Manual

Index of contents

Introduction

HSDBASE was created to log data from S7 PLCs easily for every user. The simple applicability was in the focus of attention throughout development.

For fast familiarization a popup help is integrated. It explains every button directly while using HSDBASE.

All general information for the data logging modes (*Cycle, Trigger, Change, Analyzer*) are already included in the popup help. Therefore only the *Read and Write, Ring buffer* and *QDAS* mode is explained in the following.

Additional information is available on our website under FAQ and HSDBASE examples.

Read and Write mode

Advice:

This mode shouldn't be used for data logging, but for the exchange of data in both directions.

For data logging, especially fast loggings, you should use one of the other modes. Clearly higher data rates can be achieved with them.

Control Byte

All read and write jobs are activated through the PLC via a Control Byte.

Should data be written or read, the PLC has to write the matching commando into the Control Byte. HSDBASE reads the Control Byte cyclic according to the adjusted time. If the value is bigger than 100, HSDBASE completes this commando and writes an answering code (smaller than 100) as confirmation into the Control Byte afterwards.

Dataset selection

Inside the variable table of HSDBASE, there is a column named *change value*. Only the selected values in this column are overwritten in the PLC. The unchecked variables are used for the selection of the datasets.

To select the dataset, that should be read (respectively actualized), the indentifier variable has to be created under the variables in HSDBASE and the column *change value* has to be **unchecked**. Datasets can be selected also through the combination of several variables. Before every read/write operation the values are read out of the PLC. Then only the datasets which have the same value (in the selection variables) in the PLC and the database are handled.

If you select several datasets, then all these datasets will be actualized / deleted. On a read command the values of the first selected dataset will be used.

Please have a look on the example in this guide for better understanding.

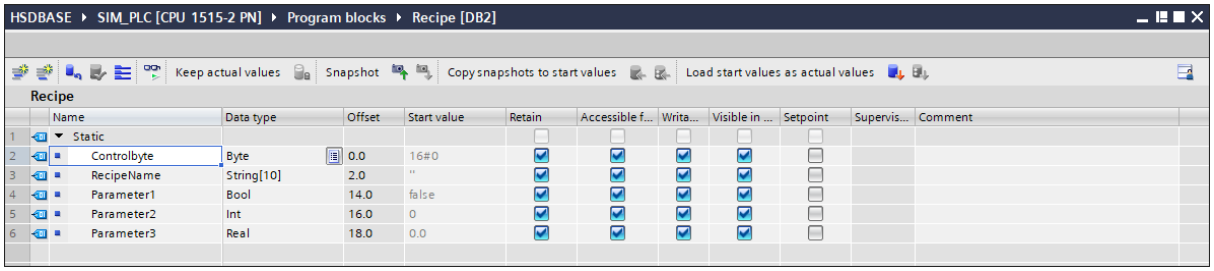
Survey of all codes

	Command codes
101	from database to PLC (dataset remains) - e.g. load recipe
102	from database to PLC (dataset will be deleted) - e.g. load batch
103	query if dataset present - e.g. request before overwriting
111	from PLC to database (overwrite, if dataset exists) - e.g. save recipe
112	from PLC to database (create doubly, if dataset exists) - e.g. create batch
113	delete dataset - e.g. delete recipe or batch
121	fill DisplayDB with available identifiers of recipes/batches
	Answering codes
1	data written to PLC
2	data written to PLC and dataset deleted
3	dataset created
4	dataset updated
5	dataset deleted
6	dataset present
7	DisplayDB was filled
	Fault codes
11	unknown command
12	error dataset does not exist
13	error reading database/file
14	error database/table/file not found
15	error headline does not exist
16	error wrong number of columns (headline)
17	error wrong number of columns (data line)
18	error writing database/table/file
19	error converting/dataset was not written complete
20	DisplayDB does not exist
21	DisplayDB too small
22	checkbox DisplayDB is unchecked
23	WhereClauseDB does not exist
24	WhereClauseDB too small

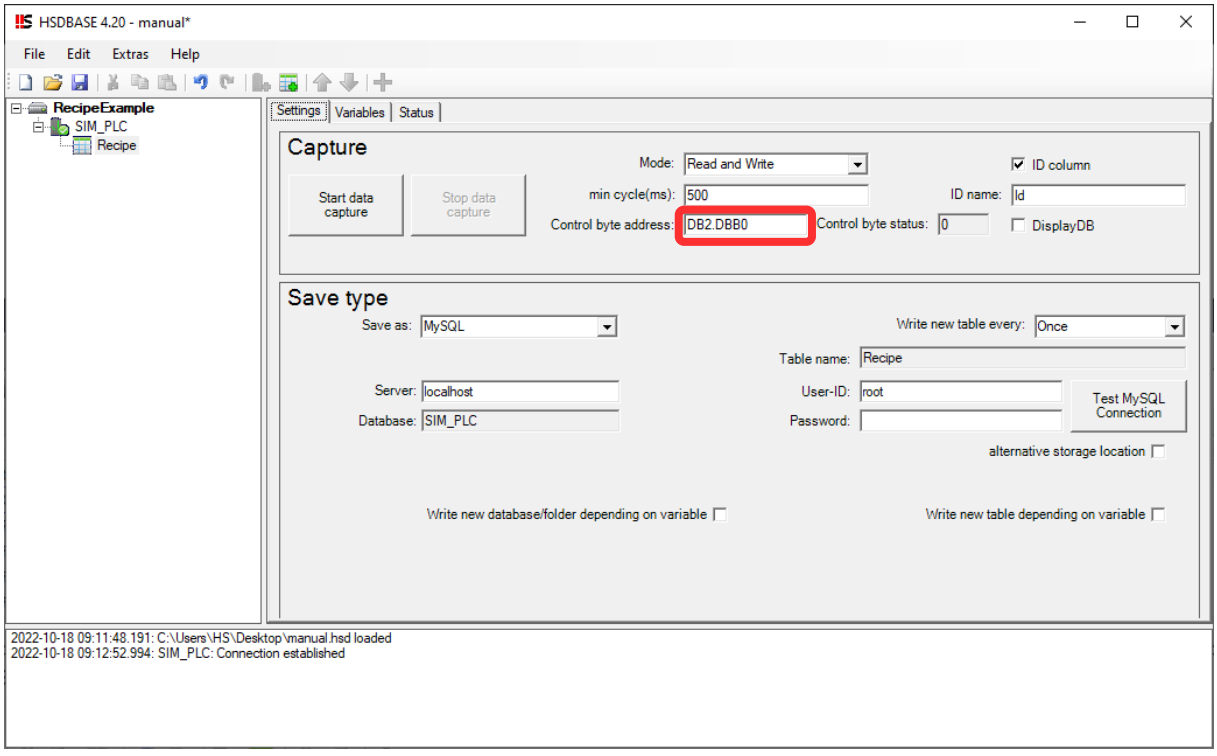
Example recipe management

The following example shows a simple recipe with 3 parameters. This example can be reproduced very easily and it is relatively simple to test all commands.

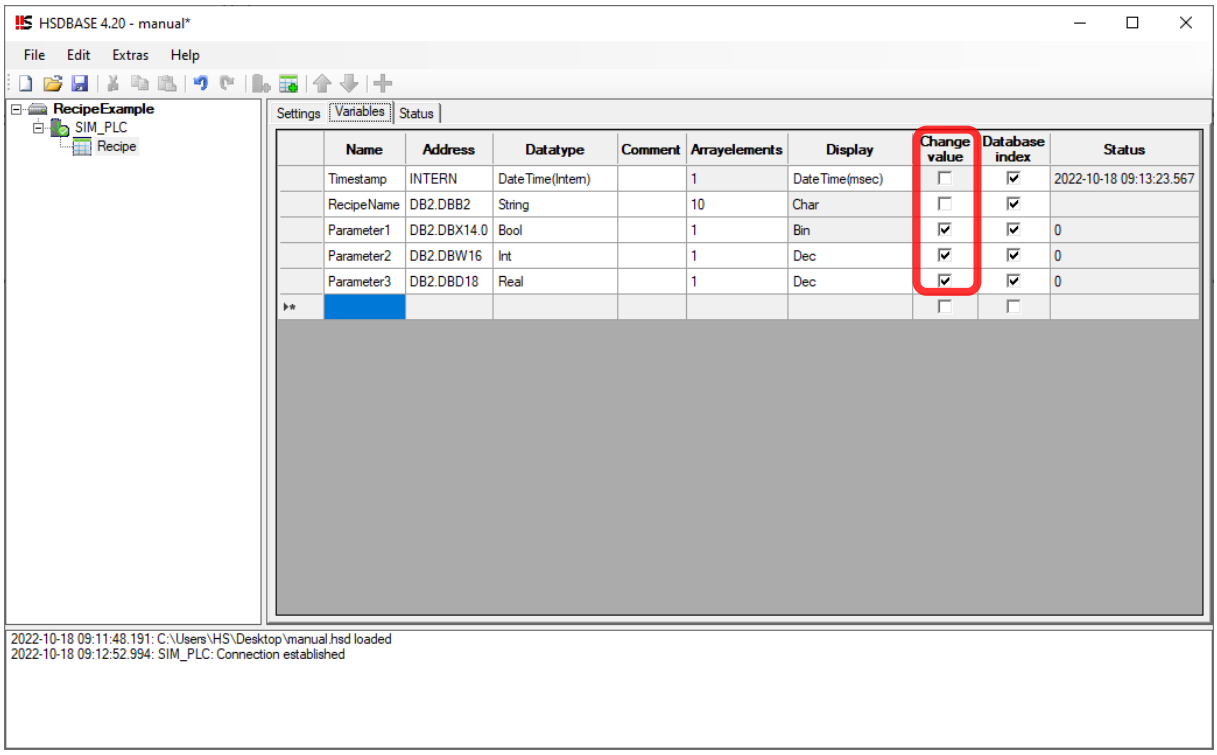
Please create the following data block.



Create a new table in *HS*DBASE. Set *Read and Write* as mode and set the address of the Control Byte.



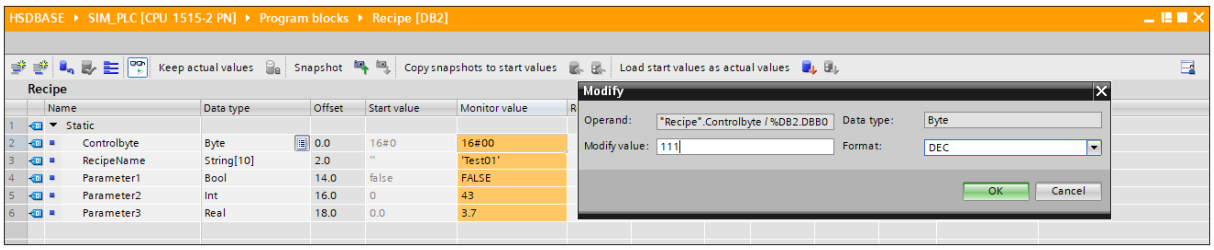
Please note that the checkmarks in the column *change value* are placed correctly. Uncheck the recipe name (identifier variable), but check the parameters because they are overwritten by HSDBASE according to the database entry.



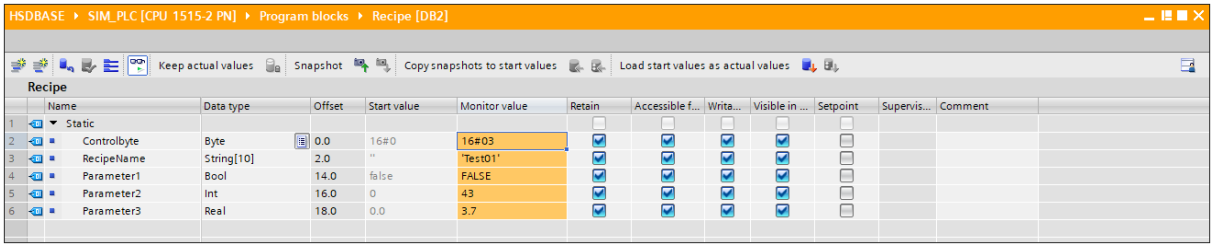
Start the capture now, so that HSDBASE can react to the commands of the PLC.

Save a recipe in the database

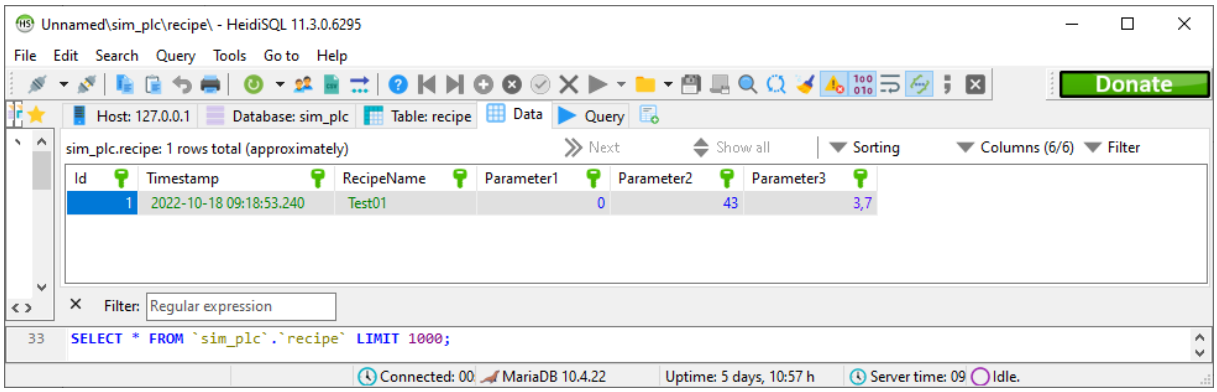
Set random values for recipe name and parameters. Modify the Control Byte to 111 (From PLC to data base).



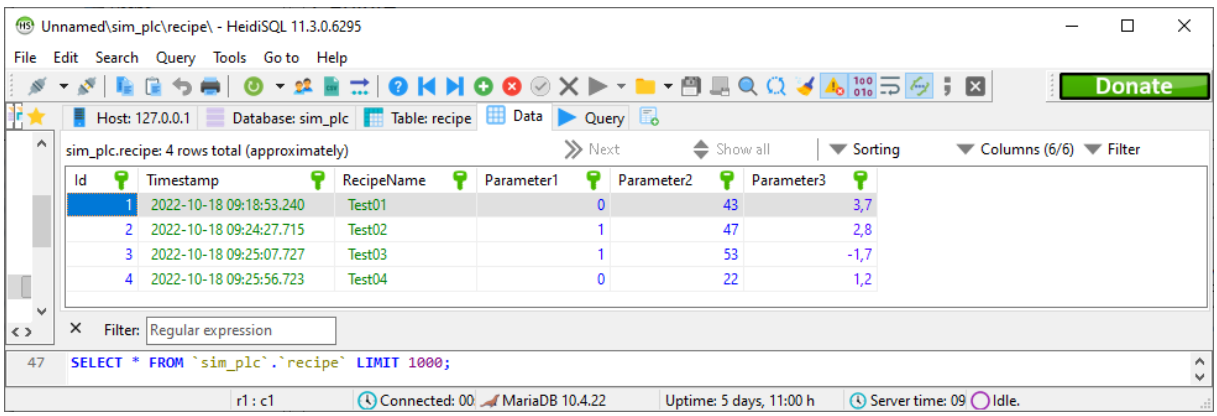
As answering code HSDBASE writes 3 (dataset was created) into the Control Byte.



Now the recipe is in the database. Database and table have been created automatically by HSDBASE.

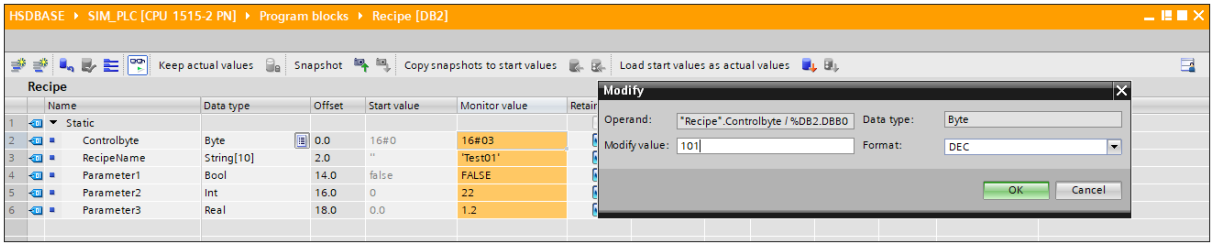


This step can be repeated as often as needed with different recipe numbers and values. If you apply an already used recipe number, the dataset will be actualized.

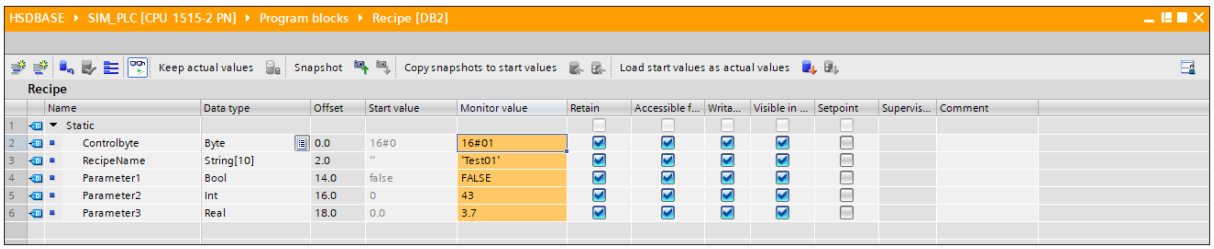


Read recipe from database

Now the recipe ‘Test01’ should be transferred back to the PLC. To do this, the recipe name has to be set first. Then modify the Control Byte to 101 (from data base to PLC).

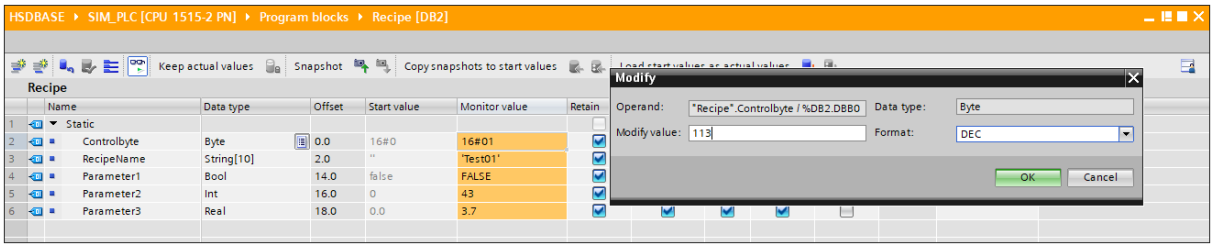


The parameters have been overwritten and the answering code 1 was set (data was written in PLC).

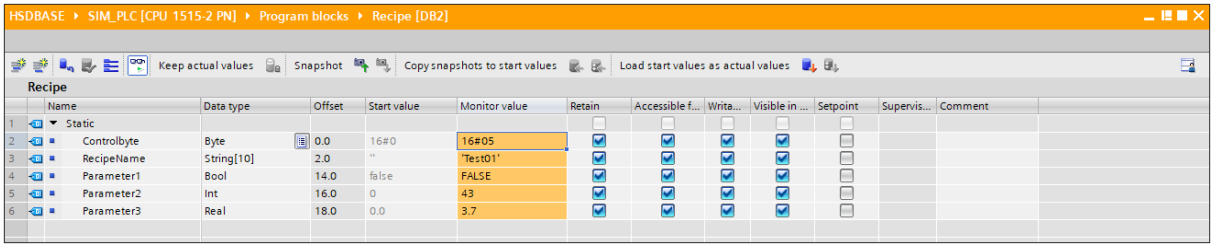


Delete recipe

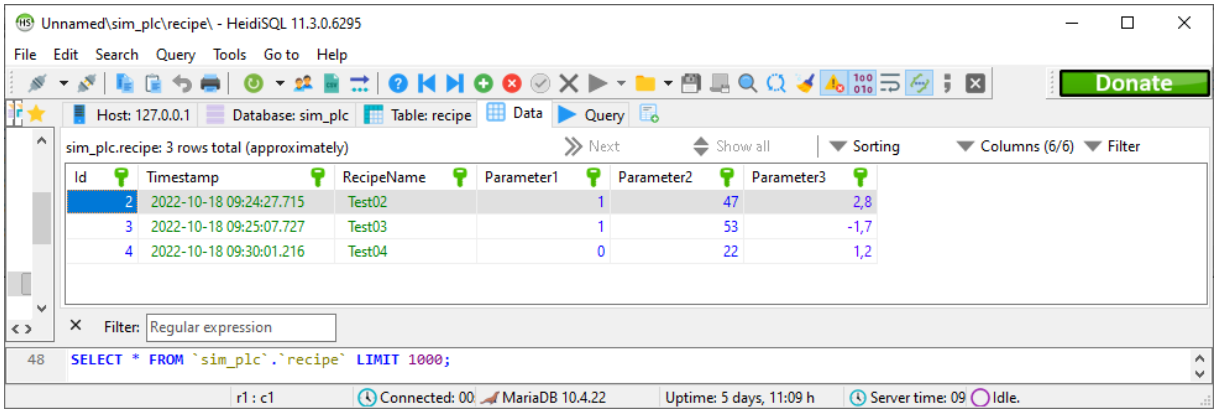
Now the recipe ‘Test01’ should be deleted. To do this, the Control Byte has to be modified to 113 (delete dataset).



The answering code is 5 (dataset was deleted).

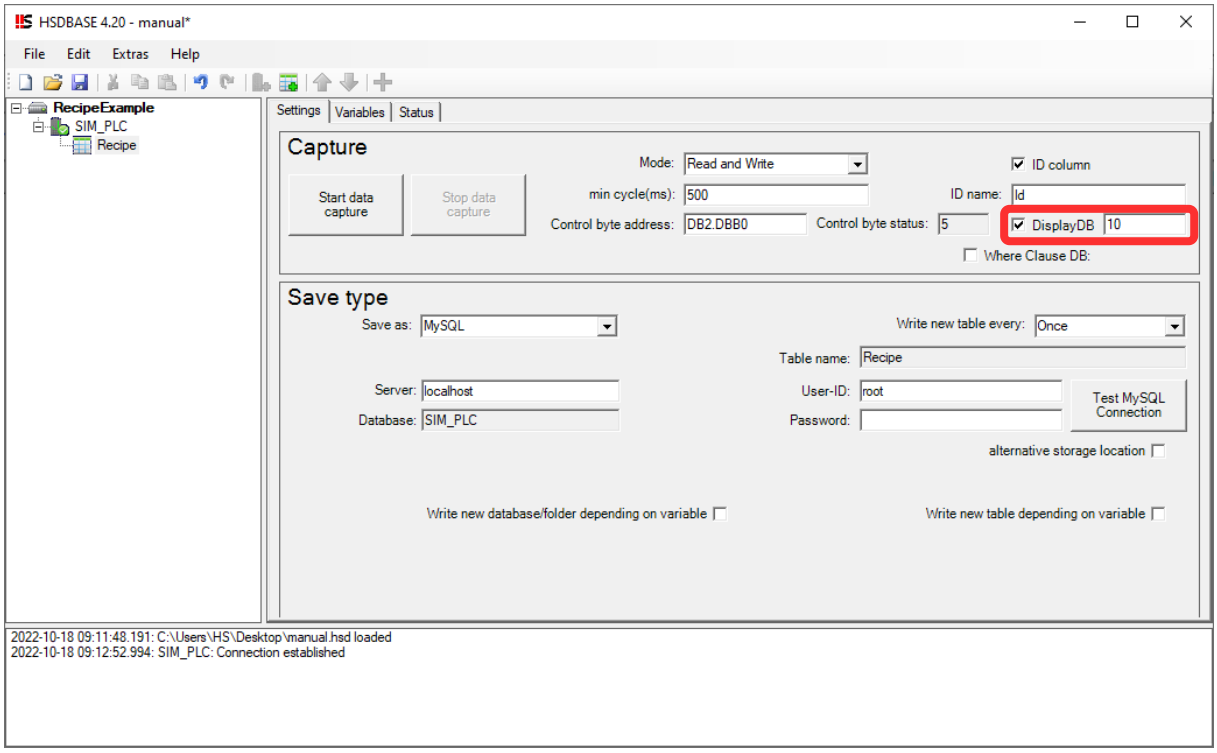


Now the database table looks as follows:

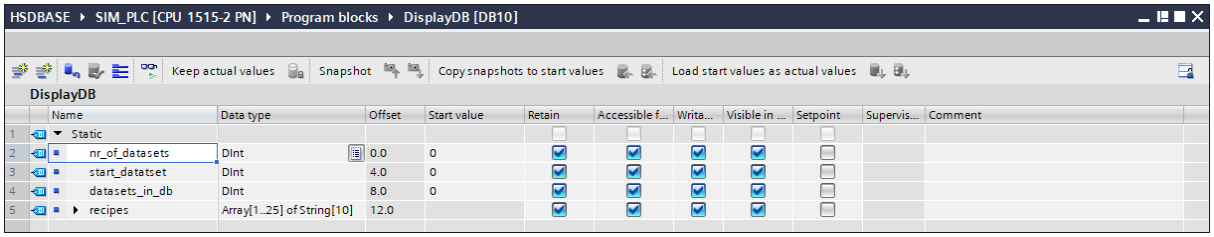


Display data block

To read out which recipes are present in the database, a data block can be created. This data block will be filled by HSDBASE.



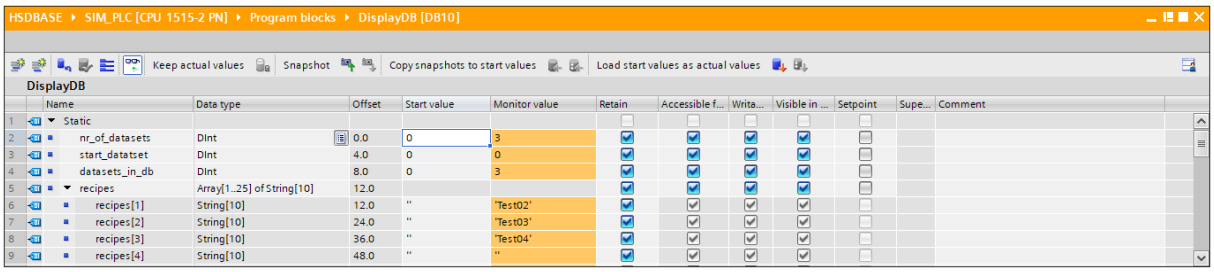
Structure of the display data block.



The first three entries have to be type of DInt. Array length of the identifier could be any size, but the data type has to be the same type as the identifier.

When the Control Byte is modified to 121 (fill DisplayDB) HSDBASE writes the number of all datasets in the database into the first DINT. The third DINT contains the number of datasets, which have been written into the data block. The rest of the data block is filled up with the identifiers of the datasets. HSDBASE detects automatically how many identifiers match into the block. If more datasets exist in the database than space is available, the beginning of the display area can be adjusted with the second DINT. If there are less datasets than space, the remaining space will be filled with w#16#00.

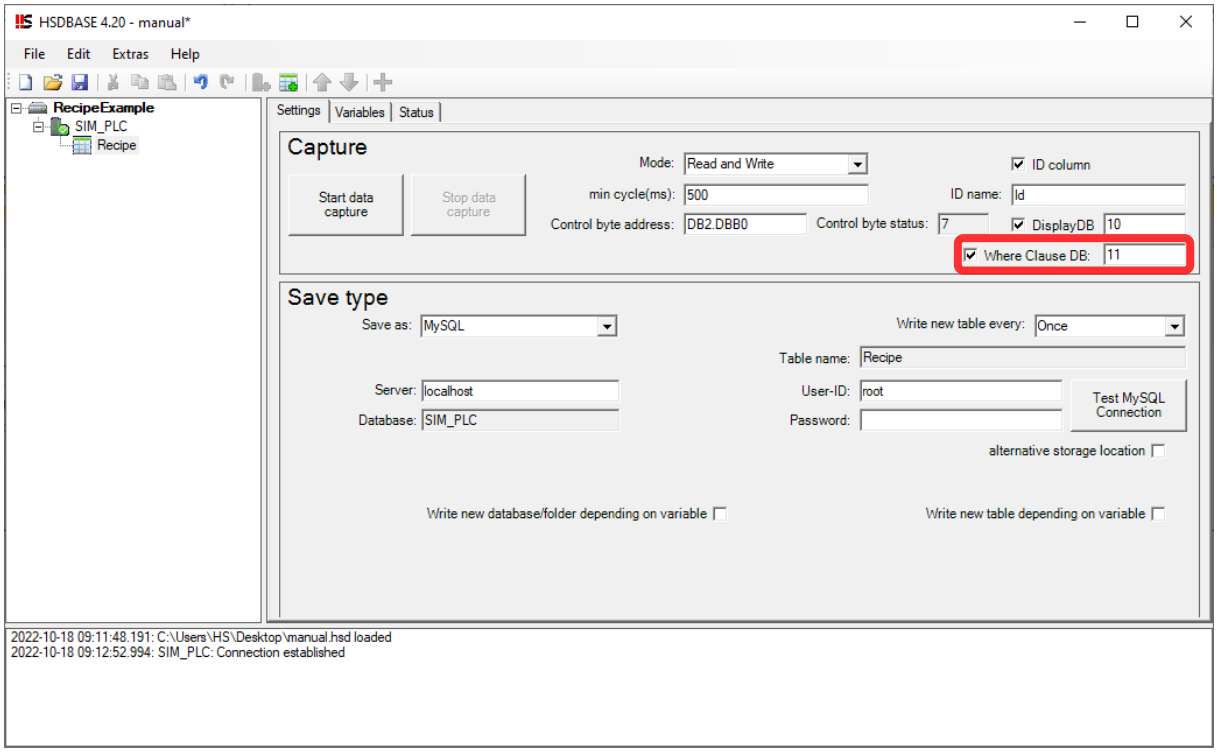
After modifying the Control Byte, the display data block shows all identifiers of the datasets in the database.



	Name	Data type	Offset	Start value	Monitor value	Retain	Accessible f...	Writa...	Visible in ...	Setpoint	Supe...	Comment
1	Static											
2	nr_of_datasets	DInt	0.0	0	3	☑	☑	☑	☑	☐		
3	start_datatset	DInt	4.0	0	0	☑	☑	☑	☑	☐		
4	datasets_in_db	DInt	8.0	0	3	☑	☑	☑	☑	☐		
5	recipes	Array[1..25] of String[10]	12.0			☑	☑	☑	☑	☐		
6	recipes[1]	String[10]	12.0	''	'Test02'	☑	☑	☑	☑	☐		
7	recipes[2]	String[10]	24.0	''	'Test03'	☑	☑	☑	☑	☐		
8	recipes[3]	String[10]	36.0	''	'Test04'	☑	☑	☑	☑	☐		
9	recipes[4]	String[10]	48.0	''	''	☑	☑	☑	☑	☐		

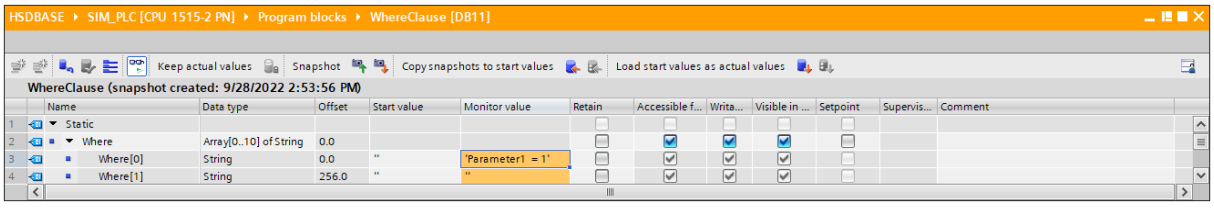
WhereClause data block

In order to create the possibility to select from a database only datasets, which fulfill certain conditions, there is in HSDBASE the possibility to enter conditions in a further DB, which are appended to the SQL query inside HSDBASE in a "Where Clause".

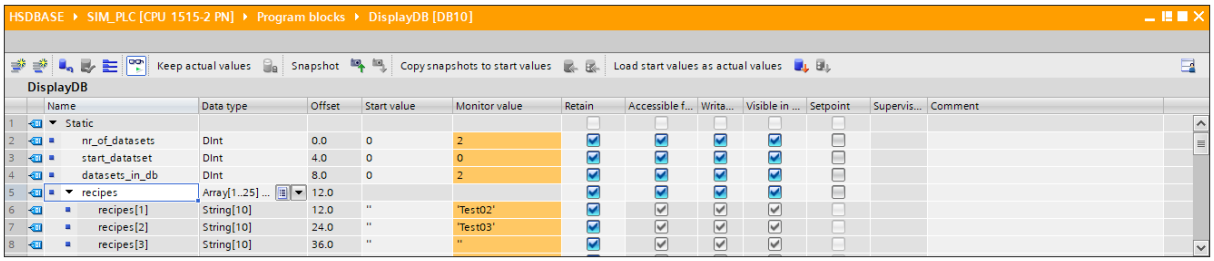


The WhereClause-DB consists of any number of strings. With these strings the internal SQL command in HSDBASE is extended behind the keyword WHERE. If the first string is empty, the query is executed without "WhereClause". When creating the strings, the corresponding SQL syntax of the selected database must be observed. A check does not occure.

Simple condition for the previous example:



After modifying the control byte to 121 (fill display BD), the display DB is filled only with the data records that meet the set conditions.



Ring buffer mode

In In this mode, data is collected in a FIFO buffer on the machine that runs HSDBASE. Data from this buffer is written to the database on command. The buffer is controlled by a Control Byte like in the Read and Write mode.

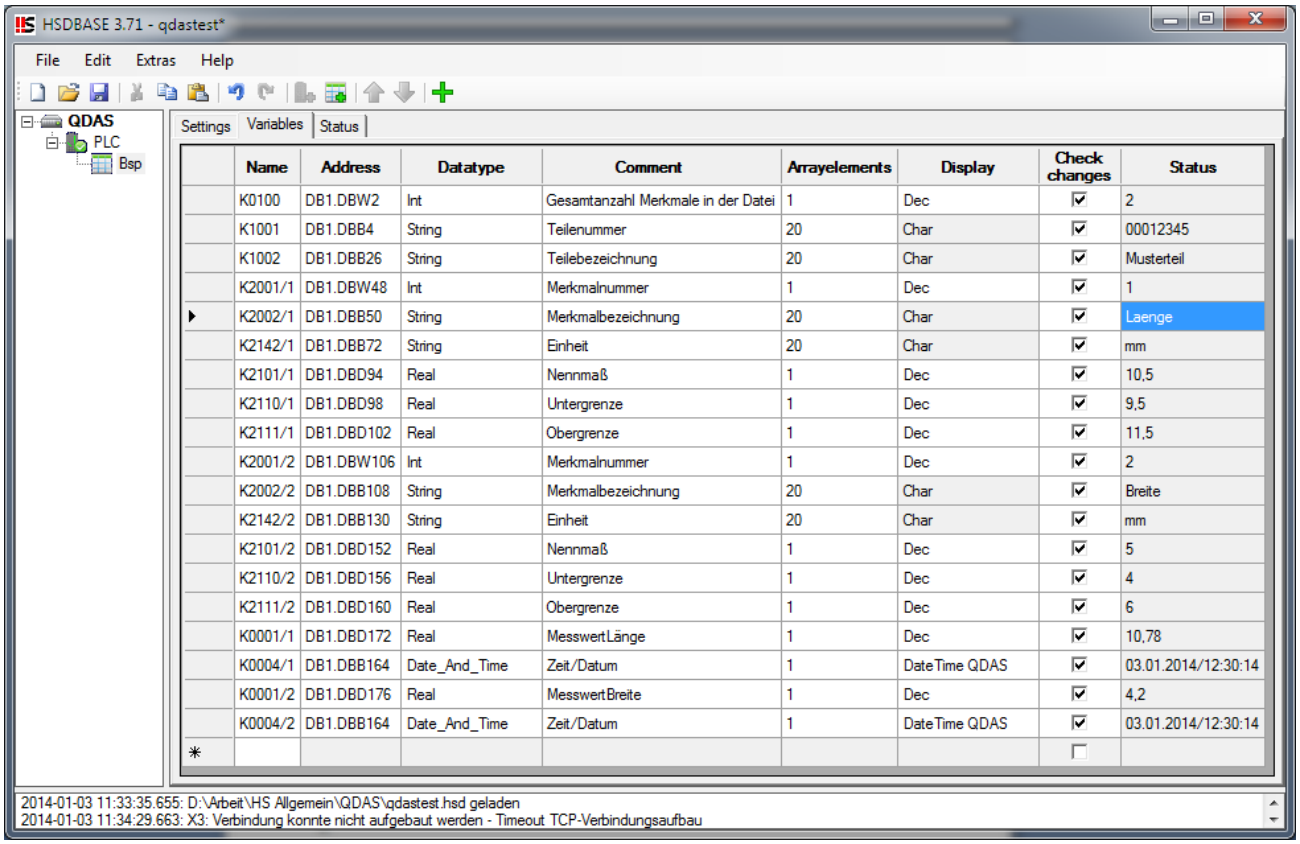
Survey of all codes:

	Command codes
101	delete buffer
102	fill buffer (no acknowledgment)
103	save buffer
	Answering codes
1	buffer deleted
2	buffer saved (and deleted)
11	unknown command

QDAS mode

The following example shows how HSDBASE has to be configured to save the data as QDAS DFQ file. These files could be read by QDAS upload tool and transferred to the QDAS database automatically.

The QDAS K numbers have to be entered in the name column. A separate DFQ file with consecutive number will be created on every trigger event.



	Name	Address	Datatype	Comment	Arrayelements	Display	Check changes	Status
	K0100	DB1.DBW2	Int	Gesamtanzahl Merkmale in der Datei	1	Dec	☑	2
	K1001	DB1.DBB4	String	Teilenummer	20	Char	☑	00012345
	K1002	DB1.DBB26	String	Teilebezeichnung	20	Char	☑	Musterteil
	K2001/1	DB1.DBW48	Int	Merkmalnummer	1	Dec	☑	1
▶	K2002/1	DB1.DBB50	String	Merkmalbezeichnung	20	Char	☑	Laenge
	K2142/1	DB1.DBB72	String	Einheit	20	Char	☑	mm
	K2101/1	DB1.DBD94	Real	Nennmaß	1	Dec	☑	10,5
	K2110/1	DB1.DBD98	Real	Untergrenze	1	Dec	☑	9,5
	K2111/1	DB1.DBD102	Real	Obergrenze	1	Dec	☑	11,5
	K2001/2	DB1.DBW106	Int	Merkmalnummer	1	Dec	☑	2
	K2002/2	DB1.DBB108	String	Merkmalbezeichnung	20	Char	☑	Breite
	K2142/2	DB1.DBB130	String	Einheit	20	Char	☑	mm
	K2101/2	DB1.DBD152	Real	Nennmaß	1	Dec	☑	5
	K2110/2	DB1.DBD156	Real	Untergrenze	1	Dec	☑	4
	K2111/2	DB1.DBD160	Real	Obergrenze	1	Dec	☑	6
	K0001/1	DB1.DBD172	Real	MesswertLänge	1	Dec	☑	10,78
	K0004/1	DB1.DBB164	Date_And_Time	Zeit/Datum	1	DateTime QDAS	☑	03.01.2014/12:30:14
	K0001/2	DB1.DBD176	Real	MesswertBreite	1	Dec	☑	4,2
	K0004/2	DB1.DBB164	Date_And_Time	Zeit/Datum	1	DateTime QDAS	☑	03.01.2014/12:30:14
*							☐	

Contact

If there are any other questions, please do not hesitate to contact us at any time.

HS Automation Software
Im Öderich 2
D-73663 Berglen
Tel.: + 49 (0) 7195 - 588 639
E-Mail: info@hs-automation-software.de
Web: www.hs-automation-software.de