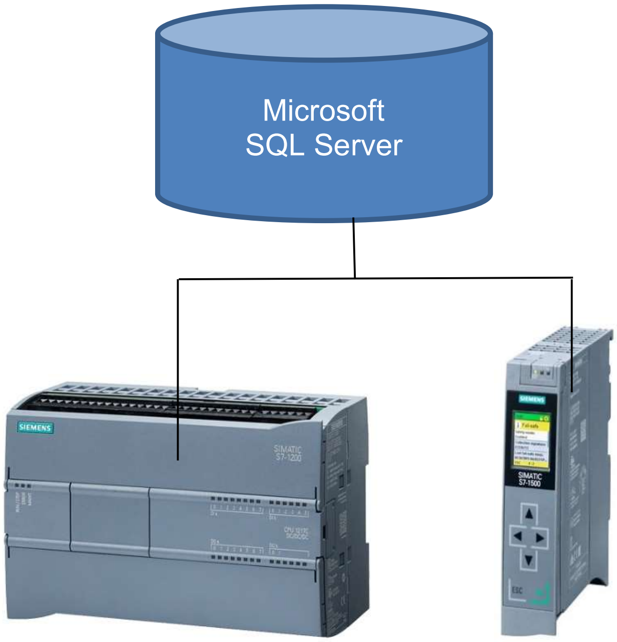
ALSMATIK

Installation Manual

PLCSQL Lite With Siemens TIA S7 1200 and S7 1500

SQL Client in a
Siemens S7 PLC



Revision date: 2020-01-24
Revised by: FBH, Alsmatik A/S
Version: 1.00

Contents

Introduction

You can operate the PLCSQL Lite with Microsoft SQL Server

You need to know the database Name, Tablename and how the data are presented in the columns.

If you still have questions after reading this manual, please send them to info@plcsql.com

Prerequisites

The user of the PLCSQL software must have good knowledge of using Tia Portal and the possibilities of "drag and drop" between different projects.

⚠ Attention!

We have been exposed to problems regarding to integrate the PLCSQL Project into a User project. The problem that comes up, is that the "SQL_Client" (protected block) has to be compiled again before there can be downloaded to the PLC.

Please ask for a Library file if you have this problem.

If the user project is integrated into the PLCSQL project, there is no problem.

Software requirements

This example project is based on following software tools:

PLC Program:	Siemens Tia Portal V 15.1 Update 1
HMI:	Siemens Tia Portal V 15.1 Update 1

In both cases the latest updates are required

If you need another version, please let us know, and we will convert it for you.

Hardware requirements

S7 1200 PLC with firmware **4.2**.

S7 1500 PLC with firmware **1.8** or **2.X**.

Open Controller with firmware **2.X**

How does the system work?

We have made a system where it is possible to communicate with a SQL data base without being an “SQL expert”, nor being an “PLC expert” regarding communication etc. the system contains a “standard” PLC program, that communicates with a “standard” SQL data base.

In the PLC, we are using the basic tag types:

Bool. Is stored in an “Bool” table in the data base.
Int. Is stored in an “Int” table in the data base.
Dint. Is stored in an “Dint” table in the data base.
Real Is stored in an “Real” table in the data base.
String Is stored in an “String” table in the data base.

To distinguish between the different tags, every tag has a specific number.
In the PLC, there is an “Array” that contains all the tags.

On the following pages, there is a schematic view of the layout and the possibilities you have with the PLCSQL system.

Please note the following.

This is a Lite system where it is possible to Log and Receive data for only 6 Columns. Furthermore, it is only possible to Log 3 sets of Data of each Data Type.

The data are as followed:
LOG:
Values are Written in the PLC:
REAL ParamREAL{1} @ real in SQL database
ParamREAL{2} @ real in SQL database
ParamREAL{3} @ real in SQL database

INT ParamINT{10001} @ smallint in SQL database
ParamINT{10002} @ smallint in SQL database
ParamINT{10003} @ smallint in SQL database

DINT ParamDINT{15001} @ int in SQL database
ParamDINT{15002} @ int in SQL database
ParamDINT{15003} @ int in SQL database

BOOL ParamBOOL{20001} @ bit in SQL database
ParamBOOL{20002} @ bit in SQL database
ParamBOOL{20003} @ bit in SQL database

STRING	ParamSTRING{30001} @ nvarchar(50) in SQL database
	ParamSTRING{30002} @ nvarchar(50) in SQL database
	ParamSTRING{30003} @ nvarchar(50) in SQL database

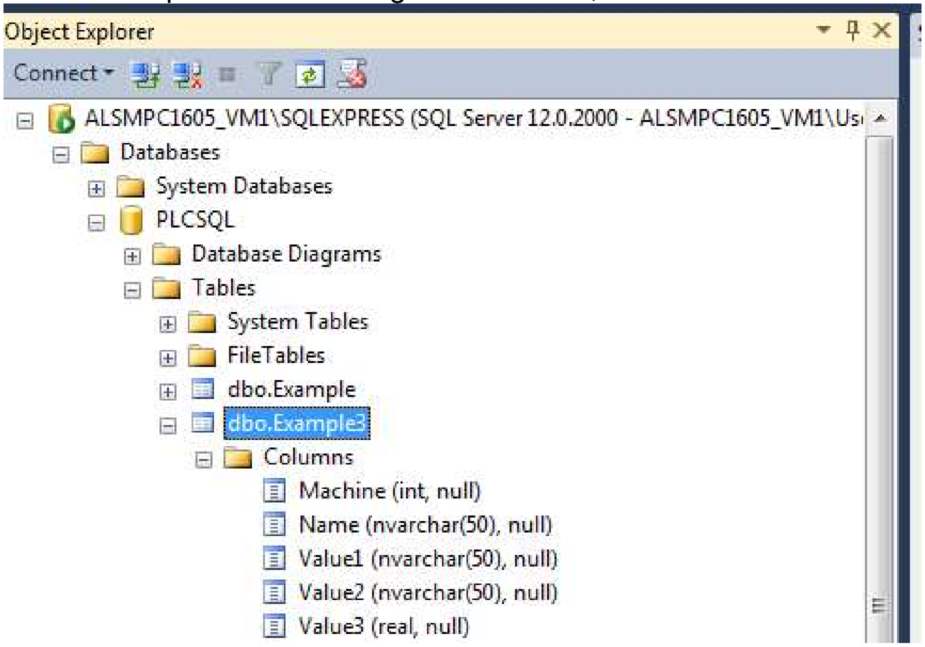
Recipe:

Values are Stored in the PLC:

REAL	ParamREAL{1} @ real in SQL database
	ParamREAL{2} @ real in SQL database
	ParamREAL{3} @ real in SQL database
INT	ParamINT{10001} @ smallint in SQL database
	ParamINT{10002} @ smallint in SQL database
	ParamINT{10003} @ smallint in SQL database
DINT	ParamDINT{15001} @ int in SQL database
	ParamDINT{15002} @ int in SQL database
	ParamDINT{15003} @ int in SQL database
BOOL	ParamBOOL{20001} @ bit in SQL database
	ParamBOOL{20002} @ bit in SQL database
	ParamBOOL{20003} @ bit in SQL database
STRING	ParamSTRING{30001} @ nvarchar(50) in SQL database
	ParamSTRING{30002} @ nvarchar(50) in SQL database
	ParamSTRING{30003} @ nvarchar(50) in SQL database

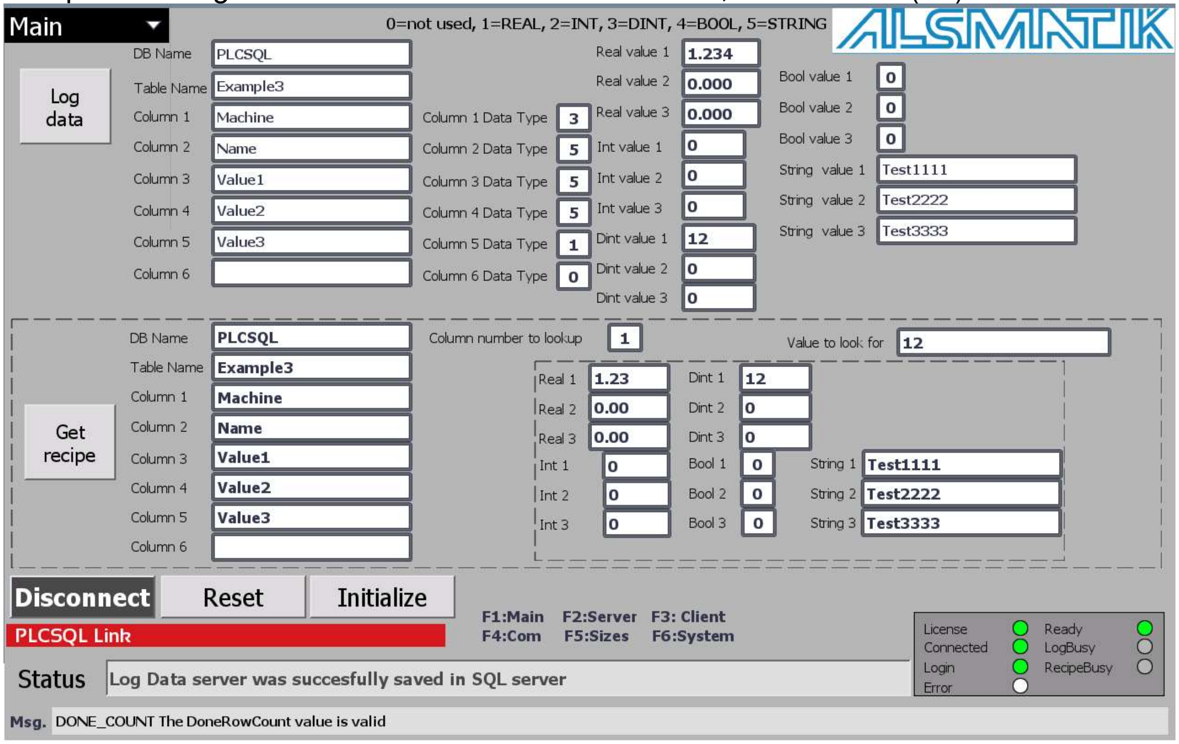
How does PLCSQL work, Log, write to SQL server
In this example From the KTP1200 basic panel (Included in the PLCSQL)

In the SQL database we have a table named Example3 we want to Log data into.
In this example we want to log 1 DINT value, 3 STRING values and 1 REAL value.



- ALSMPC1605_VM1\SQLEXPRESS (SQL Server 12.0.2000 - ALSMPC1605_VM1\Us
  - Databases
    - System Databases
    - PLCSQL
      - Database Diagrams
      - Tables
        - System Tables
        - FileTables
        - dbo.Example
        - dbo.Example3
          - Columns
            - Machine (int, null)
            - Name (nvarchar(50), null)
            - Value1 (nvarchar(50), null)
            - Value2 (nvarchar(50), null)
            - Value3 (real, null)

Setup and configure the Data from the Panel software, Main screen (F1)



0=not used, 1=REAL, 2=INT, 3=DINT, 4=BOOL, 5=STRING

DB Name	PLCSQL			Real value 1	1.234		
Table Name	Example3			Real value 2	0.000	Bool value 1	0
Column 1	Machine	Column 1 Data Type	3	Real value 3	0.000	Bool value 2	0
Column 2	Name	Column 2 Data Type	5	Int value 1	0	Bool value 3	0
Column 3	Value1	Column 3 Data Type	5	Int value 2	0	String value 1	Test1111
Column 4	Value2	Column 4 Data Type	5	Int value 3	0	String value 2	Test2222
Column 5	Value3	Column 5 Data Type	1	Dint value 1	12	String value 3	Test3333
Column 6		Column 6 Data Type	0	Dint value 2	0		
				Dint value 3	0		

DB Name	PLCSQL	Column number to lookup	1					Value to look for	12
Table Name	Example3								
Column 1	Machine	Real 1	1.23	Dint 1	12				
Column 2	Name	Real 2	0.00	Dint 2	0				
Column 3	Value1	Real 3	0.00	Dint 3	0				
Column 4	Value2	Int 1	0	Bool 1	0	String 1	Test1111		
Column 5	Value3	Int 2	0	Bool 2	0	String 2	Test2222		
Column 6		Int 3	0	Bool 3	0	String 3	Test3333		

Status: Log Data server was succesfully saved in SQL server

Msg. DONE_COUNT The DoneRowCount value is valid

DB Name: Name of the database in this case **PLCSQL**
Table Name: Name of the Table, in this case **Example3**
Column 1: Name of the first column in the SQL Table, in this case **Machine**
Column 2: Name of the second column in the SQL Table, in this case **Name**
Column 3: Name of the third column in the SQL Table, in this case **Value1**
Column 4: Name of the fourth column in the SQL Table, in this case **Value2**
Column 5: Name of the fifth column in the SQL Table, in this case **Value3**
Column 6: Name of the sixth column in the SQL Table, in this case not used.

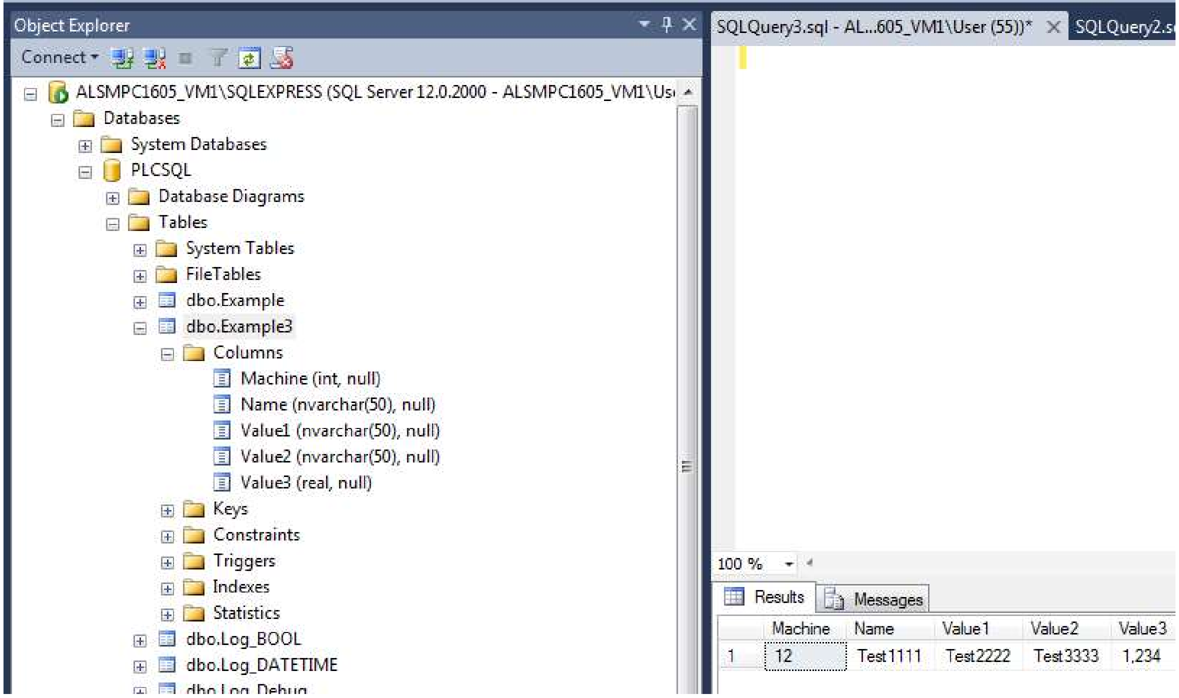
For the next setup use the following for reference:
0=not used, 1=REAL, 2=INT, 3=DINT, 4=BOOL, 5=STRING

Column 1 Data Type: Data Type of the first column, in this case **3=DINT**
Column 2 Data Type: Data Type of the second column, in this case **5=STRING**
Column 3 Data Type: Data Type of the third column, in this case **5=STRING**
Column 4 Data Type: Data Type of the fourth column, in this case **5=STRING**
Column 5 Data Type: Data Type of the fifth column, in this case 1**=REAL**
Column 6 Data Type: Data Type of the sixth column, in this case **0=Not used**

The values we want to store:
Real value 1,2,3 – In this case we only need 1 Real value, needs to be written in Real Value 1
Int value 1,2,3 – In this case we don't want to store any Int values so we can leave them empty=0
Dint value 1,2,3 – In this case we only need to 1 Dint value, needs to be written in Dint value 1

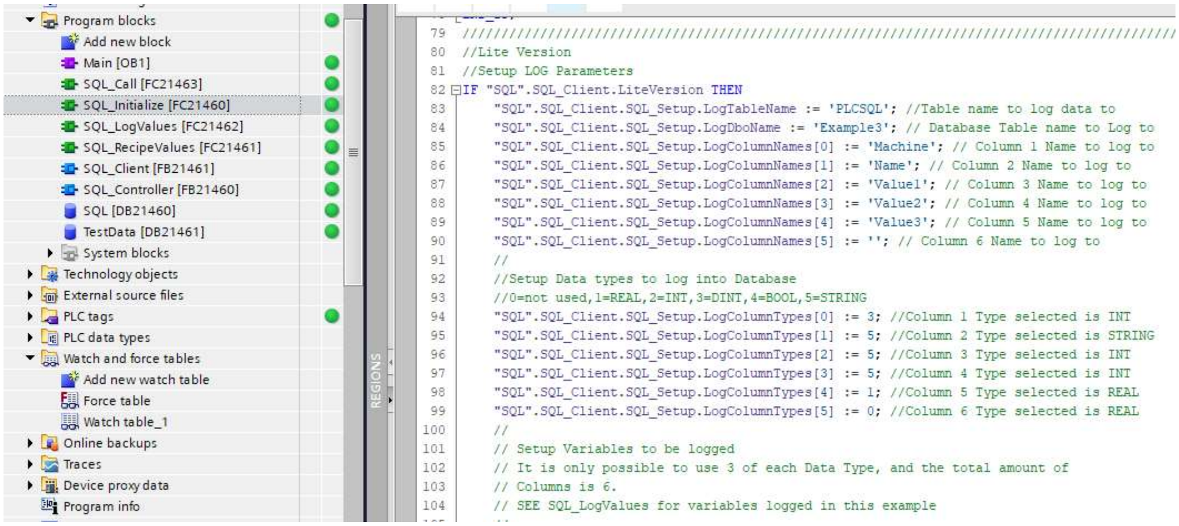
Bool value 1,2,3 - In this case we don’t want to store any Int values so we can leave them empty=0
String value 1,2,3 – In this case we want to store all 3 values so we write a value into all 3 strings

Now you can press the Log data button and the following will be logged in the database:



	Machine	Name	Value1	Value2	Value3
1	12	Test1111	Test2222	Test3333	1,234

Setup and configure the Data from the PLC
In the FC21460 called SQL_Initialize type in the name and values you want to Log.

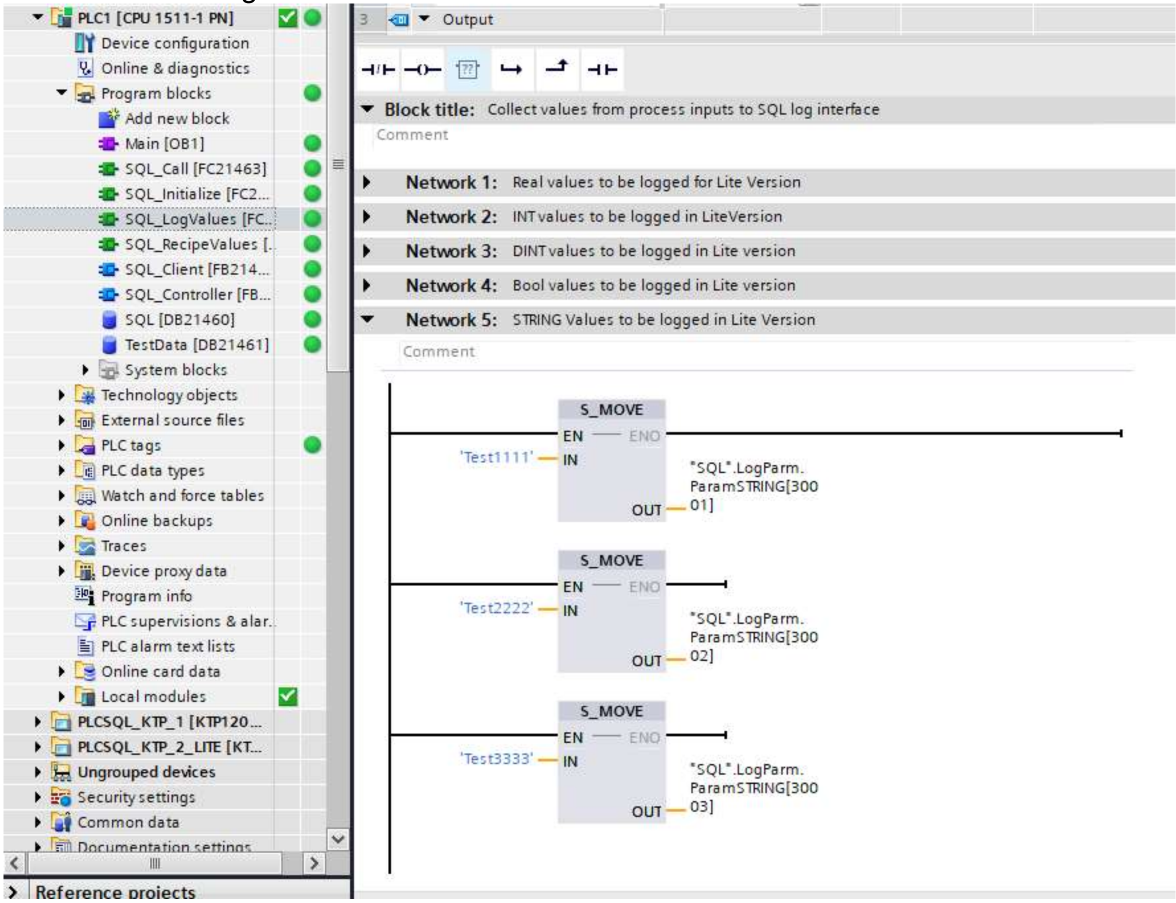


```
//////////////////////////////////////////////////////////////////////////////////////////
//Lite Version
//Setup LOG Parameters
IF "SQL".SQL_Client.LiteVersion THEN
    "SQL".SQL_Client.SQL_Setup.LogTableName := 'PLCSQL'; //Table name to log data to
    "SQL".SQL_Client.SQL_Setup.LogDboName := 'Example3'; // Database Table name to Log to
    "SQL".SQL_Client.SQL_Setup.LogColumnNames[0] := 'Machine'; // Column 1 Name to log to
    "SQL".SQL_Client.SQL_Setup.LogColumnNames[1] := 'Name'; // Column 2 Name to log to
    "SQL".SQL_Client.SQL_Setup.LogColumnNames[2] := 'Value1'; // Column 3 Name to log to
    "SQL".SQL_Client.SQL_Setup.LogColumnNames[3] := 'Value2'; // Column 4 Name to log to
    "SQL".SQL_Client.SQL_Setup.LogColumnNames[4] := 'Value3'; // Column 5 Name to log to
    "SQL".SQL_Client.SQL_Setup.LogColumnNames[5] := ''; // Column 6 Name to log to
    //
    //Setup Data types to log into Database
    //0=not used,1=REAL,2=INT,3=DINT,4=BOOL,5=STRING
    "SQL".SQL_Client.SQL_Setup.LogColumnTypes[0] := 3; //Column 1 Type selected is INT
    "SQL".SQL_Client.SQL_Setup.LogColumnTypes[1] := 5; //Column 2 Type selected is STRING
    "SQL".SQL_Client.SQL_Setup.LogColumnTypes[2] := 5; //Column 3 Type selected is INT
    "SQL".SQL_Client.SQL_Setup.LogColumnTypes[3] := 5; //Column 4 Type selected is INT
    "SQL".SQL_Client.SQL_Setup.LogColumnTypes[4] := 1; //Column 5 Type selected is REAL
    "SQL".SQL_Client.SQL_Setup.LogColumnTypes[5] := 0; //Column 6 Type selected is REAL
    //
    // Setup Variables to be logged
    // It is only possible to use 3 of each Data Type, and the total amount of
    // Columns is 6.
    // SEE SQL_LogValues for variables logged in this example
```

Type in the names and values needed , in this case the same values and numbers as described in the previous section of this document.
Save the file and download it.

Call the SQL_Initialize, here it is done from the FC21463 SQL_Call with a Marked bit M1.4 which is also used from the Panel HMI button Initialize.

Data that have to be logged are in this case made in the FC21462 SQL_LogValues. Here are the String values shown:



Block title: Collect values from process inputs to SQL log interface

Network 1: Real values to be logged for Lite Version

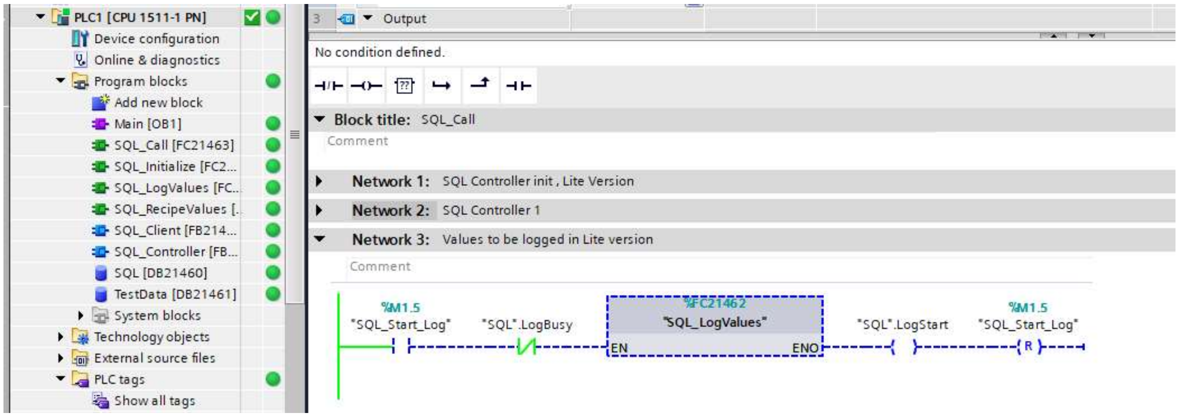
Network 2: INT values to be logged in LiteVersion

Network 3: DINT values to be logged in Lite version

Network 4: Bool values to be logged in Lite version

Network 5: STRING Values to be logged in Lite Version

Now you have to call the FC21462 SQL_LogValues. In this case done with the marked bit M1.5. LogBusy has to be inactive , and after you have set the values you set the bit SQL.LogStart.



No condition defined.

Block title: SQL_Call

Network 1: SQL Controller init , Lite Version

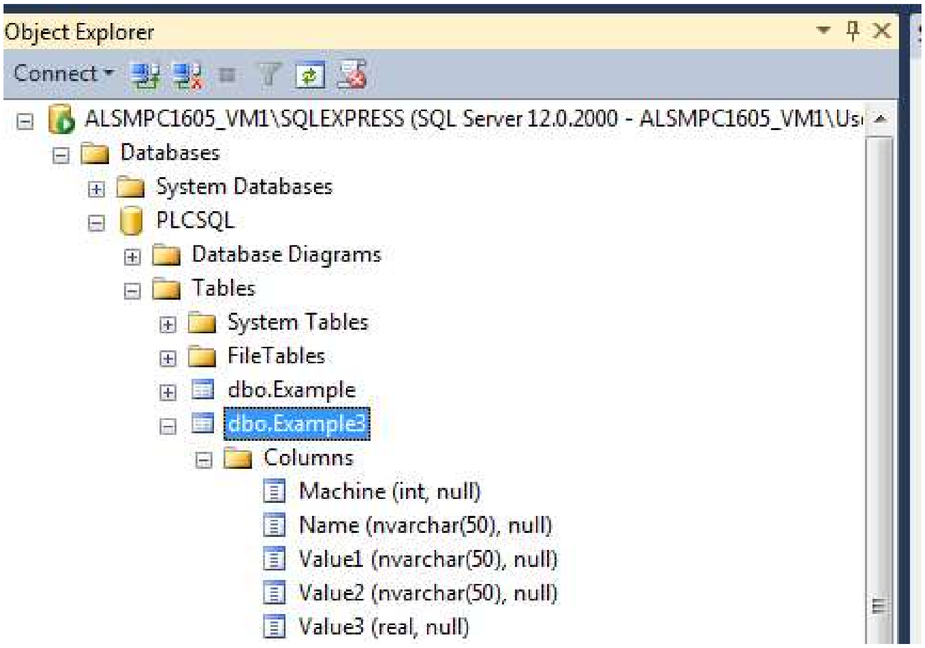
Network 2: SQL Controller 1

Network 3: Values to be logged in Lite version

How does PLCSQL work, Recipe, read from SQL server

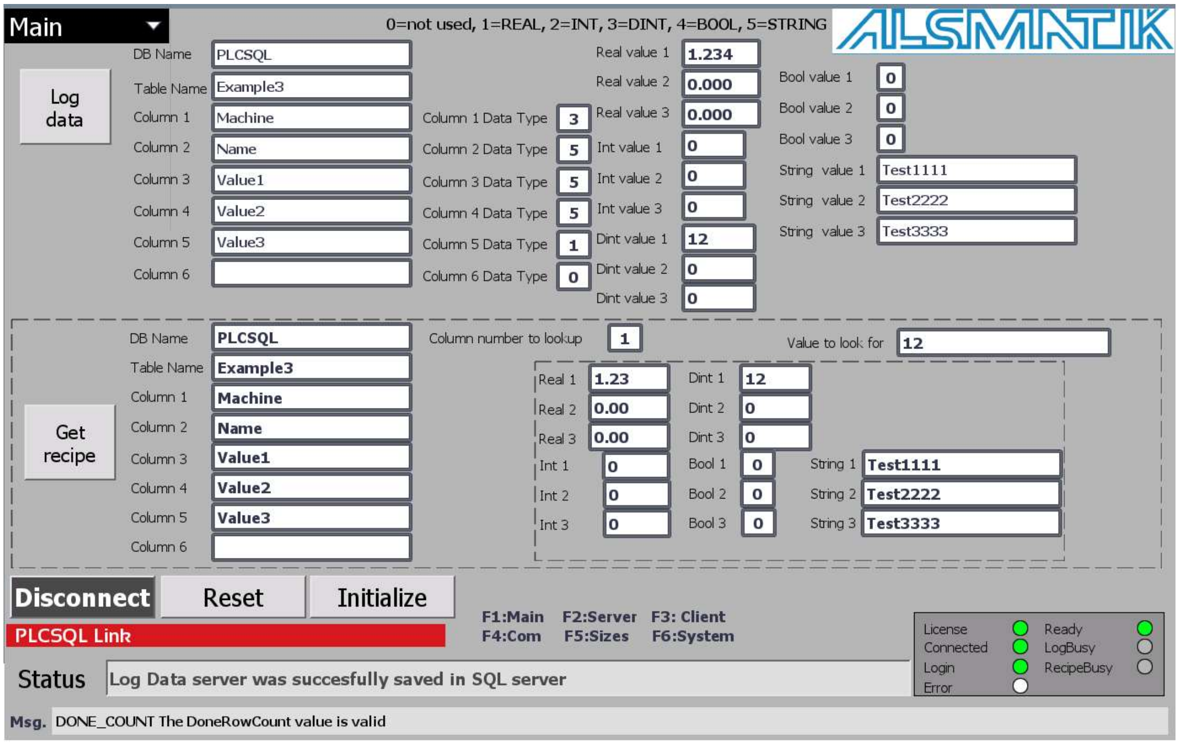
In this case from the KTP1200 basic panel (Included in the PLCSQL)

In the SQL database we have a table named Example3 we want to Recipe data from. If you look in the database there are 5 Columns, Called Machine, Name, Value1, Value2, Value3 , with the data int, nvarchar(50), nvarchar(50), nvarchar(50), real.



```
⊟ ALSMPC1605_VM1\SQLEXPRESS (SQL Server 12.0.2000 - ALSMPC1605_VM1\Us
   ⊟ Databases
      ⊞ System Databases
      ⊟ PLCSQL
         ⊞ Database Diagrams
         ⊟ Tables
            ⊞ System Tables
            ⊞ FileTables
            ⊞ dbo.Example
            ⊟ dbo.Example3
               ⊟ Columns
                     Machine (int, null)
                     Name (nvarchar(50), null)
                     Value1 (nvarchar(50), null)
                     Value2 (nvarchar(50), null)
                     Value3 (real, null)
```

In the Panel HMI we need to define the database name, Table name, and the column names used:



0=not used, 1=REAL, 2=INT, 3=DINT, 4=BOOL, 5=STRING

DB Name	PLCSQL			Real value 1	1.234		
Table Name	Example3			Real value 2	0.000	Bool value 1	0
Column 1	Machine	Column 1 Data Type	3	Real value 3	0.000	Bool value 2	0
Column 2	Name	Column 2 Data Type	5	Int value 1	0	Bool value 3	0
Column 3	Value1	Column 3 Data Type	5	Int value 2	0	String value 1	Test1111
Column 4	Value2	Column 4 Data Type	5	Int value 3	0	String value 2	Test2222
Column 5	Value3	Column 5 Data Type	1	Dint value 1	12	String value 3	Test3333
Column 6		Column 6 Data Type	0	Dint value 2	0		
				Dint value 3	0		

DB Name	PLCSQL	Column number to lookup	1			Value to look for	12		
Table Name	Example3								
Column 1	Machine	Real 1	1.23	Dint 1	12				
Column 2	Name	Real 2	0.00	Dint 2	0				
Column 3	Value1	Real 3	0.00	Dint 3	0				
Column 4	Value2	Int 1	0	Bool 1	0	String 1	Test1111		
Column 5	Value3	Int 2	0	Bool 2	0	String 2	Test2222		
Column 6		Int 3	0	Bool 3	0	String 3	Test3333		

F1:Main F2:Server F3: Client
F4:Com F5:Sizes F6:System

License	●	Ready	●
Connected	●	LogBusy	○
Login	●	RecipeBusy	○
Error	○		

Status: Log Data server was succesfully saved in SQL server

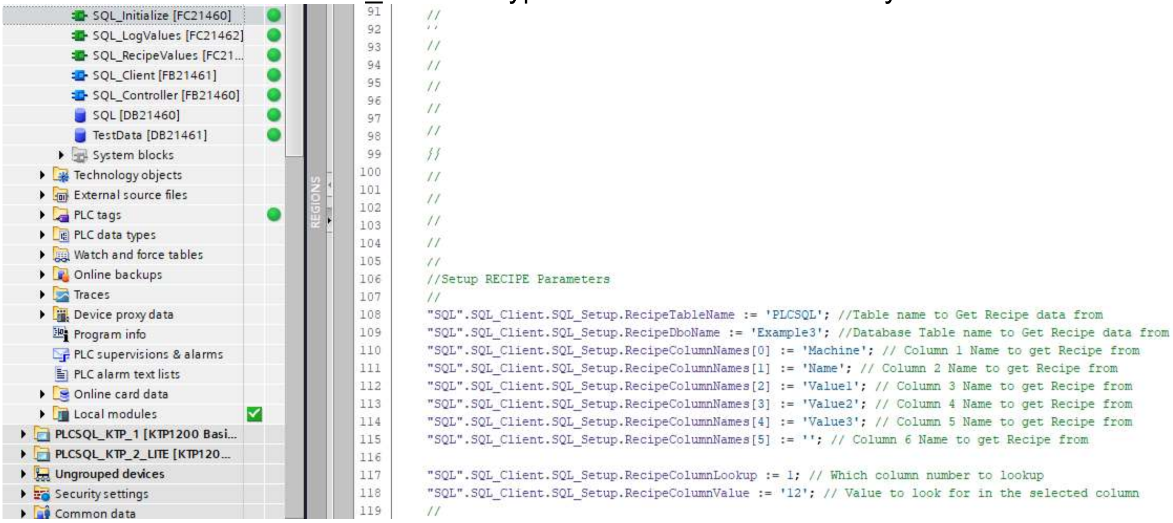
Msg. DONE_COUNT The DoneRowCount value is valid

DB Name: Name of the database to use, in this case **PLCSQL**
Table Name: Name of the Table to use, in this case **Example3**
Column 1: Name of the first column in the SQL Table, in this case **Machine**
Column 2: Name of the second column in the SQL Table, in this case **Name**
Column 3: Name of the third column in the SQL Table, in this case **Value1**
Column 4: Name of the fourth column in the SQL Table, in this case **Value2**
Column 5: Name of the fifth column in the SQL Table, in this case **Value3**
Column 6: Name of the sixth column in the SQL Table, in this case not used
Column number to lookup: in this case we want to look in **column number 1**
Value to look for: the value or word we want to look for, in this case **value 12**

Press the button Get recipe, now the values from the database are collected.
The values are stored in the tags described earlier in this documentation. In this case

Dint 1 in ParamDINT{1} from Machine in the table
String 1 ParamSTRING{1} from Name in the table
String 2 ParamSTRING{2} from Value1 in the table
String 3 ParamSTRING{3} from Value2 in then table
Real 1   ParamREAL{1} from Value3 in the table

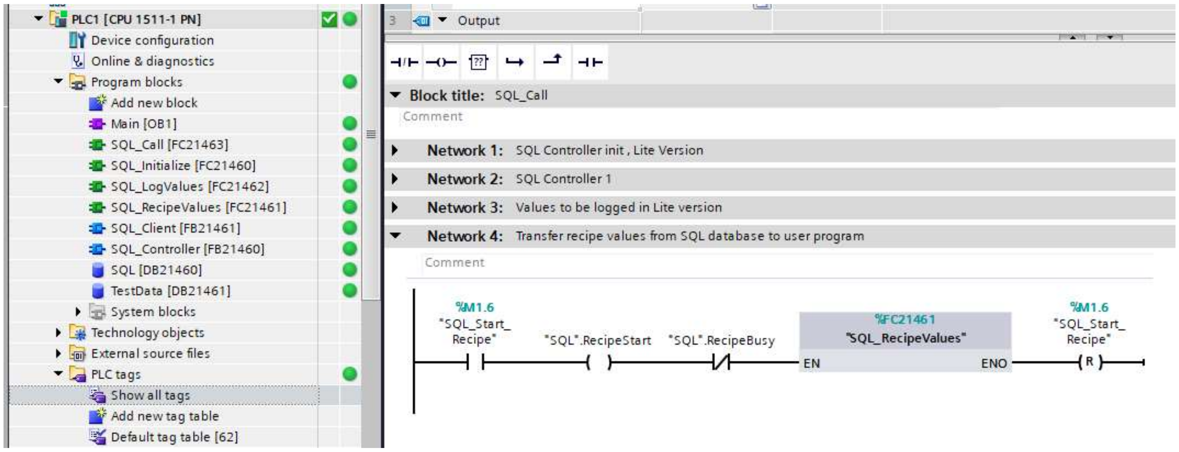
Setup and configure the Data from the PLC
In the FC21460 called SQL_Initialize type in the name and values you want to receive.



```
//
//
//
//
//
//
//
//
//
//
//
//
//
//
//
//Setup RECIPE Parameters
//
"SQL".SQL_Client.SQL_Setup.RecipeTableName := 'PLCSQL'; //Table name to Get Recipe data from
"SQL".SQL_Client.SQL_Setup.RecipeDboName := 'Example3'; //Database Table name to Get Recipe data from
"SQL".SQL_Client.SQL_Setup.RecipeColumnNames[0] := 'Machine'; // Column 1 Name to get Recipe from
"SQL".SQL_Client.SQL_Setup.RecipeColumnNames[1] := 'Name'; // Column 2 Name to get Recipe from
"SQL".SQL_Client.SQL_Setup.RecipeColumnNames[2] := 'Value1'; // Column 3 Name to get Recipe from
"SQL".SQL_Client.SQL_Setup.RecipeColumnNames[3] := 'Value2'; // Column 4 Name to get Recipe from
"SQL".SQL_Client.SQL_Setup.RecipeColumnNames[4] := 'Value3'; // Column 5 Name to get Recipe from
"SQL".SQL_Client.SQL_Setup.RecipeColumnNames[5] := ''; // Column 6 Name to get Recipe from

"SQL".SQL_Client.SQL_Setup.RecipeColumnLookup := 1; // Which column number to lookup
"SQL".SQL_Client.SQL_Setup.RecipeColumnValue := '12'; // Value to look for in the selected column
//
```

Call the SQL_Initialize, here it is done from the FC21463 SQL_Call with a Marked bit M1.4 which is also used from the Panel HMI button Initialize.

Now you have to call the FC21461 SQL_RecipeValues. In this case done with the marked bit M1.6. RecipeBusy has to be inactive, before you call SQL_RecipeValues you have to set the bit SQL.RecipeStart.



Block title: SQL_Call

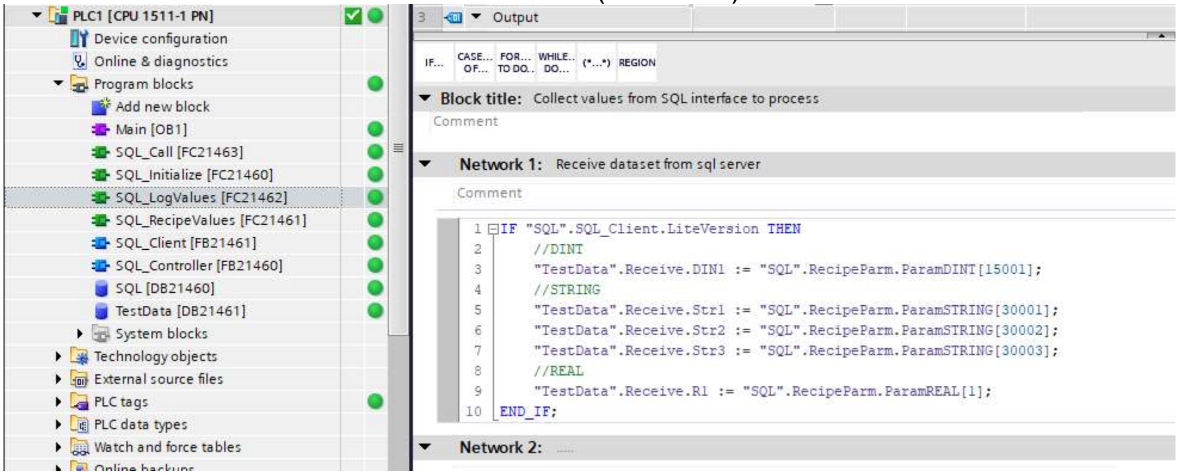
Network 1: SQL Controller init , Lite Version

Network 2: SQL Controller 1

Network 3: Values to be logged in Lite version

Network 4: Transfer recipe values from SQL database to user program

Now the data are stored in the TestData DB (DB21461)



Block title: Collect values from SQL interface to process

Network 1: Receive dataset from sql server

```
IF "SQL".SQL_Client.LiteVersion THEN
    //DINT
    "TestData".Receive.DIN1 := "SQL".RecipeParm.ParamDINT[15001];
    //STRING
    "TestData".Receive.Str1 := "SQL".RecipeParm.ParamSTRING[30001];
    "TestData".Receive.Str2 := "SQL".RecipeParm.ParamSTRING[30002];
    "TestData".Receive.Str3 := "SQL".RecipeParm.ParamSTRING[30003];
    //REAL
    "TestData".Receive.R1 := "SQL".RecipeParm.ParamREAL[1];
END_IF;
```

Network 2: .....

Used Blocks in this example project

⚠

The block numbers are the same for S7 1200 and S7 1500 PLC's but there are major differences in the code due to differences in the hardware of the 2 types of PLC's, so be careful to use the correct blocks.

In the example project, we are using the following blocks:

FC 21463	SQL_Call
FC 21460	SQL_Initialize
FC 21462	SQL_LogValues
FC 21461	SQL_RecipeValues

FB 21461	SQL_Client (Protected, cannot be renumbered)
FB 21460	SQL_Controller
DB 21460	SQL
DB 21461	Test_Data

Be sure these blocks are free if you copy them into an existing project, or renumber the blocks.

Option HMI, KTP 1200 Basic
The Basic panel is chosen because you always have the possibility to run this type of panel, no matter what version of Tia Portal (Basic / Professional) you have, and you don’t need the option software “WinCC”.

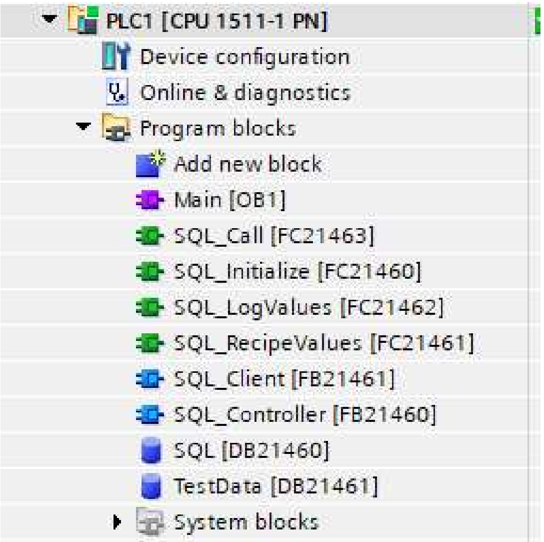
Setting up PLCSQL Link in Siemens TIA Portal

The PLCSQL software is supplied for the specific PLC type you are using (1200 / 1500 / Open Controller).
The software project is a “complete” project with PLC and HMI, if you only want the code blocks in a library, please let us know, then you will get a library.

When you opened the project, you must

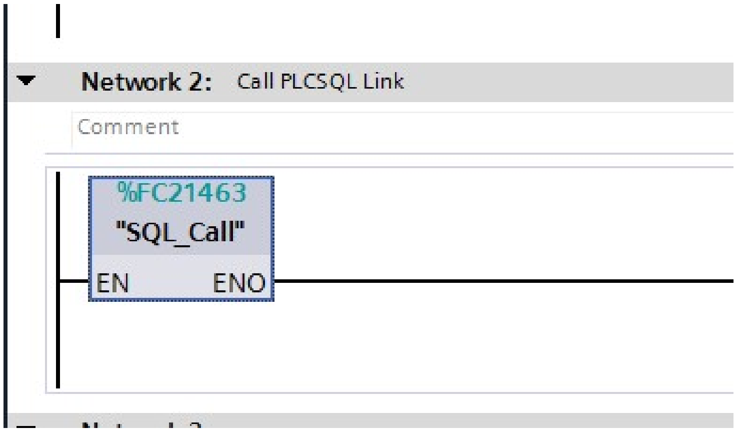
- Set up the hardware configuration to your needs
- Set IP address for the Ethernet port
- Compile it (rebuild all blocks)
- Download the configuration to the PLC.

Software layout



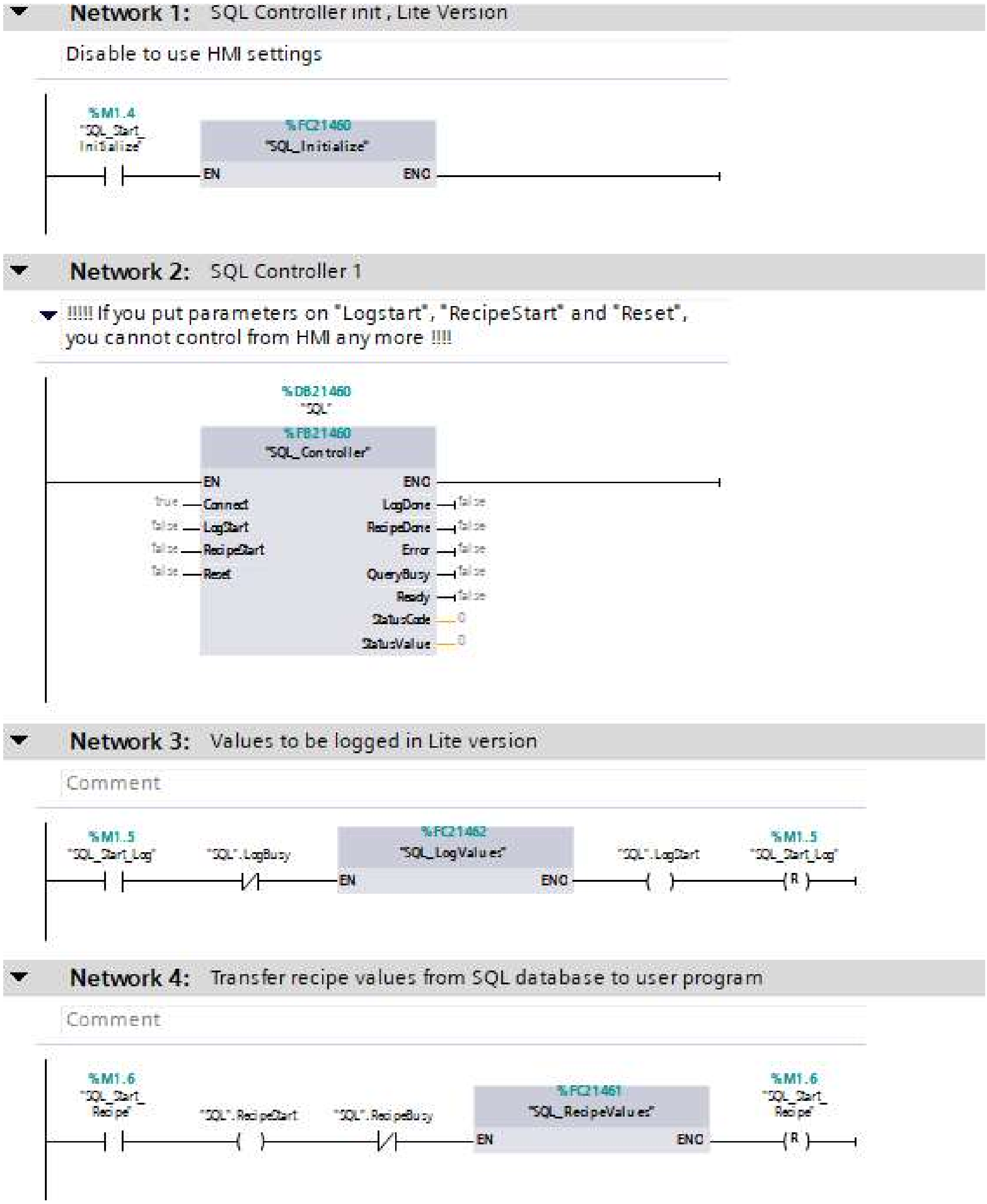
PLC1 [CPU 1511-1 PN]
Device configuration
Online & diagnostics
Program blocks
Add new block
Main [OB1]
SQL_Call [FC21463]
SQL_Initialize [FC21460]
SQL_LogValues [FC21462]
SQL_RecipeValues [FC21461]
SQL_Client [FB21461]
SQL_Controller [FB21460]
SQL [DB21460]
TestData [DB21461]
System blocks

Main OB1



This network controls the PLCSQL system.

SQL_Call FC 21463



Network 1: SQL Controller init , Lite Version

Disable to use HMI settings

Network 2: SQL Controller 1

!!!!! If you put parameters on "Logstart", "RecipeStart" and "Reset", you cannot control from HMI any more !!!!

Network 3: Values to be logged in Lite version

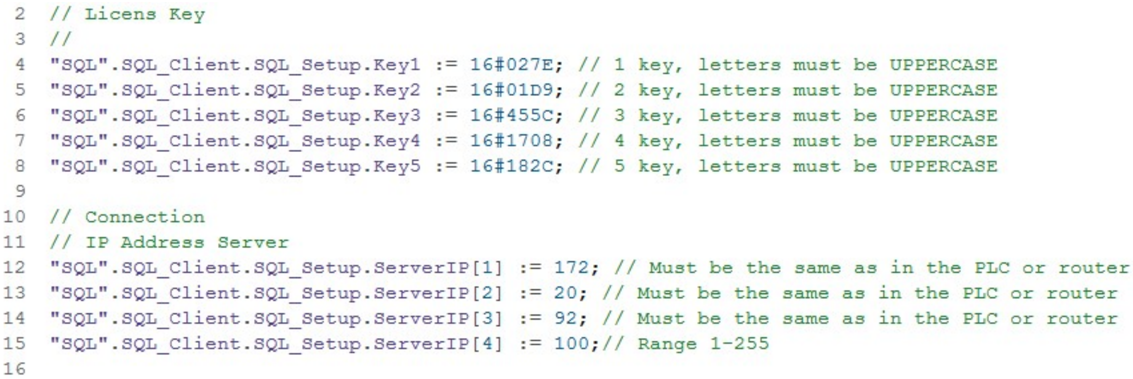
Comment

Network 4: Transfer recipe values from SQL database to user program

Comment

Call structure of the PLCSQL Link system, if the order is changed, there is no warranty for correct function.

SQL_Initialize FC 21460

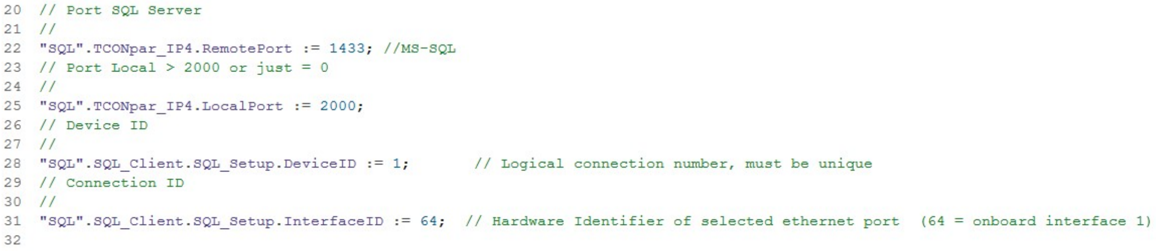


```
// Licens Key
//
"SQL".SQL_Client.SQL_Setup.Key1 := 16#027E; // 1 key, letters must be UPPERCASE
"SQL".SQL_Client.SQL_Setup.Key2 := 16#01D9; // 2 key, letters must be UPPERCASE
"SQL".SQL_Client.SQL_Setup.Key3 := 16#455C; // 3 key, letters must be UPPERCASE
"SQL".SQL_Client.SQL_Setup.Key4 := 16#1708; // 4 key, letters must be UPPERCASE
"SQL".SQL_Client.SQL_Setup.Key5 := 16#182C; // 5 key, letters must be UPPERCASE

// Connection
// IP Address Server
"SQL".SQL_Client.SQL_Setup.ServerIP[1] := 172; // Must be the same as in the PLC or router
"SQL".SQL_Client.SQL_Setup.ServerIP[2] := 20; // Must be the same as in the PLC or router
"SQL".SQL_Client.SQL_Setup.ServerIP[3] := 92; // Must be the same as in the PLC or router
"SQL".SQL_Client.SQL_Setup.ServerIP[4] := 100;// Range 1-255
```

“License Key”, here you type the license key that match the serial number of the CPU or the serial number of the used memory card.

“IP Address Server”, here you type the address of the SQL server.

Hardware setup S7 1200 /1500 / Open Controller PLC’s

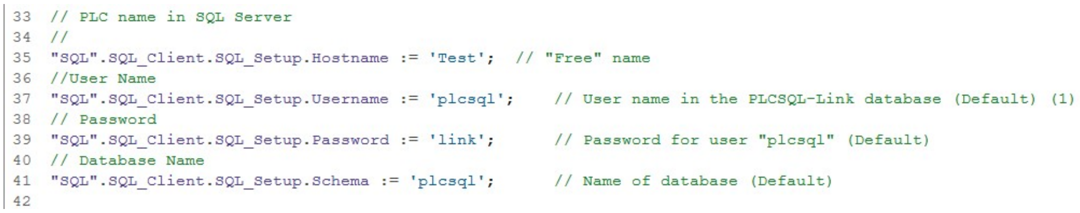


```
// Port SQL Server
//
"SQL".TCONpar_IP4.RemotePort := 1433; //MS-SQL
// Port Local > 2000 or just = 0
//
"SQL".TCONpar_IP4.LocalPort := 2000;
// Device ID
//
"SQL".SQL_Client.SQL_Setup.DeviceID := 1;       // Logical connection number, must be unique
// Connection ID
//
"SQL".SQL_Client.SQL_Setup.InterfaceID := 64;  // Hardware Identifier of selected ethernet port  (64 = onboard interface 1)
```

“Port SQL Server”, here you type the port number of the SQL server.
“Port Local”, here you type the port number to use in the local PLC.

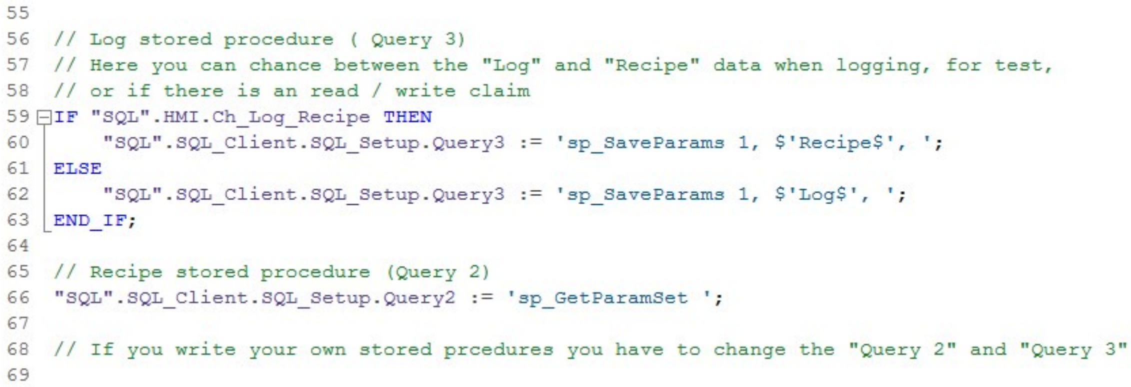
⚠ REMEMBER to restart the PLC if you change “Device ID” in RUN.
“Device ID”, here you typical type a “1”, if you want to call the “SQL” system multiple times, then this number must be unique for every instance.

“Interface ID”, here you type the “Hardware ID” of the selected Ethernet card that connects to the SQL server.
The “first” (build in) network card has always the ID “64” in all PLC types.
The “second” (build in) network card has the ID “72”, that applies only to 1500 PLC.



```
// PLC name in SQL Server
//
"SQL".SQL_Client.SQL_Setup.Hostname := 'Test';  // "Free" name
//User Name
"SQL".SQL_Client.SQL_Setup.Username := 'plcsql';    // User name in the PLCSQL-Link database (Default) (1)
// Password
"SQL".SQL_Client.SQL_Setup.Password := 'link';      // Password for user "plcsql" (Default)
// Database Name
"SQL".SQL_Client.SQL_Setup.Schema := 'plcsql';      // Name of database (Default)

```

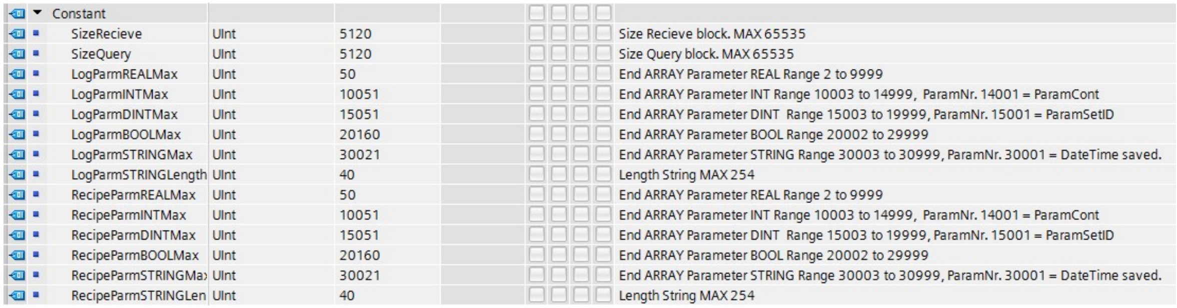
“PLC name in SQL Server”, here you can type just what you want.
“User Name”, here you type the name of the “user” that connects to the SQL Server.
!! It is the “user” that decide which database there is connected to.
“Password”, here you type the password of the “user” that connects to the server.
“Database Name”, option, no use.



```

// Log stored procedure ( Query 3)
// Here you can chance between the "Log" and "Recipe" data when logging, for test,
// or if there is an read / write claim
IF "SQL".HMI.Ch_Log_Recipe THEN
    "SQL".SQL_Client.SQL_Setup.Query3 := 'sp_SaveParams 1, $'Recipe$', ';
ELSE
    "SQL".SQL_Client.SQL_Setup.Query3 := 'sp_SaveParams 1, $'Log$', ';
END_IF;

// Recipe stored procedure (Query 2)
"SQL".SQL_Client.SQL_Setup.Query2 := 'sp_GetParamSet ';

// If you write your own stored prcedures you have to change the "Query 2" and "Query 3"

```

Constants

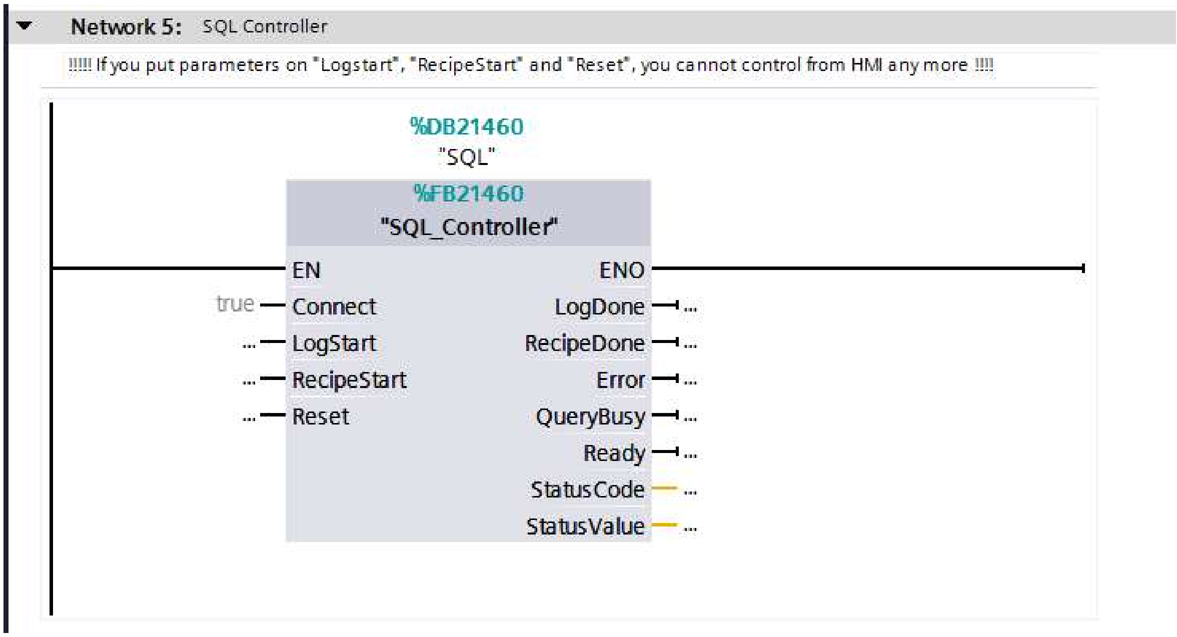
The size of the different data types etc. is defined as constants in the “SQL_Controller” block.



Constant			□ □ □ □	
SizeRecieve	UInt	5120	□ □ □ □	Size Recieve block. MAX 65535
SizeQuery	UInt	5120	□ □ □ □	Size Query block. MAX 65535
LogParmREALMax	UInt	50	□ □ □ □	End ARRAY Parameter REAL Range 2 to 9999
LogParmINTMax	UInt	10051	□ □ □ □	End ARRAY Parameter INT Range 10003 to 14999, ParamNr. 14001 = ParamCont
LogParmDINTMax	UInt	15051	□ □ □ □	End ARRAY Parameter DINT Range 15003 to 19999, ParamNr. 15001 = ParamSetID
LogParmBOOLMax	UInt	20160	□ □ □ □	End ARRAY Parameter BOOL Range 20002 to 29999
LogParmSTRINGMax	UInt	30021	□ □ □ □	End ARRAY Parameter STRING Range 30003 to 30999, ParamNr. 30001 = DateTime saved.
LogParmSTRINGLength	UInt	40	□ □ □ □	Length String MAX 254
RecipeParmREALMax	UInt	50	□ □ □ □	End ARRAY Parameter REAL Range 2 to 9999
RecipeParmINTMax	UInt	10051	□ □ □ □	End ARRAY Parameter INT Range 10003 to 14999, ParamNr. 14001 = ParamCont
RecipeParmDINTMax	UInt	15051	□ □ □ □	End ARRAY Parameter DINT Range 15003 to 19999, ParamNr. 15001 = ParamSetID
RecipeParmBOOLMax	UInt	20160	□ □ □ □	End ARRAY Parameter BOOL Range 20002 to 29999
RecipeParmSTRINGMa:	UInt	30021	□ □ □ □	End ARRAY Parameter STRING Range 30003 to 30999, ParamNr. 30001 = DateTime saved.
RecipeParmSTRINGLen	UInt	40	□ □ □ □	Length String MAX 254

Here are the definitions of the adjustable parameters in the system.
The data types are limited to 3 of each data type

SQL_Controller FB 21460



Network 5: SQL Controller

!!!!! If you put parameters on "Logstart", "RecipeStart" and "Reset", you cannot control from HMI any more !!!!

Overview of the “SQL_Controller”, all parameters can be controlled and seen from the HMI, if you want to control the block from both the HMI and the PLC, the you have to use the “Set” output in the PLC on the parameters “LogStart”, “RecipeStart”, and “Reset”.

SQL_Client FB 21461

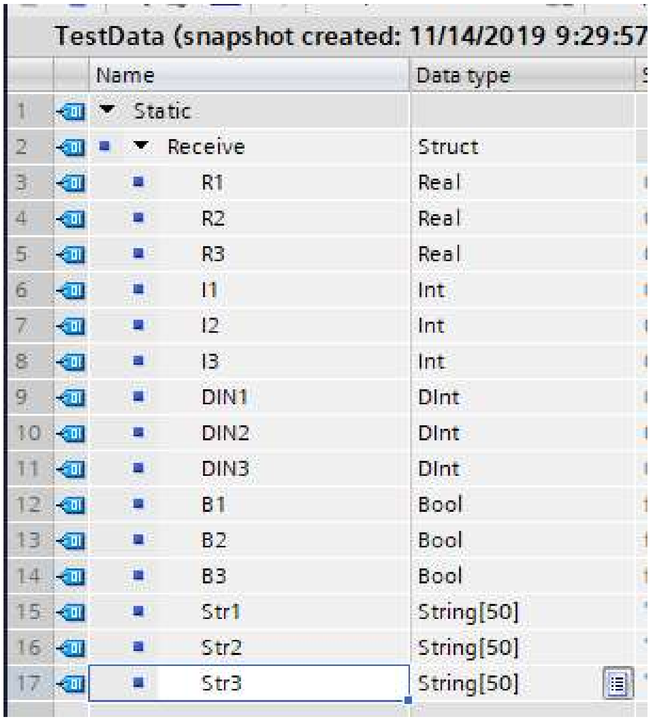
“SQL_Client” is called from “SQL_Controller”, the block is protected and cannot be read or renumbered.

SQL DB 21460

This DB is the Instance DB for the hole SQL system, the DB contains all data areas needed for the SQL system.

Due to the use of only 1 Instance DB, it is very easy to use the SQL system as an “multiple” system, where you can call “SQL_Controller” multiple times by just using a new “SQL” data block, the only limitation is the amount of memory in the PLC.

Test_Data DB 21461

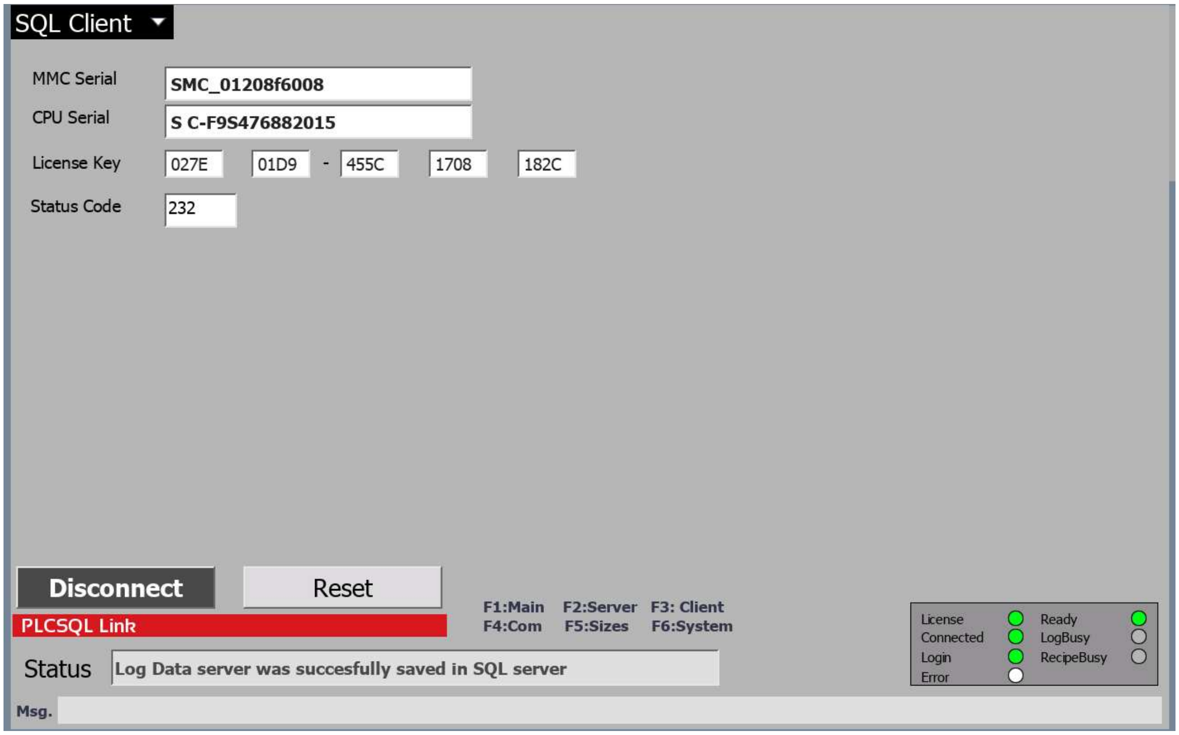


TestData (snapshot created: 11/14/2019 9:29:57

	Name	Data type
1	Static	
2	Receive	Struct
3	R1	Real
4	R2	Real
5	R3	Real
6	I1	Int
7	I2	Int
8	I3	Int
9	DIN1	DInt
10	DIN2	DInt
11	DIN3	DInt
12	B1	Bool
13	B2	Bool
14	B3	Bool
15	Str1	String[50]
16	Str2	String[50]
17	Str3	String[50]

This DB is used to present the data received from the SQL server, if you use your own DB, this block can be deleted.

HMI-SQL Client (License key)



MMC Serial: SMC_01208f6008

CPU Serial: S C-F9S476882015

License Key: 027E 01D9 - 455C 1708 182C

Status Code: 232

Status: Log Data server was succesfully saved in SQL server

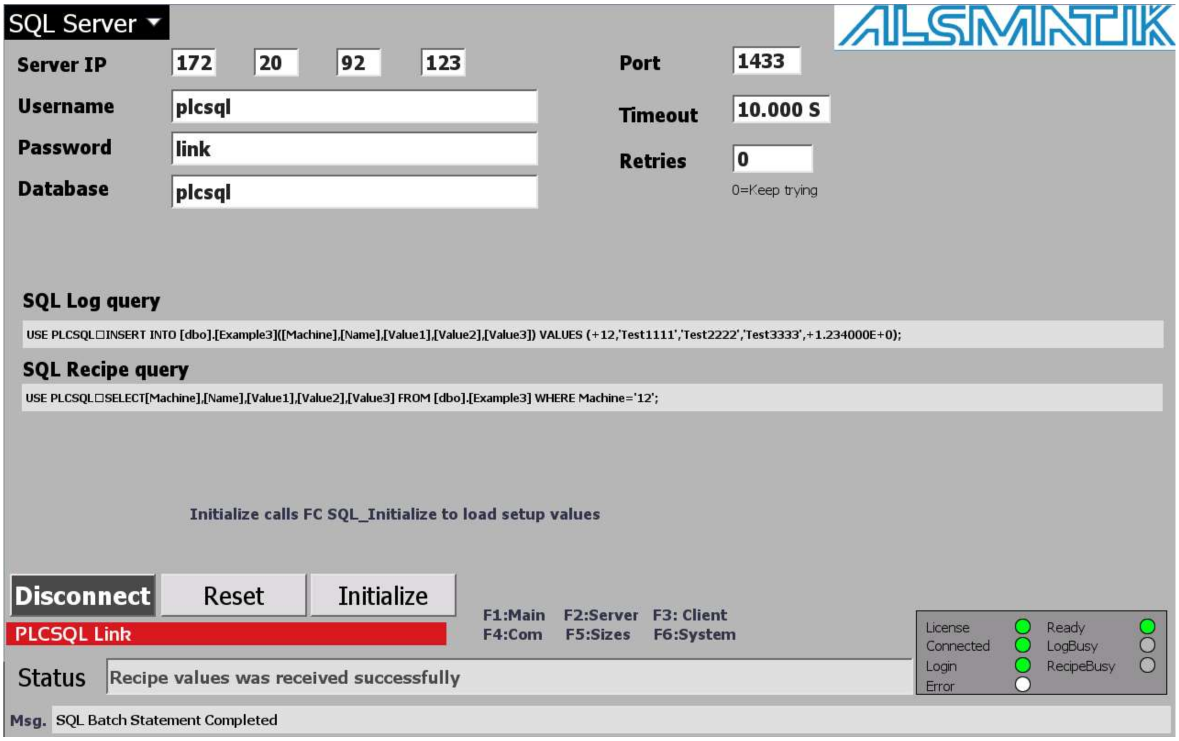
Msg.

The “SQL Client” picture, read out the serial numbers of the CPU and of the memory card.
Here you also can type the license key that you got from Automatic Syd A/S.

⚠

If “SQL_Initialize” is running, you must type the license key in the block.
As default, there is no remanence data in the “SQL” DB, so the hole block is set to default when restarting the PLC.

HMI-SQL Server



Server IP 172 20 92 123

Username plcsql

Password link

Database plcsql

Port 1433

Timeout 10.000 S

Retries 0

0=Keep trying

SQL Log query

```
USE PLCSQL□INSERT INTO [dbo].[Example3]([Machine],[Name],[Value1],[Value2],[Value3]) VALUES (+12,'Test1111','Test2222','Test3333',+1.234000E+0);
```

SQL Recipe query

```
USE PLCSQL□SELECT[Machine],[Name],[Value1],[Value2],[Value3] FROM [dbo].[Example3] WHERE Machine='12';
```

Initialize calls FC SQL_Initialize to load setup values

License	●	Ready	●
Connected	●	LogBusy	○
Login	●	RecipeBusy	○
Error	○		

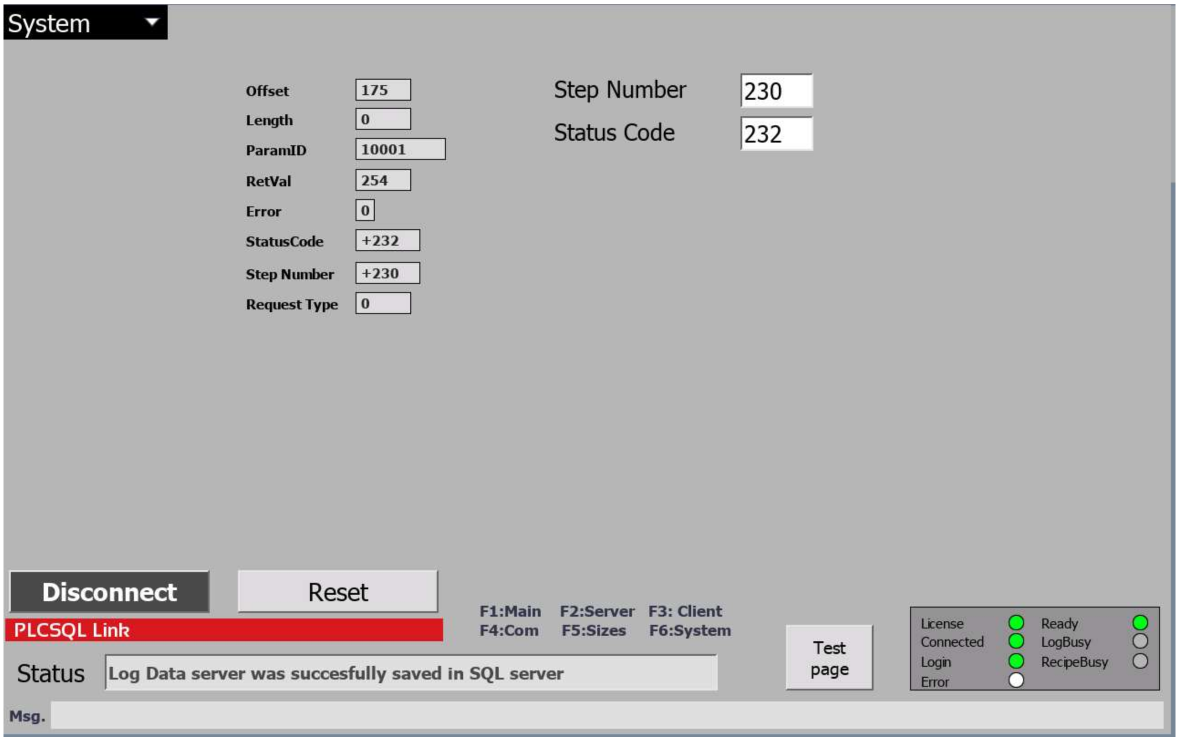
Status Recipe values was received successfully

Msg. SQL Batch Statement Completed

Here you select all the server relevant data.
The shown setup is the DEFAULT setup to match the DEFAULT setup of the Microsoft SQL Server.

If “SQL_Initialize” is running, you must type the changes in this block.
As default, there is no remanence data in the “SQL” DB, so the hole block is set to default when restarting the PLC.

HMI-System



Offset	175
Length	0
ParamID	10001
RetVal	254
Error	0
StatusCode	+232
Step Number	+230
Request Type	0

Step Number	230
Status Code	232

Status: Log Data server was succesfully saved in SQL server

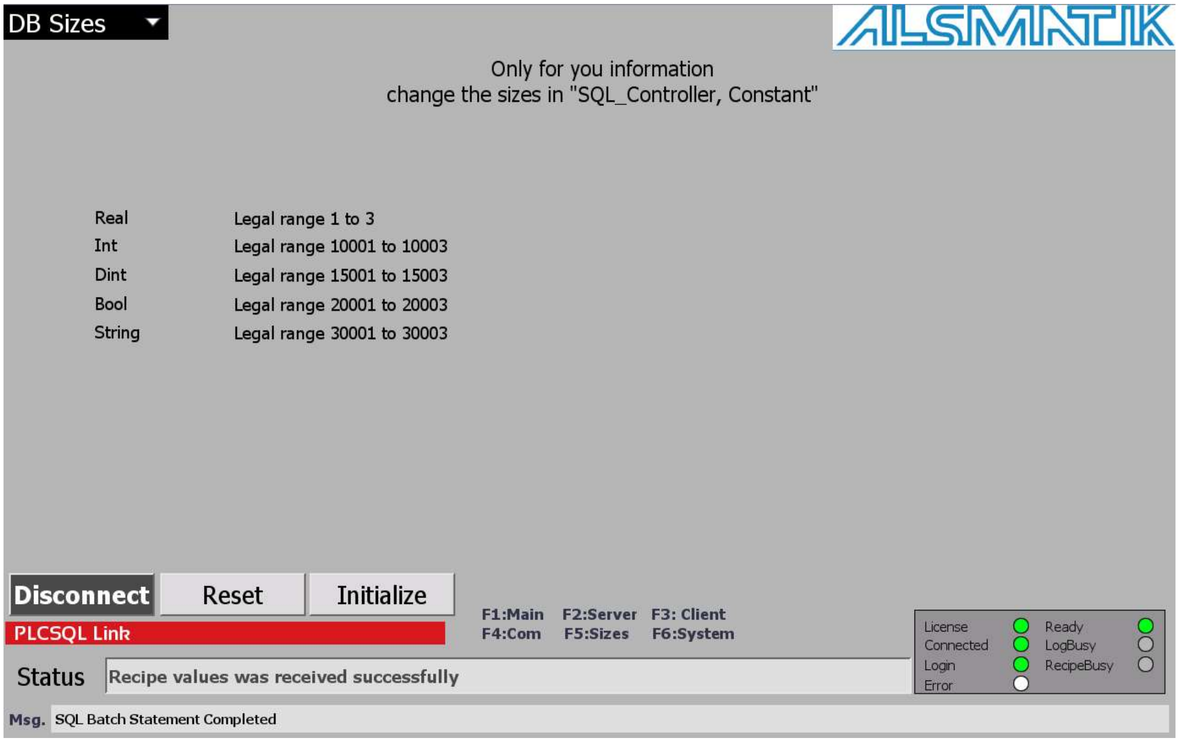
Msg.

License	●	Ready	●
Connected	●	LogBusy	○
Login	●	RecipeBusy	○
Error	○		

In the case of errors from the PLCSQL system, then it is important to get the status from these parameters.

The “Test page” button is used to start the test system, if you delete the “Test” picture you also must delete this button.

HMI-DB Sizes



Only for you information
change the sizes in "SQL_Controller, Constant"

Real	Legal range 1 to 3
Int	Legal range 10001 to 10003
Dint	Legal range 15001 to 15003
Bool	Legal range 20001 to 20003
String	Legal range 30001 to 30003

Status Recipe values was received successfully

Msg. SQL Batch Statement Completed

Only as information to the user.

Adjust the size(s) in the “Constant” area of “PLCSQL_Controller”

Revisions

2020-0127 – Ver 2.4