Application Note

FESTO

How to implement data exchange between CDPX and MSSQL databases

This application note helps the user to implement data exchange between CDPX and MSSQL databases.

CDPX

100814

Title ..... How to implement data exchange between CDPX and MSSQL databases

Version 1.10

Document no. 100814

Original ..... en

Author ..... Festo

Last saved 09.03.2026

General information:

This document is intended for qualified, trained and instructed professionals. The data provided in this document are no guaranteed specifications, in particular with regard to functionality, condition or quality in the legal sense. The information in this document is intended only as simple indications for the implementation of a specific, hypothetical application, and in no way as a substitute for the operating instructions of the respective manufacturers or the design and testing of the respective application by the user. The respective operating instructions for Festo products can be found under www.festo.com. The user of this document must ensure that each function described herein works properly in his application. The user remains solely responsible for his or her own use of this document, even by studying this document and using the information mentioned therein. This also applies to any software (in source code and/or object code) that is made available to the user as an appendix to this document.

Rights of use of the software:

If software is made available to the user as an appendix to this document, the user is granted a simple and unlimited right of use hereto. This right also includes the processing and distribution of the software to third parties in edited or unedited form. The User is not permitted to use the name "Festo" to endorse or promote products derived from the Software without express prior written permission.

Software is provided to the user "as is". Festo does not assume any warranty or guarantee with regard to software. Festo's liability for damages of any kind arising from the use of the software is limited to intent and gross negligence.

Legal Notices:

Internet: http://www.festo.com
E-Mail: service_international@festo.com

Table of contents

1 Components/Software used

1.1 Software Used

Type/Name	Version Software/Firmware
Designer Studio	4.5.2 Build517
Microsoft SQL Server	2025
SQL Server Management Studio	22.1.0

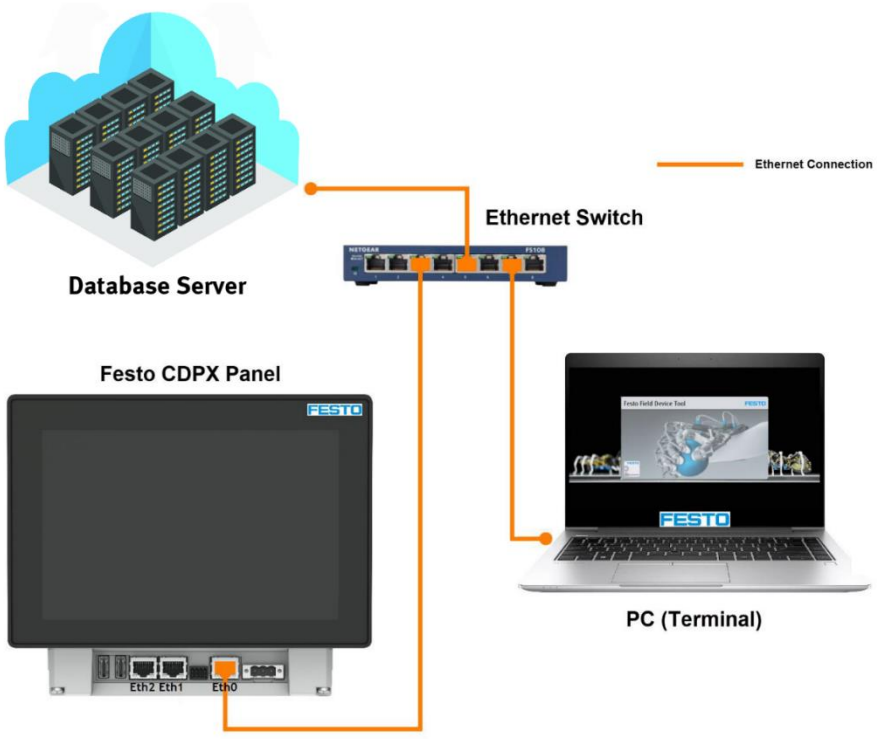
Table 1.1: Software used

1.2 Hardware Used

Type/Name	Part Number	Version Software/Firmware
Festo Operator Panel	CDPX-X-E1-W-7	Config OS:1.3.638 Main OS:1.3.638 Bootloader:1.0.25
PC	-	Windows 11 Enterprise 23H2 <X64> (Build 22631.6345)

Table 1.2: Hardware used

1.3 Topology Overview

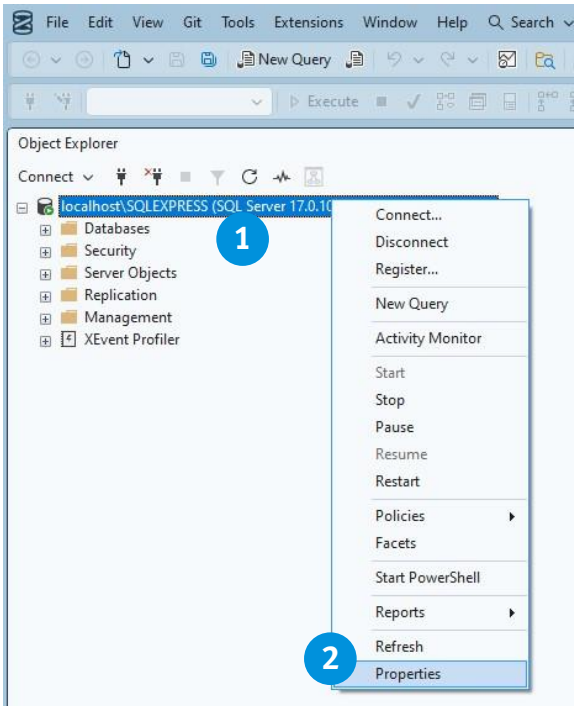


2 Configuration on Database Server

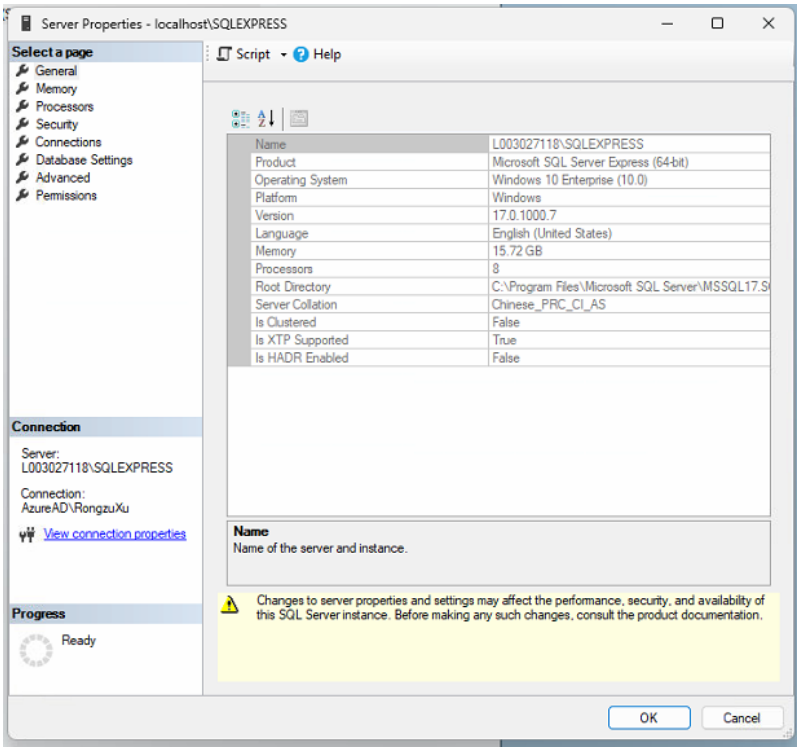
2.1 Active SQL Server Authentication mode and allow remote connection

By default, MS SQL only enables Windows Authentication. To support CDPX accessing MS SQL via the connection string, SQL Server Authentication mode must be enabled.

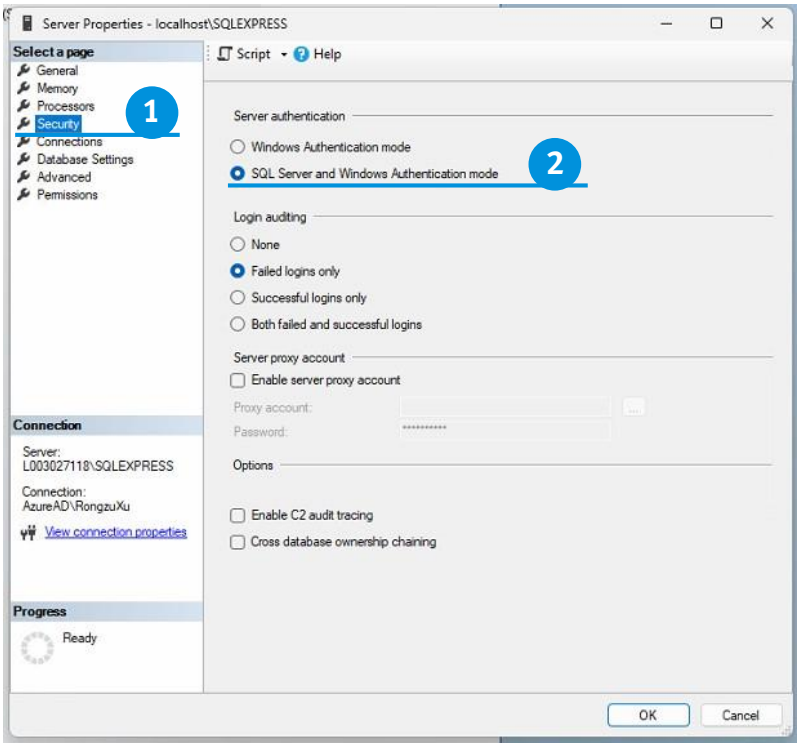
1. Right-click on the database.



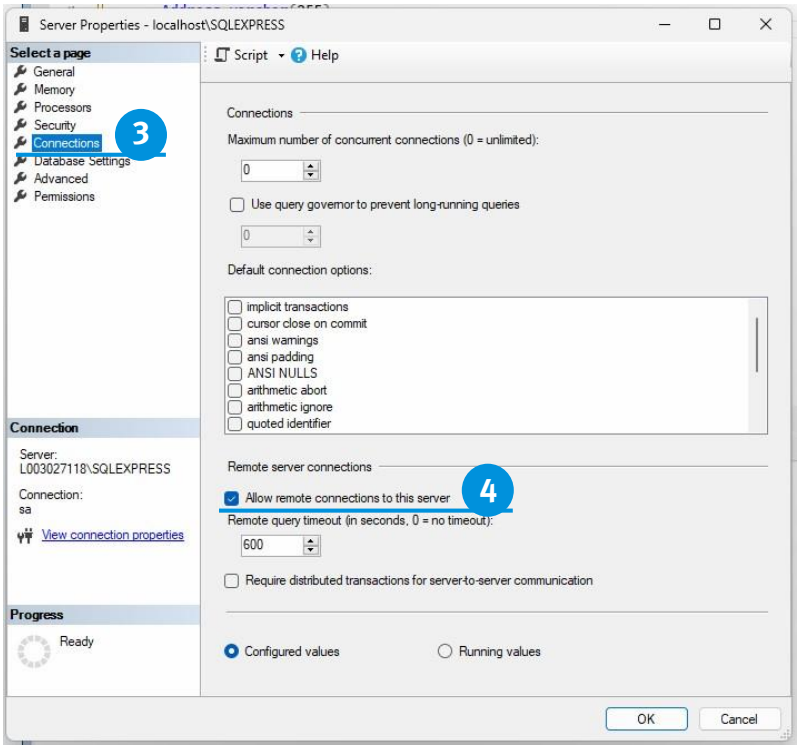
2. Open the Properties Configuration Page.



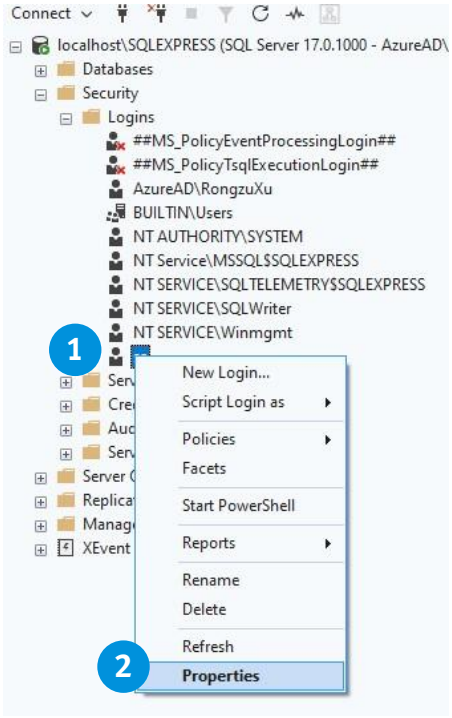
3. Choose **“SQL Server and Windows Authentication mode”** in the **Security** Tab.

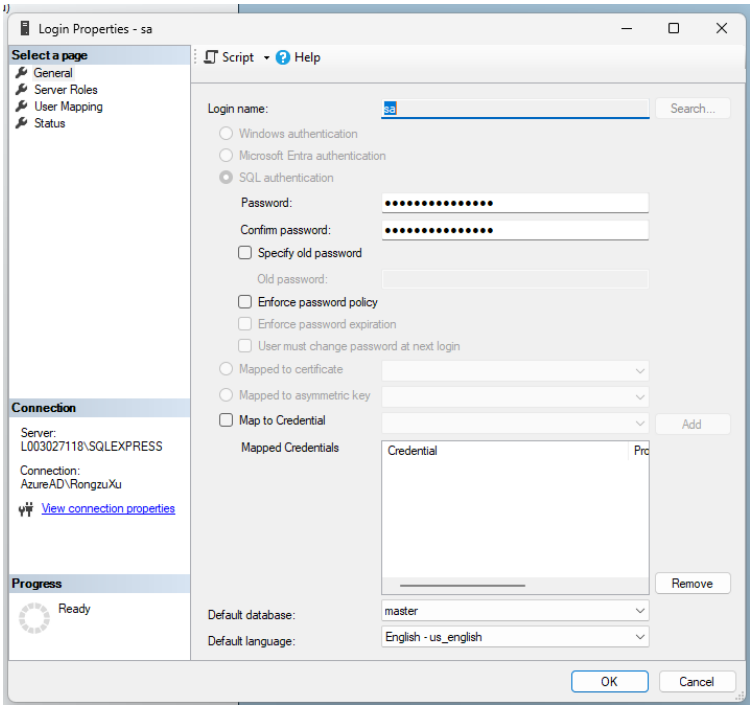


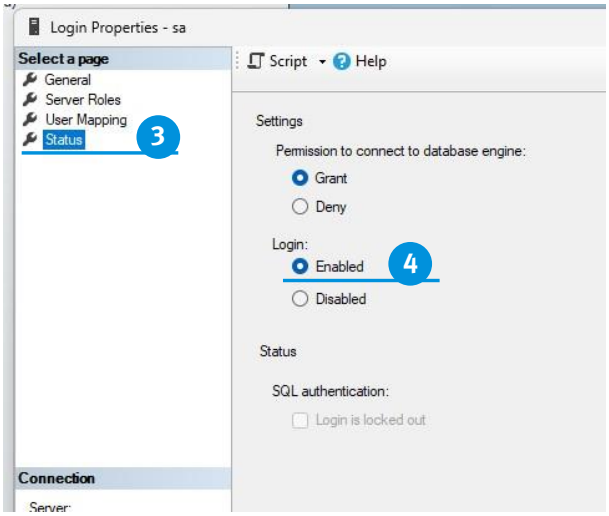
4. Choose **“Allow remote connection to the server”** in the **Connections** Tag



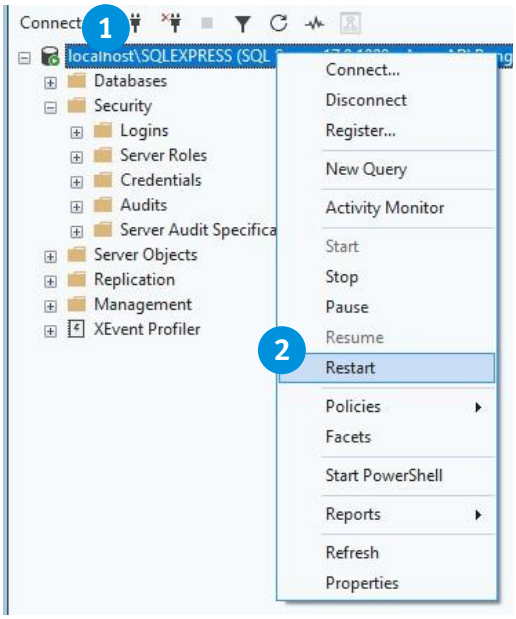
5. Enable “sa” account or Add one account.





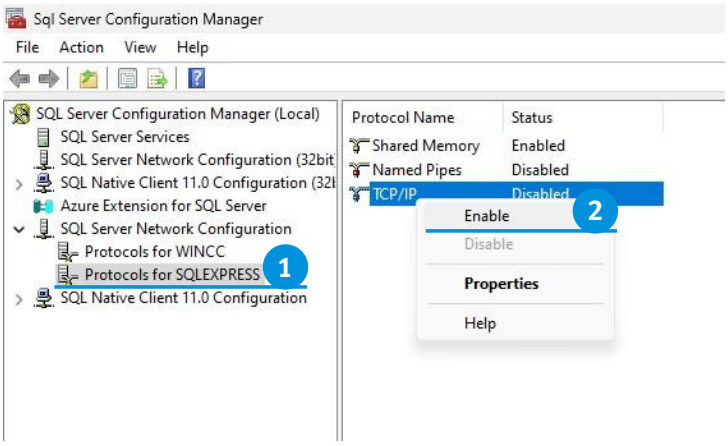


6. Restart the database

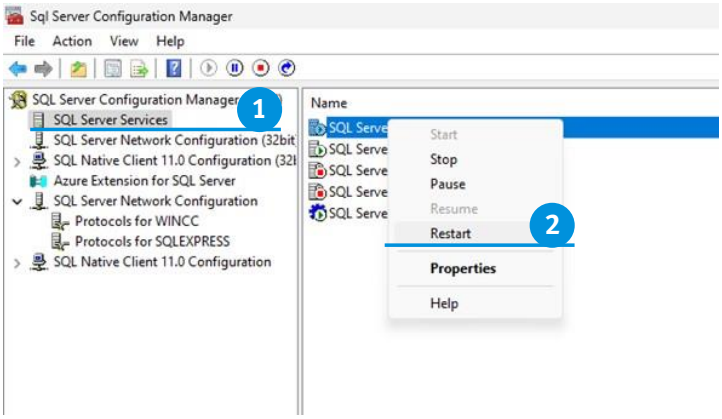


2.2 Active TCP/IP protocol for MSSQL

1. Open “Sql Server Configuration Manager”
2. Check “SQL Server Network Configuration”, choose the Protocols for your database, then right click on the “TCP/IP”, choose “Enable”.

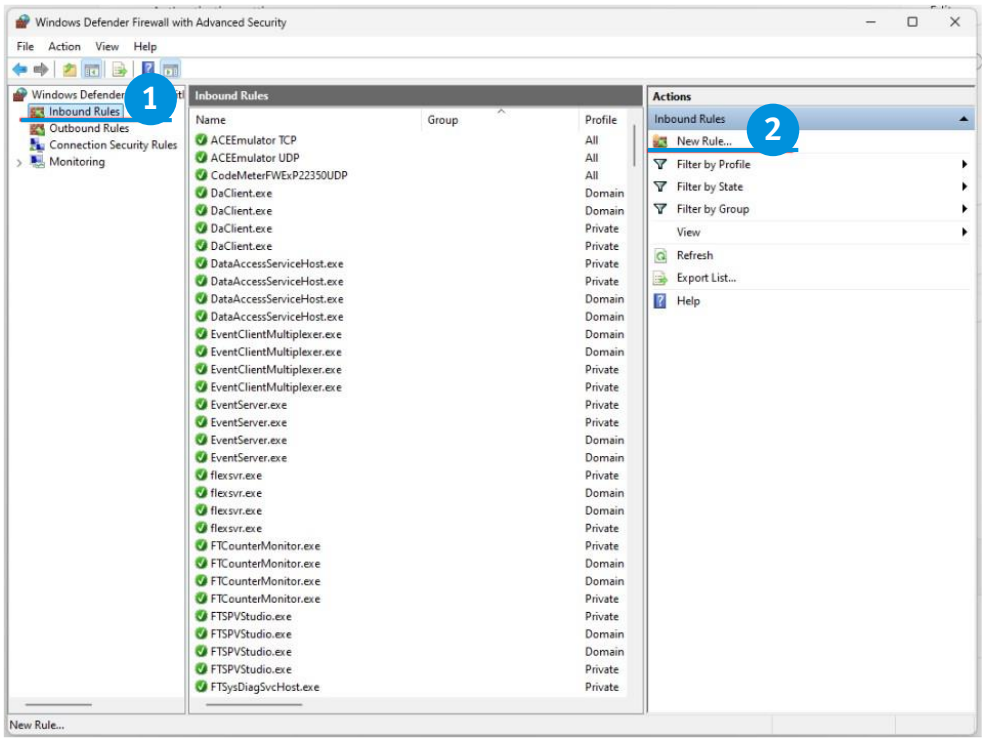


3. Restart the SQL Server in SQL Server Services

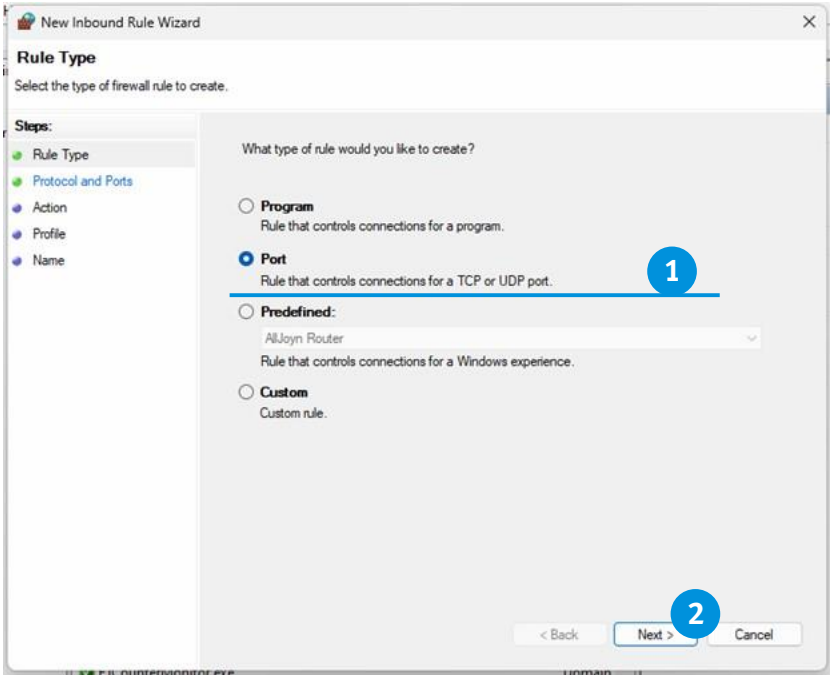


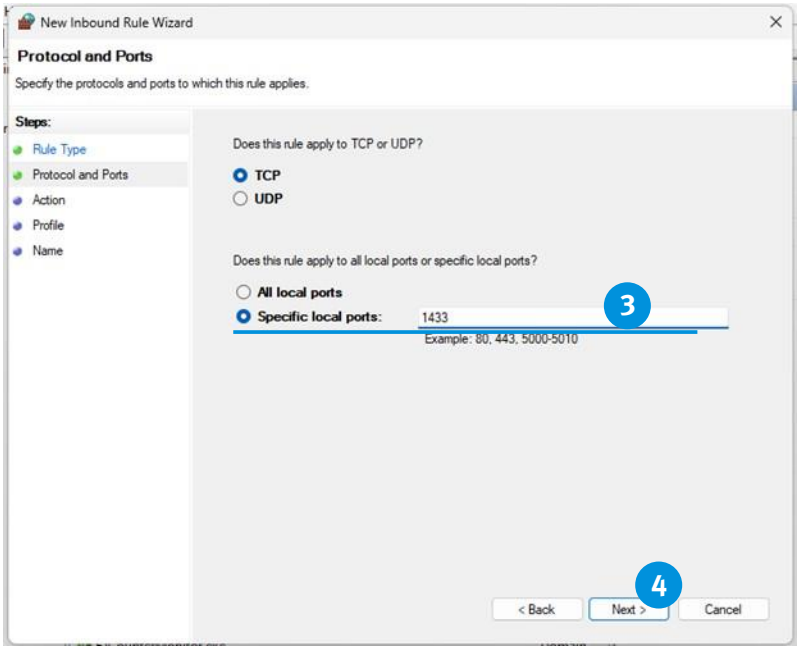
2.3 Allow Port 1433 support TCP/IP Protocol

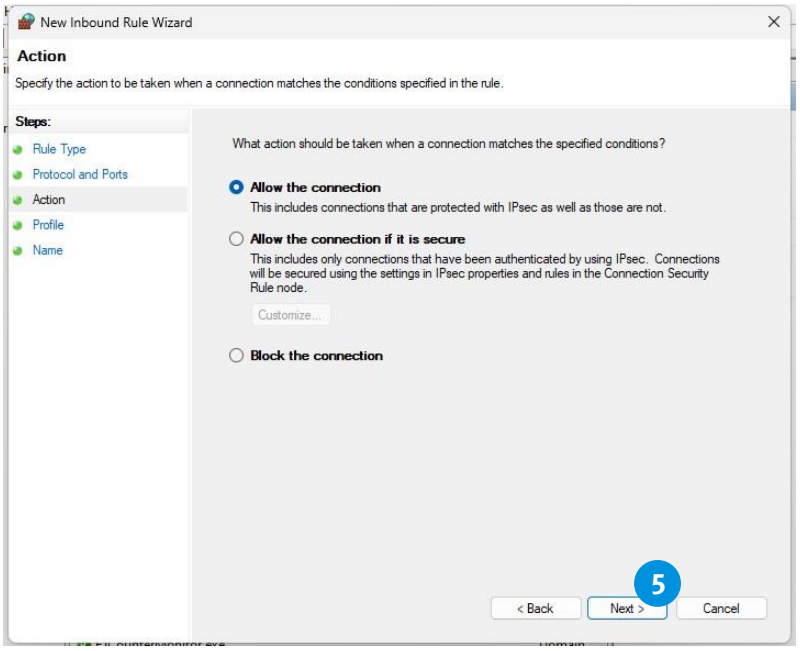
1. Open “Windows Defender Firewall with Advanced Security”
2. Choose “Inbound Rules” in the left list

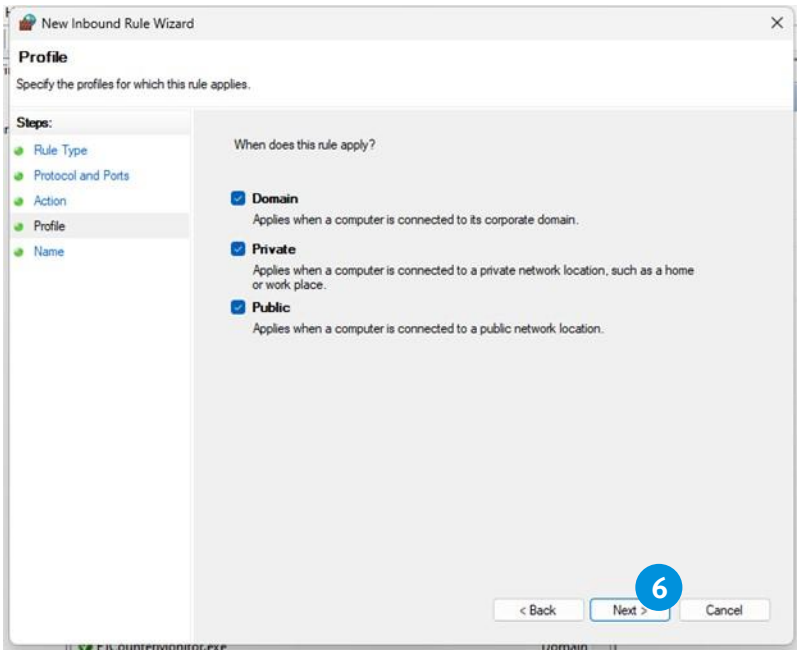


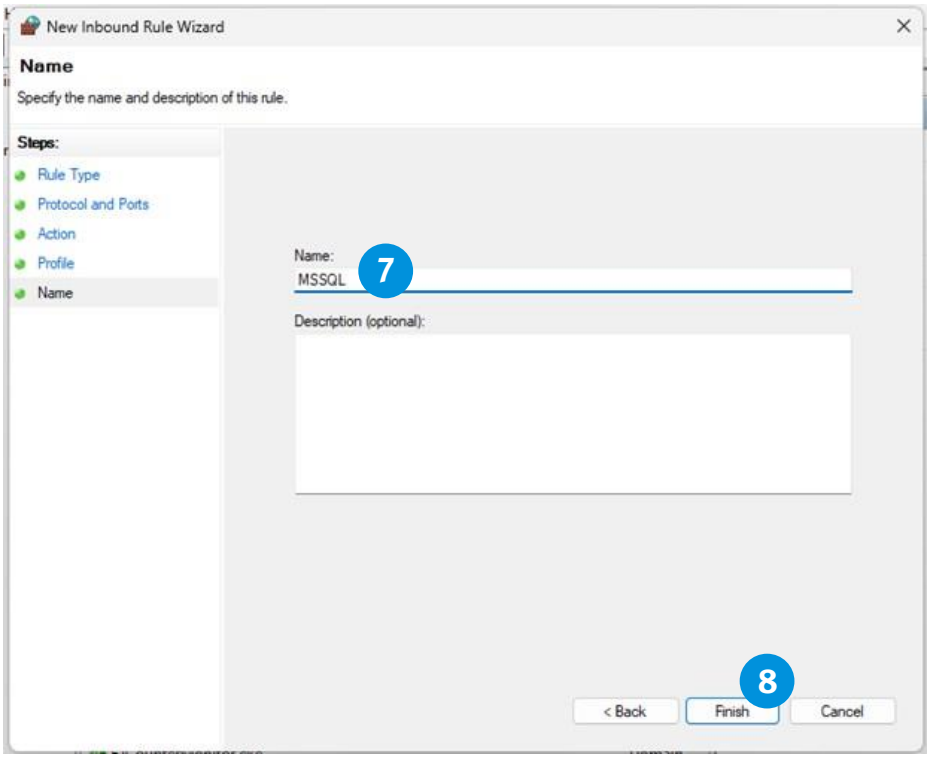
3. New Inbound Rule for Port 1433




New Inbound Rule Wizard
Protocol and Ports
Specify the protocols and ports to which this rule applies.
Steps:
Rule Type
Protocol and Ports
Action
Profile
Name
Does this rule apply to TCP or UDP?
TCP
UDP
Does this rule apply to all local ports or specific local ports?
All local ports
Specific local ports:
1433
Example: 80, 443, 5000-5010
3
4
< Back
Next >
Cancel


New Inbound Rule Wizard
Action
Specify the action to be taken when a connection matches the conditions specified in the rule.
Steps:
Rule Type
Protocol and Ports
Action
Profile
Name
What action should be taken when a connection matches the specified conditions?
Allow the connection
This includes connections that are protected with IPsec as well as those are not.
Allow the connection if it is secure
This includes only connections that have been authenticated by using IPsec. Connections will be secured using the settings in IPsec properties and rules in the Connection Security Rule node.
Customize...
Block the connection
5
< Back
Next >
Cancel


New Inbound Rule Wizard
Profile
Specify the profiles for which this rule applies.
Steps:
Rule Type
Protocol and Ports
Action
Profile
Name
When does this rule apply?
Domain
Applies when a computer is connected to its corporate domain.
Private
Applies when a computer is connected to a private network location, such as a home or work place.
Public
Applies when a computer is connected to a public network location.
6
< Back
Next >
Cancel


New Inbound Rule Wizard
Name
Specify the name and description of this rule.
Steps:
Rule Type
Protocol and Ports
Action
Profile
Name
Name:
MSSQL
7
Description (optional):
8
< Back
Finish
Cancel

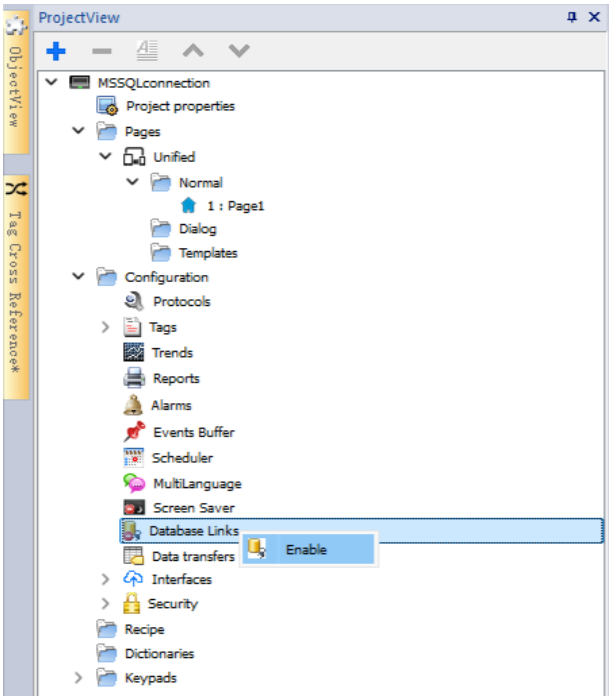
3 Designer Studio Configuration

3.1 Configuring the HMI project

3.1.1 Enable Database Links

Path: ***ProjectView*** > ***Configuration*** > right-click ***Database links*** > ***Enable***

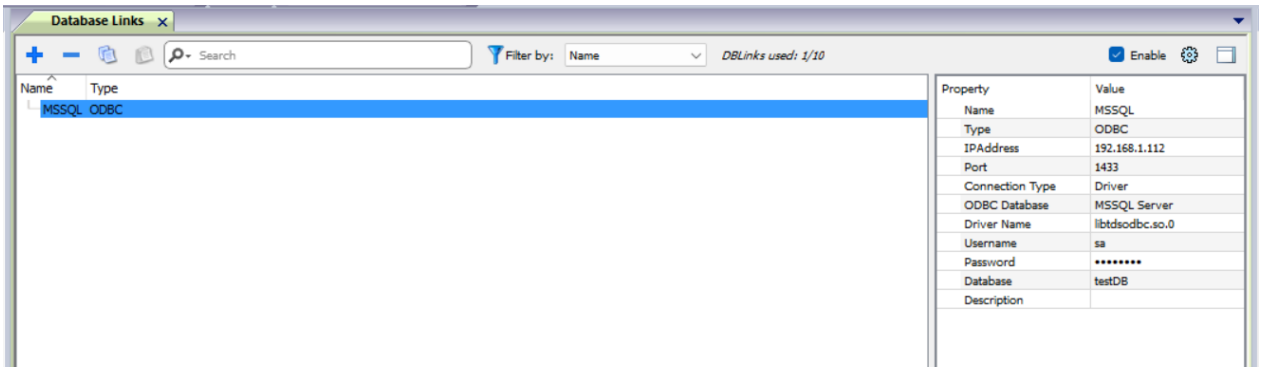
Note the icon change to indicates whether the function is enabled or disabled.



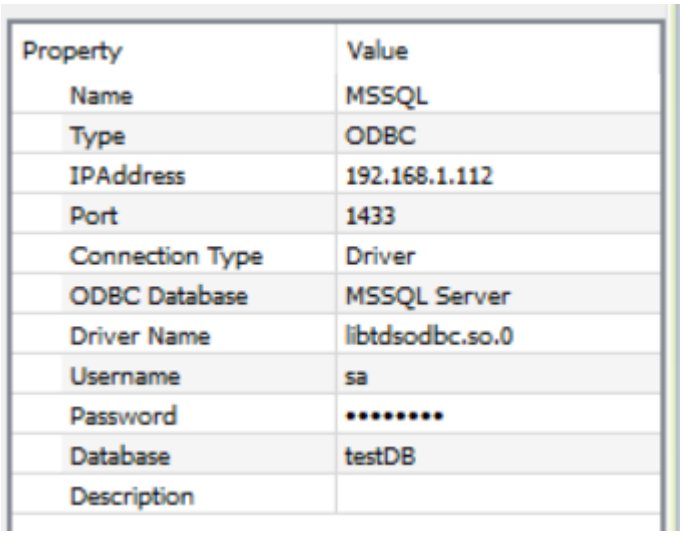
3.1.2 Configure a Database Links

Path: ***ProjectView*** > ***Configuration*** > double-click ***Database Links***

To use an external database it is necessary to create a connection with the specific database by specifying the database access parameters. Later you can use the macros and the JavaScript APIs by referencing the database by the name given to the link.



Recommended Configuration for Connecting to an MSSQL Server



Parameter	**Description**
Name	Define the name to use to reference the database defined in this item.
Type	Select: **OBDC**
IPAddress	Server IP Address
Port	Port used from MSSQL Server (generally **1433**)
Connection Type	Select: **Driver**
ODBC Database	Select: **MSSQL Server**
Driver Name	Type a driver name installed on the HMI device: "***libtdsodbc.so.0***"
Username	Username for accessing the MSSQL database
Password	Password for accessing the MSSQL database
Database	The name of the database you want to access
Description	This is just a reminder (write here what you prefer)

Table 3.1: Parameter descriptions

3.2 Using the database

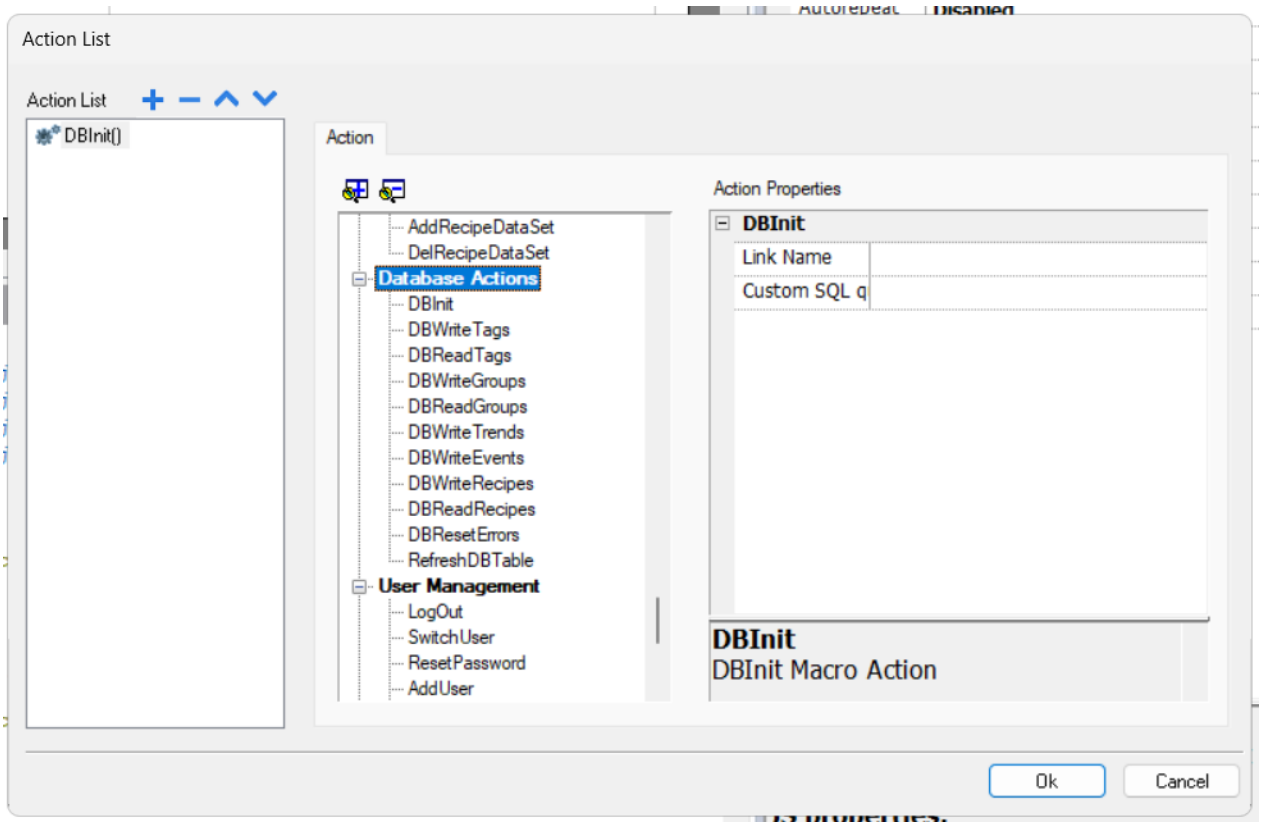
CDPX defined some **Database Actions** to exchange data with external SQL databases.

Database Actions are normally executed when events are triggered.

Events can be triggered by various widgets, for example on press and on release of a button. Not all actions are available for all the events of an object.

Actions are linked to widgets in the Event section of the Property panel (Page Editor).

The following screenshot shows all defined Database Actions.



3.2.1 Structure of the database tables

Based on common industrial database use cases, CDPX predefined several table structures:

Table Tags:

A tag is a friendly name used to identify the memory location of a device. Tags can be read or write from an external device through communication protocols.

Name	Data Type	PRIMARY KEY
FieldName	Text(255)	√
TagValue	Text(255)	

Table 3.2: Table tags

Table Trends:

Trends allow you to sample and record the values of specified tags according to specific sampling conditions. The trend function includes trend acquisition and trend display.

Name	Data Type	PRIMARY KEY
Id	Long Integer	√
TrendName	Text(255)	
SampleTime	Text(255)	
TrendValue	Text(255)	
Quality	Text(255)	
RefreshTime	Text(255)	

Table 3.3: Table trends

Table Recipes:

Recipes are collections of tag values organized in sets that satisfy specific application requirements.

For example, if you have to control room variables (temperature and humidity) in the morning, afternoon and evening. You will create three sets (morning, afternoon and evening) in which you will set the proper tag values.

Each element of the recipe is associated to a tag and can be indexed into sets for a more effective use. This feature allows you to extend the capabilities of controllers that have limited memory.

Name	Data Type	PRIMARY KEY
Recipe	Text(255)	√
SetName	Text(255)	√
ElementName	Text(255)	√
SetValue	Text(255)	

Table 3.4: Table Recipes

Table Events:

Events are used to trigger actions at project level and can be associated to:

- buttons / touch (click, press, release)
- mouse wheel
- external input devices like keyboards and mouse (click, press, hold, release, wheel)
- data changes (OnDataUpdate)
- switch of pages (OnActivate, OnDeactivate)
- alarms
- scheduler

Name	Data Type	PRIMARY KEY
Id	Long Integer	√
EventName	Text(255)	
SampledTime	Text(255)	
EventType	Text(255)	
EventSubTime	Text(255)	
EventValue	Text(255)	

Table 3.5: Table events

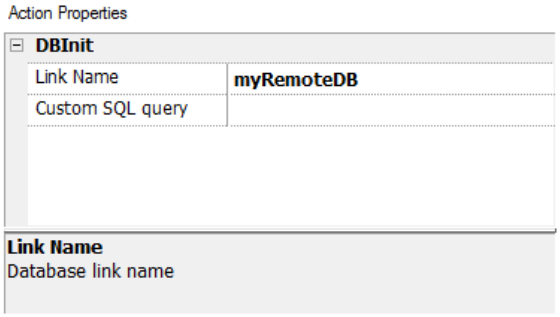
Information

Table structures are defined inside the project file **config\dbconnector.xml**

3.2.2 DBInit

Creates the set of tables required by the project. You do not need to use this action if the database already contains the necessary tables.

Leave “Custom SQL query” parameter empty to generate default table names.



Note

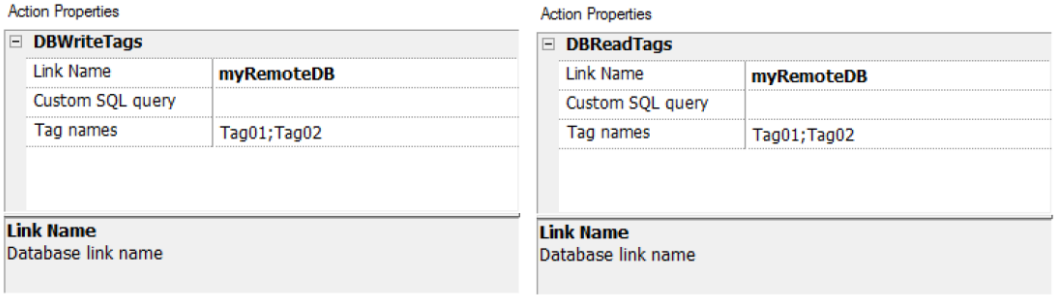
This action is used only once on an empty database.

It is not an initialization command to be called any time the HMI device starts.

3.2.3 DBWrite Tags, DBRead Tags

Transfer the values of the selected tags to/from the remote database.

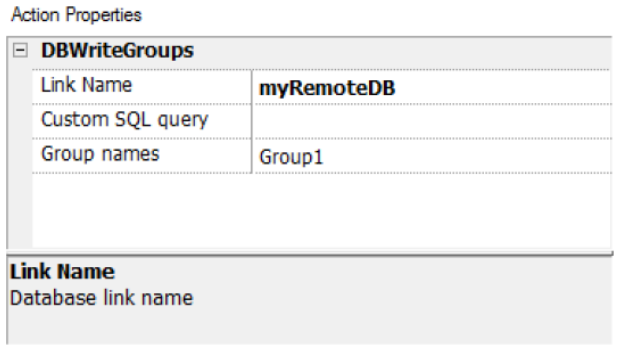
Leave “Custom SQL query” parameter empty to generate default table names.

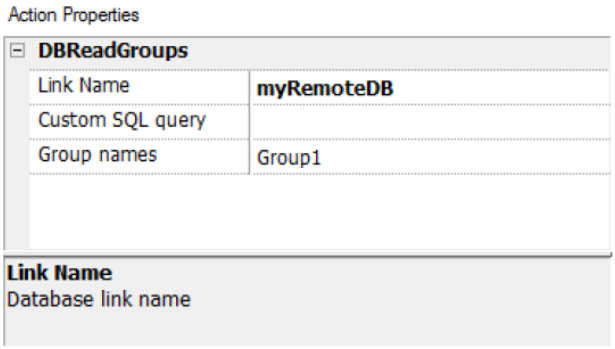


3.2.4 DBWrite Groups, DBRead Groups

Transfer groups of tags between the HMI device and the database.

Leave “Custom SQL query” parameter empty to generate default table names.

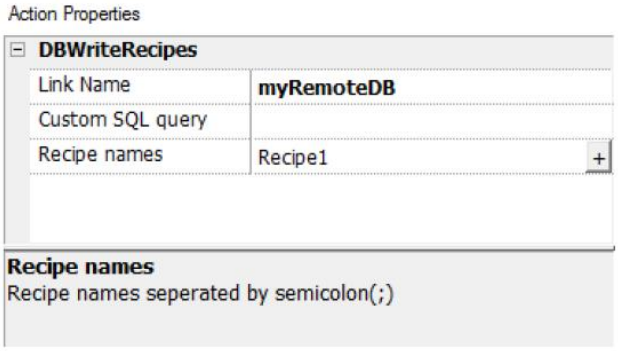


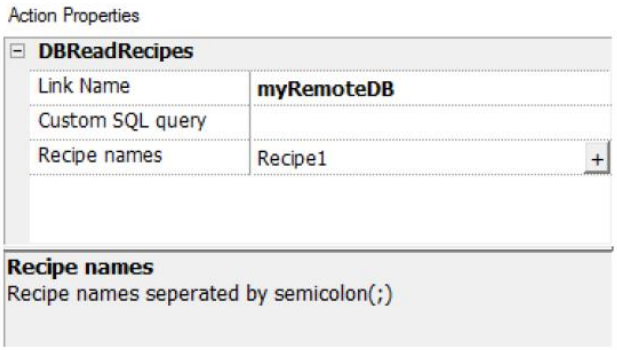


3.2.5 DBWrite Recipes, DBRead Recipes

Transfer the recipe data to/from the remote database.

Leave “Custom SQL query” parameter empty to generate default table names.

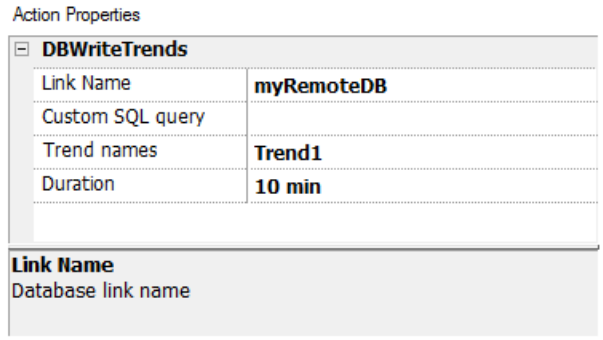




3.2.6 DBWrite Trends

Inserts the values of the last data sampled in the selected range of time inside the Trends table of the remote database.

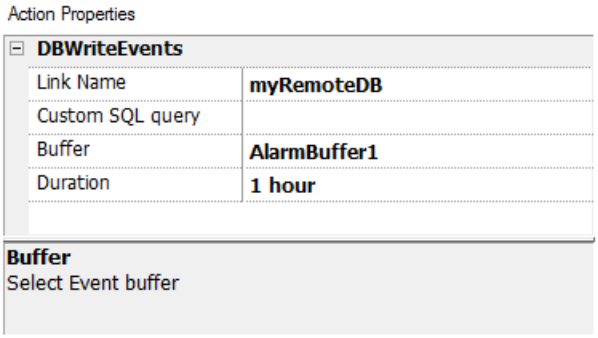
Leave “Custom SQL query” parameter empty to generate default table names.



3.2.7 DBWrite Events

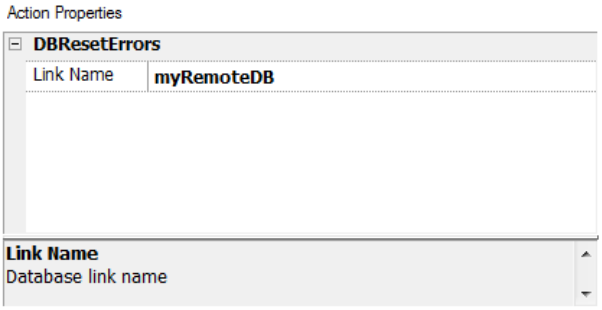
Inserts the values of the last events in the selected range of time inside the Events table of the remote database.

Leave “Custom SQL query” parameter empty to generate default table names.



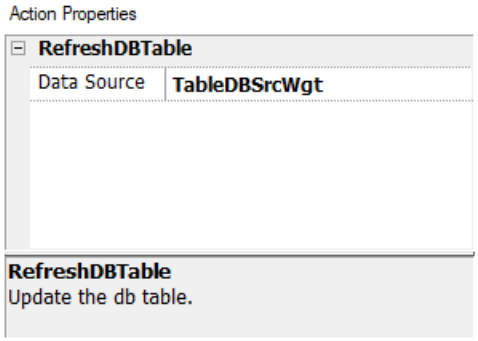
3.2.8 DBResetError

Reset all the three status variables of the selected database link.



3.2.9 RefreshDBTable

Executes the SQL query of the selected "DB table data source" widget to update its data.



3.2.10 Other JavaScript API

dbQuery

```
project.dbQuery(databaseLink, customSQL, dbCallback);
```

Using this query you can execute SQL Queries.

Parameter	Description
databaseLink	Name of the database to use.
customSQL	String with the SQL query
dbCallback	Function that will be call when query data are ready

Table 3.6: API descriptions

dbCallBack

```
project.dbCallBack(dbStatus, dbResponse);
```

Parameter	Description
dbStatus	0: no error found
dbResponse	Query response. Table column names followed by its rows: In the example: TagnName - Tagvalue Tag09 - 103 Tag10 - 302

Table 3.7: dbCallBack descriptions

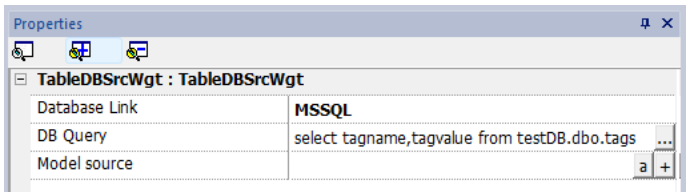
Script

```
1
2  function JS1_onMouseClick(me, eventInfo) {
3
4      var customSQL = "SELECT Tagname, Tagvalue FROM Tags WHERE Tagname='Tag09' OR Tagname='Tag10' ORDER BY Tagname"
5      var databaseLink ="Link1";
6      project.dbQuery(databaseLink, customSQL, dbCallback)
7  };
8
9
10 function dbCallback(dbStatus, dbResponse){
11
12     alert("SQL Answer = " + dbResponse + "\ndbStatus =" + dbStatus);
13 };
14
15
```

3.2.11 Configure a table with a "DB table data source"

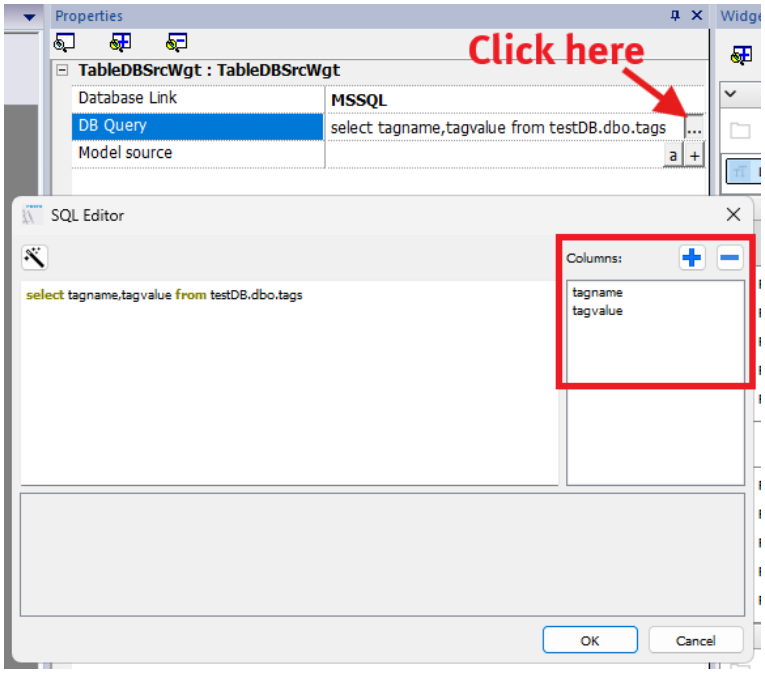
Put a DB table data source on the page

Path: ***Widget Gallery*** > ***Basic*** > ***Table*** > ***DB table data source***



Configure DB Query

1. Open the SQL Editor.
2. Type in the SQL query into left area.
3. Add the Columns with the column name in your database.



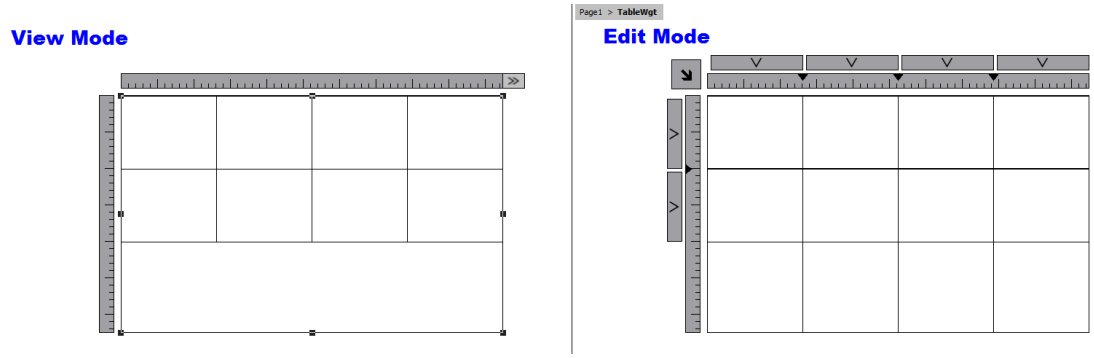
Put a **Table** widget on the page

Path: ***Widget Gallery*** > ***Basic*** > ***Table***

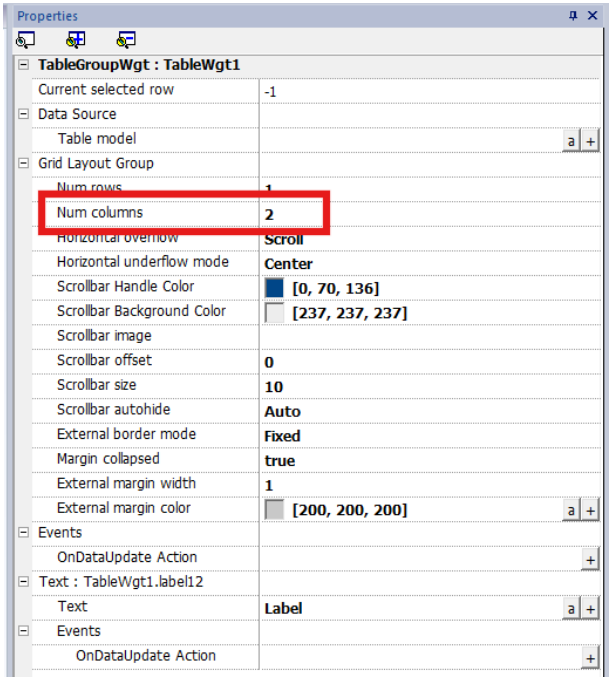
Configure the Table widget

Click on the table to manage the widget in view mode, double click to enter in the edit mode.

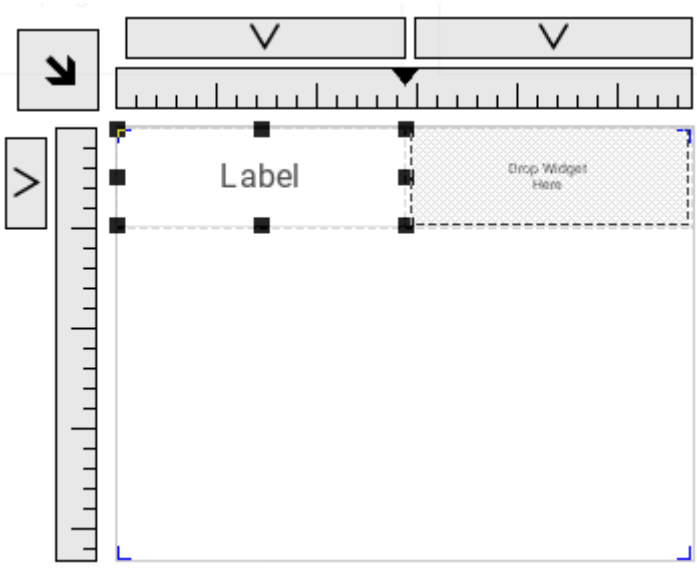
To exit and return to view mode click outside the table.



1. Enter the required number of columns.



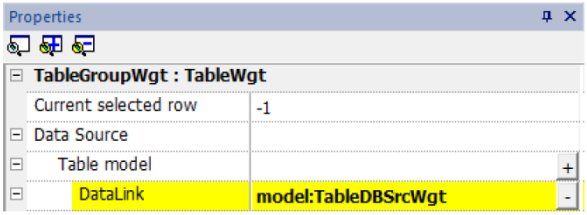
2. Double click on the table widget into edit view, drop label widget into each cell.

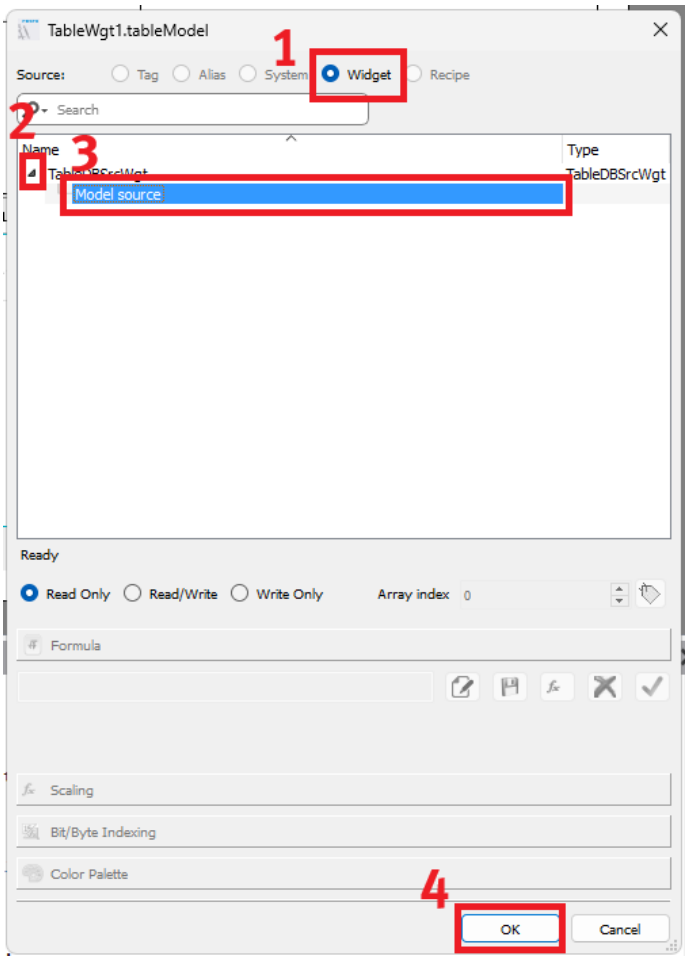


Bind Table widget with DB table data source.

1. Select the defined data source.

Table group





2. Select the defined column to each label.