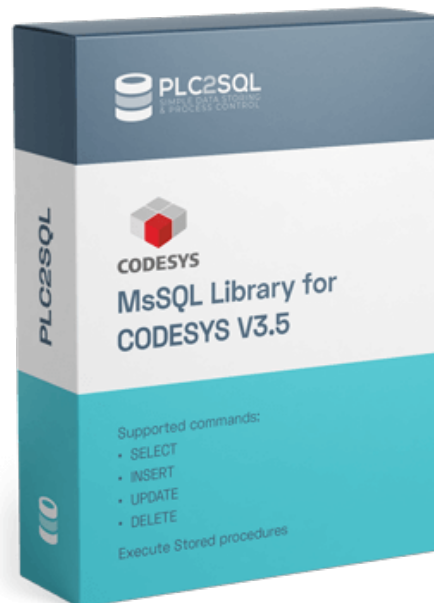




MsSQL Library SL for CODESYS V3.5

USER GUIDE

V1.4.0.3



WWW.PLC2SQL.COM

MAIL: SUPPORT@PLC2SQL.COM

Last revision: January 2020

Copyright © by PLC2SQL 2020
All rights reserved

Contents

1	Product description	2
1.1	Range of function	2
1.2	Supported datatypes	2
1.3	Function blocks	3
1.4	Transition diagram	6
1.5	Functions	7
1.6	Parameters	10
1.7	Errors	11
2	Installation	13
2.1	Installation of Microsoft SQL Server	13
2.1.1	Enable SQL authentication	13
2.1.2	Create database	15
2.1.3	Create new user	15
2.1.4	Create new table	17
2.1.5	Test connection to the database with new user	18
2.1.6	Setup firewall permission for SQL Server	21
2.1.7	SQL Server configuration	24
2.2	Installation package for CODESYS	28
3	Example application	29
3.1	prgCompactExample	29
3.2	prgExample	31
3.3	Example Select query	32
3.4	Example Insert query	33
3.5	Example commands for SQL Server	33
3.5.1	tblTestPLC	33
4	ChangeLog	35

Chapter 1

Product description

This library allows you to connect your CODESYS V3.5 application to Microsoft SQL database. With this library you are able to store and read process data. Read recipes from ERP, save breakdown report and etc. No more OPC servers and other middleware.

Basic requirement for this library is CODESYS V3.5 and Microsoft SQL Server (2005-2014).

Microsoft SQL Server 2014 Express is there: <https://www.microsoft.com/en-us/download/details.aspx?id=42299>

1.1 Range of function

This library allows to the user connect to Microsoft SQL Server from 2005 to 2016. This library use for communication with server TDS protocol.

Supported commands:

- SELECT
- INSERT
- UPDATE
- DELETE
- Execute stored procedures

1.2 Supported datatypes

Other datatypes, which are not in the table below, will cause **xExecuteSQLError** Unsupported datatype.

Datatype	SQL datatype	PLC datatype	ID of datatype
boolean	bit	BOOL	16#32 or 16#68
string	nvarchar(n)	STRING(255)	16#E7 or #A7
int	int	DINT	16#26 or 16#38 or 16#34
bigInt	bigint	DINT	16#7F
float	float	REAL	16#6D
real	real	REAL	16#3E or 16#3B
datetime	datetime	DATETIME(DT)	16#3D or 16#6F

1.3 Function blocks

FUNCTION_BLOCK fbMsSQL

FB for complete connecting and executing commands with Microsoft SQL from 2005 to 2016

Author: krajcart

v:1.3.0.0

InOut:

Scope	Name	Type	Initial	Comment
Input	sHost	STRING		hostname or ip address of mssql server
	uiPort	UINT	1433	port of mssql server
	tTimeout	TIME	TIME#1s0ms	timeout for connection
	sHostName	STRING		hostname for database
	sUserName	STRING		username
	sPassword	STRING		password
	sDatabase	STRING		database name
Output	xConnectSucessfully	BOOL		indication of successful connection to SQL
	xLoginFailed	BOOL		login to SQL failed
	xExecuteSQLError	BOOL		execution of SQL command failed
	xCommandExecutedOK	BOOL		SQL command was executed OK
	wRowDoneCnt	WORD		no. of rows done
	dwError	DWORD	16#400000	error ID
	xDone	BOOL		Job don
	xBusy	BOOL		Job in process
	xError	BOOL		Job has error
	xAborted	BOOL		Job was aborted
Inout	astQuery	<u>astQuery</u>		structure for stQuery
	xConnectSQL	BOOL		trigger for login to SQL
	xDisconnectSQL	BOOL		trigger for logout from SQL
	xExecuteSQL	BOOL		trigger to execute SQL commang executing
	stSqlResponse	<u>stSqlResponse</u>		structure for SQL Response
	sStatus	STRING(255)		response of SQL server
	sErrorMessage	STRING(255)		error message from SQL server

FUNCTION_BLOCK fbMsSQL_compact

FB for complete connecting and executing commands with Microsoft SQL from 2005 to 2016

Author: krajcart

v:1.3.0.0

InOut:

Scope	Name	Type	Initial	Comment
Input	sHost	STRING		hostname or ip address of mssql server
	uiPort	UINT	1433	port of mssql server
	tTimeout	TIME	TIME#1s0ms	timeout for connection
	sHostName	STRING		hostname for database
	sUserName	STRING		username
	sPassword	STRING		password
	sDatabase	STRING		database name
	queryString	STRING(255)		string for SQL query
Output	xConnectSuccessfully	BOOL		indication of successful connection to SQL
	xLoginFailed	BOOL		login to SQL failed
	xExecuteSQLError	BOOL		execution of SQL command failed
	xCommandExecutedOK	BOOL		SQL command was executed OK
	wRowDoneCnt	WORD		no. of rows done
	dwError	DWORD	16#400000	error ID
	xDone	BOOL		Job don
	xBusy	BOOL		Job in process
	xError	BOOL		Job has error
	xAborted	BOOL		Job was aborted
	bFIFOreadAdr	BYTE		FIFO address from which are data stored to Database
	bFIFOwriteAdr	BYTE		FIFO address to which is written actual data
	xFIFOEmpty	BOOL		tag if request in FIFO were processed
Inout	xWriteData	BOOL		trigger to store data to FIFO buffer and send it to SQL
	stSqlResponse	<u>stSqlResponse</u>		structure for SQL Response
	sStatus	STRING(255)		response of SQL server
	sErrorMessage	STRING(255)		error message from SQL server

FUNCTION_BLOCK fbPing

FB for testing ping function to the other side

Author: krajcart

v:1.3.0.0

InOut:

Scope	Name	Type	Comment
Input	sIpAddress	STRING	
Output	xResult	BOOL	
Inout	xTryPing	BOOL	var for try ping to host

FUNCTION_BLOCK fbFIFOQuery

FB for storing queries into FIFO array

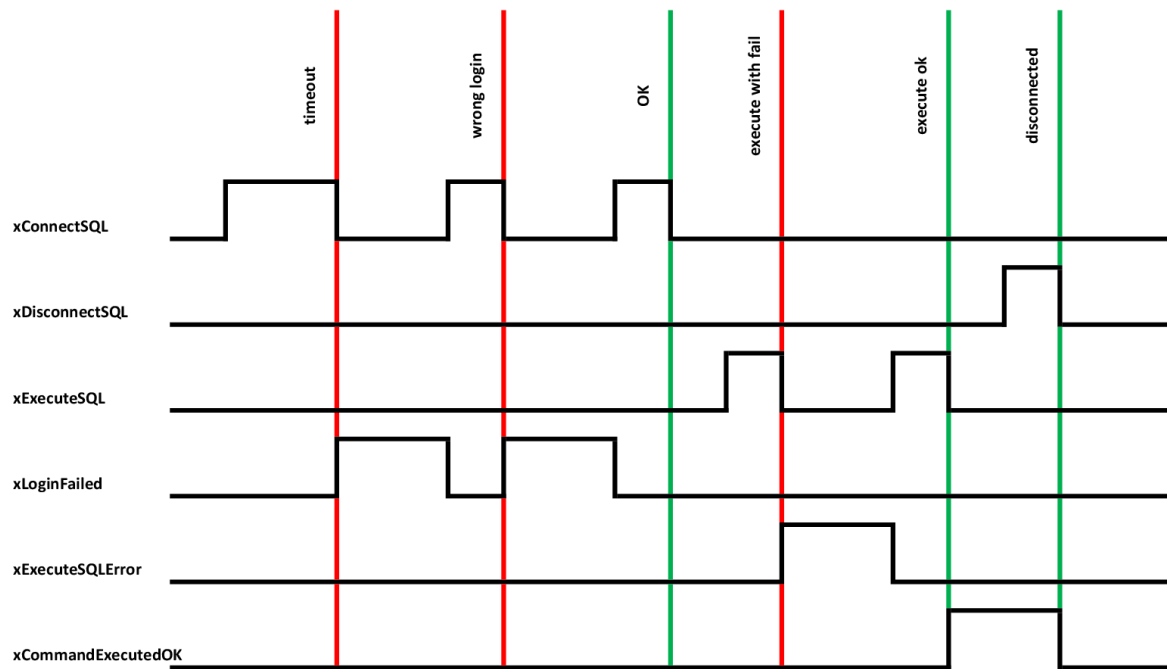
Author: krajcart

v:1.3.0.0

InOut:

Scope	Name	Type	Comment
Input	sInput	STRING(255)	input data
Output	sOutput	STRING(255)	output data
	xFIFOEmpty	BOOL	
Inout	xRead	BOOL	command to read
	xWrite	BOOL	command to write
Output	bFIFOreadAdr	BYTE	FIFO address from which are data stored to Database
	bFIFOwriteAdr	BYTE	FIFO address to which is written actual data

1.4 Transition diagram



1.5 Functions

FUNCTION **fcGetBool** : *eConvertErrors*

FC for converting raw SQL data to datatype **BOOL**

allowed data types: 16#32 or 16#68

Author: krajcart

v:1.3.0.0

InOut:

Scope	Name	Type	Comment
Return	fcGetBool	<u><i>eConvertErrors</i></u>	
Input	iCol	INT	number of column of converted value
	iRow	INT	number of row of converted value
Inout	xValue	BOOL	return boolean value
	stSqlResponse	<u><i>stSqlResponse</i></u>	structure with response data

FUNCTION **fcGetDINT** : *eConvertErrors*

FC for converting raw SQL data to datatype **DINT**

allowed data types: 16#26 or 16#38 or 16#7F(bigint)

Author: krajcart

v:1.3.0.0

InOut:

Scope	Name	Type	Comment
Return	fcGetDINT	<u><i>eConvertErrors</i></u>	
Input	iCol	INT	number of column of converted value
	iRow	INT	number of row of converted value
Inout	diValue	DINT	return doubleinteger value
	stSqlResponse	<u><i>stSqlResponse</i></u>	structure with response data

FUNCTION **fcGetREAL** : eConvertErrors

FC for converting raw SQL data to datatype **REAL**

allowed data types: 16#6D or 16#3E or 16#3B

Author: krajcart

v:1.3.0.0

InOut:

Scope	Name	Type	Comment
Return	fcGetREAL	<u>eConvertErrors</u>	
Input	iCol	INT	number of collumn of converted value
	iRow	INT	number of row of converted value
Inout	rValue	REAL	return real value
	stSqlResponse	<u>stSqlResponse</u>	structure with response data

FUNCTION **fcGetSTRING** : eConvertErrors

FC for converting raw SQL data to datatype **STRING**

allowed data types: 16#E7

Author: krajcart

v:1.3.0.0

InOut:

Scope	Name	Type	Comment
Return	fcGetSTRING	<u>eConvertErrors</u>	
Input	iCol	INT	number of collumn of converted value
	iRow	INT	number of row of converted value
Inout	sValue	STRING(255)	return string value
	stSqlResponse	<u>stSqlResponse</u>	structure with response data
	uiLen	UINT	length of string

FUNCTION **fcGetDATETIME** : eConvertErrors

FC for converting raw SQL data to datatype **DATETIME**

allowed data types: 16#6F or 16#3D

Author: krajcart

v:1.3.0.0

InOut:

Scope	Name	Type	Comment
Return	fcGetDATETIME	<u>eConvertErrors</u>	
Input	iCol	INT	number of column of converted value
	iRow	INT	number of row of converted value
Inout	dtValue	DT	return datetime value
	stSqlResponse	<u>stSqlResponse</u>	structure with response data

FUNCTION **fcGetIpAddr** : STRING

FC for conversion hostname to IP address

Author: krajcart

v:1.3.0.0

InOut:

Scope	Name	Type
Return	fcGetIpAddr	STRING
Input	sHost	STRING

FUNCTION **fcSQLInsertDateTimeFormat** : STRING

FC for **DT** datatype conversion to SQL string datetime format

Author: krajcart

v:1.3.0.0

InOut:

Scope	Name	Type
Return	fcSQLInsertDateTimeFormat	STRING
Input	dtValue	DT

1.6 Parameters

Here is a list of global parameters for setup length of arrays according data output PLC. Changeable by double click on value in Library Manager.

GlobalParameters (PARAMS)

InOut:

Scope	Name	Type	Initial	Comment
Constant	gc_iNoOfColumns	INT	10	number of columns in result
	gc_iNoOfRows	INT	10	number of rows in result
	gc_iLengthOfDataArray	INT	50	length of byte array for result =(length of datatypes in bytes)*gc_iNoOfCollumns
	gc_iLengthOfByteArrays	INT	2000	internal value for bytes arrays
	gc_iLengthOfQuery	INT	5	length of string array for query
	gc_bLengthFIFO	INT	10	length of FIFO array

1.7 Errors

List of errors stored in value **dwError** as output from **fbMsSql** or **fbMsSql_compact**.

Table 1.1: Error table

Code	Error message	Advice
16#400000	No error.	
16#400001	Timeout destination server is unreachable.	Remote server unreachable, try fbPing and firewall.
16#400002	Login failed.	Wrong login or security settings on server side.
16#400003	Command failed.	Executed command failed.
16#400004	SQL error occurred.	Read the description of error.
16#400005	Query has unsupported data type.	Use supported datatype according 1.2 Supported datatypes.
16#400006	Error data row affected.	MS SQL error.
16#400007	Wrong flag on received row affected.	MS SQL error.
16#400008	Wrong row done.	MS SQL error.
16#400009	No command to execute.	Write some SQL command.
16#400010	More columns than are defined in gc.iNoOfColumns.	Increase gc.iNoOfColumns constant in global parameters.
16#400011	More rows than are defined in gc.iNoOfRows.	Increase gc.iNoOfRows constant in global parameters
16#400012	More raw data than are defined in gc.iLengthOfDataArray.	Increase gc.iLengthOfDataArray constant in global parameters.
16#400013	gc.iLengthOfByteArrays is longer than PLC can handle.	Decrease gc.iLengthOfByteArrays.
16#400014	License is not valid! Demo expired.	Buy the license.
16#400015	Network socket error.	Connect again.
16#400016	sUserName is not filled.	Fill sUserName.
16#400017	sHost is not filled.	Fill sHost.
16#400018	sHost wasn't recognised. DNS-request failed.	Change sHost or check DNS settings on your PLC.
16#400019	Receiving invalid or unexpected PACKET from server.	
16#400020	MSSQL-ERROR: Login Ack Token not received.	MS SQL error.
16#400021	MSSQL-ERROR: Receiving unknown TOKEN FROM server.	MS SQL error.
16#400022	MSSQL-ERROR: Could NOT obtain length information of column data.	MS SQL error.
16#400023	Name of column is longer than is defined in c.iMaxIdentifierLength.	MS SQL error.
16#400024	MSSQL Library DO NOT support SQL-Batch.	

Table 1.2: Error table convert functions

eError	Error message	Advice
16#00	No error.	
16#01	iRow is 0.	Set variable iRow bigger than 0.
16#02	iRow is out of range.	Set variable iRow between 1 and stSqlResponse.uiRowCount.
16#03	iColumn is 0.	Set variable iColumn bigger than 0.
16#04	iColumn is out of range.	Set variable iRow between 1 and stSqlResponse.uiColumnCount.
16#05	Requested value is NULL.	Fill this column in database.
16#06	Structure stSqlResponse is empty.	Execute some SQL query.
16#68	Wrong datatype for BOOL to convert.	Allowed is TDS_DATA_BITN or TDS_DATA_BIT1.
16#3D6F	Wrong datatype for DATETIME to convert.	Allowed is TDS_DATA_DATETIME8 or TDS_DATA_DATETIMEN.
16#2638	Wrong datatype for DINT to convert.	Allowed is TDS_DATA_INT4 or TDS_DATA_INT2 or TDS_DATA_INT8 or TDS_DATA_INTN.
16#6D	Wrong datatype for REAL to convert.	Allowed is TDS_DATA_FLOATN or TDS_DATA_FLOAT4 or TDS_DATA_FLOAT8.
16#E7	Wrong datatype for STRING to convert.	Allowed is TDS_DATA_NVARCHAR or TDS_DATA_BIGVARCHAR.

Chapter 2

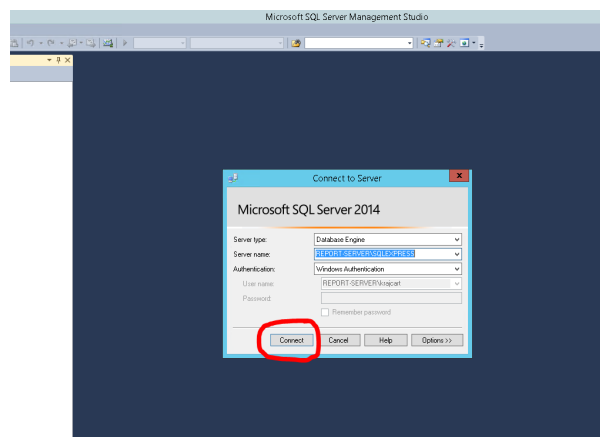
Installation

2.1 Installation of Microsoft SQL Server

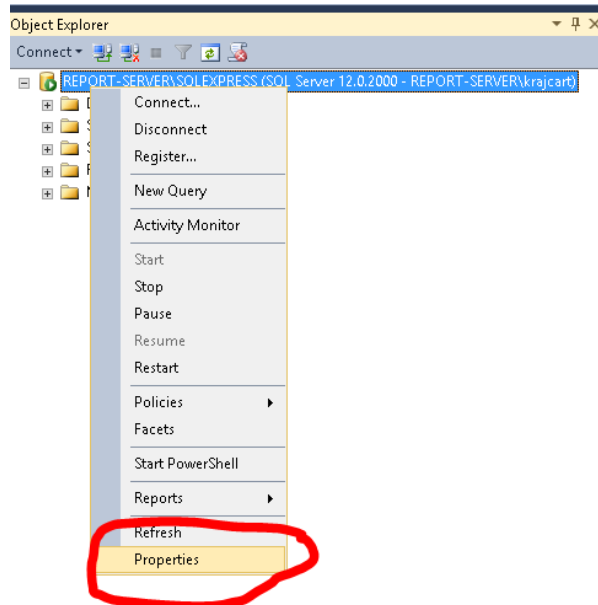
How to install Microsoft SQL Server is shown on this video: <https://www.youtube.com/watch?v=QFyetK805bo>

In subsections below are shown necessary steps which has to be done on SQL Server side to be accessible from PLC.

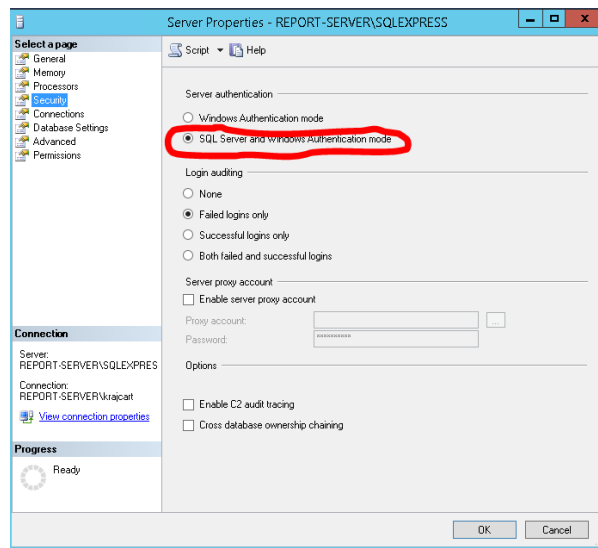
2.1.1 Enable SQL authentication



Connect to your database with Microsoft SQL Management Studio. You will login with **Windows authentication**.

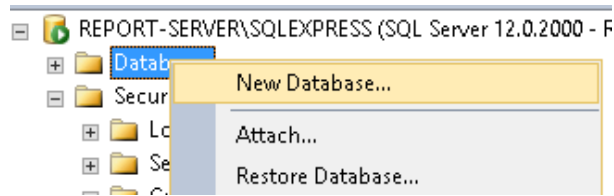


After successful connection to the SQL Server. Right click on the SQL server icon and choose **Properties**.

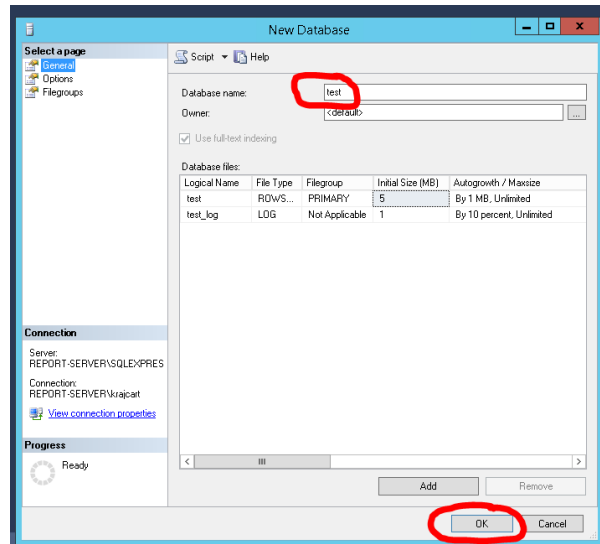


In Server Properties choose page **Security**, and choose **SQL Server and Windows Authentication mode**.

2.1.2 Create database

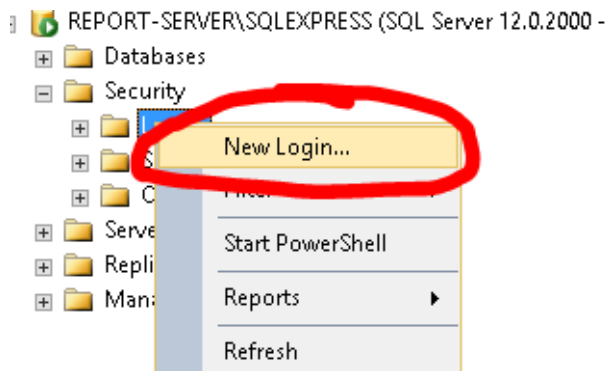


Now we need to create test database. Right click on tab **Database**, a choose **New Database**.

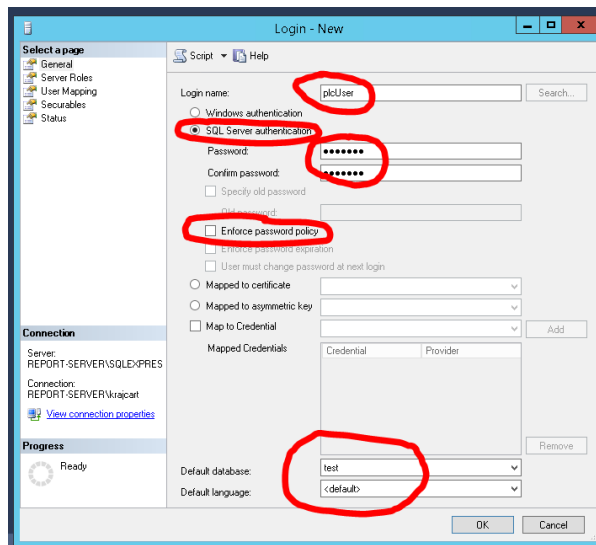


In this window we will set only **Database name**: in our case name will be **test** and press **OK**.

2.1.3 Create new user

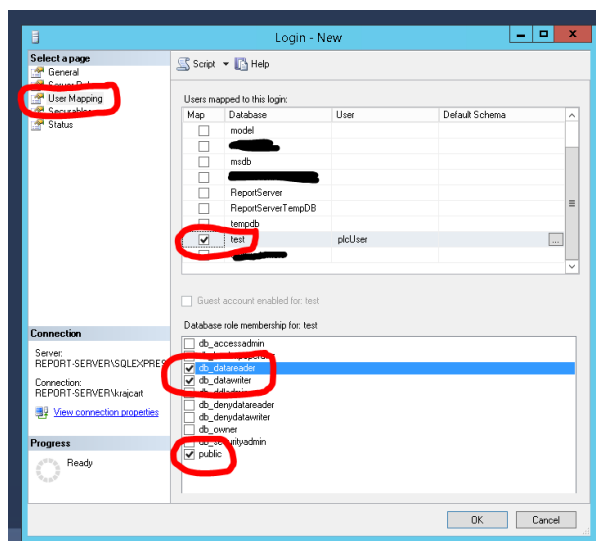


Right click on tab **Security** and click on **New Login**.



In this window we will fill these columns:

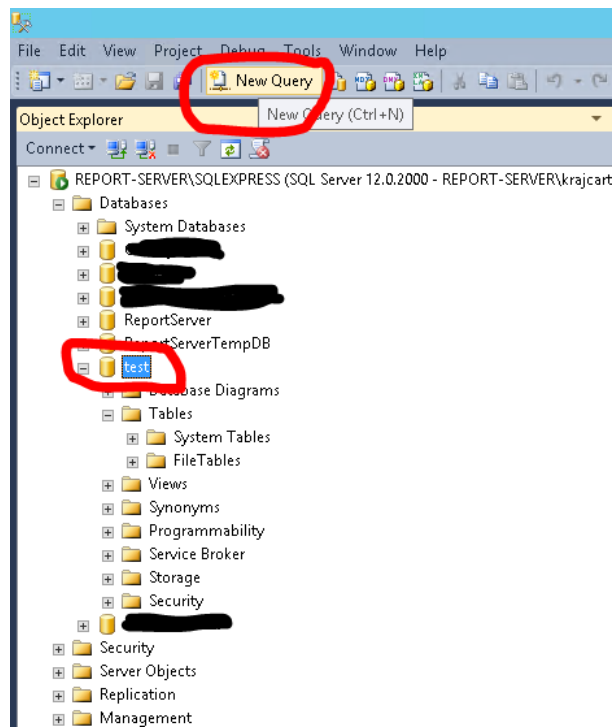
- **Login name** - name of new user
- **SQL Server authentication** - and set password
- uncheck **Enforce password policy**
- **Default database** - our case db test



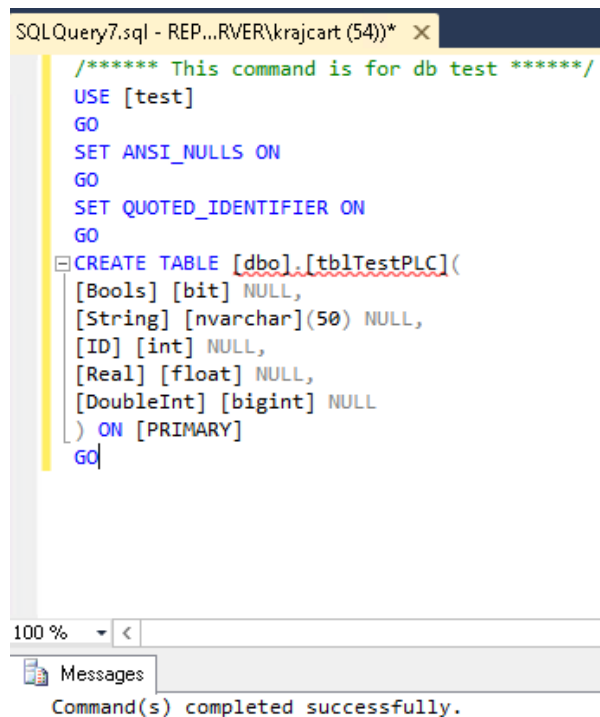
Another tab is **User Mapping**. In this tab we will set permissions for this user. We will map our database **test**. And for this database we will set these permissions:

- **db_datareader** - allows to read data from tables
- **db_datawriter** - allows to write data into tables
- **public** - this one is set by default

2.1.4 Create new table



Left click on our database **test** in tab Databases and then choose **New Query**.

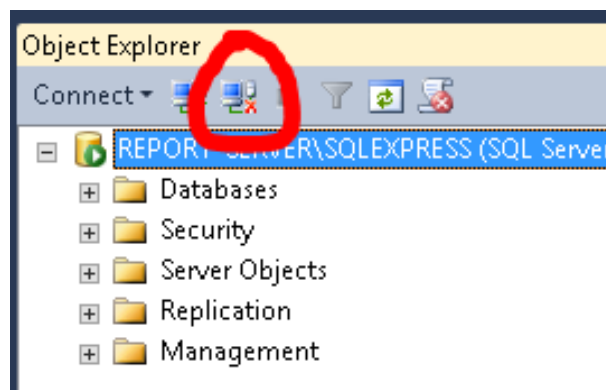


Into the new window we will insert code from figure: 2.1. And after **press F5** to execute command.

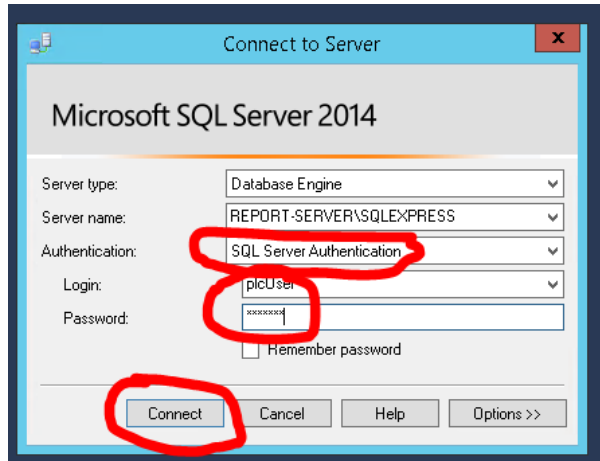
```
/****** This command is for db test *****/  
  
USE [test]  
GO  
  
SET ANSI_NULLS ON  
GO  
  
SET QUOTED_IDENTIFIER ON  
GO  
  
CREATE TABLE [dbo].[tblTestPLC](  
  [Bools] [bit] NULL,  
  [String] [nvarchar](50) NULL,  
  [ID] [int] NULL,  
  [Real] [float] NULL,  
  [DoubleInt] [bigint] NULL  
  
  ) ON [PRIMARY]  
  
GO
```

Figure 2.1: Create tblTestPLC

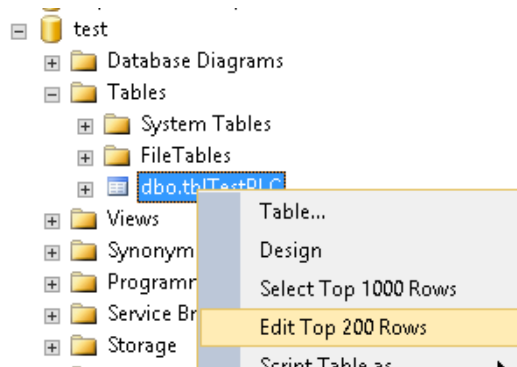
2.1.5 Test connection to the database with new user



First thing is to disconnect from the database.



In the login screen we will change **Windows authentication** to **SQL Server Authentication**, and fill **Login** and **Password** of our new user.



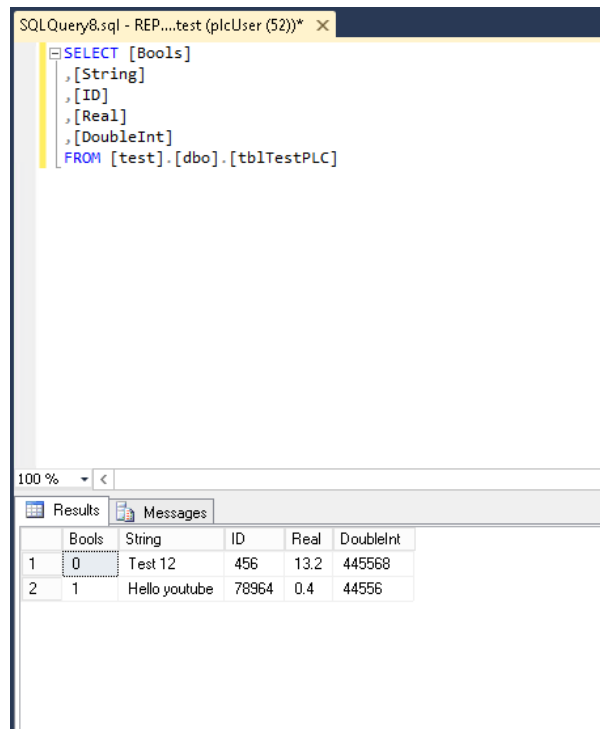
Next step is to try if permission which we set earlier to our user. So in tab Databases -> Test -> Tables -> tblTestPLC right click and choose **Edit Top 200 Rows**. And this command will open window where we can modify data in our table.

Bools	String	ID	Real	DoubleInt
False	Test 12	456	13.2	445568
True	Hello youtube	78964	0.4	44556
NULL	NULL	NULL	NULL	NULL

In table 2.1 are shown sample data which you can put into this table.

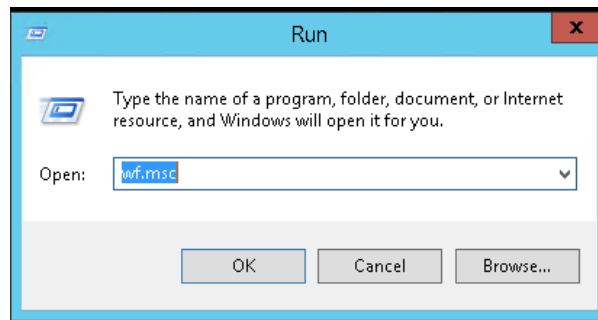
Table 2.1: Example of values, which you can add to your tblTestPLC table

[rows]	Bools	String	ID	Real	DoubleInt
1	0	Test 12	456	13.2	445568
2	1	Hello youtube	78964	0.4	44556

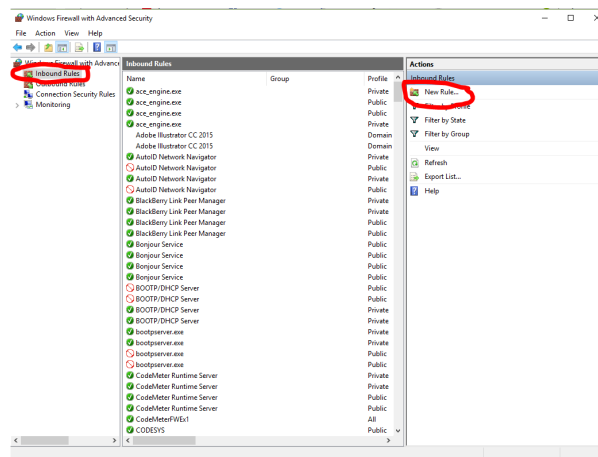


Last step is to try which data are in the table. Example of Select command is here: 2.1.

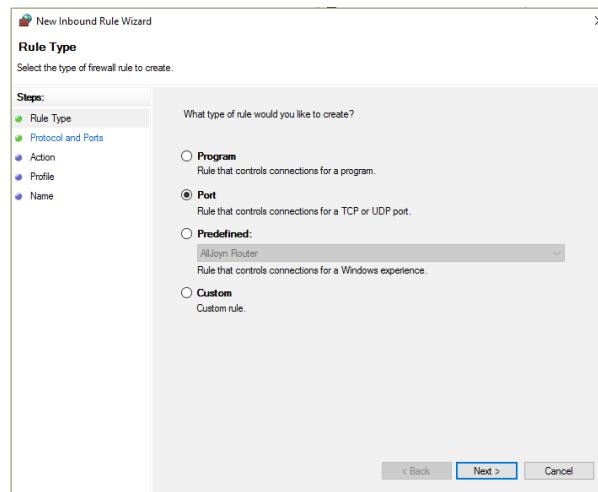
2.1.6 Setup firewall permission for SQL Server



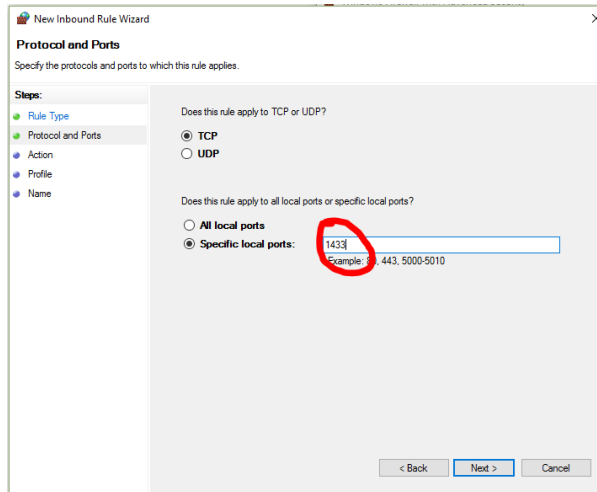
Open **Windows Firewall with Advanced Security**. Open **Run** and insert command **wf.msc**



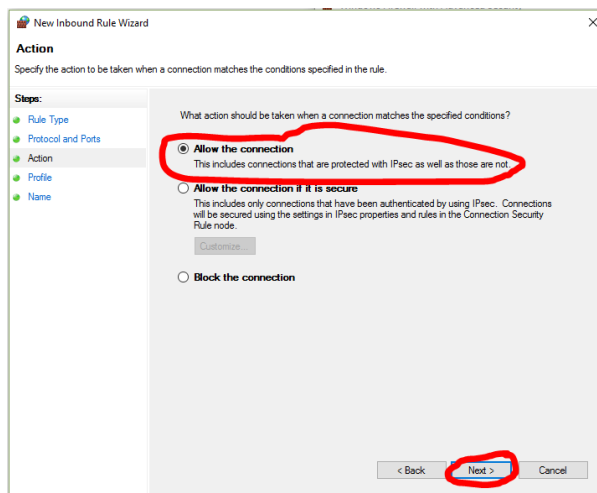
On left side choose **Inbound Rules** by left click. This will open list with all Inbound rules set for your PC. On right side choose **New Rule**.



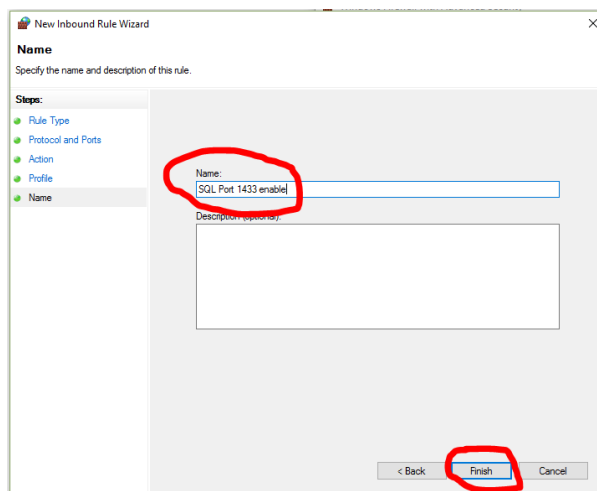
This wizard will help us to set correct firewall rule for our PC. First we have to choose **Port** and press **Next**.



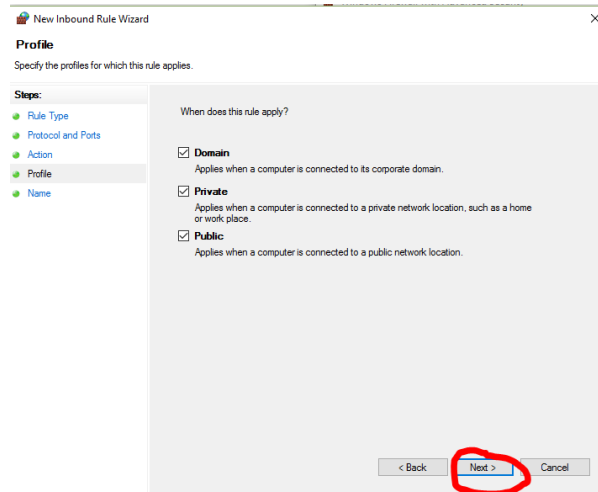
Here we will specify on which port is our SQL server running, default port is **1433**. It's adjustable in settings of SQL Server.



Just choose **Allow the connection** and press **Next**.



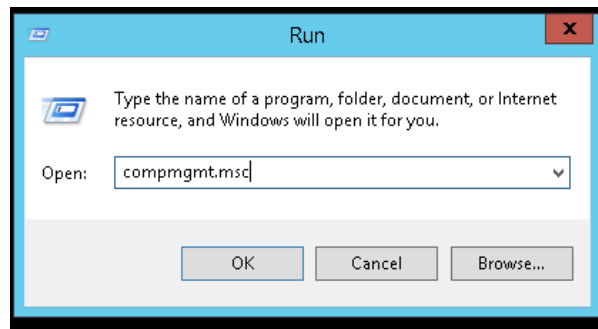
Insert name of the firewall rule and press **Next**.



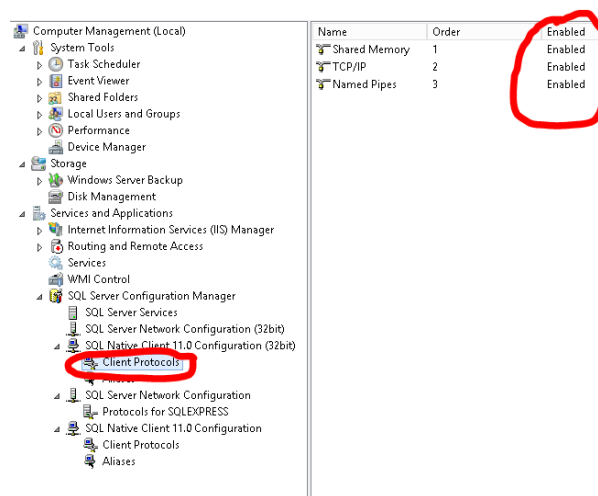
In this last windows we will choose for which types of zones will be rule applied.

Note: this is just basic example for enable firewall!!

2.1.7 SQL Server configuration

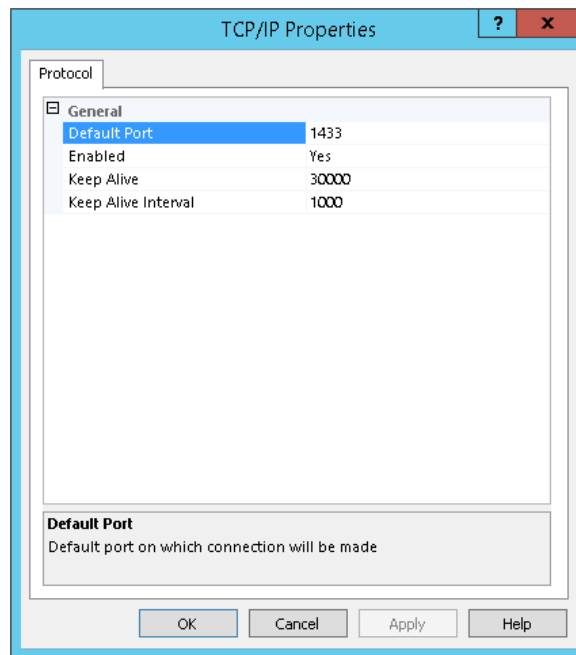


First thing is to open **Computer management**. Easiest way how to do that is to open **Run**: insert command **compmgmt.msc**



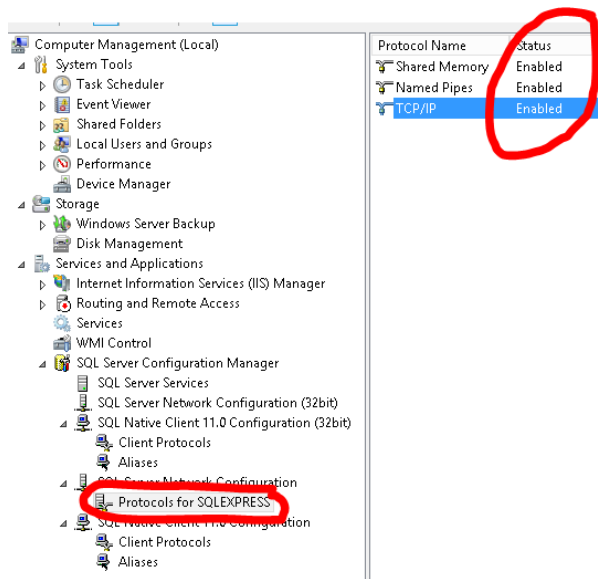
On left side choose Services and Applications -> SQL Server Configuration Manager -> SQL Native Client 11.0 Configuration -> Client Protocols. All 3 protocols has to be **Enabled**.

Note: SQL Native Client 11.0 Configuration naming is different in every version of SQL Server and also version(32 or 64 bits)!!

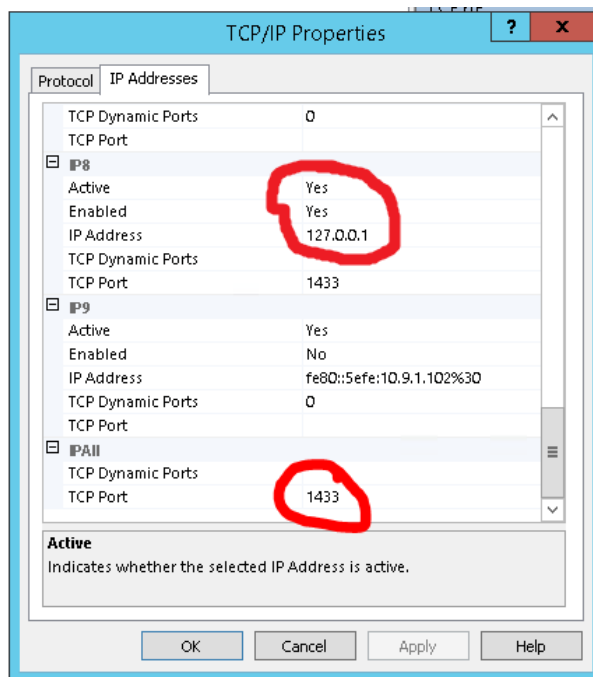


By right click on **TCP/IP** you can edit **Properties**:

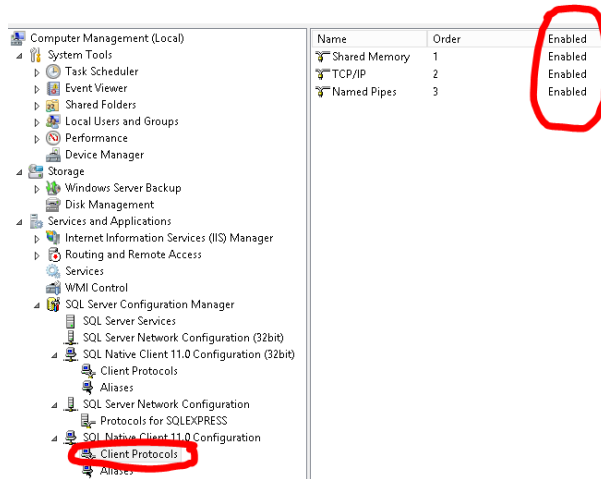
- **Default Port** - port on which will be your SQL server reachable across network
- **Enabled** - Enabling of TCP/IP protocol



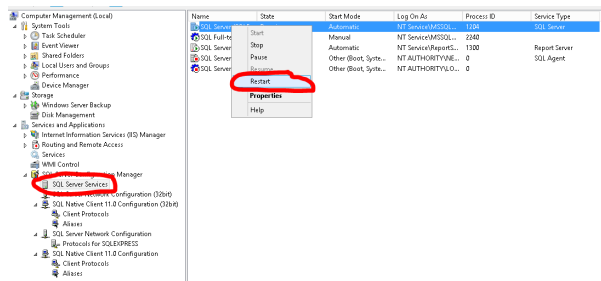
Another settings will be made in SQL Server Network Configuration. Click on **Protocols for SQL-EXPRESS**. Also all 3 protocols has to be **Enabled**. We need to check if port is set correctly right click on **TCP/IP** and choose tab **IP Addresses**.



Here is set the default port for **SQL Server** and also all **IP addresses** for connection to the SQL Server. And click yes to parameter **Enabled** on each IP address which you want to have active for SQL. We activated 127.0.0.1(localhost). So for your ip 192.168.1.1 you have to scroll up and find the right interface.



In case you have 64 bit SQL Server you have to also Enable Protocols in 64 bits client.



All these changes requires restart of SQL Server.

2.2 Installation package for CODESYS

Download MsSQL library package from Codesys e-shop. Install it into your CODESYS development.

This library package contains:

- Example CODESYS Control for Raspberry Pi
- Example CODESYS Control for PFC200
- Example CODESYS Control WIN V3

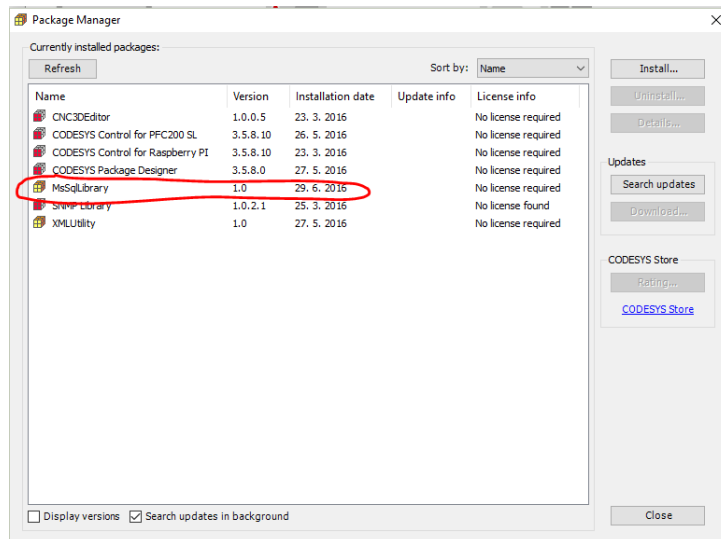


Figure 2.2: Package manager CODESYS V3.5

Chapter 3

Example application

Example project have two subprograms: **prgCompactExample** and **prgExample**.

3.1 prgCompactExample

In this program is used function block **fbMsSQL_compact**. To make this function block work you have to fill:

- **sHost** fill IP address or Hostname of MsSQL server
- **uiPort** set TCP port on which MsSQL server listen (eg. 1433)
- **tTimeout** timeout for connection
- **sUserName** username for login to MsSQL server
- **sPassword** password for login to MsSQL server
- **sDatabase** name of target database on MsSQL server
- **sActivationNumber** activation number for full functionality(if it's empty it runs 2 hours after start)
- **queryString** fill this string with SQL query (eg. SELECT * FROM tblData)

To execute **queryString** on MsSQL server put **xWriteData** to **TRUE**. Function block will connect to MsSQL server if it's reachable and execute **queryString** and if there is no more request, will disconnect from server. This function block has **FIFO buffer** for queries so it can handle string queries from 0 to **gc_bLengthFIFO**.

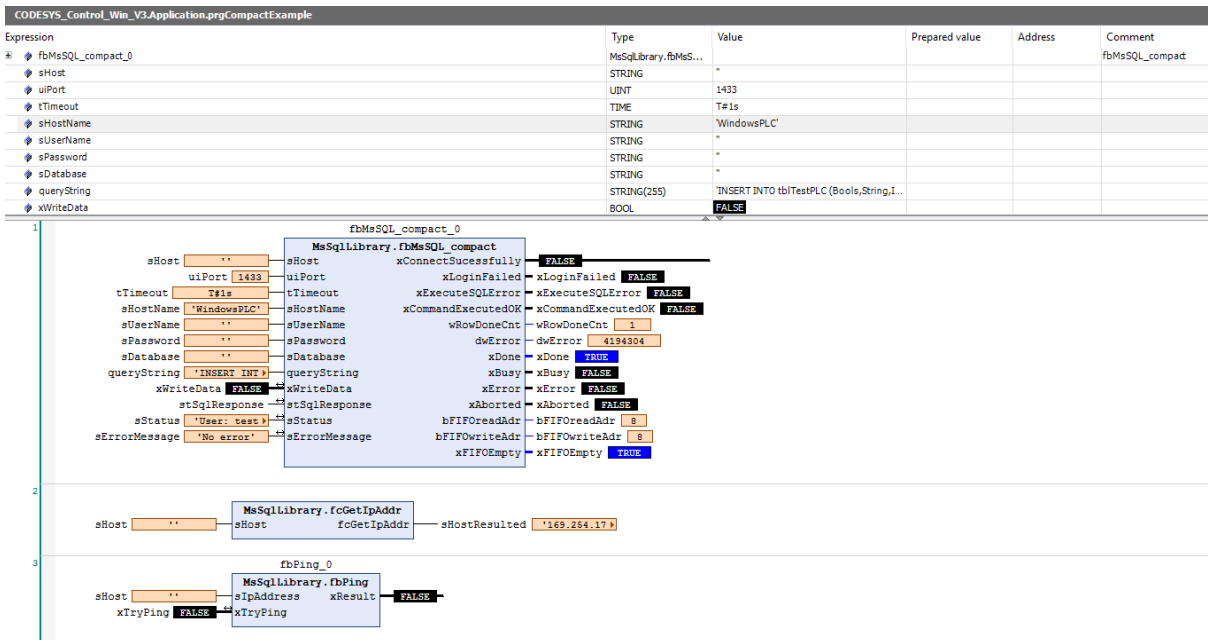


Figure 3.1: prgCompactExample

MsSQL Library SL Compact example for Windows V3

MsSQL Library SL V1.2.0.0 for CODESYS V3.5

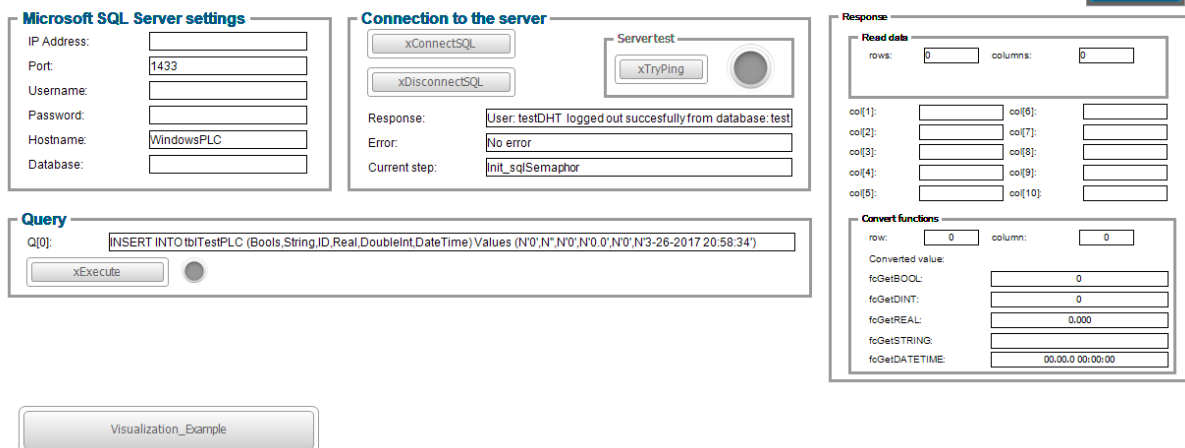


Figure 3.2: visuCompactExample

3.2 prgExample

In this program is used function block **fbMsSQL**. To make this function block work you have to fill:

- **sHost** fill IP address or Hostname of MsSQL server
- **uiPort** set TCP port on which MsSQL server listen (eg. 1433)
- **tTimeout** timeout for connection
- **sUserName** username for login to MsSQL server
- **sPassword** password for login to MsSQL server
- **sDatabase** name of target database on MsSQL server
- **sActivationNumber** activation number for full functionality(if it's empty it runs 2 hours after start)
- **astQuery** is array for strings with SQL queries (eg. SELECT * FROM tblData)

To execute **query** in **astQuery** on MsSQL server log to MsSQL Server by setting **xConnect** to **TRUE**. When is PLC connected to MsSQL execute SQL query by **xExecuteSQL** to **TRUE**. Disconnecting from database by **xDisconnectSQL** to **TRUE**.

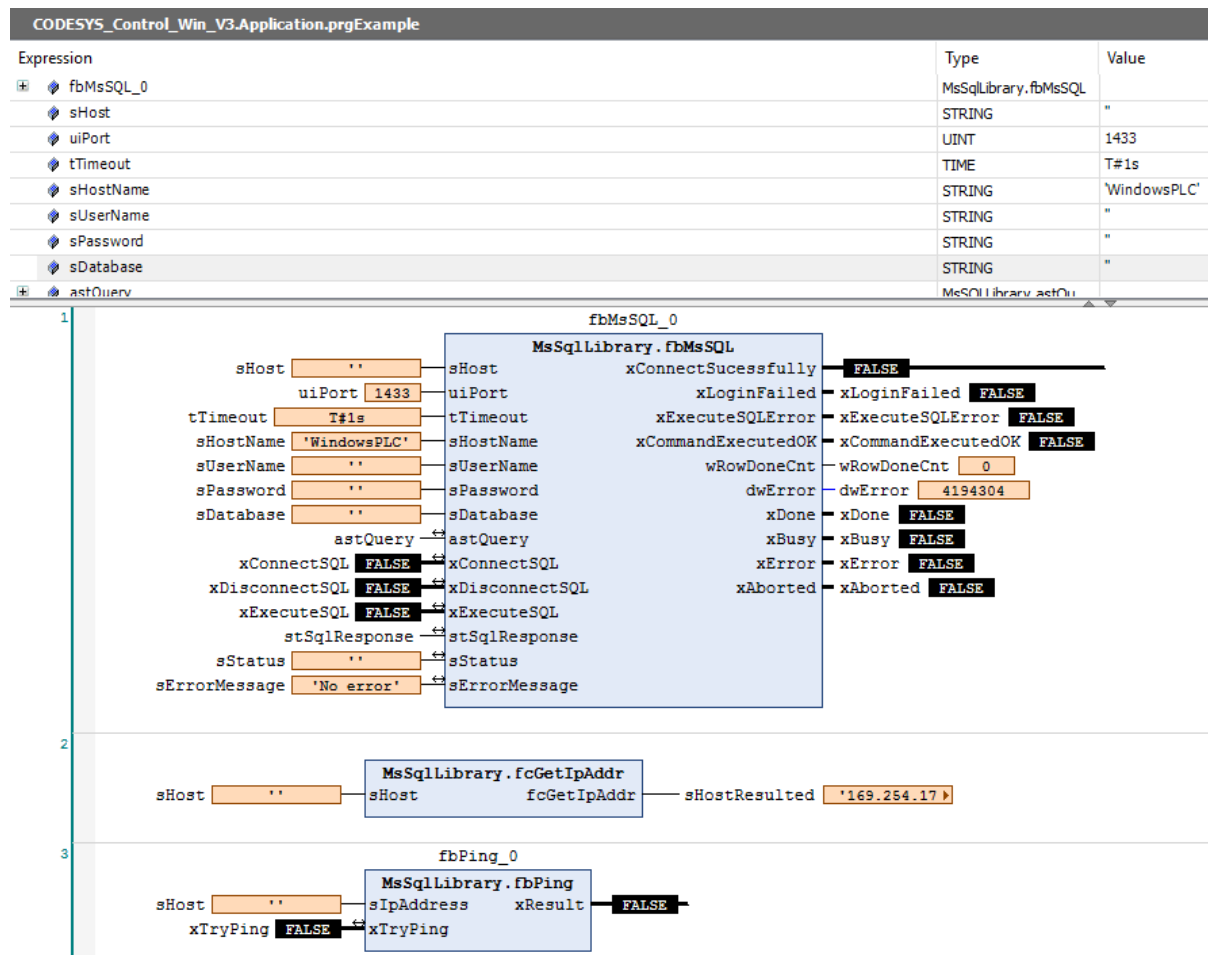


Figure 3.3: prgExample

MsSQL Library SL example for Raspberry PI

MsSQL Library SL V1.2.0.0 for CODESYS V3.5

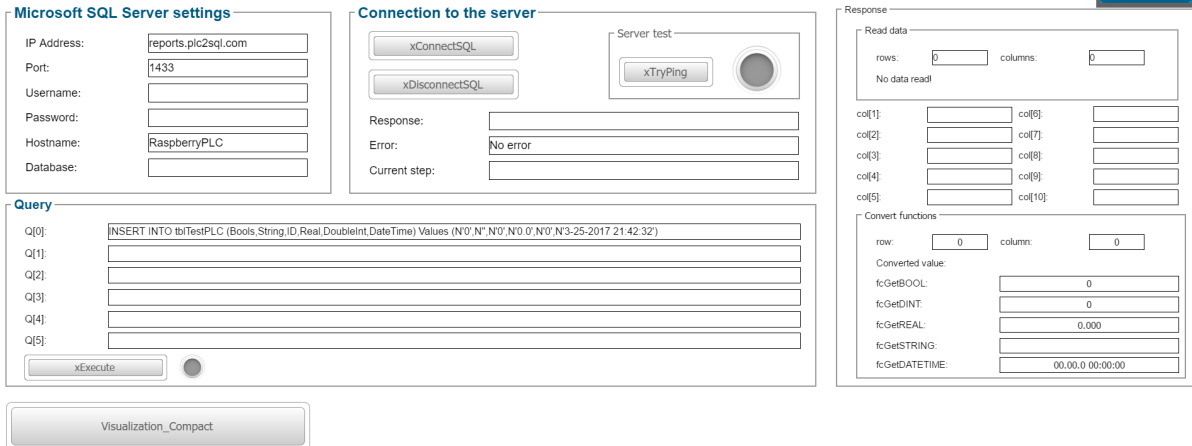


Figure 3.4: visuExample

3.3 Example Select query

`fcSelectQuery` in the picture 3.5 is example of `fc` which creates `SELECT` query string for MsSQL Server. This example returns **TOP 5** records from table `tblTestPLC1` WHERE is **column ID** equal variable `ID_TO_SELECT`.

```
FUNCTION fcSelectQuery : STRING(255)
VAR_INPUT
    ID_TO_SELECT : DINT;
END_VAR
VAR
    sSelect : STRING(255);
END_VAR
```

```
sSelect:='SELECT TOP (5) Ident,Bools,String,ID,Real,DoubleInt,DateTime FROM tblTestPLC1 WHERE ID = N$';
sSelect:=CONCAT(STR1:=sSelect, STR2:=DINT_TO_STRING(ID_TO_SELECT));
sSelect:=CONCAT(STR1:=sSelect, STR2='$');
fcSelectQuery:=sSelect;
```

Figure 3.5: fcSelectQuery

3.4 Example Insert query

fcInsertQuery in the picture 3.6 shows how to create insert query string for MsSQL server.

```

FUNCTION fcInsertQuery : STRING(255)
VAR_INPUT
    TestBool:BOOL;
    TestString:STRING;
    TestInt:INT;
    TestReal:REAL;
    TestDint:DINT;
    TestDateTime:DT;
END_VAR
VAR
    sQuery:STRING(255);
sQuery:='INSERT INTO tblTestPLC1 (Bools,String,ID,Real,DoubleInt,DateTime) Values (N$';
sQuery:=CONCAT (STR1:=sQuery, STR2:=INT_TO_STRING(BOOL_TO_INT(TestBool)));
sQuery:=CONCAT (STR1:=sQuery, STR2:='$',N$');
sQuery:=CONCAT (STR1:=sQuery, STR2:=TestString);
sQuery:=CONCAT (STR1:=sQuery, STR2:='$',N$');
sQuery:=CONCAT (STR1:=sQuery, STR2:=INT_TO_STRING(TestInt));
sQuery:=CONCAT (STR1:=sQuery, STR2:='$',N$');
sQuery:=CONCAT (STR1:=sQuery, STR2:=REAL_TO_STRING(TestReal));
sQuery:=CONCAT (STR1:=sQuery, STR2:='$',N$');
sQuery:=CONCAT (STR1:=sQuery, STR2:=DINT_TO_STRING(TestDint));
sQuery:=CONCAT (STR1:=sQuery, STR2:='$',');
sQuery:=CONCAT (STR1:=sQuery, STR2:=MsSqlLibraryTurck.fcSQLInsertDateTimeFormat (dtValue:=TestDateTime));
sQuery:=CONCAT (STR1:=sQuery, STR2:='')');
fcInsertQuery:=sQuery;

```

Figure 3.6: fcInsertQuery

3.5 Example commands for SQL Server

3.5.1 tblTestPLC

In this table we have 6 collumns:

- **Bools** [bit]
- **String** [nvarchar(50)]
- **ID** [int]
- **Real** [float]
- **DoubleInt** [bigint]
- **DateTime** [datetime]

Table 3.1: Example of values, which you can add to your tblTestPLC table

[rows]	Bools	String	ID	Real	DoubleInt	DateTime
1	0	Test 12	456	13.2	445568	2014-08-05 00:00:00.000
2	1	Hello youtube	78964	0.4	44556	2012-03-04 00:00:00.000

```
SELECT [Bools]
      ,[String]
      ,[ID]
      ,[Real]
      ,[DoubleInt]
      ,[DateTime]
FROM [test].[dbo].[tblTestPLC]
```

Figure 3.7: SELECT from tblTestPLC

Chapter 4

ChangeLog

V 1.0.0.0

- Released

V 1.1.0.0

- **BUG** - Changed declaration of variables **sStatus**, **sErrorMessage**, **astQuery.sQuery** from **STRING** to **STRING(255)**.
- **BUG** - Overwrite variable **uiPort** in case of longer **sErrorMessage** than 80 chars.

V 1.1.0.1

- **BUG** - Parsing raw data fails in case of only one non null int. value
- **UPDATE** - Return datatype of function **fcGetSTRING** is now **STRING(255)**.
- **UPDATE** - Supported sql datatype of function **fcGetDINT** is now **bigint**.
- **FEATURE** - Function for getting data-type datetime, new function **fcGetDATETIME**.
- **BUG** - If **xExecuteSQLError** was in true state also **xCommandExecutedOk** was in true state. Now only one of these two states can be true.

V 1.1.0.2

- **FEATURE** - New global parameter **gc_iNoOfRows** for defining how many rows is needed for result.
- **FEATURE** - New output value **dwError**. Which show actual error according table in **1.7 Errors**.
- **BUG** - Overflow error in case when result of query is longer than space in **stSqlResult**. Function block send error in these cases.
- **BUG** - fbMySQL is unable to connect to server in case that byte array with length **gc_iLengthOfdataArray** for reading data is longer than PLC can handle. Error 16#400013

V 1.1.0.3

- **UPDATE** - All converting functions have improved process for retrieving variables from raw data.

V 1.1.0.4

- **BUG** - Data type `real(16#3E)` wasn't correctly converted to `REAL`.
- **BUG** - Name of column had ghost symbols.
- **UPDATE** - Function `fcGetREAL` now support 4byte `REAL(16#3B)`.
- **UPDATE** - Error message `sErrorMessage`, now in case of error "Query has unsupported data type." shows which data-type is wrong.

V 1.1.0.5

- **BUG** - Product code on side of CODESYS Store was changed.
- **BUG** - Library misinterpreted `DONEINPROC` token in case of negative integer in result set.
- **UPDATE** - Library performance improved.
- **UPDATE** - Error message `sErrorMessage`, now in case of error "Network socket error."
- **UPDATE** - `fbMsSQL` has now IEC behavioural statuses(`xDone`, `xBusy`, `xError`, `xAbort`).

V 1.2.0.0

- **FEATURE** - New function `fcGetIpAddress` to resolve `ipAddress` from DNS name.
- **UPDATE** - `sIpAddress` is no more input. Instead of it there is `sHost` where is possible to insert `ipAddress` or `hostname`.
- **FEATURE** - Library has now two main function blocks `fbMsSQL_compact` which has integrated FIFO buffer and StateMachine automat for storing data into SQL database easy on one trigger. And `fbMsSQL` for advanced users.
- **FEATURE** - New fb for handling FIFO `fbFIFOQuery`.
- **FEATURE** - New function `fcSQLInsertDateTimeFormat` for creating `DATETIME` string for MsSQL server.
- **UPDATE** - 3 new errors to check if `sUserName`, `sHost` was filled and info if DNS-request failed.

V 1.2.0.1

- **BUG** - Product code checked wrongly.

V 1.2.0.2

- **BUG** - Product code on side of CODESYS Store was wrongly assigned.

V 1.2.0.3

- **UPDATE** - Improvement of license check.

V 1.3.0.0

- **BUG** - Library now works with all types of processors.

V 1.3.0.1

- **UPDATE** - Function fcGetString now support VARCHAR(16#A7) datatype.

V 1.3.0.2

- **BUG** - Reading errors from Server.

V 1.3.0.3

- **BUG** - Reading datatypes with nullable parameter.

V 1.3.0.4

- **UPDATE** - All necessary libraries are now set to newest version due to failure with Codesys V3.5 SP14 with x64 target.
- **UPDATE** - Library now requests only necessary core libraries.
- **UPDATE** - Visualization was deleted from origin library and moved to example project.

V 1.4.0.0

- **UPDATE** - Refactoring of whole library, delete unused variables and improve performance.

V 1.4.0.1

- **BUG** - Library stucked during connection. Wrong pointer to send data and short timeout for connection.

V 1.4.0.2

- **BUG** - Library wrongly interpreted SQL error message longer than 255 bytes and causes crash of Codesys RTE. Also when there is SQL Execute error this bit is set to TRUE.

V 1.4.0.3

- **BUG** - Error Message longer than 80 chars was out of memory due using variable STRING instead of STRING(255) caused crash of Codesys Runtime. Now it's fixed.
- **UPDATE** - Library has now implemented skipping tokens TABNAME and COLINFO. This feature is necessary for some advanced stored procedures.