1. Praktikum

Page • 1 backlink • Tag

Erstellt von Erik Khelifa, Gruppe 15.

1.1 Datenbank-Account

erfolgreich eingewählt und das Passwort geändert.

1.2 Hello World

Datei erstelle:

Shell ⌄ Unwrap Copy

```
┌──(kali㉿kali)-[~/DBS/P1]
└─$ cat hello.sql
SELECT 'Hello World!';
```

Verzeichnis wechseln:

Shell ⌄

```
dbs169=> \cd /home/kali/DBS/P1/
dbs169=> \! pwd
/home/kali/DBS/P1
```

Skript ausführen:

Shell ⌄

```
dbs169=> \i hello.sql
   ?column?
--------------
 Hello World!
(1 row)
```

1.3 Arbeiten mit der psql shell

1. Legen Sie eine beliebige Tabelle mit nur einer Spalte direkt in psql an (CREATE TABLE ...). [3, Folie 8]

```
Shell
dbs169=> CREATE TABLE beispieltabelle(
  spalte1 int
);
CREATE TABLE
```

2. Zeigen Sie die Struktur dieser Tabelle an (welches Metakommando?).

```
Shell
dbs169=> \d
             List of relations
 Schema |       Name        | Type  | Owner
--------+-------------------+-------+--------
 public | beispieltabelle | table | dbs169
```

3. Erstellen Sie ein SQL-Script insert.sql, welches SQL-Befehle zum Befüllen der Tabelle mit Inhalten Ihrer Wahl enthält [3, Folien 45-47]. Führen Sie dieses Skript in psql aus (welches Metakommando?).

   Skript erstelle:

```
Shell
┌──(kali㉿kali)-[~/DBS/P1]
└─$ nano insert.sql

┌──(kali㉿kali)-[~/DBS/P1]
└─$ cat insert.sql
INSERT INTO beispielstabelle (spalte1) VALUES (1);
INSERT INTO beispielstabelle (spalte1) VALUES (2);
INSERT INTO beispielstabelle (spalte1) VALUES (3);
INSERT INTO beispielstabelle (spalte1) VALUES (4);
INSERT INTO beispielstabelle (spalte1) VALUES (5);
INSERT INTO beispielstabelle (spalte1) VALUES (6);
INSERT INTO beispielstabelle (spalte1) VALUES (7);
INSERT INTO beispielstabelle (spalte1) VALUES (8);
INSERT INTO beispielstabelle (spalte1) VALUES (9);
```

   Skript ausführen:

```
Shell
dbs169=> \i insert.sql
INSERT 0 1
```

```
INSERT 0 1
INSERT 0 1
INSERT 0 1
INSERT 0 1
INSERT 0 1
INSERT 0 1
INSERT 0 1
INSERT 0 1
```

Inhalt der Tabelle:

```
Shell
dbs169=> SELECT * FROM beispieltabelle
;
 spalte1
---------
       1
       2
       3
       4
       5
       6
       7
       8
       9
(9 rows)
```

4. Entfernen Sie die Tabelle aus der Datenbank. [3, Folie 39]

```
Shell
dbs169=> DROP TABLE beispieltabelle;
DROP TABLE
dbs169=> \d
Did not find any relations.
```

1.4 Tabellen anlegen und Datenimport

Definition der 3 Tabellen:

```
Shell
CREATE TABLE Import (
    BestellID VARCHAR(69),
```

```
    KundenID VARCHAR(69),
    Bestelldatum VARCHAR(69),
    Betrag VARCHAR(69),
    Status VARCHAR(69),
    Titel VARCHAR(69),
    Vorname VARCHAR(69),
    Name VARCHAR(69),
    PLZ VARCHAR(69),
    Ort VARCHAR(69)
);

CREATE TABLE Kunde (
    KundenID SERIAL PRIMARY KEY,
    Titel VARCHAR(69),
    Vorname VARCHAR(69) NOT NULL,
    Name VARCHAR(69) NOT NULL,
    PLZ VARCHAR(5) NOT NULL,
    Ort VARCHAR(69) NOT NULL
);

CREATE TABLE Bestellung (
    BestellID SERIAL PRIMARY KEY,
    KundenID INT NOT NULL,
    Bestelldatum DATE NOT NULL,
    Betrag DECIMAL(10,2) NOT NULL,
    Status VARCHAR(69) NOT NULL,
    FOREIGN KEY (KundenID) REFERENCES Kunde(KundenID)
);
```

Erstellen der 3 Tabellen:

Shell

```
dbs169=> \i import.sql
CREATE TABLE
CREATE TABLE
CREATE TABLE
```

Importieren der daten mit:

Shell

```
dbs169=> \copy import FROM 'P1-data.csv' WITH (FORMAT csv, DELIMITER
';', HEADER true, ENCODING 'UTF-8')
COPY 100
```

10x100 Daten, also 1000 eingelesen!

Mit `SELECT * FROM Import`; die Importierten daten angezeigt und gesehen, dass alle umlaute korrekt sind.

Skript für das insert Into:

```shell
INSERT INTO Kunde (Titel, Vorname, Name, PLZ, Ort)
Select DISTINCT Titel, Vorname, Name, PLZ, Ort
FROM Import;

INSERT INTO Bestellung (KundenID, Bestelldatum, Betrag, Status)
Select DISTINCT CAST(KundenID AS int), to_date(Bestelldatum, 'YYYY-MM-DD'), CAST(Betrag AS DECIMAL(10,2)), Status
FROM Import;
```

SELECT DISTINCT wird benötigt, damit Kunden mit mehreren Bestellungen nicht mehrfach angelegt werden.

25 Kunden und 100 Bestellungen erfolgreich importiert.

```shell
dbs169=> SELECT count(*) FROM Kunde;
 count
-------
    25
(1 row)

dbs169=> SELECT count(*) FROM bestellung;
 count
-------
   100
(1 row)
```

1.5 Änderungen und Abfragen

1.

```shell
dbs169=> INSERT INTO Bestellung (KundenID, Bestelldatum, Betrag, Status) VALUES (
    (SELECT KundenID FROM Kunde WHERE Vorname = 'Julie' AND Name = 'Schuster'),
    to_date('2024-10-14', 'YYYY-MM-DD'),
```

```
    49.95,
    'In Bearbeitung'
);
INSERT 0 1
```

2.

```
Shell
dbs169=> DELETE FROM Bestellung WHERE Status = 'Storniert';
DELETE 34
```

3.

```
Shell
dbs169=> UPDATE Bestellung SET Status = 'Abgeschlossen' WHERE Status =
'In Bearbeitung' AND Bestelldatum < '2024-06-01';
UPDATE 24
```

4.

Shell

```
dbs169=> SELECT * FROM Bestellung WHERE Betrag > 400 ORDER BY Betrag
DESC;
 bestellid | kundenid | bestelldatum | betrag |     status
-----------+----------+--------------+--------+----------------
        52 |        5 | 2024-06-06   | 497.76 | In Bearbeitung
        44 |       15 | 2024-04-13   | 496.81 | Abgeschlossen
         4 |        8 | 2024-02-22   | 490.43 | Abgeschlossen
        84 |       16 | 2024-06-25   | 490.04 | In Bearbeitung
        99 |        6 | 2024-06-22   | 487.99 | Abgeschlossen
        18 |       15 | 2024-01-27   | 483.94 | Abgeschlossen
        95 |       20 | 2024-02-24   | 460.58 | Abgeschlossen
        79 |        7 | 2024-05-02   | 462.60 | Abgeschlossen
        85 |       12 | 2024-01-25   | 460.61 | Abgeschlossen
         8 |        8 | 2024-04-25   | 454.94 | Abgeschlossen
         6 |       21 | 2024-05-30   | 450.98 | Abgeschlossen
        78 |        8 | 2024-03-15   | 445.88 | Abgeschlossen
        97 |        8 | 2024-05-25   | 436.79 | Abgeschlossen
        14 |       19 | 2024-05-30   | 436.31 | Abgeschlossen
        46 |       14 | 2024-02-21   | 435.78 | Abgeschlossen
        92 |       12 | 2024-03-20   | 424.13 | Abgeschlossen
        35 |        6 | 2024-06-24   | 423.95 | Abgeschlossen
        23 |       10 | 2024-06-14   | 413.50 | In Bearbeitung
       100 |       10 | 2024-06-25   | 403.23 | Abgeschlossen
(19 rows)
```

5.

Shell

```
dbs169=> SELECT k.Ort FROM Kunde k JOIN Bestellung b on k.KundenID =
b.KundenID WHERE b.bestelldatum >= '2024-06-01' AND b.bestelldatum <
'2024-07-01'  GROUP BY k.Ort;
       ort
-----------------
 Duisburg
 Viersen
 Krefeld
 Mönchengladbach
(4 rows)
```

6.

Shell

```
dbs169=> SELECT k.Ort, Count(b.*) from Kunde k JOIN Bestellung b on
k.KundenID = b.KundenID GROUP BY Ort ORDER BY Count(*) DESC;
```

```
       ort       | count
-----------------+-------
 Viersen         |    19
 Düsseldorf      |    16
 Duisburg        |    14
 Krefeld         |    12
 Mönchengladbach |     6
(5 rows)
```

Komplettes Skript für 1.4 und 1.5

Shell

```
CREATE TABLE Import (
    BestellID VARCHAR(69),
    KundenID VARCHAR(69),
    Bestelldatum VARCHAR(69),
    Betrag VARCHAR(69),
    Status VARCHAR(69),
    Titel VARCHAR(69),
    Vorname VARCHAR(69),
    Name VARCHAR(69),
    PLZ VARCHAR(69),
    Ort VARCHAR(69)
);

CREATE TABLE Kunde (
    KundenID SERIAL PRIMARY KEY,
    Titel VARCHAR(69),
    Vorname VARCHAR(69) NOT NULL,
    Name VARCHAR(69) NOT NULL,
    PLZ VARCHAR(5) NOT NULL,
    Ort VARCHAR(69) NOT NULL
);

CREATE TABLE Bestellung (
    BestellID SERIAL PRIMARY KEY,
    KundenID INT NOT NULL,
    Bestelldatum DATE NOT NULL,
    Betrag DECIMAL(10,2) NOT NULL,
    Status VARCHAR(69) NOT NULL,
    FOREIGN KEY (KundenID) REFERENCES Kunde(KundenID)
);

\copy import FROM 'P1-data.csv' WITH (FORMAT csv, DELIMITER ';', HEADER
```

```
true, ENCODING 'UTF-8');

INSERT INTO Kunde (Titel, Vorname, Name, PLZ, Ort)
Select DISTINCT Titel, Vorname, Name, PLZ, Ort
FROM Import;

INSERT INTO Bestellung (KundenID, Bestelldatum, Betrag, Status)
Select DISTINCT CAST(KundenID AS int), to_date(Bestelldatum, 'YYYY-MM-
DD'), CAST(Betrag AS DECIMAL(10,2)), Status
FROM Import;

INSERT INTO Bestellung (KundenID, Bestelldatum, Betrag, Status) VALUES
(
    (SELECT KundenID FROM Kunde WHERE Vorname = 'Julie' AND Name =
'Schuster'),
    to_date('2024-10-14', 'YYYY-MM-DD'),
    49.95,
    'In Bearbeitung'
);

DELETE FROM Bestellung WHERE Status = 'Storniert';

UPDATE Bestellung SET Status = 'Abgeschlossen' WHERE Status = 'In
Bearbeitung' AND Bestelldatum < '2024-06-01';

SELECT * FROM Bestellung WHERE Betrag > 400 ORDER BY Betrag DESC;

SELECT k.Ort FROM Kunde k JOIN Bestellung b on k.KundenID = b.KundenID
WHERE b.bestelldatum >= '2024-06-01' AND b.bestelldatum < '2024-07-01'
GROUP BY k.Ort;

SELECT k.Ort, Count(b.*) from Kunde k JOIN Bestellung b on k.KundenID =
b.KundenID GROUP BY Ort ORDER BY Count(*) DESC;
```