

## Datenbanksysteme II

### SS 2018 – Übungsblatt 1

#### Aufgabe 1 (Embedded SQL, CLI)

- Vergleichen Sie die Konzepte *Embedded SQL* und *Call Level Interface* zur Kopplung von DBS mit einer Programmiersprache.
- Unterscheiden Sie statisches und dynamisches SQL.
- Was versteht man unter dem Cursor-Konzept? Welche Operatoren werden durch das Cursor-Konzept bereitgestellt und was bewirken sie?
- In Aufgabe 2 wird ein Schema zur Verwaltung von Freundschaftsbeziehungen definiert. Skizzieren Sie ein vereinfachtes C-Programm, welches mittels *Embedded SQL* zu jedem Nutzer dessen Namen sowie die Anzahl seiner Freunde ausgibt. Die Freundschaftsbeziehungen sind ungerichtet, jede Kante wird nur einmal abgespeichert. Orientieren Sie sich dabei an folgendem Template:

```
exec sql include sqlca;
```

```
main() {
```

```
    exec sql begin declare section;
```

```
        int    uid;
```

```
        int    count;
```

```
        char   name[100];
```

```
    exec sql end declare section;
```

```
    exec sql connect to dbname;
```

```
    exec sql declare C1 cursor for
```

```
        [ ]
```

```
    exec sql declare C2 cursor for
```

```
        [ ]
```

```
    exec sql open C1;
```

```
    while (sqlcode == ok) {
```

```
        [ ]
```

```
        printf ("%s: ", name);
```

```
        [ ]
```

```
        while (sqlcode == ok) {
```

```
            [ ]
```

```
            printf ("%d\n", count);
```

```
        }
```

```
    exec sql close C2;
```

```
    [ ]
```

```
    exec sql disconnect;
```

```
}
```

## Aufgabe 2 (JDBC)

Es sei folgendes Datenbankschema zum Thema „Soziales Netzwerk“ gegeben.

### Nutzer (

```
uid          INT NOT NULL PRIMARY KEY,  
email        VARCHAR(50),  
name         VARCHAR(100)
```

);

### Freundschaft (

```
uid1         INT NOT NULL PRIMARY KEY,  
uid2         INT NOT NULL PRIMARY KEY,  
Fgrad        VARCHAR(100),  
FOREIGN KEY uid1 REFERENCES Nutzer (uid),  
FOREIGN KEY uid2 REFERENCES Nutzer (uid)
```

);

- Erläutern Sie zunächst, was der grundlegende Unterschied zwischen den JDBC-Methoden *executeQuery()* und *executeUpdate()* ist! Was ist jeweils deren Rückgabewert?
- Angenommen ein JDBC *ResultSet* *rs* enthält Spalten der Tabelle *Nutzer* nach Ausführung der Query `SELECT uid, name FROM Nutzer`. Geben Sie zwei Varianten an, um auf die Spalte *name* in *rs* zuzugreifen.
- Erläutern Sie kurz die zwei wesentlichen Vorteile der Verwendung von *Prepared Statements* im Vergleich zu normalen Statements.
- Nehmen Sie an, der Freundschaftsgrad zweier Nutzer im sozialen Netzwerk soll geändert werden. Skizzieren Sie hierzu eine Lösung in Java mit JDBC unter Verwendung von *Prepared Statements*. Dabei sollen die Nutzer-IDs *uid1* und *uid2* sowie der neue Freundschaftsgrad *fgradNew* als Parameter übergeben werden können.
- Erstellen Sie eine zusätzliche Java-Methode, um alle Freundschaftsbeziehungen eines Nutzers (Angabe der Namen) auszugeben. Als Parameter soll die *uid* des Nutzers übergeben werden.
- Passen Sie ihre Lösung aus d) so an, dass anstelle der Nutzer-IDs deren Namen (*uname1*, *uname2*) übergeben werden können, d. h. die entsprechenden Nutzer-IDs müssen anhand der übergebenen Namen ermittelt werden. Nehmen Sie an, dass Nutzer eindeutig durch ihren Benutzernamen gekennzeichnet werden.
- Vergleichen Sie die Methoden eines JDBC *ResultSets* mit den Operatoren eines Cursors.

## Aufgabe 3 (JDBC Transaktionen)

Die Transaktionskontrolle mit JDBC erfolgt über die Methoden der Klasse *Connection*.

- Was versteht man unter dem *AutoCommit*-Modus?
- Wie erreicht man die Kapselung mehrerer SQL-Anweisungen in eine Transaktion?
- Wie viele Tupel werden durch das folgende Programmfragment in die Tabelle *Verlag* eingefügt? Die Variable *con* enthalte eine aktive *Connection* zur Datenbank mit *AutoCommit*-Status *true*. Die Variable *pStmt* sei ein *Prepared Statement* zum Einfügen eines Verlages.

```

try {
    pstmt.setInt(1, 123);
    pstmt.setString(2, "Springer");
    pstmt.setString(3, "Berlin");
    pstmt.executeUpdate();
    con.rollback();
    con.setAutoCommit(false);

    pstmt.setInt(1, 124);
    pstmt.setString(2, "Heyne");
    pstmt.setString(3, "München");
    pstmt.executeUpdate();

    pstmt.setInt(1, 123);
    pstmt.setString(2, "Forum");
    pstmt.setString(3, "Leipzig");
    pstmt.executeUpdate();
    con.commit();
} catch(SQLException e) {
    try {
        con.rollback();
        con.close();
    } catch(SQLException e) {}
}

```

### Ausschnitt Datenbankschema „Bibliothek“

```

buch (
    buchid      INT NOT NULL PRIMARY KEY,
    titel       VARCHAR(255),
    isbn        VARCHAR(15),
    auflage     VARCHAR(70),
    jahr        SMALLINT,
    preis       NUMERIC(10,2),
    waehrung    VARCHAR(15),
    signatur    VARCHAR(30),
    verlagsid   INT REFERENCES verlag (verlagsid)
);

verlag (
    verlagsid   INT NOT NULL PRIMARY KEY,
    name        VARCHAR(100) NOT NULL,
    ort         VARCHAR(100)
);

```