Labor 3 - Datenbank mit MySQL

Hinweis: Dieses Labor entstand z.T. aus Scripten von Prof. Dr. U. Bannier.

1. Starten des MySQL-Systems

MySQL ist ein unter www.mysql.com kostenlos erhältliches Datenbankmanagementsystem. Im Labor müssen Sie den MySQL-Server Stand-Alone über einen Batch-File starten. Rufen Sie unter **Labor 3** den Befehl **`start-mysql.bat`** auf. Rufen Sie dann den Befehl **`win-mysqladmin.exe`** auf, unter **`c:\mysql\bin`**.

Zur Eingabe der (unten beschriebenen) SQL-Befehle öffnen Sie ein DOS-Fenster, gehen, über **`cd,`** in das Verzeichnis **`c:\mysql\bin`** und starten dort den MySQL.Client **`mysql.exe`**.

Dieses Programm meldet sich mit dem Prompt

```
mysql>
```

MySQL.exe beendet man durch **`exit`**.

2. Struktur der Beispiel-Datenbank „personal“

Wir betrachten ein hypothetisches Unternehmen, das mehrere Niederlassungen in verschiedenen Orten hat. Jede Niederlassung hat Abteilungen mit gleichlautenden Bezeichnungen. Jeder Mitarbeiter des Unternehmens hat eine Personalnummer, die im ganzen Unternehmen eindeutig ist. Es soll möglich sein, dass ein Mitarbeiter mehrere Telefonnummern hat.

Die einzelnen Tabellen enthalten folgende Attribute:

Tabelle niederlassung**:**

nnr	Niederlassungsnummer (Primärschlüssel)
ort	Ort der Niederlassung

Tabelle abteilung**:**

nnr	Niederlassungsnummer (Primärschlüsselattribut)
anr	Abteilungsnummer (Primärschlüsselattribut)
aname	Name der Abteilung

Tabelle person**:**

pnr	Personalnummer (Primärschlüssel)
name	Nachname der Person
nnr	Niederlassungsnummer
anr	Abteilungsnummer

Tabelle telefon**:**

tnr	Telefonnummer (Primärschlüssel)
pnr	Personalnummer

3. Anlegen einer Datenbank

Alle vorhandenen Datenbanken anzeigen lassen: ***SHOW DATABASES;***
Falls die DB „personal“ bereits existiert, diese löschen: ***DROP DATABASE personal;***

Neue Datenbank „personal“ anlegen: ***CREATE DATABASE personal;***

Auswahl von „personal“ zur weiteren Bearbeitung: ***USE personal;***

4. Definition der Tabellen

Tabellen-Spalten mit Namen und Datentyp definieren, hier für Tabelle „niederlassung“.

Falls die Tabelle bereits existiert, diese löschen mit ***DROP TABLE niederlassung;***

```
CREATE TABLE niederlassung
(
  nnr          INT NOT NULL,     // Niederlassungsnummer
  ort          VARCHAR(100),     // Ort
  PRIMARY KEY  (nnr)             // nnr als Primärschlüssel
);
```

Anzeige der Tabelle „niederlassung“: ***EXPLAIN niederlassung;***

Alle Tabellen der aktuellen DB anzeigen: ***SHOW TABLES;***

Aufgaben:

1. Definieren Sie die Tabellen ***abteilung,*** definiert in Abschnitt 2. Verwenden Sie für ***aname*** `VARCHAR(100)`, für die anderen beiden Attribute `INT`. Legen Sie **keinen** Primärschlüssel fest

2. Definieren Sie die Tabellen ***person,*** definiert in Abschnitt 2. Verwenden Sie für ***pnr*** `CHAR(3)`, für ***name*** `VARCHAR(100)`, für die anderen beiden Attribute `INT`. Legen Sie den Primärschlüssel `(pnr)` fest.

3. Definieren Sie die Tabellen ***telefon*** aus Abschnitt 2.1. Verwenden Sie für ***tnr*** `CHAR(20)`. Legen Sie den Primärschlüssel (tnr) fest.

Achten Sie darauf, welche Attribute `NOT NULL` sein müssen, speziell die für den PRIMARY KEY, und dass Sie Strings bei der Eingabe zwischen Hochkomma setzen müssen..

5. Eingabe von Daten

Daten in die Tabelle „niederlassung“ eingeben (Niederlassungsnummer und Ort) :

INSERT INTO niederlassung(nnr,ort) VALUES
(1, 'Berlin'),
(2, 'Hamburg');

6. Ausgabe von Tabellen

Ausgabe der Tabelle „niederlassung“: ***SELECT * FROM niederlassung;***
Ausgabe ordnen: ***SELECT * FROM niederlassung ORDER BY nnr;***

Aufgabe:

Geben Sie alle Daten der übrigen Tabellen vollständig ein, Daten s.u. Kontrollieren Sie anschließend den Inhalt der Tabellen mit dem `SELECT` Befehl :

SELECT * FROM abteilung;

nnr	anr	aname
1	1	Vertrieb
1	2	Personal
2	1	Vertrieb
2	3	Konstruktion

SELECT * FROM person;

pnr	name	nnr	anr
112	Boss	1	1
147	Klose	1	1
301	Klose	1	2
100	Jung	2	1
200	Burg	2	3

SELECT * FROM telefon;

tnr	pnr
030-21-200	112
030-21-211	112
030-21-231	147
030-21-244	301
040-33-151	200
040-33-153	100

7. Ändern eines Datensatzes

Einfügen eines Datensatzes:

INSERT INTO telefon (tnr,pnr) VALUES ('040-33-155','100');

Ändern dieses Datensatzes mit UPDATE, tnr wird ausgewählt und geändert :

UPDATE telefon SET tnr = '040-33-154' WHERE tnr = '040-33-155';

Löschen eines Datensatzes:

DELETE FROM telefon WHERE tnr = '040-33-154';

8. Hinzufügen einer Spalte

Ändern der Tabelle „person“ durch neue Spalte ***vname*** :

ALTER TABLE person ADD vname VARCHAR(100);

Daten der Vornamen mit UPDATE hinzufügen, da die Datensätze schon existieren:

```
UPDATE  person  SET     vname = 'Peter'     WHERE pnr = '112';
UPDATE  person  SET     vname = 'Jan'       WHERE pnr = '147';
UPDATE  person  SET     vname = 'Ilona'     WHERE pnr = '301';
UPDATE  person  SET     vname = 'Thomas'    WHERE pnr = '100';
UPDATE  person  SET     vname = 'Paul'      WHERE pnr = '200';
```

Löschen eines Attributs:

ALTER TABLE person DROP vname;

Löschen des Primärschlüssels:

ALTER TABLE person DROP PRIMARY KEY;

9. Projektion

Spalten in einer bestimmten Reihenfolge auszuwählen.

```
SELECT DISTINCT pnr, vname, name FROM person;
```

10. Selektion

Mit Hilfe der WHERE-Klausel:

```
SELECT pnr,vname,name FROM person WHERE name='Klose';
```

11. Join

Abfrage über mehrere Tabellen :

Qualifizierung: Spalten der einzelnen Tabellen eindeutig ansprechen, indem man den Tabellennamen vor den Spaltennamen schreibt, z.B. `person.pnr` die Personalnummer in der Tabelle `person` und `telefon.pnr` diejenige in der Tabelle `telefon`.

Die folgende Abfrage soll eine Telefonliste mit Personalnummer, Nachnamen und Telefon der Mitarbeiter erstellen. Für die Personalnummer und den Nachnamen benötigen wir die Basistabelle `person`, für die Telefonnummer die Basistabelle `telefon`.

```
SELECT person.pnr, person.name, telefon.tnr
FROM   person, telefon
WHERE person.pnr=telefon.pnr;
```

Die Verknüpfung erfolgt über gleiche Fremdschlüssel- und Primärschlüsselwerte.

Aliasnamen: z.B. für `person` der Aliasname `p` und für `telefon` der Aliasname `t` :

```
SELECT  p.pnr, p.name, t.tnr
FROM    person p, telefon t
WHERE  p.pnr=t.pnr;
```

Weiteres Beispiel einer Abfrage :

```
SELECT p.pnr, p.name, t.tnr, a.aname
FROM   person p, telefon t, abteilung a
WHERE  p.pnr = t.pnr  AND a.nnr = p.nnr AND a.anr = p.anr;
```

Lösungen

1. Tabelle **abteilung** :

```
CREATE TABLE abteilung
(
  anr              INT NOT NULL,
  nnr              INT NOT NULL,
  aname            VARCHAR(100)
);

INSERT INTO abteilung (nnr,anr,aname) VALUES
(1, 1, 'Vertrieb'),
(1, 2, 'Personal'),
(2, 1, 'Vertrieb'),
(2, 3, 'Konstruktion');
```

Andere Version: anr als Zeichen VARCHAR(1) statt als INT:

```
INSERT INTO abteilung (nnr,anr,aname) VALUES
(1, V, 'Vertrieb'),
(1, P, 'Personal'),
(2, V, 'Vertrieb'),
(2, K, 'Konstruktion');
```

2. Tabelle **person** :

```
CREATE TABLE person
(
   pnr             CHAR(3) NOT NULL,
  name             VARCHAR(100) NOT NULL,
   anr             INT NOT NULL,
   nnr             INT NOT NULL,
   PRIMARY KEY     (pnr)
);

INSERT INTO person (pnr,name,nnr,anr) VALUES
('112', 'Boss',  1, 1 ),
('147', 'Klose', 1,1 ),
('301', 'Klose', 1,2  ),
('100', 'Jung',  2,1 ),
('200', 'Burg',  2, 3 ) ;
```

3. Tabelle **telefon** :

```
CREATE TABLE telefon
(
   tnr                    VARCHAR(20) NOT NULL,
   pnr                CHAR(3) ,
   PRIMARY KEY    (tnr)
);

INSERT INTO telefon  (tnr,pnr) VALUES
( '030-21-200',  '112' ),
( '030-21-211', '112' ),
( '030-21-231', '147' ),
( '030-21-244',  '301' ),
( '040-33-151',  '200' ),
( '040-33-153',  '100' ) ;
```