Modul 100

Material: bmLP1 Kubuntu, Konsole, SQLite3, Sqliteman, Datei: Modul100/training.db, CSV's, AB's
SQLite: Kein Right Join
m-m: Dies muss zu einer 1-m m-1 aufgelöst werden, indem man eine Zwischentabelle macht.

Abteilung - Mitarbeiter 1-m
Mitarbeiter - Arbeitet_an 1-m
Arbeitet_an - Projekt m-1

Linux Konsole

Befehl	Beschreibung
mkdir Modul100	Ordner erstellen (alle Skript+Datenbanken+CSV dort drinnen)
cd Modul100	In den Ordner
sqlite3 training.db	Datenbank erstellen/öffnen
sqlite3	Öffnet interaktiven Modus
.databases	Zeigt Datenbanken an
.help	Zeigt alle Befehle an
.tables	Zeigt Tabellen an
Ctrl+d	SQLite beenden
Man	Anleitung
CSV Dateien einfügen	
CREATE TABLE Tabellennamen (Attributname1 Datentyp,...);	Tabelle in Linux machen (sqlite3 datenbank muss geöffnet sein!)
.separator „;“	Trennungen definieren
.import Datei.csv Tabelle	Datei importieren in Tabelle

Data Definition Language (DDL)

Befehl	Beschreibung
CREATE / ALTER / DROP TABLE / DATABASE / INDEX	Erstellen / ändern / löschen
CREATE TABLE Tabellennamen (Attributname1 Datentyp PRIMARY KEY, Attributname2 Datentyp,...);	Tabelle erstellen Datentyp: TEXT, INTEGER, REAL
ALTER TABLE Tabellennamen ADD COLUMN AttributnameN Datentyp;	Spalte hinzufügen
ALTER TABLE Tabellennamen DROP COLUMN AttributnameN;	Spalte löschen

Data Manipulation Language (DML)

Befehl optional	Beschreibung
SELECT [AGGREGATION()] Attributname1 [AS „Alias“], Attributname2,... FROM Tabellennamen [WHERE AttributnameN [=, <, <=, >, >=, LIKE, BETWEEN, IS, NOT] Datenwert] [AND, OR, NOT] [ORDER BY Attributname1 [DESC, ASC]] [GROUP BY Attributname1] [HAVING COUNT(*) > 2] [LIMIT N];	[Aggregation wie MIN/MAX/DISTINCT],[Ein Alias setzen] Eine Einschränkung setzen z.B. IS NULL, NOT IN [DESC=Absteigend ZYX ASC=Aufsteigend ABC] [Gruppieren] [Wenn eine Aggregatsfunktion gebraucht wird] [Eine Limite setzen]
INSERT INTO Tabellennamen (Attributname1, Attributname2,...) VALUES ('Datenwert1','Datenwert2',...);	Eine Zeile hinzufügen
UPDATE Tabellennamen SET Attributname1 = Datenwert1 AttributnameN = DatenwertN WHERE AttributnameN [=, <, <=, >, >=, LIKE, BETWEEN, IS, NOT] Datenwert [AND, OR, NOT]	Einen Datenwert aktualisieren
DELETE FROM Tabellennamen WHERE AttributnameN [=, <, <=, >, >=, LIKE, BETWEEN, IS, NOT] Datenwert [AND, OR, NOT]	Eine Zeile löschen
SELECT AttributnameN FROM TabelleL [INNER / LEFT / RIGHT JOIN] TabelleR ON AttrL=AttrR;	Eine weitere Tabelle hinzufügen

Aggregatsfunktionen	Beschreibung
SUM	Summiert die Werte in einer Spalte. Der Datentyp muss numerisch sein!
AVG	Gibt den Mittelwert in einer Spalte zurück. Der Datentyp muss numerisch sein!
COUNT	Gibt die Anzahl von Einträgen in einer Gruppe zurück.
DISTINCT	Jeden Wert nur einmal
COUNT DISTINCT	Gibt die Anzahl von eindeutigen Werten ungleich NULL in einer Gruppe zurück.
MIN	Gibt den kleinsten Wert in einer Gruppe zurück.
MAX	Gibt den grössten Wert in einer Gruppe zurück.

Bestimmte Befehle	Beschreibung
SELECT ROUND(AVG(GehaltAttribut)/5,2)*5 FROM Tabelle;	Auf Franken runden
SELECT * FROM Personen WHERE Vorname BETWEEN ‚F%‘ AND ‚I%‘;	%, * heisst es könnte unendlich weiter gehen _, ? ist für ein Buchstabe

Domäne = Wertebereich / Datentyp

Semantik - Bedeutung

INTEGER, TEXT, REAL

Alias klein

CREATE TABLE Kunde

```
CREATE TABLE Kunde (
   kID       INTEGER PKey
   Name      TEXT
   Anrede    TEXT
   PLZ       INTEGER
);
```

Tabelle Kunde

kID	Name	Anrede	PLZ
1	Blum	Herr	3012
2	Meier	Herr	3072
3	Haller	Frau	3036
4	Huber	Herr	3072

SELECT / LIMIT

alle Attribute (*):

```
SELECT * FROM Kunde;
```

Attribute gefiltert:

```
SELECT Anrede, Name
          FROM Kunde;
```

Ausgabe von 10 Datensätzen:

```
SELECT * FROM Kunde
   LIMIT 10;
```

ORDER-Klausel

Aufsteigend sortiert:

```
SELECT Anrede, Name
          FROM Kunde
   ORDER BY Name ASC;
```

Absteigend sortiert:

```
SELECT Anrede, Name
          FROM Kunde
   ORDER BY Name DESC;
```

WHERE Klausel

Datensätze filtern:

```
SELECT Name, PLZ
          FROM Kunde
   WHERE PLZ = 3000 OR PLZ = 3072
   ORDER BY Name;
```

```
SELECT Name, PLZ
          FROM Kunde
   WHERE Name LIKE 'M%'
   ORDER BY PLZ;
```

AND, OR: filtert Datensätze mit mehreren Bedingungen

GROUP BY / COUNT / Alias

Gruppieren und zählen:

```
SELECT COUNT(*) AS "Anzahl",
       Anrede FROM Kunde
   GROUP BY Anrede;
```

Alias: Benennung / Kürzel
Spalte: ... COUNT(*) AS "Anzahl", ...
Tabelle: ... FROM Kunde AS k

Anzahl	Anrede
1	Herrn
2	Frau

INSERT INTO

```
INSERT INTO Kunde
   (Name, Anrede, PLZ)
   VALUE
   ('Huber', 'Herrn', 3036);
```

kID wird vom System ermittelt

DELETE

Löscht Datensatz 3:

```
DELETE FROM Kunde
   WHERE kID = 3;
```

Löscht alle Datensätze mit Name M:

```
DELETE FROM Kunde
   WHERE Name LIKE M%;
```

UPDATE

```
UPDATE  Kunde
   SET Name = 'Meier',
       Anrede = 'Frau'
   WHERE kID = 2;
```

```
UPDATE  Kunde
   SET PLZ = 3000
   WHERE PLZ = 3012;
```

Tabelle Ort und Kunde

PLZ	Ort
3000	Bern
3036	Frieswil
3072	Ostermundigen
3110	Münsigen
3270	Aarberg
3280	Murten

kID	Name	Anrede	PLZ
1	Blum	Herrn	3000
2	Meier	Frau	3072
3	Haller	Frau	3000
4	Haller	Herrn	3072
5	Huber	Herrn	3072

(INNER) JOIN

```
SELECT k.Name, o.Ort FROM Ort as o
   JOIN Kunde As k ON o.PLZ = k.PLZ
      ORDER BY o.Ort ASC;
```

Liefert alle Kunden mit ihren Orten (d.h. nur grau markierte Datensätze)

Ausgabe JOIN-Abfrage

Name	Ort
{null}	Aarberg
Blum	Bern
Haller	Bern
Haller	Frieswil
{null}	Murten
{null}	Münsigen
Meier	Ostermundigen
Huber	Ostermundigen

LEFT JOIN

```
SELECT k.Name, o.Ort FROM Ort as o
   LEFTJOIN Kunde As k ON
            o.PLZ = k.PLZ
      ORDER BY o.Ort ASC;
```

Liefert alle Orte auch ohne Kunden (d.h. die ganze Tabellenausgabe im Feld links)

Vergleichsoperatoren

Operator	Erklärung
= < <= >= > <>	vergleicht ein Attributwert mit einem anderen
BETWEEN ... AND ...	vergleicht, Attributwert zwischen zwei Grenzen
LIKE	vergleich von Zeichenketten (Muster) %: Platzhalter für beliebige Zeichen _: Platzhalter für ein Zeichen
IS (NOT) NULL	prüft, ob ein Attributwert (nicht) undefiniert ist

DISTINCT

```
SELECT DISTINCT(Name)
          FROM Kunde;
```

Gibt Mehrfachdatensätze (z.B.: Haller) nur einmal aus

MIN / MAX / SUM

MIN: liefert den kleinsten Attributwert

```
SELECT MIN(Lohn) FROM Person;
```

MAX: liefert den grössten Attributwert

```
SELECT MAX(Lohn) FROM Person;
```

SUM: liefert die Summe einer Spalte

```
SELECT SUM(Lohn) FROM Person;
```

ROUND / AVG

```
SELECT ROUND(AVG(Lohn), 2)
          FROM Person;
```

ROUND: 2 Kommastellen runden

AVG: Mittelwert der Attributwerte

CTRL F für Suche