archive #2 - 22/10/2020 07:07:00
      - Archive #3 - 21/10/2020 10:25:00
      - Archive #4 - 21/10/2020 10:20:00
      - Archive #5 - 21/10/2020 08:52:00
      - Archive #6 - 20/10/2020 13:08:00

3 Useful information

3.1 Microsoft SQL Server Express basics

Microsoft SQL Server Express

The Microsoft SQL server is a high-performance database management system for SQL databases. The free Express version is designed for desktop and server applications. It supports up to 10 gigabytes of storage per database.

The 2017 version can be downloaded here: https://www.microsoft.com/en-US/download/details.aspx?id=55994

Microsoft SQL Server Management Studio

The free Microsoft SQL Server Management Studio provides tools for configuring, monitoring and managing instances or SQL Servers and databases. It makes it possible to send queries and scripts to databases in the form of SQL instructions. In this way you can enter new data to the database table, or read existing data.

Version 18 can be downloaded here: https://docs.microsoft.com/en-us/sql/ssms/download-sql-server-management-studio-ssms?view=sql-server-ver15

TDS – Tabular Data Stream protocol

The Tabular Data Stream protocol is a protocol on the application layer (layer 7) of the ISO/OSI reference model. It facilitates interaction with a Microsoft SQL server, including authentication and encryption of communication. After successfully logging in to the SQL server, SQL instructions can be exchanged with the server's databases using this protocol. Data are transported over TCP/IP.

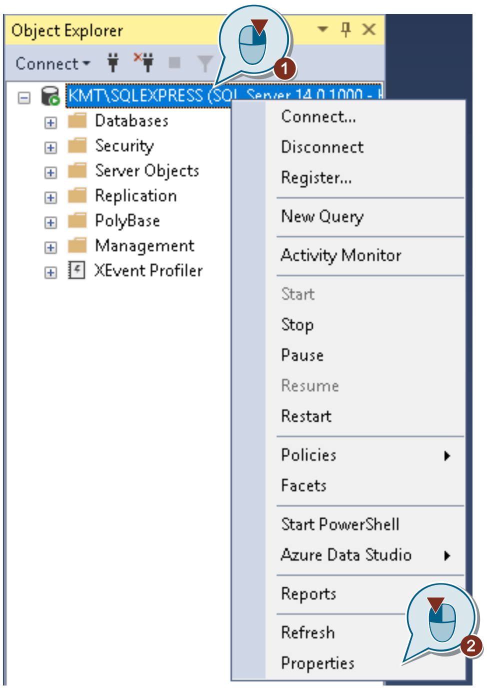
The Tabular Data Stream is described extensively in the Microsoft Technical Documentation: https://docs.microsoft.com/en-us/openspecs/windows_protocols/ms-tds/b46a581a-39de-4745-b076-ec4dbb7d13ec

3.2 Settings in Microsoft SQL Server Express

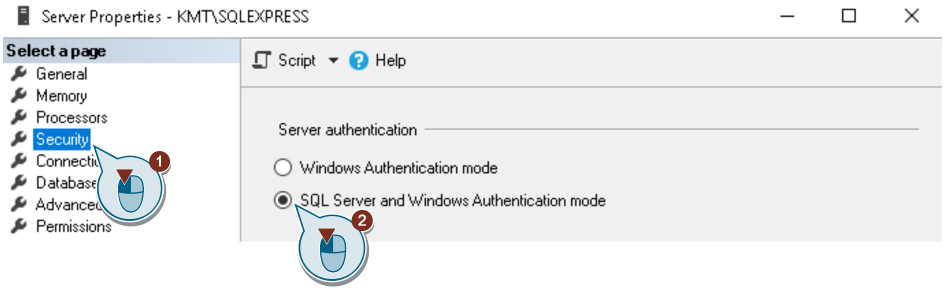
Logging in to the SQL server

The login to a Microsoft SQL Server Express database via the SQL server authentication mode must be authorized in the database settings. This is required in order to log in to the database with username and password via the TDS protocol. The necessary steps for this are described below.

1. Start the Microsoft SQL Server Management Studio.
2. Open the properties of the SQLEXPRESS instance.



3. In the security settings, activate the "SQL Server Authentication Mode" and confirm this change with "OK".

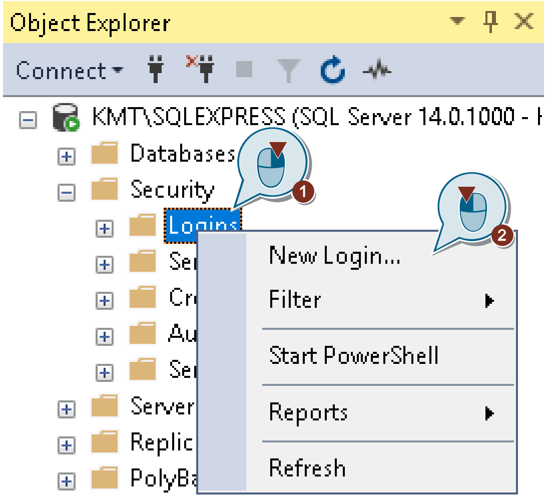


Server authentication

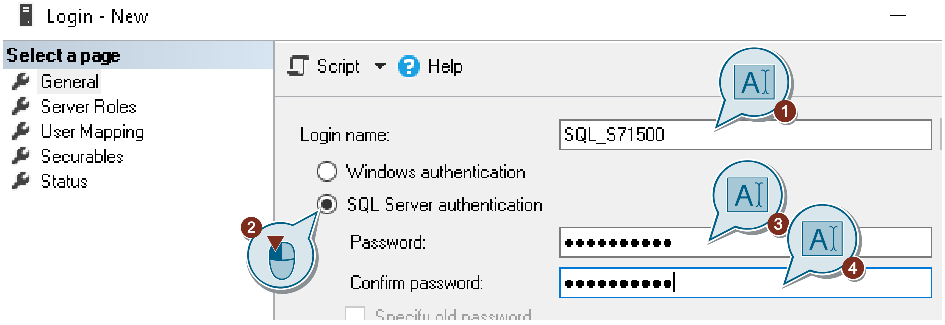
○ Windows Authentication mode

● SQL Server and Windows Authentication mode

4. Create a user account under "Security > Logins > New Login…".



5. Assign a user name and select "SQL Server authentication". Set a password and confirm the entries with "OK".



Login name: SQL_S71500

○ Windows authentication

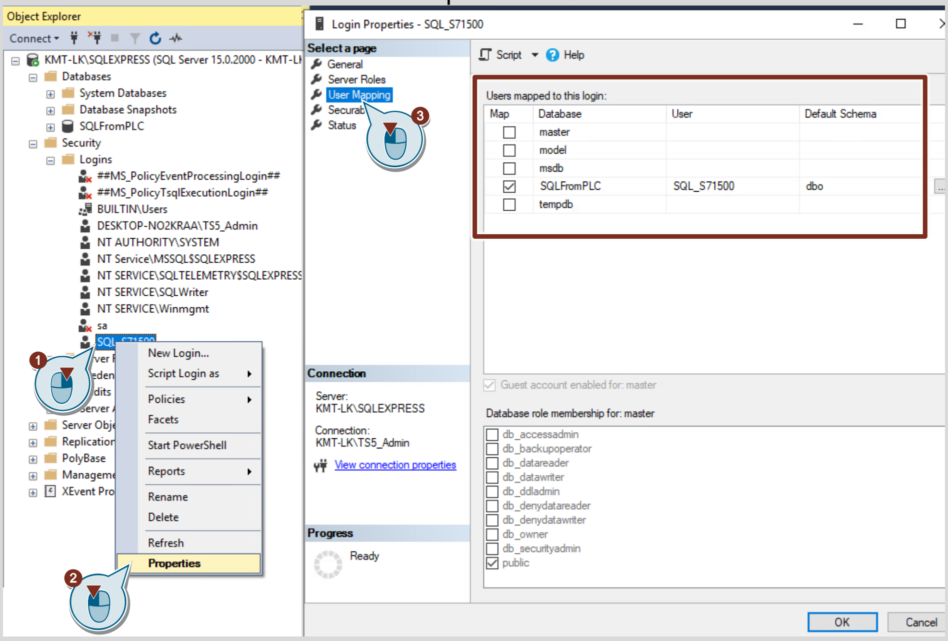
● SQL Server authentication

Password: ●●●●●●●●●●●

Confirm password: ●●●●●●●●●●●

NOTE

Make sure that this user receives the permissions to access this database.



Users mapped to this login:

Map	Database	User	Default Schema
□	master		
□	model		
□	msdb		
☑	SQLFromPLC	SQL_S71500	dbo
□	tempdb		

☑ Guest account enabled for: master

Database role membership for: master

□ db_accessadmin
□ db_backupoperator
□ db_datareader
□ db_datawriter
□ db_ddladmin
□ db_denydatareader
□ db_denydatawriter
□ db_owner
□ db_securityadmin
☑ public

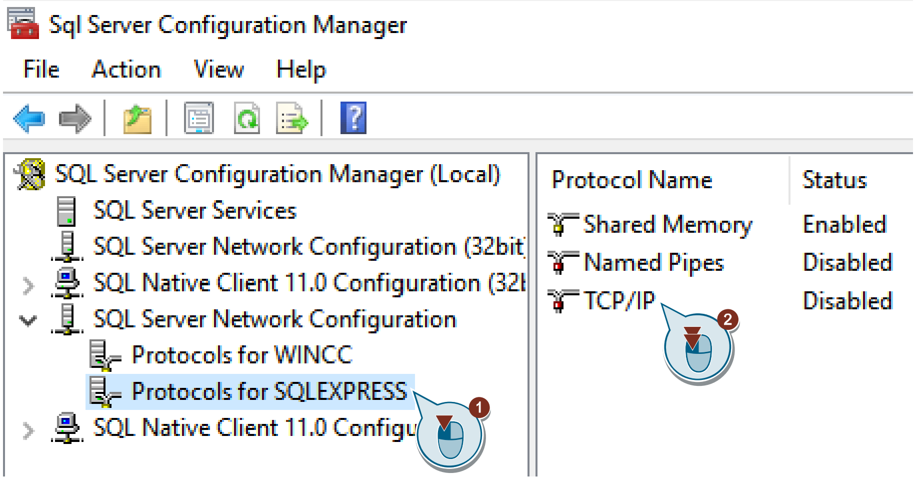
Port authorization in the SQL server

NOTE Port 1433 is the default port for Microsoft SQL server databases.

NOTE With a firewall active on the PC with the Microsoft SQL server database, the TCP port "1433" must be allowed in the firewall for incoming connections.

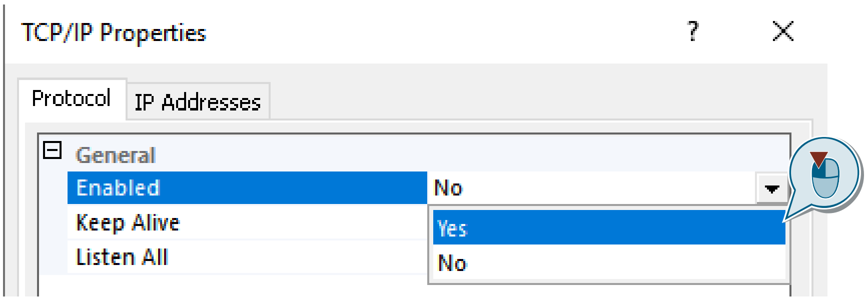
A port authorization must be set up in the SQL server so that the SQL server is reachable on the network. The necessary steps are described below.

1. Start the "Microsoft SQL Server Configuration Manager" and navigate to the protocols for "SQLEXPRESS". Double click to open the TCP/IP settings.


Sql Server Configuration Manager
File
Action
View
Help
SQL Server Configuration Manager (Local)
SQL Server Services
SQL Server Network Configuration (32bit
SQL Native Client 11.0 Configuration (32l
SQL Server Network Configuration
Protocols for WINCC
Protocols for SQLEXPRESS
SQL Native Client 11.0 Configu
1
2

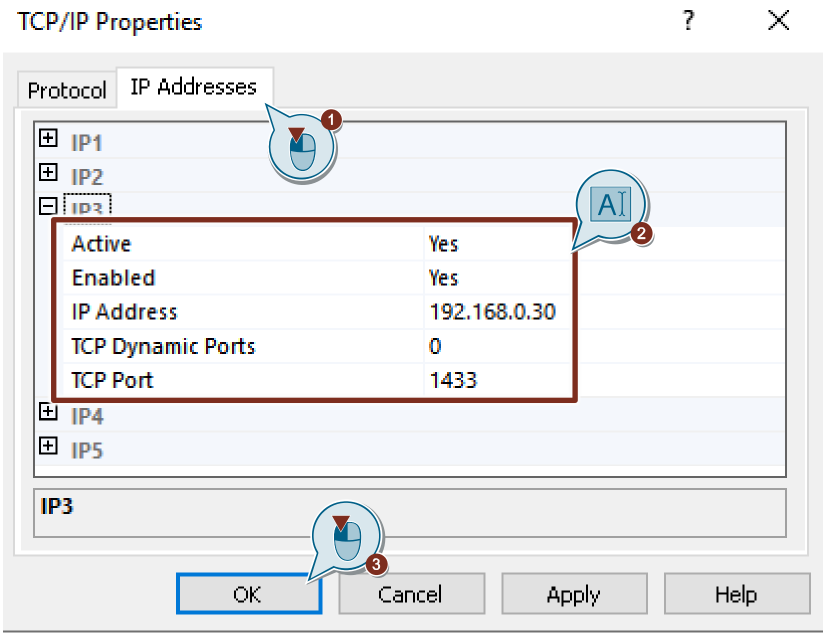
Protocol Name	Status
Shared Memory	Enabled
Named Pipes	Disabled
TCP/IP	Disabled

2. Enable TCP/IP.



Enabled	No
Keep Alive	
Listen All	

3. Under IP3, configure the IP address of the network interface and the port 1433. Enable the interface and confirm the settings with "OK".



Active	Yes
Enabled	Yes
IP Address	192.168.0.30
TCP Dynamic Ports	0
TCP Port	1433

4. Restart the SQL server service for the changes to take effect.

4 Appendix

4.1 Service and support

Industry Online Support

Do you have any questions or need assistance?

Siemens Industry Online Support offers round the clock access to our entire service and support know-how and portfolio.

The Industry Online Support is the central address for information about our products, solutions and services.

Product information, manuals, downloads, FAQs, application examples and videos – all information is accessible with just a few mouse clicks:
support.industry.siemens.com

Technical Support

The Technical Support of Siemens Industry provides you fast and competent support regarding all technical queries with numerous tailor-made offers – ranging from basic support to individual support contracts. Please send queries to Technical Support via Web form:
www.siemens.com/industry/supportrequest

SITRAIN – Digital Industry Academy

We support you with our globally available training courses for industry with practical experience, innovative learning methods and a concept that's tailored to the customer's specific needs.

For more information on our offered trainings and courses, as well as their locations and dates, refer to our web page:
www.siemens.com/sitrain

Service offer

Our range of services includes the following:

- Plant data services
- Spare parts services
- Repair services
- On-site and maintenance services
- Retrofitting and modernization services
- Service programs and contracts

You can find detailed information on our range of services in the service catalog web page:
support.industry.siemens.com/cs/sc

Industry Online Support app

You will receive optimum support wherever you are with the "Siemens Industry Online Support" app. The app is available for iOS and Android:
support.industry.siemens.com/cs/ww/en/sc/2067

4.2 Links and literature

Table 4-1

No.	Subject
\1\	Siemens Industry Online Support https://support.industry.siemens.com
\2\	Link to the article page of the application example https://support.industry.siemens.com/cs/ww/en/view/109779336

4.3 Change documentation

Table 4-2

Version	Date	Change
V1.0	05/2020	First edition
V2.0	11/2020	Extension by the "select" function
V2.1	02/2021	Addition of some notes in the documentation.