ARBEITSBLATT ZU SQL-BEFEHLEN

Die Syntax eines Standard-SELECT-Befehls in Backus-Naur-Form sieht wie folgt aus:

```
SELECT [ALL|DISTINCT]{spalten|*}
FROM tabelle [alias] [tabelle[alias]]...
[WHERE {bedingung|unterabfrage}]
[GROUP BY spalten[HAVING {bedingung|unterabfrage}]]
[ORDER BY spalten[ASC|DESC]...];
```

Aufgabe 1: Gegeben seien die Relationen

Student(<u>MatrNr</u>, Name)
Vorlesung(<u>VorlNr</u>, Titel, PersNr)
Professor(<u>PersNr</u>, Name)

und die Beziehungsrelation

hört(<u>MatrNr</u>, <u>VorlNr</u>)

Was bewirken die folgenden SQL-Anweisungen?

```
SELECT *
FROM Student
```

```
SELECT *
FROM Student
WHERE (MatrNr LIKE "156-*") OR (Name LIKE "Ab*")
```

```
SELECT MatrNr
FROM Student
```

```
SELECT DISTINCT PersNr
FROM Vorlesung
```

```
SELECT Name, MatrNr AS Matrikelnummer
FROM Student
```

```
SELECT Titel, VorlNr
FROM Vorlesung
WHERE PersNr = 12
```

```
SELECT Titel, VorlNr
FROM Vorlesung
ORDER BY PersNr
```

```
SELECT Vorlesung.VorlNr, Vorlesung.Titel, Professor.Name, Professor.PersNr
FROM Professor INNER JOIN Vorlesung ON Professor.PersNr = Vorlesung.PersNr
```

```
SELECT Vorlesung.VorlNr, Vorlesung.Titel, Professor.Name, Professor.PersNr
FROM Professor LEFT OUTER JOIN Vorlesung ON Professor.PersNr = Vorlesung.PersNr
```

```
SELECT Professor.Name, Professor.PersNr
FROM Professor LEFT OUTER JOIN Vorlesung ON Professor.PersNr = Vorlesung.PersNr
WHERE Vorlesung.PersNr IS NULL
```

```
SELECT Professor.Name, Professor.PersNr
FROM Professor
WHERE NOT EXISTS (SELECT * FROM Vorlesung WHERE Vorlesung.PersNr = Professor.PersNr)
```

```
SELECT COUNT(Vorlesung.PersNr) AS Anzahl, Professor.Name, Professor.PersNr
FROM Professor LEFT OUTER JOIN Vorlesung ON Professor.PersNr = Vorlesung.PersNr
GROUP BY Professor.Name, Professor.PersNr
```

Aufgabe 2: Folgende Tabelle sind gegeben:

Abteilung

ID	Name	Leiter
0	Raumfahrt	1
1	Fuhrpark	4
2	Verwaltung	2

arbeitet_an

ID	Mitarbeiter	Projekt
0	1	0
1	1	1
2	2	2
3	5	0
4	0	1
5	0	2

Mitarbeiter

ID	Name	Vorname	Abteilung
0	Müller	Anton	NULL
1	Geiger	Sven	0
2	Schwab	Anita	2
4	Görgens	Margit	1
5	Hurz	Willy	NULL

Projekt

ID	Bezeichner	Abteilung	Verantwortlicher
0	Apollo 13	0	5
1	Challenger	0	4
2	Webseiten	2	0

a) Erstellen Sie ein Entity-Relationship-Diagramm.

b) Geben Sie die SQL-Befehle an, welche nötig sind, um die folgenden Abfragen zu erhalten.

1. Wer ist Leiter der Raumfahrtabteilung?
2. Geben Sie alle Abteilungsleiter aus.
3. Welche Projekte gehören zur Verwaltungsabteilung?
4. Wer ist für das Apollo 13 Projekt verantwortlich?
5. Wer arbeitet am Challenger Projekt?
6. Geben Sie alle Projektverantworlichen aus.
7. Wer arbeitet am Challenger Projekt und ist gleichzeitig Leiter einer Abteilung?
8. Wer arbeitet am Apollo 13 Projekt oder am WebSeiten-Projekt?

c) Geben Sie die aus den Abfragen resultierenden Tabellen an.

```
SELECT * FROM Student
```
Listet die Werte aller Spalten aus der Tabelle Student auf.

```
SELECT MatrNr FROM Student
```
Projektion: Listet die Spalte *MatrNr* der Tabelle *Student* auf.

```
SELECT DISTINCT PersNr FROM Vorlesung
```
Projektion: Listet die vorhandenen, verschiedenen Ausprägungen der Spalte *PersNr* aus der Tabelle *Vorlesung* auf.

```
SELECT Name, MatrNr AS Matrikelnummer FROM Student
```
Die Spalte *MatrNr* heißt in der Ergebnisrelation jetzt *Matrikelnummer*.

```
SELECT Titel, VorlNr FROM Vorlesung WHERE PersNr = 12
```
Selektion: Listet alle Vorlesungen eines Professors auf.

```
SELECT Titel, VorlNr FROM Vorlesung ORDER BY PersNr
```
Projektion mit Gruppierung: Listet alle Vorlesungs-Titel sortiert nach unterrichtenden Professoren auf.

```
SELECT a.VorlNr, a.Titel, b.Name, b.PersNr FROM Professor b
INNER JOIN Vorlesung a ON b.PersNr = a.PersNr
```
Innerer natürlicher Verbund: Listet die Werte der Spalten *VorlNr* und *Titel* aus der Tabelle *Vorlesung* sowie der Spalten *Name* und *Persnr* aus der Tabelle *Professor* für alle Vorlesungen auf.

```
SELECT a.VorlNr, a.Titel, b.Name, b.PersNr FROM Professor b
LEFT OUTER JOIN Vorlesung a ON b.PersNr = a.PersNr
```
Äußerer linker natürlicher Verbund: Listet die Werte der Spalten *VorlNr* und *Titel* aus der Tabelle *Vorlesung* sowie der Spalten *Name* und *PersNr* aus der Tabelle *Professor* für alle Vorlesungen auf. Professoren die keine Vorlesungen halten werden auch mit aufgelistet.

```
SELECT c.Name, c.PersNr
FROM Professor c LEFT OUTER JOIN Vorlesung b ON c.PersNr = b.PersNr
WHERE b.PersNr IS NULL
```
Äußerer linker natürlicher Verbund, Selektion und Projektion: Listet alle Professoren auf, die keine Vorlesungen halten.

```
SELECT a.Name, a.PersNr
FROM Professor a
WHERE NOT EXISTS (SELECT * FROM Vorlesung WHERE PersNr = a.PersNr)
```
Unterabfrage mit *Existenz-Quantor*: Das gleiche mit einer Unterabfrage.

```
SELECT COUNT(b.PersNr) AS Anzahl, a.Name, a.PersNr FROM Professor a
LEFT OUTER JOIN Vorlesung b on a.PersNr = b.PersNr GROUP BY a.Name, a.PersNr
```
Gruppierung, *Aggregation* und *äußerer linker natürlicher Verbund*: Zählt die Anzahl der Vorlesungen pro Professor.
Merke: *COUNT(a.PersNr)* oder *COUNT(*)* wären falsch (Nullwerte sollen nicht mitgezählt werden).

Frage: Wer ist Leiter der Raumfahrtabteilung?
Antwort:

```
mysql> select mitarbeiter.name,mitarbeiter.vorname from
-> mitarbeiter,abteilung where
-> abteilung.name="Raumfahrt" and
-> leiter=mitarbeiter.ID;
+--------+---------+
| name | vorname |
+--------+---------+
| Geiger | Sven |
+--------+---------+
1 row in set (0.01 sec)
```

Frage: Geben Sie alle Abteilungsleiter aus.
Antwort:

```
mysql> select mitarbeiter.name,mitarbeiter.vorname from
-> mitarbeiter,abteilung where
-> leiter=mitarbeiter.id;
+---------+---------+
| name | vorname |
+---------+---------+
| Geiger | Sven |
| Görgens | Margit |
| Schwab | Anita |
+---------+---------+
3 rows in set (0.00 sec)
```

Frage: Welche Projekte gehören zur Verwaltungsabteilung?
Antwort:

```
mysql> select projekt.bezeichner from
-> projekt,abteilung where
-> abteilung.name="Verwaltung" and
-> abteilung.id=projekt.zugeord_abt;
+------------+
| bezeichner |
+------------+
| WebSeiten |
+------------+
1 row in set (0.01 sec)
```

Frage: Wer ist für das Apollo 13 Projekt verantwortlich?
Antwort:

```
mysql> select mitarbeiter.name from
-> mitarbeiter,projekt where
-> projekt.bezeichner="Apollo13" and
-> projekt.verantw_mitarb=mitarbeiter.id;
+------+
| name |
+------+
| Hurz |
+------+
1 row in set (0.00 sec)
```

Frage: Wer arbeitet am Challenger Projekt?
Antwort:

```
mysql> select mitarbeiter.name,mitarbeiter.vorname from
-> mitarbeiter,arbeitet_an,projekt where
-> projekt.bezeichner="Challenger" and
-> projekt.id=arbeitet_an.projekt and
-> arbeitet_an.mitarbeiter=mitarbeiter.id;
+--------+---------+
| name | vorname |
+--------+---------+
| Geiger | Sven |
| Müller | Anton |
+--------+---------+
2 rows in set (0.00 sec)
```

Frage: Geben Sie alle Projektverantworlichen aus.
Antwort:

```
mysql> select mitarbeiter.name,mitarbeiter.vorname from
```

```
-> projekt,mitarbeiter where
-> projekt.verantw_mitarb=mitarbeiter.id;
+---------+---------+
| name | vorname |
+---------+---------+
| Hurz | Willy |
| Görgens | Margit |
| Müller | Anton |
+---------+---------+
3 rows in set (0.00 sec)
```

Frage: Wer arbeitet am Challenger Projekt und ist gleichzeitig Leiter einer Abteilung?
Antwort:

```
mysql> select mitarbeiter.name,mitarbeiter.vorname from
-> abteilung,arbeitet_an,mitarbeiter,projekt where
-> projekt.bezeichner="Challenger" and
-> projekt.id=arbeitet_an.projekt and
-> arbeitet_an.mitarbeiter=abteilung.leiter and
-> arbeitet_an.mitarbeiter=mitarbeiter.id;
+--------+---------+
| name | vorname |
+--------+---------+
| Geiger | Sven |
+--------+---------+
1 row in set (0.00 sec)
```

Frage: Wer arbeitet am Apollo 13 Projekt oder am WebSeiten-Projekt?
Antwort:

```
mysql> select mitarbeiter.name,mitarbeiter.vorname from
-> mitarbeiter,arbeitet_an,projekt where
-> (projekt.bezeichner="Apollo13" or projekt.bezeichner="WebSeiten") and
-> projekt.id=arbeitet_an.projekt and
-> arbeitet_an.mitarbeiter=mitarbeiter.id;
+--------+---------+
| name | vorname |
+--------+---------+
| Geiger | Sven |
| Hurz | Willy |
| Schwab | Anita |
| Müller | Anton |
+--------+---------+
4 rows in set (0.01 sec)
```