Quick SQL Access Tool

Dokumentation

Inhaltsverzeichnis

Lattwein-Informationen

Dieses Handbuch wird als Anleitung für die Verwendung des QSAT und des SQL-Preprocessors ausgegeben.

Anschrift: Lattwein GmbH
Otto-Brenner-Straße 25

52353 Düren

Telefon: 02421 81051

Telefax: 02421 82127

Auskunft: In der unten angegebenen Arbeitszeit steht den Benutzern des QSAT ein zentraler Telefon-Wartungsdienst zur Verfügung, der bemüht ist, alle Fragen zu beantworten, die in diesem Handbuch nicht behandelt sind.

Arbeitszeit: 08:30 – 16:00 Uhr

Allgemeines

QSAT besteht aus einer Online- und einer Batch-Komponente.

Mit QSAT können online Tabellen beschrieben werden. Aus diesen Tabellen können Teile herausgenommen und als Access abgestellt werden. Diese Accesse dienen dazu, dass zwischen SQL/DB2 und dem Anwendungsprogramm kommuniziert werden kann.

Die im QSAT hinterlegten Datendefinitionen werden vom Preprozessor in das Anwendungsprogramm gestellt. Der Programmierer braucht lediglich anzugeben, welche Accesse er benötigt.

Der Programmierer führt im Programm ein CONNECT durch und kann anschließend die gewünschte WHERE-Clause definieren. Der SQL/DS Befehl kann dann vom HL1/SQL-Preprozessor interpretiert und vom Anwendungsprogramm ausgeführt werden.

Die Sicherungsmechanismen von SQL/DB2 werden in Anspruch genommen. Es kann jederzeit ein neuer transportabler Source Code erstellt werden, der auch bei nicht QSAT-Benutzern umgewandelt werden kann.

Bei HL1-Programmen können in einem Programmablauf gleichzeitig bis zu 100 CURSOR aktiv sein.

In einem HL1-Programm können bis zu 500 Hostvariablen definiert werden.

Es können bis zu 32 CURSOR definiert werden, dabei dürfen jedem CURSOR maximal 32 Hostvariablen als Suchargument zugewiesen werden.

Der Einsatz des SQL-Preprocessors und QSAT setzt die Verwendung des Command-Level-ESA-Modes im CPG3 voraus.

SQL kann sowohl in Batch- als auch in Online-Anwendungen eingesetzt werden.

Auch in HL1-Modulen kann SQL verwendet werden. HL1-Module sind Bausteine, die sowohl von Online- als auch von Batchprogrammen aus aufgerufen werden können.

Hier eine Übersicht über die verschiedenen Feldtypen und ihre Zuordnung:

SQL	CPG
CHAR	Alpha
VARCHAR	Alpha (bis 254 Bytes)
LONG VARCHAR	Alpha (im SQL bis zu einer Länge von 32765 Bytes in CPG max. 256 Bytes)
DECIMAL	Numerisch
INTEGER	Fullword 'F' ...
SMALLINT	Halfword 'H' .
DATE	Alpha (10 Bytes 'D') . Eintrag bei Dez.
TIME	Alpha (10 Bytes 'T') .
TIMESTAMP	Alpha (26 Bytes 'S' ...
FLOAT	Numerisch (werden von SQL/DB2 in numerisches Format konvertiert)
GRAPHIC	z. Z. nicht unterstützt
VARGRAPHIC	z. Z. nicht unterstützt
LONG VARGRAPHIC	z. Z. nicht unterstützt

SQL-Befehle und Definitionen werden nur in der Data Division (D-Karten) und in der Procedure Division (C-Karten) verwendet.

Die Syntax des SQL im CPG-Programm kann dem RPG- oder dem CPG-Formalismus angepasst werden:

- in der Data Division (D-Karten)

```
- -D.
-     SQL BEGIN DECLARE SECTION
-         USER       8
-         :
-     SQLDD ACCNAM
-         :
-     SQL END DECLARE SECTION
```

```
DSQ      BEGIN DECLARE SECTION
D        USER           8
D        :
DACCNAM        SQ
D        :
DSQ      END DECLARE SECTION
```

Bei Verwendung von QSAT werden die für einen Access definierten Felder mit dem Befehl SQLDD ins Programm geholt.

Felder können auch über die sonst üblichen Data-Dictionary-Einträge Angezogen werden.

Hinter dem ersten "SQL END DECLARE SECTION"-Statement werden vom Precompiler CPGPREP 100 Bytes zur Kommunikation mit SQL eingefügt.

- in der Procedure Division (C-Karten)

```
- -C.
-     SQL  CONNECT :USER IDENTIFIED BY :PASSWD


CSQ    CONNECT :USER IDENTIFIED BY :PASSWD
```

Reicht eine Zeile nicht aus, um einen SQL-Befehl abzusetzen, so kann der Befehl durch den Eintrag '*' in Spalte 72 in der nächsten Zeile fortgesetzt werden. Das Ende des SQL-Statements wird am Fehlen des Fortsetzungszeichens in Stelle 72 erkannt.

Beispiel:

```
- -C.
-     SQL  DECLARE BSP CURSOR FOR                                  *
-     SQL  SELECT  KDNR,FIRMA,PLZ,STR                              *
-     SQL  FROM    SQLDBA.KUNDEN                                   *
-     SQL  WHERE   KDNR = :KDNR


CSQ    DECLARE BSP CURSOR FOR                                       *
CSQ    SELECT  KDNR,FIRMA,PLZ,STR                                   *
CSQ    FROM    SQLDBA.KUNDEN                                        *
CSQ    WHERE   KDNR = :KDNR
```

SQL-Statements dürfen nicht mit einem Punkt abgeschlossen werden. Kommentare sind nicht unterstützt.

Anwendung des CPG3 SQL Interface Programms

In einem Anwendungsprogramm können direkt SQL-Befehle abgesetzt werden.Dazu müssen zunächst in der Data Division bzw. in den D-Bestimmungen alle Felder, die als SQL Hostvariable fungieren, definiert werden. Damit der Preprocessor diese Felder erkennt, wird vor den Feldern folgendes SQL Statement eingefügt:

```
-    SQL BEGIN DECLARE SECTION
```

Hinter den Feldern wird dieses SQL Statement eingefügt:

```
-    SQL END DECLARE SECTION
```

Beispiel:
```
- -D.
- SQL BEGIN DECLARE SECTION
-     USER     8.
-     PASSWD   8.
-     KDNR     7.
-     FIRMA.
-     UMSATZ   9 2.
-     PROV    10 0.
- SQL END DECLARE SECTION
```

Die Felder KDNR und FIRMA sind Character-Felder und können Daten aus Character-Feldern oder Varcharacter-Feldern aufnehmen. Das Feld UMSATZ ist ein numerisches Feld und kann Werte aus einem dezimal definierten Feld aufnehmen. Das Feld PROV ist ebenfalls numerisch und kann einen numerischen Wert aus einem dezimal definierten Feld oder aus einem Smallinteger- oder einem Integer-Feld aufnehmen. Es erfolgt eine automatische Umsetzung von SQL. Wichtig ist, dass dieses Feld keine Dezimalstellen hat (0).

Es werden keine Feldgruppen als Hostvariable unterstützt. Sollen Feldgruppen verarbeitet werden, so können über Datenstrukturen Felder zusammengefasst werden.

SQL-Befehle über QSAT

Folgende SQL-Befehle sind im HL1 über QSAT unterstützt.

- SQL CHAIN
identisch zu - SQL GETDIR

- SQL CHAIN-SINGLE

- SQL SETLL
identisch zu - SQL STARTRD

- SQL READ
identisch zu - SQL NEXTRD

- SQL SETWC

- SQL PUTWC

- SQL RNDOM
identisch zu - SQL ENDRD

- SQL WRITE
identisch zu - SQL ADDROW

- SQL UPDAT
identisch zu - SQL UPDROW

- SQL DYNPREP1/2/3 SETLL
- SQL DYNPREP1/2/3 READ
- SQL DYNPREP1/2/3 RNDOM

SQL-Befehle

Hier ist eine Übersicht aufgeführt über die SQL-Befehle, die von HL1 aus aufgerufen werden können:

DECLARE	CURSNAM CURSOR FOR SELECT ...
OPEN	CURSNAM
CLOSE	CURSNAM
FETCH	CURSNAM
DECLARE	CURSNAM CURSOR FOR INSERT ...
PUT	CURSNAM
SELECT	INTO
UPDATE	
INSERT	
DELETE	
ACQUIRE	
ALTER	
CREATE	
COMMENT ON	
COMMIT	
CONNECT	
DROP	
GRANT	
LABEL ON	
LOCK	
REVOKE	
ROLLBACK	
UPDATE	STATISTICS
DYNPREP1/2/3	SELECT
DYNPREP1/2/3	OPEN
DYNPREP1/2/3	FETCH
DYNPREP1/2/3	CLOSE

Erweiterte SQL-Befehle über QSAT

Es sind erweiterte Befehle im HL1 unterstützt, wenn QSAT installiert ist. Dabei entspricht ein HL1-Befehl teilweise mehreren nacheinander auszuführenden SQL-Befehlen.

Für den Programmierer besteht damit die Möglichkeit, im Programm die vom HL1 gewohnten Befehle zu verwenden, womit die Lesbarkeit des Programms verbessert wird.

CHAIN	DECLARE OPEN FETCH CLOSE
CHAIN-SINGLE	SELECT ... INTO
DYNPREP1/2/3 SETLL	SELECT OPEN
DYNPREP1/2/3 READ	FETCH
DYNPREP1/2/3 RNDOM	CLOSE
READ	FETCH
RNDOM	CLOSE
SETLL	DECLARE OPEN
UPDAT	UPDATE
WRITE	INSERT

SQL - wichtigste Befehle in der Anwendung

In einer Anwendung muss sich der Benutzer zunächst dem SQL bekannt geben. Dazu ist ein CONNECT notwendig. Wird kein CONNECT gegeben, so wird automatisch ein CICS-USR angemeldet. Für Batch-Programme muss auf jeden Fall ein CONNECT gegeben werden.

Das CONNECT gilt auch für alle HL1-Module, die angesprochen werden.

Beispiel:
```
- -C.
-       USER   = 'SQLDBA  '
-       PASSWD = 'SQLDBAPW'

-       SQL CONNECT :USER IDENTIFIED BY :PASSWD

-       IF SQCODE >< 0
-          GOTO ERROR
-       END
```

Den Hostvariablen wird ein Doppelpunkt vorangestellt, damit der Preprocessor erkennt, dass es sich in diesem Fall um eine Hostvariable handelt.

Nach Durchführung eines SQL-Statements sollte der SQL-Return-Code abgefragt werden. Das geschieht, indem ein vom Preprocessor definiertes Feld SQCODE auf den Wert Null abgefragt wird. Wenn der Wert Null ist, wurde vom SQL kein Fehler gefunden und der Befehl ordnungsgemäß durchgeführt.

Der User muss für das aufgerufene Programm berechtigt sein, sonst wird der SQCODE = -551 gesetzt.

Um dem User für das entsprechende Programm eine Berechtigung zu geben, muss der Owner des Programms/Packages einen

```
GRANT RUN ON Program/Package TO Username/Public
```

geben.

Auch für die Befehle SELECT, UPDATE, INSERT, DELETE muss eine Berechtigung des Users vorliegen, eine Tabelle oder View zu bearbeiten, sonst erfolgt ein Abbruch mit SQCODE = -551.

Um in diesem Fall dem User die entsprechende Berechtigung zu erteilen, muss vom Owner der Tabelle/View ein

```
GRANT TABLE PRIVILEGES ON Tablename TO Username/Public
```

gegeben werden.

Wird mit mehreren Datenbankadministratoren gearbeitet, so müssen Eventuell mehrere GRANTs gegeben werden.

Ist der Creator der Tabelle derselbe wie der Creator des Programms, so braucht er nur ein

```
GRANT RUN ON Programmname TO Username/Public
```

zu geben.

Wurde allerdings die Tabelle bzw. View von einem anderen DBA erstellt als das Programm, so müssen folgende Befehle ausgeführt werden:

```
GRANT ALL PRIVILEGES ON Tablename TO Xdba2 WITH GRANT OPTION
GRANT CONNECT TO Programmname IDENTIFIED BY PROGNAME
GRANT RUN ON Programmname TO Username/Public
```

Dabei ist zu beachten, dass der erste Befehl nur von dem Creator der Tabelle gegeben werden kann. Der zweite Befehl kann nur vom Creator des Programms gegeben werden.

Sollte das Programm versehentlich umgewandelt worden sein, ohne dass eine entsprechende Berechtigung bestand, so sind folgende Schritte vorzunehmen:

```
DROP Programm/Package ...
```

Anschließend muss das Programm erneut durch den Preprozessor laufen, damit dem Programm GRANT Autorität in der SYSTEM.SYSPROGAUTH vergeben wird.

Der SQL-Lesezugriff über CURSOR

Sollen mit SQL Tabellen gelesen werden, so unterstützt der Preprocessor den Zugriff über CURSOR. Das heißt: Der Programmierer definiert einen CURSOR, führt einen OPEN für diesen CURSOR durch, liest dann mit einem FETCH-Befehl die Daten in sein Programm und - wenn keine weiteren Daten verfügbar sind - beendet den Zugriff mit einem CLOSE für diesen CURSOR. Im Beispiel heißt der definierte Cursor CCCC.

Bei HL1-Programmen können in einem Programmablauf gleichzeitig bis zu 100 CURSOR aktiv sein.

Wenn der komplette SQL-Befehl nicht in eine Zeile passt, wird in Stelle 72 ein Stern (oder ein Zeichen ungleich Blank) gesetzt. Vom Preprozessor wird nun eine Folgezeile erwartet.

Beispiel:

```
- SQL  DECLARE  CCCC CURSOR  FOR                                  *
- SQL  SELECT KUNDNR,FIRMENBEZ,JAHRESUMSATZ                      *
- SQL           FROM SQLDBA.KUNDEN                               *
- SQL  WHERE  KUNDNR = :KDNR

- SQL  OPEN CCCC

- DO UNTIL SQCODE >< 0                                           *
-    SQL FETCH  CCCC
-    SQL INTO   :KDNR, :FIRMA, :UMSATZ
- END

- SQL  CLOSE  CCCC
```

Durch einen Zugriff können ein oder mehrere Sätze gelesen werden. Es hängt ausschließlich vom Schlüsselbegriff ab, wie eng die Auswahl getroffen wurde. Der Returncode SQCODE wird auf '100' gesetzt, wenn kein Satz gefunden wurde.

Die Reihenfolge DECLARE ... OPEN ... FETCH für einen Cursor muss im Programm unbedingt eingehalten werden, da der Source Code vom Preprozessor nur einmal verarbeitet wird.

Direktes Lesen mit SELECT ... INTO

CPG unterstützt den Befehl SINGLE ROW SELECT zum Direktzugriff.

Dieser SELECT-Befehl kann immer dann angewendet werden, wenn sicher ist, dass nur eine Zeile als Ergebnis der WHERE-Clause die Bedingungen erfüllt.

Beim SINGLE ROW SELECT wird kein Cursor definiert.

Sollte mehr als eine Ergebnis-Zeile erzielt werden, erscheint der SQL Fehlercode -810. An das Programm wird nur eine Zeile übertragen.

Beispiel 1:

```
- SQL  SELECT KUNDNR,FIRMENBEZ,JAHRESUMSATZ                    *
- SQL           INTO :KDNR, :FIRMA, :UMSATZ                    *
- SQL           FROM SQLDBA.KUNDEN                             *
- SQL  WHERE   KUNDNR = 04711
```

Beispiel 2:

```
- SQL  SELECT MAX(PLZ)                                         *
- SQL         INTO :MPLZ                                       *
- SQL         FROM SQLDBA.KUNDEN
```

Der SQL-Schreibzugriff über CURSOR

Auch beim Hinzufügen von Zeilen in SQL-Tabellen wird der Zugriff über CURSOR unterstützt. Dazu definiert der Programmierer einen CURSOR, führt einen OPEN für diesen CURSOR durch und setzt dann mit PUT die Hostvariablen in die Tabelle.

Bei HL1-Programmen können in einem Programmablauf gleichzeitig bis zu 100 CURSOR aktiv sein.

Diese Form des Einfügens ist vor allem dann sinnvoller, wenn es sich um Gruppeneinfügungen von Zeilen handelt (d. h. wenn viele Zeilen eingefügt werden). Diese Operation ist performancegünstiger als das direkte Hinzufügen mit INSERT. Die Programmierung kann mit und ohne QSAT durchgeführt werden.

Beispiel ohne QSAT:

```
- SQL  DECLARE  CCCC CURSOR  FOR
- SQL  INSERT INTO Tabellenname
- SQL         (KUNDNR,FIRMENBEZ,JAHRESUMSATZ)
- SQL  VALUES (:KDNR,:FIRMA,:UMSATZ)

- SQL  OPEN CCCC

- DO UNTIL SQCODE >< 0
-    SQL PUT CCCC FROM :KDNR,:FIRMA,:UMSATZ
- END

- SQL  CLOSE CCCC
```

Beispiel mit QSAT:

```
- SQL  SETWC CCCC Accnam

- DO UNTIL SQCODE >< 0
-    SQL PUTWC CCCC Accnam
- END

- SQL  CLOSE CCCC
```

Direktes Schreiben mit UPDATE, INSERT, DELETE

Update erfolgt über den Befehl UPDATE, Löschen über den Befehl DELETE und Hinzufügen von Zeilen über den Befehl INSERT.

Diese Befehle entsprechen den SQL-Befehlen, wie sie auch für andere Programmiersprachen gelten (z. B. COBOL, FORTRAN, ASSEMBLER).

Beispiel:

```
- SQL   UPDATE SQLDBA.KUNDEN SET FIRMENBEZ = :FIRMA,          *
- SQL          JAHRESUMSATZ = :UMSATZ                        *
- SQL          WHERE KUNDNR = :KDNR


- SQL   INSERT INTO SQLDBA.KUNDEN                            *
- SQL         (KUNDNR,FIRMENBEZ,JAHRESUMSATZ)                *
- SQL          VALUES (:KDNR, :FIRMA, :UMSATZ)


- SQL   DELETE FROM SQLDBA.KUNDEN WHERE KUNDNR = :KEY
```

Achtung: Es ist wichtig, dass beim UPDATE und beim DELETE immer eine WHERE-Clause definiert wird, da sonst die gesamte Tabelle geändert bzw. gelöscht wird.

Verarbeitung von Null-Feldern

Falls in einer Tabelle Null-Werte in Feldern abgespeichert werden, so können diese nur in der folgenden Form verarbeitet werden :

1. In der Declare Section müssen sogenannte Indicator-Variablen als Smallinteger definiert werden.

```
 - SQL    BEGIN DECLARE SECTION
 -        KDNR         5
 -        FIRMA       25
 -        FBEZ        24
 -        IFBEZ        2 HALF
 - SQL    END DECLARE SECTION
```

2. Im FETCH Befehl bei einer SQL-Anweisung muss diese Indicator-Variable direkt als Anhang der Hostvariablen definiert sein.

```
   - SQL    FETCH CURS001 INTO
   - SQL    :KDNR, :FIRMA, :FBEZ:IFBEZ
```

3. Wenn IFBEZ den Wert X'FFFF' enthält, so ist das Feld FBEZ auf NULL im SQL gesetzt. Enthält IFBEZ den Wert X'0000' , so hat das Feld FBEZ einen gültigen Inhalt.

```
  -        IF IFBEZ = X'FFFF'
  -           FILL  ' ' TO FBEZ
  -           MOVEL '?' TO FBEZ
  -        END
```

4. Bei Update- oder Insert-Befehlen kann die Indicator-Variable ebenfalls verwendet werden, um Null-Werte abzuändern oder einzufügen. Hierbei muss die Indicator-Variable entsprechend mit X'FFFF' (=NULL) oder mit X'0000' (= nicht NULL) gefüllt werden.

```
-      IF FBEZ = '? '
-         FILL X'FF' TO IFBEZ
-      END
- SQL  UPDATE KUNDEN SET KFIRMA= :FIRMA, KFBEZ= :FBEZ:IFBEZ
```

```
-      IF FBEZ = '? '
-         FILL X'FF' TO IFBEZ
-      END
- SQL  INSERT INTO KUNDEN (KFIRMA, KFBEZ)
- SQL     VALUES (:FIRMA, :FBEZ:IFBEZ )
```

Integer- und Smallinteger-Felder

Wenn im SQL Integer- oder Smallinteger-Felder definiert werden, so werden die Hostvariablen als Halbwort oder Vollwort definiert und enthalten im binären Format die Datenwerte. CPG kann aber nur mit gepackten Datenfeldern rechnen. Um Integer- und Smallinteger-Felder ins gepackte Format zu übersetzen, wird vom CPG das Macro SCONVD mitgeliefert.

Beispiel:

```
-  SINT   2 H.
-  FINT   4 F
-  PAC5   5 0.
-  PACA  10 0.

-  MACRO SCONVD  SINT,PAC5        * Konvertiere Smallinteger
                                    nach gepackt
-  MACRO SCONVD  FINT,PACA        * Konvertiere Integer nach
                                    gepackt
```

Umgekehrt können mit dem Macro SCONVB die gepackten Datenfelder wieder ins binäre Format umgesetzt werden. Dies geschieht wie folgt:

```
-  MACRO SCONVB  PAC5,SINT        * Konvertiere gepackt nach
                                    Smallinteger
-  MACRO SCONVB  PACA,FINT        * Konvertiere gepackt nach
                                    Integer
```

Beispielprogramm

Im folgenden wird ein Programm aufgelistet, mit dem ein Zugriff auf die SQL-Tabelle KUNDEN durchgeführt wird. Es werden die Spalten KKDNR und KFIRMA gelesen und am Bildschirm angezeigt.

```
- OPTIONS MAIN ESA TITLE SQL#TESTPROGRAMM PHASE TST026.
- -D.
-   SQL   BEGIN DECLARE SECTION
-         USER        8
-         PASSW       8
-         PROGN       8
-         KDNR        5 0
-         FIRMA      30
-   SQL   END DECLARE SECTION
- -C.
-   MOVEL 'SQLDBA  ' TO USER
-   MOVEL 'SQLDBAPW' TO PASSW
-   SQL CONNECT  :USER IDENTIFIED BY :PASSW
-   SQL DECLARE  C2 CURSOR FOR                                    *
-   SQL SELECT   KKDNR,KFIRMA                                     *
-   SQL FROM     SQLDBA.KUNDEN                                    *
-   SQL WHERE    KKDNR < 3000
-   SQL OPEN     C2
-   DO UNTIL SQCODE >< 0
-      SQL FETCH C2 INTO :KDNR,:FIRMA
-      MAPD ANZEIGE
-      IF COND P3
-         BREAK
-      END
-   ENDDO
-   SQL CLOSE C2
-   SQL COMMIT WORK
```

Beispiel Umwandlungsprozedur

Hier ist ein Muster einer Umwandlungsprozedur für ein CPG-Programm mit SQL-Statements aufgelistet.

```
// JOB CPGSQL
// LIBDEF PROC,SEARCH=(PRD2.SQL220)
// EXEC PROC=ARISLIBP   *-- SQL/DS PRODUCTION LIBRARY ID PROC
// ON $ABEND GOTO REASS
// EXEC CPGPREP,SIZE=512K,PARM='USERID=SQLDBA/SQLDBAPW'
        :
        :   Hier Programm einfügen
        :
/*
// IF $RC NE 0 THEN
// GOTO ENDE
*  STEP HL1
// LIBDEF PHASE,CATALOG=SP4U.ULIBL
// DLBL IJSYSIN,'F4.WORK.04',0,SD,,CISIZE=8192
// EXTENT SYSIPT,PRD201,1,0,46000,4000
ASSGN SYSIPT,122
// EXEC HL1
/*
// IF $RC NE 0 THEN GOTO
// GOTO REASS
*  STEP ASSEMBLER
CLOSE SYSIPT,READER
ASSGN SYSIN,122
*  STEP LNKEDT
// EXEC LNKEDT
/*
// IF $RC EQ 0 THEN
// GOTO ENDE
/. REASS
CLOSE SYSIPT,READER
/. ENDE
/&
```

SQL-Benutzer

Soll die Umwandlung mit einem anderen SQL-Benutzer als SQLDBA durchgeführt werden, so sind folgende Schritte nach der Installation zu beachten:

1. Anlegen eines Users, der CPGPREP aufrufen darf z. B. PREP. Der ISQL-Befehl dazu lautet:

   ```
   GRANT CONNECT TO PREP IDENTIFIED BY PREPPW
   ```

2. Der User PREP soll das Programm CPGPREP benutzen dürfen. Dazu muss ihm das Privileg der Run-Autorität gegeben werden.

   ```
   GRANT RUN ON SQLDBA.CPGPREP TO PREP
   GRANT RUN ON SQLDBA.CPGPREP2 TO PREP
   ```

3. Alle von QSAT benutzten Tabellen und Views sollten dem User PREP zur Benutzung bereitstehen. Dazu die Befehle:

   ```
   GRANT SELECT ON SQLDBA.CPGADD TO PREP
   GRANT SELECT ON SQLDBA.CPGFDD TO PREP
   GRANT SELECT ON SQLDBA.CPGSDD TO PREP
   GRANT SELECT ON SQLDBA.CPGTDD TO PREP
   GRANT SELECT ON SQLDBA.CPGZDD TO PREP
   ```

4. Alle Tabellen/Views, die im Anwendungsprogramm angesprochen werden, müssen dem User PREP zur Verfügung stehen, und zwar mit dem Service, den das Programm durchführen will:

   ```
   GRANT SELECT,INSERT,DELETE,UPDATE ON SQLDBA.USERTAB TO PREP
   ```

Wenn alle 4 Schritte eingehalten werden, kann der User PREP als Umwandlungsadministrator eingesetzt werden.

Sollen mehrere Benutzer auf QSAT zugreifen können (SQLDBA ist automatisch autorisiert), so sind folgende Schritte zu beachten:

1. Anlegen des Users, der QSAT aufrufen darf z.B. UQSAT01.
   Der ISQL-Befehl dazu lautet:

   ```
   GRANT CONNECT TO UQSAT01 IDENTIFIED BY UQSPW
   ```

2. Der User UQSAT01 soll das Programm QSATPRG benutzen dürfen. Dazu muss dem User UQSAT01 das Privileg der Run-Autorität gegeben werden.

   ```
   GRANT RUN ON SQLDBA.QSATPRG TO UQSAT01
   GRANT RUN ON SQLDBA.HQSAL TO UQSAT01
   GRANT RUN ON SQLDBA.HQSAD TO UQSAT01
   ```

3. Alle von QSAT benutzten Tabellen und Views sollen dem User UQSAT01 zur Benutzung bereitstehen. Dazu die Befehle:

   ```
   GRANT SELECT ON SQLDBA.CPGADD TO UQSAT01
   GRANT SELECT ON SQLDBA.CPGZDD TO UQSAT01
   GRANT SELECT,INSERT,UPDATE,DELETE ON SQLDBA.CPGFDD TO UQSAT01
   GRANT SELECT,INSERT,UPDATE,DELETE ON SQLDBA.CPGSDD TO UQSAT01
   GRANT SELECT,INSERT,UPDATE,DELETE ON SQLDBA.CPGTDD TO UQSAT01
   ```

SQL-Preprozessor

Der Aufruf von CPGPREP kann mehrere Parameterbefehle verarbeiten. So kann dem Programm CPGPREP mitgeteilt werden, wie die USERID des Starters ist und wie sein Passwort heißt.

Dies geschieht mit dem JCL Aufruf:

```
// EXEC CPGPREP,SIZE=256K,PARM='USERID=PREP/PREPPW,PRINT,BLOCK,  X
               ISOL(CS),PUNSOURCE,PREPNAME=PNAME'
```

Das Fortsetzungszeichen (X) bei JCL steht auf Spalte 72, das Folgestatement muss ab Stelle 16 beginnen.

Folgende Parameter können gesetzt werden:

PREPNAME=Pname...	Der SQL-Prepname, der dem Accessmodul gegeben wird. Falls nicht angegeben, so wird der Name des PHASE-Parameters des OPTIONS-Statements, oder der H-Karte Spalte 73 – 80, genommen.
NOPRINT/PRINT	NOPRrint ist der Defaultwert.
BLOCK/NOBLOCK	BLOCK ist der Defaultwert.
USERID=UID/PWD	Als Default wird SQLDBA/SQLDBAPW angenommen.
PUNSOURCE/ NONSOURCE	Die vom QSAT erstellten SQL-Befehle sollen mit dem Sourceprogramm auf SYSPCH ausgegeben werden. Default: NONSOURCE
ISOL(RR)/ISOL(CS)	Isolation Level (RR) ist default. Wird Cursor Stability gewünscht, so muss ISOL(CS) gesetzt gesetzt werden.
ISOLATION(RR)/(CS)	Langform des ISOL-Parameters.
SEPERATOR=; (.)	Statementende-Zeichen. Für englische Schreibweise kann das Semikolon gesetzt werden. Der Punkt ist der Defaultwert.
DB(SQLHUGO)	SQLHUGO ist dabei der Name einer alternativen Datenbank, in der das Package erstellt wird.
QSATPW(GEHEIM)	In Verbindung mit dem Parameter DB enthält QSATPW das Passwort – hier GEHEIM – zur Verarbeitung der alternativen Datenbank.

Andere Parameter werden zur Zeit nicht unterstützt.

Operative Befehle

Es besteht die Möglichkeit, per Programm z.B. Tabellen anzulegen, Tabellen zu löschen etc. Dabei spielt es keine Rolle, ob es sich um ein Batchprogramm oder eine Online-Anwendung handelt.

Die einzelnen Befehle sind hier nur kurz erläutert, für weitere Erklärungen siehe Handbuch 'SQL - Application Programming for VSE'.

ACQUIRE PUBLIC/PRIVATE DBSPACE ...
: Dieser Befehl legt programmbezogen einen DBSPACE an. Falls vor dem ACQUIRE ein CONNECT mit DBA-Autorität erfolgte, können auch Tabellen für andere User definiert werden.

ALTER DBSPACE ...
: Mit diesem Befehl können die Parameter eines DBSPACEs verändert werden, z. B. der Lockmode (Page, DBSpace, Row) oder der Freiplatz.

DROP DBSPACE
: Dieser Befehl löscht einen DBSPACE, wenn die entsprechende Berechtigung vorliegt. Vor dem Löschen müssen alle Tabellen in dem DBSPACE gelöscht werden.

LOCK DBSPACE/TABLE
: Mit diesem Befehl wird ein DBSPACE bzw. eine Tabelle für andere Zugriffe gesperrt.

COMMIT WORK
: Dieser Befehl beendet eine LUW (Logical Unit of Work). Dies ist z.B. erforderlich, wenn in einem logischen Ablauf ein Cursor wiederholt verwendet werden soll. COMMIT WORK sollte gegeben werden, wenn viele Änderungen in einer Tabelle durchgeführt wurden, da die geänderten Sätze für andere User gesperrt sind.

  COMMIT WORK gibt auch VSAM Strings frei.

Operative Befehle

ROLLBACK WORK (RELEASE)
: Mit diesem Befehl können Daten wieder in ihren alten Zustand zurückgesetzt werden, die während einer LUW geändert wurden.

UPDATE (ALL) STATISTICS FOR DBSPACE/TABLE
: Dieser Befehl aktualisiert die Indizes dieser Tabelle bzw. dieses DBSPACEs für den Zugriff.

CREATE TABLE/VIEW
: Mit diesem Befehl wird eine Tabelle bzw. eine View angelegt.

ALTER TABLE/VIEW
: Mit diesem Befehl können die Parameter einer Tabelle bzw. einer View geändert werden.

DROP TABLE/VIEW
: Dieser Befehl löscht eine Tabelle bzw. eine View.

CREATE INDEX
: Dieser Befehl legt einen Index an.

DROP INDEX
: Mit diesem Befehl wird ein Index gelöscht.

CREATE SYNONYM
: Mit diesem Befehl wird ein Synonym für eine bestimmte Tabelle angelegt.

COMMENT ON

LABEL ON

Operative Befehle

CONNECT :USER IDENTIFIED BY :PASSWORD

Dieser Befehl wird verwendet, um einen besonderen User anzumelden (z.B. wegen Berichtigungen). USER und PASSWORD sind mit jeweils 8 Stellen als Character-Felder definiert. Der CONNECT kann bei Online-Anwendungen entfallen, bei Batchprogrammen muss er durchgeführt werden.

GRANT TABLE/VIEW/PROGRAM PRIVILEGES

GRANT AUTHORITIES

REVOKE TABLE/VIEW/PROGRAM

REVOKE AUTHORITIES

Programmauszug

```
- -C.
-  USER = 'SQLDBA  '
-  PW   = 'SQLDBAPW'
-  SQL CONNECT :USER IDENTIFIED BY :PW
   ...
-  SQL SELECT DBSPACENAME,OWNER INTO :DBSPNM, :OWNER              *
-  SQL FROM   SYSTEM.SYSDBSPACES                                  *
-  SQL WHERE  DBSPACENAME = 'MEIER' AND OWNER = 'PUBLIC'
-  IF SQCODE = 100.                                    * NOT FOUND
-     SQL  ACQUIRE PUBLIC DBSPACE NAMED MEIER (PAGES = 128)
-  END
-  SQL SELECT TNAME,CREATOR INTO :TN,:CR FROM SYSTEM.SYSCATALOG   *
-  SQL WHERE TNAME = 'INTADR_PRIVAT' AND CREATOR = 'SQLDBA'
-  IF SQCODE = 100.                                    * NOT FOUND
-     SQL CREATE TABLE INTADR_PRIVAT                              *
-     SQL      ( KDNRA   CHAR(5)    NOT NULL,                     *
-     SQL        FIRMA   CHAR(30)   NOT NULL,                     *
-     SQL        NAME    CHAR(28)   NOT NULL,                     *
-     SQL        PLZ     CHAR(5)    NOT NULL,                     *
-     SQL        ORT     CHAR(30)   NOT NULL,                     *
-     SQL        TEL     CHAR(20)   NOT NULL,                     *
-     SQL        FAX     CHAR(15)   NOT NULL,                     *
-     SQL        PROGSP  CHAR(8),                                 *
-     SQL        SQLDS   CHAR(1),                                 *
-     SQL        DB2     CHAR(1)  )  IN MEIER
-     SQL CREATE UNIQUE INDEX INTADRP_IDX ON SQLDBA.INTADR_PRIVAT *
-     SQL        (KDNRA) PCTFREE=10
-     SQL GRANT  SELECT ON SQLDBA.INTADR_PRIVAT TO PUBLIC
-  ELSE.                                                * FOUND
-     SQL DELETE FROM SQLDBA.INTADR_PRIVAT
-  END.
-  SQL INSERT INTO SQLDBA.INTADR_PRIVAT (KDNRA,FIRMA,NAME,        *
-  SQL        PLZ,ORT,TEL,FAX,PROGSP,SQLDS,DB2)                   *
-  SQL SELECT KDNRA,FIRMA,NAME,PLZ,ORT,TEL,FAC,PROGSP,SQLDS,     *
-  SQL        DB2 FROM SQLDBA.INTADR                              *
-  SQL WHERE  SQLDS > ' ' OR DB2 > ' '
   ...
```

Abfrage der SQLCA

CPG in Verbindung mit dem Preprocessor stellt dem Anwendungsprogramm alle Variablen der SQLCA zur Verfügung. Damit keine zusätzliche TWA erforderlich ist, werden die SQLCA-Informationen nur nach explizitem Coding aktiviert. Außerdem ist es möglich, diese Abfragen in einem unabhängigen HL1-Modul auszuwerten.

Die SQLCA-Informationen werden in einem Feld mit Namen "SQLCAF" zur Verfügung gestellt. Dieses Feld wird in der Data Division mit einer Länge von 125 Bytes definiert. Das COPY-Buch CPGSQLCA überträgt die SQLCA Daten in das Feld SQLCAF. Über eine SELCT-Anweisung können die Inhalte in entsprechende Datenfelder übertragen werden. Im folgenden ist ein Beispiel dargestellt.

```
- -D.
-     SQLCAF 125.

- -I.
-     FIELD SQLCAF.
-          BIN      1    2 0 SQELEN
-                   3   72   SQERRM
-                  73   80   SQERRP
-          BIN     81   84 0 SQERD1
-          BIN     85   88 0 SQERD2
-          BIN     89   92 0 SQERD3
-          BIN     93   96 0 SQERD4
-          BIN     97  100 0 SQERD5
-          BIN    101  104 0 SQERD6
-                 105  115   SQWARN
-                 116  120   SQSTAT
-                 121  125   SQFILL.              * Inhalt: X'FF'

- -C.
-     ...
-     COPY  CPGSQLCA
-     SELCT SQLCAF
-     ...
```

Abfrage der SQLCA

Die Bedeutung der SQLCA Felder ist im Handbuch 'SQL Reference for IBM VM System and VSE' ausführlich beschrieben. Damit man eine ungefähre Vorstellung hat, welche Informationen in der SQLCA stehen, sind im folgenden einige dieser Felder erläutert:

SQELEN . Länge des Feldes: SQERRM

SQERRM . ein oder mehrere TOKENS, die durch X'FF' unterteilt sind.

SQERRP . Name des internen SQL-Moduls, in dem ein Fehler festgestellt wurde. Dieses Feld ist nur gefüllt, wenn der SQCODE >< 0 ist.

SQERD1 . enthält den Relational Data System (RDS-) Error Code.

SQERD2 . Database Storage System Return Code

SQERD3 . enthält die Anzahl der Zeilen, die durch INSERT, UPDATE, DELETE modifiziert wurden.

SQERD4 . ungefährer COST-Faktor

SQERD5 . Anzahl von abhängigen Zeilen, die durch DELETE in abhängigen Tabellen modifiziert oder gelöscht wurden.

SQERD6 . Reserviert

SQSTAT . enthält den SAA CPI Database Return Code. Dieser Return Code ist für all DB-Systeme (IBM) gleich. Dies gilt nicht für SQCODE !!!

Abfrage der SQLCA

SQWARN	.	0-10	11 Schalter, die folgende Bedeutung haben:
SQWRN0	.	' '	wenn alle anderen auch ' ', sonst 'W','N','Z' oder 'S'
SQWRN1	.	'W	wenn ein String (Spaltenwert) verkürzt in eine Hostvariable gesetzt wurde.
SQWRN2	.	'W'	wenn ein Null-Wert durch ein Argument einer Funktion eliminiert wurden.
SQWRN3	.	'W'	wenn die Anzahl der Spalten größer ist als die Anzahl der definierten Hostvariablen bei einem SELECT oder FETCH.
SQWRN4	.	'W'	wenn ein UPDATE oder ein DELETE ohne WHERE-Clause durchgeführt wurde.
SQWRN5	.	'W'	wenn ein SQL-Befehl gravierende Performance-Einbußen verursacht.
SQWRN6	.	'W'	wenn SQL/DS gezwungenermaßen eine LUW terminiert
		'S'	wenn ein schwerwiegender SQCODE vorliegt. (z.B. DB zerstört).
SQWRN7	.	'W'	wenn Datumswerte an den letzten Tag eines Monats angeglichen wurden.
SQWRN8	.	'W'	wenn ein SQL-Befehl nicht im Blocking Modus arbeiten kann.
SQWRN9	.	'W'	wenn Blocking wegen zu wenig virtuellem Speicher ausgeschaltet wurde.
SQWRNA	.	'W'	wenn Blocking für mindestens 2 Zeilen nicht durchgeführt wurde.

QSAT online

Im Folgenden wird die Anwendung des QSAT-Onlineprogramms beschrieben

Diese Maske erscheint nach Eingabe des Transaktionscodes 'QSAT':

```
==========================================================================
  QQQQQ                           V.L   OID  TERM  tt.mm.jj  11.53UHR
 QQ     QQ       Q uery                             Dienstag  CICSTEST
QQ       QQ        U ser
QQ       QQ          I nformation
QQ    QQ QQ           C ontrol
 QQ    QQQ              K it
   QQQQQ QQ                                    Quick  S-Q-L - Access
_________________________________________________________________________-

QSAT ist ein Programm der Lattwein GmbH Deutschland.
Dieses Programm darf nur von berechtigten Personen benutzt werden.




Userid   ===>   ________
Passwort ===>




_________________________________________________________________________
PF1 ==> Hilfe          Datenfreigabe für DB2-Sign on     PF12 ==> Ende
==========================================================================
```

Mit der PF1-Taste kann in der gesamten Anwendung eine Hilfe angezeigt werden, mit PF12 oder mit der Löschtaste wird die Verarbeitung beendet.

Für die Anmeldung muss die SQL-User-Id und das Passwort angegeben werden.

Nach Eingabe der User-Id und des Passworts erscheint nach Betätigen der Datenfreigabe diese Maske.

```
================================================================================
  QQQQQ                          V.L   OID  TERM  tt.mm.jj  11.53UHR
 QQ     QQ       Q uery                            Dienstag  CICSTEST
QQ       QQ       U ser
QQ       QQ         I nformation
QQ    QQ QQ         C ontrol
 QQ    QQQ            K it
  QQQQQ QQ                                    Quick  S-Q-L - Server
_______________________________________________________________________________

Funktion  auswählen .......         Tabelle ____________________ ._________

Anzeige   SYSCATALOG        VC      MS  Merge Catalog to CPGSDD
Anzeige   CPGSDD            VS      XS  Erweitern  CPGSDD
Erstellen CPGSDD            CS      PS  Pruefen    CPGSDD
Löschen   CPGSDD            ES      LS  Löschen PS Liste
Ändern    CPGSDD       AS / US      UA  Update  Access Namen
Ändern Langnamen       AL / UL      DL  Tabelle nach QDDS kopieren
Erstellen Access Namen      CA
Löschen   Access Namen      EA
Anzeige   Access Namen      VA      EX  Exit mit Logoff QSAT
Anzeige   Access Übersicht  SA
Kopieren  Access nach QDD   DD
_______________________________________________________________________________
PF1 ==> Hilfe               PF3 ==> Return                PF6 ==> QDD
================================================================================
```

Von diesem Menue aus können die einzelnen Funktionen zur Pflege der Access-Namen und der Spaltenbeschreibungen der Tabellen aufgerufen werden.

Die Funktionen werden im Einzelnen auf den folgenden Seiten beschrieben.

Durch Betätigung der PF6-Taste wird ins CPG2..Data Dictionary (QDD) verzweigt.

Nach Eingabe der Funktion 'VC' - Anzeige SYSCATALOG – erscheint dieses Bild:

```
================================================================================
Quick  S-Q-L - Server  List Tables V.L   OID  TERM  tt.mm.jj 11.53UHR
--------------------------------------------------------------------------------
No.   Tabellenname         Creator     No.    Tabellenname     Creator
--------------------------------------------------------------------------------
  1   ACTIVITY             SQLDBA        17   CPGSDDS          SQLDBA
  2   ARTIKELSTAMM         SQLDBA        18   CPGTDD           CPGDBA
  3   BEZUGSQUELLE_STAMM   SQLDBA        19   CPGTDD           SQLDBA
  4   CL_SCHED             SQLDBA        20   CPGTDDS          SQLDBA
  5   COST                 SQLDBA        21   CPGWRK           SQLDBA
  6   COST_TABLE           SQLDBA        22   CPGZDD           CPGDBA
  7   CPGADD               CPGDBA        23   CPGZDD           SQLDBA
  8   CPGADD               SQLDBA        24   CPGZDDS          SQLDBA
  9   CPGFDD               CPGDBA        25   DATTAPE          SQLDBA
 10   CPGFDD               SQLDBA        26   DATUSED          SQLDBA
 11   CPGFDDS              SQLDBA        27   DBFPD            SQLDBA
 12   CPGKDN               SQLDBA        28   DBNAME           DBADM
 13   CPGKSD               DBADM         29   DBPPD            SQLDBA
 14   CPGKSD               SQLDBA        30   DB1_B_SEGMENT    SQLDBA
 15   CPGSDD               CPGDBA        31   DEMADR_PRIVAT    SQLDBA
 16   CPGSDD               SQLDBA        32   DEPARTMENT       SQLDBA
--------------------------------------------------------------------------------
Ab Tabelle DEPARTMENT
--------------------------------------------------------------------------------
PF1 ==> Hilfe          PF3 ==> Funktionsauswahl         PF12 ==> Ende
================================================================================
```

Alle Tabellen, die im SYSTEM.SYSCATALOG vorhanden sind, werden hier mit dem zugehörigen Creator aufgeführt.

Mit der Datenfreigabetaste wird in der Übersicht vorwärts geblättert. Durch einen Eintrag bei 'Ab Tabelle' kann an jeder beliebigen Stelle der Übersicht aufgesetzt werden.

Erscheint unten rechts der Returncode '100', so bedeutet dies, dass das Ende der Übersicht (EOF) erreicht ist.

Mit PF3 gelangt man wieder zur Funktionsauswahl zurück.

Nach Eingabe der Funktion 'VS' erscheint folgende Maske:

Quick S-Q-L - Server List Tables V.L OID TERM tt.mm.jj 11.53UHR

Tabelle	Creator	Colname	Länge	Type	S-Name	Länge2
CPGADD	SQLDBA	ACCNAM	8	CHAR	ACCNAM	8
CPGADD	SQLDBA	CNAME	18	CHAR	CNAME	18
CPGADD	SQLDBA	CNAME6	6	CHAR	CNAME6	6
CPGADD	SQLDBA	COLNO		SMALLINT	COLNO	(2 , H)
CPGADD	SQLDBA	COLTYPE	8	CHAR	COLTYP	8
CPGADD	SQLDBA	CREATOR	8	CHAR	CREATO	8
CPGADD	SQLDBA	IDENT	1	CHAR	IDENT	1
CPGADD	SQLDBA	INDIC6	6	CHAR	INDIC6	6
CPGADD	SQLDBA	LENGCP	7	CHAR	LENGCP	7
CPGADD	SQLDBA	LENGTH	7	CHAR	LENGTH	7
CPGADD	SQLDBA	NULLS	1	CHAR	NULLS	1
CPGADD	SQLDBA	TIDENT	1	CHAR	TIDENT	1
CPGADD	SQLDBA	TNAME	18	CHAR	TNAME	18
CPGFDD	SQLDBA	ACCNAM	8	CHAR	ACCNAM	8
CPGFDD	SQLDBA	CNAME	18	CHAR	FCNAME	18
CPGFDD	SQLDBA	IDENT	1	CHAR	FIDENT	1

Ab Tabelle CPGFDD ab Colname IDENT

PF1 ==> Hilfe PF3 ==> Funktionsauswahl PF12 ==> Ende

Hier werden die im CPG-SQL-DD vorhandenen Einträge angezeigt.

Es werden alle Spalten der Tabelle aufgeführt, einschließlich der Feldlänge, des Feldtyps, des CPG-Namens und der Feldlänge im CPG.

Mit der Datenfreigabetaste kann vorwärts geblättert werden. Um bei einer bestimmten Tabelle / Spalte einer Tabelle aufzusetzen, können Einträge bei 'Ab Tabelle' und 'ab Colname' vorgenommen werden.

Wird unten rechts der Returncode '100' eingeblendet, so ist das Ende der Übersicht erreicht.

Mit PF3 gelangt man wieder zur Funktionsauswahl zurück.

Zur Pflege der Tabellenbeschreibungen im QSAT gehört, dass diese erstellt, geändert und gelöscht werden können.

Um eine Tabellenbeschreibung anzulegen, gibt man in der Funktionsauswahl den Befehl 'CS' - Erstellen CPGSDD - an sowie den Tabellennamen und den Namen des Creators.

Zum Löschen gibt man den Befehl 'ES' - Löschen CPGSDD - an, einschließlich des Tabellen- und Creatornamens.

Bei Betätigung der Datenfreigabetaste wird der jeweilige Befehl ausgeführt. Anschließend erscheint folgende Maske:

```
================================================================================
   QQQQQ                              V.L   OID  TERM  tt.mm.jj  11.53UHR
 QQ     QQ       Q uery                                DIENSTAG  CICSTEST
QQ       QQ        U ser
QQ       QQ          I nformation
QQ    QQ QQ            C ontrol
 QQ    QQQ               K it
   QQQQQ QQ                                      Quick  S-Q-L – Server
--------------------------------------------------------------------------------

Funktion  auswählen .......   CS      Tabelle TEST       . SQLDBA
                              __              ____________  _______

Fehler : OK
SQCODE :
       :
       :


                                  ANGELEGT.


--------------------------------------------------------------------------------
PF1 ==> Hilfe                PF3 ==> Return            PF12 ==> Ende
================================================================================
```

Der Benutzer erhält an dieser Stelle einen Hinweis, ob die gewünschte Funktion korrekt ausgeführt wurde.

Tritt ein Fehler auf, so erscheint statt der Meldung 'OK' eine entsprechende Fehlermeldung. Dabei werden bis zu drei SQL-Returncodes angezeigt.

Auch von dieser Stelle aus kann der Benutzer jede gewünschte Funktion ausführen. Dies bedeutet z. B., dass eine Tabelle mit dem Befehl 'ES' gelöscht und sofort mit 'CS' wieder angelegt werden kann. Damit werden die Zuordnungen aus der SYSTEM.SYSCOLUMNS wieder übernommen.

Mit der PF3-Taste gelangt man wieder zurück zur Funktionsauswahl.

Nach Auswahl der Funktion 'US' - Ändern CPGSDD - erscheint eine Auflistung der Columns einer Tabelle. Dies könnte folgendermaßen aussehen:

```
Quick  S-Q-L - Server  UPD  Tables  V.L  OID  TERM  tt.mm.jj 11.53UHR
_____________________________________________________________________
CPG-Feldname Feldlg Indicv. Upd SQL-Col.Name Länge  Type    Nulls  CT
_____________________________________________________________________
TDATUM       10   D  A0      ?   TDATUM              DATE    Y       ?
TESTNR        2   H  ------  ?   TESTNR              SMALLINTN
TESTRO        5      H0      ?   TESTROWA          5 CHAR    Y
TTDATE       10   D  C0      ?   TTDATE              DATE    Y       ?
TTDATE       10   D  H1      ?   TTDATEN             DATE    Y       ?
TTIME        10   T  KG      ?   TTIME              TIME    Y       ?
TTSTMP       26   S  H2      ?   TTSTMP             TIMESTMP Y       ?
TTSTMP       26   S  C1      ?   TTSTMPN            TIMESTMP Y       ?
TTTIM        10   T  C2      ?   TTTIM              TIME    Y       ?
TTTIMN       10   T  H3      ?   TTTIMN             TIME    Y       ?


_____________________________________________________________________
Tabelle TEST                  SQLDBA
_____________________________________________________________________
PF1 ==> Hilfe          PF3 ==> Funktionsauswahl        PF12 ==> Ende
=====================================================================
```

In der ersten Spalte werden die Namen der Hostvariablen definiert.

In der zweiten Spalte wird die Feldlänge angegeben (1 - 255).

In der dritten Spalte werden die Dezimalstellen für numerische Felder bzw. 'H' für Smallinteger- oder 'F' für Integer-Felder definiert.

Sind für diese Row Nullwerte erlaubt, so sollte auch eine Indicatorvariable in der 4. Spalte eingetragen werden. Der Eintrag 'N' in Spalte UPD * bewirkt, dass diese Spalten beim Update nicht verändert werden.

Unter CT kann für Spalten vom Typ DATE/TIME oder TIMESTMP ein 'C' eingetragen werden. Hieraus wird dann bei Update oder Add 'Current Date' bzw. 'Current Time' oder 'Current Timestamp'.

Nach Auswahl einer der Funktionen 'CA', 'EA' oder 'VA' zum Erstellen, Löschen oder Anzeigen von Access-Namen erscheint folgendes Bild:

```
================================================================================
    QQQQQ                          V.L   OID  TERM  tt.mm.jj  11.53UHR
 QQ     QQ       Q uery                              DIENSTAG  CICSTEST
QQ       QQ       U ser
QQ       QQ         I nformation
QQ    QQ QQ           C ontrol
 QQ    QQQ              K it
   QQQQQ QQ                                    Quick  S-Q-L - Server
--------------------------------------------------------------------------------
                                        Anlegen
Funktion  auswählen ....... CA
Accessname ................ _________
Beschreibung .............. ____________________________________________
Auswahl nach Colno......... _
Welche Definitionen sollen angezeigt werden ?

Tabelle.Creator............ ____________________  . ____________    A
                            ____________________    ____________    B
                            ____________________    ____________    C
                            ____________________    ____________    D
                            ____________________    ____________    E
                            ____________________    ____________    F

--------------------------------------------------------------------------------
PF1 ==> Hilfe               PF3 ==> Return              PF12 ==> Ende
================================================================================
```

Um einen neuen Access anzulegen, gibt der Benutzer hier einen Namen für den Access sowie eine Beschreibung und mindestens einen Tabellennamen mit Creator an. Fehlt einer dieser Einträge, so erscheint eine Fehlermeldung. Es können bis zu sechs Tabellen angegeben werden, aus denen dann eine Auswahl der Spalten getroffen werden kann. Die Tabellen müssen in der gewünschten Reihenfolge definiert werden. Dabei wird ihnen auch der Correlation-Name (A - F) zugeordnet.

Um einen Accessnamen zu löschen, muss hier lediglich der gewünschte Name eingetragen werden. Bei Betätigung der Datenfreigabe ist der Accessname gelöscht.

Um einen Access anzuzeigen, muss nur der Name angegeben werden.

Um eine Anzeige von allen Accessen zu erhalten, kann bei 'Accname' ein '*' eingetragen werden.

Beim Anlegen eines Access-Namens erscheint nach Betätigung der Datenfreigabe folgende Maske mit entsprechenden Feldern:

```
================================================================================
Quick  S-Q-L - Server  ADD  TEST01
--------------------------------------------------------------------------------
Mod. Column-Name      Tabelle        Creator    I   CPG-Name  Länge
--------------------------------------------------------------------------------
 x   TDATUM           TEST           SQLDBA     A   TDATUM    (10, D)
 _   TESTNR           TEST           SQLDBA     A   TESTNR    ( 2, H)
 _   TESTROWA         TEST           SQLDBA     A   TESTRO         5
 _   TTDATE           TEST           SQLDBA     A   TTDATE    (10, D)
 _   TTDATEN          TEST           SQLDBA     A   TTDATE    (10, D)
 x   TTIME            TEST           SQLDBA     A   TTIME     (10, T)
 _   TTSTMP           TEST           SQLDBA     A   TTSTMP    (26, S)
 _   TTSTMPN          TEST           SQLDBA     A   TTSTMP    (26, S)
 _   TTTIM            TEST           SQLDBA     A   TTTIM     (10, T)
 _   TTTIMN           TEST           SQLDBA     A   TTTIMN    (10, T)
 _   ACCNAM           TSTFDD         SQLDBA     B   ACCNAM        8
 x   CNAME            TSTFDD         SQLDBA     B   CNAME         18
 _   IDENT            TSTFDD         SQLDBA     B   IDENT          1
 x   CNAME            TSTSDD         SQLDBA     C   CNAME         18
 _   CNAME6           TSTSDD         SQLDBA     C   CNAME6        18
 _   COLNO            TSTSDD         SQLDBA     C   COLNO     ( 2, H)
--------------------------------------------------------------------------------
Tabelle   TSTSDD              SQLDBA                 Action
--------------------------------------------------------------------------------
PF1 ==> Hilfe       PF5  ==> Proceed (add Selection)    PF12 ==> Ende
================================================================================
```

Hier werden die einzelnen Spalten der ausgewählten Tabellen angelistet. Zu jeder Spalte sind der CPG-Name und die Länge aufgeführt.

Angezeigt wird auch der Correlation Name (A, B, ... F), der den Tabellen beim Anlegen des Accesses zugeordnet wurde.

Reicht die Anzeige nicht aus, um alle Spalten der ausgewählten Tabellen anzuzeigen, so ist in der unteren Zeile zu sehen, mit welcher Tabelle/Spalte die Liste fortgesetzt wird.

Um einen Spaltennamen unter diesem Access zu definieren, muss in der ersten Spalte ein Zeichen größer Blank eingetragen werden.

Zum Speichern muss die Taste PF5 gedrückt werden.

Mit PF3 gelangt man zur Funktionsauswahl zurück.

Wurde die Funktion 'VA' zur Anzeige eines Accesses ausgewählt und der gewünschte Accessname eingetragen, so erscheint folgendes Bild:

```
================================================================================
Quick  S-Q-L - Server  View  TEST01 V.L  OID  TERM  tt.mm.jj 11.53UHR
--------------------------------------------------------------------------------
ID Column-Name    Tabelle       Creator   CPG-Name Laenge Null Accnam
--------------------------------------------------------------------------------
 A TDATUM         TEST          SQLDBA    TDATUM   (10, D)  Y  TEST01
 A TTIME          TEST          SQLDBA    TTIME    (10, T)  Y  TEST01
 B CNAME          TSTFDD        SQLDBA    CNAME         18  N  TEST01
 C CNAME          TSTSDD        SQLDBA    CNAME         18  N  TEST01
--------------------------------------------------------------------------------
VA                                         Action
--------------------------------------------------------------------------------
PF1 ==> Hilfe           PF3 ==> Funktionsauswahl        PF12 ==> Ende
================================================================================
```

Hier werden die Spalten eines Accesses aufgeführt.

Zu jeder Spalte wird angezeigt, zu welchem Access sie gehört, aus welcher Tabelle sie stammt, unter welchem Correlation-Namen (ID) sie geführt wird (A, B, ... F), welcher CPG-Name ihr zugeordnet wurde, wie lang das Feld ist und ob es sich um ein Nullfeld handelt.

Wurde die Funktion 'VA' zur Anzeige eines Accesses ausgewählt und statt eines Accessnamens ein '*' eingetragen, so erscheint folgendes Bild:

```
================================================================================
Quick  S-Q-L - Server  View  *    V.L   OID  TERM  tt.mm.jj  11.53UHR
--------------------------------------------------------------------------------
ID Column-Name      Tabelle     Creator   CPG-Name Laenge Null Accnam
--------------------------------------------------------------------------------
 A TESTNR           TEST        SQLDBA    TESTNR   ( 2, H)  N  TEREV
 A TTDATE           TEST        SQLDBA    TTDATE   (10, D)  Y  TEREV
 A TTDATEN          TEST        SQLDBA    TTDATE   (10, D)  Y  TEREV
 A TTIME            TEST        SQLDBA    TTIME    (10, T)  Y  TEREV
 A TTSTMP           TEST        SQLDBA    TTSTMP   (26, S)  Y  TEREV
 B KLBAO            KLB         SQLDBA    KLBAO          1  Y  TEREV
 B KLBB12           KLB         SQLDBA    KLBB12         3  Y  TEREV
 A TDATUM           TEST        SQLDBA    TDATUM   (10, D)  Y  TEST01
 A TTIME            TEST        SQLDBA    TTIME    (10, T)  Y  TEST01
 B CNAME            TSTFDD      SQLDBA    CNAME         18  N  TEST01
 C CNAME            TSTSDD      SQLDBA    CNAME         18  N  TEST01
 A KLBC1            KLB         SQLDBA    KLBC1          1  Y  TST20
 A KLBC2            KLB         SQLDBA    KLBC2          2  Y  TST20
 A KLBKEY           KLB         SQLDBA    KLBKEY         8  N  TST20
 B TESTNR           TEST        SQLDBA    TESTNR   ( 2, H)  N  TST20
 B TDATE            TEST        SQLDBA    TDATE    (10, D)  Y  TST20
--------------------------------------------------------------------------------
VA                                        Action
--------------------------------------------------------------------------------
PF1 ==> Hilfe          PF3 ==> Funktionsauswahl        PF12 ==> Ende
================================================================================
```

Hier werden die Spalten aller Accesse aufgeführt.

Zu jeder Spalte wird angezeigt, zu welchem Access sie gehört, aus welcher Tabelle sie stammt, unter welchem Correlation-Namen (ID) sie geführt wird (A, B, ... F), welcher CPG-Name ihr zugeordnet wurde, wie lang das Feld ist und ob es sich um ein Nullfeld handelt.

Nach Auswahl der Funktion 'SA' - Anzeige Access-Übersicht - erscheint eine Auflistung der Accessnamen mit Beschreibung:

```
=====================================================================
Quick  S-Q-L - Server  Overview   V.L   OID  TERM  tt.mm.jj  11.53UHR
_____________________________________________________________________
Accnam    Beschreibung    Owner    Accnam    Beschreibung     Owner
---------------------------------------------------------------------
AA        A               MAASSEN  ACCNEU3   TEST           MAASSEN
AAA       AAA             MAASSEN  ACCNEU4   TEST           MAASSEN
AABZ      TEST            MAASSEN  ACC3271   TEST           SQLDBA
AABZ1     TETST           MAASSEN  ADBFPD    A              MAASSEN
AABZ2     TETDT           MAASSEN  AHUGO     TEST SEMINAR   MAASSEN
AAMAS1    TEST            MAASSEN  AHUGO1    TEST           MAASSEN
AAMAS2    TEST            MAASSEN  AKOVERTB  AKO VERTAB DD  MAASSEN
AAMAS3    TEST            MAASSEN  ALONGN    TEST BAU-BG    MAASSEN
AATBZ3    TETETS          MAASSEN  ALONG01   TEST LONGNAME  MAASSEN
AA11      OHNE KOMMENTAR  MAASSEN  AMORR01   ANY TEXT       MAASSEN
AA12      TEST            MAASSEN  ARTIKPR   TEST PR        SQLDBA
AA121     TEST            MAASSEN  ASA       TEST           MAASSEN
AA122     TEST            MAASSEN  ASG001    TEST ACCESS    MAASSEN
ACCNEU    TEST            MAASSEN  ASG01     TEST EDEKA     MAASSEN
ACCNEU1   TEST            MAASSEN  ASG02     STAHLGRUBER    MAASSEN
ACCNEU2   TEST            MAASSEN  ASG03     TEST BAU-BG    MAASSEN
_____________________________________________________________________
                                                   Action
_____________________________________________________________________
PF1 ==> Hilfe          PF3 ==> Funktionsauswahl        PF12 ==> Ende
=====================================================================
```

Nach Auswahl der Funktion 'DD' - Kopieren Access nach QDD - erscheint folgende Maske:

```
================================================================================
   QQQQQ                          V.L   OID  TERM  tt.mm.jj  11.53UHR
 QQ     QQ       Q uery                             DIENSTAG  CICSTEST
QQ       QQ        U ser
QQ       QQ          I nformation
QQ    QQ QQ            C ontrol
 QQ    QQQ               K it
   QQQQQ QQ                                    Quick  S-Q-L - Server
--------------------------------------------------------------------------------

Mit diesem Programm kann ein mit QSAT definierter Access ins Data
Dictionary übertragen werden.


Accessname ... _________

Datei ........ _________
Satzart ...... __
DD-Version.... 2.4                                                          *

Replace ...... _

--------------------------------------------------------------------------------
PF1 ==> Hilfe            PF3 ==> Funktionsauswahl         PF6 ==> QDD
================================================================================
```

Nach Betätigung der Datenfreigabe werden die Felder des angegebenen Accesses in die gewünschte Datei im Data Dictionary übertragen.

Die Datei muss bereits im Data Dictionary angelegt sein.

Der Eintrag 'Y' oder 'J' bei Replace bewirkt, dass eventuell in der Struktur vorhandene Felder gelöscht werden. Wird bei Replace 'N' eingetragen, so werden die Felder des Accesses an die schon vorhandenen Felder angefügt.

Programmieren mit QSAT

QSAT-Befehle verwenden die Accessnamen, die online definiert wurden.

Aus der Verwendung von QSAT-Befehlen ergeben sich einige Vorteile:

- Es werden nur die Columns übertragen, die im Zugriff tatsächlich benutzt werden.
- Der Preprocessor ordnet die Column-Namen nach der Definition der Tabelle (also nach COLNO).
- Der Codieraufwand reduziert sich auf ein Minimum, was vor allem bei globalen Zugriffen in der Auflösung zu sehen ist.
- Die QSAT-Befehle lehnen sich an CPG-/HL1-IO-Befehle an, wodurch das Programm für den CPG-Programmierer besser lesbar wird.

Auf den folgenden Seiten werden die QSAT-Befehle mit ihrer Auflösung nach dem Durchlaufen des Preprocessors aufgeführt.

CHAIN

Der Befehl

```
-    SQL CHAIN ACCNAM WHERE ...
```

wird aufgelöst in folgende SQL-Commands:

```
- SQL   DECLARE PROGNCPGCUnn CURSOR FOR SELECT .. FROM .. WHERE..
-       SQCODD = SQCODE
- SQL   OPEN PROGNCPGCUnn
-       SQCODO = SQCODE
- SQL   FETCH PROGNCPGCUnn INTO ..
-       SQCODF = SQCODE
- SQL   CLOSE PROGNCPGCUnn
-       SQCODC = CQCODE
```

Für die Zugriffe wird ein interner CURSOR definiert, dessen Name sich folgendermaßen zusammensetzt: Aus dem Programmnamen (PROGN), der Konstanten CPG und der Konstanten CU mit einer laufenden Nummer (nn).

Nach jedem SQL-Befehl kann das Feld SQCODE auf einen Fehler abgefragt werden. Da jedoch in diesem SQL-Befehle mit QSAT mehrere Befehle verknüpft sind, können nach dem Befehl auch verschiedene Felder auf Fehler abgefragt werden. Damit wird verhindert, dass z. B. ein Fehler beim FETCH durch ein fehlerfreies CLOSE unbemerkt bleibt.

Nach dem OPEN wird der Wert aus SQCODE dem Feld SQCODO zugewiesen, nach dem FETCH dem Feld SQCODF und nach dem CLOSE dem Feld SQCODC. Nach dem DECLARE wird zwar der Wert aus SQCODE dem Feld SQCODD zugewiesen, allerdings ist der Wert aus internen Gründen immer 0, d. h. er kann zur Fehlererkennung z. Z. nicht genutzt werden.

War der FETCH-Befehl erfolgreich, so wurden auch das DECLARE und das OPEN ohne Fehler ausgeführt.

Steht nach dem CHAIN im Feld SQCODF der Wert '100', so wurde der Satz nicht gefunden.

CHAIN-SINGLE

Der Befehl

- SQL CHAIN-SINGLE ACCNAM WHERE ...

wird umgesetzt in folgenden SQL Command:

- SQL SELECT ... INTO ... FROM ... WHERE ...

Die Operation CHAIN-SINGLE kann immer dann angewendet werden, wenn sicher ist, dass nur eine Zeile als Ergebnis der WHERE-Clause die Bedingungen erfüllt.

Alle Informationen außer der WHERE-Clause werden dem angegebenen Access entnommen.

Nach der Operation kann das Feld SQCODE auf einen Fehler abgefragt werden.

Steht nach dem Befehl im Feld SQCODE der Wert '100', so wurde der Satz nicht gefunden.

Sollte mehr als eine Ergebnis-Zeile erzielt werden, so hat SQCODE den Wert -810. An das Programm wird nur eine Zeile übertragen.

SETLL

Der Befehl

- SQL SETLL CURNAM ACCNAM WHERE ...

wird aufgelöst in folgende SQL Commands:

```
- SQL   DECLARE CURNAM CURSOR FOR SELECT ... FROM ... WHERE ...
-       SQCODD = SQCODE.
- SQL   OPEN CURNAM
-       SQCODO = SQCODE.
```

Alle Informationen außer der WHERE-Clause werden dem angegebenen Access entnommen.

Der CURSOR muss vom Programmierer definiert werden und auch beim READ-, RNDOM- und CLOSE-Befehl verwendet werden. Hierbei ist darauf zu achten, dass innerhalb einer LUW immer nur ein CURSOR mit einem Namen eröffnet werden kann. Daher ist darauf zu achten, dass nicht der gleiche Name mehrmals vergeben wird.

Besonders bei der Verwendung von HL1-Modulen, bei denen der Programmierer nicht weiß, welche Cursor-Namen verwendet wurden, sollte der Cursor-Name immer aus dem Programm/Modulnamen und einem veränderlichen Anhang bestehen.

Nach der Operation SETLL kann das Feld SQCODE geprüft werden, um festzustellen, ob die Operationen erfolgreich ausgeführt wurden. SQCODO stellt einen eventuellen Fehler beim OPEN fest. Ein Fehler beim DECLARE ist nur indirekt festzustellen, da das Feld SQCODD aus internen Gründen z. Z. immer den Wert 0 enthält.

Mit diesem Befehl wird der CURSOR positioniert, um ab dieser Stelle sequentiell lesen zu können.

READ

Der Befehl

```
-    SQL READ CURNAM ACCNAM
```

wird umgesetzt in folgenden SQL Command:

```
- SQL   FETCH CURNAM INTO ...
-       SQCODF = SQCODE.
```

Alle Informationen werden dem angegebenen Access entnommen.

Der Cursor muss vom Programmierer definiert werden. Der Cursor-Name muss identisch sein mit dem für SETLL definierten Namen.

Nach der Operation READ kann das Feld SQCODF geprüft werden, um festzustellen, ob der FETCH korrekt ausgeführt wurde.

Diese Operation übergibt sequentiell Ergebniszeilen an das Programm.

RNDOM

Der Befehl

```
-    SQL RNDOM CURNAM
```

wird aufgelöst in folgende SQL-Commands:

```
- SQL   CLOSE CURNAM
-       SQCODC = SQCODE.
```

Alle Informationen werden dem angegebenen Access entnommen.

Der Cursor muss vom Programmierer definiert werden. Der Cursor-Name muss identisch sein mit dem für SETLL definierten Namen.

Nach der Operation RNDOM kann das Feld SQCODC geprüft werden, um festzustellen, ob das CLOSE auf den CURSOR korrekt ausgeführt wurde.

Diese Operation beendet die sequentielle Verarbeitung für diesen CURSOR.

UPDAT

Der Befehl

- SQL UPDAT ACCNAM WHERE ...

wird umgesetzt in folgenden SQL Command:

- SQL UPDATE TABLE SET ... WHERE ...

Alle Informationen werden dem angegebenen Access entnommen.

Nach der Operation UPDAT kann das Feld SQCODE geprüft werden, um festzustellen, ob das UPDATE korrekt ausgeführt wurde.

Dieser Befehl ähnelt sehr dem eigentlichen SQL-Befehl. Daher ist darauf zu achten, dass der Befehl UPDAT lautet. Wird stattdessen UPDATE verwendet, so erwartet der Preprocessor anschließend einen Tabellen-Namen. Da dort aber kein Tabellen-Name steht, kommt es zu einer Fehlermeldung bei der Umwandlung.

Wichtig ist, dass immer eine WHERE-Bedingung definiert wird, da sonst die gesamte Tabelle verändert wird.

WRITE

Der Befehl

- SQL WRITE ACCNAM

wird umgesetzt in folgenden SQL-Command:

- SQL INSERT INTO TABLE ... VALUES ...

Alle Informationen werden dem angegebenen Access entnommen.

Nach der Operation WRITE kann das Feld SQCODE geprüft werden, um fest zustellen, ob das INSERT korrekt ausgeführt wurde.

Hier ein Beispiel im CPG-Format:

```
 - OPTIONS MAIN COM PHASE TST033.
 - *---------------------------------------------------------------*
 - *     SQL-BEFEHLE UND FORMATE                                    *
 - *---------------------------------------------------------------*
 - -D.
 - SQL    BEGIN DECLARE SECTION.
 -        USER      8.
 -        PASSW     8.
 -        KEY       5 0.
 - SQLDD KKDA.
 - SQL    END DECLARE SECTION
 - -C.
 - *---------------------------------------------------------------*
 -     USER  = 'SQLDBA  '
 -     PASSW = 'SQLDBAPW'
 - SQL CONNECT :USER IDENTIFIED BY :PASSW
 -     CAB SQCODE >< 0 ERROR.               * ON ERROR STOP
 - *---------------------------------------------------------------*
 - SQL CHAIN KKDA WHERE KKDNR = :KEY
 - SQL CHAIN KKDA                                                  *
 - SQL       WHERE KKDNR = :KEY AND KFIRMA LIKE  'LATTW%'
 -     CAB SQCODE >< 0 ERROR.               * ON ERROR STOP
 - *---------------------------------------------------------------*
 - SQL SETLL TST033CURS01 KKDA WHERE A.KKDNR = :KEY
 -     CAB SQCODE >< 0 ERROR.               * ON ERROR STOP
 - *---------------------------------------------------------------*
 - SQL READ TST033CURS01 KKDA
 -     CAB SQCODE >< 0 ERROR.               * ON ERROR STOP
 - *---------------------------------------------------------------*
 - SQL RNDOM TST033CURS01
 -     CAB SQCODE >< 0 ERROR.               * ON ERROR STOP
 - *---------------------------------------------------------------*
 - SQL UPDAT KKDA WHERE KKDNR = :KEY
 -     CAB SQCODE >< 0 ERROR.               * ON ERROR STOP
 - *---------------------------------------------------------------*
 - SQL WRITE KKDA
 -     CAB SQCODE >< 0 ERROR.               * ON ERROR STOP
 - *---------------------------------------------------------------*
 -     ...
 -     ERROR.
 -        EXHM ERRCHK.                      * CHECK FÜR SQLERROR
 -        EXSR COMIT.
 -     COMIT BEGSR.
 -        SQL COMMIT WORK.
 -     ENDSR.
 - -O.
 -   FIELD FEHLER.
 -                24 '     SQL FEHLER:       '.
 -      KEY        5.
 -      SQCODE    40 EDITCODE Z.
```

Hier wird nun die Verwendung eines Accesses im Programm gezeigt.
Hier das Beispiel im RPG-Formalismus:

```
H    A                   8               L   C             TST033
H*-------------------------------------------------------------------*
H*     SQL-BEFEHLE UND FORMATE                                     *
H*-------------------------------------------------------------------*
DSQ      BEGIN DECLARE SECTION
D        USER           8
D        PASSW          8
D        KEY            5 0
DKKDA          SQ
DSQ      END DECLARE SECTION
D        FEHLER        40
C*-------------------------------------------------------------------*
C                     MOVEL'SQLDBA  'USER
C                     MOVEL'SQLDBAPW'PASSW
CSQ      CONNECT :USER IDENTIFIED BY :PASSW
C           SQCODE    CABNE0         ERROR
C*-------------------------------------------------------------------*
CSQ      CHAIN KKDA WHERE   KKDNR = :KEY
C           SQCODE    CABNE0         ERROR
C*-------------------------------------------------------------------*
CSQ      SETLL TST033CURS01 KKDA WHERE A.KKDNR = :KEY
C           SQCODE    CABNE0         ERROR
C*-------------------------------------------------------------------*
CSQ      READ  TST033CURS01 KKDA
C           SQCODE    CABNE0         ERROR
C*-------------------------------------------------------------------*
CSQ      RNDOM TST033CURS01
C           SQCODE    CABNE0         ERROR
C*-------------------------------------------------------------------*
CSQ      UPDAT KKDA WHERE KKDNR = :KEY
C           SQCODE    CABNE0         ERROR
C*-------------------------------------------------------------------*
CSQ      WRITE KKDA
CSQ         SQCODE    CABNE0         ERROR
C*-------------------------------------------------------------------*
C           ERROR     TAG
C                     EXHM ERRCHK              CHECK FÜR SQLERROR
C                     EXSR COMIT
C           COMIT     BEGSR
CSQ   COMMIT WORK
C                     ENDSR
OFEHLER  F
O                                      24 '     KEY NICHT GEFUNDEN'
O                         KEY          5
O                         SQCODE      40 '      '
```

Wenn der CPG-SQL Preprocessor durchlaufen wird, werden die codierten SQL-Befehle aufgelöst. Es wird folgende Liste erstellt:

```
C P G  -  S Q L  PREPROCCESSOR 24.01.06 14.07 UHR CPGPREP 2.4 SEITE 1

        FOLGENDE HOSTVARIABLEN WURDEN GEFUNDEN
        UND VOM SQL/DS CPG PREPROCESSOR DEFINIERT
             NAME              SQL TYPE     LAENGE
        ------------------  ------------   -------
        USER                  CHARACTER          8
        PASSW                 CHARACTER          8
        KEY                   DECIMAL          5,0
        KKDNR                 DECIMAL          5,0
        KFIRMA                CHARACTER         30
        KFBEZ                 CHARACTER         24
        H1                    SMALLINTEG         2
        KPLZ                  CHARACTER          5
        H4                    SMALLINTEG         2
        KLAND                 CHARACTER          5
        C2                    SMALLINTEG         2
        KORT                  CHARACTER         20
        H3                    SMALLINTEG         2
        KSTR1                 CHARACTER         25
        H6                    SMALLINTEG         2
        KSTR2                 CHARACTER         11
        H7                    SMALLINTEG         2
        KGPART                CHARACTER         25
        KG                    SMALLINTEG         2
        KTELNR                CHARACTER         20
        H8                    SMALLINTEG         2
        KKNAM                 CHARACTER          9
        C1                    SMALLINTEG         2
        BKDNR                 DECIMAL          5,0
        A0                    SMALLINTEG         2

         ENDE DER DEFINITIONEN

        PROGRAMM NAME   =  TST033
        COMPILER OPTIONS = NOLIST
        COMMIT WORK DONE
```

```
PROGRAMM OHNE TITEL    99999 LATTWEIN    19.01.06    14.08 UHR     HL1 2.6     SEITE 1

 1 - OPTIONS MAIN COM HL1 H PHASE TST033.                TST033 BLOCKDIAGRAMM     TST033
   - *-------------------------------------------------* TST033
   - *     SQL-BEFEHLE UND FORMATE                     * TST033
   - *-------------------------------------------------* TST033
   - -D.                                                 TST033
   - *-------------------------------------------------*   TST033
   - *   CPG-SQL/DS STATEMENT                        *   TST033
   - SQL    BEGIN DECLARE SECTION.                       TST033
 2 -        USER           8.                            TST033        USER       1   8
 3 -        PASSW          8.                            TST033        PASSW      9  16
 4 -        KEY            5 0.                          TST033        KEY       17  19
   - * SQLDD KKDA                                        TST033
 5 -        KKDNR          5 0.    * KKDNR               TST033        KKDNR     20  22
 6 -        KFIRMA        30.      * KFIRMA              TST033        KFIRMA    23  52
 7 -        KFBEZ         24.      * KFBEZ               TST033        KFBEZ     53  76
 8 -        H1             2 HALF. * KFBEZ               TST033        H1        77  78
 9 -        KPLZ           5.      * KPLZ                TST033        KPLZ      79  83
10 -        H4             2 HALF. * KPLZ                TST033        H4        84  85
11 -        KLAND          5.      * KLAND               TST033        KLAND     86  90
12 -        C2             2 HALF. * KLAND               TST033        C2        91  92
13 -        KORT          20.      * KORT                TST033        KORT      93 112
14 -        H3             2 HALF. * KORT                TST033        H3       113 114
15 -        KSTR1         25.      * KSTR1               TST033        KSTR1    115 139
16 -        H6             2 HALF. * KSTR1               TST033        H6       140 141
17 -        KSTR2         11.      * KSTR2               TST033        KSTR2    142 152
18 -        H7             2 HALF. * KSTR2               TST033        H7       153 154
19 -        KGPART        25.      * KGPART              TST033        KGPART   155 179
20 -        KG             2 HALF. * KGPART              TST033        KG       180 181
21 -        KTELNR        20.      * KTELNR              TST033        KTELNR   182 201
22 -        H8             2 HALF. * KTELNR              TST033        H8       202 203
23 -        KKNAM          9.      * KKNAM               TST033        KKNAM    204 212
24 -        C1             2 HALF. * KKNAM               TST033        C1       213 214
25 -        BKDNR          5 0.    * BKDNR               TST033        BKDNR    215 217
26 -        A0             2 HALF. * BKDNR               TST033        A0       218 219
27 -        SQPARM     0 * 100                           TST033 AQPARM -------- 220 319
28 -        SQPROG         8                             TST033        SQPROG   220 227
29 -        SQUSER         8                             TST033        SQUSER   228 235
30 -        SQCURS        18                             TST033        SQCURS   236 253
31 -        SQCALL         8                             TST033        SQCALL   254 261
32 -        SQCODE         7 0                           TST033        SQCODE   262 265
33 -        SQSTMT         7 0                           TST033        SQSTMT   266 269
34 -        SQLISL         1                             TST033        SQLISL   270 270
35 -        SQFREE         1                             TST033        SQFREE   271 271
36 -        SQCNTL         4                             TST033        SQCNTL   272 275
37 -        SQI001         4                             TST033        SQI001   276 279
38 -        SQCNTS         4                             TST033        SQCNTS   280 283
39 -        SQI00S         4                             TST033        SQI00S   284 287
40 -        SQSQCA         4                             TST033        SQSQCA   288 291
41 -        SQI002        12                             TST033        SQI002   292 303
42 -        SQCODD         7 0                           TST033        SQCODD   304 307
43 -        SQCODO         7 0                           TST033        SQCODO   308 311
44 -        SQCODF         7 0                           TST033        SQCODF   312 315
45 -        SQCODC         7 0                           TST033        SQCODC   316 319
   - SQL    END DECLARE SECTION                          TST033
46 -        FEHLER        40.                            TST033        FEHLER   320 359
```

```
PROGRAMM OHNE TITEL    99999 LATTWEIN    19.01.06    14.08 UHR     HL1 2.6     SEITE 2
```

```
    - -C.                                                              TST033
    - *-----------------------------------------------------------*    TST033
 47 -     USER  = 'SQLDBA  '                                          TST033      I
 48 -     PASSW = 'SQLDBAPW'                                          TST033      I
    - SQL CONNECT :USER IDENTIFIED BY :PASSW                          TST033
 59 -     CAB SQCODE >< 0 ERROR.             * ON ERROR STOP         TST033   <>---ERROR
    - *-----------------------------------------------------------*    TST033
    - SQL CHAIN KKDA WHERE KKDNR = :KEY                               TST033
    - SQL DECLARE  TST033CPGCU01 CURSOR FOR SELECT                  *  TST033
    - SQL  A.KKDNR,A.KFIRMA,A.KFBEZ,A.KPLZ,A.KLAND,A.KORT,A.KSTR1,  *  TST033
    - SQL  A.KSTR2,A.KGPART,A.KTELNR,A.KKNAM,A.BKDNR                *  TST033
    - SQL  FROM SQLDBA.KUNDENB A                                    *  TST033
    - SQL  WHERE KKDNR = :KEY                                         TST033
 70 -    SQCODD =  SQCODE.                                            TST033      I
    - SQL  OPEN TST033CPGCU01                                         TST033
 81 -    SQCODO =  SQCODE.                                            TST033      I
    - SQL  FETCH TST033CPGCU01 INTO :KKDNR,:KFIRMA,:KFBEZ:H1,       *  TST033
    - SQL  :KPLZ:H4,:KLAND:C2,:KORT:H3,:KSTR1:H6,:KSTR2:H7,         *  TST033
    - SQL  :KGPART:KG,:KTELNR:H8,:KKNAM:C1,:BKDNR:A0                  TST033
113 -    SQCODF =  SQCODE.                                            TST033      I
    - SQL  CLOSE TST033CPGCU01                                        TST033
123 -    SQCODC =  SQCODE.                                            TST033      I
    - SQL CHAIN KKDA                                                *  TST033
    - SQL       WHERE KKDNR = :KEY AND KFIRMA LIKE  'LATTW%'          TST033
    - SQL DECLARE  TST033CPGCU02 CURSOR FOR SELECT                  *  TST033
    - SQL  A.KKDNR,A.KFIRMA,A.KFBEZ,A.KPLZ,A.KLAND,A.KORT,A.KSTR1,  *  TST033
    - SQL  A.KSTR2,A.KGPART,A.KTELNR,A.KKNAM,A.BKDNR                *  TST033
    - SQL  FROM SQLDBA.KUNDENB A                                    *  TST033
    - SQL       WHERE KKDNR = :KEY AND KFIRMA LIKE  'LATTW%'          TST033
134 -    SQCODD =  SQCODE.                                            TST033      I
    - SQL  OPEN TST033CPGCU02                                         TST033
145 -    SQCODO =  SQCODE.                                            TST033      I
    - SQL  FETCH TST033CPGCU02 INTO :KKDNR,:KFIRMA,:KFBEZ:H1,       *  TST033
    - SQL  :KPLZ:H4,:KLAND:C2,:KORT:H3,:KSTR1:H6,:KSTR2:H7,         *  TST033
    - SQL  :KGPART:KG,:KTELNR:H8,:KKNAM:C1,:BKDNR:A0                  TST033
177 -    SQCODF =  SQCODE.                                            TST033      I
    - SQL  CLOSE TST033CPGCU02                                        TST033
187 -    SQCODC =  SQCODE.                                            TST033      I
188 -     CAB SQCODE >< 0 ERROR.              * ON ERROR STOP         TST033   <>---ERROR
    - *-----------------------------------------------------------*    TST033
    - SQL SETLL TST033CURS01 KKDA WHERE A.KKDNR = :KEY                TST033
    - SQL DECLARE  TST033CURS01 CURSOR FOR SELECT                   *  TST033
    - SQL  A.KKDNR,A.KFIRMA,A.KFBEZ,A.KPLZ,A.KLAND,A.KORT,A.KSTR1,  *  TST033
    - SQL  A.KSTR2,A.KGPART,A.KTELNR,A.KKNAM,A.BKDNR                *  TST033
    - SQL  FROM SQLDBA.KUNDENB A                                    *  TST033
    - SQL  WHERE A.KKDNR = :KEY                                       TST033
199 -    SQCODD =  SQCODE.                                            TST033      I
    - SQL  OPEN TST033CURS01                                          TST033
210 -    SQCODO =  SQCODE.                                            TST033      I
211 -     CAB SQCODE >< 0 ERROR.              * ON ERROR STOP         TST033   <>---ERROR
    - *-----------------------------------------------------------*    TST033
```

```
PROGRAMM OHNE TITEL    99999 LATTWEIN    19.01.06    14.08 UHR     HL1 2.6     SEITE 3
```

```
    - SQL READ TST033CURS01 KKDA
    - SQL  FETCH TST033CURS01 INTO :KKDNR,:KFIRMA,:KFBEZ:H1,        * TST033
    - SQL  :KPLZ:H4,:KLAND:C2,:KORT:H3,:KSTR1:H6,:KSTR2:H7,         * TST033
    - SQL  :KGPART:KG,:KTELNR:H8,:KKNAM:C1,:BKDNR:A0                  TST033
243 -    SQCODF =  SQCODE.                                           TST033     I
244 -     CAB SQCODE >< 0 ERROR.               * ON ERROR STOP       TST033   <>---ERROR
    - *-------------------------------------------------------------* TST033
    - SQL RNDOM TST033CURS01                                          TST033
    - SQL  CLOSE TST033CURS01                                         TST033
254 -    SQCODC =  SQCODE.                                           TST033     I
255 -     CAB SQCODE >< 0 ERROR.               * ON ERROR STOP       TST033   <>---ERROR
    - *-------------------------------------------------------------* TST033
    - SQL UPDAT KKDA WHERE KKDNR = :KEY                               TST033
    - SQL UPDATE SQLDBA.KUNDENB SET                                 * TST033
    - SQL  KKDNR=:KKDNR,KFIRMA=:KFIRMA,KFBEZ=:KFBEZ:H1,             * TST033
    - SQL  KPLZ=:KPLZ:H4,KLAND=:KLAND:C2,KORT=:KORT:H3,             * TST033
    - SQL  KSTR1=:KSTR1:H6,KSTR2=:KSTR2:H7,                         * TST033
    - SQL  KGPART=:KGPART:KG,KTELNR=:KTELNR:H8,                     * TST033
    - SQL  KKNAM=:KKNAM:C1,BKDNR=:BKDNR:A0                          * TST033
    - SQL  WHERE KKDNR = :KEY                                         TST033
288 -     CAB SQCODE >< 0 ERROR.               * ON ERROR STOP       TST033   <>---ERROR
    - *-------------------------------------------------------------* TST033
    - SQL WRITE KKDA                                                  TST033
    - SQL INSERT INTO SQLDBA.KUNDENB (                              * TST033
    - SQL  KKDNR,KFIRMA,KFBEZ,KPLZ,KLAND,KORT,KSTR1,KSTR2,KGPART,   * TST033
    - SQL  KTELNR,KKNAM,BKDNR )                                     * TST033
    - SQL  VALUES ( :KKDNR,:KFIRMA,:KFBEZ:H1,:KPLZ:H4,:KLAND:C2,    * TST033
    - SQL  :KORT:H3,:KSTR1:H6,:KSTR2:H7,:KGPART:KG,:KTELNR:H8,      * TST033
    - SQL  :KKNAM:C1,:BKDNR:A0 )                                      TST033
320 -     CAB SQCODE >< 0 ERROR.               * ON ERROR STOP       TST033   <>---ERROR
    - *-------------------------------------------------------------* TST033
321 -     ERROR.                                                      TST033 ERROR ---I
322 -        EXHM ERRCHK.                      * CHECK FÜR SQLERROR  TST033
323 -        EXSR COMIT.                                              TST033    I---COMIT
    - *-------------------------------------------------------------* TST033
324 -     COMIT BEGSR.                                                TST033 COMIT ------
    -       *SQL COMMIT WORK.                                         TST033
325 -     ENDSR.                                                      TST033     I
    - -O.                                                             TST033
326 -   FIELD FEHLER.                                                 TST033 FEHLER--EDIT
327 -               24 '     SQL FEHLER:       '.                    TST033      2  23
328 -      KEY       5.                                               TST033 KEY  2   5,0
329 -      SQCODE   40 EDITCODE Z.                                    TST033 SQCO 34  7,0
```

ENDE DER UMWANDLUNG

KEINE FEHLER

Dynamische Befehle

Bei den dynamischen Befehlen ergibt sich die WHERE-Clause erst zur Ausführungszeit, z.B. durch Eingaben des Anwenders.

Das bedeutet, dass der gesamte Befehl bis auf die WHERE-Clause codiert wird. Als WHERE-Clause wird ein Feld angegeben, das dann zur Ausführungszeit die entsprechende Bedingung enthält.

Es können bis zu drei dynamische Befehle in einem Programm definiert werden (DYNPREP1 - DYNPREP3), die intern in vorhandene Statements gesetzt und dann dynamisch ausgeführt werden.

DYNPREP1 ist reserviert für Abfragen, die viele Spaltennamen abfragen. Das interne Feld hierfür ist 2046 Stellen groß.

DYNPREP2 und DYNPREP3 können je bis zu 510 Stellen groß definiert werden.

Bei den dynamischen Befehlen sind in der WHERE-Clause lediglich Konstanten als Parameter unterstützt.

Zurzeit sind nur dynamische SELECT-Befehle möglich.

Der Programmierer braucht bei den dynamischen Befehlen keinen CURSOR zu definieren, da jedem DYNPREP-Befehl intern ein CURSOR zugeordnet ist.

DYNPREP1 SETLL

Der Befehl

- SQL DYNPREP1 SETLL ACCNAM :WCOND

wird umgesetzt in folgende SQL-Befehle:

- SQL DYNPREP1 SELECT ... FROM ... :WCOND
- SQL DYNPREP1 OPEN

DYNPREP1 SETLD

Der Befehl

- SQL DYNPREP1 SETLD ACCNAM :WCOND

wird umgesetzt in folgende SQL-Befehle:

- SQL DYNPREP1 SELECT DISTINCT ... FROM ... :WCOND
- SQL DYNPREP1 OPEN

Alle Informationen werden dem angegebenen Access entnommen.

Die WHERE-Clause wird als Variable zur Ausführungszeit im Feld WCOND zusammengestellt und dem SQL-Ausführungsprogramm übergeben. Es werden bis zu 3 Variable unterstützt, die eine WHERE-Bedingung enthalten

Demnach kann eine Abfrage je nach Anforderung gestaltet werden. Es kann z.B. einmal in der Kundentabelle nach Ort und Straße gesucht werden oder nach Name oder nach anderen Suchargumenten. All diese Abfragen können im Programm an einer Stelle mit dem gleichen Befehl abgefragt werden.

Es wird eine interne Fehlerprüfung durchgeführt. Dies bedeutet, dass der OPEN nur ausgeführt wird, wenn der SELECT ohne Fehler abgewickelt wurde.

DYNPREP1 READ

Der Befehl

- SQL DYNPREP1 READ ACCNAM

wird umgesetzt in folgenden SQL Command:

- SQL DYNPREP1 FETCH ... INTO

Alle Informationen werden dem angegebenen Access entnommen.

Die Host-Variablen werden in die INTO-Clause eingesetzt und mit den entsprechenden Spalten-Daten gefüllt. Der Cursor-Name wird über den Befehl (DYNPREP1/2/3) intern zugeordnet.

DYNPREP1 RNDOM

Der Befehl

- SQL DYNPREP1 RNDOM

wird umgesetzt in folgenden SQL Command:

- SQL DYNPREP1 CLOSE

Der interne CURSOR für DYNPREP1 wird geclosed.

Besonderheiten

Um unnötige Fehlerquellen zu vermeiden, sind hier einige Besonderheiten aufgeführt, die entweder im Programm oder bei der Ausführung zu berücksichtigen sind.

- Bevor eine Umwandlung mit SQL gestartet wird, ist sicherzustellen, dass SQL verfügbar ist.

- Wird mit SETLL und READ programmiert, so muss SETLL im Programmcode vor dem READ liegen, d.h. es ist nicht möglich, das SETLL in einer Subroutine durchzuführen, während das READ im Hauptprogramm liegt.

- Der Vergleich mit LIKE in der WHERE-Clause mit einer Konstanten ist dahingehend unterstützt, dass die gesuchten Zeichen und ein %-Zeichen angegeben werden.

  Beispiel:

```
... WHERE ORT LIKE 'K”%'
```

  Wird der Vergleich mit einer Variablen durchgeführt, so sind alle Unbekannten in dem Feld mit %-Zeichen anzugeben.

  Beispiel:

```
FILL '%' TO ALPHA
MOVEL 'K”' TO ALPHA
... WHERE ORT LIKE ALPHA
```

SQL-Tabellen für CPG5 und QSAT

Packages:

Name	Creator
CPGSDD	SQLDBA
CPGFDD	SQLDBA
CPGTDD	SQLDBA
CPGADD	SQLDBA
CPGZDD	SQLDBA

SQL Packages:

- CPGPREP0
- CPGPREP2
- QSATPRG
- HMHQSAD
- HMHQSAL
- HMHQSAX
- HMHQSDX
- HQSCOL
- HMXPRSQ
- HMYPRSQ

Installation

Die folgende Installationsbeschreibung ist für VSE-Anwender und das Produkt CPG3/SQL - Interface.

Die Installation des CPG3/SQL-Interface erfordert den beschriebenen Ablauf:

1. Installationsprozeduren anlegen	(910)	X
2. Installation des CPG3/SQL-Bandes	(920)	X
3. Installation der SQL-Access-Module	(930)	X
Unbedingt bei Releasewechsel erforderlich		X

Installationsprozeduren anlegen

Es werden die Installationsprozeduren CPGINST1 und CPGINST2 aus der CPG 2.6 Installation verwendet (siehe Abschnitt 9030 in der CPG-Installationsanweisung).

Die CPG Installationsprozedur CPGINST2 muss in den Searchanweisungen die aktuelle DB2 Produktionslibrary beinhalten. Sie sollte wie folgt aussehen:

```
// LIBDEF SOURCE,SEARCH=(Lib.Sublib,PRD2.DB2720),TEMP
// LIBDEF OBJ,SEARCH=(Lib.Sublib,PRD2.DB2720),TEMP
```

Statt PRD2.DB2720 kann je nach DB2 Stand auch eine andere Sublib definiert sein.

Installation des Bandes

Das Installationsband wird im VM mit den Befehlen:

```
TAPE FSF 3
TAPE LOAD * * FM
```

geladen. "FM" gibt die Minidisk an, auf der sich der Bandinhalt nachher befindet. Natürlich sollte dem VM-USER eine Bandstation als 181 zugeordnet sein.

Installation der SQL-Access-Module

Die Installation von QSAT wird im VM über eine REXX Prozedur durchgeführt. Hierbei sind einige Fragen zu beantworten, damit JCL etc. richtig erstellt werden. Der Aufruf erfolgt über

```
QSATINST                        <<== Eingabe VM
```

Folgende Fragen sind zu beantworten:

```
Sind in der Procedure-Library die CPG Installationsproze-
duren CPGINST1 und CPGINST2 enthalten?
Sind die Proceduren CPGINST1 und CPGINST2 an Ihre
Systemumgebung angepasst?                                    Y/N
Y                               <<== Eingabe VM
```

```
Ist CPG-Version 2.6 installiert?                             Y/N
Y                               <<== Eingabe VM
```

```
In welcher VSE-Maschine soll installiert werden?              x
VSETST                          <<== Eingabe der VSE-Maschine
```

```
Unter welchem Filemode ist QSAT installiert?                  x
Q                               <<== Filemode
```

```
CPG Rel. = Y  VSE = VSETST  CPG Release = 2.6  FM = Q
```

```
Power Job Klasse angeben:                                     x
W                               <<== Power RDR Klasse
```

```
Power SYSID oder nichts angeben:                                        x
                              <<== (nichts angeben)
```

```
Power LST Klasse angeben:                                               x
V                             <<== Powr LST Klasse
```

```
Power SPOOLTO DEST angeben: (z. B. MAINT)                               x
MAINT                         <<== DEST für Spoolto
```

```
Um bestimmte SQL Jobs ablaufen zu lassen, muss der SQLUSER:
SQLDBA mit Passwort bekannt sein. Daher bitte das Passwort
für SQLDBA hier eingeben:
SQLDBAPW                      <<== Passwort für SQLDBA
```

```
Wie heißt die Installationslibrary für SQL?                             x
Bitte Lib.Sublib hier eingeben (z. B. PRD2.DB2720):       Lib.Sublib
PRD2.DB2720                   <<== Power RDR Klasse
```

```
Wie heißt die Prozedur, in der die SQLDBSPACES definiert
sind? (z. B. ARISLIBP)
Wenn diese im Standardlabel definiert sind, hier NONE eingeben.
Ansonsten hier den Procedure Namen eintragen:               xxxxxxxx
ARISLIBP                      <<== SQL Procedure ARISLIBP
```

```
Wenn QSAT schon vorher installiert war ohne die Funktion
Current Date ...
Soll die Erweiterung installiert werden, wenn ja Migrate,
sonst NO eintragen:                                       NO/MIGRATE
NO                            <<== Bei Neuinstallation NO
```

```
Ist ein DB-Space mit dem Namen SQLDBA.CPGDD als Public
vorhanden?                                                        Y/N
N                          <<== N, wenn Neuinstallation
```

```
Sind die Tabellen CPGSDD, CPGFDD, CPGTDD, CPGADD, CPGZDD
als Public vorhanden?                                             Y/N
Soll die Erweiterung installiert werden, wenn ja Migrate,
sonst NO eintragen:                                          NO/MIGRATE
N                          <<== N, wenn Neuinstallation
```

```
Es wird ein Job erstellt, der die Tabellen CPGSDD, CPGFDD,
CPGTDD, CPGADD, CPGZDD löscht.
Wenn dieser Job laufen soll, bitte eingeben:                      DROP
Wenn nicht, bitte eingeben:                                       SKIP
SKIP                       <<== SKIP, wenn Neuinstallation
```

```
Der folgende Job braucht eine Bandstation. Bitte Y/N ein-
geben, wenn der Job laufen kann (Laden QSF Masken).               Y/N
```