Bedienungsanleitung



Inhalt

Einleitung	2
Nodus Lesen und Schreiben	3
Controlbyte	3
Datensatzauswahl	3
Alle Codes im Überblick	4
Beispiel Rezeptverwaltung	5
Rezeptur in Datenbank speichern	7
Rezeptur aus Datenbank lesen	8
Rezeptur löschen	9
Anzeige Datenbaustein1	.0
WhereClause – Datenbaustein1	.2
Nodus Ringspeicher1	.4
Nodus QDAS1	.5
Kontakt 1	.6

Einleitung

HSDBASE wurde konzipiert um es für jeden Anwender möglich zu machen, Daten aus S7-Steuerungen mitzuschreiben. Auf die einfache Anwendbarkeit wurde bei der Entwicklung dieser Software das Hauptaugenmerk gelegt.

Zur schnellen Einarbeitung gibt es im Programm eine integrierte Popup-Hilfe, die direkt bei der Verwendung von HSDBASE jede Schaltfläche erklärt.

Alle grundlegenden Informationen zu **HS**DBASE und den Modi zum Datenloggen (*Zyklisch, Trigger, Änderung, Analyzer*) sind im Programm in der Popup-Hilfe bereits enthalten.

Diese Kurzanleitung beschreibt deshalb lediglich die Modi *Lesen und Schreiben, Ringspeicher,* sowie die Datenerfassung im *QDAS Datenformat*.

Weitere Informationen finden Sie auf unserer Homepage unter FAQ und **HS**DBASE Beispiele.

Modus Lesen und Schreiben

Hinweis:

Dieser Modus ist nicht zum Datenloggen gedacht, sondern nur zum Austausch von Daten in beide Richtungen.

Verwenden Sie zum Datenloggen, besonders bei schnellen Aufzeichnungen, einen der anderen Modi, da hier deutlich höhere Datenraten erzielt werden können.

Controlbyte

Alle Lese- und Schreibaufträge werden von der Steuerung über ein Controlbyte angestoßen.

Sollen Daten geschrieben bzw. gelesen werden, muss die Steuerung das entsprechende Kommando in dieses Controlbyte schreiben. **HS**DBASE liest das Controlbyte zyklisch entsprechend der eingestellten Zeit. Wird ein Wert größer 100 gelesen, arbeitet **HS**DBASE dieses Kommando ab und schreibt anschließend als Quittierung einen Antwortcode (kleiner 100) in das Controlbyte.

Datensatzauswahl

In **HS**DBASE befindet sich in der Variablentabelle eine Spalte *Wert ändern*. Nur die angewählten Werte in dieser Spalte werden von **HS**DBASE in der Steuerung überschrieben. Die nicht angewählten Variablen werden zur Auswahl der Datensätze verwendet.

Um zu bestimmen, welcher Datensatz aus der Datenbank gelesen (bzw. aktualisiert) werden soll, muss die Auswahlvariable in **HS**DBASE unter den Variablen angelegt werden und das Häkchen in der Spalte *Wert ändern* muss **abgewählt** sein. Der Datensatz kann auch anhand einer Kombination von mehreren Variablen bestimmt werden. Vor jeder Schreib- / Leseoperation werden die Werte aus der Steuerung ausgelesen. Dann werden nur die Datensätze bearbeitet, bei denen die Auswahlvariablen in Steuerung und Datenbank übereinstimmen.

Werden in der Datenbank mehrere Datensätze ausgewählt, so werden alle diese Datensätze aktualisiert/gelöscht. Bei einem Lesebefehl werden die Werte des ersten ausgewählten Datensatzes verwendet.

Zum besseren Verständnis wird empfohlen sich das folgende Beispiel in diesem Handbuch anzuschauen.

Alle Codes im Überblick

	Kommandocodes
101	Aus Datenbank in SPS (Datensatz bleibt erhalten) - z.B. Rezeptur laden
102	Aus Datenbank in SPS (Datensatz wird gelöscht) - z.B. Auftrag laden
103	Abfrage Datensatz vorhanden - z.B. für Abfrage ob überschrieben werden soll
111	Aus SPS in Datenbank (Überschreiben, wenn vorhanden) - z.B. Rezeptur speichern
112	Aus SPS in Datenbank (Doppelt anlegen, wenn vorhanden) - z.B. Auftrag anlegen
113	Datensatz löschen - z.B. Rezept oder Auftrag löschen
121	AnzeigeDB mit verfügbaren Rezepten/Aufträgen füllen
	Antwortcodes
1	Daten wurden in SPS geschrieben
2	Daten wurden in SPS geschrieben und Datensatz wurde gelöscht
3	Datensatz wurde erstellt
4	Datensatz wurde aktualisiert
5	Datensatz wurde gelöscht
6	Datensatz vorhanden
7	AnzeigeDB wurde gefüllt
	Fehlercodes
11	Unbekannter Befehl
12	Fehler Datensatz nicht vorhanden
13	Fehler beim Lesen aus Datenbank
14	Fehler Datenbank/Tabelle/Datei nicht gefunden
15	Fehler keine Kopfzeile vorhanden
16	Fehler Anzahl Spalten Kopfzeile falsch
17	Fehler Anzahl Spalten Datenzeile falsch
18	Fehler beim Schreiben in Datenbank/Tabelle/Datei
19	Fehler beim Konvertieren / Datensatz wurde nicht komplett geschrieben
20	AnzeigeDB nicht vorhanden
21	AnzeigeDB zu klein
22	AnzeigeDB nicht angewählt
23	WhereClauseDB nicht vorhanden
24	WhereClauseDB zu klein

Beispiel Rezeptverwaltung

Folgendes Beispiel zeigt eine einfache Rezeptur mit 3 Parametern. Dieses Beispiel lässt sich einfach nachstellen um alle Kommandos zu testen.

Legen Sie folgenden Datenbaustein an:

HS	HSDBASE > SIM_PLC [CPU 1515-2 PN] > Program blocks > Recipe [DB2]													_ ∎∎×
1	1	ء ا	👆 🋃 🚞 🤓 Keep ac	tual values 🛛 🔒 Sn	apshot 🛤	🖌 🖳 🛛 Copy snap	shots to star	tvalues 🖳 🛛	Loa	d start value:	as actual va	lues 其	B),	_
	Ree	cipe	•											
		Nan	ne	Data type	Offset	Start value	Retain	Accessible f	Writa	Visible in	Setpoint	Supervis	Comment	
1	-0	•	Static											
2	-		Controlbyte	Byte 🔳	0.0	16#0								
З	-		RecipeName	String[10]	2.0									
4	-0	•	Parameter1	Bool	14.0	false								
5	-		Parameter2	Int	16.0	0								
6	-	•	Parameter3	Real	18.0	0.0								

Legen Sie in **HS**DBASE eine neue Tabelle an. Stellen Sie als Modus *Lesen und Schreiben* ein. Anschließend geben sie die Adresse des Controlbytes ein:

IS HSDBASE 4.20 - manual*		-		×
File Edit Extras Help				
0 🐸 🖬 1 🗴 🖻 🛍 1 🤊 🕫 1 🖍 🐻 1 🔶 1 🕂 -				
RecipeExample Settings Variables St.	stus			
Recipe	Mode: Read and Write	olumn		
Start data capture	Stop data min cycle(ms): 500 ID name: Id			
	Control byte address: DB2.DBB0 Control byte status: 0 Dist	playDB		
Save type				_
Save as:	MySQL Write new table every: On	ce		-
	Table name: Recipe			
Server	: localhost User-ID: root	Te	st MySQL	
Database	SIM_PLC Password:	- c	onnection	
	alternative	storage lo	ocation 🥅	
	Write new database/folder depending on variable 🗌 Write new table dependence	ding on v	ariable 厂	
2022-10-18 09:11:48.191: C:\Users\HS\Desktop\manual hsd loaded 2022-10-18 09:12:52.994: SIM PLC: Connection established				

Beim Anlegen der Variablen ist zu beachten, dass die Häkchen in der Spalte *Wert ändern* richtig gesetzt sind. Bei Rezeptnamen (Auswahlvariablen) nicht gesetzt, bei den Parametern gesetzt, da diese von **HS**DBASE entsprechend des Datenbankeintrags überschrieben werden:

KSDBASE 4.20 - manual*									-		×
File Edit Extras Help											
🗋 📴 🛃 🎽 🛍 i 🤊 🖤 i 🌆 📆 i 4	+++										
Settings	Variables	itatus									
Recipe	Name	Address	Datatype	Comment	Arrayelements	Display	Change value	Database index	Sta	tus	
	Timestamp	INTERN	DateTime(Intem)		1	DateTime(msec)			2022-10-18 0	9:13:2	3.567
	RecipeName	DB2.DBB2	String		10	Char					
	Parameter1	DB2.DBX14.0	Bool		1	Bin		•	0		
	Parameter2	DB2.DBW16	Int		1	Dec			0		
	Parameter3	DB2.DBD18	Real		1	Dec			0		
							Γ				
2022 10 19 09 11 49 191 C \ been \ H0\ Doubles mail											
2022-10-18 09:11:48.191: C::Users'HS:Users'HS:Userstop vinanu 2022-10-18 09:12:52:994: SIM_PLC: Connection establi	al.hsd loaded shed										

Starten Sie nun die Erfassung, damit **HS**DBASE auf die Kommandos der SPS reagiert.

Rezeptur in Datenbank speichern

Steuern Sie den Rezeptnamen und die Parameter auf beliebige Werte und dann das Controlbyte auf 111 (Aus SPS in Datenbank):

HSDB	ASE → SIM_PLC [CPU 151	5-2 PN] 🔸 Progran	n blocks	• Recipe [DB2]]		_ ⊫∎×
				1			
1 🖻 🖻	🤊 🔩 🥪 는 🍄 Keepa	ctual values 🛛 🔒 Si	napshot 🇯	🖣 🔍 Copy sna	apshots to start values	👝 🔣 🛛 Load start values as actual values 🛛 💐 💷	
Re	ipe					Modify X	
	Name	Data type	Offset	Start value	Monitor value		
1 📲	▼ Static					Operand: "Recipe".Controlbyte / %DB2.DBB0 Data type: Byte	
2 🕣	 Controlbyte 	Byte	0.0	16#0	16#00	Modifyvalue: 111 Format: DEC	
з 🐽	 RecipeName 	String[10]	2.0		'Test01'		
4 🕣	Parameter1	Bool	14.0	false	FALSE		
5 📲	 Parameter2 	Int	16.0	0	43	OK Cancel	
6 🕣	 Parameter3 	Real	18.0	0.0	3.7		

Als Antwortcode schreibt **HS**DBASE eine 3 (Datensatz wurde erstellt) in das Controlbyte:

HS	HSDBASE → SIM_PLC (CPU 1515-2 PN) → Program blocks → Recipe [DB2]														
2	🤹 🐏 🍓 😹 🏣 🅎 Keep actual values 🔒 Snapshot 👒 🧠 Copy snapshots to start values 😹 🐼 Load start values as actual values 👪 🕮														
	Reci	ipe													
	-	Name	Data type	Offset	Start value	Monitor value	Retain	Accessible f	Writa	Visible in	Setpoint	Supervis	Comment		
1	-	 Static 													
2		 Controlbyte 	Byte	0.0	16#0	16#03									
з		 RecipeName 	String[10]	2.0		'Test01'									
4		Parameter1	Bool	14.0	false	FALSE									
5		 Parameter2 	Int	16.0	0	43									
6	-	 Parameter3 	Real	18.0	0.0	3.7									

Das Rezept befindet sich nun in der Datenbank. Datenbank und Tabelle wurden von **HS**DBASE automatisch angelegt:

	🚯 Ur	name	d\sim_	plc\recipe\ - He	eidiSQL 11.3.0.6	5295								_		×
F	ile f	dit	Search	Query Tool	s Goto He	lp										
****	1	- 🔊		🔋 👈 🖶 📄	3 - 🙎 🖥) 📰 🛛 🕅	$H \odot \Theta$	$\odot \times \blacktriangleright$	- 📒 - 🗎		. Q 🏼 🖌 🖊	b 100 → 6	; 🗙		Donat	e
	*		Host: 1	27.0.0.1 📃 D	atabase: sim_p	lc 📑 Table: re	ecipe 🔠 Da	ata 🕨 Que	ry 🖪							
•	^	sim_	plc.reci	pe: 1 rows total	(approximate	y)		≫ Ne	t 🗧	Show	all i	 Sorting 	Columns	(6/6)	🖝 Filter	
		ld	9	Timestamp	9	RecipeName	💡 Param	eter1 💡	Parameter2	9 F	Parameter3	9				
			1	2022-10-18 09	:18:53.240	Test01		0		43		3,7				
	~			[
<	>	×	Filter:	Regular expres	ssion											
	33	SEL	ECT *	FROM `sim_p	lc`.`recipe	* LIMIT 1000);									0
F							ed: 00 🖌 Mai	riaDB 10.4.22	Untir	ne: 5 dav	s 10:57 h	() Server tir	me: 09 Oldle.			

Dieser Schritt kann nun beliebig oft mit verschiedenen Rezeptnamen und Werten wiederholt werden. Wird eine bereits verwendete Rezepturnummer benutzt, wird der Datensatz aktualisiert:

🛞 U	> Unnamed\sim_plc\recipe\ - HeidiSQL 11.3.0.6295 − ×													
File	le Edit Search Query Tools Go to Help													
	- 💰	¥ 📭	📴 🦘 👼 🛛 🗸 💶		🖸 😣 🕢 🗙 🕨	- 🗀 - 💾 📖 (Q 💭 🍯 👍 😳 🖉	😔 🗦 🖂 🕴	Dona	te				
1		Host: 1	127.0.0.1 📄 Database: sim_j	olc 📑 Table: recipe	🔠 Data 🕨 Que	ery 🖪								
^	sim	_plc.rec	ipe: 4 rows total (approximate	ly)	≫ Ne	kt 🔶 Sho	w all 🛛 🔍 Sorting	🔻 Columns (6/6)	🔻 Filter					
	ld	9	Timestamp 💡	RecipeName 💡	Parameter1 💡	Parameter2 💡	Parameter3 💡							
		1	2022-10-18 09:18:53.240	Test01	0	43	3,7							
		2	2022-10-18 09:24:27.715	Test02	1	47	2,8							
		3	2022-10-18 09:25:07.727	Test03	1	53	-1,7							
		4	2022-10-18 09:25:56.723	Test04	0	22	1,2							
- v														
\mathbf{O}	×	Filter:	Regular expression]										
47	SE	LECT *	FROM `sim_plc`.`recip	≥` LIMIT 1000;						\$				
			r1 : c1	Connected: 00	利 MariaDB 10.4.22	Uptime: 5 d	ays, 11:00 h 🕔 Server t	time: 09 🔘 Idle.						

© HS Automation Software 2022-10-19

Rezeptur aus Datenbank lesen

Nun soll das Rezept mit dem Namen "Test01" von der Datenbank zurück in die SPS übertragen werden. Hierzu muss erst der Rezeptname in der Steuerung eingestellt werden und dann das Controlbyte auf 101 (Aus Datenbank in SPS) gesteuert werden:

HSD	BAS	E → SIM_PLC [CPU 15	15-2 PN] 🕨 Program	blocks	Recipe [DB2]						_ ⊫∎×
, ⇒	2	🔩 🋃 🗮 🎇 Keep	actual values 🛛 🔒 Sn	apshot 🏾	🎙 🖳 Copysn	apshots to start values	R- 6	🕵 Load start	values as actual values 🛛 🛃 🗐			
1	Recip	e						Modify			×	
	Na	ame	Data type	Offset	Start value	Monitor value	Retai					
1 -	•	Static						Operand:	"Recipe".Controlbyte / %DB2.DBB0	Data type:	Byte	
2 .	•	Controlbyte	Byte	0.0	16#0	16#03		Modify value:	101	Format:	DEC	
з -	•	RecipeName	String[10]	2.0		'Test01'	1				bee	
4 -	•	Parameter1	Bool	14.0	false	FALSE						
5	•	Parameter2	Int	16.0	0	22					OK Cancel	
6	•	Parameter3	Real	18.0	0.0	1.2						

Die Parameter werden mit denen aus der Datenbank überschrieben und der Antwortcode 1 (Daten wurden in SPS geschrieben) wird von **HS**DBASE gesetzt:

SDBASE ► SIM_PLC [CPU 1515-2 PN] ► Program blocks ► Recipe [DB2]														
🛫 🐏 💐 😹 🚰 🧮 🕎 Keep actual values 🔒 Snapshot 🏘 🧠 Copy snapshots to start values 象 🛞 Load start values as actual values 💐 🕸														
Recipe														
Name	Data type	Offset	Start value	Monitor value	Retain	Accessible f	Writa	Visible in	Setpoint	Supervis	Comment			
1 🕣 🔻 Static														
2 🕣 🔹 Controlbyte	Byte	0.0	16#0	16#01										
3 💶 = RecipeName	String[10]	2.0		'Test01'										
4 📲 🔹 Parameter1	Bool	14.0	false	FALSE										
5 💶 🖷 Parameter2	Int	16.0	0	43										
6 📶 = Parameter3	Real	18.0	0.0	3.7										

Rezeptur löschen

Nun soll das Rezept ("Test01") gelöscht werden. Hierzu muss das Controlbyte auf 113 (Datensatz löschen) gesetzt werden:

HSDBASE	SIM_PLC [CPU 151	5-2 PN] 🔸 Program		Recipe [DB2]									_ ⊫ ■ ×
🥩 🔮 🖣	🐛 🋃 🗮 🏋 Keep ad	tual values 🛛 🔒 Sn	apshot 🕷	🛉 🖳 Copysnap	oshots to start values	R - R -	Load start va	luer ar actu	al valuer 🗧	. 8.		X	
Recipe	e						моатту					~	
Nar	me	Data type	Offset	Start value	Monitor value	Retain	Operand:	"Recipe".Co	ontrolbyte / %	DB2.DBB0	Data type:	Byte	
1 🕣 🔻	Static						Modifyvalue:	113			Format:	DEC	
2 📲 🖷	Controlbyte	Byte 🔳	0.0	16#0	16#01		incomy rande.						
3 🕣 🗉	RecipeName	String[10]	2.0		'Test01'								
4 📲 🔳	Parameter1	Bool	14.0	false	FALSE							OK Cancel	
5 📲 🖷	Parameter2	Int	16.0	0	43								
6 📲	Parameter3	Real	18.0	0.0	3.7								

Als Antwortcode erhalten Sie 5 (Datensatz wurde gelöscht):

HS	HSDBASE + SIM_PLC [CPU 1515-2 PN] + Program blocks + Recipe [DB2]													_ ∎∎×	
1	🛫 🐏 🔩 🛃 🗮 🎇 Keep actual values 🔒 Snapshot 降 ı Copy snapshots to start values 😹 🖳 Load start values as actual values 🖡 🕄														
	Recip	e													
	Na	me	Data type	Offset	Start value	Monitor value	Retain	Accessible f	Writa	Visible in	Setpoint	Supervis	Comment		
1		Static													
2		Controlbyte	Byte	0.0	16#0	16#05									
з		RecipeName	String[10]	2.0		'Test01'									
4		Parameter1	Bool	14.0	false	FALSE									
5		Parameter2	Int	16.0	0	43									
6		Parameter3	Real	18.0	0.0	3.7									

Die Tabelle in der Datenbank sieht nun wie folgt aus:

(1 5)	Jnn	ame	d\sim	_plc\recipe\ -	HeidiSQL 11.3.0.	6295							-		×
File	file Edit Search Query Tools Go to Help														
	-	Ň		🗎 👈 🖨	0 - 👥	☴ 0 4 ▶	0 😣 🥥	×►	- 🗀 - 🗐 🛛		🔍 🤞 🔥 🔝 🚍 🎸	, X		Donat	e
1			Host:	127.0.0.1	Database: sim_	olc 📑 Table: reci	pe 🔠 Data	🕨 Qu	ery 🐻						
^		im_	plc.red	ipe: 3 rows to	otal (approximate	ly)		≫ Ne	xt 🔶	Show al	 Sorting 	💌 Columns (6/6)	Filter	
		ld	9	Timestamp	9	RecipeName	Parameter	1 🥊	Parameter2	💡 Pa	irameter3 💡				
			2	2022-10-18	09:24:27.715	Test02		1		47	2,8				
			3	2022-10-18	09:25:07.727	Test03		1		53	-1,7				
	Ш		4	2022-10-18	09:30:01.216	Test04		0		22	1,2				
$\langle \rangle$		×	Filter	Regular exp	pression]									
48		SEL	ECT '	* FROM `sin	_plc`.`recip	≥` LIMIT 1000;									\$
					r1:c1	Connected:	00 🛹 MariaDi	B 10.4.22	Uptime	5 days,	11:09 h 🕔 Server ti	me: 09 🔿 Idle.			

Anzeige Datenbaustein

Um auszulesen welche Rezepturen in der Datenbank vorhanden sind, kann ein Anzeige-Datenbaustein erstellt werden. Dieser wird dann von **HS**DBASE beschrieben.

IS HSDBASE 4.20 - manual*			_		×
File Edit Extras Help					
🗋 💕 🛃 🎉 🖻 🛍 🔊 🕫 🖍 🥁 🕂 –					
E RecipeExample	tus				
	Mode:	Read and Write	✓ ID column		
Start data	Stop data min cycle(ms):	500 ID na	me: Id		-
capture	Control byte address:	DB2.DBB0 Control byte status: 5	DisplayDB	10	ר
		Γ	Where Clause DB:		
Savatina					=1
Save type	Mysol -	Write new table	every: Once		, II
	injour .	Table annual Paging			
		Table name: Inecipe			
Server	localhost	User-ID: root	Te	st MySQL	
Database	SIM_PLC	Password:		Jimeedon	
			alternative storage lo	scation 🥅	
	Write new database/folder depending on varia	able 🗌 Write new	table depending on v	ariable 🥅	
2022-10-18 09:11:48.191: C:\Users\HS\Desktop\manual.hsd loaded					
2022-10-18 09:12:52.994: SIM_PLC: Connection established					

Der Anzeige DB muss wie folgt aufgebaut sein:

HSD	HSDBASE → SIM_PLC [CPU 1515-2 PN] → Program blocks → DisplayDB [DB10]													
*	🖉 🕐 🔩 😹 🗮 🖤 Keep actual values 🔒 Snapshot 🦄 🧐 Copy snapshots to start values 🔹 🚱 Load start values as actual values 🕷 🚱													
0	DisplayDB													
	Name	Data type	Offset	Start value	Retain	Accessible f	Writa	Visible in	Setpoint	Supervis	Comment			
1 +	🔟 🔻 Static													
2 ┥	nr_of_datasets	Dint 🔳	0.0	0										
з +	🔟 = start_datatset	Dint	4.0	0										
4 +	🔟 = datasets_in_db	Dint	8.0	0										
5 🔫	🔟 = 🕨 recipes	Array[125] of String[10]	12.0											

Die ersten drei Einträge müssen vom Typ DInt sein. Der Array kann beliebig lang sein, muss aber dem Datentyp des verwendeten Bezeichners der Datensätze entsprechen.

Wird das Controlbyte auf 121 (AnzeigeDB füllen) gesetzt, schreibt **HS**DBASE die Anzahl aller Datensätze in der Datenbank in das erste Datendoppelwort. Im dritten Datendoppelwort steht die Anzahl der Datensätze, die in den Datenbaustein geschrieben wurden. Der Rest des DBs wird mit den Bezeichnern der Datensätze aufgefüllt. **HS**DBASE bestimmt automatisch wie viele Bezeichner in den Baustein passen. Sollten mehr Datensätze in der Datenbank vorhanden sein, als in den DB passen, kann der Beginn des Anzeigebereichs über das zweite Datendoppelwort vorgegeben werden. Sind weniger Datensätze vorhanden, wird der restliche DB mit w#16#00 gefüllt.

Nach dem Steuern des Controlbytes zeigt der Anzeige DB nun die vorhandenen Datensätze an:

HSI			SIM_PLC [CPU 1]											_ ⊫∎×
	2	١,	, 🋃 🗮 😤 Keep	actual values 🔋 🔒 Snapshot	🖣 🖳 (Copy snapshots t	ostart values 🛛 🖳 🖳	Load start	values as actua	l values	e , 8,			=
	Disp	olay	DB											
	1	Nam	e	Data type	Offset	Start value	Monitor value	Retain	Accessible f	Writa	Visible in	Setpoint	Supe Comment	
1	•	• s	tatic											^
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[125] of String[10]	12.0									
6			recipes[1]	String[10]	12.0		'Test02'			V				
7			recipes[2]	String[10]	24.0		'Test03'		V	V	¥			
8	-		recipes[3]	String[10]	36.0		'Test04'		V	V	V			
9			recipes[4]	String[10]	48.0				¥	 Image: A start of the start of				~

WhereClause Datenbaustein

Um die Möglichkeit zu schaffen, aus einer Datenbank nur Datensätze auszuwählen, die bestimmte Bedingungen erfüllen, gibt es in **HS**DBASE die Möglichkeit in einem weiteren DB Bedingungen einzugeben, die bei der SQL-Abfrage von **HS**DBASE in einer "Where Clause" angehängt werden.

IS HSDBASE 4.20 - manual*					- 🗆 ×
File Edit Extras Help					
i 🗅 💕 🔛 i 🕹 🖻 🛍 🔊 🕅 📗	■ 🛧 🖊 🕂 👘				
RecipeExample	Settings Variables Status	5			
Recipe	Capture		Mode: Read and Write	▼ ID co	lumn
	Start data	Stop data min c	/cle(ms): 500	ID name: Id	
	capture	Control byte	address: DB2.DBB0 Con	trol byte status: 7 🔽 Displ	ayDB 10
				✓ Where Clause	a DB: 11
	Save type				
	Save as: My	AySQL 👻		Write new table every: Once	e 🗸
			Table nam	e: Recipe	
	Server: lo	ocalhost	User-II	D: root	Test MySQL
	Database: SI	SIM_PLC	Passwor	d:	Connection
				alternative st	torage location
	l w	Write new database/folder depend	ing on variable 🕅	Write new table dependi	ng on variable 🕅
2022-10-18 09:11:48.191: C:\Users\HS\Deskt	op \manual.hsd loaded				
2022-10-18 09:12:52.994: SIM_PLC: Connectio	on established				
L					

Der WhereClause-DB besteht aus beliebig vielen Strings. Mit diesen Strings wird der interne SQL-Befehl in **HS**DBASE hinter dem Schlüsselwort WHERE erweitert. Ist der erste String leer, findet die Abfrage ohne "WhereClause" statt. Bei der Erstellung der Strings ist die entsprechende SQL-Syntax der gewählten Datenbank zu beachten. Eine Überprüfung findet nicht statt.

Einfache Bedingung für das vorherige Beispiel:

HS	DBA	SE	SIM_PLC [CPU 15	515-2 PN] 🕨 Program	blocks 🕨	WhereClause	[DB11]								_ ⊫ ■ ×
=	₹¢	2	, 🛃 🗮 🚏 Кеер	actual values 🔒 Sna	pshot 🏛	🛉 💐 Copysna	pshots to start values	🛃 🖳 Lo	ad start values	as actual	values 其	Ð,			
	Whe	ereC	lause (snapshot cr	eated: 9/28/2022 2:5	3:56 PM)										
	- 1	Name	e	Data type	Offset	Start value	Monitor value	Retain	Accessible f	Writa	Visible in	Setpoint	Supervis	Comment	
1		▼ S ¹	tatic												^
2	-	• •	Where	Array[010] of String	0.0										=
з	-		Where[0]	String	0.0		'Parameter1 = 1'			V					
4	-		Where[1]	String	256.0						Image: A start and a start				~
	<							111							>

Nach Steuern des Controlbytes auf 121 (Anzeige BD füllen) wird der AnzeigeDB nur mit den Datensätzen gefüllt, die die eingestellten Bedingungen erfüllen.

HS	DBA	SE	SIM_PLC [CPU 151	5-2 PN] > Program	blocks	DisplayDB [D	0B10]								_ ₪ ■ ×
1	1	۹,	🕻 🏷 🚬 🎇 Keepa	ctual values 🛛 🔒 Sn	apshot 🏾	🛉 🖳 Copysna	pshots to start values	R- R- L	.oad start value	s as actu	al values 🛛 📃	, B,			_
	Dis	play	DB												
		Nam	e	Data type	Offset	Start value	Monitor value	Retain	Accessible f	Writa	Visible in	Setpoint	Supervis	Comment	
1	-11	▼ S	tatic												^
2	-	•	nr_of_datasets	DInt	0.0	0	2								=
З	-	•	start_datatset	DInt	4.0	0	0								
4	-0	•	datasets_in_db	DInt	8.0	0	2								
5	-01	• •	 recipes 	Array[125] 🔳 💌	12.0										
6	-		recipes[1]	String[10]	12.0		'Test02'		V	V	V				
7	-		recipes[2]	String[10]	24.0		'Test03'		 Image: A start of the start of	V	V				
8	-		recipes[3]	String[10]	36.0										~

Modus Ringspeicher

In diesem Modus werden die Daten im Arbeitsspeicher des Rechners, auf dem **HS**DBASE läuft gesammelt und erst auf Befehl in die Datenbank geschrieben. Der Ringspeicher wird wie im Modus Lesen und Schreiben über ein Controlbyte gesteuert.

Alle Codes im Überblick:

	Kommandocodes
101	Ringspeicher löschen
102	Ringspeicher füllen (wird nicht quittiert)
103	Ringspeicher abspeichern
	Antwortcodes
1	Ringspeicher wurde gelöscht
2	Ringspeicher wurde gespeichert (und gelöscht)
11	unbekannter Befehl

Modus QDAS

Der folgende Abschnitt zeigt anhand eines Beispiels, wie **HS**DBASE konfiguriert werden muss, um Daten im QDAS-Datenformat DFQ abzuspeichern. Diese können dann bspw. mithilfe des QDAS-Uploadtools automatisch in die QDAS-Datenbank eingelesen werden.

Die für QDAS benötigten K-Nummern sind in der Spalte Name einzutragen. Pro Trigger wird eine separate DFQ-Datei mit fortlaufender Nummer generiert.

IS HSDBASE 3.71 - q	S HSDBASE 3.71 - qdastest*													
File Edit Extra	s He	lp												
i 🗅 📂 🔜 🐰 🖻	a 🛍	9 CH	6 🐻 🛧 🗟	▶ +										
	Setting	ys Variable:	s Status											
PLC Bsp		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					
	*													
2014-01-03 11:33:35.65 2014-01-03 11:34:29 66	5: D:\Ar 3: X3: V	beit \HS Allg	emein\QDAS\qd	lastest.hsd geladen	t TCP-Verbindungsaufbau				4					
2014-01-03 11.34.23.00	ы. ла. V	cibindung K	onine monit durge	ebaar werden - rimeor	a ror-verbindungsaurbau				*					

Kontakt

Für noch offene Fragen sind wir jederzeit für Sie erreichbar.

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