Aufgabe 1: SQL

Gegeben sind folgende Relationen aus einer Personalverwaltung:

```
Mitarbeiter (MitarbeiterID), Vorname, Nachname, Vorgesetzter[Mitarbeiter],
↪ AbteilungsID[Abteilung], Telefonnummer, Gehalt)
Abteilung (AbteilungsID, Bezeichnung)
```

(a) Schreiben Sie eine SQL-Anfrage, die *Vor-* und *Nachnamen* der *Mitarbeiter* aller *Abteilungen* mit der Bezeichnung *„Buchhaltung“* ausgibt, absteigend sortiert nach *Mitarbeiter-ID*.

```sql
SELECT Vorname, Nachname
FROM Mitarbeiter m, Abteilung a
WHERE
  m.AbteilungsID = a.AbteilungsID AND
  a.Bezeichung = 'Buchhaltung'
ORDER BY m.MitarbeiterID DESC;
```

(b) Schreiben Sie eine SQL-Anfrage, die die Nachnamen aller Mitarbeiter mit dem Nachnamen ihres jeweiligen direkten Vorgesetzten ausgibt. Mitarbeiter ohne Vorgesetzten sollen in der Ausgabe ebenfalls enthalten sein. In diesem Fall soll der Nachname des Vorgesetzten NULL sein.

```sql
SELECT m.Nachname AS Mitarbeiter, v.Nachname AS Vorgesetzter
FROM Mitarbeiter m LEFT OUTER JOIN Mitarbeiter v
ON m.Vorgesetzter = v.MitarbeiterID;
```

(c) Schreiben Sie eine SQL-Anfrage, die die 10 Abteilungen ausgibt, deren Mitarbeiter das höchste Durchschnittsgehalt haben. Ausgegeben werden sollen der Rang (1 = höchstes Durchschnittsgehalt bis 10 = niedrigstes Durchschnittsgehalt), die Bezeichnung sowie das Durchschnittsgehalt der Abteilung. Gehen Sie davon dass es keine zwei Abteilungen mit gleichem Durchschnittsgehalt gibt. Sie können der Übersichtlichkeit halber Views oder With-Anweisungen verwenden. Verwenden Sie jedoch keine datenbanksystemspezifischen Erweiterungen wie `limit` oder `rownum`.

```sql
CREATE VIEW Durchschnittsgehaelter AS
SELECT Abteilung.AbteilungsID, Bezeichnung,
  AVG (Gehalt) AS Durchschnittsgehalt
FROM Mitarbeiter, Abteilung
WHERE Mitarbeiter.AbteilungsID = Abteilung.AbteilungsID
GROUP BY Abteilung.AbteilungsID, Bezeichnung

SELECT a.Bezeichnung, a.Durchschnittsgehalt, COUNT (*) AS Rang
FROM Durchschnittsgehaelter a, Durchschnittsgehaelter b
WHERE a.Durchschnittsgehalt <= b.Durchschnittsgehalt
GROUP BY a.AbteilungsID, a.Bezeichnung, a.Durchschnittsgehalt
HAVING COUNT (*) <= 10
ORDER BY Rang ASC
```

(d) Schreiben Sie eine SQL-Anfrage, die das Gehalt aller Mitarbeiter aus der Abteilung mit der AbteilungsID 42 um 5% erhöht.

```sql
UPDATE Mitarbeiter
SET Gehalt = 1.05 * Gehalt
WHERE AbteilungsID = 42
```

(e) Alle *Abteilungen* mit Bezeichnung *„Qualitätskontrolle “* sollen zusammen mit den Datensätzen ihrer *Mitarbeiter* gelöscht werden. `ON DELETE CASCADE` ist für keine der Tabellen gesetzt. Schreiben Sie die zum Löschen notwendigen SQL-Anfragen.

```sql
DELETE FROM Mitarbeiter
WHERE AbteilungsID ANY (
  SELECT a.AbteilungsID
  FROM Abteilung a
  WHERE a.Bezeichnung = 'Qualitaetskontrolle'
);

DELETE FROM Abteilung
WHERE Bezeichnung = 'Qualitaetskontrolle';
```

(f) Alle Mitarbeiter sollen mit SQL-Anfragen nach den Telefonnummern anderer Mitarbeiter suchen können. Sie dürfen jedoch das Gehalt der Mitarbeiter nicht sehen können. Erläutern Sie in zwei bis drei Sätzen eine Möglichkeit, wie dies in einem Datenbanksystem realisiert werden kann, ohne die gegebenen Relationen, die Tabellen als abgelegt sind, zu verändern. Sie brauchen hierzu keinen SQL-Code schreiben.

> VIEW Erstellen, die zwar Namen und ID der anderen Mitarbeiter, sowie ihre Telefonnummern enthält (evtl. auch Abteilungsbezeichnung und ID), aber eben nicht das Gehalt: Mitarbeiter arbeiten auf eingeschränkter Sicht
>
> Alternativ mit `GRANT`:
>
> explizit mit `SELECT` die Spalten auswählen, die man lesen können soll (auf nicht angegebene Spalten ist kein Zugriff möglich)

```sql
GRANT SELECT (Vorname, Nachname, Telefonnummer)
ON Mitarbeiter TO alle anderen Mitarbeiter
```

```sql
-- sudo mysql  < Personalverwaltung.sql
DROP DATABASE IF EXISTS Personalverwaltung;
CREATE DATABASE Personalverwaltung;
USE Personalverwaltung;

CREATE TABLE Mitarbeiter(
MitarbeiterID Integer PRIMARY KEY,
Vorname VARCHAR(30),
Nachname VARCHAR(30),
Vorgesetzter Integer REFERENCES Mitarbeiter(MitarbeiterID),
AbteilungsID Integer REFERENCES Abteilung(AbteilungsID),
Telefonnummer Long,
Gehalt Double
);

CREATE TABLE Abteilung(
AbteilungsID Integer PRIMARY KEY,
Bezeichnung VARCHAR(30)
```

```
19  );
20
21  INSERT INTO Abteilung VALUES (1,"Buchhaltung");
22  INSERT INTO Abteilung VALUES (2,'Geschaeftemacher');
23  INSERT INTO Abteilung VALUES (3,"Buchhaltung");
24  INSERT INTO Abteilung VALUES (4, "Qualitaetskontrolle");
25  INSERT INTO Abteilung VALUES (5, "Qualitaetskontrolle");
26
27  INSERT INTO Mitarbeiter VALUES (1,"Hans", 'Meier', 11, 1, 2313432, 12345);
28  INSERT INTO Mitarbeiter VALUES (2,"Fred", 'Maier', 11, 2, 233413432, 1233);
29  INSERT INTO Mitarbeiter VALUES (11,"Hans", 'Mueller', NULL,3, 3432, 1230454);
30  INSERT INTO Mitarbeiter VALUES (3,"Hans", 'Fuchs', 2, 1, 2313344, 2345);
31  INSERT INTO Mitarbeiter VALUES (4,"Fred", 'Hase', 11, 2, 432, 12334);
32  INSERT INTO Mitarbeiter VALUES (12,"Gerd", 'Mueller', NULL,3, 345552, 154);
33  INSERT INTO Mitarbeiter VALUES (5, "Josef", "Huber", 12, 4, 786787, 3443);
34  INSERT INTO Mitarbeiter VALUES (6, "Juergen", "Schmidt", 12, 5, 97854, 6654);
```