

**SIEMENS**

*Ingenuity for life*

*Industry Online Support*

Home

## Exporting Archived Data from WinCC with the OLE DB Provider

SIMATIC WinCC + WinCC/Connectivity Pack

<https://support.industry.siemens.com/cs/ww/en/view/38132261>

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: <https://www.siemens.com/industrialsecurity>.

# Table of contents

	<b>Legal information</b> .....	<b>2</b>
<b>1</b>	<b>Introduction</b> .....	<b>4</b>
	1.1 Overview .....	4
	1.2 Principle of operation .....	5
	1.3 Description of the WinCC application .....	6
	1.3.1 Overview and description of the reverse osmosis system .....	6
	1.3.2 Functionality of the reverse osmosis system .....	9
	1.4 Components used .....	13
<b>2</b>	<b>Archive data exporting with WinCC</b> .....	<b>14</b>
	2.1 Principle of operation .....	14
	2.2 Configuration and project planning .....	15
	2.2.1 Configurations in the example project .....	15
	2.2.2 Project engineering of scripts for the export of archive data .....	16
	2.2.3 Script for exporting archived variables .....	16
	2.2.4 Script for exporting archived messages .....	27
	2.3 Operation .....	39
	2.3.1 VBS cript for exporting archived variables .....	39
	2.3.2 VBS Script for exporting archived messages .....	39
<b>3</b>	<b>Archive data exporting with Excel</b> .....	<b>40</b>
	3.1 Principle of operation .....	40
	3.2 The "Osmosis" client .....	42
	3.2.1 Principle of operation details .....	42
	3.2.2 Configuration and project planning .....	49
	3.2.3 Operation .....	53
	3.2.4 Error handling .....	55
	3.3 The "Universal" client .....	56
	3.3.1 Principle of operation details .....	56
	3.3.2 Configuration and project planning .....	67
	3.3.3 Operation .....	71
	3.3.4 Error handling .....	78
<b>4</b>	<b>Export archive data with SQL Server</b> .....	<b>79</b>
	4.1 Configuration .....	79
<b>5</b>	<b>Useful information</b> .....	<b>88</b>
	5.1 Basics .....	88
	5.1.1 SIMATIC WinCC/Connectivity Pack .....	88
	5.1.2 WinCC OLE DB Provider .....	88
	5.1.3 Prerequisite for the connection to the WinCC database .....	88
	5.1.4 Using string tags .....	89
	5.2 Alternative solutions .....	90
<b>6</b>	<b>Appendix</b> .....	<b>91</b>
	6.1 Service and support .....	91
	6.2 Links and literature .....	93
	6.3 Change documentation .....	93

# 1 Introduction

## 1.1 Overview

### Motivation

In order to visualize processes of modern automation systems, numerous process variables are required. During the process the user is informed about events by means of messages. The past process variables and messages are stored in archives and are therefore available to you at any time.

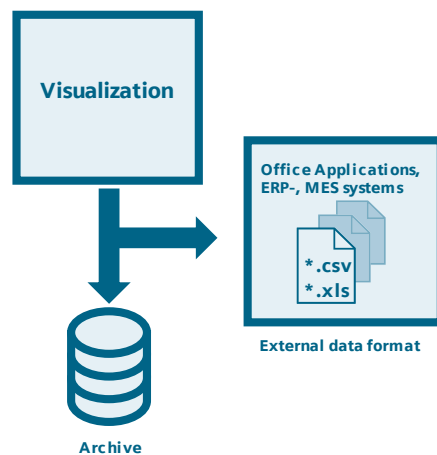
WinCC stores the data compressed and binary coded in the archives. The archived process variables and messages can be displayed via WinCC controls.

However, the underlying format is not suitable for analyzing and processing the data.

### The task

The requirement is to export the data in other formats, e.g. as a "\*.csv" or "\*.xls" file. The data can then be conveniently further processed in Office applications, ERP and MES systems.

Figure 1-1



### Required knowledge

Basic knowledge of WinCC V7 is required.

### Note

Basics are taught in the SITRAIN course "SIMATIC WinCC, System Course."

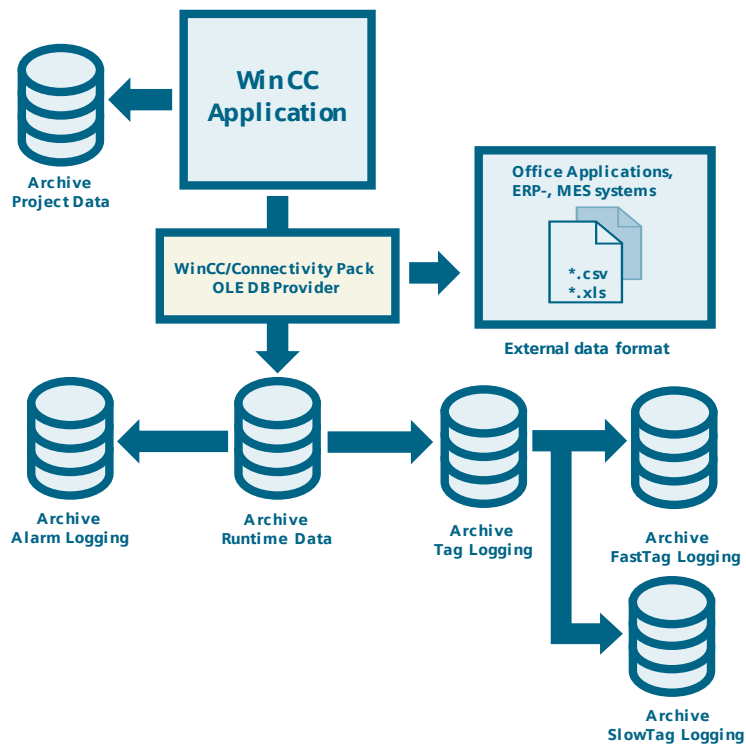
- [SIMATIC WinCC, Systemkurs \(de\)](#)
- [SIMATIC WinCC, System Course \(en\)](#)

## 1.2 Principle of operation

The option SIMATIC WinCC/Connectivity Pack is used to export the WinCC archive data. It provides the WinCC OLE DB Provider.

The WinCC OLE DB Provider enables transparent access to the WinCC archive data.

Figure 1-2



This document describes possible solutions for exporting archived data using Visual Basic (VB) scripts and the SQL Manager. The WinCC OLE DB provider is used here.

The following data is exported:

- Logged process values
- Archived messages

Three variants are considered for the creation of the new data formats:

- Export of archive data via a VB script in the WinCC project
- Export of archive data via Visual Basic for Applications (VBA) with Excel
- Export of archive data via MS SQL Server Management Studio

## 1.3 Description of the WinCC application

In this application example, a reverse osmosis plant for the treatment of process water is simulated for data acquisition, as it is used, for example, in the pharmaceutical industry or in breweries.

**Note** The reverse osmosis system is designed as a WinCC project and is included as a download in the application example.

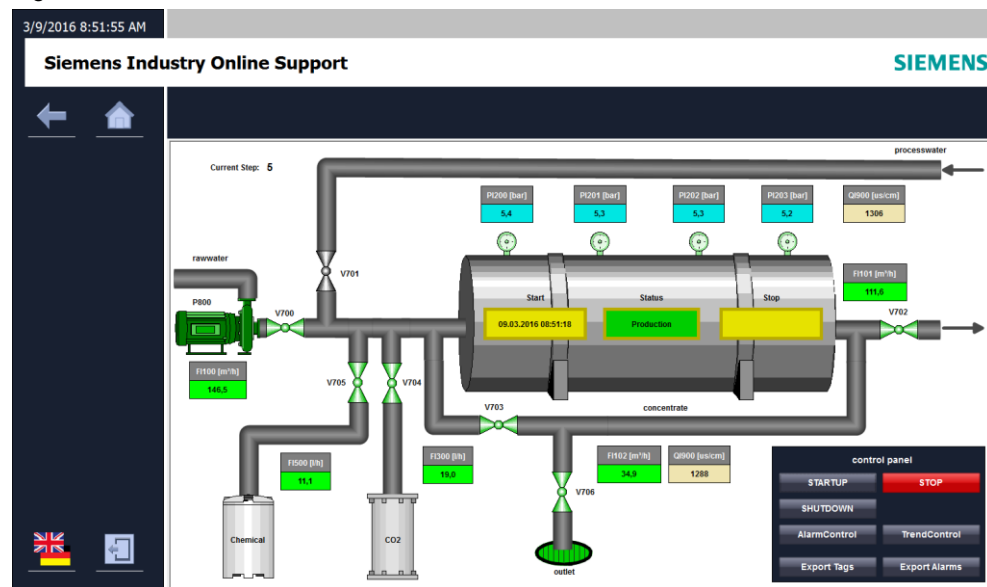
The system has the operating states:

- Approach
- Production
- Retract
- Off

VB scripts completely simulate the operating states. WinCC archives store the process variables and messages.

### 1.3.1 Overview and description of the reverse osmosis system

Figure 1-3



The application example has several buttons that control the system and export archive data.

Visually represented:

- Valves
- Various pressure displays
- A drain and motor

Depending on the operating state, the associated HMI variables of the motor and valves are set to "ON" (1) or "OFF" (0). In the active state, the color of the objects is changed. [Table 1-1](#) briefly describes how the buttons work.

Table 1-1





Buttons	Description
	Use this button to activate the previous image.
	With this button you activate the start screen of the application example.
	Use this button to switch between German and English.
	Use this button to end the WinCC runtime of the application example.
APPROACH	Sets the system to the operating state "Approach".
RETRACT	Sets the system to the operating state "Approach".
STOP	Sets the system to the operating state "Off".
Alarm Control	Opens the ActiveX control "Alarm Control". This shows the past messages on the status of the plant.
Trend control	Opens the ActiveX control "Trend Control". This shows the temporal course of the simulated, analog measuring points of the system.
Export Variable	Starts the VBS script for exporting measured values to a "*.csv" file.
Export messages	Starts the VBS script to export the messages to a "*.csv" file.

Table 1-2

Valves (V) / Pump (P)	Description
V700	Fresh water valve
V701	Processed water valve
V702	Tank outflow valve
V703	Concentrate valve
V704	Valve CO <sub>2</sub>
V705	Chemical valve
V706	Outflow valve
P800	Motor/Pump for fresh water

The current states of the simulated process variables are displayed via analog measuring points (I/O fields). Process variables described ([Table 1-3](#)):

- Flows in the pipelines
- Conductivity of liquids
- The pressure conditions at different measuring points within the tank

Table 1-3

Analog Process tags	unit	Description
FI100	m <sup>3</sup> /h	Fresh water inflow
FI101	m <sup>3</sup> /h	Outflow tank
FI102	m <sup>3</sup> /h	Discharge outside the tank
FI300	l/h	Inflow CO <sub>2</sub>
FI500	l/h	Inflow chemical
QI901	µS/cm	Conductivity concentrate
QI900	µS/cm	Conductivity processed water
PI200	bar	Pressure sensor1 tank
PI201	bar	Pressure sensor2 tank
PI202	bar	Pressure sensor3 tank
PI203	bar	Pressure sensor4 tank

### Core functionality process

Table 1-4

No.	Operating mode	Note
1.	Off	After starting the application example in WinCC runtime, the system is in the "Off" operating state. The valves and motors are in the basic position (state: „0“). Analog measuring points have either their start value or the value "0".  The "STOP" button is used to simulate an emergency intervention in the running system. The system is switched off and switches to the operating state "Off".
2.	Approach	After the "start-up" process, the system automatically switches to the "production" process. The valves and the engine open and start step by step. All analog measuring points are described with simulated values.
3.	Production	All valves and motor are activated. The analog measuring points are also described with simulated, slightly fluctuating values.
4.	Retract	After "shutting down" the system, the operating state "Off" is restored.  The valves and the motor close or stop gradually. Still active measuring points are described with simulated values.



### 1.3.2 Functionality of the reverse osmosis system

After starting the runtime, the actions "Measurement.bac" and "SimReverseOsmosis.bac" are called event-triggered, in which various procedures and functions are executed. They control the process flow and the simulation of the analog measured variables. The scripts were created with VBS. The complete plant process is simulated in twelve steps. Five steps for the start-up process, seven steps for the shutdown process, where the 0 step reflects the status "Off" and the 5th step "Production".

#### Actions

Table 1-5

Action	Call Trigger	Description
Measurement.bac	cyclic Timer = 1 s	Controls the calculation of values of the individual analog measuring points by calling the scripts "MeasMode" and "Measurement". The required simulation parameters are transferred within the action.
SimReverseOsmosis.bac	cyclic Timer = 1 s	Simulates the reverse osmosis process by calling the scripts "StepControl" and "ActuatorControl".

#### Procedures

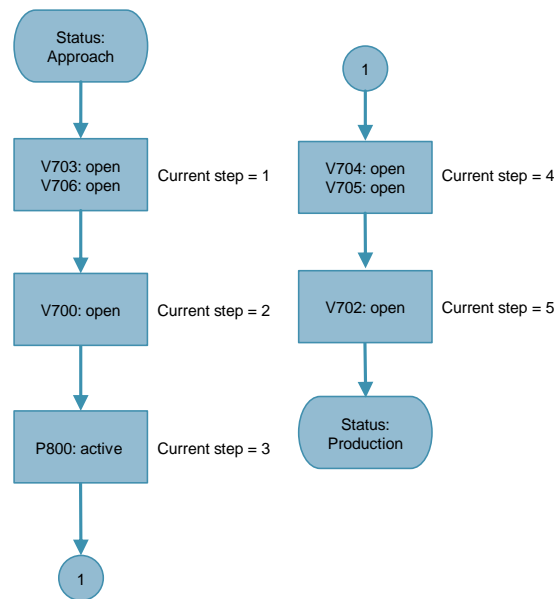
Table 1-6

Procedure	Call	Description
StepControl.bmo	Action „SimReverseOsmosis“	Simulates the individual steps of the plant using various sequences (StartupSequence, ShutdownSequence, StopSequence).
MeasMode.bmo	Action „Measurement“	Simulates the status of the individual analog measuring points.
Measurement.bmo	Action „Measurement“	Simulates the sizes of the individual analog measuring points.
ActuatorControl.bmo	Action “SimReverseOsmosis“	Simulates the states of valves and motors depending on the state of the system.
WriteArchiveToCSV.bmo	Trigger Button „WriteArchiveToCSV“	Creates a "*.csv" file, accesses the archive via the WinCC OLE DB Provider and writes the archive entries for the variable "QI901_ConductivityPermeatIn" into it. Only variable names, measurement time, variable ID and value are saved.
WriteMessagesToCSV.bmo	Trigger Button „WriteMessagesToCSV“	Creates a "*.csv" file, accesses the archive via the WinCC OLE DB Provider and writes the archive entries where the message number is <4 to it. Only the measurement time, message number, message type, and message class are saved.

**Button „Start up“**

- The start time is determined and entered in the "Start" output field.
- The "StartupSequence" sequence is started.
- The "Status" operating status display changes to the "Start-up" operating mode.
- The valves and motors open and start step by step. All analog measuring points are described with simulated values. The "Start-up" process contains the following five steps:

Figure 1-4



All analog measuring points are described with simulated values. The values are freely selectable and can be adjusted at any time in the script.

**Variables**

Table 1-7

Variables	Start value	Min	Max	Increment values
FI100_FlowRawwater	0	60	150	9
FI101_FlowConcentrate	0	10	7	120
FI102_SupplyWater	= FI100 – FI101 + (FI300/1000) + (FI500/1000)	-	-	-
FI300_FlowCO2	0	1	19	2
FI500_FlowChemicals	0	1	11	1.5
QI900_ConductivityPermeatIn	1511	1310	1556	33
QI901_ConductivityConcentrate	1900	1 300	2000	56
PI200_Tank	0	1.1	5.5	0.35
PI201_Tank	0	1.1	5.4	0.32
PI202_Tank	0	1.1	5.3	0.31
PI203_Tank	0	1.1	5.2	0.30

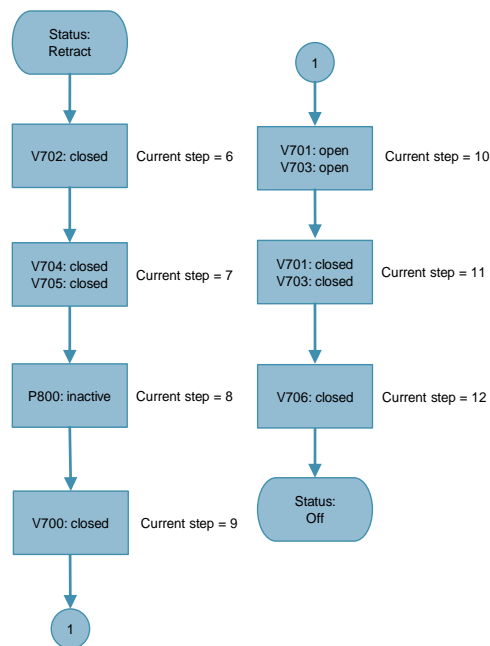
- The curve recording starts event-triggered with the start of the system.
- The start-up process takes approx. 25 s until the system switches to the "Production" operating state.

**Button "Shutdown"**

The "Status" operating status display changes to the "shutdown" operating mode.

- The sequence "ShutdownSequence" is started.
- The valves and motors close and stop step by step. Still active analog measuring points are still described with simulated values. The "shutdown" process contains the following seven steps (steps 6-12).

Figure 1-5



- The shutdown process takes approx. 25 s until the system changes to the "Off" operating state.
- The curve recording stops event-triggered with "Off" of the system.
- The stop time is determined and entered in the "Off" output field.

**Stop button**

- The "Status" operating status display changes to the "Off" operating mode.
- The "StopSequence" sequence is started.
- The valves and motors close or stop immediately, analog measuring points are reset to their start value.
- The curve recording stops event-triggered with the "Stop" of the system.
- The stop time is determined and entered in the "Off" output field.
- The system switches to the operating state "Off".

**Button „Production“**

All valves and the motor are active (state: "1"). The analog measuring points are also described with simulated and slightly fluctuating values. (see [Table 1-7](#)).

**Button „off“**

The valves and motors are in the basic position (state: "0"). Analog measuring points have either their start value (see [Table 1-7](#)) or the value "0".

## 1.4 Components used

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

Table 1-8

Components	Quantity	Article number	Note
Engineering station	1		Development computer with Windows 7x64 SP1
SIMATIC WinCC V7.4 SP1	1	6AV63.1-....7-4...	
SIMATIC WinCC/Connectivity Pack V7.4 SP1	1	6AV6371-1DR07-4...	
MS SQL Server 2008 R2	1		
MS Excel 2007, 2010, 2013	1		

This application example consists of the following components:

Table 1-9

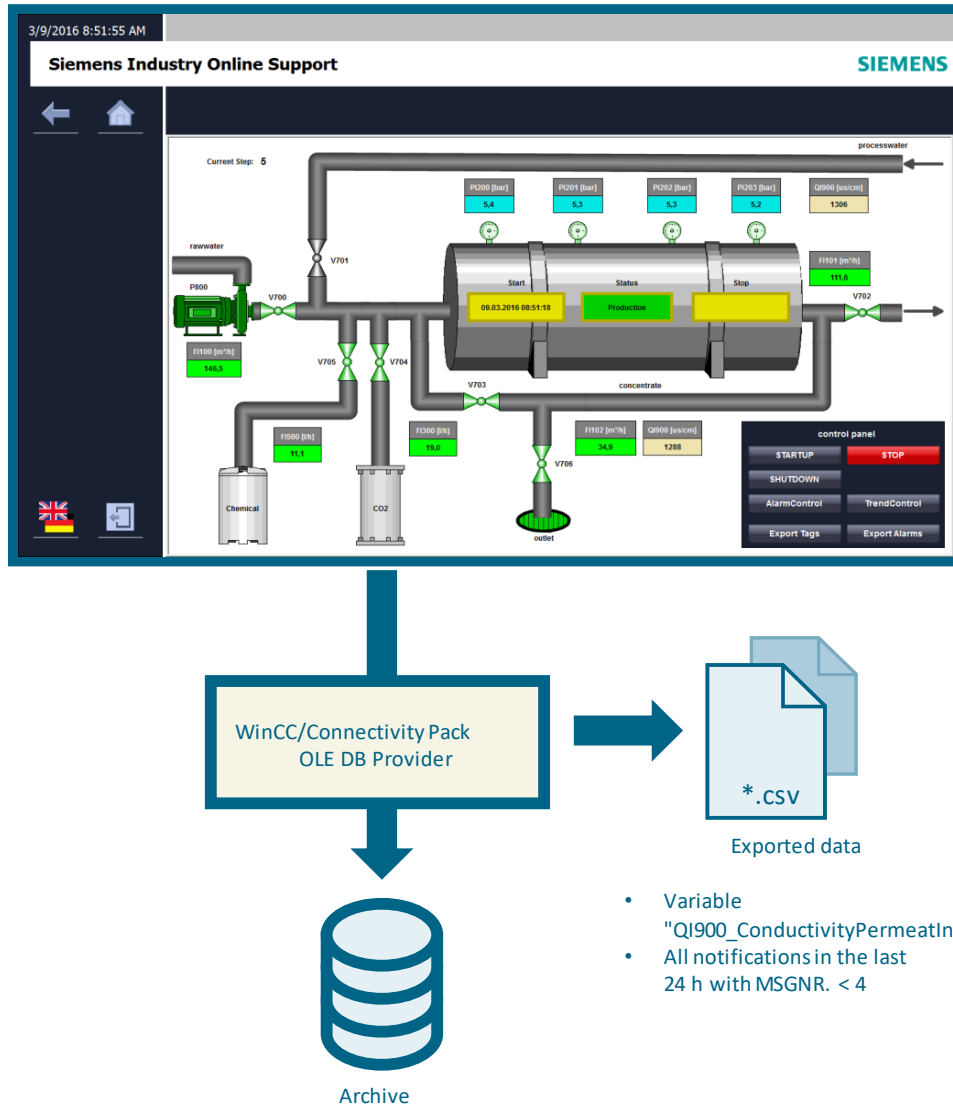
Components	File name
WinCC project	38132261_Application_Reverse_Osmosis_PROJ.zip
Excel files	38132261_Application_Reverse_Osmosis.xls 38132261_Application_Universal_Client.xls
Documentation	38132261_Application_Reverse_Osmosis_DOC_de.pdf

## 2 Archive data exporting with WinCC

### 2.1 Principle of operation

- Data access to the runtime archive is via VBS scripts in WinCC.
- The archive data is exported to CSV files.

Figure 2-1



## 2.2 Configuration and project planning

This chapter describes which configurations you have to perform in your own project to export archive data to a csv file using the WinCC OLE DB provider.

**Note** The scripts shown in the section 2 are for illustration purposes only and have already been created in the WinCC project. The WinCC project is therefore immediately executable after the computer name has been adapted.

### 2.2.1 Configurations in the example project

#### Adapting the computer name

1. Unpack the file with the name „38132261\_Application\_Reverse\_Osmosis\_PROJ.zip“ in any WinCC project directory on your computer.
2. Open the WinCC project with the name „WinCC73\_SE\_UPD8\_Connectivity\_Pack.MCP“ and adjust the computer name in the computer properties in the project. To do this, use the context menu under "Computer > Properties" to adopt the Windows computer name of your PC.
3. Start WinCC.

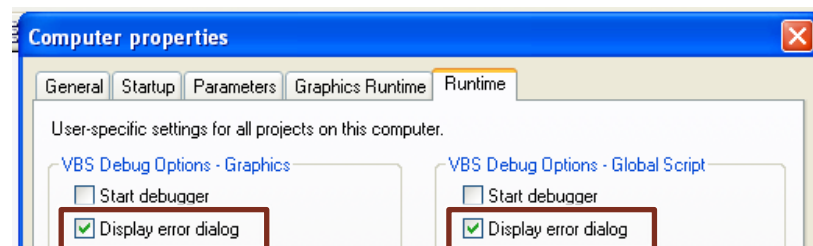
**Note** The set resolution is 1520 x 900 pixels.  
The project is bilingual (switchable German/English) and therefore has a language switch.

#### Script debugger

If you want to debug the VB scripts, you can use a script debugger. As of SIMATIC WinCC V6.2, it is no longer included in the scope of delivery of SIMATIC WinCC for licensing reasons.

To enable the display and debugging of script errors, set the following settings on the Runtime tab of the Machine Properties:

Figure 2-2



**Note** You can install the MS Script Debugger via your WinCC installation disk. Further information on this can be found in the online help for WinCC in the chapter "Testing with the Debugger".

### Start Debugger

If you activate the "Start Debugger" function, the debugger is started when the first script is started in the runtime. This function is used for quick troubleshooting during the project planning phase.

You configure the "Start Debugger" function separately for VBS scripts in Graphics Designer images and for VBS scripts in Global Script. Activate the respective option box "Start Debugger".

#### Note

Further information can be found in the documentation "WinCC V7.4 Scripting: VBS, ANSI-C, VBA", in the chapter "How to activate the debugger".

<https://support.industry.siemens.com/cs/ww/en/view/109736230/70687471243>

### Display error dialog

If the function "Show error dialog" is activated, an error dialog with information about the occurred error is displayed when an error occurs in a VB script. A debugger can be started via a button in the error dialog. Prerequisite is an installed debugger for Visual Basic.

The function "Show error dialog" can be configured separately for VB scripts in Graphics Designer images and for VB scripts in Global Script. For this reason, there are separate check boxes.

### 2.2.2 Project engineering of scripts for the export of archive data

After completion of a production cycle of the reverse osmosis system, the measured values archived between the start and stop times are entered in a "\*.csv" file.

The stop, start-up and shut-down times of the osmosis plant are entered in the message archive as operating messages. They have the registration numbers 1, 2 and 3. In this application example, the start-up, shutdown and stop messages of the past 24 hours are to be written to a "\*.csv" file.

The solution of this task is shown by means of a script. It was created with the Global Script Editor of WinCC. Event triggered (by keystroke or variable trigger) it can be called up.

### 2.2.3 Script for exporting archived variables

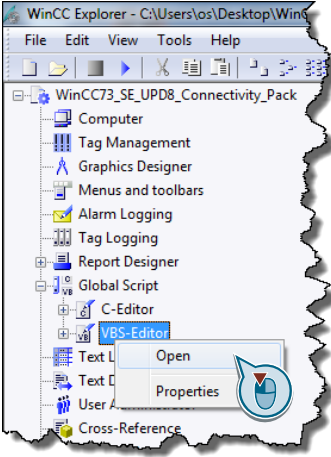
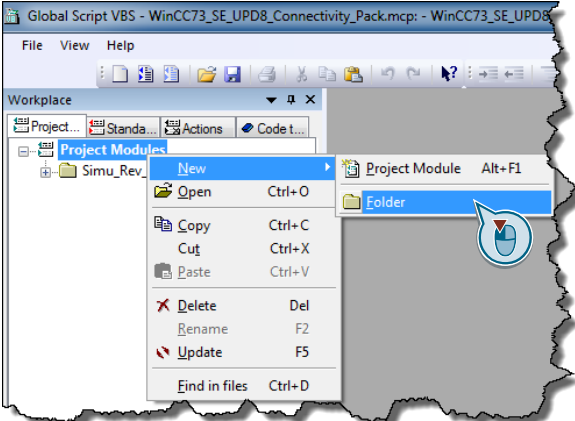
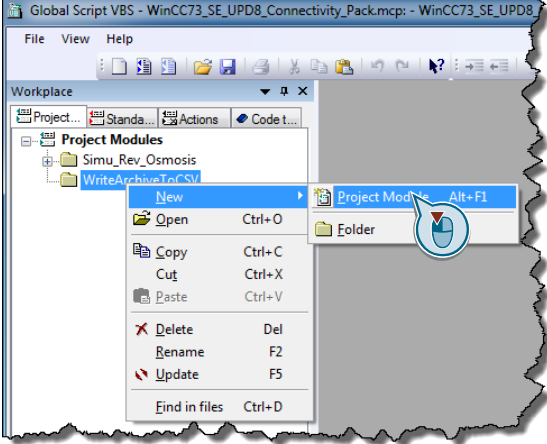
For better illustration, only the measuring point QI900 (conductivity of the generated process water) is considered in this example. Their archived values are written to a "\*.csv" file at the press of a key.

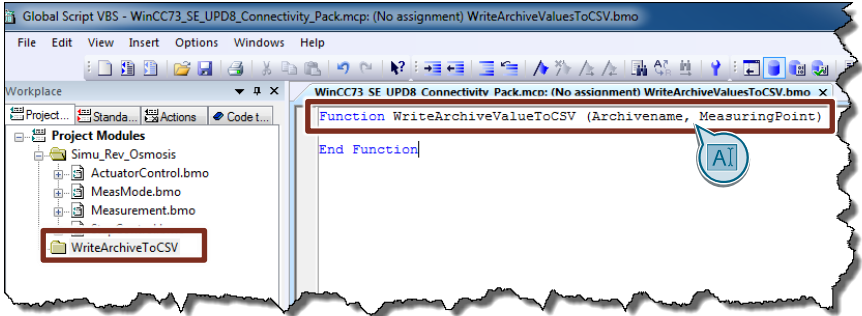
Alternatively, it is also possible to query several measured values simultaneously in VBS code.



Creation of the VBS module

Table 2-1

No.	Action
1.	<p>Start the "Global Script VBS editor" editor.</p> 
2.	<p>Select the Project Module tab and create a new folder named WriteArchiveToCSV.</p> 
3.	<p>Create a new project module in this folder.</p> 

No.	Action
4.	<p>Change the sub-procedure to a function. Add the parameters "ArchiveName" and "MeasuringPoint" to them. Save the project module under the name "WriteArchiveValuesToCSV".</p>  <p>Archive name (parameter Archive name) and measuring point name (parameter MeasuringPoint) are not transferred until the function is called.</p>

**VBS-Code: Creating the "\*.csv" file**

In the first step of the VBS code, the "\*.csv" file is first created in the folder "C:\Temp\". The file name should consist of the plant name "Osmosis", the start time of the osmosis plant and the measuring point name of the measuring point to be archived.

(E.G.: C:\Temp\Osmosis\_15.02.2015 10:30:00 QI900.csv)

Table 2-2

No.	Action
1.	<p>Start the code by declaring the following variables:</p> <pre data-bbox="472 1216 1375 1518"> 'Declaration of local Tags Dim fso                'FileSystemObject Dim f                  'File Dim ts                 'TextStream Dim path               'Path Dim StartArchive      'Starttime of Archiving Dim StopArchive       'Stoptime of Archiving Dim TimeStamp         'Timestamp for bulding filename                     </pre>
2.	<p>Use the following code to read out the start and stop times of the reverse osmosis system. The internal WinCC variables for this are "DateTimeLastStart" and "DateTimeLastStop". When the system is started and stopped, the respective times (format: dd.mm.yyyy hh:mm:ss) are stored in the system.</p> <pre data-bbox="472 1648 1375 1749"> 'Read Start- and Stoptime of Osmosis Set StartArchive = HMIRuntime.Tags("DateTimeLastStart") Set StopArchive = HMIRuntime.Tags("DateTimeLast")                     </pre> <p>If the start or stop time is missing, a message is to be output and the processing of the function is to be terminated.</p> <pre data-bbox="472 1827 1375 1962"> If StartArchive.Read = "" Or StopArchive.Read = "" Then     MsgBox "Start- or Stoptime of Archiving is missing"     Exit Function End If                     </pre>

No.	Action
3.	<p>Specify the path and file name. The variable "path" contains the complete file name with path in the form:                      C:\Temp\Osmosis_+ start time +_measuring point name                      The start time is used in the application example to create the file name and must first be converted to a suitable format.</p> <pre data-bbox="470 427 1359 656">                     'Generate String for the *.csv filename and replace ":" with                     "_"                     TimeStamp = FormatDateTime(StartArchive.Read, vbGeneralDate)                     TimeStamp = Replace(TimeStamp,":", "_")                     'Path and name for the *.csv -File                     path = "C:\Temp\" &amp; "Osmosis_" &amp; TimeStamp &amp; "_" &amp;                     MeasuringPoint &amp; ".csv"                     </pre>
4.	<p>You then use the following code to create the file system object (fso) and the file object (f) using the path and file name (path) previously created. Before this, the system checks whether the file already exists and, in this case, terminates the processing of the function after a message has been output.</p> <pre data-bbox="470 797 1359 1088">                     'Create FileSystemObject and *.csv file if it not exist:                     Set fso = CreateObject("Scripting.FileSystemObject")                     If Not fso.FileExists(path) Then                         fso.CreateTextFile(path)                     Else                         MsgBox "File already exist:" &amp; vbCrLf &amp; path                         Exit Function                     End If                     </pre> <p>Finally, the "*.csv" file with the TextStream object (ts) is opened for writing.</p> <pre data-bbox="470 1137 1359 1352">                     'Create File-Object and open this File                     Set f = fso.GetFile(path)                     'iomode = 2, Writing                     'format = -2, TristateUseDefault                     Set ts = f.OpenAsTextStream(2,-2)                     '*.csv file is now ready for Writing                     </pre> <p>The script is set up. The file is not created until the script has been executed.</p>

© Siemens AG 2019. All rights reserved

### Connection to the Database

In the second step of the function, the connection to the database is established using the WinCC OLE DB Provider. The measured value archive data are entered in a RecordSet (a copy of several data records). The following fields are transferred per record:

Table 2-3

Field No.	Field name	Type	Meaning
0	ValueID	Integer (4 Byte)	ID of the process value
1	TimeStamp	DateTime	Time stamp of the process value
2	RealValue	Real (8 Byte)	Process value
3	Quality	Integer (4 Byte)	Quality code of the process value

**Establishment of connection**

To establish a connection to the database, a connection object is created and the connection string is transferred. The ConnectionString requires:

- Provider names
- The provider instance number
- The User Data Source Name (DSN, GUID of the WinCC project)
- As data source WinCC incl. the computer name

General format of the "ConnectionString":

ConnectionString = "Provider=<Provider name>.<Instance no.>;Catalog=<DSN of the database>R;Data Source=<Computer name>WinCC".

**Note**

Access is also possible without specifying the computer name (.WinCC). However, the access speed decreases.  
For transparent access for "Catalog", enter the name of the WinCC project, e.g.: „Catalog = WinCC\_Project\_Name“.

**Requesting of data**

The data is queried via a command object to which a "CommandText" is then passed for processing. In the "CommandText" must be passed:

- The action (R = read access)
- The ValueID or the symbolic name of the archive variable.
- The start and stop time ("Start time" and "End time")

Format of the „CommandText“:

CommandText = "Tag:<Action>,< ValueID or ArchiveName\Tagname>', '<Starttime>','<Stoptime>".

**Note**

To query multiple archived values simultaneously, specify the ValueIDs or variable names in parentheses and separated by semicolons (e.g. „TAG:R,('ValueName\_1';'ValueName\_2';'ValueName\_x'), <TimeBegin>, <TimeEnd>“).

The start and stop times can be absolute or relative.

Table 2-4

Start time	Stop Time	Requested time range
2016-02-15 14:10:00:000	2016-02-15 14:30:00:000	15.02.2016 14:10 to 14:30
2016-02-15 14:10:00:000	0000-00-00 01:00:00:000	15.02.2016 14:10 to 15:10
0000-00-00 01:00:00:000	2016-02-15 14:30:00:000	15.02.2016 13:30 to 14:30
0000-00-00 00:00:00:000	0000-00-00 00:01:00:000	Archive start to archive end plus 1 min.
0000-00-00 00:01:00:000	0000-00-00 00:00:00:000	Archive end minus 1 min until archive end

**Note** The time stamp of the process values is stored in UTC (coordinated world time). Therefore, the regional time zone and, if applicable, summer and winter time must be taken into account when specifying the start and end times in absolute form.  
 You can find information on converting the local computer time to UTC under <https://support.industry.siemens.com/cs/ww/en/view/24201113>

**Note** The query with "ValueID" is more performant than the query via „ArchiveName\TagName“. However, the ValueID is not identical to the sequence of the variables in the tag logging editor. The ValueID of a tag can be determined via the SQL Server Management Studio. Open the dbo.Archive table in the project's runtime database using SQL Server Management Studio. The ValueID is listed in the first column of the table.

**Establishing the connection and querying the data**

Table 2-5

No.	Action
1.	Declare the other variables: <pre> Dim Pro           'Provider Dim DSN           'Data Source Name Dim DS            'Data Source Dim ConnString   'Connection String Dim MachineNameRT 'Name of the PC from WinCC-RT Dim DSNRT        'Data Source Name from WinCC-RT Dim Conn         'Connection to ADODB Dim RecSet       'RecordSet Dim Command      'Query Dim CommandText  'Command-Text Dim CommandTextStart 'Starttime for SQL-String Dim Duration     'Duration of Production-Cycle Dim DurationSec  'Duration of Production-Cycle Dim DurationMin  'Duration of Production-Cycle Dim DurationHour 'Duration of Production-Cycle Dim DurationDay  'Duration of Production-Cycle Dim Language     'Language tag                     </pre>

No.	Action
2.	<p>Use the following code to read the host name and GUID of the project from the @LocalMachineName and @DataSourceNameRT environment variables into the previously declared local variables.</p> <pre data-bbox="488 360 1369 495"> 'Read the name of the PC-Station and the DSN-Name from 'WinCC-RT Set MachineNameRT = HMIRuntime.Tags("@LocalMachineName") Set DSNRT = HMIRuntime.Tags("@DataSourceNameRT")                     </pre> <p>You use these variables to form the "ConnectionString".</p> <pre data-bbox="488 539 1369 869"> 'Preparing the Connection-String 'First instance of WinCCOLEDB Pro="Provider=WinCCOLEDBProvider.1;" 'Name of Runtime-Database DSN="Catalog=" &amp; DSNRT.Read &amp; ";" 'Data Source DS= "Data Source=" &amp; MachineNameRT.Read &amp; "\WinCC" 'Build the complete String: ConnString = Pro + DSN + DS                     </pre>
3.	<p>Connect to the database using the ConnectionString application:</p> <pre data-bbox="488 925 1369 1104"> 'Make Connection Set Conn = CreateObject("ADODB.Connection") Conn.ConnectionString = ConnString Conn.CursorLocation = 3 Conn.open                     </pre> <p><b>Note:</b> To specify the location of the RecordSet for retrieving archive data, assign a value to the "CursorLocation" property. In the example, the value "3" is assigned. The RecordSet is thus created on the client.</p>
4.	<p>To query the data in the WinCC database, the "Command Text" is required. In this example, the relative time is used for the query. The start time is the difference between the start time and the end time. The query thus covers the period "archive end minus time difference to archive end". (see <a href="#">Table 2-4</a>).</p> <p>The full time difference is first calculated in seconds and then converted to the time format "Days:Hours:Minutes:Seconds". During formatting, the leading zeros are also generated for numerical values greater than 10.</p> <pre data-bbox="488 1451 1369 1966"> 'Duration between Start and Stop in seconds: Duration = DateDiff ("s", StartArchive.Read, StopArchive.Read) 'Split the Duration in days, hours, minutes and seconds: DurationMin = Fix(Duration/60) DurationSec = Duration - (DurationMin * 60) DurationHour = Fix(DurationMin/60) DurationMin = DurationMin - (DurationHour * 60) DurationDay = Fix(DurationHour/ 24) DurationHour = DurationHour - (DurationDay * 24) 'Creating leading zeros: DurationSec = Right("00" &amp; DurationSec,2) DurationMin = Right("00" &amp; DurationMin,2) DurationHour = Right("00" &amp; DurationHour,2) DurationDay = Right("00" &amp; DurationDay,2)                     </pre>

No.	Action
5.	<p>Form the "CommandText" from the archive name, the archive variable that you want to read, the start time and the calculated time difference.</p> <pre data-bbox="486 331 1361 555"> 'Formatting Starttime: CommandTextStart = "'0000-00-" &amp; DurationDay &amp; " " &amp; DurationHour &amp; ":" &amp; DurationMin &amp; ":" &amp; DurationSec &amp; ".000'" 'Building the complete String: CommandText="Tag:R,'" &amp; Archivename &amp; "\" &amp; MeasuringPoint &amp; "', " &amp; CommandTextStart &amp; ", '0000-00-00 00:00:00.000'"                     </pre>
6.	<p>Create the command object and define the "CommandType". Then the query with the previously created CommandText follows. The RecordSet is then set to the first data record in which the first, and therefore oldest, process value is stored.</p> <pre data-bbox="486 696 1361 976"> 'Create the recordset, read the records and set to first 'redcordset: Set Command = CreateObject("ADODB.Command") Command.CommandType = 1 Set Command.ActiveConnection = Conn Command.CommandText=CommandText Set RecSet = Command.Execute RecSet.MoveFirst                     </pre>

**Note**

When using the time and date format in WinCC and Visual Basic, note that the date format depends on the region set in Windows. The DDPS code is designed in such a way that date and time are evaluated independently of these computer-specific settings.

### Writing the "\*.csv" file

The records from the previously opened RecordSet are written to the "\*.csv" file.

Table 2-6

No.	Action
1.	<p>The following loop writes the data records (measuring point name, variable ID, time stamp (UTC), process value) line by line into the "*.csv" file. A row with the column headings is created first. The current runtime language setting is taken into account (HMIRuntime.Language; 1031 = German, 1033 = English). The RecordSet is placed on the next record with each run.</p> <pre> write recordsets To *.csv-File Header in *.csv-File Language = HMIRuntime.Language Select Case Language Case 1031      'German = 1031 ts.WriteLine ("Tag-Name;ValueID;Datum/Zeit;Prozesswert") 'MsgBox "Tag-Name;ValueID;Datum/Zeit;Prozesswert" Case 1033      'English = 1033 ts.WriteLine ("Tag-Name;ValueID;Date/Time;Process-Value") 'MsgBox "Tag-Name;ValueID;Date/Time;Process-Value" End Select  Do While Not RecSet.EOF ts.WriteLine (MeasuringPoint &amp; ";" &amp; RecSet.Fields("ValueID").Value &amp; ";" &amp; RecSet.Fields("TimeStamp").Value &amp; ";" &amp; RecSet.Fields("RealValue").Value) RecSet.MoveNext Loop  Select Case Language Case 1031      'German = 1031 MsgBox "Schreiben der Datei" &amp; vbCrLf &amp; "C:\Users\os\Documents\" &amp; "Osmosis_" &amp; TimeStamp &amp; "_" &amp; MeasuringPoint &amp; ".csv" &amp; vbCrLf &amp; "erfolgreich!" Case 1033      'English = 1033 MsgBox "Writing of File" &amp; vbCrLf &amp; "C:\Users\os\Documents\" &amp; "Osmosis_" &amp; TimeStamp &amp; "_" &amp; MeasuringPoint &amp; ".csv" &amp; vbCrLf &amp; "successful!" End Select </pre>



## 2 Archive data exporting with WinCC

No.	Action
2.	<p>At the end of the script, the connection to the database is terminated and the objects are released.</p> <pre data-bbox="464 331 1359 734">' Close all ts.Close RecSet.Close Set ReCset = Nothing Set Command = Nothing conn.Close           'Close connection Set Conn = Nothing Set fso = Nothing Set f = Nothing Set ts = Nothing End Function</pre>
3.	Save the script. The script is now available for call.

### Calling the Function in the Process Screen

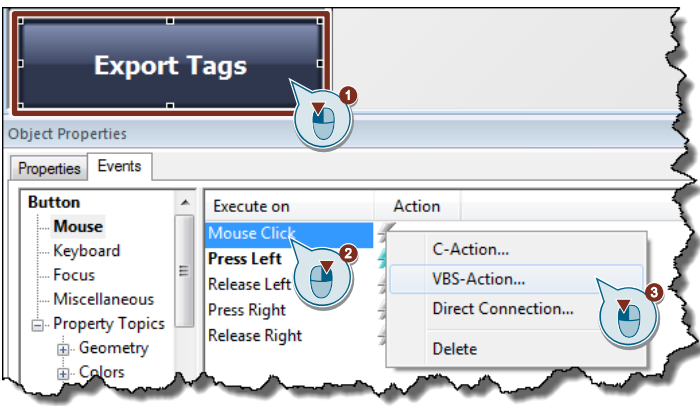
The function "WriteArchiveValuesToCSV" needs the two parameters "Archivename" and "MeasuringPoint" for the call.

The parameter "Archive name" corresponds to the name of the archive in which the variable to be read is located. In this application example, the archive is "Process value archive".

The "MeasuringPoint" corresponds to the name of the archive variable to be read (tag), for example the variable "QI900\_ConductivityPermeatIn" is to be used.

In the image "ReverseOsmosis.Pdl" the script is called via the corresponding button.

Table 2-7

No.	Action
1.	Open the image "ReverseOsmosis.Pdl" in the Graphics Designer.
2.	<p>Open the properties dialog of the "Export Variable" button and open the DDPS editor under "Events &gt; Mouse &gt; Mouse Click &gt; DDPS Action".</p> 
3.	<p>Call the previously created script with the following code.</p> <pre data-bbox="467 1279 1369 1529"> Sub OnClick(Byval Item) Dim ArchiveName Dim MeasuringPoint ArchiveName = "Prozesswertarchiv" MeasuringPoint = "QI900_ConductivityPermeatIn" Call WriteArchiveValueToCSV (ArchiveName, MeasuringPoint) End Sub                     </pre>
4.	Save the image.

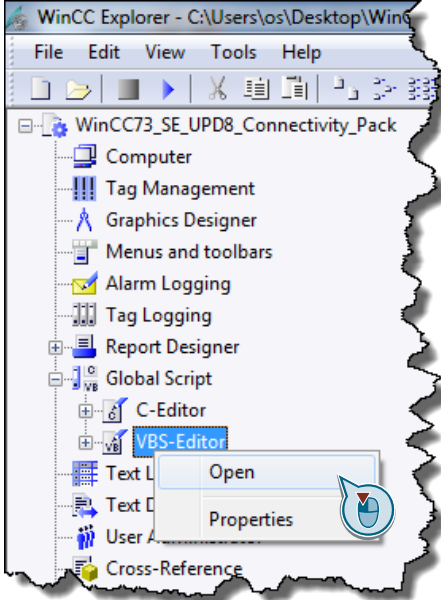
### 2.2.4 Script for exporting archived messages

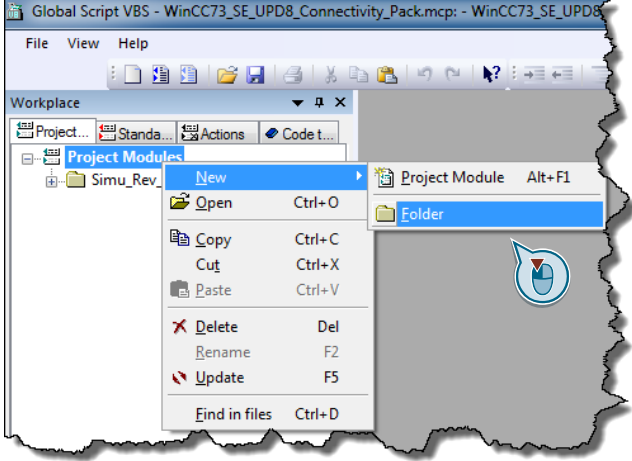
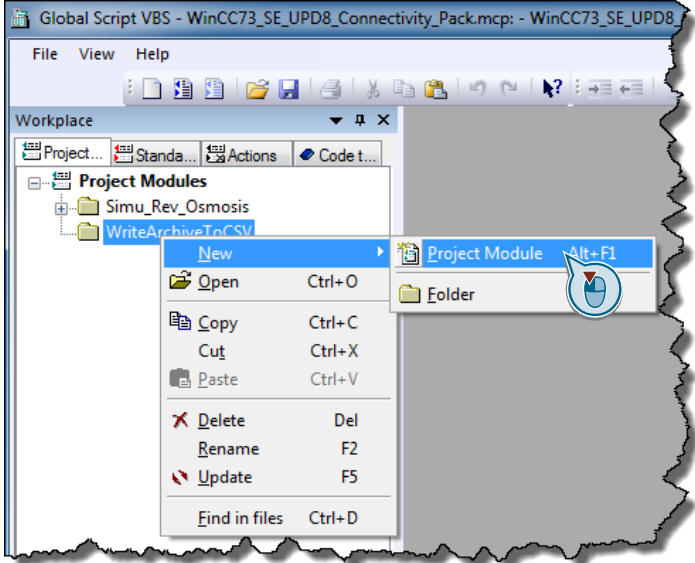
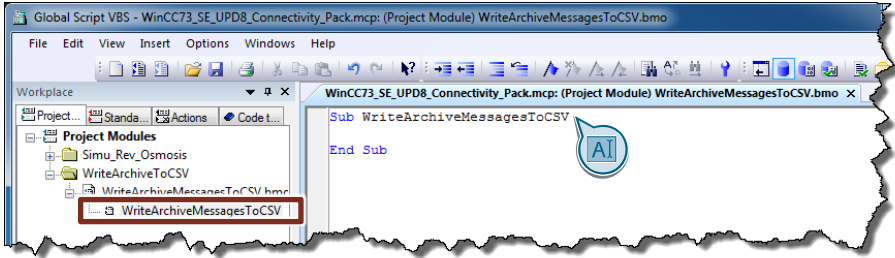
The start-up and shut-down times as well as the stop time of the osmosis plant are entered in the message archive as operating messages. You have the registration number 1-3. In this application example, the start-up, shutdown and stop messages that have occurred in the last 24 hours are to be written to a ".csv" file.

The procedure corresponds here essentially to the writing of measured value archives (section 2.2.3) in a ".csv file". Differences exist only in the structure of the RecordSet and in the query of the data records.

#### Creation of the VBS module

Table 2-8

No.	Action
1.	<p>Start the Global VBS Editor in the WinCC application example.</p>  <p>The screenshot shows the WinCC Explorer interface. The 'Global Script' folder is expanded, and the 'VBS-Editor' icon is highlighted. A context menu is open over the 'VBS-Editor' icon, showing 'Open' and 'Properties' options. The 'Open' option is selected.</p>

No.	Action
2.	<p>If section 2.2.3 has already been edited, step no. 2 can be omitted.                      Select the Project Module tab and create a new folder named WriteArchiveToCSV.</p>  <p>The screenshot shows the WinCC Project Modules window. The 'Project Modules' tree is expanded to show 'Simu_Rev.'. A right-click context menu is open, and the 'New' option is selected. A sub-menu is displayed with 'Project Module' and 'Folder' as options. The 'Folder' option is highlighted.</p>
3.	<p>Create a new project module in this folder.</p>  <p>The screenshot shows the WinCC Project Modules window. The 'Project Modules' tree is expanded to show 'Simu_Rev_Osmosis'. A new project module named 'WriteArchiveToCSV' has been created and is highlighted. A right-click context menu is open, and the 'New' option is selected. A sub-menu is displayed with 'Project Module' and 'Folder' as options. The 'Project Module' option is highlighted.</p>
4.	<p>Change the name of the procedure and save the Project module as "WriteArchiveMessagesToCSV".</p>  <p>The screenshot shows the WinCC Project Modules window. The 'Project Modules' tree is expanded to show 'Simu_Rev_Osmosis'. The project module 'WriteArchiveToCSV' has been renamed to 'WriteArchiveMessagesToCSV' and is highlighted. A right-click context menu is open, and the 'New' option is selected. A sub-menu is displayed with 'Project Module' and 'Folder' as options. The 'Project Module' option is highlighted.</p>

**VBS-Code: Creating the "\*.csv" file**

In the first step of the VBS code, the "\*.csv" file is first created in the folder "C:\Temp". The file name should consist of the plant name "Osmosis", the start time of the osmosis plant and the addition "Messages".

For Example: C:\Temp\Osmosis\_15.02.2015 10:30:00\_Messages.csv

Table 2-9

No.	Action
1.	<p>Declare the following variables:</p> <pre data-bbox="464 568 1362 824"> 'Declaration of local Tags Dim fso           'FileSystemObject Dim f             'File Dim ts           'TextStream Dim path         'path Dim StartTime    'Date and Time when writing is triggered Dim TimeStamp    'Timestamp for bulding filename                     </pre>
2.	<p>Specify the path and file name. The variable "path" contains the complete file name with path in the form:                      C:\users\<username&gt;\documents\osmosis_ +="" _messages.csv<br="" start="" time=""></username&gt;\documents\osmosis_>                     Use the following code to read out the current system time and define it as the start time. The start time is used in the application example to create the file name and must first be converted to a suitable format.</p> <pre data-bbox="464 1028 1362 1308"> 'Date and Time when writing is triggered StartTime = Now 'Generate String for the *.csv filename and replace ":" with "_" TimeStamp = FormatDateTime(StartTime, vbGeneralDate) TimeStamp = Replace(TimeStamp, ":", "_") 'Path and name for the *.csv file path = "C:\Temp\" &amp; "Osmosis_" &amp; TimeStamp &amp; "_Messages.csv"                     </pre>

No.	Action
3.	<p>Use the following code to create the file system object (fso) and the file object (f) using the previously created path and file name (path). The system first checks whether the file already exists. In this case, processing of the function ends after a message has been issued.</p> <pre>'Create FileSystemObject and *.csv file if it not exist: Set fso = CreateObject("Scripting.FileSystemObject") If Not fso.FileExists(path) Then     fso.CreateTextFile(path) Else     MsgBox "File already exist:" &amp; vbCrLf &amp; path Exit Sub End If</pre> <p>The "*.csv" file with the TextStream object (ts) is then opened for writing.</p> <pre>'Create File-Object and open this File Set f = fso.GetFile(path) Set ts = f.OpenAsTextStream(2,-2) 'iomode = 2, Writing 'format = -2, TristateUseDefault '*.csv-File is now ready for Writing</pre>

### Connection to the Database

In the second step of the sub-procedure, the connection to the database is established using the WinCC OLE DB Provider. At least one of the following points is required for the compound.

- The archive from which the data is to be read is part of a WinCC runtime project.
- The archive from which the data is to be read is connected in the SQL Manager via "Attach Database".
- The archive from which the data is to be read is connected via the "Archive Connector".

An SQL command set is formed for the query of the message archive data. A message archive data record is structured accordingly [Table 2-10](#). The field names can be addressed directly in the SQL query.

Table 2-10

Field No.	Field name	Type	Meaning
1	MsgNr	Integer (4 Byte)	Message number
2	State	Small Integer 2 Byte	Status of the message
3	DateTime	DateTime 8 Byte	Message timestamp (date time without milliseconds)
4	Ms	Small Integer 2 Byte	Message time stamp (milliseconds)
5	Instance	VarChar(255)	Instance name of the message
6	Flags1	Integer 4 Byte	(internal use only)
7	PValueUsed	Integer 4 Byte	Process Values used
8-17	PValue1 to	Real 8 Byte	Numerical process value 1 to 10

## 2 Archive data exporting with WinCC

Field No.	Field name	Type	Meaning
	PValue10		
18-28	PText1 to PText10	VarChar(255)	Process value text 1 to 10
28	Computer name	VarChar(255)	Computer name
29	Application	VarChar(255)	Application name
30	Comment	VarChar(255)	Comment
31	User name	VarChar(255)	User name
32	Counter	Integer 4 Byte	Continuous message counter
33	TimeDiff	Integer 4 Byte	Time difference to "Coming" state
34	Class name	VarChar(255)	Name of reporting class
35	Type name	VarChar(255)	Name of message type
36	Class	Small Integer 2 Byte	ID of the message class
37	Type	Small Integer 2 Byte	ID of the message type
38 to 47	Text1 to Text10	VarChar(255)	Message Text 1 to 10
48	AG_NR	Small Integer 2 Byte	Number of the AG
49	CPU_NR	Small Integer 2 Byte	CPU number
50	CrComeFore	Integer 4 Byte	Foreground color for "come" status
51	CrComeBack	Integer 4 Byte	Background color for status "come"
52	CrGoFore	Integer 4 Byte	foreground color for status "gone"
53	CrGoBack	Integer 4 Byte	Background color for status "gone"
54	CrAckFore	Integer 4 Byte	Foreground color for status "acknowledged"
55	CrAckBack	Integer 4 Byte	Background Color for the "Acknowledged" status
56	LocalID	Integer 4 Byte	Location of the alarm
57	Priority	Integer 4 Byte	Priority
58	AP_type	Integer 4 Byte	Loop in Alarm
59	AP_name	VarChar(255)	Loop-in-Alarm Function Name
60	AP_PAR	VarChar(255)	Loop-in-Alarm Screen
61	InfoText	VarChar(255)	Infotext
62	TxtCame	VarChar(255)	Text came in
63	TxtWent	VarChar(255)	Text went out
64	TxtCameNWent	VarChar(255)	Text came in and went out
65	TxtAck	VarChar(255)	Text acknowledged
66	AlarmTag	Integer 4 Byte	Message tag
67	AckType	Small Integer 2 Byte	Acknowledgment Type
68	Params	Integer 4 Byte	Parameters
69	Server name	VarChar(255)	Server name

**Establishment of connection**

To establish the connection to the database, a connection object is created and the "ConnectionString" is passed. The "ConnectionString" requires the following parameters.

- Provider names
- The provider instance number
- The user DSN (GUID of the WinCC project)
- As data source WinCC incl. the computer name.

General format of the "ConnectionString":

ConnectionString = "Provider=<Provider name>.<Instance no.>;Catalog=<DSN of the database>R;Data Source=<Computer name>\WinCC".

**Note**

Access is also possible without specifying the computer name (.WinCC). However, the performance is lower.  
For transparent access for "Catalog", enter the name of the WinCC project, e.g.: "Catalog=WinCC\_Project\_Name".

**Requesting of data**

To query the data, a command object is created to which a "CommandText" is passed for processing. The "CommandText" contains the following parameters.

- One command set (ALARMVIEWEX>Select)
- The name of the database table (ViewName)
- Conditions:

General format of the "CommandText":

CommantText = "<Command set> \* FROM <Name of database table> WHERE <Condition>"

Table 2-11

Parameters	Description
ViewName	<p>Name of the database table. The table must be specified in the desired language. The "ViewName" for the five European language is e.g.:</p> <p>ALGVIEWEXDEU: German message archive data                      ALGVIEWEXENU: English message archive data                      ALGVIEWEXESP: Spanish message archive data                      ALGVIEWEXFRA: French message archive data                      ALGVIEWEXITA: Italian message archive data</p> <p>The "ViewName" for the Asian language is e.g.:</p> <p>ALGVIEWEXCHS: Chinese (simplified) message archive data                      ALGVIEWEXCHT: Chinese (traditional) message archive data                      ALGVIEWEXJPN: Japanese message archive data                      ALGVIEWEXKOR: Korean message archive data</p>
Condition	Specifies the condition(s) for the message archive query.



Parameters	Description
	<p>Example of requests                      DateTime&gt;'2003-06-01' AND DateTime&lt;'2003-07-01'                      DateTime&gt;'2003-06-01 17:30:00'                      MsgNr = 5                      MsgNr in (4, 5)                      State = 2</p> <p><b>Note</b>                      Only absolute times can be used with DateTime.</p>

Example:

The "CommandText" reads all German entries of message number 5 which were entered after 01.01.2016:

```
ALARMVIEWEX:Select * FROM ALGVIEWEXDEU WHERE MsgNr=5 and DateTime > '2016-01-01'
```

**Note**

From WinCC V7.2 UNICODE is supported. Therefore the archive access of alarms has changed. With older versions of WinCC (less than V7.2) the alarm archive is accessed via ALARMVIEW (instead of ALARMVIEWEX).

It is possible to query several conditions at the same time. Strings must be linked with an "&" character. Condition enumeration by "AND", e.g:  
 CommandText = "<Command set>\* FROM <Language setting> WHERE" & "<Condition 1>" & "AND <Condition 2>" & "AND <Condition 2>"

The languages that are installed in the WinCC base system or that are configured in the WinCC Text Library are supported. Information concerning the possible query-languages or the respective "ViewName" can be found in the SQL-Server in the linked alarm archives under "Views". All languages that are supported in the corresponding archive are shown here with their IDs, e.g. "ALGVIEWEXENU".

**Note**

The start time and stop time can be transferred absolute or relative (see Table 2-4).

**Establishing a connection and requesting data**

Table 2-12

No.	Action
1.	<p>Declare the other variables:</p> <pre> 'Declaration of Script Tags Dim Pro           'Provider Dim DSN           'Data Source Name Dim DS            'Data Source Dim ConnString   'Connection String Dim MachineNameRT 'Name of the PC from WinCC-RT Dim DSNRT         'Data Source Name from WinCC-RT Dim Conn         'Connection to ADO DB Dim CommandText  'Command-Text Dim SqlSec       'Duration of Production-Cycle Dim SqlMin       'Duration of Production-Cycle Dim SqlHour      'Duration of Production-Cycle Dim SqlDay       'Duration of Production-Cycle Dim SqlMonth     'Duration of Production-Cycle Dim SqlYear      'Duration of Production-Cycle Dim RecSet       'RecordSet Dim Command      'Query Dim Language     'Language Tag                     </pre>
2.	<p>Use the following code to read the host name and GUID of the project from the @LocalMachineName and @DataSourceNameRT environment variables into the previously declared local variables.</p> <pre> 'Read the name of the PC-Station and the DSN-Name from 'WinCC-RT Set MachineNameRT = HMIRuntime.Tags("@LocalMachineName") Set DSNRT = HMIRuntime.Tags("@DataSourceNameRT")                     </pre> <p>You use these variables to form the "ConnectionString".</p> <pre> 'Preparing the Connection-String 'First instance of WinCCOLEDB Pro = "Provider=WinCCOLEDBProvider.1;" 'Name of Runtime-Database DSN = "Catalog=" &amp; DSNRT.Read &amp; ";" 'Data Source DS = "Data Source=" &amp; MachineNameRT.Read &amp; "\WinCC" 'Build the complete String: ConnString = Pro + DSN + DS                     </pre>
3.	<p>You then establish the connection to the database.</p> <pre> 'Connection based on ODB-Provider Set Conn = CreateObject("ADODB.Connection") Conn.ConnectionString = ConnString Conn.CursorLocation = 3 Conn.open                     </pre>

No.	Action
4.	<p>To query the data in the WinCC database, the "Command Text" is required. In this example, the previously saved start time of the query (pressing the button) is recorded. This point in time must be reformatted to the display format "YYYY-MM-DD hh:mm:ss".</p> <p>During formatting, the leading zeros are also generated for numerical values smaller than 10.</p> <pre data-bbox="467 454 1356 1182"> 'FormatStarttime for SQL-Statement 'Fomat needed for StartTime: jjjj-mm-dd hh:mm:ss 'Date and time 24 hours before StartTime = DateAdd("h", -24, StartTime) 'Split in years, months, days, hours, minutes, seconds SqlSec = Second(StartTime) SqlMin = Minute(StartTime) SqlHour = Hour(StartTime) SqlDay = Day(StartTime) SqlMonth = Month(StartTime) 'Creating leading zeroes SqlSec = Right("00" &amp; sqlSec,2) SqlMin = Right("00" &amp; sqlMin,2) SqlHour = Right("00" &amp; sqlHour,2) SqlDay = Right("00" &amp; sqlDay,2) SqlMonth = Right("00" &amp; sqlMonth,2) SqlYear = Right("00" &amp; sqlYear,2) 'Create string StartTime = "" &amp; SqlYear &amp; "-" &amp; SqlMonth &amp; "-" &amp; SqlDay &amp; " " &amp; SqlHour &amp; ":" &amp; SqlMin &amp; ":" &amp; SqlSec &amp; ""                     </pre>
5.	<p>The complete query is displayed in the variable "CommandText" for further processing.</p> <p>In this example, all messages stored in the message archive with message number &lt; 4 are read out. If the runtime language is set to German (HMIRuntime.Language=1031), ALGVIEWEXDEU is entered for the "ViewName" parameter. If the English runtime language HMIRuntime.Language=1033) is active, "ALGVIEWEXENU" is selected.</p> <pre data-bbox="467 1413 1356 1753"> 'Building the complete string Language = HMIRuntime.Language Select Case Language Case 1031          'German = 1031 CommandText="ALARMVIEWEX:Select*FROM ALGVIEWEXDEU WHERE DateTime &gt;" &amp; StartTime&amp; "MsgNr&lt; 4 AND STATE = 1" Case 1033          'English = 1033 CommandText="ALARMVIEWEX:Select*FROM ALGVIEWEXENU WHERE DateTime &gt;" &amp; StartTime&amp; "MsgNr&lt; 4 AND STATE = 1" End Select                     </pre>

No.	Action
6.	<p>The command object is then created and the query is executed with the previously created "CommandText".</p> <p>The RecordSet is set to the command object and then set to the first record in which the first recorded and therefore oldest message is stored.</p> <pre data-bbox="464 394 1369 676"> 'Create the recordset, read the records and set the first recordset: Set Command = CreateObject("ADODB.Command") Command.CommandType = 1 Set Command.ActiveConnection = Conn Command.CommandText = CommandText Set RecSet = Command.Execute RecSet.MoveFirst                     </pre>

**Note**

When working with time and date in WinCC and Visual Basic, please note that the date format depends on the current region settings in Windows. The DDPS code should be designed so that the date and time can be evaluated independently of these computer-specific settings.

**Writing the "\*.csv" file**

The records will then be written from the previously opened RecordSet to the "\*.csv" file.

Table 2-13

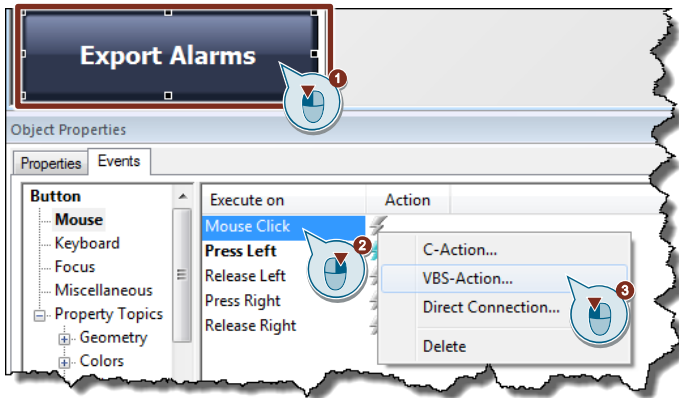
No.	Action
1.	<p>A progress bar indicates writing of the data records. In the example, the following Recordset fields are read from the data record and written to the "*.csv" file.</p> <ul style="list-style-type: none"> <li>• DateTime (Time stamp)</li> <li>• MsgNr (Message number)</li> <li>• Text1 (Event text1)</li> <li>• Classname (Message class)</li> <li>• Type name (Message type)</li> </ul> <p>A line with the column headings is created once before, the currently set runtime language is also taken into account here.</p> <p>The RecordSet is placed on the next record with each run.</p> <pre data-bbox="467 813 1369 1529"> 'write recordsets To *.csv-File 'Header in CSV-File Language = HMIRuntime.Language Select Case Language Case 1031           'German = 1031 ts.WriteLine ("Datum/Zeit;Meldenr.;Ereignis;Meldeklasse;Meldetyp") Case 1033           'English = 1033 ts.WriteLine ("Date/Time;MsgNr.;Event;Messages Class;Messages Typ") End Select  'writing recordsets Do While Not RecSet.EOF ts.WriteLine(RecSet.Fields("DateTime").Value &amp; ";" &amp; RecSet.Fields("MsgNr").Value &amp; ";" &amp; RecSet.Fields("Text1").Value &amp; ";" &amp; RecSet.Fields("Classname").Value &amp; ";" &amp; RecSet.Fields("Typename").Value) RecSet.MoveNext Loop </pre>
2.	<p>At the end of the script, the connection to the database is terminated and the objects are released.</p> <pre data-bbox="467 1615 1369 1977"> ' Close all ts.Close RecSet.Close Set RecSet=Nothing Set Command = Nothing conn.Close           'Close connection Set Conn = Nothing Set fso = Nothing Set f = Nothing Set ts = Nothing </pre>

No.	Action
3.	Save the script.
4.	The script is then available for the call. The script does not require any parameters.

### Calling the Function in the Process Screen

The created procedure "WriteArchiveMessagesToCSV" does not require any transfer parameters for the call. In the image "ReverseOsmosis.Pdl" the script is called up via the corresponding button.

Table 2-14

No.	Action
1.	Open the image "ReverseOsmosis.Pdl" in the Graphics Designer.
2.	Open the properties dialog of the "Export messages" button and open the DDPS editor under "Events > Mouse > Mouse Click > DDPS Action". 
3.	Create the following code to call the previously created VBS script: <pre>Sub OnClick(ByVal Item) WriteArchiveMessagesToCSV End Sub</pre>
4.	Save the image.


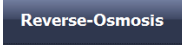


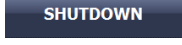

**Note**

Alternatively, the function "WriteArchiveMessageToCSV" can also be called automatically. To do this, create a time-triggered or variable-triggered action in the DDPS editor.


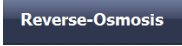


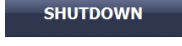
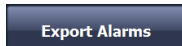
## 2.3 Operation

### 2.3.1 VBS cript for exporting archived variables

Table 2-15

No.	Action	Note
1.	Start the runtime.	
2.	Click the button ("Reverse Osmosis") to open the image "Reverse Osmosis.Pdl".	
3.	Start the reverse osmosis system here via the button ("START") in the control panel on the left-hand side.	
4.	Wait until the plant has reached the status "Production".	
5.	After a short run time, shut down the system with the "DEPTH" button.	
6.	After the system has reached the "Off" status and the stop time has been entered automatically under "Stop", press the "Export Variable" button.	
7.	Then open the "*.csv" file created on the drive under "C:\Temp\".	

### 2.3.2 VBS Script for exporting archived messages

No.	Action	Note
1.	Start the runtime.	
2.	Click the button ("Reverse Osmosis") to open the image "Reverse Osmosis.Pdl".	
3.	Start the osmosis system here via the button ("START") in the control panel on the left-hand side.	
4.	Wait until the plant has reached the status "Production".	
5.	After a short run time, shut down the system with the "DEPTH" button.	
6.	After the system has reached the "Off" status and the stop time has been entered automatically under "Stop", press the "Export Variable" button.	
7.	Then open the "*.csv" file created on the drive under "C:\Temp\".	

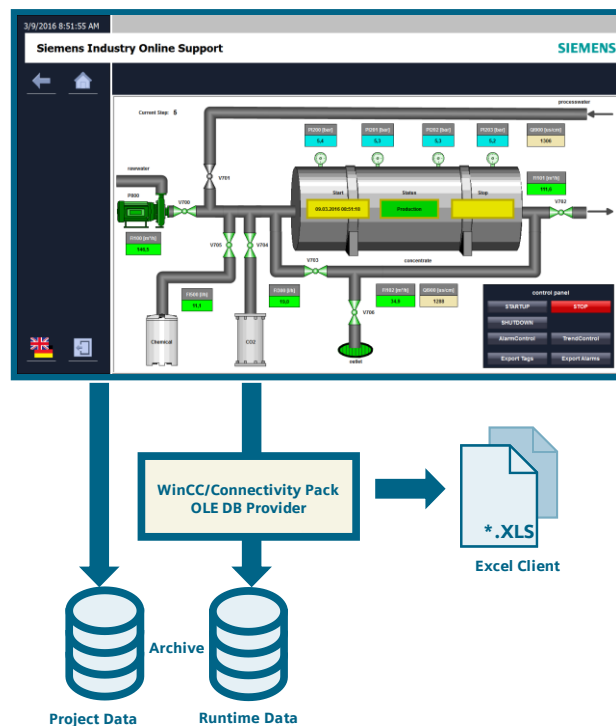
## 3 Archive data exporting with Excel

### 3.1 Principle of operation

The Excel applications in this application example provide the user with simple tools with which he can access the process values and message data archived by WinCC.

- Data access to the runtime archive is done with VBA scripts in Excel.
- The archive data is transferred to Excel via the OLE DB Provider.

Figure 3-1



In this application example we provide you with two Excel clients for download:

- The “Osmosis” client
- The “Universal” client



#### The “Osmosis” client

- The Excel client "38132261\_Application\_Reverse\_Osmosis.xls" has a similar appearance to the associated WinCC project.
- The time interval for the query (process values and messages) is freely adjustable.
- The simultaneous readout of all relevant process values of the osmosis plant and their clear representation is possible.
- The included scripts are optimized for easy traceability. Plausibility checks and UTC time adjustments were therefore deliberately omitted in order to maintain clarity.
- Before using the client, the name of the server and the name of the WinCC RT database must be adapted in the respective script call.

#### The “Universal” client

- The Excel client "38132261\_Application\_Universal\_Client.xls" can be used independently of the respective WinCC project.
- It is possible to read the data from the WinCC runtime database of an OS server as well as a SIMATIC Process Historian.
- Individual compilation of process value and message information is possible.
- The free choice of interpolation of missing values (process values) is supported.
- The time interval for the query (process values and messages) is freely adjustable.
- The conversion between UTC (WinCC runtime database) and local time (Excel client) is performed automatically.
- Access to the archives of different languages (messages) is supported.
- Use of filters such as time span and message type (Messages) is supported.
- An interpretation of the placeholders in the message texts (messages) is possible.
- There is no need to modify scripts in the Excel client, the Excel client can be used for any WinCC project.
- Plausibility queries regarding incorrect entries (e.g. time format) in the Excel client are carried out automatically.

## 3.2 The “Osmosis” client

### 3.2.1 Principle of operation details

This chapter contains information that will make it easier for you to extend the Excel application or adapt it to your individual needs.

**Note** This chapter is only relevant for you if you are interested in the programmatic implementation or if you want to make changes to the Excel client.

Basic knowledge of VBA programming is required.

#### SQL command

This section will help you to understand the function of the SQL command for reading the data. If you have basic knowledge of VBA programming and SQL database language, you will be able to extend the application in various ways after reading this manual.

**Note** The SQL command used here contains a modification that is specifically required for communication with the WinCC OLE DB Provider. This causes the command to deviate partially from the specified standard.

**Reading the variable archive**

The call by the button "ReadValues":

```
'Execute writing archive values to excel sheet
1 MeasuringPoint = "FI100_FlowRawwater"
2 ExcelSheetName = "FI100"
3 ErrorNumber = WriteArchiveValue(MeasuringPoint, ExcelSheetName)
4 If ErrorNumber > 0 Then GoTo ErrorHandler
```

Table 3-1

Step	Explanation
1.	<b>MeasuringPoint = "FI100_FlowRawwater"</b> Enter here the name of the WinCC variable whose values are to be read from WinCC Runtime; here: FI100_FlowRawwater.
2.	<b>ExcelSheetName = "FI100"</b> Enter the name of the Excel worksheet in which the read values are to be written; here: FI100.
3.	<b>ErrorNumber = WriteArchiveValue(MeasuringPoint, ExcelSheetName)</b> The function "WriteArchiveValue" is called with the previously defined parameters. The variable "ErrorNumber" serves as the return value.
4.	<b>If ErrorNumber &gt; 0 Then GoTo ErrorHandler</b> If the return value is greater than 0, further processing is aborted and error handling is called.

The function "WriteArchiveValue" (in extracts):

```
1 'Preparing the Connection-String
Pro = "Provider=WinCCOLEDBProvider.1;" 'First instance of WinCCOLEDB
DSN = "Catalog=" & cDSNRT & ";" 'Name of Runtime-Database
DS = "Data Source=" & cMachineNameRT & "\WinCC" 'Data Source

2 'Building the complete connection string
ConnString = Pro + DSN + DS

3 'Make Connection
Set Conn = CreateObject("ADODB.Connection")
Conn.ConnectionString = ConnString
Conn.CursorLocation = 3
Conn.Open

4 'Building the complete command string:
CommandText = "TAG:R,'" & cArchiveName & "\" & MeasuringPoint &
"', '" & StartArchive & ".000','" & StopArchive & ".000','TIMESTEP="
& TimeStep & ",5'"

5 'Create the recordset, read the records and set to first recordset
Set Command = CreateObject("ADODB.Command")
Command.CommandType = 1
Set Command.ActiveConnection = Conn
Command.CommandText = CommandText
Set RecSet = Command.Execute
RecSet.MoveFirst

6 'Write recordset To Excel-Sheet
With Worksheets(ExcelSheetName)
.Range(.Rows(FirstLine), .Rows(.Rows.Count)).ClearContents
End With
ActualRow = FirstLine

Do While Not RecSet.EOF
    'Sequence: MeasuringPoint | ValueID | TimeStamp | RealValue
    With Worksheets(ExcelSheetName)
        .Cells(ActualRow, 1).Value = MeasuringPoint
        .Cells(ActualRow, 2).Value = RecSet.Fields("ValueID").Value
        .Cells(ActualRow, 3).Value =
RecSet.Fields("TimeStamp").Value
        .Cells(ActualRow, 4).Value =
RecSet.Fields("RealValue").Value
    End With

    RecSet.MoveNext
    ActualRow = ActualRow + 1
Loop
```

Table 3-2

Step	Explanation
1.	<p><b>Pro = "Provider=WinCCOLEDBProvider.1;"</b>                      An instance of the WinCC OLE DB provider is assigned to the provider variable;                      here: Provider=WinCCOLEDBProvider.1;</p> <p><b>DSN = "Catalog=" &amp; cDSNRT &amp; ";"</b>                      The data source name is assigned to the data source name variable;                      here: Catalog=CC_WinCC73__17_09_14_13_34_26R;</p> <p><b>DS = "Data Source=" &amp; cMachineNameRT &amp; "\WinCC"</b>                      The MS SQL Server instance is assigned to the Data Source variable as the data source;                      here: Data Source=HMI-1\WinCC</p>
2.	<p><b>ConnString = Pro + DSN + DS</b>                      The connection string variables are assigned the defined single variables as a whole string;                      here:                      Provider=WinCCOLEDBProvider.1;Catalog=CC_WinCC73__17_09_14_13_34_26R;Data Source=HMI-1\WinCC</p>
3.	<p><b>Set Conn = CreateObject("ADODB.Connection")</b>  <b>Conn.ConnectionString = ConnString : Conn.CursorLocation = 3 : Conn.Open</b>                      The abstraction layer of ADODB opens a connection to the SQL database of WinCC.</p>
4.	<p><b>CommandText = "TAG:R," &amp; cArchiveName &amp; "\" &amp; MeasuringPoint &amp; "," &amp; StartArchive &amp; ".000'," &amp; StopArchive &amp; ".000','TIMESTEP=" &amp; TimeStep &amp; ",5"</b>                      Definition of the variables to be read, with the following specifications:</p> <ul style="list-style-type: none"> <li>• Archive name:</li> <li>• Archive tag</li> <li>• Start time ("StartArchive") of the period to be read out in the archive in <b>UTC</b> format</li> <li>• End time ("StopArchive") of the period to be read out in the archive in <b>UTC</b> format</li> <li>• Specifies the time period ("TimeStep") between two variable values and the aggregation type for summarization.</li> </ul> <p><b>Note:</b>                      For more information on summarization ("TimeStep") and the aggregation type ("5"), see <a href="#">"Optional Adjustments for Reading the Variable Archive"</a>.</p>
5.	<p><b>Set Command = CreateObject("ADODB.Command")</b>  <b>Command.CommandType = 1 : Set Command.ActiveConnection = Conn</b>  <b>Command.CommandText = CommandText : Set RecSet = Command.Execute</b>  <b>RecSet.MoveFirst</b>                      Via the abstraction layer of ADODB ("Command"), the values of the defined variables ("CommandText") are read out via the open connection to the SQL database ("Conn") and assigned to a field ("RecSet").</p>
6.	<p><b>Do While Not RecSet.EOF</b>  <b>With Worksheets(ExcelSheetName)</b>              <b>.Cells(ActualRow, 1).Value = MeasuringPoint</b>              <b>.Cells(ActualRow, 2).Value = RecSet.Fields("ValueID").Value</b>              <b>.Cells(ActualRow, 3).Value = RecSet.Fields("TimeStamp").Value</b>              <b>.Cells(ActualRow, 4).Value = RecSet.Fields("RealValue").Value</b>  <b>End With</b>  <b>RecSet.MoveNext : ActualRow = ActualRow + 1</b>  <b>Loop</b>                      The entire field ("RecSet") is written line by line into the Excel worksheet ("ExcelSheetName") via a loop. Four columns are described for each row:</p> <ul style="list-style-type: none"> <li>• the variable name ("ExcelSheetName")</li> <li>• the variable ID ("ValueID")</li> <li>• der Time stamp ("TimeStamp") in <b>UTC</b>-Format</li> <li>• the variable value ("RealValue")</li> </ul>

### Reading the message archive

The call by the button "ReadValues":

```
'Execute writing archive values to excel sheet  
ExcelSheetName = "Alarms"  
Call WriteAlarms (ExcelSheetName)
```

Table 3-3

Step	Explanation
1.	<b>ExcelSheetName = "Alarms"</b> Enter the name of the Excel worksheet in which the read messages are to be written; here: Alarms.
2.	<b>Call WriteAlarms(ExcelSheetName)</b> The subroutine "WriteAlarms" is called with the previously defined parameter.

The subroutine "WriteAlarms" (in extracts):

```
1 ——— 'Preparing the Connection-String
Pro = "Provider=WinCCOLEDBProvider.1;" 'First instance of WinCCOLEDB
DSN = "Catalog=" & cDSNRT & ";" 'Name of Runtime-Database
DS = "Data Source=" & cMachineNameRT & "\WinCC" 'Data Source

2 ——— 'Building the complete connection string
ConnString = Pro + DSN + DS

3 ——— 'Make Connection
Set Conn = CreateObject("ADODB.Connection")
Conn.ConnectionString = ConnString
Conn.CursorLocation = 3
Conn.Open

——— 'Building the complete command string:
CommandText = "ALARMVIEWEX:Select * FROM ALGVIEWEXENU WHERE DateTime
>' & StartArchive & "' AND DateTime <' & StopArchive & "'"

——— 'Create the recordset, read the records and set the first recordset
Set Command = CreateObject("ADODB.Command")
Command.CommandType = 1
Set Command.ActiveConnection = Conn
Command.CommandText = CommandText
Set RecSet = Command.Execute
RecSet.MoveFirst

6 ——— 'Write recordset To Excel-Sheet
With Worksheets(ExcelSheetName)
.Range(.Rows(FirstLine), .Rows(.Rows.Count)).ClearContents
End With
ActualRow = FirstLine

Do While Not RecSet.EOF
    'Sequence in Excel-Sheet: Date/Time | MsgNr. | Event | Message
    Class | Message Typ
    With Worksheets(ExcelSheetName)
        .Cells(ActualRow, 1).Value = RecSet.Fields("DateTime").Value
        .Cells(ActualRow, 2).Value = RecSet.Fields("MsgNr").Value
        .Cells(ActualRow, 3).Value = RecSet.Fields("Text1").Value
        .Cells(ActualRow, 4).Value =
RecSet.Fields("Classname").Value
        .Cells(ActualRow, 4).Value = RecSet.Fields("Typename").Value
    End With

    RecSet.MoveNext
    ActualRow = ActualRow + 1
Loop
```

### 3 Archive data exporting with Excel

Table 3-4

Step	Explanation
1.	<p><b>Pro = "Provider=WinCCOLEDBProvider.1;"</b>            An instance of the WinCC OLE DB provider is assigned to the provider variable;            here: Provider=WinCCOLEDBProvider.1;</p> <p><b>DSN = "Catalog=" &amp; cDSNRT &amp; ";"</b>            The data source name is assigned to the data source name variable;            here: Catalog=CC_WinCC73__17_09_14_13_34_26R;</p> <p><b>DS = "Data Source=" &amp; cMachineNameRT &amp; "\WinCC"</b>            The MS SQL Server instance is assigned to the Data Source variable as the data source;            here: Data Source=HMI-1\WinCC</p>
2.	<p><b>ConnString = Pro + DSN + DS</b>            The connection string variables are assigned the defined single variables as a whole string;            here:            Provider=WinCCOLEDBProvider.1;Catalog=CC_WinCC73__17_09_14_13_34_26R;Data Source=HMI-1\WinCC</p>
3.	<p><b>Set Conn = CreateObject("ADODB.Connection")</b>  <b>Conn.ConnectionString = ConnString : Conn.CursorLocation = 3 : Conn.Open</b>            The abstraction layer of ADODB opens a connection to the SQL database of WinCC.</p>
4.	<p><b>CommandText = "ALARMVIEWEX&gt;Select * FROM ALGVIEWEXENU WHERE DateTime &gt;" &amp; StartArchive &amp; " AND DateTime &lt;" &amp; StopArchive &amp; ""</b>            Definition of the messages to be read, with the following specifications:</p> <ul style="list-style-type: none"> <li>• Language of the messages to be read („ALGVIEWEXENU“)</li> <li>• Start time ("StartArchive") of the period to be read out in the archive in <b>UTC</b> format</li> <li>• End time ("StopArchive") of the period to be read out in the archive in <b>UTC</b> format</li> </ul> <p><b>Note:</b>            Further information on the language of the messages to be read out can be found under <a href="#">"Optional adjustments for reading out the message archive"</a>.</p>
5.	<p><b>Set Command = CreateObject("ADODB.Command")</b>  <b>Command.CommandType = 1 : Set Command.ActiveConnection = Conn</b>  <b>Command.CommandText = CommandText : Set RecSet = Command.Execute</b>  <b>RecSet.MoveFirst</b>            Via the abstraction layer of ADODB ("Command"), the values of the defined variables ("CommandText") are read out via the open connection to the SQL database ("Conn") and assigned to a field ("RecSet").</p>
6.	<p><b>Do While Not RecSet.EOF</b>  <b>With Worksheets(ExcelSheetName)</b>  <b>.Cells(ActualRow, 1).Value = RecSet.Fields("DateTime").Value</b>  <b>.Cells(ActualRow, 2).Value = RecSet.Fields("MsgNr").Value</b>  <b>.Cells(ActualRow, 3).Value = RecSet.Fields("Text1").Value</b>  <b>.Cells(ActualRow, 4).Value = RecSet.Fields("Classname").Value</b>  <b>.Cells(ActualRow, 4).Value = RecSet.Fields("Typename").Value</b>  <b>End With</b>  <b>RecSet.MoveNext : ActualRow = ActualRow + 1</b>  <b>Loop</b>            The entire field ("RecSet") is written line by line into the Excel worksheet ("ExcelSheetName") via a loop. Four columns are described for each row:</p> <ul style="list-style-type: none"> <li>• the Time stamp ("TimeStamp") in <b>UTC</b> format</li> <li>• MsgNr (Message number)</li> <li>• the message text</li> <li>• classname (Message class)</li> <li>• type name (Message type)</li> </ul>



### 3.2.2 Configuration and project planning

#### Storage Location

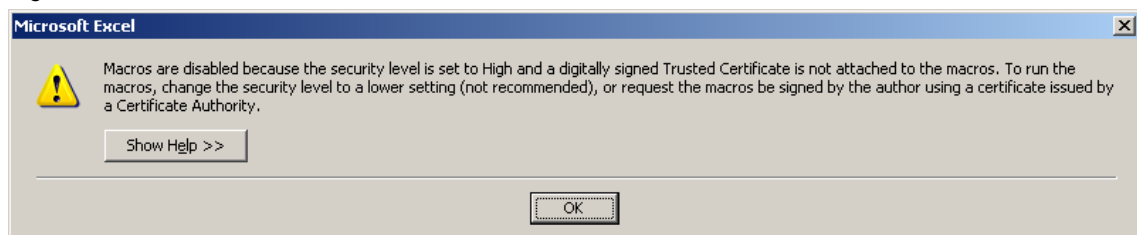
The application does not have to be installed separately. Copy the file "38132261\_Application\_Reverse\_Osmosis.xls" directly to the desired location. This Excel file contains the entire source code in the form of macros, which are executed in the background.

#### Application call up

Start the application by double-clicking on the Excel file.

As soon as the application has opened, the following warning may appear, depending on your Excel settings:

Figure 3-2

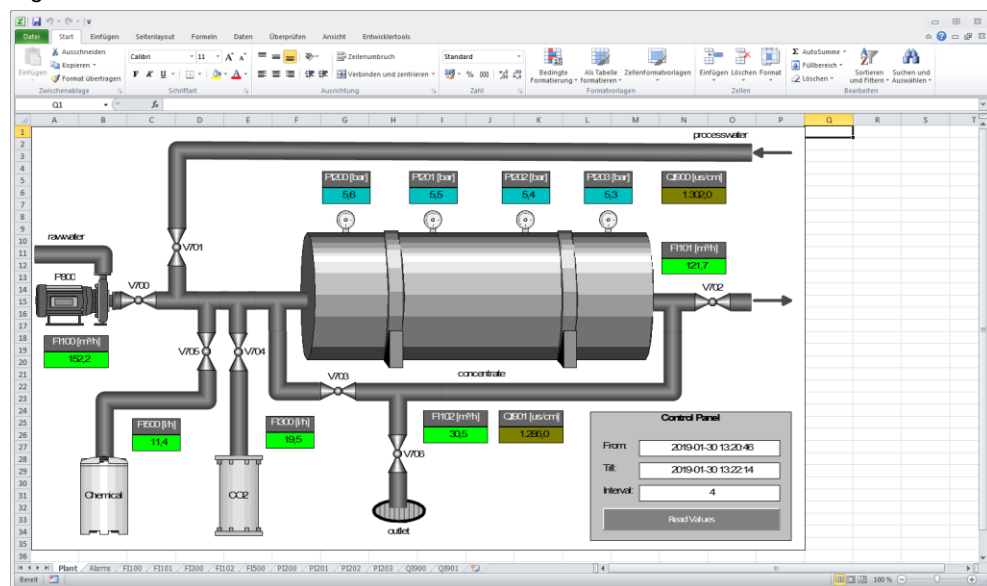


The reason for this is that your macro security in Excel is set to "High". Change this setting accordingly.

#### Overview

The [Figure 3-3](#) shows the application directly after its first call. Which interface is displayed depends on which register was active during the last save operation. The entries are also those that were made when you last saved.

Figure 3-3

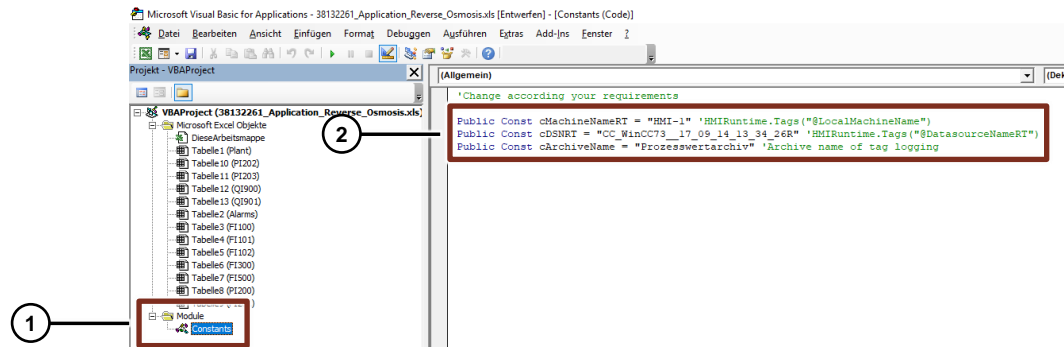


### Required adjustments for reading the archives

Computer-specific adjustments in the declaration area of the Excel client are required to read out the archives:

1. Open the script editor of Excel with the key combination <ALT> + <F11>.
2. Click on the "Constants" module (1) in the navigation area.

Figure 3-4



3. Change the name of the PC (2) on which the WinCC Runtime is running; here: **HMI-1**.

**Note**

The name can also be read in WinCC Runtime via the system variable "@LocalMachineName".

4. Change the Data Source Name of the WinCC Runtime Database (2); here: **CC\_WinCC73\_17\_09\_14\_13\_34\_26R**.

**Note**

- Make sure that the data source name of the WinCC Runtime database ends with an "R".
- The Data Source Name can also be read in WinCC Runtime via the system variable "@DataSourceNameRT".
- Alternatively, you can also follow steps 2 to 4 in [Table 4-1](#) to read the Data Source Name of the WinCC Runtime Database under "Databases".
- Alternatively, you can also use the name "CC\_ExternalBrowsing".

5. Enter the name of the variable archive (2) of the WinCC Runtime; here: **Process value archive**



**Optional adaptations for reading the message archive**

Depending on the desired language of the queried messages, you can make appropriate changes in the "WriteAlarms" script.

Scroll down to the entry "Building the complete command string":

Figure 3-6

```
'Building the complete command string:
CommandText = "ALARMVIEWEX:Select * FROM ALGVIEWEXENU WHERE DateTime >' & StartArchive & "' AND DateTime <' & StopArchive & ""
'ALARMVIEWEX:Select * FROM ALGVIEWEXDEU WHERE DateTime >'StartArchive' AND DateTime <'StopArchive'"
-----condition1
-----condition 2
----- ViewName for german messages
----- Command for alarmview
```

The following parameters are reserved for the ViewName:

Table 3-6

ViewName (European languages)	Language of the message texts
ALGVIEWEXDEU	German message archive data
ALGVIEWEXENU	English message archive data
ALGVIEWEXESP	Spanish message archive data
ALGVIEWEXFRA	French message archive data
ALGVIEWEXITA	Italian message archive data

Table 3-7

ViewName (Asian languages)	Language of the message texts
ALGVIEWEXCHS	Chinese (simplified) message archive data
ALGVIEWEXCHT	Chinese (traditional) message archive data
ALGVIEWEXJPN	Japanese message archive data
ALGVIEWEXKOR	Korean message archive data

**Note**

The languages that are installed in the WinCC base system or that are configured in the WinCC Text Library are supported. Information concerning the possible query-languages or the respective "ViewName" can be found in the SQL-Server in the linked alarm archives under "Views". It displays all languages with their identifiers, e.g. "ALGVIEWEXENU", which are supported in the respective archive.

### 3.2.3 Operation

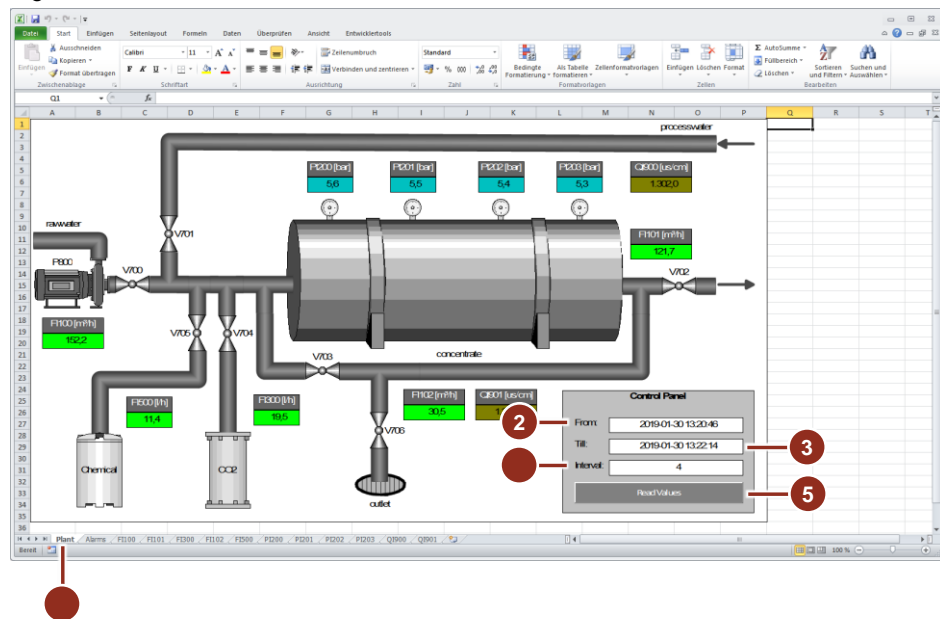
#### Basic Steps

1. Set the WinCC project "38132261\_Application\_Reverse\_Osmosis\_PROJ.zip" to runtime as described in chapter [2.2.1](#) "Configurations in the sample project".
2. Open the supplied Excel client "38132261\_Application\_Reverse\_Osmosis.xls".
3. Make the settings on the Excel client as described in chapter [3.2.2](#) "Configuration and project engineering".

#### Read tag values

1. Select the "Plant" tab.
2. Enter the start time of the archive query in the format "YYYY-MM-DD hh:mm:ss". The time is specified in **UTC** format.  
**Note:**  
No plausibility checks are performed.
3. Enter the start time of the archive query in the format "YYYY-MM-DD hh:mm:ss". The time is specified in **UTC** format.  
**Note:**  
No plausibility checks are performed.
4. Specify the query interval in seconds.
5. Click on the "ReadValues" button.

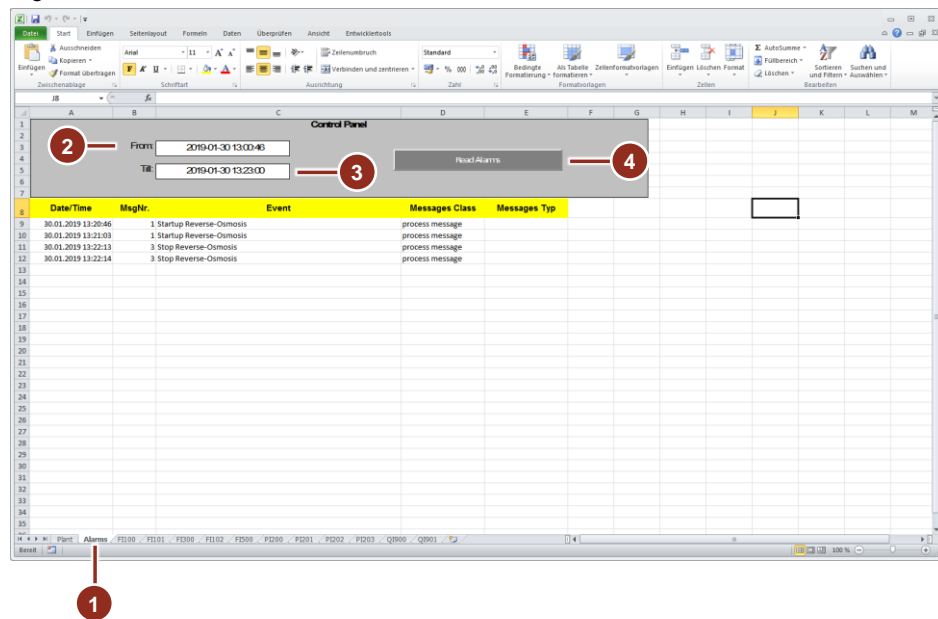
Figure 3-7



#### Read messages

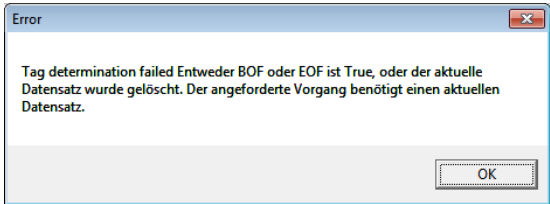
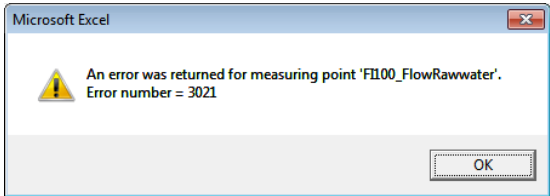
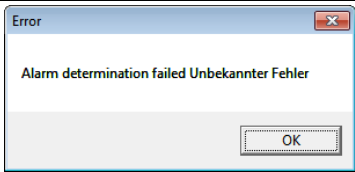
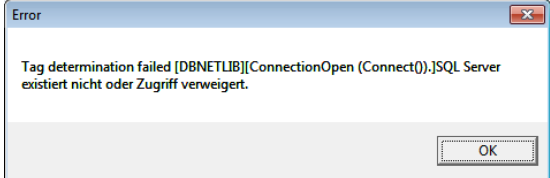
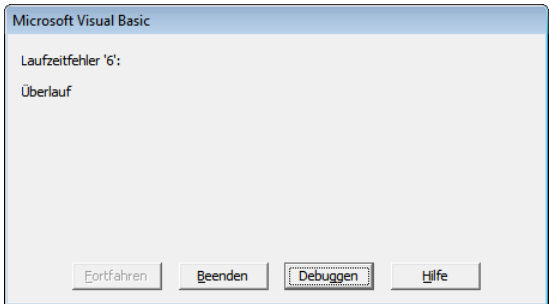
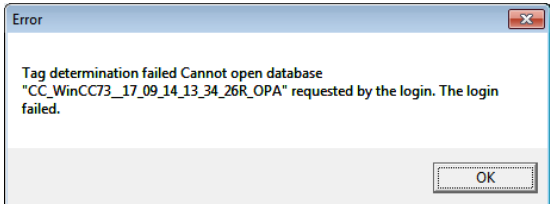
1. Select the "Alarms" tab.
2. Enter the start time of the archive query in the format "YYYY-MM-DD hh:mm:ss". The time is specified in **UTC** format.  
**Note:**  
No plausibility checks are performed.
3. Enter the start time of the archive query in the format "YYYY-MM-DD hh:mm:ss". The time is specified in **UTC** format.  
**Note:**  
No plausibility checks are performed.
4. Click on the "ReadAlarms" button.

Figure 3-8



### 3.2.4 Error handling

Table 3-8

Error	Possible cause
	<p><b>Access to variable archive and/or message archive</b></p> <ul style="list-style-type: none"> <li>The Excel client tries to read the variable archive, but the WinCC project was not started (RT not active).</li> <li>The Excel client attempts to read the variable archive, but there are no values for the period queried.</li> <li>The Excel client attempts to read the message archive, but there are no values for the period queried.</li> </ul>
	<p>Follow-up message Access to variable archive</p> <ul style="list-style-type: none"> <li>The Excel client tries to read the variable archive, but the WinCC project was not started (RT not active).</li> <li>The Excel client attempts to read the variable archive, but there are no values for the period queried.</li> </ul>
	<p>Access to message archive</p> <ul style="list-style-type: none"> <li>The Excel client tries to read the message archive, but the WinCC project was not started (RT not active).</li> <li>The Excel client tries to read the message archive, but the PC name in the script call is wrong.</li> <li>The Excel client tries to read the message archive, but the DataSourceName (DSN) in the script call is incorrect.</li> </ul>
	<p>Access to variable archive</p> <ul style="list-style-type: none"> <li>The Excel client tries to read the variable archive, but the PC name in the script call is wrong.</li> </ul>
	<p>Follow-up message Access to variable archive</p> <ul style="list-style-type: none"> <li>The Excel client tries to read the variable archive, but the PC name in the script call is wrong.</li> <li>The Excel client tries to read the variable archive, but the DataSourceName (DSN) in the script call is incorrect.</li> </ul>
	<p>Access to variable archive</p> <ul style="list-style-type: none"> <li>The Excel client tries to read the variable archive, but the DataSourceName (DSN) in the script call is incorrect.</li> </ul>

## 3.3 The “Universal” client

### 3.3.1 Principle of operation details

In this chapter you will find information that will make it easier for you to extend the "Alarms" tab of the existing application or to adapt it to your individual needs.

**Note** This chapter is only relevant to you if you are interested in the programmatic implementation or if you want to make changes to the application.  
Basic knowledge of VBA programming is required.

#### SQL command

This section will help you to understand the function of the SQL command for reading the data. If you have basic knowledge of VBA programming and SQL database language, you will be able to extend the application in various ways after reading this manual.

**Note** The SQL command used here contains a modification that is specifically required for communication with the WinCC OLE DB Provider. This causes the command to deviate partially from the specified standard.

#### Components of the SQL command

The explanation of the components of the SQL command will be based on an example and illustrate how the SQL command is composed in full length. The main focus of the description will be to show the variable parameters.  
In the following points, we will go into more detail on how these variable parameters are determined.

#### Example SQL

The following example shows an SQL command that uses all filter options of the application.

```
"ALARMVIEW: SELECT TOP 65000 * FROM AlgViewEnu WHERE DateTime > '2009-3-17 0:0:0' AND DateTime < '2009-3-17 23:59:59' AND (CLASS = 1 OR CLASS = 10) AND (Text2 = 'WEIGHT' OR (Text2 = '@2%s@' AND PText2 = 'WEIGHT')) ORDER BY DateTime ASC"
```



**Explanation of command elements:**

In order to better understand the function of the various elements of the command, they are now explained in detail.

Table 3-9

Command element	Explanation	Var <sup>1</sup>	Opt <sup>2</sup>
ALARMVIEW:	The relevant database is the one for the alarm view (provider-specific addition).	(-)	(-)
SELECT *	All data fields (columns) in the table or view to be determined are searched for.	(-)	-
TOP 65000	The number of messages returned by the query is 65000.	(-)	(-)
FROM AlgViewEnu	The data records are in the view „AlgViewEnu“. This name in turn contains several pieces of information: <ul style="list-style-type: none"> <li>• General: Prefix for AlarmLogging</li> <li>• View: It is a view (not a table).</li> <li>• Enu: These are the alarms of the English project planning.</li> </ul> <b>Note:</b> Unlike a table, a view does not contain any data records. It simply compiles the desired information by referring to different tables.	x	-
WHERE DateTime > '2009-3-17 0:0:0' AND DateTime < '2009-3-17 23:59:59'	The messages searched for occurred in the specified period (UTC). <b>Note:</b> The conversion from local time to UTC and back is done automatically by the application.	x	(-)
AND (CLASS = 1 OR CLASS = 10)	Messages of the type "Alarm" (coded as class 1 within the database) and "Operating message" (class 10) are searched for.	x	x
AND (Text2 = 'WEIGHT' ...)	Only those messages will be returned that were created in the plant area (internally stored directly under "Text2" or referenced from there to the accompanying value "PText2") with the configured name "WEIGHT".	x	x
ORDER BY DateTime ASC	The messages are sorted in ascending order according to their date of origin.	(-)	(-)

<sup>1</sup> Does the command element change depending on the settings made?

<sup>2</sup> Can the command element be omitted, depending on the settings made?

x – Yes

- – No

(-) – No (related to the application in unchanged form)

### Add reporting languages

In this application, you can choose between archived notifications in five different languages. However, since WinCC is used worldwide, it is conceivable that you may want to access messages in other languages.

This section describes how the language selection is currently being implemented and how you can adapt it to your requirements if necessary.

It cannot be guaranteed that the procedure described below will be successful in every case. This depends above all on whether the name and structure of the view belonging to the language match the structure of the languages already implemented.

### Overview

To change the message language, the user proceeds as described under "[Operation - Edit basic settings](#)". The language is selected via the "Language" drop-down list, which the application queries when creating the SQL command.

The following table gives an overview of the currently available languages, the corresponding entries in the drop-down list and the names of the respective views.

Table 3-10

Language	Drop-down list entry	Name of view
German	German	AlgViewDeu
English	English	AlgViewEnu
French	Français	AlgViewFra
Italian	Italiano	AlgViewIta
Spanish	Español	AlgViewEsp

### Explanation

As you can see from the above overview, the names of the views differ only in their last three letters, which are used to define the reporting language.

Furthermore, it becomes clear that in almost all cases the first three letters of the drop-down list entries correspond to the last three letters of the view names.

#### Procedure

The following code is used to create the SQL command in the required form (VBAProject > Microsoft Excel Objects > Sheet2 > cmdAlarmRead\_Click):

```
'Select all columns
strSQL = "ALARMVIEW: SELECT * FROM AlgView"
'Select the table according to the project language
If Left(cmbAlarmLang.Text, 3) = "Eng" Then
    strSQL = strSQL & "Enu"
Else
    strSQL = strSQL & Left(cmbAlarmLang.Text, 3)
End If
```

After assigning the constant part of the SQL command, the system checks whether the selected entry in the drop-down list begins with "Eng". If this is the case, "Enu" is appended to the command. Otherwise, the first three letters of the drop-down list entry are appended.

### Add Language

To add a reporting language to the existing selection, proceed as follows:

Table 3-11

	Step	Description
1.	Expanding the drop-down list	To expand the drop-down list, edit the initialization instructions in the VBAProject > Microsoft Excel Objects > ThisWorkBook > Workbook_Open module. Add another message language to the existing command lines in the form <code>cmbAlarm Lang.AddItem "English"</code> .
2.	Checking the identifiers	Use your SQL Server Management Studio to determine the name of the view belonging to this language.
3.	Case 1: No necessary changes	If the first three letters of your drop-down list entry match the last three letters of the view name, no further changes are required.
4.	Case 2: Extending the If Statement	If the letters differ, add a suitable Elself element to the If statement in the above code excerpt.

### Adding message types

In the present version, the application is able to filter the archived messages according to up to six different message types.

In addition to these six most common species, there are seven more in WinCC. It is also possible to create your own reporting types.

It may therefore be necessary to add these to the filter.

As in the previous section, the current conversion is presented here first, in order to show a possible change afterwards.

### Overview

The use of the filter for message types is described in detail under ["Operation - Edit filter settings"](#). The option boxes are used to select the individual types and add them to the SQL command when it is created.

#### Procedure

The following code implements the required extension of the SQL command "VBAProject > Microsoft Excel Objects > Sheet2 > cmdAlarmRead\_Click":

```
'Apply filter
If chkAlarmAlarms.Value = True Or _
  chkAlarmWarnings.Value = True Or _
  chkAlarmASPLCMessages.Value = True Or _
  chkAlarmOSPLCMessages.Value = True Or _
  chkAlarmOperatingMessages.Value = True Or _
  chkAlarmSystemMessages.Value = True Then

  'Flag to signalize that a part of the filter has
  already been entered
  'Depending on the flag, further parts of the filter
  will be added in a different format
  Dim boolIsFirst As Boolean
  boolIsFirst = True

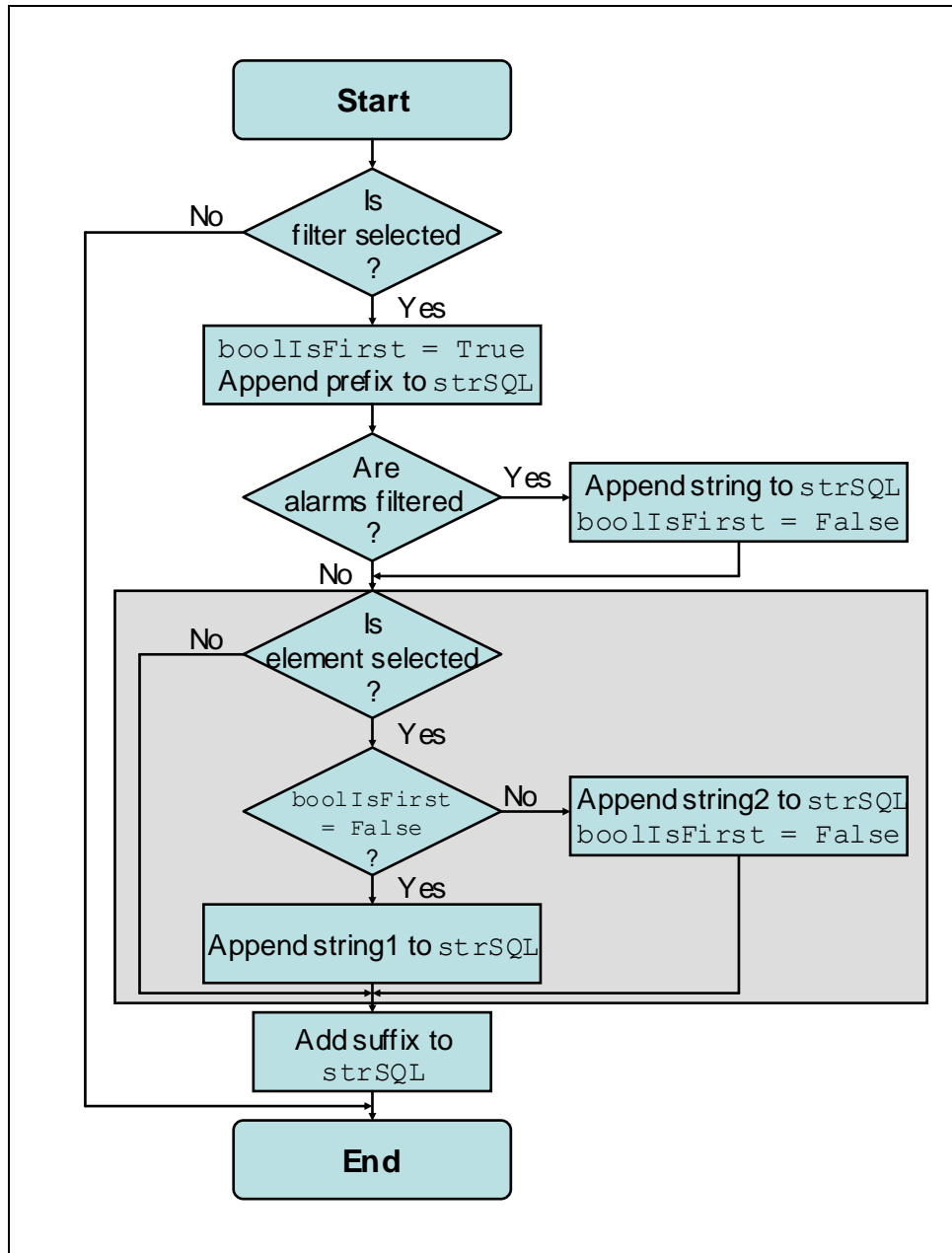
  'If EventCategories are filtered a prefix is
  necessary
  strSQL = strSQL & "AND ("

  If chkAlarmAlarms.Value = True Then
    strSQL = strSQL & "CLASS = 1"
    boolIsFirst = False
  End If
  If chkAlarmWarnings.Value = True Then
    If Not boolIsFirst Then
      strSQL = strSQL & " OR CLASS = 2"
    Else
      strSQL = strSQL & "CLASS = 2"
      boolIsFirst = False
    End If
  End If
  (...)
  If chkAlarmSystemMessages.Value = True Then
    If Not boolIsFirst Then
      strSQL = strSQL & " OR CLASS = 18"
    Else
      strSQL = strSQL & "CLASS = 18"
      boolIsFirst = False
    End If
  End If

  'If EventCategories are filtered a suffix is
  necessary
  strSQL = strSQL & ") "
End If
```

The Fig. 3-9 describes the structure of the above code excerpt. The grey area is executed once in the application for each filterable message type, with the exception of alarms. In order to improve the clarity, however, this section is only included representative once.

Fig. 3-9



### Add Language

To extend the filter elements by another message type, follow the instructions in the table below:

Table 3-12

	Step	Description
1.	Remove sheet protection	Before you can insert the control required for the extension, you must remove Excel's sheet protection. This is version-dependent and described in the Excel help.
2.	Add control element	<p>Add another one to the existing radio buttons and give it a representative name.</p> <p><b>Note:</b> To avoid layout problems, decouple the position and size properties of the control from the cell properties:</p> <ol style="list-style-type: none"> <li>1. Open the menu item "Format &gt; Control...".</li> <li>2. Switch to the "Properties" tab.</li> <li>3. Select „Dont move or size with cells“.</li> <li>4. Confirm the change.</li> </ol>
3.	Expanding the source code	<p>Extend the above code section by one element of the shape:</p> <pre> If chkAlarmSystemMessages.Value = True Then     If Not boolIsFirst Then         strSQL = strSQL &amp; " OR CLASS = 18"     Else         strSQL = strSQL &amp; "CLASS = 18"         boolIsFirst = False     End If End If                     </pre> <p>Change the name of the radio button and the number of the reporting type.</p> <p>Add the following to the parent query (If chkAlarmAlarms.Value = True Or ...) a corresponding condition (... Or chkAlarmSystemMessages.Value = True ...).</p> <p><b>Note:</b> You will find an overview of the coding of the message types used in the current project using SQL Server Management Studio.</p> <ol style="list-style-type: none"> <li>1. Select your WinCC Server when starting the application.</li> <li>2. Navigate to the following table (example): „Databases &gt; CC_OS_1__09_03_19_18_33_31 &gt; Tables &gt; dbo.MSMsgGroup“</li> <li>3. Take the corresponding coding from the column "GROUPDAT" in this table. The relevant entries are from "TYPE" 3.</li> </ol>
4.	Set sheet protection	Create sheet protection again. This is version-dependent and described in the Excel help.

#### **Expandability in message structure**

This application offers the possibility to display all message information (columns) that are also available under WinCC. However, the archived messages also contain further information such as "AlarmTag", "AckType" and "Servername", which are not available for selection with the present settings.

In order to facilitate understanding, the current implementation is first explained. On this basis, step-by-step instructions are given on how to adjust the message structure.

#### **Overview**

The operation for adjusting the notification structure is described in detail under "[Operation - Adjust notification structure](#)". The settings are made in a separate window. After confirming the changes, the workspace is automatically adapted to the application.



**Procedure**

The message structure influences the application in various areas. For the extension it is only important to know how the necessary basic information is initialized. All further processes adapt to the respective settings (as long as they correspond to the program logic).

Three arrays form the basis for the message structure:

Table 3-13

Array	Contents
strAlarmDataNames	<p>Contains the names that are displayed to the user for selection. The current entries correspond to the names that are also used when editing the WinCC message structure in the Graphics Designer.</p> <p><b>Note:</b> These names are different from those used in the database. They serve to enable the user to understand them as intuitively as possible.</p>
intAlarmRecordSetID	<p>Contains the position of each message information within the record set.</p> <p><b>Note:</b> After the SQL command messages have been requested, they are returned as a record set. The structure of a record set is always the same as long as neither the data source (view) nor the queried structure (SELECT *) changes. Due to this fact, every piece of information in this record set has a fixed position.</p>
dblAlarmColumnWidth	<p>Contains the column width that is assigned to reporting information by default.</p> <p><b>Note:</b> When adjusting the reporting structure, the width of the assigned column is automatically adjusted for each selected reporting information. (see <a href="#">"Operation - Adapt message structure"</a>). This is done to ensure that the messages are displayed as optimally as possible.</p>

The declaration of the arrays is made with the help of a constant, so that a uniform structure can be ensured "VBA-Project > Modules > MngAlarmPositions":

```
Public strAlarmDataNames(1 To MAX_ALARM_DATA_COLUMNS) As String
```

This constant corresponds to the number of reporting information, which is 38 by default.

The arrays are addressed, e.g. during their initialization, via further constants, each of which corresponds to a message information "VBA-Project > Modules > MngAlarmPositions > InitializeAlarmArrays":

```
strAlarmDataNames(C_DATE_TIME) = "Date / Time"
```

This will improve the readability of the source code and simplify the process of making changes.

The constants for the message information contain consecutive numbers "VBA-Project > Modules > Constants" for shared constants or "VBA-Project > Modules > ConstantsAlarm" for alarm specific constants according to their function as index of the arrays:

```
Public Const C_DATE_TIME = 1
Public Const C_MS = 2
Public Const C_PRIORITY = 3
```

The standard information is arranged according to its order in the WinCC Graphics Designer.

### Expand message structure

To extend the message structure, you only need to adjust the constants and array initializations. Proceed as described in the table:

Table 3-14

	Step	Description
1.	Adjusting the maximum value constant	Under "VBA-Project > Modules > ConstantsAlarm" type enter the new number of reporting information for the constant MAX_ALARM_DATA_COLUMNS.  <b>Note:</b> The constant adapts the application to the changed conditions. A correct value for this constant is therefore essential.
2.	Adding the index constants	In the same module, add as many index constants as required to the existing index constants (for example, C_PRIORITY).  <b>Note:</b> Make sure that you assign the new numbers consecutively and that you do not exceed the value defined in MAX_ALARM_DATA_COLUMNS.
3.	Extending the Array initialization	The module "VBA-Project > Modules > MngPositions > InitializeArrays" initializes the presented arrays. Add as many elements to each of the three arrays as you want to add reporting information and fill them with the corresponding data (see Table 3-13).

### 3.3.2 Configuration and project planning

#### Storage Location

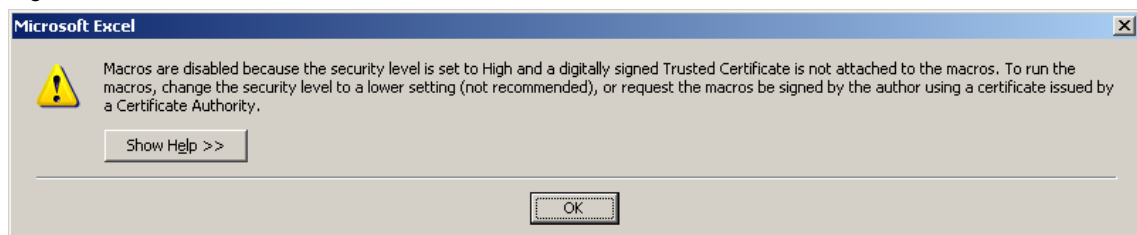
The application does not have to be installed separately. Copy the file "38132261\_Application\_Universal\_Client.xls" directly to the desired storage location. This Excel file contains the entire source code in the form of macros, which are executed in the background.

#### Application call up

Start the application by double-clicking on the Excel file.

As soon as the application has opened, the following warning may appear, depending on your Excel settings:

Figure 3-10

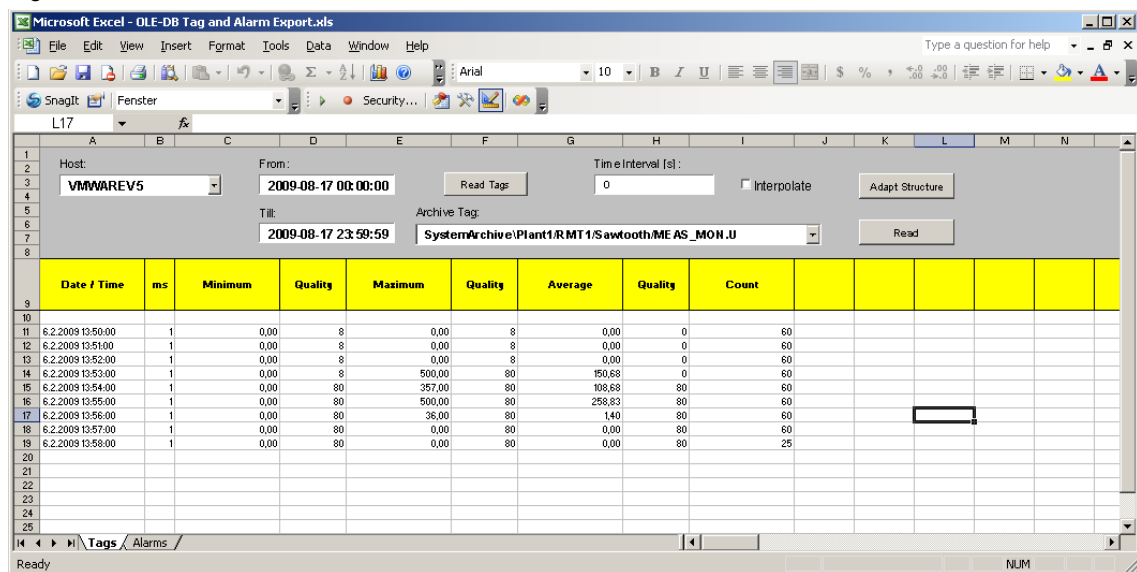


The reason for this is that your macro security in Excel is set to "High". Change this setting accordingly.

#### Application window setup

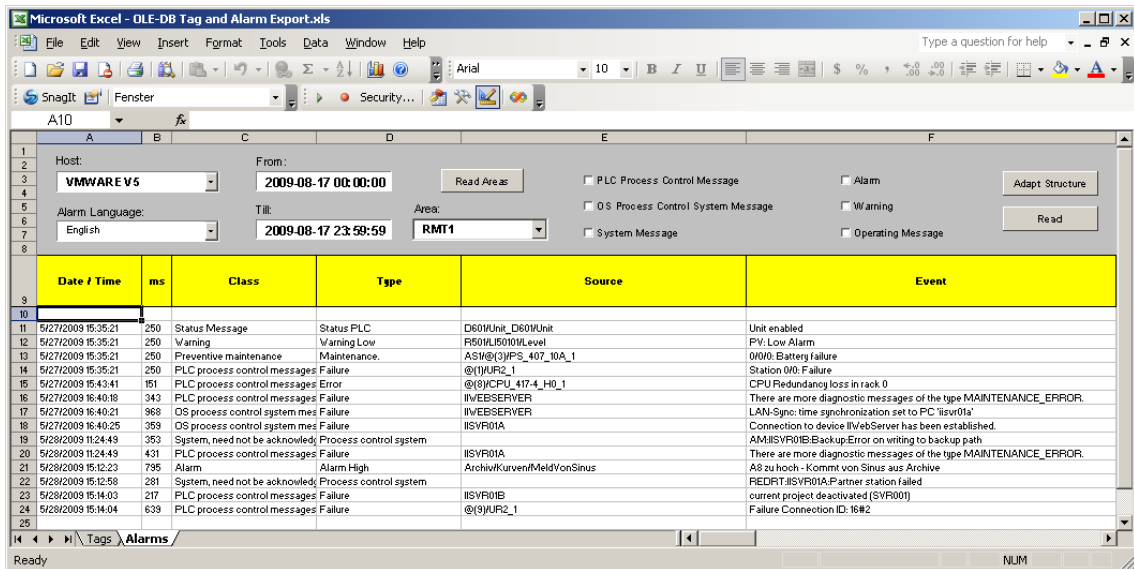
Figure 3-11 and Figure 3-12 show the application directly after its call. Which interface is displayed depends on which register was active during the last save operation. The entries are also those that were made when you last saved.

Figure 3-11



### 3 Archive data exporting with Excel

Figure 3-12



**Note**

The user interface is completely in English. The selection in the "Alarm Language" drop-down list in the "Alarms" tab refers to the language of the messages to be read out.

Switching between "Tags" and "Alarms" is done by selecting the corresponding tab at the bottom of the Excel window.

Figure 3-13



**Structure of the work area**

The structure of the work area is represented by the register "Alarm" as representative for both registers.

Figure 3-14

Date / Time	ms	Class	Type
5/27/2009 15:35:21	250	Status Message	Status PLC
5/27/2009 15:35:21	250	Warning	Warning Low
5/27/2009 15:35:21	250	Preventive maintenance	Maintenance.
5/27/2009 15:35:21	250	PLC process control messages	Failure
5/27/2009 15:43:41	151	PLC process control messages	Error
5/27/2009 16:40:18	343	PLC process control messages	Failure
5/27/2009 16:40:21	968	OS process control system mes	Failure
5/27/2009 16:40:25	359	OS process control system mes	Failure
5/28/2009 11:24:49	353	System, need not be acknowledg	Process control system
5/28/2009 11:24:49	431	PLC process control messages	Failure
5/28/2009 15:12:23	795	Alarm	Alarm High

Each of the three selected areas has its own function, as shown in the Table 3-15.

Table 3-15

Area	Function
Control bar	They make all the necessary settings via the control bar. This includes the selection of the data source and the definition of the message or process value filter. You can find a more detailed description of this area under <a href="#">"Configuration and project engineering - Structure of the operating bar"</a> .
Column headers	The column headings determine the structure of the displayed messages. <b>Note:</b> You can adjust both the sequence and the composition according to your requirements. The exact procedure is described under <a href="#">"Operation - Adjust message structure"</a> .
Messages (Tab „Tags“: Process values)	This area contains the read messages or process values with the structure you selected.

**Note** The fields of the "Toolbar" and "Column headings" areas are protected to avoid unwanted changes. It is recommended not to remove this protection in order not to impair the functionality of the application.

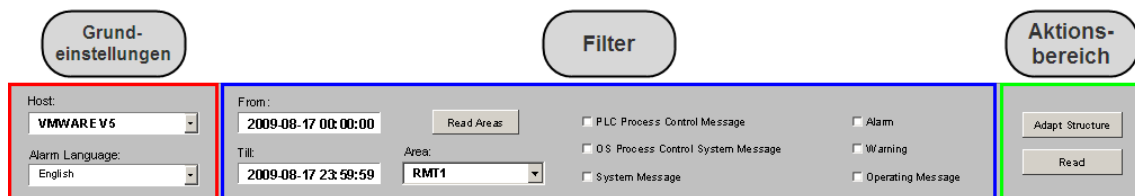
#### Design of the operating elements

In the control bar, you make all the settings necessary to display the data you want in a format of your choice. The structure of the control bar can be divided into three areas (for example, see Figure 3-15 for the "Alarms displayed" tab):

- The basic settings which, once set, in most cases remain the same for the entire session.
- The filter, which makes it possible to describe the searched archive data more exactly and thus to improve the search result.
- The action area from which the structural adjustment of the displayed messages (selection of message blocks) or process values (selection of statistical values) and the actual read operation can be started.

Each of these areas in turn contains several control elements, which are described in more detail in the section [3.3.3](#) on operating the application.

Figure 3-15



#### The Table View

The display of the received archive data is variable and allows you to define an optimal structure for you. For process values, for example, various statistical functions can be executed and their results displayed.

The table view for displaying the messages is based on the chronicle list of the WinCC message system. The display of the different message blocks can be designed individually, similar to the original (see ["Operation - Adapt message structure"](#)).

### 3.3.3 Operation

This section describes how process values and messages can be read from the archives. With the exception of the explanation of the filter, this will be done using the procedure for messages as an example, since the differences in the handling of process values are very small.

#### Application call up

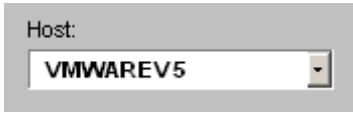
Table 3-16

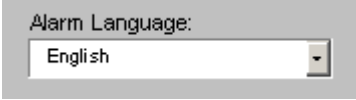
No.	Description
1.	<p>Start the application.</p> <p><b>Note:</b> After the start of the application it is possible that this short time does not react. This is because the network is busy and the list of available computers is progressing slowly.</p> <p>If there are several subnets, only members of the own workgroup are determined. However, you can also access computers that are not displayed as long as there is a valid connection to them. All you have to do is enter the name of the target computer in the text field of the "Host" drop-down list.</p>
2.	If necessary, activate the macros.

#### Edit basic settings

In the basic settings, you make settings that in most cases remain the same for the entire session. This includes defining the source computer and the reporting language.

Table 3-17

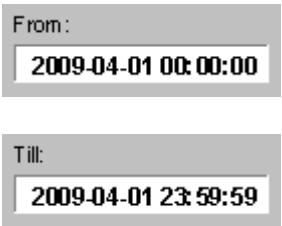
	Description	control element
1.	Select the "Alarms" tab.	
2.	<p>In the "Host" drop-down list, select the computer that serves as the information source. After the program start, the name of the calling computer is entered.</p> <p><b>Note:</b> It is necessary that the WinCC runtime is started on the computer specified here, since the WinCC OLEDB provider communicates with their services.</p> <p>In addition, it should be noted that only the data locally available on this computer is read. If you are using a CAS (Central Archive Server), you must choose between this (long-term archive) and the OS server (short-term archive) in order to receive messages for a certain time period.</p>	

	Description	control element
3.	<p>In the drop-down list, select the language in which the messages to be exported were configured.</p> <p>The following languages are available:</p> <ul style="list-style-type: none"> <li>• German</li> <li>• English</li> <li>• French</li> <li>• Italian</li> <li>• Spanish</li> </ul> <p><b>Note:</b> The language selection is not available for process values.</p> <p>If one of these reporting languages is not planned separately, the messages in this language contain standard texts.</p>	

### Edit filter settings

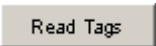

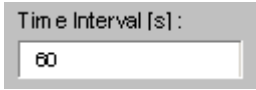

You can use the filter settings to reduce the archive data returned to those that really interest you with regard to day, time period and time interval (process values) or time, type and origin (messages). Differences between the procedure for process values and messages are explained in parallel steps.

Table 3-18

	Description	control element
1.	<p>In the two input fields, specify the time period in which the archive data you are searching for was created.</p> <p>Make sure that the start time is before the end time.</p> <p>By default, the current date is set from 0 - 24 o'clock.</p> <p><b>Note:</b> The time and date must be entered in a fixed form: YYYY-MM-DD hh:mm:ss</p> <p>In addition, it must be ensured that the input is logical in itself (e.g. no 30 February, no 70 minutes, etc.). If the value is invalid when leaving the input field, a warning will appear and the previously entered value will be restored.</p>	



### 3 Archive data exporting with Excel

	Description	control element
2.	<p>For the tags tab: To read out the archived process values (tags) or the configured (plant) areas, click the button „Read Tags“ / „Areas“.</p> <p><b>Note:</b> In order to perform the readout, the corresponding host must have been selected beforehand.</p> <p><b>Error handling:</b> If you receive the following message, communication with the database has failed: "Area" / "Tag determination failed" + error description. Possible causes are:</p> <ul style="list-style-type: none"> <li>• Network problems (host not reachable)</li> <li>• Missing rights (same user and same passwords required)</li> <li>• OS not in runtime</li> </ul> <p>As soon as you have read the information, the corresponding drop-down list makes it available to you.</p>	
3.	<p><b>For the tags tab:</b> In the "Archive Tags" drop-down list, select the process value whose archived values you want to read out.</p>	
4.	<p><b>For the tags tab:</b> Enter the desired time interval in seconds over which the originally occurred values ("native") are to be summarized. If no interval is to be used (values remain "native"), enter "0".</p> <p><b>Note:</b> The time interval can be used to reduce the number of process values read without losing much information. The time interval allows you to combine all process values that have occurred in a particular time grid (e.g. all values in a minute or hour). Evaluations can then be made for this set of values (for example, maximum value, minimum value, total). This principle is illustrated in the Figure 3-16.</p> <p><b>For the "Alarms" tab:</b> To filter messages according to different types (classes), you have a total of six option boxes at your disposal, which you can combine as you wish. If you do not make a selection here, filtering by messages is deactivated.</p>	  

	Description	control element
5.	<p><b>For the tags tab:</b> Select whether an interpolation is to take place.</p> <p><b>Note:</b> Interpolation requires a time interval other than zero.</p> <p>Interpolation ensures that the time interval between two received values always remains the same. If no value has been archived for a point in time, a corresponding intermediate value is determined from the directly preceding and the directly following archived value.</p> <p>Interpolation only takes place between two archived values within the time period. There is no interpolation between a value within and a value outside the time domain.</p>	<input type="checkbox"/> Interpolate

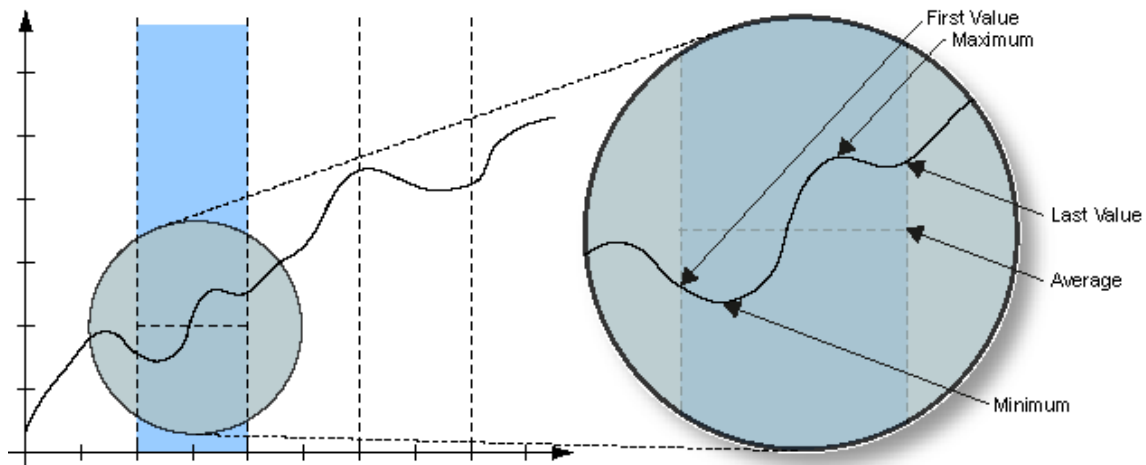
**Note** You should use the filter options to limit the selection criteria as far as possible. This reduces the amount of relevant archive data and thus the access time to the database and the subsequent processing time.

For messages, the number of data records returned is additionally limited to 65000 by the SQL string.

**Explanation of the time interval**

The following figure shows the course of a process value over several hours. With the help of the vertical dashed lines, the curve is divided into the individual hours.

Figure 3-16



Assuming that the process values were archived every second, the listing of up to 18000 values (for 5 hours) would be quite confusing. If, however, you specify a time interval of one hour, only a maximum of five values per column are specified.

In order to still get a good idea of the actual curve, it is possible to obtain various information about the number of values affected. This is illustrated in the detail view of the illustration. It is possible to determine the extreme values of the curve, the first and last values, the average and (not shown) the sum of all the values occurring in this interval and their number.

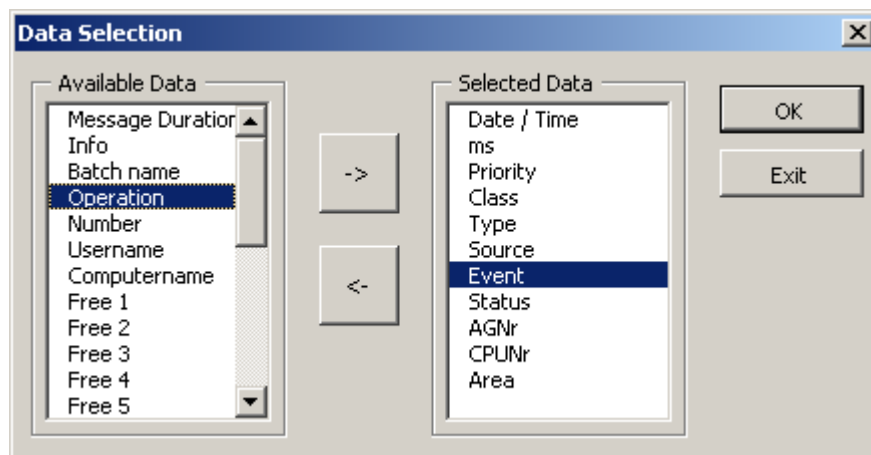
If an interval according to these criteria shows a particularly conspicuous behaviour, you can then limit the time range accordingly and reduce the time interval so that all information is available to investigate the cause exactly.

#### Adapt message structure

Adjusting the message structure (selection of message blocks) allows you to display the received messages individually in a way that is optimal for you.

If the message structure already corresponds to your wishes, you can skip this point.

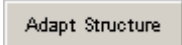
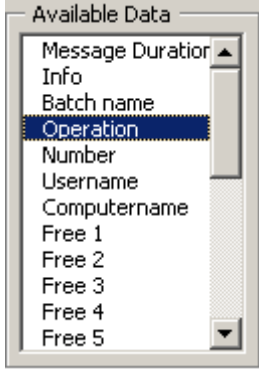
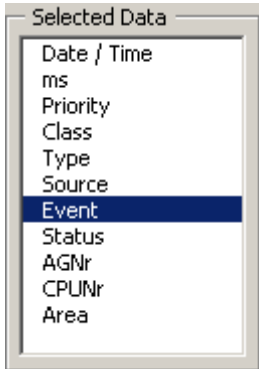
Figure 3-17



### 3 Archive data exporting with Excel

The following table describes how you can adapt the notification structure to your individual requirements.

Table 3-19

	Description	control element
1.	Click the button „Adapt Structure“. The Data Selection dialog box opens.	
2.	In the "Available Data" list on the left, select the element you want to add to the selection and click the button „->“. New elements of the right list "Selected Data" are always inserted after the currently selected element in this list.	
3.	To remove an element from the selection, select it from the list on the right and click on the button „<-“. Removed elements are added at the bottom of the left list.	
4.	Confirm your settings by clicking "OK". To discard the changes, click on the button „Exit“.  <b>Note:</b> The adjustment of the data structure leads to the deletion of all currently listed messages. There is no additional query after confirmation of the changes.	
5.	After you confirm the changes, all column headings and messages are deleted. The new column headings are then entered and the column widths adjusted.  <b>Note:</b> If you have not selected any columns for display, the subsequent read operation is aborted prematurely. The text "No Data to be read" is displayed in the status line.	

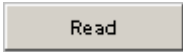
**Note** When the application is started, it automatically determines the last used settings for the message structure from the entries in the "Column headings" subarea of the workspace.

When selecting the information for process values, make sure that each piece of information other than the time stamp and the quality code triggers its own read operation. The reason for this is the functionality of the associated SQL query.

**Read messages**

After you have made all the settings, you can now read the required messages from the archive.

Table 3-20

	Description	control element
1.	<p>Click the "Read" button to read the messages corresponding to the filter from the archive.</p> <p><b>Caution:</b> Pressing the "Read" button deletes all currently listed messages. There is no additional query.</p> <p><b>Note:</b> You can end the listing of messages prematurely by pressing the ESC key.</p>	
2.	If necessary, save the read data separately.	

**Note** After triggering the read command, the SQL query is first created and executed. During this time, the application is not reactive because it has to wait for a response from the SQL server. Depending on the number of requested messages, this can take up to a few minutes.

The formatting (replacement of the placeholders) and listing of the individual messages is then carried out. During this process, updating the workspace is disabled to increase message throughput and the status bar displays the progress of the process.

### 3.3.4 Error handling

All error categories that can be assigned to the read operation are listed below:

- Connection error
  - Process:  
Connect with the database
  - Message:  
"Error during Connect" + error description
  - Possible causes:  
Network problems (host not reachable); missing rights; missing installation of the WinCC OLE DB provider on the client
  
- Read error
  - Process:  
Opening the database and reading the messages
  - Message:  
"Error during Open" + error description
  - Possible cause:  
WinCC is not in runtime on the server.
  
- Processing error
  - Process:  
Arranging and interpreting the message blocks; entering the messages
  - Message:  
"Error during Processing" + error description
  - Possible cause:  
The structure of the database and therefore of the messages returned has changed.

## 4 Export archive data with SQL Server

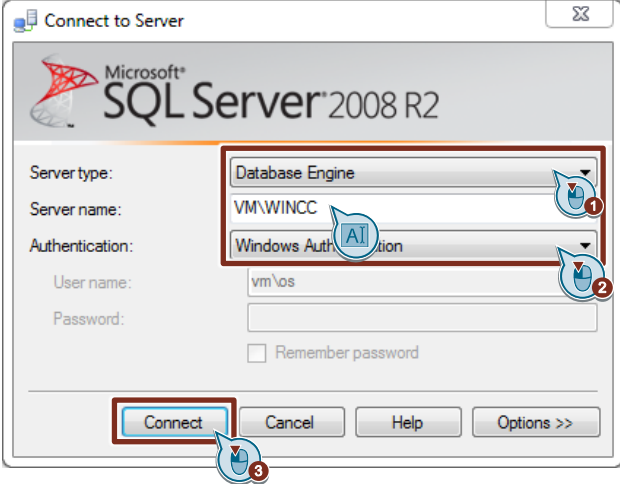
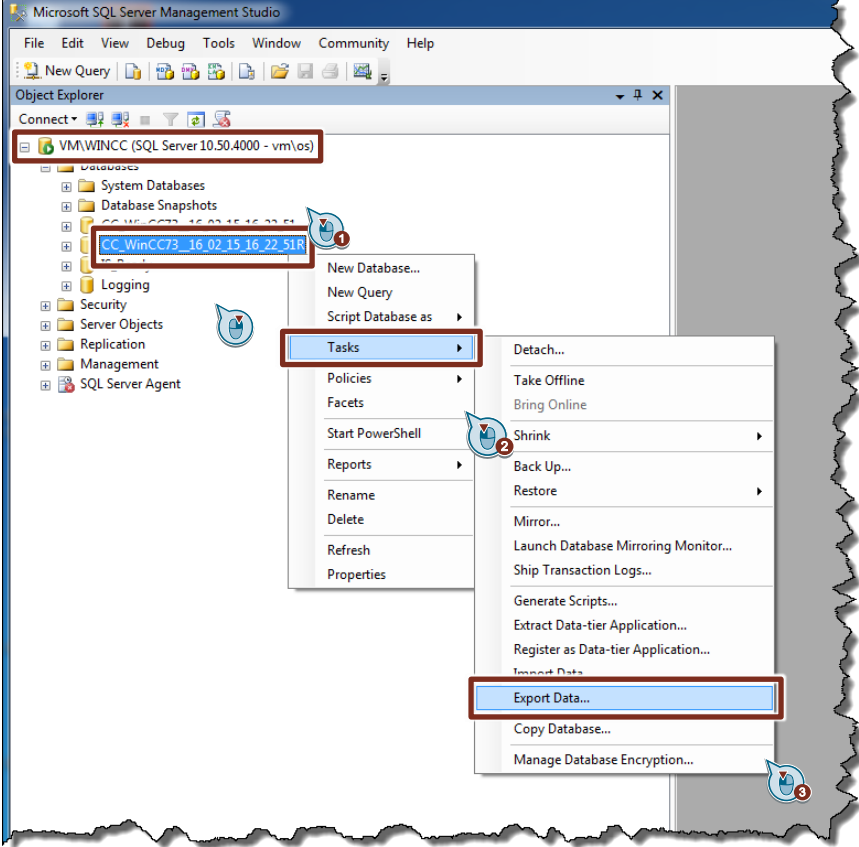
### 4.1 Configuration

- The export of archive values from SQL is parameterized directly in the "SQL Server Management Studio" with the help of the "SQL Server Import/Export Wizard". It is triggered after completion of the parameterization from the wizard.
- It is also possible to save the created parameterization as an SSIS package (.DTSX file) in the Windows file system. The file allows the export at a later time. The SQL query and thus the scope of the data to be exported is stored in the SSIS package.
- When accessing WinCC archive data, the source must be the WinCC OLE DB provider, e.g. a newly created, separate SQL database or a "\*.xls", "\*.xlsx", "\*.txt" or "\*.csv" file can serve as the data target.
- The following instructions describe the parameterization of the "SQL Server Import/Export Wizard" for exporting archived measured values to a "\*.csv" file.
- The following instructions apply to SQL Server 2008 R2.

Table 4-1

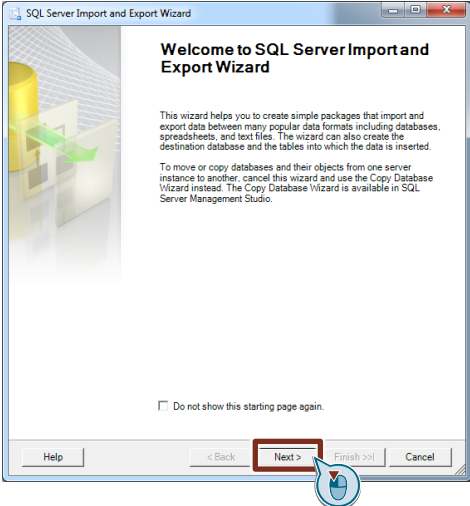
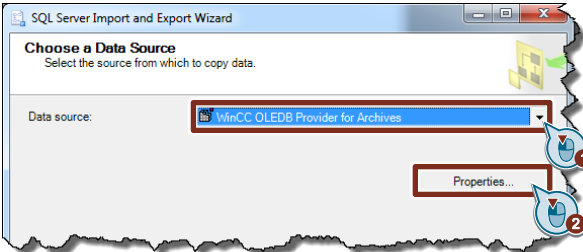
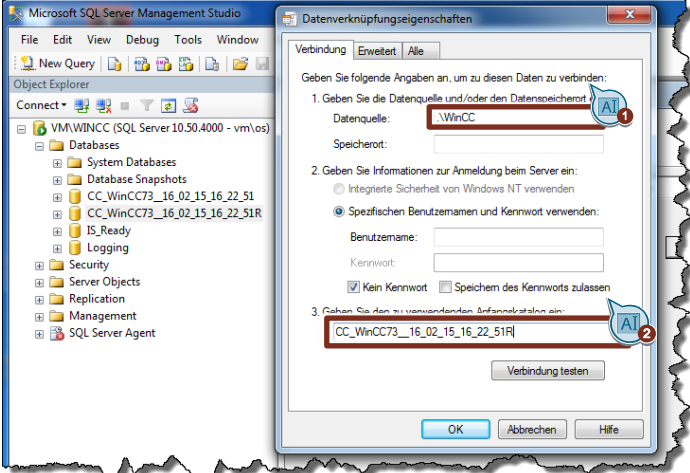
No.	Action
1.	<ul style="list-style-type: none"> <li>• Start WinCC-Explorer. Open the WinCC project. Enable runtime.</li> <li>• Click the "Reverse Osmosis" button to switch to the image „ReverseOsmosis.Pdl“.</li> <li>• Start the reverse osmosis system via the "START UP" button. Wait until the plant has reached the status "Production".</li> <li>• After a few minutes, you can shut down the system by clicking on the "DEPARTURE" button.</li> <li>• Create an empty file "TagsToCSV.csv" and "AlarmsToCSV.csv" in the directory of your choice.</li> </ul>
2.	Start the "SQL Server Management Studio" under "Start > Programs > Microsoft SQL Server".

#### 4 Export archive data with SQL Server

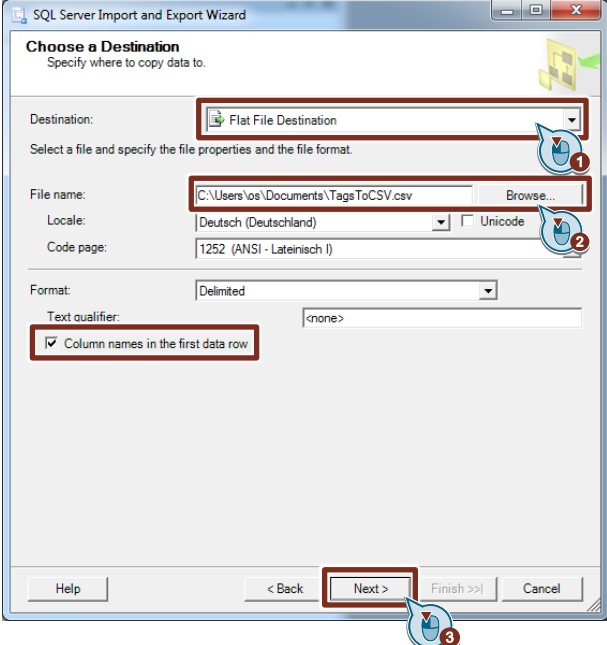
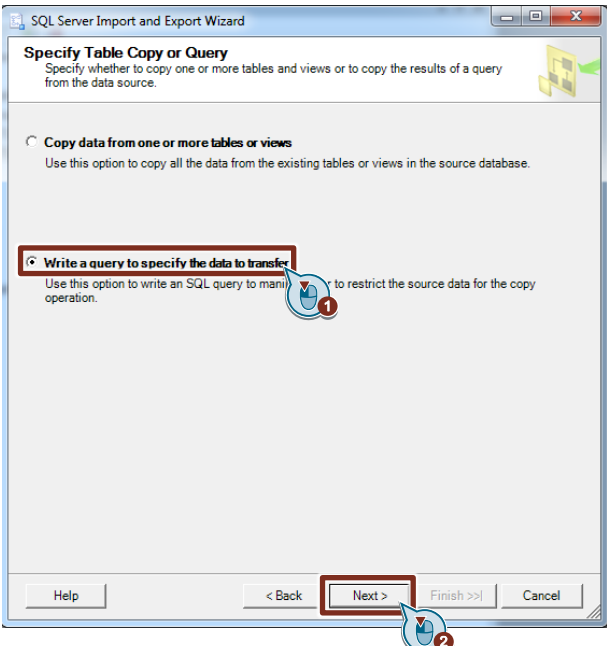
No.	Action
3.	<ul style="list-style-type: none"> <li>Log in.</li> <li>Select "Database Engine" as server type, "&lt;Your Computer Name&gt;\WINCC" as server name and "Windows Authentication" as Authentication (your current user is used).</li> <li>Connect to the SQL Server via the "Connect" button.</li> </ul> 
4.	<ul style="list-style-type: none"> <li>Right-click the runtime and archive data database. To do this, use either             <ul style="list-style-type: none"> <li>- „Databases &gt; CC_WinCC73_16_02_15_16_22_51R“, or</li> <li>- „Databases &gt; CC_ExternalBrowsing“.</li> </ul> </li> <li>Start the SQL Server Import and Export Wizard with "Tasks &gt; Export Data".</li> </ul> 

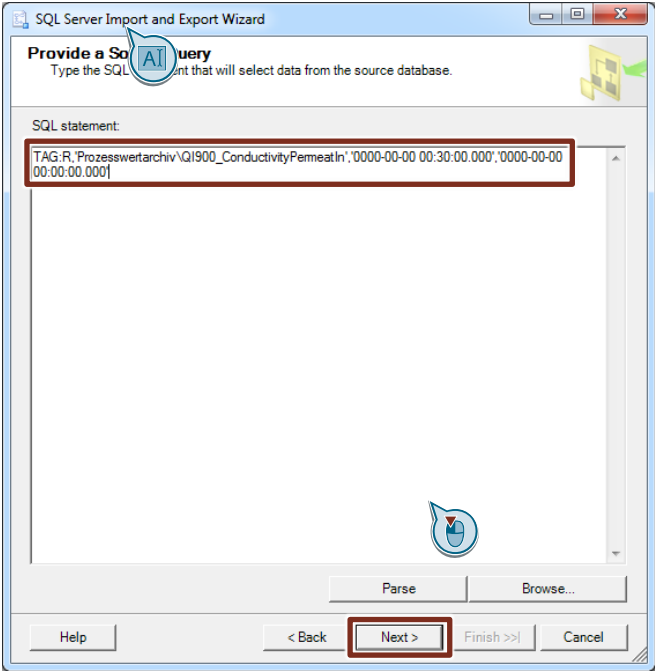


## 4 Export archive data with SQL Server

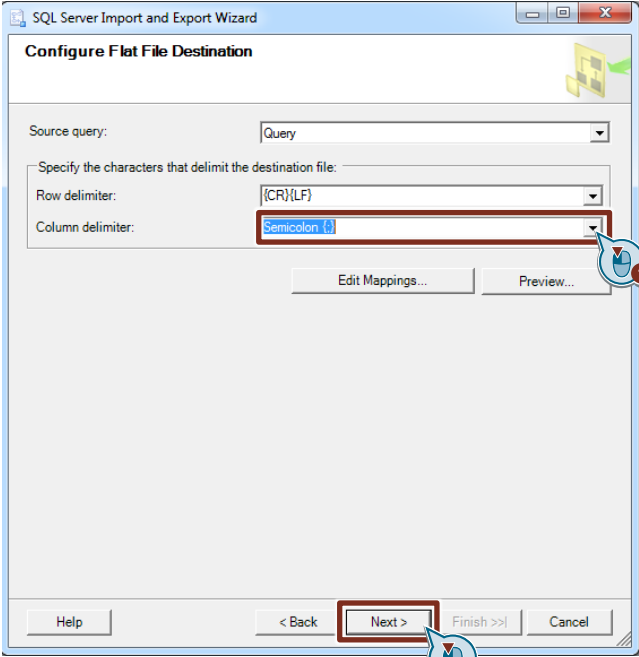
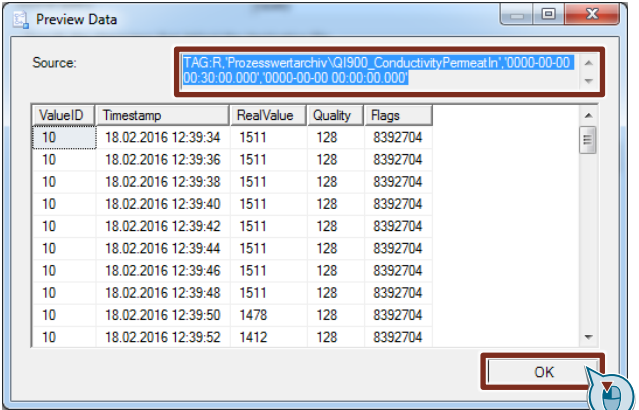
No.	Action
5.	<p>Click on the „Next“ button</p> 
6.	<p>Select "WinCC OLE DB-Provider for Archives" as data source and then click on the "Properties" button.</p> 
7.	<ul style="list-style-type: none"> <li>• Enter ".\WinCC" as the data source and the name of the WinCC runtime database as the catalog.</li> <li>• As the WinCC Runtime database, enter either             <ul style="list-style-type: none"> <li>- „CC_WinCC73__16_02_15_16_22_51R“, or</li> <li>- „CC_ExternalBrowsing“.</li> </ul> </li> <li>• Then exit the masks with the "OK" and "Next" buttons.</li> </ul> 

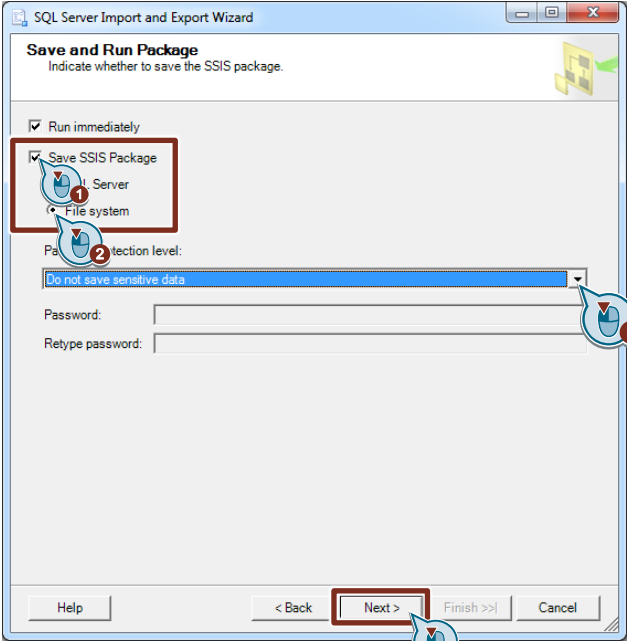
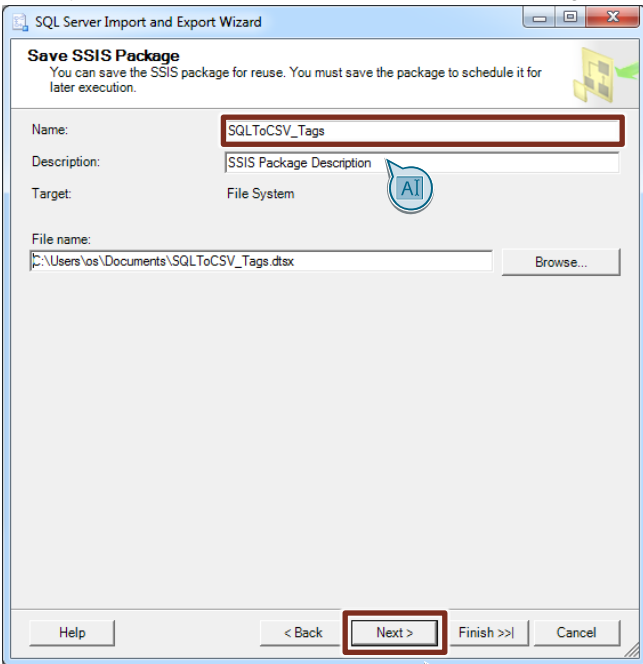
## 4 Export archive data with SQL Server

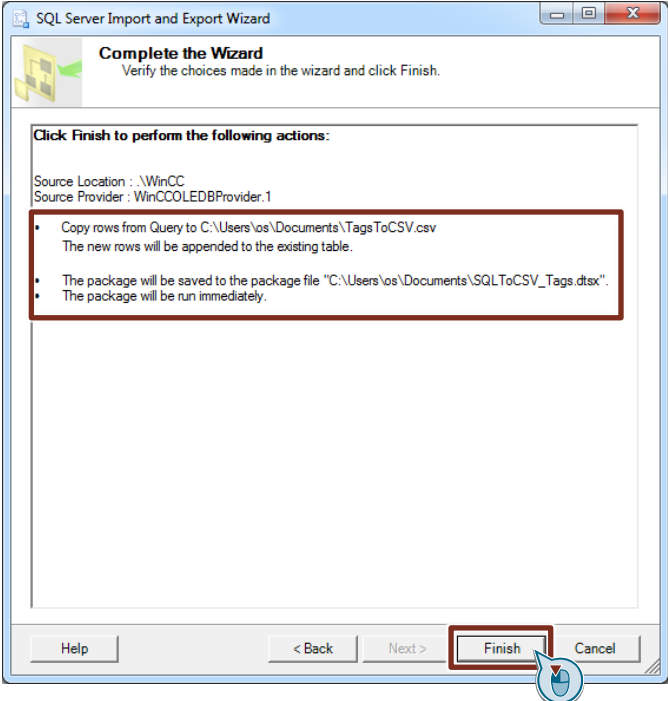
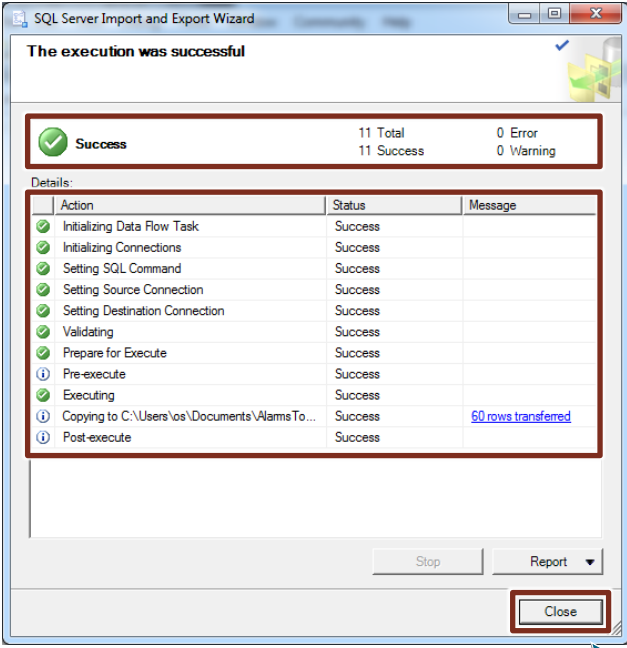
No.	Action
8.	<ul style="list-style-type: none"> <li>• Select "Flat File Destination" as data destination and the file "TagsToCSV.csv" (for variable export).</li> <li>• Activate the checkbox "Column names in first data row" to write the column headers to the first line of the "*.csv" file.</li> <li>• Exit the mask with the "Next" button.</li> </ul> 
9.	<ul style="list-style-type: none"> <li>• Activate the option "Write a query to specify the data to transfer" to specify the SQL query in the next mask.</li> <li>• Continue the wizard via the "Next" button.</li> </ul> 

No.	Action
10.	<ul style="list-style-type: none"> <li>Enter the desired SQL query here. The structure of the query corresponds to the description in chapter 2.2.3 or chapter 2.2.4.</li> <li>In this example, the archive tag "QI900_ConductivityPearnatIn" from the process value archive from the period of the last 30 minutes is queried:   <pre> TAG:R,'Process value archive\QI900_ConductivityPearnatIn','0000-00-00 00:30:00.000', '0000-00-00 00:00:00.000'"""                     </pre> </li> </ul> <p><b>Notice</b> When entering the text string, pay attention to the American spelling for quotation marks (see picture).</p> <ul style="list-style-type: none"> <li>Exit the mask via the "Next" button.</li> </ul> 

## 4 Export archive data with SQL Server

No.	Action																																																							
11.	<p>Set the column delimiter "Column delimiter" to semicolon.</p> 																																																							
12.	<ul style="list-style-type: none"> <li>You can check the result of the query via "Preview".</li> <li>Close the dialog with the "OK" button.</li> </ul> <p>Preview variable QI900:</p>  <table border="1" data-bbox="496 1200 903 1447"> <thead> <tr> <th>ValueID</th> <th>Timestamp</th> <th>RealValue</th> <th>Quality</th> <th>Flags</th> </tr> </thead> <tbody> <tr><td>10</td><td>18.02.2016 12:39:34</td><td>1511</td><td>128</td><td>8392704</td></tr> <tr><td>10</td><td>18.02.2016 12:39:36</td><td>1511</td><td>128</td><td>8392704</td></tr> <tr><td>10</td><td>18.02.2016 12:39:38</td><td>1511</td><td>128</td><td>8392704</td></tr> <tr><td>10</td><td>18.02.2016 12:39:40</td><td>1511</td><td>128</td><td>8392704</td></tr> <tr><td>10</td><td>18.02.2016 12:39:42</td><td>1511</td><td>128</td><td>8392704</td></tr> <tr><td>10</td><td>18.02.2016 12:39:44</td><td>1511</td><td>128</td><td>8392704</td></tr> <tr><td>10</td><td>18.02.2016 12:39:46</td><td>1511</td><td>128</td><td>8392704</td></tr> <tr><td>10</td><td>18.02.2016 12:39:48</td><td>1511</td><td>128</td><td>8392704</td></tr> <tr><td>10</td><td>18.02.2016 12:39:50</td><td>1478</td><td>128</td><td>8392704</td></tr> <tr><td>10</td><td>18.02.2016 12:39:52</td><td>1412</td><td>128</td><td>8392704</td></tr> </tbody> </table>	ValueID	Timestamp	RealValue	Quality	Flags	10	18.02.2016 12:39:34	1511	128	8392704	10	18.02.2016 12:39:36	1511	128	8392704	10	18.02.2016 12:39:38	1511	128	8392704	10	18.02.2016 12:39:40	1511	128	8392704	10	18.02.2016 12:39:42	1511	128	8392704	10	18.02.2016 12:39:44	1511	128	8392704	10	18.02.2016 12:39:46	1511	128	8392704	10	18.02.2016 12:39:48	1511	128	8392704	10	18.02.2016 12:39:50	1478	128	8392704	10	18.02.2016 12:39:52	1412	128	8392704
ValueID	Timestamp	RealValue	Quality	Flags																																																				
10	18.02.2016 12:39:34	1511	128	8392704																																																				
10	18.02.2016 12:39:36	1511	128	8392704																																																				
10	18.02.2016 12:39:38	1511	128	8392704																																																				
10	18.02.2016 12:39:40	1511	128	8392704																																																				
10	18.02.2016 12:39:42	1511	128	8392704																																																				
10	18.02.2016 12:39:44	1511	128	8392704																																																				
10	18.02.2016 12:39:46	1511	128	8392704																																																				
10	18.02.2016 12:39:48	1511	128	8392704																																																				
10	18.02.2016 12:39:50	1478	128	8392704																																																				
10	18.02.2016 12:39:52	1412	128	8392704																																																				

No.	Action
13.	<p>Then activate the saving of the "SSIS package" in the Windows file system without protective function and exit the mask with "Next".</p>  <p><b>Note</b> No protective function is required for the example project. In productive operation, this should be set up, depending on the application.</p>
14.	<p>Specify the file name and location for the SSIS package and click the Next button.</p> 

No.	Action																																				
15.	<p>The set parameters are summarized in the next screen. With the button "Finish" the export is started.</p>  <p><b>SQL Server Import and Export Wizard</b>  <b>Complete the Wizard</b>          Verify the choices made in the wizard and click Finish.</p> <p><b>Click Finish to perform the following actions:</b></p> <p>Source Location : ..\WinCC          Source Provider : WinCCOLEDBProvider.1</p> <ul style="list-style-type: none"> <li>Copy rows from Query to C:\Users\os\Documents\TagsToCSV.csv              The new rows will be appended to the existing table.</li> <li>The package will be saved to the package file "C:\Users\os\Documents\SQLToCSV_Tags.dtsx".              The package will be run immediately.</li> </ul> <p>Buttons: Help, &lt; Back, Next &gt;, <b>Finish</b>, Cancel</p>																																				
16.	<ul style="list-style-type: none"> <li>The result of the export is finally logged.</li> <li>With the button "Reports" you have the possibility to save the log as a text file.</li> </ul>  <p><b>SQL Server Import and Export Wizard</b>  <b>The execution was successful</b></p> <p><b>Success</b> 11 Total 0 Error          11 Success 0 Warning</p> <p>Details:</p> <table border="1"> <thead> <tr> <th>Action</th> <th>Status</th> <th>Message</th> </tr> </thead> <tbody> <tr><td>Initializing Data Flow Task</td><td>Success</td><td></td></tr> <tr><td>Initializing Connections</td><td>Success</td><td></td></tr> <tr><td>Setting SQL Command</td><td>Success</td><td></td></tr> <tr><td>Setting Source Connection</td><td>Success</td><td></td></tr> <tr><td>Setting Destination Connection</td><td>Success</td><td></td></tr> <tr><td>Validating</td><td>Success</td><td></td></tr> <tr><td>Prepare for Execute</td><td>Success</td><td></td></tr> <tr><td>Pre-execute</td><td>Success</td><td></td></tr> <tr><td>Executing</td><td>Success</td><td></td></tr> <tr><td>Copying to C:\Users\os\Documents\Alarms To...</td><td>Success</td><td>60 rows transferred</td></tr> <tr><td>Post-execute</td><td>Success</td><td></td></tr> </tbody> </table> <p>Buttons: Stop, Report, <b>Close</b></p>	Action	Status	Message	Initializing Data Flow Task	Success		Initializing Connections	Success		Setting SQL Command	Success		Setting Source Connection	Success		Setting Destination Connection	Success		Validating	Success		Prepare for Execute	Success		Pre-execute	Success		Executing	Success		Copying to C:\Users\os\Documents\Alarms To...	Success	60 rows transferred	Post-execute	Success	
Action	Status	Message																																			
Initializing Data Flow Task	Success																																				
Initializing Connections	Success																																				
Setting SQL Command	Success																																				
Setting Source Connection	Success																																				
Setting Destination Connection	Success																																				
Validating	Success																																				
Prepare for Execute	Success																																				
Pre-execute	Success																																				
Executing	Success																																				
Copying to C:\Users\os\Documents\Alarms To...	Success	60 rows transferred																																			
Post-execute	Success																																				
17.	<p>Check the generated "*.csv" file.</p>																																				

**Note**

The archive can also be accessed via the ValueID instead of the variable name. This is more powerful than the symbolic name (e.g. "TAG:R,4,'0000-00-00 00:30:00.000', '0000-00-00 00:00:00.000'").

A query of several archive variables with one SQL statement is possible with a list of ValueIDs. This is placed in parentheses in the SQL query, the ValueIDs are separated by semicolons.  
(e.g. „TAG:R,(4,8,12),'0000-00-00 00:30:00.000', '0000-00-00 00:00:00.000'").

The ValueID of the archive tags can be found in the table "dbo.Archive". Open this "Tables>dbo.Archive" via the context menu "Select Top 1000 Rows".

**Note**

When entering the text string, pay attention to the American spelling for quotation marks and quotation marks.

**Calling the "DTSX Packages"**

To repeat the export, execute the "SSIS Packages" (DTSX file) by double-clicking and then confirm the "Execute" button. The result is displayed in a Progress window.

## 5 Useful information

### 5.1 Basics

#### 5.1.1 SIMATIC WinCC/Connectivity Pack

The SIMATIC WinCC/Connectivity Pack is an option for WinCC. It enables licensed access to online and archive data from WinCC. The following standardized interfaces are available for this purpose:

- OPC UA (Unified Architecture)  
With OPC UA WinCC supports the platform-independent successor technology of OPC.
- OPC HDA (Historical Data Access)  
Access to archived process values.
- OPC A&E (Alarm & Events)  
Access to reporting events and archived messages.
- OPC XML DA (Web-based data exchange, cross-platform)
- Archive Connector  
With it, swapped out WinCC archive databases can be connected to or disconnected for the SQL server. In doing so, an overview of the individual database segments is generated. The Archive Connector can monitor folders and automatically connect copied in archives.
- WinCC OLE DB Provider  
Provider for direct access to the process and message archives in the SQL server database on the WinCC RT machine and on a long-term archive server. The WinCC OLE DB Provider also provides access to the archives connected to Archive Connector.

#### 5.1.2 WinCC OLE DB Provider

In this document and the associated sample project, the use and application of the WinCC OLE DB provider is described. In addition to access to process value and message archives, this provides additional analysis functions, e.g. to determine extremes of archive variables.

Since WinCC V6.0 the WinCC archive databases are segmented and stored in binary, encrypted form. With the WinCC OLE DB Provider, you make this data available transparently and in a readable form. When accessing this data, you do not need to take any measures to decrypt it or take segmentation into account.

A WinCC/Connectivity Pack client has direct access to the archive data via the provider. Whether the data is available in compressed or decompressed form is therefore irrelevant.

#### 5.1.3 Prerequisite for the connection to the WinCC database

To be able to connect to a WinCC archive, one of the following conditions must be fulfilled.

- Part of a WinCC Runtime Project
- Connected via "Attach Database" in SQL Manager
- Connected via the Archive Connector



#### 5.1.4 Using string tags

As of WinCC V7.4 SP1, string variables can also be written to the variable archive. However, the WinCC OLE DB Provider requires a special SQL expression to read the string variables:

Instead of

```
"TAG:R, 1, '0000-00-00 00:10:00.00', '0000-00-00 00:00:00.000"
```

the

```
"TAG_EX:R, 1, '0000-00-00 00:10:00.00', '0000-00-00 00:00:00.000"
```

must be used

## 5.2 Alternative solutions

Several options and add-ons are available for exchanging data between SIMATIC WinCC and applications (e.g. Office applications, MES and ERP systems). The basis for this is a WinCC application.

A distinction must be made between access to online values (current values from the variable household) of SIMATIC WinCC and historical data (e.g. archived variables or messages).

The following list gives an overview of the possibilities offered by SIMATIC WinCC with its associated options and add-ons. The table does not claim to be exhaustive.

Table 5-1

Product	WinCC integrated <sup>1</sup>	Option	Add-on <sup>2</sup>	For data exchange
SIMATIC WinCC	x	-	-	Accessibility via: <ul style="list-style-type: none"> <li>• OPC DA</li> </ul>
SIMATIC WinCC/Connectivity Pack	-	x	-	Accessibility via: <ul style="list-style-type: none"> <li>• OPC UA</li> </ul> The successor of OPC with all access possibilities, like DA, HDA and A&E <ul style="list-style-type: none"> <li>• OPC XML DA</li> </ul> Online values of the variables <ul style="list-style-type: none"> <li>• OPC HDA</li> </ul> Logged process values <ul style="list-style-type: none"> <li>• OPC A&amp;E</li> </ul> Messages and alarms <ul style="list-style-type: none"> <li>• WinCC OLE DB Provider</li> </ul> Logged process values/messages
SIMATIC WinCC/IndustrialData Bridge	-	x	-	Configuration tool and runtime environment for data exchange between different data sources (providers) and data targets (consumers). Can also be used standalone. <p><b>Note:</b> From WinCC V7.4 ".txt"-, ".csv"-, ".html"-, ".xml"- formats permitted.</p>
SIMATIC Information Server	-	-	x	Allows you to save the report in various formats, including ".xls" format.
PM-OPEN EXPORT	-	-	x	Export of online values and archive data to text files (".asci", ".csv", ".html", ".xml").

<sup>1</sup> SIMATIC WinCC RT / RC – license sufficient

<sup>2</sup> Add-Ons are products from our Premium Add-On Partners

## 6 Appendix

### 6.1 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:

<https://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](http://www.siemens.com/industry/supportrequest)

#### SITRAIN – Training for Industry

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](http://www.siemens.com/sitrain)

#### Note

Basics are taught in the SITRAIN course "SIMATIC WinCC, System Course."

- [SIMATIC WinCC, Systemkurs \(de\)](#)
- [SIMATIC WinCC, System Course \(en\)](#)

Topics for cross-system communication with the WinCC/Connectivity Pack are taught in the course "SIMATIC WinCC, Advanced Course", among many other topics.

- [SIMATIC WinCC, advanced course \(de\)](#)
- [SIMATIC WinCC, advanced course \(en\)](#)

### **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:

<https://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 Apple iOS, Android and Windows Phone:

<https://support.industry.siemens.com/cs/ww/en/sc/2067>

## 6.2 Links and literature

Table 6-1

No.	Topic
\1\	Siemens Industry Online Support <a href="https://support.industry.siemens.com">https://support.industry.siemens.com</a>
\2\	Link to the entry page of the application example <a href="https://support.industry.siemens.com/cs/ww/en/view/38132261">https://support.industry.siemens.com/cs/ww/en/view/38132261</a>
\3\	Manual SIMATIC WinCC/Connectivity Pack V7.4 SP1 <a href="https://support.industry.siemens.com/cs/ww/en/view/109746336">https://support.industry.siemens.com/cs/ww/en/view/109746336</a>
\4\	Application example "Export of WinCC/CAS archive data with an independent application". <a href="https://support.industry.siemens.com/cs/ww/en/view/35840700">https://support.industry.siemens.com/cs/ww/en/view/35840700</a>
\5\	Application example "WinCC/IndustrialDataBridge: Write data from SIMATIC WinCC or from SIMATIC controllers to Office files". <a href="https://support.industry.siemens.com/cs/ww/en/view/109483465">https://support.industry.siemens.com/cs/ww/en/view/109483465</a>

## 6.3 Change documentation

Table 6-2

Version	Date	Change
V1.0	08/2009	First edition, WinCC 7.0 + ConnPack
V2.0	02/2016	WinCC 7.3 + ConnPack, new layout of the application example + revised scripts. Changes in the documentation.
V3.0	03/2018	WinCC 7.4 SP1 + ConnPack, revised scripts + export of archive files to Excel. Changes in the documentation.
V4.0	06/2019	WinCC 7.5 + ConnPack, <u>additional Excel client with basic functionality based on the WinCC project</u> . Changes in the documentation.