



BüroWARE

SQL Funktionen

Inhalt

SQLGET:3

SQLPUT:4

SQLEXEC:5

SQLSELECT:6

SQL-Statements in Kalkulationen (ab Version 5.4)

Es gibt nun neben GET 910/912 und PUT 430/432/436 zusätzliche Möglichkeiten, selbst definierte SQL-Statements innerhalb von Kalkulationen auszuführen.

SQLGET

Mit SQLGET ist der Lesezugriff auf einzelne Spalten einer SQL-Tabelle oder einer Sicht möglich.

Syntax:

```
<Variable> =  
SQLGET[Tabellenname!Selektion!Sortierung!MaxZeilen!Ausgabedateiname!  
Spalte 1!Spalte 2!...!Spalte n]
```

Tabellenname

Der Name der SQL-Tabelle oder der Sicht, auf die der Zugriff erfolgen soll (z.B. RARTIK). Dem Namen können Zusatzinformationen nach folgender Syntax mitgegeben werden:

- Es kann ein Datenbank- und/oder Schemaname vorangestellt werden. Anstelle eines fixen Namens können auch die Systemvariablen \$(SYSDDB), \$(MAILDB), \$(MANDDB) verwendet werden.
- Es können Verbindungsdaten für den SQL-Server mittels @ angehängt werden: Hierbei sind alle Komponenten optional:

```
@SERVER=<Server>;UID=<Name>;PWD=<Kennwort>
```

Selektion (optional)

Eine gültige SQL-WHERE-Klausel (ohne "WHERE") .

BW-Einschränkung: Textlitterale müssen in einfache Anführungszeichen eingeschlossen werden.

Sortierung (optional)

Eine gültige SQL-ORDER BY-Klausel (ohne "ORDER BY").

BW-Einschränkung: Kommata müssen durch Doppelpunkte ersetzt werden, also z.B.

```
ART_36_5 ASC:ART_1_25 ASC
```

Wird dieser Parameter leer gelassen erfolgt keine Sortierung

MaxZeilen (optional)

Maximalanzahl zurückgegebener Zeilen, wird dieser Parameter weggelassen oder ist 0, wird die erste passende Zeile zurückgegeben.

BW-Einschränkung: insgesamt kann das Resultat maximal 5119 Zeichen groß sein (incl. Trennzeichen), sofern die Ausgabe nicht in eine Datei erfolgt.

Ausgabedateiname (optional)

Anstatt in den Speicherpuffer wird das Ergebnis in die spezifizierte Datei geschrieben. In der Zielvariable steht bei Erfolg "1", bei Fehler "0".

Spalte 1 ... Spalte n

Namen der zurückgelieferten Spalten. Wenigstens eine Spalte muss angegeben werden, weitere sind optional.

Hinweis: Um zu verhindern, dass Variablennamen von der BüroWARE aufgelöst werden, können diese in runde Klammern gesetzt werden

Beispiel:

```
TMP_0_85=SQLGET[RARTIK!SUBSTRING(ART_1_25,1,3)="CPU" AND  
(ART_36_5)="HW"! (ART_36_5) DESC: (ART_1_25)!1!! (ART_1_25)! (ART_51_60) ]
```

SQLPUT

Mit SQLPUT ist der Schreibzugriff auf einzelne Spalten einer SQL-Tabelle oder eines Views möglich.

Syntax:

```
<Variable> =  
SQLPUT[Tabellenname!Selektion!Spalte 1=Wert 1!...!Spalte n=Wert n]
```

Tabellenname

Der Name der SQL-Tabelle oder der Sicht, auf die der Zugriff erfolgen soll (z.B. RARTIK). Dem Namen können Zusatzinformationen nach folgender Syntax mitgegeben werden:

- Es kann ein Datenbank- und/oder Schemaname vorangestellt werden. Anstelle eines fixen Namens können auch die Systemvariablen \$(SYSDB), \$(MAILDB), \$(MANDDB) verwendet werden.
- Es können Verbindungsdaten für den SQL-Server mittels @ angehängt werden: Hierbei sind alle Komponenten optional: @SERVER=<Server>;UID=<Name>;PWD=<Kennwort>

```
@SERVER=<Server>;UID=<Name>;PWD=<Kennwort>
```

Selektion (optional)

Eine gültige SQL-WHERE-Klausel (ohne "WHERE") . Wenn angegeben, wird ein SQL-UPDATE ausgelöst, andernfalls erfolgt ein SQL-INSERT.

BW-Einschränkung: Textlitterale müssen in einfache Anführungszeichen eingeschlossen werden.

Spalte 1 = Wert 1 ... Spalte n = Wert n

Zuweisen der Spaltenwerte.

BW-Einschränkung: Textlitterale müssen in einfache Anführungszeichen eingeschlossen werden.

Hinweis: Um zu verhindern, dass Variablennamen von der BüroWARE aufgelöst werden, können diese in runde Klammern gesetzt werden

Beispiel:

```
TMP_0_1=SQLPUT[RARTIK!ID=123!(ART_51_60)="Testartikel"!
(ART_36_5)="WG001"]
```

Rückgabewert: „0“ bei Fehler, „1“ sonst.

SQLEXEC

Mit SQLEXEC ist der Aufruf eines frei definierbaren SQL-Scripts möglich. Scripte müssen in der IDB SE0205 hinterlegt werden und können auch aus mehreren SQL-Statements bestehen. In dem Fall müssen die einzelnen Statements durch die Angabe von „go“ getrennt werden. „go“ muss alleine und am Anfang einer Zeile stehen.

Vor der Scriptausführung können Parameter an das Script übergeben werden, dazu wird der globale MEM-Bereich verwendet. Der Zugriff aus dem SQL-Script auf die MEM-Variablen erfolgt mittels Variablen der Form \$(MEM_pos_len).

Man kann im Script auch die Parameter \$()

Syntax:

```
<Variable> =
SQLEXEC[ScriptIdent!Connect!StopBeiFehler!reserviert!Param1!...!Paramn]
```

Script-Ident

Der eindeutige Script-Identifizierer, dies entspricht dem Primärindex des Scriptes in der IDB SE0205.

Verbindung (optional)

Wird hier nichts angegeben, wird eine Verbindung zur aktiven Mandantendatenbank aufgebaut. Andernfalls können Verbindungsdaten in folgender Form angegeben werden:
SERVER=<Server>; DATABASE=<Datenbank>; UID=<Name>; PWD=<Kennwort>

StopBeiFehler (optional)

= 0: Skriptverarbeitung fortsetzen, wenn ein Statement fehlschlägt.
= 1: Skriptverarbeitung abbrechen, wenn ein Statement fehlschlägt (Standardverhalten)

Asynchron (optional)

= 0: Es wird gewartet, bis das komplette Script abgearbeitet ist. (Standardverhalten)
= 1: Die Scriptausführung findet in einem Hintergrundthread statt, es wird nicht gewartet.

Param 1 ... Param n

Übergabe von Werten/Feldern an die Parameter im SQL Statement der IDB.

SQLSELECT

Mit SQLSELECT kann ein frei definierbares SQL-Script zu Abfrage von Werten aufgerufen werden. Scripte müssen in der IDB SE0205 hinterlegt werden und können auch aus mehreren SQL-Statements bestehen. In dem Fall müssen die einzelnen Statements durch die Angabe von „go“ getrennt werden. „go“ muss dabei alleine in und am Anfang einer Zeile stehen. Das letzte SELECT-Statement des Scripts bildet den Rückgabewert.

Wie bei SQLEXEC können vor der Scriptausführung Parameter an das Script übergeben werden, dazu wird auch hier der globale MEM-Bereich verwendet. Der Zugriff aus dem SQL-Script auf die MEMVariablen erfolgt mittels Variablen der Form \$(MEM_pos_len).

Syntax:

```
<Variable> =
SQLSELECT[ScriptIdent!Connect!Ausgabedateiname!Spaltentrenner!Zeilentrenner
!StopBeiFehler!Param1!...!Paramn]
```

Script-Ident

Der eindeutige Script-Identifizierer, dies entspricht dem Primärindex des Scriptes in der IDB SE0205.

Verbindung (optional)

Wird hier nichts angegeben, wird eine Verbindung zur aktiven Mandantendatenbank aufgebaut. Andernfalls können Verbindungsdaten in folgender Form angegeben werden:
SERVER=<Server>; DATABASE=<Datenbank>; UID=<Name>; PWD=<Kennwort>

Ausgabedateiname (optional)

Anstatt in den Speicherpuffer wird das Ergebnis in die spezifizierte Datei geschrieben.

Spaltentrenner

Muss angegeben werden, wenn mehrere Spalten abgefragt werden und definiert dann das Trennzeichen zwischen den Spalten. Kann leer bleiben, wenn nur eine Spalte abgefragt wird.

Zeilentrenner (optional)

Definiert das Trennzeichen zwischen den Zeilen des Ergebnisses der Abfrage. Standardwert ist Wagenrücklauf gefolgt von Zeilenvorschub. (ASCII 13 + ASCII 10).

StopBeiFehler (optional)

= 0: Skriptverarbeitung fortsetzen, wenn ein Statement fehlschlägt.
= 1: Skriptverarbeitung abbrechen, wenn ein Statement fehlschlägt
(Standardverhalten)

Param 1 ... Param n

Übergabe von Werten/Feldern an die Parameter im SQL Statement der IDB.