Hofstetter

- M141 - Datenbanksystem in Betrieb nehmen
  - Db2 Hofstetter Aufgabe 2
  - Db2 Hofstetter Aufgabe 3
  - Db2 Hofstetter Aufgabe 5
  - db2 Befehle

M141 - Datenbanksystem in Betrieb nehmen

Db2 Hofstetter Aufgabe 2

Aufgabenstellung:

In dieser Übung sollen Sie die folgenden Objekte erstellen. Ziel dieses Auftrages ist es, dass Sie sich mit den entsprechenden CREATE Statements auseinandersetzen und auch lernen wie Sie mit SQL-Fehlern umgehen müssen resp. wie Sie diese beheben.

Es sollen zwei Tabellen in der Datenbank DBBW001 erstellt werden. Für das Erstellen der benötigten Objekte soll der Instanz-User db2inst1 verwendet werden.

• Erstellen Sie eine Tabelle BBWM141.TKLASSE mit folgenden Attributen (die Daten dieser Tabelle sollen physisch im Tablespace SKLASSE gespeichert werden):

◦ Die Spalte TKLASSEID soll als Primärschlüssel definiert werden. Als Datentyp soll INTEGER für diese Spalte gewählt werden.

◦ Die Spalte KLASSE ist ein eindeutiger Wert mit der Bezeichnung der Klasse (z.B. 5ip21a, 5ip21b, etc.). Für diese Spalte muss immer ein Wert erfasst werden. Als Datentyp soll VARCHAR mit einer maximalen Grösse von 15 Zeichen definiert werden.

◦ In der Spalte KLASSENLEHRPERSON soll der Name des zuständigen Klassenlehrers erfasst werden. Als Datentyp soll VARCHAR mit einer maximalen Grösse von 50 Zeichen definiert werden. Das Erfassen dieser Information ist optional.

◦ In der Spalte LASTUPDATE soll festgehalten werden WANN dieser Datensatz gespeichert wurde. Dazu soll als Datentyp TIMESTAMP (Datum und Zeit) festgelegt werden. Falls beim Erfassen des Datensatzes kein Wert (Timestamp) angegeben wird, soll der DBM (Database Manager) diesen Wert automatisch einfügen.

• Erstellen Sie eine Tabelle BBWM141.TLERNENDE mit folgenden Attributen (die Daten dieser Tabelle sollen physisch im Tablespace SLERNENDER gespeichert werden):

◦ TLERNENDEID soll als Primärschlüssel festgelegt werden. Als Datentyp soll INTEGER für diese Spalte definiert werden. Wird beim Erfassen kein Primärschlüssel mitgegeben, soll der DBM diesen selbständig vergeben (IDENTITY).

◦ Die Spalte TKLASSEID soll als Fremdschlüssel definiert werden, damit die Zuteilung eines Lernenden zu einer Klasse möglich ist. Das Erfassen der Klasse ist optional. Die Beziehung zwischen den beiden Tabellen soll vom DBM verifiziert werden (RESTRICT Option).

◦ Die Spalten NAME und VORNAME sind als VARCHAR Felder mit einer maximalen Grösse von 50 Zeichen zu definieren. Beide Werte müssen zwingend erfasst werden. Zusätzlich soll für diese beiden Spalten auch ein kombinierter Index (Name und Vorname) definiert werden, damit Abfragen mit den Namen schnell sind.

◦ MAILADDR, in diesem Feld soll optional die Mail-Adresse erfasst werden können. Eine Mail-Adresse kann maximal 50 Zeichen lang sein.

◦ In der Spalte LASTUPDATE soll festgehalten werden WANN dieser Datensatz gespeichert wurde. Dazu soll als Datentyp TIMESTAMP (Datum und Zeit) festgelegt werden. Falls beim Erfassen des Datensatzes kein Wert (Timestamp) angegeben wird, soll der DBM (Database Manager) diesen Wert automatisch einfügen. Erstellen Sie dazu folgende drei SQL-Scripts (Files):

• AUFG02D.sql – in diesem SQL-Script sollen die Objekte vorgängig gelöscht werden, damit jederzeit eine "saubere" Ausgangslage erstellt werden kann.

• AUFG02C.sql – in diesem SQL-Script sollen alle benötigten Objekte erstellt werden.

• AUFG02INS.sql – in diesem SQL-Script sollen Sie in beiden Tabellen mittel INSERT Statement mindestens einen Datensatz einfügen (z.B. Ihre Klasse und Ihre Personen-Daten).

Scripte:

```
-- Löschen der Tabellen, falls vorhanden
DROP TABLE IF EXISTS BBWM141.TLERNENDE;
DROP TABLE IF EXISTS BBWM141.TKLASSE;
```

```
/*
CREATE TABLESPACE SKLASSE
   PAGESIZE 4K
   MANAGED BY AUTOMATIC STORAGE;

CREATE TABLESPACE SLERNENDER
   PAGESIZE 4K
   MANAGED BY AUTOMATIC STORAGE;
*/


-- Erstellen der Tabelle BBWM141.TKLASSE
CREATE TABLE BBWM141.TKLASSE (
    TKLASSEID INTEGER NOT NULL GENERATED ALWAYS AS IDENTITY (START WITH 1, INCREMENT BY 1),
    KLASSE VARCHAR(15) NOT NULL,
    KLASSENLEHRPERSON VARCHAR(50),
    LASTUPDATE TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL,
    PRIMARY KEY (TKLASSEID)
) IN SKLASSE;

-- Erstellen der Tabelle BBWM141.TLERNENDE
```

```
CREATE TABLE BBWM141.TLERNENDE (
    TLERNENDEID INTEGER NOT NULL GENERATED ALWAYS AS IDENTITY (START WITH 1, INCREMENT BY 1),
    TKLASSEID INTEGER,
    NAME VARCHAR(50) NOT NULL,
    VORNAME VARCHAR(50) NOT NULL,
    MAILADDR VARCHAR(50),
    LASTUPDATE TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL,
    PRIMARY KEY (TLERNENDEID),
FOREIGN KEY (TKLASSEID) REFERENCES BBWM141.TKLASSE (TKLASSEID) ON DELETE RESTRICT ON UPDATE NO A
) IN SLERNENDER;

-- Erstellen des kombinierten Index für BBWM141.TLERNENDE
CREATE INDEX IX_TLERNENDE_NAME_VORNAME ON BBWM141.TLERNENDE (NAME, VORNAME);
```

```
-- Einfügen von Daten in BBWM141.TKLASSE
INSERT INTO BBWM141.TKLASSE (KLASSE, KLASSENLEHRPERSON) VALUES ('5ip21a', 'Max Mustermann');
INSERT INTO BBWM141.TKLASSE (KLASSE, KLASSENLEHRPERSON) VALUES ('5ip21b', 'Maria Musterfrau');

-- Einfügen eines Datensatzes mit MAILADDR
INSERT INTO BBWM141.TLERNENDE (NAME, VORNAME, MAILADDR) VALUES ('Schmidt', 'Peter', 'peter.schmidt@ex

-- Einfügen eines Datensatzes ohne MAILADDR
INSERT INTO BBWM141.TLERNENDE (NAME, VORNAME) VALUES ('Müller', 'Anna');

-- Einfügen eines Datensatzes mit einem nicht existierenden TKlasseID
INSERT INTO BBWM141.TLERNENDE (NAME, VORNAME, MAILADDR) VALUES ('Bauer', 'Hans', 'hans.bauer@example
```

Db2 Hofstetter Aufgabe 3

Aufgabenstellung:

In der Praxis muss ein DBA (Datenbank-Administrator) für die Wartung von Datenbanken und Applikationen sehr oft SQL-Scripts für mehrere Datenbanken ausführen. Das Ausführen von SQL-Scripts via einem SQL GUI Client (z.B. DBeaver) lässt sich kaum automatisieren und ist zudem auch fehleranfällig, da sehr viele manuelle Tasks ausgeführt werden müssen.

Entsprechend wird in der Regel bei Wartungs- und Admin-Aufgaben das CLI verwendet und die Commands oder SQL-Statements werden via dem Command Line Processor (CLP) ausgeführt. Eine Übersicht der verschiedenen Parameter und Optionen des Db2 CLP's haben Sie bereits erhalten (siehe OLAT Verzeichnis 10-Theorie, Dokument M141_Einfuehrung_CLP_V1.2.pdf). Hier nochmals die wichtigsten Parameter des Db2 CLPs für das Ausführen eines SQL-Scripts:

db2 -tsv -f /bbw/myscripts/meinScript.sql

Die Optionen haben dabei folgende Bedeutung:

-f file Eingabedatei lesen, die Datei enthält die auszuführenden Statements -t Statements sind mit einem Semikolon (;) terminiert -s Abbruch der Verarbeitung beim ersten Fehler -v Ausgabe des aktuellen Befehls während der Ausführungs Fehler (auf stdout)

Der CLP gibt bei der Ausführungs Fehler eines Commands oder Statements immer Meldungen aus, via denen ersichtlich ist ob die Ausführungs Fehler erfolgreich war oder ein Fehler aufgetreten ist. Die Messages von Db2 haben eine sogenannte Message-ID (Bsp: DB20000I, DB21034E, SQL0204N, etc.) und einen Message-Text.

Bsp: SQL0204N "MIGDBADM.MYTABLE" is an undefined name. SQLSTATE=42704

Der Suffix einer Message-ID definiert dabei ob es sich um eine Info-, Warning- oder Error-Message handelt.

C Indicates a severe or critical error message. E Indicates an urgent error message. The E suffix is for non-SQL messages. N Indicates an error message. W Indicates a warning message. I Indicates an informational message.

Da Meldungen oder Message-IDs in Scripts nicht einfach zu interpretieren sind, setzt der Db2 CLP auch einen ExitCode via dem die Ausführungs Fehler protokolliert wird:

0 command or SQL statement executed successfully 1 SELECT or FETCH statement returned no rows 2 Db2 command or SQL statement warning 4 Db2 command or SQL statement error 8 Command line processor system error

Ein Return-Code wird nur gesetzt wenn ein Script beendet wird oder der interactive Mode des CLPs verlassen wird. Basierend auf dem Exit-Code des CLPs und den verschiedenen CLP Options lässt sich das Resultat der Ausführungs Fehler in Shell Scripts sehr einfach verifizieren.

Im vorangegangenen Auftrag haben Sie zwei Tabellen erstellt und die zugehörigen SQL-Statements vermutlich via DBeaver ausgeführt. In diesem Auftrag sollen Sie nun ausschliesslich mit dem CLP arbeiten.

Kopieren Sie aus dem OLAT die folgenden Files auf Ihren DB-Server

OLAT-Source-Directory: Unterlagen / Aufträge → 50-Uebungen → Auftrag-03 Ziel-Verzeichnis auf Server: /bbw/uebungen/ueb03/

Zu kopierende Files:

UEB03-00.sql leeres File für Vorbereitung und/oder Cleanup UEB03-01.sql Erstellt 2 Tabellen im Schema UEB03A UEB03-02.sql Erstellt 2 Tabellen im Schema UEB03B UEB03-03.sql Erstellt 2 Tabellen im Schema UEB03C UEB03-04.sql Erstellt 16 Tabellen im Schema UEB03D

Diese SQL-Scripts sollen nun via CLP ausgeführt werden. Sie werden bemerken, dass einige dieser SQL Scripts fehlerhaft sind, d.h. sie müssen korrigiert werden oder Sie müssen die Voraussetzungen schaffen, dass die Scripts laufen und die Objekte in der Datenbank DBBW001 erstellt werden.

Ziel dieser Übung ist es, dass Sie den Umgang mit dem CLP lernen und auch Fehler-Meldungen interpretieren können. Notieren Sie die wesentlichen Punkte, die Sie ausführen (Bsp: 1. Login, 2. db2 ....). Die in dieser Übung erarbeiteten Kompetenzen können analog in einer Prüfung erforderlich sein.

Beispiel: Ausführen des SQL-Scripts UEB03-00.sql

• Login mit User db2inst1 via puTTY • Wechsel ins Directory: /bbw/uebungen/ueb03 • Ausführen des Scripts: db2 -tvsf UEB03-00.sql ◦ Oups: SQL1032N No start database manager command was issued. SQLSTATE=57019

Anscheinend wurde der DBM noch nicht gestartet ... (Starten Sie den DBM)

• Ausführen des Scripts: db2 -tvsf UEB03-00.sql ◦ Oups: SQL1024N A database connection does not exist. SQLSTATE=08003

Anscheinend haben Sie noch keinen CONNECT ausgeführt ... (Machen Sie den CONNECT)

• Ausführen des Scripts: db2 -tvsf UEB03-00.sql

COMMIT WORK DB20000I The SQL command completed successfully.

Fazit:

1. Kontrollieren ob DBM läuft
2. Verbindung zur Datenbank aufbauen mit dem richtigen User (db2 CONNECT TO ...)
3. Ausführen des/der SQL-Scripts (db2 -tvsf ...)
4. Kontrolle der Ausgabe (darf keine Fehler-Meldungen enthalten)
5. Verbindung zur Datenbank trennen (db2 CONNECT RESET / db2 TERMINATE)

Führen Sie nun das Script UEB03-01.sql aus und analysieren Sie den Fehler. Ausführen des Scripts: db2 -tvsf UEB03-01.sql

Fehler: SQL0204N "STBDPOSTFILESTREET" is an undefined name.

Der Error SQL0204N bedeutet, dass das referenzierte Objekt in der Datenbank nicht existiert.

Der Grund kann dabei ein falscher Name sein (Schreibfehler) oder das Objekt wurde noch nicht erstellt.

Hier existiert der Tablespace STBDPOSTFILESTREET noch nicht, d.h. der TABLESPACE muss zuerst mit dem CREATE TABLESPACE Statement angelegt werden (vor dem Erstellen der Tabelle).

Erstellen Sie nun den benötigten Tablespace, die Syntax für das CREATE TABLESPACE Statement finden Sie im Knowledge Center (Internet) oder der bereitgestellten PDF-Dokumentation.

Überlegen Sie WO Sie das/die notwendigen SQL Statements einfügen (in einem eigenen SQL-Script, im Script UEB03-00.sql oder am Anfang des Files UEB03-01.sql).

Führen Sie die vier SQL-Scripts (UEB03-01.sql bis UEB03-04.sql) aus und analysieren Sie die Fehler-Meldungen. Korrigieren Sie die Scripts, d.h. erstellen Sie die fehlenden Objekte oder korrigieren Sie die Syntax-Fehler.

Überlegen Sie sich auch wie Sie mit den Fehlern umgehen (ev. bricht das Script ja in der Hälfte ab, d.h. ein Restart ab Beginn ist ev. nicht möglich). Baue Sie ein „Cleanup-Script“, das alle Objekte zuerst löscht (DROP), damit Sie immer eine saubere Ausgangslage bereitstellen können.

Hier noch ein Hinweis:

- wenn Sie eine Tabelle löschen, werden auch automatisch alle Indizes der Tabelle gelöscht
- wenn Sie einen Tablespace löschen werden auch die darin enthaltenen Tabellen, Indizes, etc. gelöscht

---

Beispiel:

Um den DBM (Database Manager) zu starten, müssen Sie das folgende Kommando ausführen:

Befehl:

```
db2start
```

Ausgabe:

```
[db2inst1@localhost ueb03]$ db2start
03/25/2023 14:20:43     0   0   SQL1026N  The database manager is already active.
```

Nachdem der DBM erfolgreich gestartet wurde, können Sie sich mit dem Datenbank-Server verbinden und das SQL-Skript ausführen, indem Sie die folgenden Schritte ausführen:

Login mit User db2inst1 via puTTY

Wechseln Sie ins Verzeichnis: /bbw/uebungen/ueb03

Verbinden Sie sich mit dem Datenbank-Server durch Ausführen des Befehls:

Befehl:

```
db2 connect to <database_name>
```

Ausgabe:

```
[db2inst1@localhost ueb03]$ db2 connect to dbbw001

   Database Connection Information

 Database server        = DB2/LINUXX8664 11.5.8.0
 SQL authorization ID   = DB2INST1
 Local database alias   = DBBW001
```

<database_name> ist der Name der Datenbank, mit der Sie sich verbinden möchten.

Führen Sie das SQL-Skript aus, indem Sie das folgende Kommando ausführen:

Befehl:

```
db2 -tvsf UEB03-00.sql
```

Ausgabe:

```
[db2inst1@localhost ueb03]$ db2 -tvsf UEB03-00.sql
COMMIT WORK
DB20000I  The SQL command completed successfully.
```

```
********************* END OF SCRIPT *********************
```

Wenn das SQL-Skript erfolgreich ausgeführt wird, sollte die Meldung "DB20000I The SQL command completed successfully." angezeigt werden.

```
-- Löschen der Tabellen, falls sie existieren
DROP TABLE IF EXISTS UEB03A.TBDPOSTFILESTREET;
DROP TABLE IF EXISTS UEB03A.TBDPOSTFILEZIP;

-- Löschen der Tablespaces
DROP TABLESPACE STBDPOSTFILESTREET;
DROP TABLESPACE STBDPOSTFILEZIP;
```

UEB03-01.sql:

Ausführungs Fehler:

```
[db2inst1@localhost ueb03]$ db2 -tvsf UEB03-01.sql
CREATE TABLE UEB03A.TBDPOSTFILESTREET ( TBDPOSTFILESTREETID  BIGINT NOT NULL GENERATED BY DEFAULT
DB21034E  The command was processed as an SQL statement because it was not a
valid Command Line Processor command.  During SQL processing it returned:
SQL0204N  "STBDPOSTFILESTREET" is an undefined name.  SQLSTATE=42704
```

Zuvor:

```
-- ------------------------------------------------------------------------------------------------
-- Dies ist ein Kommentar, d.h. diese Zeile wird NICHT ausgefuehrt
-- ------------------------------------------------------------------------------------------------


CREATE TABLE UEB03A.TBDPOSTFILESTREET (
      TBDPOSTFILESTREETID  BIGINT NOT NULL GENERATED BY DEFAULT AS IDENTITY( START WITH 1, INCREMENT
      STREET               VARCHAR(120),
      ZIP4                 CHAR(4),
      ZIPADDON             CHAR(2),
      ZIP6                 CHAR(6),
      SHORTCITY            VARCHAR(54),
      COUNTRY              CHAR(2),
      ADDRSCANTON          CHAR(2),
      STREETNR             INTEGER,
      STREETNRADDON        VARCHAR(6),
      POLGNAME             VARCHAR(60),
      POLGCANTON           CHAR(2),
      BFSNR                INTEGER,
      LASTUPDATED          TIMESTAMP NOT NULL DEFAULT CURRENT TIMESTAMP
)
IN STBDPOSTFILESTREET
;
```

```
COMMENT ON COLUMN UEB03A.TBDPOSTFILESTREET.TBDPOSTFILESTREETID IS '''Primary Key''';
COMMENT ON COLUMN UEB0A.TBDPOSTFILESTREET.STREET IS 'Street name';
COMMENT ON COLUMN UEB03A.TBDPOSTFILESTREET.ZIP4 IS '4 digits of ZIP';
COMMENT ON COLUMN UEB03A.TBDPOSTFILESTREET.ZIPADDON IS 'ZIP addon';
COMMENT ON COLUMN UEB03A.TBDPOSTFILESTREET.ZIP6 IS 'Full ZIP (6 digits)';
COMMENT ON COLUMN UEB03A.TBDPOSTFILESTREET.SHORTCITY IS 'Short city name (27 characters)';
COMMENT ON COLUMN UEB03A.TBDPOSTFILESTREET.COUNTRY IS 'Country';
COMMENT ON COLUMN UEB03A.TBDPOSTFILESTREET.ADDRSCANTON IS 'Address canton';
COMMENT ON COLUMN UEB03A.TBDPOSTFILESTREET.STREETNR IS 'Street no w/o addon';
COMMENT ON COLUMN UEB03A.TBDPOSTFILESTREET.STREETNRADDON IS 'Addon (like 22A)';
COMMENT ON COLUMN UEB03A.TBDPOSTFILESTREET.POLGNAME IS 'Pol. municipality name';
COMMENT ON COLUMN UEB03A.TBDPOSTFILESTREET.POLGCANTON IS 'Municipality canton';
COMMENT ON COLUMN UEB03A.TBDPOSTFILESTREET.LASTUPDATED IS 'timestamp of last change';


CREATE TABLE UEB03A.TBDPOSTFILEZIP (
      TBDPOSTFILEZIPID     BIGINT NOT NULL GENERATED BY DEFAULT AS IDENTITY( START WITH 1, INCREMENT BY
      ZIP4                 CHAR(4),
      ZIPADDON             CHAR(2),
      ZIP6                 CHAR(6),
      CITY                 VARCHAR(60),
      SHORTCITY            VARCHAR(54),
      COUNTRY              CHAR(2),
      BFSNR                INTEGER,
      POLGNAME             VARCHAR(60),
      CANTON               CHAR(2),
      LASTUPDATED          TIMESTAMP NOT NULL DEFAULT CURRENT TIMESTAMP
)
IN STBDPOSTFILEZIP
;

COMMENT ON COLUMN UEB03A.TBDPOSTFILEZIP.TBDPOSTFILEZIPID IS '''Primary Key''';
COMMENT ON COLUMN UEB03A.TBDPOSTFILEZIP.ZIP4 IS '4 digits of ZIP';
COMMENT ON COLUMN UEB03A.TBDPOSTFILEZIP.ZIPADDON IS 'ZIP addon';
COMMENT ON COLUMN UEB03A.TBDPOSTFILEZIP.ZIP6 IS 'Full ZIP (6 digits)';
COMMENT ON COLUMN UEB03A.TBDPOSTFILEZIP.CITY IS 'City name
';
COMMENT ON COLUMN UEB03A.TBDPOSTFILEZIP.SHORTCITY IS 'Short city name (27 characters)';
COMMENT ON COLUMN UEB03A.TBDPOSTFILEZIP.COUNTRY IS 'Country';
COMMENT ON COLUMN UEB03A.TBDPOSTFILEZIP.POLGNAME IS 'Pol. municipality name';
COMMENT ON COLUMN UEB03A.TBDPOSTFILEZIP.CANTON IS 'Municipality canton';
COMMENT ON COLUMN UEB03A.TBDPOSTFILEZIP.LASTUPDATED IS 'timestamp of last change';



CREATE UNIQUE INDEX UEB03A.XPKTBDPOSTFILESTRE ON UEB03A.TBDPOSTFILESTREET
(
      TBDPOSTFILESTREETID            ASC
);

CREATE INDEX UEB03A.XIE1TBDPOSTFILESTR ON UEB03A.TBDPOSTFILESTREET
(
```

```
    ZIP6                    ASC
);

CREATE INDEX UEB03A.XIE2TBDPOSTFILESTR ON UEB03A.TBDPOSTFILESTREET
(
    STREET                  ASC,
    ZIP6                    ASC,
    STREETNR                ASC
);

ALTER TABLE UEB03A.TBDPOSTFILESTREET  ADD CONSTRAINT XPKTBDPOSTFILESTRE PRIMARY KEY (
        TBDPOSTFILESTREETID
);


CREATE INDEX UEB03A.XIE3TBDPOSTFILESTR ON UEB03A.TBDPOSTFILESTREET
(
    ZIP4                    ASC
);

CREATE INDEX UEB03A.XIE4TBDPOSTFILESTR ON UEB03A.TBDPOSTFILESTREET
(
    STREET                  ASC,
    ZIP4                    ASC,
    STREETNR                ASC
);

CREATE INDEX UEB03A.XIE5TBDPOSTFILESTR ON UEB03A.TBDPOSTFILESTREET
(
    BFSNR                   ASC
);


CREATE UNIQUE INDEX UEB03A.XPKTBDPOSTFILEZIP ON UEB03A.TBDPOSTFILEZIP
(
    TBDPOSTFILEZIPID            ASC
);


ALTER TABLE UEB03A.TBDPOSTFILEZIP  ADD CONSTRAINT XPKTBDPOSTFILEZIP PRIMARY KEY (
    TBDPOSTFILEZIPID
);

CREATE INDEX UEB03A.XIE1TBDPOSTFILEZIP ON UEB03A.TBDPOSTFILEZIP
(
    BFSNR                   ASC
);

CREATE INDEX UEB03A.XIE2TBDPOSTFILEZIP ON UEB03A.TBDPOSTFILEZIP
(
    ZIP4                    ASC
);
```

```
CREATE INDEX UEB03A.XIE3TBDPOSTFILEZIP ON UEB03A.TBDPOSTFILEZIP
(
      ZIP6                          ASC
);


ECHO ******************** END OF SCRIPT ********************;
```

Anpassungen:

```
-- Ganz Oben Hinzugefügt
CREATE TABLESPACE STBDPOSTFILESTREET
  PAGESIZE 4096
  MANAGED BY AUTOMATIC STORAGE
;

CREATE TABLESPACE STBDPOSTFILEZIP
  PAGESIZE 4096
  MANAGED BY AUTOMATIC STORAGE
;
```

UEB03-02.sql

Ausführungs Fehler:

```
COMMENT ON COLUMN UEB03B.TBDPOSTFILESTREET.ZIP4 IS '4 digits of ZIP' COMMENT ON COLUMN
UEB03B.TBDPOSTFILESTREET.ZIPADDON IS 'ZIP addon'
DB21034E  The command was processed as an SQL statement because it was not a
valid Command Line Processor command.  During SQL processing it returned:
SQL0104N  An unexpected token "COMMENT ON COLUMN UEB03B.TBDPOSTFILESTREET."
was found following "BEGIN-OF-STATEMENT".  Expected tokens may include:
"<space>".  SQLSTATE=42601
```

Zuvor:

```
-- ------------------------------------------------------------------------------------------------
-- Dies ist ein Kommentar, d.h. diese Zeile wird NICHT ausgefuehrt
-- ------------------------------------------------------------------------------------------------

-- disable STOP ON ERROR
UPDATE COMMAND OPTIONS USING S OFF ;

DROP TABLE UEB03B.TBDPOSTFILESTREET;
DROP TABLE UEB03B.TBDPOSTFILEZIP;

-- enable STOP ON ERROR again
UPDATE COMMAND OPTIONS USING S ON ;
```

```
CREATE TABLE UEB03B.TBDPOSTFILESTREET (
      TBDPOSTFILESTREETID  BIGINT NOT NULL GENERATED BY DEFAULT AS IDENTITY( START WITH 1, INCREMENT
      STREET               VARCHAR(120),
      ZIP4                 CHAR(4),
      ZIPADDON             CHAR(2),
      ZIP6                 CHAR(6),
      SHORTCITY            VARCHAR(54),
      COUNTRY              CHAR(2),
      ADDRSCANTON          CHAR(2),
      STREETNR             INTEGER,
      STREETNRADDON        VARCHAR(6),
      POLGNAME             VARCHAR(60),
      POLGCANTON           CHAR(2),
      BFSNR                INTEGER,
      LASTUPDATED          TIMESTAMP NOT NULL DEFAULT CURRENT TIMESTAMP,
   PRIMARY KEY (TBDPOSTFILESTREETID)
)
IN STBDPOSTFILESTREET
;


COMMENT ON COLUMN UEB03B.TBDPOSTFILESTREET.TBDPOSTFILESTREETID IS '''Primary Key''';
COMMENT ON COLUMN UEB03B.TBDPOSTFILESTREET.STREET IS 'Street name';
COMMENT ON COLUMN UEB03B.TBDPOSTFILESTREET.ZIP4 IS '4 digits of ZIP'
COMMENT ON COLUMN UEB03B.TBDPOSTFILESTREET.ZIPADDON IS 'ZIP addon';
COMMENT ON COLUMN UEB03B.TBDPOSTFILESTREET.ZIP6 IS 'Full ZIP (6 digits)'
COMMENT ON COLUMN UEB03B.TBDPOSTFILESTREET.SHORTCITY IS 'Short city name (27 characters)';
COMMENT ON COLUMN UEB03B.TBDPOSTFILESTREET.COUNTRY IS 'Country';
COMMENT ON COLUMN UEB03B.TBDPOSTFILESTREET.ADDRSCANTON IS 'Address canton';
COMMENT ON COLUMN UEB03B.TBDPOSTFILESTREET.STREETNR IS 'Street no w/o addon';
COMMENT ON COLUMN UEB03B.TBDPOSTFILESTREET.STREETNRADDON IS 'Addon (like 22A)';
COMMENT ON COLUMN UEB03B.TBDPOSTFILESTREET.POLGNAME IS 'Pol. municipality name';
COMMENT ON COLUMN UEB03B.TBDPOSTFILESTREET.POLGCANTON IS 'Municipality canton';
COMMENT ON COLUMN UEB03B.TBDPOSTFILESTREET.LASTUPDATED IS 'timestamp of last change';


CREATE TABLE UEB03B.TBDPOSTFILEZIP (
      TBDPOSTFILEZIPID     BIGINT NOT NULL GENERATED BY DEFAULT AS IDENTITY( START WITH 1, INCREMENT BY
      ZIP4                 CHAR(4),
      ZIPADDON             CHAR(2),
      ZIP6                 CHAR(6),
      CITY                 VARCHAR(60),
      SHORTCITY            VARCHAR(54),
      COUNTRY              CHAR(2)
      BFSNR                INTEGER,
      POLGNAME             VARCHAR(60)
      CANTON               CHAR(2),
      LASTUPDATED          TIMESTAMP NOT NULL DEFAULT CURRENT TIMESTAMP,
   PRIMARY KEY (TBDPOSTFILEZIPID)
)
IN STBDPOSTFILEZIP
```

```
;

COMMENT ON COLUMN UEB03B.TBDPOSTFILEZIP.TBDPOSTFILEZIPID IS '''Primary Key''';
COMMENT ON COLUMN UEB03B.TBDPOSTFILEZIP.ZIP4 IS '4 digits of ZIP';
COMMENT ON COLUMN UEB03B.TBDPOSTFILEZIP.ZIPADDON IS 'ZIP addon';
COMMENT ON COLUMN UEB03B.TBDPOSTFILEZIP.ZIP6 IS 'Full ZIP (6 digits)';
COMMENT ON COLUMN UEB03B.TBDPOSTFILEZIP.CITY IS 'City name';
COMMENT ON COLUMN UEB03B.TBDPOSTFILEZIP.SHORTCITY IS 'Short city name (27 characters)';
COMMENT ON COLUMN UEB03B.TBDPOSTFILEZIP.COUNTRY IS 'Country';
COMMENT ON COLUMN UEB03B.TBDPOSTFILEZIP.POLGNAME IS 'Pol. municipality name';
COMMENT ON COLUMN UEB03B.TBDPOSTFILEZIP.CANTON IS 'Municipality canton';
COMMENT ON COLUMN UEB03B.TBDPOSTFILEZIP.LASTUPDATED IS 'timestamp of last change';
□
□

CREATE UNIQUE INDEX UEB03B.XPKTBDPOSTFILESTRE ON  TBDPOSTFILESTREET
(
      TBDPOSTFILESTREETID            ASC
);

CREATE INDEX UEB03B.XIE1TBDPOSTFILESTR ON UEB03.TBDPOSTFILESTREET
(
      ZIP6                       ASC
);

CREATE INDEX UEB03B.XIE2TBDPOSTFILESTR ON UEB03B.TBDPOSTFILESTREET
(
      STREET                       ASC,
      ZIP6                       ASC,
      STREETNR                       ASC
);
□
CREATE INDEX UEB03B.XIE3TBDPOSTFILESTR ON UEB03B.TBDPOSTFILESTREET
(
      ZIP4                       ASC
);

CREATE INDEX UEB03B.XIE4TBDPOSTFILESTR ON UEB03B.TBDPOSTFILESTREET
(
      STREET                       ASC,
      ZIP4                       ASC,
      STREETNR                       ASC
);

CREATE INDEX UEB03B.XIE5TBDPOSTFILESTR ON UEB03B.TBDPOSTFILESTREET
(
      BFSNR                        ASC
);

□□□
CREATE UNIQUE INDEX UEB03B.XPKTBDPOSTFILEZIP ON UEB03B.TBDPOSTFILEZIP
(
      TBDPOSTFILEZIPID               ASC
```

```
);


CREATE INDEX UEB03B.XIE1TBDPOSTFILEZIP ON UEB03B.TBDPOSTFILEZIP
(
      BFSNR                         ASC
);

CREATE INDEX UEB03B.XIE2TBDPOSTFILEZIP ON UEB03B.TBDPOSTFILEZIP
(
      ZIP4                          ASC
);

CREATE INDEX UEB03B.XIE3TBDPOSTFILEZIP ON UEB03B.TBDPOSTFILEZIP
(
      ZIP6                          ASC
);


ECHO ******************** END OF SCRIPT *******************************;
```

Anpassungen:

```
-- Vorher
DROP TABLE UEB03B.TBDPOSTFILESTREET;
DROP TABLE UEB03B.TBDPOSTFILEZIP;

-- Nachher
DROP TABLE IF EXISTS UEB03B.TBDPOSTFILESTREET;
DROP TABLE IF EXISTS UEB03B.TBDPOSTFILEZIP;

CREATE TABLESPACE STBDPOSTFILESTREET
  PAGESIZE 4096
  MANAGED BY AUTOMATIC STORAGE
;

CREATE TABLESPACE STBDPOSTFILEZIP
  PAGESIZE 4096
  MANAGED BY AUTOMATIC STORAGE
;
```

```
-- Vorher
COMMENT ON COLUMN UEB03B.TBDPOSTFILESTREET.ZIP4 IS '4 digits of ZIP'
COMMENT ON COLUMN UEB03B.TBDPOSTFILESTREET.ZIPADDON IS 'ZIP addon';
COMMENT ON COLUMN UEB03B.TBDPOSTFILESTREET.ZIP6 IS 'Full ZIP (6 digits)'

-- Nachher
COMMENT ON COLUMN UEB03B.TBDPOSTFILESTREET.ZIP4 IS '4 digits of ZIP';
COMMENT ON COLUMN UEB03B.TBDPOSTFILESTREET.ZIPADDON IS 'ZIP addon';
COMMENT ON COLUMN UEB03B.TBDPOSTFILESTREET.ZIP6 IS 'Full ZIP (6 digits)';
```

```
-- Vorher
      COUNTRY            CHAR(2)
      BFSNR              INTEGER,
      POLGNAME           VARCHAR(60)

-- Nachher
      COUNTRY            CHAR(2),
      BFSNR              INTEGER,
      POLGNAME           VARCHAR(60),
```

```
-- Vorher
CREATE UNIQUE INDEX UEB03B.XPKTBDPOSTFILESTRE ON  TBDPOSTFILESTREET
(
      TBDPOSTFILESTREETID          ASC
);

CREATE INDEX UEB03B.XIE1TBDPOSTFILESTR ON UEB03.TBDPOSTFILESTREET
(
      ZIP6                         ASC
);

-- Nachher
CREATE UNIQUE INDEX UEB03B.XPKTBDPOSTFILESTRE ON UEB03B.TBDPOSTFILESTREET
(
      TBDPOSTFILESTREETID          ASC
);

CREATE INDEX UEB03B.XIE1TBDPOSTFILESTR ON UEB03B.TBDPOSTFILESTREET
(
      ZIP6                         ASC
);
```

UEB03-03.sql

Ausführungs Fehler:

```
CREATE TABLE UEB03C.TBDPOSTFILEZIP ( TBDPOSTFILEZIPID     BIGINT NOT NULL GENERATED BY DEFAULT AS IDENTITY( START WITH 1, INCREMENT BY 1, CACHE 100000), ZIP4                CHAR(4), ZIPADDON CHAR(2), ZIP6                CHAR(6), CITY               VARCHAR(60), SHORTCITY          VARCHAR(54), COUNTRY            CHAR(2), BFSNR              INTEGER, POLGNAME           VARCHAR, CANTON             CHAR(2), LASTUPDATED         TIMESTAMP NOT NULL DEFAULT CURRENT TIMESTAMP ) IN STBDPOSTFILEZIP
DB21034E  The command was processed as an SQL statement because it was not a
valid Command Line Processor command.  During SQL processing it returned:
SQL0604N  The length, precision, or scale attribute for column, distinct type,
structured type, array type, attribute of structured type, routine, cast
```

target type, type mapping, or global variable "VARCHAR" is not valid.

SQLSTATE=42611

Zuvor:

```
-- ------------------------------------------------------------------------------------------------
-- Dies ist ein Kommentar, d.h. diese Zeile wird NICHT ausgefuehrt
-- ------------------------------------------------------------------------------------------------

-- mit dieser Anweisung wird der AUTO COMMIT (Schreiben in die Datenbank ausgeschaltet)
-- alle Veränderungen sind nur innerhalb der aktuellen Session gültig
UPDATE COMMAND OPTIONS USING C OFF;

--DROP TABLE UEB03C.TBDPOSTFILESTREET;
--DROP TABLE UEB03C.TBDPOSTFILEZIP;

CREATE TABLE UEB03C.TBDPOSTFILESTREET (
      TBDPOSTFILESTREETID  BIGINT NOT NULL GENERATED BY DEFAULT AS IDENTITY( START WITH 1, INCREMENT
      STREET               VARCHAR(120),
      ZIP4                 CHAR(4),
      ZIPADDON             CHAR(2),
      ZIP6                 CHAR(6),
      SHORTCITY            VARCHAR(54),
      COUNTRY              CHAR(2),
      ADDRSCANTON          CHAR(2),
      STREETNR             INTEGER,
      STREETNRADDON        VARCHAR(6),
      POLGNAME             VARCHAR(60),
      POLGCANTON           CHAR(2),
      BFSNR                INTEGER,
      LASTUPDATED          TIMESTAMP NOT NULL DEFAULT CURRENT TIMESTAMP
)
IN STBDPOSTFILESTREET
;


COMMENT ON COLUMN UEB03C.TBDPOSTFILESTREET.TBDPOSTFILESTREETID IS '''Primary Key''';
COMMENT ON COLUMN UEB03C.TBDPOSTFILESTREET.STREET IS 'Street name';
COMMENT ON COLUMN UEB03C.TBDPOSTFILESTREET.ZIP4 IS '4 digits of ZIP';
COMMENT ON COLUMN UEB03C.TBDPOSTFILESTREET.ZIPADDON IS 'ZIP addon';
COMMENT ON COLUMN UEB03C.TBDPOSTFILESTREET.ZIP6 IS 'Full ZIP (6 digits)';
COMMENT ON COLUMN UEB03C.TBDPOSTFILESTREET.SHORTCITY IS 'Short city name (27 characters)';
COMMENT ON COLUMN UEB03C.TBDPOSTFILESTREET.COUNTRY IS 'Country';
COMMENT ON COLUMN UEB03C.TBDPOSTFILESTREET.ADDRSCANTON IS 'Address canton';
COMMENT ON COLUMN UEB03C.TBDPOSTFILESTREET.STREETNR IS 'Street no w/o addon';
COMMENT ON COLUMN UEB03C.TBDPOSTFILESTREET.STREETNRADDON IS 'Addon (like 22A)';
COMMENT ON COLUMN UEB03C.TBDPOSTFILESTREET.POLGNAME IS 'Pol. municipality name';
COMMENT ON COLUMN UEB03C.TBDPOSTFILESTREET.POLGCANTON IS 'Municipality canton';
COMMENT ON COLUMN UEB03C.TBDPOSTFILESTREET.LASTUPDATED IS 'timestamp of last change';


CREATE TABLE UEB03C.TBDPOSTFILEZIP (
```

```
    TBDPOSTFILEZIPID     BIGINT NOT NULL GENERATED BY DEFAULT AS IDENTITY( START WITH 1, INCREMENT BY
    ZIP4                 CHAR(4),
    ZIPADDON             CHAR(2),
    ZIP6                 CHAR(6),
    CITY                 VARCHAR(60),
    SHORTCITY            VARCHAR(54),
    COUNTRY              CHAR(2),
    BFSNR                INTEGER,
    POLGNAME             VARCHAR,
    CANTON               CHAR(2),
    LASTUPDATED          TIMESTAMP NOT NULL DEFAULT CURRENT TIMESTAMP
)
IN STBDPOSTFILEZIP
;

COMMENT ON COLUMN UEB03C.TBDPOSTFILEZIP.TBDPOSTFILEZIPID IS '''Primary Key''';
COMMENT ON COLUMN UEB03C.TBDPOSTFILEZIP.ZIP4 IS '4 digits of ZIP';
COMMENT ON COLUMN UEB03C.TBDPOSTFILEZIP.ZIPADDON IS 'ZIP addon';
COMMENT ON COLUMN UEB03C.TBDPOSTFILEZIP.ZIP6 IS 'Full ZIP (6 digits)';
COMMENT ON COLUMN UEB03C.TBDPOSTFILEZIP.CITY IS 'City name'
COMMENT ON COLUMN UEB03C.TBDPOSTFILEZIP.SHORTCITY IS 'Short city name (27 characters)';
COMMENT ON COLUMN UEB03C.TBDPOSTFILEZIP.COUNTRY IS 'Country';
COMMENT ON COLUMN UEB03C.TBDPOSTFILEZIP.POLGNAME IS 'Pol. municipality name';
COMMENT ON COLUMN UEB03C.TBDPOSTFILEZIP.CANTON IS 'Municipality canton';
COMMENT ON COLUMN UEB03C.TBDPOSTFILEZIP.LASTUPDATED IS 'timestamp of last change';
□
□

CREATE UNIQUE INDEX UEB03C.XPKTBDPOSTFILESTRE ON UEB03C.TBDPOSTFILESTREET
(
    TBDPOSTFILESTREETID          ASC
);

CREATE INDEX UEB03C.XIE1TBDPOSTFILESTR ON UEB03C.TBDPOSTFILESTREET
(
    ZIP6                     ASC
);

CREATE INDEX UEB03C.XIE2TBDPOSTFILESTR ON UEB03C.TBDPOSTFILESTREET
(
    STREET                     ASC,
    ZIP6                     ASC,
    STREETNR                     ASC
);

ALTER TABLE UEB03C.TBDPOSTFILESTREET  ADD CONSTRAINT XPKTBDPOSTFILESTRE PRIMARY KEY (
          TBDPOSTFILESTREETID
);
□
□
CREATE INDEX UEB03C.XIE3TBDPOSTFILESTR ON UEB02.TBDPOSTFILESTREET
(
    ZIP4                     ASC
```

```
);

CREATE INDEX UEB03C.XIE4TBDPOSTFILESTR ON UEB03C.TBDPOSTFILESTREET
(
      STREET                        ASC,
      ZIP4                          ASC,
      STREETNR                      ASC
);

CREATE INDEX UEB03C.XIE5TBDPOSTFILESTR ON UEB03C.TBDPOSTFILESTREET
(
      BFSNR                         ASC
);

□□□
CREATE UNIQUE INDEX UEB03C.XPKTBDPOSTFILEZIP ON UEB03C.TBDPOSTFILEZIP
(
      TBDPOSTFILEZIPID              ASC
);


ALTER TABLE UEB03C.TBDPOSTFILEZIP  ADD CONSTRAINT XPKTBDPOSTFILEZIP PRIMARY KEY (
     TBDPOSTFILEZIPID
);
□
CREATE INDEX UEB03C.XIE1TBDPOSTFILEZIP ON UEB03C.TBDPOSTFILEZIP
(
      BFSNR                         ASC
);
□
CREATE INDEX UEB03C.XIE2TBDPOSTFILEZIP ON UEB03C.TBDPOSTFILEZIP
(
      ZIP4                          ASC
);

CREATE INDEX UEB03C.XIE3TBDPOSTFILEZIP ON UEB03C.TBDPOSTFILEZIP
(
      ZIP6                          ASC
);



ECHO ******************** END OF SCRIPT ********************;
```

Anpassungen:

```
-- Vorher
--DROP TABLE UEB03C.TBDPOSTFILESTREET;
--DROP TABLE UEB03C.TBDPOSTFILEZIP;

-- Nachher
DROP TABLE IF EXISTS UEB03B.TBDPOSTFILESTREET;
DROP TABLE IF EXISTS UEB03B.TBDPOSTFILEZIP;
```

```
CREATE TABLESPACE STBDPOSTFILESTREET
  PAGESIZE 4096
  MANAGED BY AUTOMATIC STORAGE
;

CREATE TABLESPACE STBDPOSTFILEZIP
  PAGESIZE 4096
  MANAGED BY AUTOMATIC STORAGE
;
```

```
-- Vorher
CREATE TABLE UEB03C.TBDPOSTFILEZIP (
      TBDPOSTFILEZIPID     BIGINT NOT NULL GENERATED BY DEFAULT AS IDENTITY( START WITH 1, INCREMENT BY
      ZIP4              CHAR(4),
      ZIPADDON             CHAR(2),
      ZIP6              CHAR(6),
      CITY              VARCHAR(60),
      SHORTCITY            VARCHAR(54),
      COUNTRY              CHAR(2),
      BFSNR             INTEGER,
      POLGNAME             VARCHAR,
      CANTON            CHAR(2),
      LASTUPDATED          TIMESTAMP NOT NULL DEFAULT CURRENT TIMESTAMP
)
IN STBDPOSTFILEZIP
;

-- Nachher
CREATE TABLE UEB03C.TBDPOSTFILEZIP (
      TBDPOSTFILEZIPID     BIGINT NOT NULL GENERATED BY DEFAULT AS IDENTITY( START WITH 1, INCREMENT BY
      ZIP4              CHAR(4),
      ZIPADDON             CHAR(2),
      ZIP6              CHAR(6),
      CITY              VARCHAR(60),
      SHORTCITY            VARCHAR(54),
      COUNTRY              CHAR(2),
      BFSNR             INTEGER,
      POLGNAME             VARCHAR(60),
      CANTON            CHAR(2),
      LASTUPDATED          TIMESTAMP NOT NULL DEFAULT CURRENT TIMESTAMP
)
IN STBDPOSTFILEZIP
;
```

```
-- Vorher
COMMENT ON COLUMN UEB03C.TBDPOSTFILEZIP.CITY IS 'City name'

-- Nachher
COMMENT ON COLUMN UEB03C.TBDPOSTFILEZIP.CITY IS 'City name';
```

cleanup

```
-- Vorher
CREATE INDEX UEB03C.XIE3TBDPOSTFILESTR ON UEB02.TBDPOSTFILESTREET
(
      ZIP4                          ASC
);


-- Nachher
CREATE INDEX UEB03C.XIE3TBDPOSTFILESTR ON UEB03C.TBDPOSTFILESTREET
(
      ZIP4                          ASC
);
```

UEB03-04.sql

Ausführungs Fehler:

```
CREATE TABLE  "ARTIKEL" ( "ARTIKELID"        INTEGER NOT NULL , "TITEL"           VARCHAR(255) ,
"JAHR"            SMALLINT , "AUSTITEL"        VARCHAR(255) "SEITE_VON"        SMALLINT , "SEITE_BIS"
SMALLINT , "BUCHID"           INTEGER , "URL"             VARCHAR 200 , "CR"              VARCHAR(80) ,
"BEMERKUNG"        VARCHAR(100) ) IN "MYTABLESPACE1"
DB21034E  The command was processed as an SQL statement because it was not a
valid Command Line Processor command.  During SQL processing it returned:
SQL0104N  An unexpected token "SMALLINT" was found following "HAR(255)
"SEITE_VON"".  Expected tokens may include:  "<references_spec>".
SQLSTATE=42601
```

Zuvor:

```
------------------------------------------------------------------------
-- Set autocommit off
------------------------------------------------------------------------
UPDATE COMMAND OPTIONS USING C OFF;

------------------------------------------------------------------------
-- Set Schema UEB03D, Set
--   - setzt Default-Schema, falls in Objekt-Referent nicht angegeben
------------------------------------------------------------------------
SET SCHEMA UEB03D;

------------------------------------------------------------------------
-- Stop execution on command error OFF
------------------------------------------------------------------------
UPDATE COMMAND OPTIONS USING S OFF;
```

```
-------------------------------------------------------------------------
-- DROP objects in schema UEB03D
-------------------------------------------------------------------------
DROP TABLE "ARTIKEL" ;
DROP TABLE "ART_SW" ;
DROP TABLE "ART_AUT" ;
DROP TABLE "AUTOR" ;
DROP TABLE "BUCH_AUT" ;
DROP TABLE "BUCH_SW" ;
DROP TABLE "BUCHTITEL" ;
DROP TABLE "DISS_AUT" ;
DROP TABLE "DISS_SW" ;
DROP TABLE "DISSERTATION" ;
DROP TABLE "INSTITUTION" ;
DROP TABLE "REPORT" ;
DROP TABLE "REPORT_AUT" ;
DROP TABLE "REPORT_SW" ;
DROP TABLE "SCHLAGWORT" ;
DROP TABLE "VERLAG" ;

-------------------------------------------------------------------------
-- Stop execution on command error ON
-------------------------------------------------------------------------
UPDATE COMMAND OPTIONS USING S ON;

-------------------------------------------------------------------------
-- CREATE tables in schema UEB03D
-------------------------------------------------------------------------

CREATE TABLE  "ARTIKEL"
(
       "ARTIKELID"        INTEGER NOT NULL ,
□    "TITEL"            VARCHAR(255) ,
       "JAHR"             SMALLINT ,
       "AUSTITEL"         VARCHAR(255)
       "SEITE_VON"         SMALLINT ,
       "SEITE_BIS"        SMALLINT ,
       "BUCHID"           INTEGER ,
       "URL"              VARCHAR 200 ,
       "CR"               VARCHAR(80) ,
       "BEMERKUNG"        VARCHAR(100)
)
IN "MYTABLESPACE1" ;

CREATE TABLE  "ART_SW"
(
      "ARTIKELID"         INTEGER NOT NULL ,
      "SWID"              INTEGER NOT NULL
)
IN "MYTABLESPACE1"

CREATE TABLE  "ART_AUT"
(
```

```
      "ARTIKELID"        INTEGER NOT NULL ,
      "AUTORID"          INTEGER NOT NULL ,
      "ROLLE"            VARCHAR(1) ,
      "RANG"             SMALLINT
)
IN "MYTABLESPACE1" ;

CREATE TABLE  "AUTOR"
(
      "AUTORID"          INTEGER NOT NULL ,
      "NAME"             VARCHAR(30) ,
      "ZUSATZ"           VARCHAR(10) ,
      "INITIALEN"        VARCHAR(6) ,
      "VORNAMEN"         VARCHAR(30) ,
      "INSTITUTION       VARCHAR(50) ,
      URL"              VARCHAR(200) ,
      "BEMERKUNG"        VARCHAR(10000) ,
      "PSEUDONYM"        INTEGER
)
IN "MYTABLESPACE1" ;

CREATE TABLE  "BUCH_AUT"
(
      "BUCHID"           INTEGER NOT NULL ,
      "AUTORID"          INTEGER NOT NULL ,
      "ROLLE"            VARCHAR(1) ,
      "RANG"             SMALLINT
)
IN "MYTABLESPACE1" ;

CREATE TABLE  "BUCH_SW"
(
       "BUCHID"          INTEGER NOT NULL ,
       "SWID"            INTEGER NOT NULL
)
IN "MYTABLESPACE1"

CREATE TABLE  "BUCHTITEL"
(
       "BUCHID"          INTEGER NOT NULL ,
       "GESAMTID"         INTEGER ,
       "BANDNR"          SMALLINT ,
       "HSTITEL"         VARCHAR(255) ,
       "ESTITEL"         VARCHAR(100) ,
       "ISBN"            VARCHAR(15) ,
       "ISBN2"           VARCHAR  ,
       "AUFLAGE"         VARCHAR(70) ,
       "JAHR"            SMALLINT ,
       "MEDIUM"          VARCHAR(20) ,
       "UMFANG"          VARCHAR(50) ,
       "PREIS"           VARCHAR(20) ,
       "BEZUGINFO"       VARCHAR(30) ,
       "VERF_IN_VORLFORM" VARCHAR(130) ,
```

```
	"REIHE1"	VARCHAR(150) ,
	"REIHE2"	VARCHAR(150) ,
	"KSCHAFT1"	VARCHAR(100) ,
	"KSCHAFT2"	VARCHAR(100) ,
	"HSV"	VARCHAR(50) ,
	"SIGNATUR"	VARCHAR(30) ,
	"ZUGANGSNR"	VARCHAR(30) ,
	"BESTAND"	VARCHAR(50) ,
	"WEITEXEMP"	VARCHAR(30) ,
	"ERFDATUM"	VARCHAR(17) ,
	"AENDATUM"	VARCHAR(17) ,
	"VERLAGSID"	INTEGER ,
	"URL"	VARCHAR(200) ,
	"CR"	VARCHAR(80) ,
	"BEMERKUNG"	VARCHAR(100) ,
	"BAENDE"	SMALLINT
)
IN "MYTABLESPACE1" ;

CREATE TABLE  "DISS_AUT"
(
	"DISSID"	INTEGER NOT NULL
	"AUTORID"	INTEGER NOT NULL ,
	"ROLLE"	VARCHAR(1) ,
	"RANG"	SMALLINT
)
IN "MYTABLESPACE1" ;

CREATE TABLE  "DISS_SW"
(
	"DISSID"	INTEGER NOT NULL ,
	"SWID"	INTEGER NOT NULL
)
IN "MYTABLESPACE1" ;

CREATE TABLE  "DISSERTATION"
(
	"DISSID"	INTEGER NOT NULL ,
	"TITEL"	VARCHAR(255) ,
	"INSTID"	INTEGER ,
	"ORT"	VARCHAR(80) ,
	"JAHR"	SMALLINT ,
	"ISBN"	VARCHAR(15) ,
	"TYP"	VARCHAR(1) ,
	"UMFANG"	VARCHAR(50)
	"SIGNATUR"	VARCHAR(30) ,
	"ZUGANGSNR"	VARCHAR(50) ,
	"BESTAND"	VARCHAR(30) ,
	"URL"	VARCHAR(200) ,
	"CR"	VARCHAR(80) ,
	"BEMERKUNG"	VARCHAR(100)
)
IN "MYTABLESPACE1" ;
```

```
CREATE TABLE  "INSTITUTION"
(
      "INSTID"           INTEGER NOT NULL ,
      "NAME"             VARCHAR(80) ,
      "ORT"              VARCHAR(80) ,
      "ADRESSE"          VARCHAR(80) ,
      "TEL"              VARCHAR(20) ,
      "URL"              VARCHAR(200 ,
      "BEMERKUNG"        VARCHAR(100) ,
      "FAX"              VARCHAR(20)
)
IN "MYTABLESPACE1" ;

CREATE TABLE  "REPORT"
(
      "REPORTID"         INTEGER NOT NULL ,
      "TITEL" □□□   VARCHAR(255) ,
      "INSTID" □□   INTEGER ,
      "JAHR" □□□   SMALLINT ,
      "LFDNR" □□□   SMALLINT ,
      "URL" □□□   VARCHAR(200) ,
      "CR" □□□   VARCHAR(80) ,
      "UMFANG" □□   VARCHAR(50) ,
      "BEMERKUNG"        VARCHAR(100) ,
      "REIHE" □□□   VARCHAR(150) ,
      "SIGNATUR"         VARCHAR(30) ,
)
IN "MYTABLESPACE1" ;

 CREATE TABLE  "REPORT_AUT"
 (
       "REPORTID"        INTEGER NOT NULL ,
       "AUTORID"         INTEGER NOT NULL ,
       "RANG"            SMALLINT
)
IN "MYTABLESPACE1" ;

CREATE TABLE  "SCHLAGWORT"
(
      "SWID"             INTEGER NOT NULL ,
      "SCHLAGWORT"       VARCHAR(50)
)
IN "MYTABLESPACE1" ;

CREATE TABLE  "REPORT_SW"
(
      "REPORTID"         INTEGER NOT NULL ,
      "SWID"             INTEGER NOT NULL
)
IN "MYTABLESPACE1" ;

CREATE TABLE  "VERLAG"
```

```
(
      "VERLAGSID"        INTEGER NOT NULL ,
      "NAME"             VARCHAR(80) ,
      "ORT"              VARCHAR(80) ,
      "ISBNKZ"           VARCHAR(10) ,
      "ADRESSE"          VARCHAR 80 ,
      "TEL"              VARCHAR(20) ,
      "URL"              VARCHAR(200) ,
      "BEMERKUNG"        VARCHAR(100) ,
      "FAX"              VARCHAR(20)
)
IN "MYTABLESPACE1" ;

------------------------------------------------------------------------
-- Add constraint for primary keys
------------------------------------------------------------------------
ALTER TABLE  "ARTIKEL"      ADD CONSTRAINT "PK_ARTIKEL"      PRIMARY KEY ("ARTIKELID");
ALTER TABLE  "ART_SW"       ADD CONSTRAINT "PK_ART_SW"       PRIMARY KEY ("ARTIKELID", "SWID");
ALTER TABLE  "ART_AUT"      ADD CONSTRAINT "PK_ART_AUT"      PRIMARY KEY ("ARTIKELID", "AUTORID");
ALTER TABLE  "AUTOR"        ADD CONSTRAINT "PK_AUTOR"        PRIMARY KEY ("AUTOR");
ALTER TABLE  "BUCH_AUT"     ADD CONSTRAINT "PK_BUCH_AUT"     PRIMARY KEY ("BUCHID", "AUTORID");
ALTER TABLE  "BUCH_SW"      ADD CONSTRAINT "PK_BUCH_SW"      PRIMARY KEY ("BUCHID", "SWID");
ALTER TABLE  "BUCHTITEL"    ADD CONSTRAINT "PK_BUCHTITEL"    PRIMARY KEY ("BUCHID");
ALTER TABLE  "DISS_AUT"     ADD CONSTRAINT "PK_DISS_AUT"     PRIMARY KEY ("DISSID", "AUTORID");
ALTER TABLE  "DISS_SW"      ADD CONSTRAINT "PK_DISS_SW"      PRIMARY KEY ("DISSID", "SWID");
ALTER TABLE  "DISSERTATION" ADD CONSTRAINT "PK_DISSERTATION" PRIMARY KEY ("DISSID");
ALTER TABLE  "INSTITUTION"  ADD CONSTRAINT "PK_INSTITUTION"  PRIMARY KEY ("INSTID");
ALTER TABLE  "REPORT"       ADD CONSTRAINT "PK_REPORT"       PRIMARY KEY ("REPORTID");
ALTER TABLE  "REPORT_AUT"   ADD CONSTRAINT "PK_REPORT_AUT"   PRIMARY KEY ("REPORTID", "AUTORID");
ALTER TABLE  "REPORT_SW"    ADD CONSTRAINT "PK_REPORT_SW"    PRIMARY KEY ("REPORTID", "SWID");
ALTER TABLE  "SCHLAGWORT"   ADD CONSTRAINT "PK_SCHLAGWORT"   PRIMARY KEY ("SWID");
ALTER TABLE  "VERLAG"       ADD CONSTRAINT "PK_VERLAG"       PRIMARY KEY ("VERLAGSID");

------------------------------------------------------------------------
-- Add additional indexes
------------------------------------------------------------------------
CREATE INDEX  "I_ARTIKEL_BUCHID" ON  "TARTIKEL"  ("BUCHID" ASC) ALLOW REVERSE SCANS;

CREATE INDEX  "I_AUTOREN_NAME" ON  "AUTOR"  ("NACHNAME" ASC  "VORNAMEN" ASC)
    PCTFREE REVERSE SCANS;


------------------------------------------------------------------------
-- Commit work
------------------------------------------------------------------------
COMMIT WORK;


ECHO ******************** END OF SCRIPT ********************;
```

Anpassungen:

```
-----------------------------------------------------------------------
-- Set autocommit off
-----------------------------------------------------------------------
UPDATE COMMAND OPTIONS USING C OFF;

-----------------------------------------------------------------------
-- Set Schema UEB03D, Set
--   - setzt Default-Schema, falls in Objekt-Referent nicht angegeben
-----------------------------------------------------------------------
SET SCHEMA UEB03D;

-----------------------------------------------------------------------
-- Stop execution on command error OFF
-----------------------------------------------------------------------
UPDATE COMMAND OPTIONS USING S OFF;

-- Ganz Oben Hinzugefügt
CREATE TABLESPACE STBDPOSTFILESTREET
  PAGESIZE 4096
  MANAGED BY AUTOMATIC STORAGE
;

-----------------------------------------------------------------------
-- DROP objects in schema UEB03D
-----------------------------------------------------------------------
DROP TABLE "ARTIKEL" ;
DROP TABLE "ART_SW" ;
DROP TABLE "ART_AUT" ;
DROP TABLE "AUTOR" ;
DROP TABLE "BUCH_AUT" ;
DROP TABLE "BUCH_SW" ;
DROP TABLE "BUCHTITEL" ;
DROP TABLE "DISS_AUT" ;
DROP TABLE "DISS_SW" ;
DROP TABLE "DISSERTATION" ;
DROP TABLE "INSTITUTION" ;
DROP TABLE "REPORT" ;
DROP TABLE "REPORT_AUT" ;
DROP TABLE "REPORT_SW" ;
DROP TABLE "SCHLAGWORT" ;
DROP TABLE "VERLAG" ;

-----------------------------------------------------------------------
-- Stop execution on command error ON
-----------------------------------------------------------------------
UPDATE COMMAND OPTIONS USING S ON;

-----------------------------------------------------------------------
-- CREATE tables in schema UEB03D
-----------------------------------------------------------------------

CREATE TABLE  "ARTIKEL"
(
```

```
        "ARTIKELID"        INTEGER NOT NULL ,
        "TITEL"            VARCHAR(255) ,
        "JAHR"             SMALLINT ,
        "AUSTITEL"         VARCHAR(255) ,
        "SEITE_VON"        SMALLINT ,
        "SEITE_BIS"        SMALLINT ,
        "BUCHID"           INTEGER ,
        "URL"              VARCHAR(200) ,
        "CR"               VARCHAR(80) ,
        "BEMERKUNG"        VARCHAR(100)
)
IN "STBDPOSTFILESTREET" ;

CREATE TABLE  "ART_SW"
(
      "ARTIKELID"        INTEGER NOT NULL ,
      "SWID"             INTEGER NOT NULL
)
IN "STBDPOSTFILESTREET" ;

CREATE TABLE  "ART_AUT"
(
      "ARTIKELID"        INTEGER NOT NULL ,
      "AUTORID"          INTEGER NOT NULL ,
      "ROLLE"            VARCHAR(1) ,
      "RANG"             SMALLINT
)
IN "STBDPOSTFILESTREET" ;

CREATE TABLE  "AUTOR"
(
      "AUTORID"          INTEGER NOT NULL ,
      "NAME"             VARCHAR(30) ,
      "ZUSATZ"           VARCHAR(10) ,
      "INITIALEN"        VARCHAR(6) ,
      "VORNAMEN"         VARCHAR(30) ,
      "INSTITUTION"      VARCHAR(50) ,
      "URL"              VARCHAR(200) ,
      "BEMERKUNG"        VARCHAR(10000) ,
      "PSEUDONYM"        INTEGER
)
IN "STBDPOSTFILESTREET" ;

CREATE TABLE  "BUCH_AUT"
(
      "BUCHID"           INTEGER NOT NULL ,
      "AUTORID"          INTEGER NOT NULL ,
      "ROLLE"            VARCHAR(1) ,
      "RANG"             SMALLINT
)
IN "STBDPOSTFILESTREET" ;

CREATE TABLE  "BUCH_SW"
```

```
(
       "BUCHID"          INTEGER NOT NULL ,
       "SWID"            INTEGER NOT NULL
)
IN "STBDPOSTFILESTREET" ;

CREATE TABLE  "BUCHTITEL"
(
       "BUCHID"          INTEGER NOT NULL ,
       "GESAMTID"         INTEGER ,
       "BANDNR"          SMALLINT ,
       "HSTITEL"         VARCHAR(255) ,
       "ESTITEL"         VARCHAR(100) ,
       "ISBN"            VARCHAR(15) ,
       "ISBN2"           VARCHAR(15)  ,
       "AUFLAGE"         VARCHAR(70) ,
       "JAHR"            SMALLINT ,
       "MEDIUM"          VARCHAR(20) ,
       "UMFANG"          VARCHAR(50) ,
       "PREIS"           VARCHAR(20) ,
       "BEZUGINFO"        VARCHAR(30) ,
       "VERF_IN_VORLFORM" VARCHAR(130) ,
       "REIHE1"          VARCHAR(150) ,
       "REIHE2"          VARCHAR(150) ,
       "KSCHAFT1"         VARCHAR(100) ,
       "KSCHAFT2"         VARCHAR(100) ,
       "HSV"             VARCHAR(50) ,
       "SIGNATUR"         VARCHAR(30) ,
       "ZUGANGSNR"        VARCHAR(30) ,
       "BESTAND"          VARCHAR(50) ,
       "WEITEXEMP"        VARCHAR(30) ,
       "ERFDATUM"         VARCHAR(17) ,
       "AENDATUM"         VARCHAR(17) ,
       "VERLAGSID"        INTEGER ,
       "URL"             VARCHAR(200) ,
       "CR"              VARCHAR(80) ,
       "BEMERKUNG"        VARCHAR(100) ,
       "BAENDE"          SMALLINT
)
IN "STBDPOSTFILESTREET" ;

CREATE TABLE  "DISS_AUT"
(
       "DISSID"          INTEGER NOT NULL ,
       "AUTORID"          INTEGER NOT NULL ,
       "ROLLE"           VARCHAR(1) ,
       "RANG"            SMALLINT
)
IN "STBDPOSTFILESTREET" ;

CREATE TABLE  "DISS_SW"
(
       "DISSID"          INTEGER NOT NULL ,
```

```
      "SWID"            INTEGER NOT NULL
)
IN "STBDPOSTFILESTREET" ;

CREATE TABLE  "DISSERTATION"
(
      "DISSID"          INTEGER NOT NULL ,
      "TITEL"           VARCHAR(255) ,
      "INSTID"          INTEGER ,
      "ORT"             VARCHAR(80) ,
      "JAHR"            SMALLINT ,
      "ISBN"            VARCHAR(15) ,
      "TYP"             VARCHAR(1) ,
      "UMFANG"          VARCHAR(50) ,
      "SIGNATUR"        VARCHAR(30) ,
      "ZUGANGSNR"       VARCHAR(50) ,
      "BESTAND"         VARCHAR(30) ,
      "URL"             VARCHAR(200) ,
      "CR"              VARCHAR(80) ,
      "BEMERKUNG"       VARCHAR(100)
)
IN "STBDPOSTFILESTREET" ;

CREATE TABLE  "INSTITUTION"
(
      "INSTID"          INTEGER NOT NULL ,
      "NAME"            VARCHAR(80) ,
      "ORT"             VARCHAR(80) ,
      "ADRESSE"         VARCHAR(80) ,
      "TEL"             VARCHAR(20) ,
      "URL"             VARCHAR(200) ,
      "BEMERKUNG"       VARCHAR(100) ,
      "FAX"             VARCHAR(20)
)
IN "STBDPOSTFILESTREET" ;

CREATE TABLE  "REPORT"
(
      "REPORTID"        INTEGER NOT NULL ,
      "TITEL" □□□  VARCHAR(255) ,
      "INSTID" □□  INTEGER ,
      "JAHR" □□□  SMALLINT ,
      "LFDNR" □□□  SMALLINT ,
      "URL" □□□  VARCHAR(200) ,
      "CR" □□□  VARCHAR(80) ,
      "UMFANG" □□  VARCHAR(50) ,
      "BEMERKUNG"       VARCHAR(100) ,
      "REIHE" □□□  VARCHAR(150) ,
      "SIGNATUR"        VARCHAR(30)
)
IN "STBDPOSTFILESTREET" ;

 CREATE TABLE  "REPORT_AUT"
```

```
 (
       "REPORTID"         INTEGER NOT NULL ,
       "AUTORID"          INTEGER NOT NULL ,
       "RANG"             SMALLINT
)
IN "STBDPOSTFILESTREET" ;

CREATE TABLE  "SCHLAGWORT"
(
      "SWID"             INTEGER NOT NULL ,
      "SCHLAGWORT"       VARCHAR(50)
)
IN "STBDPOSTFILESTREET" ;

CREATE TABLE  "REPORT_SW"
(
      "REPORTID"         INTEGER NOT NULL ,
      "SWID"             INTEGER NOT NULL
)
IN "STBDPOSTFILESTREET" ;

CREATE TABLE  "VERLAG"
(
      "VERLAGSID"        INTEGER NOT NULL ,
      "NAME"             VARCHAR(80) ,
      "ORT"              VARCHAR(80) ,
      "ISBNKZ"           VARCHAR(10) ,
      "ADRESSE"          VARCHAR (80) ,
      "TEL"              VARCHAR(20) ,
      "URL"              VARCHAR(200) ,
      "BEMERKUNG"        VARCHAR(100) ,
      "FAX"              VARCHAR(20)
)
IN "STBDPOSTFILESTREET" ;

------------------------------------------------------------------------
-- Add constraint for primary keys
------------------------------------------------------------------------
ALTER TABLE  "ARTIKEL"      ADD CONSTRAINT "PK_ARTIKEL"      PRIMARY KEY ("ARTIKELID");
ALTER TABLE  "ART_SW"       ADD CONSTRAINT "PK_ART_SW"       PRIMARY KEY ("ARTIKELID", "SWID");
ALTER TABLE  "ART_AUT"      ADD CONSTRAINT "PK_ART_AUT"      PRIMARY KEY ("ARTIKELID", "AUTORID");
ALTER TABLE  "AUTOR"        ADD CONSTRAINT "PK_AUTOR"        PRIMARY KEY ("AUTORID");
ALTER TABLE  "BUCH_AUT"     ADD CONSTRAINT "PK_BUCH_AUT"     PRIMARY KEY ("BUCHID", "AUTORID");
ALTER TABLE  "BUCH_SW"      ADD CONSTRAINT "PK_BUCH_SW"      PRIMARY KEY ("BUCHID", "SWID");
ALTER TABLE  "BUCHTITEL"    ADD CONSTRAINT "PK_BUCHTITEL"    PRIMARY KEY ("BUCHID");
ALTER TABLE  "DISS_AUT"     ADD CONSTRAINT "PK_DISS_AUT"     PRIMARY KEY ("DISSID", "AUTORID");
ALTER TABLE  "DISS_SW"      ADD CONSTRAINT "PK_DISS_SW"      PRIMARY KEY ("DISSID", "SWID");
ALTER TABLE  "DISSERTATION" ADD CONSTRAINT "PK_DISSERTATION" PRIMARY KEY ("DISSID");
ALTER TABLE  "INSTITUTION"  ADD CONSTRAINT "PK_INSTITUTION"  PRIMARY KEY ("INSTID");
ALTER TABLE  "REPORT"       ADD CONSTRAINT "PK_REPORT"       PRIMARY KEY ("REPORTID");
ALTER TABLE  "REPORT_AUT"   ADD CONSTRAINT "PK_REPORT_AUT"   PRIMARY KEY ("REPORTID", "AUTORID");
ALTER TABLE  "REPORT_SW"    ADD CONSTRAINT "PK_REPORT_SW"    PRIMARY KEY ("REPORTID", "SWID");
ALTER TABLE  "SCHLAGWORT"   ADD CONSTRAINT "PK_SCHLAGWORT"   PRIMARY KEY ("SWID");
```

```
ALTER TABLE  "VERLAG"       ADD CONSTRAINT "PK_VERLAG"       PRIMARY KEY ("VERLAGSID");

------------------------------------------------------------------------
-- Add additional indexes
------------------------------------------------------------------------
CREATE INDEX  "I_ARTIKEL_BUCHID" ON  "ARTIKEL"  ("BUCHID" ASC) ALLOW REVERSE SCANS;

CREATE INDEX  "I_AUTOREN_NAME" ON  "AUTOR"  ("NAME" ASC,  "VORNAMEN" ASC)
       ALLOW REVERSE SCANS;


------------------------------------------------------------------------
-- Commit work
------------------------------------------------------------------------
COMMIT WORK;


ECHO ******************** END OF SCRIPT ********************;
```

Db2 Hofstetter Aufgabe 5

	Modul – 141 – Übung Database Security M141 – Theorie, Übungen und Praktikum	DBSEC - 1

Übungen zu DB-Security

Ziel der Übung: Sie lernen User für verschiedene Zwecke bereitzustellen und mit minimalen Rechten zu versehen. Beachten Sie, dass Sie die Autorisierungen in zwei Datenbanken (**DBBW001** und **DBBW002**) machen müssen. Speichern Sie die jeweiligen GRANT Statements in den vorbereiteten SQL-Scripts, damit Sie diese als Referenz verwenden können.

Die Resultate müssen auch verifiziert werden, d.h. es genügt nicht, nur die Rechte zu setzen, überprüfen Sie ob die Autorisierungen wie gewünscht funktionieren. Sie können dazu z.B. Verschiedene CLP Sessions öffnen oder in Ihrem SQL-Frontend verschiedene Connections (mit den jeweiligen Benutzern) definieren.

Übung 1

Vorbereitete Scripts: **Ueb1_CREATE_User.sh** (UNIX-Script für das Erstellen User und Gruppen)
Ueb1_GRANT_User.sql (SQL-Script für GRANTs)

- Erstellen Sie in Ihrer VM die UNIX Accounts **dbuser10**, **dbuser11** und **dbuser12** (setzen Sie als Passwort jeweils den Namen des entsprechenden Users). Für das Passwort soll kein Ablauf-Datum gesetzt werden.
- Erstellen Sie zudem beiden UNIX Gruppen **dbusrgrp** und **dbadmgrp**.

Diese Benutzer werden lediglich als „Connection-User" verwendet, d.h. es ist nicht vorgesehen, dass sie als UNIX Accounts für das Ausführen von Programmen verwendet werden (User können den CLP somit nicht aufrufen).

- Dem Benutzer **dbuser10** sollen keine weiteren UNIX Gruppen oder Rechte zugewiesen werden.
- Dem Benutzer **dbuser11** soll zusätzlich die UNIX Gruppe ***dbusrgrp*** zugewiesen werden
- Dem Benutzer **dbuser12** soll zusätzlich die UNIX Gruppe ***dbadmgrp*** zugewiesen werden
- Der Gruppe ***dbusrgrp*** soll das Privileg **DATAACCESS** in den Datenbanken **DBBW001** und **DBBW002** zugewiesen werden
- Der Gruppe ***dbadmgrp*** soll das Privileg **DBADM ohne *Datenzugriff*** in den Datenbanken **DBBW001** und **DBBW002** zugewiesen werden

Hinweise: Group-Commands: `groupadd groupmod groupdel`
User-Commands: `useradd usermod userdel chage passwd`

Damit ein Unix-User automatisch Zugriff auf alle DB2 Tools erhält (z.B. Aufruf des CLP's) müssen verschiedene Environment-Variablen und der PATH konfiguriert werden. Dies erfolgt automatisch, wenn Sie z.B. in **.bashrc** die folgenden Zeilen einfügen (muss für die in diesem Auftrag festgelegten User **nicht** gemacht werden):

```
if [ -f /home/db2inst1/sqllib/db2profile ]; then
    . /home/db2inst1/sqllib/db2profile
fi
```

Übung 2

Mit diesen Benutzern soll eine Connection zur Datenbank **DBBW001** erstellt und jeweils das folgende SQL-Statement (SELECT COUNT(*) …) ausgeführt werden, achten Sie darauf, dass die Verbindung zur DB mit dem richtigen User gemacht wird!!.

```
db2   CONNECT TO DBBW001 USER dbuser10
db2   "SELECT  COUNT(*) FROM  BIBLIO.TARTIKEL"
```

Hinweis: Im vorbereiten Verzeichnis finden Sie SQL-Scripts, die jeweils den CONNECT und das SELECT Statement ausführen (z.B. *Ueb2_DBBW001_dbuser10.sql*).

Dokumentieren Sie das Resultat der einzelnen Abfragen (als Resultat ist entweder die Anzahl Datensätze oder die SQL-Fehler-Meldung zu notieren):

mit **dbuser10** Resultat: _____

_____

mit **dbuser11** Resultat: _____

_____

mit **dbuser12** Resultat: _____

_____

Erstellen Sie in der Datenbank **DBBW002** mit dem **Instanz-User** die Tabelle **BIBLIO.TARTIKEL**. Das CREATE Statement finden Sie im Übungs-Verzeichnis (CREATE_TABLE_TARTIKEL.sql). Führen Sie nun das SELECT Statement nochmals für diese Tabelle in der Datenbank **DBBW002** aus.

mit **dbuser10** Resultat: _____

_____

mit **dbuser11** Resultat: _____

_____

mit **dbuser12** Resultat: _____

_____

Der Unterschied zwischen den beiden Datenbanken ist: die Datenbank **DBBW002** wurde beim CREATE mit der Option **RESTRICT** erstellt, d.h. es werden **keine** Autorisierungen automatisch erteilt (ausser für Developer-DB's sollte dies immer angegeben werden). Somit müssen alle Autorisierungen explizit erteilt werden.

Autorisieren Sie nun die Benutzer **dbuser10**, **dbuser11** und **dbuser12** damit Sie das SELECT-Statement in der Datenbank ausführen können. Speichern Sie sich die GRANT's im File *Ueb2_GRANT_User.sql*.

Übung 3

In dieser Übung sollen Sie mit den Benutzern **dbuser10**, **dbuser11** und **dbuser12** zwei Tabellen in der Datenbank **DBBW002** anlegen. D.h. die Verbindung zur DB muss jeweils mit diesen Benutzern aufgebaut werden

Verwenden Sie dazu das bereitgestellte SQL Script ***Ueb3_CREATE_TABLES.sql***. Beachten Sie, dass die beiden Tabellen im Default-Tablespace angelegt und **kein** Schema gesetzt ist, d.h. die Tabellen sollen im Default-Schema der User angelegt werden.

Was stellen Sie fest und was müssen Sie unternehmen (welche GRANT's müssen Sie erteilen), damit die Tabellen tatsächlich mit diesen Benutzern erstellt werden können?

Notieren Sie sich alle GRANT's, die Sie ausführen im File: *Ueb3_GRANT_User.sql*

Achten Sie darauf, dass Sie lediglich die minimal erforderlichen Rechte erteilen!!!

Übung 4

Neben expliziten Autorisierungen für User und Gruppen können auch Autorisierungen für funktionale Rollen erteilt werden. Diese Rollen können bei Bedarf Benutzern oder Gruppen zugewiesen werden.

Erstellen Sie in der Datenbank **DBBW002** eine Rolle **TESTER**. und erteilen Sie dieser Rolle die unten aufgeführten Privilegien auf alle Tabellen, die Sie in der Übung 3 erstellt haben, d.h. die Rolle TESTER muss die folgenden Rechte haben:

Tabelle:	**DBUSER10.TDBS_PERSON**	Rechte:	SELECT, INSERT, UPDATE und DELETE
Tabelle:	**DBUSER11.TDBS_PERSON**	Rechte:	SELECT, INSERT, UPDATE und DELETE
Tabelle:	**DBUSER12.TDBS_PERSON**	Rechte:	SELECT, INSERT, UPDATE und DELETE
Tabelle:	**DBUSER10.TDBS_ABTEILUNG**	Rechte:	SELECT, INSERT, UPDATE und DELETE
Tabelle:	**DBUSER11.TDBS_ABTEILUNG**	Rechte:	SELECT, INSERT, UPDATE und DELETE
Tabelle:	**DBUSER12.TDBS_ABTEILUNG**	Rechte:	SELECT, INSERT, UPDATE und DELETE

Speichern Sie die entsprechenden GRANT Statement in der Datei *Ueb4_GRANT_ROLE.sql*

Erstellen Sie nun die beiden UNIX Benutzer **tester01** und **tester02** und weisen Sie diesen Benutzern die Rolle **TESTER** zu.

Was muss zudem Autorisiert werden, damit neue Test-User Accounts automatisch den Zugriff auf diese Tabellen erhalten und entsprechende SQL-Statement ausführen können?

Die „Tester" sollen lediglich **minimale** Rechte erhalten (nur was sie tatsächlich benötigen) !!!

Übung 1

```
#!/usr/bin/ksh
#
# ACHTUNG: dies ist ein UNIX Script und KEIN SQL-Script !!!!!
#
# Sie koennen in diesem UNIX Script die notwendigen Statements fuer das Erstellen der
# Gruppen und User erfassen, denken Sie jedoch daran, dass SIe dieses Script als root
# User ausfuehren muessen.
#
# Hier die Commands fuer das Erstellen, Modifizieren und Loeschen von User und Groups:
#
# Groups:     groupadd    groupmod   groupdel
# User:       useradd     usermod    userdel     chage    passwd
#

if [[ $(whoami) != "root" ]]; then
   echo "sie muessen die Gruppen und User mit dem root User anlegen!!!"
   exit 16
fi

# Erstellen der Groups
groupadd dbusrgrp   # Befehl zum Erstellen einer Gruppe mit dem Namen "dbusrgrp"
groupadd dbadmgrp   # Befehl zum Erstellen einer Gruppe mit dem Namen "dbadmgrp"



# Erstellen der User
useradd dbuser10    # Befehl zum Erstellen eines Benutzers mit dem Namen "dbuser10"
passwd dbuser10     # Befehl zum Festlegen des Passworts für den Benutzer "dbuser10"
chage -I -1 -m 0 -M 99999 -E -1 dbuser10   # Befehl zur Änderung der Passwortrichtlinien für den Benutzer
"dbuser10"

useradd dbuser11    # Befehl zum Erstellen eines Benutzers mit dem Namen "dbuser11"
passwd dbuser11     # Befehl zum Festlegen des Passworts für den Benutzer "dbuser11"
chage -I -1 -m 0 -M 99999 -E -1 dbuser11   # Befehl zur Änderung der Passwortrichtlinien für den Benutzer
"dbuser11"

useradd dbuser12    # Befehl zum Erstellen eines Benutzers mit dem Namen "dbuser12"
passwd dbuser12     # Befehl zum Festlegen des Passworts für den Benutzer "dbuser12"
chage -I -1 -m 0 -M 99999 -E -1 dbuser12   # Befehl zur Änderung der Passwortrichtlinien für den Benutzer
"dbuser12"
```

```
usermod -a -G dbusrgrp dbuser11   # Befehl zum Hinzufügen des Benutzers "dbuser11" zur Gruppe "dbusrgrp"
usermod -a -G dbadmgrp dbuser12   # Befehl zum Hinzufügen des Benutzers "dbuser12" zur Gruppe
"dbadmgrp"



exit   # Beenden des Skripts
```

```
--
-- Speichern Sie in diesem SQL Script die notwendigen GRANT Statements
--

CONNECT TO DBBW001;
-- Autorisierungen für die Gruppe dbusrgrp
GRANT DATAACCESS ON DATABASE TO GROUP dbusrgrp;

-- Autorisierungen für die Gruppe dbadmgrp
GRANT DBADM WITHOUT DATAACCESS ON DATABASE TO GROUP dbadmgrp;

CONNECT TO DBBW002;
-- Autorisierungen für die Gruppe dbusrgrp
GRANT DATAACCESS ON DATABASE TO GROUP dbusrgrp;

-- Autorisierungen für die Gruppe dbadmgrp
GRANT DBADM WITHOUT DATAACCESS ON DATABASE TO GROUP dbadmgrp;
```

Übung 2

Dokumentieren Sie das Resultat der einzelnen Abfragen (als Resultat ist entweder die Anzahl Datensätze oder die SQL-Fehler-Meldung zu notieren):

mit **dbuser10** Resultat:

```
SQL0551N  The statement failed because the authorization ID does not have the
required authorization or privilege to perform the operation.  Authorization
ID: "DBUSER10".  Operation: "SELECT". Object: "BIBLIO.TARTIKEL".
SQLSTATE=42501
```

mit **dbuser11** Resultat:

```
[db2inst1@localhost ueb05]$ db2 "SELECT COUNT(*) FROM BIBLIO.TARTIKEL"
```

```
1
-----------
        19

  1 record(s) selected.
```

mit **dbuser12** Resultat:

```
[db2inst1@localhost ueb05]$ db2 "SELECT COUNT(*) FROM BIBLIO.TARTIKEL"
SQL0551N  The statement failed because the authorization ID does not have the
required authorization or privilege to perform the operation.  Authorization
ID: "DBUSER12".  Operation: "SELECT". Object: "BIBLIO.TARTIKEL".
SQLSTATE=42501
```

Erstellen Sie in der Datenbank DBBW002 mit dem Instanz-User die Tabelle BIBLIO.TARTIKEL. Das CREATE Statement finden Sie im Übungs-Verzeichnis (CREATE_TABLE_TARTIKEL.sql). Führen Sie nun das SELECT Statement nochmals für diese Tabelle in der Datenbank DBBW002 aus.

mit **dbuser10** Resultat:

```
[db2inst1@localhost ueb05]$ db2 "SELECT COUNT(*) FROM BIBLIO.TARTIKEL"

1
-----------
         0

  1 record(s) selected.
```

mit **dbuser11** Resultat:

```
[db2inst1@localhost ueb05]$ db2 "SELECT COUNT(*) FROM BIBLIO.TARTIKEL"

1
-----------
         0

  1 record(s) selected.
```

mit **dbuser12** Resultat:

```
[db2inst1@localhost ueb05]$ db2 "SELECT COUNT(*) FROM BIBLIO.TARTIKEL"

1
-----------
```

```
          0

  1 record(s) selected.
```

Übung 3

```
--
-- Create Statements fuer Uebung Database Security
--

CREATE TABLE TDBS_PERSON (
    PERSONID    INTEGER GENERATED BY DEFAULT AS IDENTITY (START WITH 1, INCREMENT BY 1, CACHE 20)
NOT NULL,
    NAME        VARCHAR(50) NOT NULL,
    VORNAME     VARCHAR(50) NOT NULL,
    KLASSE      VARCHAR(25) NOT NULL,
    LEHRBETRIEB VARCHAR(50)
)
;

CREATE TABLE TDBS_ABTEILUNG (
    ABTEILUNGID INTEGER GENERATED BY DEFAULT AS IDENTITY (START WITH 1, INCREMENT BY 1, CACHE 20)
NOT NULL,
    NAMEID      VARCHAR(10) NOT NULL,
    BEZEICHNUNG VARCHAR(50) NOT NULL,
    MANAGERIDFS INTEGER
)
;

CREATE UNIQUE INDEX IU03ABTEILUNG_PK
    ON TDBS_ABTEILUNG (ABTEILUNGID)
    PCTFREE 10
    MINPCTUSED 10
    ALLOW REVERSE SCANS
;

ALTER TABLE TDBS_ABTEILUNG
    ADD CONSTRAINT PK_TDBS_ABTEILUNG
    PRIMARY KEY (ABTEILUNGID)
```

```
;

CREATE UNIQUE INDEX IU03PERSON_PK
    ON TDBS_PERSON (PERSONID)
    PCTFREE 10
    MINPCTUSED 10
    ALLOW REVERSE SCANS
;

ALTER TABLE TDBS_PERSON
    ADD CONSTRAINT PK_TDBS_PERSON
    PRIMARY KEY (PERSONID)
;



SELECT * FROM TDBS_PERSON;

SELECT * FROM TDBS_ABTEILUNG;
```

```
-- Autorisierungen für User dbuser10
GRANT CREATETAB, IMPLICIT_SCHEMA ON DATABASE TO USER dbuser10;
GRANT USE OF TABLESPACE USERSPACE1 TO USER dbuser10;

-- Autorisierungen für User dbuser11
GRANT CREATETAB, IMPLICIT_SCHEMA ON DATABASE TO USER dbuser11;
GRANT USE OF TABLESPACE USERSPACE1 TO USER dbuser11;

-- Autorisierungen für User dbuser12
GRANT CREATETAB, IMPLICIT_SCHEMA ON DATABASE TO USER dbuser12;
GRANT USE OF TABLESPACE USERSPACE1 TO USER dbuser12;
```

Übung 4

```
--
-- Speichern Sie in diesem SQL Script die notwendigen GRANT Statements
--

CONNECT TO DBBW002
```

```
-- Erstellen der Rolle "TESTER"
CREATE ROLE TESTER;

-- Autorisierungen für die Rolle "TESTER"
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE DBUSER10.TDBS_PERSON TO ROLE TESTER;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE DBUSER11.TDBS_PERSON TO ROLE TESTER;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE DBUSER12.TDBS_PERSON TO ROLE TESTER;

GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE DBUSER10.TDBS_ABTEILUNG TO ROLE TESTER;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE DBUSER11.TDBS_ABTEILUNG TO ROLE TESTER;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE DBUSER12.TDBS_ABTEILUNG TO ROLE TESTER;

-- Setzen der Rolle "TESTER" für die Benutzer "tester01" und "tester02"
GRANT ROLE TESTER TO USER tester01;
GRANT ROLE TESTER TO USER tester02;

GRANT CONNECT ON DATABASE TO ROLE TESTER;
grant usage on workload SYSDEFAULTUSERWORKLOAD to ROLE TESTER;
GRANT EXECUTE ON PACKAGE NULLID.SQLC2P31 TO ROLE TESTER;
```

db2 Befehle

SCRIPT AUSFÜHREN

Für das Ausführen des SQL-Skripts "UEB03-00.sql" und das Anzeigen der Ergebnisse, nutzt du den "db2 -tvf"-Befehl. Hier ist der komplette Befehl:

```
db2 -tvf UEB03-00.sql
```

KONFIGURATION/PARAMETER AUSLESEN

Um die Konfiguration der DB2-Datenbankmanager-Instanz auszulesen und die Ergebnisse nach dem Parameter "numdb" zu filtern, verwendest du den folgenden Befehl:

```
db2 get dbm cfg | grep -i numdb
```

Zum Verbinden mit der Datenbank "DBBW001" und Auslesen ihrer Konfigurationsparameter, nutze diesen Befehl:

```
db2 connect to DBBW001
db2 get db cfg | grep -i log
```

KONFIGURATION/PARAMETER ANPASSEN

Zur Aktualisierung des Diagnoselevels der Datenbankmanager-Instanz auf "1" und zur Anpassung von Konfigurationsparametern, verwendest du diese Befehle:

```
db2 update dbm cfg using DIAGLEVEL 1
db2 update db cfg
db2 UPDATE DBM CFG USING NUMDB 32
db2 UPDATE DB CFG USING LOGSECOND 15
db2 UPDATE DB CFG FOR dbbw001 USING LOGSECOND 15
```

BERECHTIGUNGEN VERWALTEN

Um Berechtigungen zu vergeben, nutzt du die folgenden SQL-Statements. Sie spezifizieren die gewünschten Autorisierungen für bestimmte Nutzergruppen und individuelle Nutzer.

```
db2 CONNECT TO DBBW001;

-- Autorisierungen für die Gruppe dbusrgrp
db2 GRANT DATAACCESS ON DATABASE TO GROUP dbusrgrp;

-- Autorisierungen für die Gruppe dbadmgrp
db2 GRANT DBADM WITHOUT DATAACCESS ON DATABASE TO GROUP dbadmgrp;

db2 CONNECT TO DBBW002;

-- Autorisierungen für die Gruppe dbusrgrp
db2 GRANT DATAACCESS ON DATABASE TO GROUP dbusrgrp;

-- Autorisierungen für die Gruppe dbadmgrp
db2 GRANT DBADM WITHOUT DATAACCESS ON DATABASE TO GROUP dbadmgrp;

-- Autorisierungen für User dbuser10
db2 GRANT CREATETAB, IMPLICIT_SCHEMA ON DATABASE TO USER dbuser10;
db2 GRANT USE OF TABLESPACE USERSPACE1 TO USER dbuser10;

-- Autorisierungen für User dbuser11
db2 GRANT CREATETAB, IMPLICIT_SCHEMA ON DATABASE TO USER dbuser11;
db2 GRANT USE OF TABLESPACE USERSPACE1 TO USER dbuser11;

-- Autorisierungen für User dbuser12
db2 GRANT CREATETAB, IMPLICIT_SCHEMA ON DATABASE TO USER dbuser12;
db2 GRANT USE OF TABLESPACE USERSPACE1 TO USER dbuser12;
```

Zum Erstellen einer Rolle und zum Setzen von Berechtigungen für diese Rolle, nutze die folgenden Befehle:

```
db2 CONNECT TO DBBW002;

-- Erstellen der Rolle "TESTER"
db2 CREATE ROLE TESTER;

-- Autorisierungen für die Rolle "TESTER"
db2 GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE DBUSER10.TDBS_PERSON TO ROLE TESTER;
db2 GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE DBUSER11.TDBS_PERSON TO ROLE TESTER;
db2 GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE DBUSER12.TDBS_PERSON TO ROLE TESTER;

db2 GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE DBUSER10.TDBS_ABTEILUNG TO ROLE TESTER;
db2 GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE DBUSER11.TDBS_ABTEILUNG TO ROLE TESTER;
db2 GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE DBUSER12.TDBS_ABTEILUNG TO ROLE TESTER;

-- Setzen der Rolle "TESTER" für die Benutzer "tester01" und "tester02"
db2 GRANT ROLE TESTER TO USER tester01;
db2 GRANT ROLE TESTER TO USER tester02;

db2 GRANT CONNECT ON DATABASE TO ROLE TESTER;
db2 GRANT USAGE ON WORKLOAD SYSDEFAULTUSERWORKLOAD TO ROLE TESTER;
db2 GRANT EXECUTE ON PACKAGE NULLID.SQLC2P31 TO ROLE TESTER;
```