


Übungsprotokoll

INSY

	Übungsdatum: KW 36 – 02	Klasse: 3AHIT	Name: Felix Schneider
	Abgabedatum: 13.01.2022	Gruppe: INSY	Note:
Leitung: DI (FH) Alexander MESTL	Mitübende: Clemens Schlipfinger		
Übungsbezeichnung: Arbeiten mit SQL			

Inhaltsverzeichnis:

1	Aufgabenstellung.....	2
2	Theoretische Grundlagen	2
3	Übungsdurchführung	2
3.1	Einloggen, Erstellen, Loslegen	2
3.2	select und zugehörige Klausel (viele)	4
3.2.1	order by, concat(), where	5
3.2.2	like, not, in, between, limit.....	6
3.2.3	join	7
3.2.3.1	Gedankenexperiment.....	8
3.2.4	count, coalesce, min, max	9
3.2.5	group by, having(, max, min), avg	10
3.2.6	Kombinationen	12
3.2.7	Zusatzaufgabe für die treuen LeserInnen.....	14
4	Kommentar.....	14

1 Aufgabenstellung

SQL-Beispiele aus dem Aufgabendokument mit der angefügten Datenbank durchführen.

2 Theoretische Grundlagen

Als Datenbankprogramm wird maria-db verwendet.

3 Übungsdurchführung

3.1 Einloggen, Erstellen, Loslegen

Damit man sich bei der Datenbank einloggen und loslegen kann, benötigt man folgenden Befehl und das Passwort von einem Benutzer, der die richtigen Rechte hat, um mit dieser Datenbank auch sinnvoll arbeiten zu können (notfalls einfach root):

-u gibt den Benutzer an, mithilfe dessen man sich einloggen möchte.

-p gibt an, dass ein Passwort benötigt wird.

```
felix@debian:~$ sudo mysql -u root -p
[sudo] Passwort für felix:
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 30
Server version: 10.5.12-MariaDB-0+deb11u1 Debian 11

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> █
```

Mit „show databases;“ kann man sich alle Datenbanken anzeigen lassen.

```
MariaDB [(none)]> show databases;
+-----+
| Database |
+-----+
| hr        |
| information_schema |
| mydb      |
| mysql     |
| performance_schema |
+-----+
5 rows in set (0.000 sec)
```

Mit „show tables;“ kann man sich alle Tabellen einer Datenbank anzeigen lassen. Damit dies funktioniert muss auch eine Datenbank ausgewählt sein (folgt unten!).

```
MariaDB [hr]> show tables;
+-----+
| Tables_in_hr |
+-----+
| COUNTRIES    |
| DEPARTMENTS  |
| EMPLOYEES    |
| LOCATIONS    |
+-----+
4 rows in set (0.000 sec)
```

Damit man eine neue leere Datenbank erstellt, verwendet man „create database <Name>;“.

```
MariaDB [(none)]> create database hr1;
Query OK, 1 row affected (0.001 sec)
```

Wie oben angekündigt, kann man mit „use <Name>;“ die Datenbank auswählen.

```
MariaDB [(none)]> use hr1;
Database changed
```

Um eine Datenbank zu löschen, braucht man „drop database <Name>;“.

```
MariaDB [(none)]> drop database hr1;
Query OK, 4 rows affected (0.047 sec)
```

Mit dem „source <Datei>;“-Befehl kann man sich eine Datenbank importieren (diese kann bereits vollkommen funktionsfähig sein, sprich nicht leer).

```
MariaDB [hr]> source /mnt/sweets.sql
Query OK, 0 rows affected (0.000 sec)
```

```
Query OK, 0 rows affected (0.000 sec)
```

```
Query OK, 0 rows affected (0.000 sec)
```

Um sich alle Spalten einer Tabelle einer Datenbank anzeigen zu lassen, muss man 1. die Datenbank use und 2. diesen Befehl ausführen (EMPLOYEES ist hier die Datenbank):

```
MariaDB [hr]> show columns from EMPLOYEES;
```

Field	Type	Null	Key	Default	Extra
EMPLOYEE_ID	int(11)	NO	PRI	NULL	auto_increment
FIRST_NAME	varchar(20)	NO		NULL	
LAST_NAME	varchar(25)	NO		NULL	
EMAIL	varchar(25)	NO	UNI	NULL	
PHONE_NUMBER	varchar(20)	YES		NULL	
HIRE_DATE	date	NO		NULL	
JOB_ID	varchar(10)	NO		NULL	
SALARY	decimal(8,2)	YES		NULL	
COMMISSION_PCT	decimal(2,2)	YES		NULL	
MANAGER_ID	int(11)	YES	MUL	NULL	
DEPARTMENT_ID	int(11)	YES	MUL	NULL	

11 rows in set (0.001 sec)

3.2 select und zugehörige Klausel (viele)

Mit dem select-Befehl kann man sich jegliche Daten von einer Datenbank ausgeben lassen, die man will. Dieser Befehl gehört zur DQL (Data Query Language) und umfasst die meisten Klausel der gesamten SQL (denke ich).

Database changed

```
MariaDB [hr]> select first_name, last_name, salary, email from EMPLOYEES order by last_name, first_name;
```

first_name	last_name	salary	email
Ellen	Abel	11000.00	EABEL
Sundar	Ande	6400.00	SANDE
Mozhe	Atkinson	2800.00	MATKINSO
David	Austin	4800.00	DAUSTIN
Hermann	Baer	10000.00	HBAER
Shelley	Beider	2900.00	SBEIDR

3.2.1 order by, concat(), where

Mithilfe der „order by“ Klausel kann man sich die Ausgabe nach einer oder mehreren Kriterien sortieren lassen (Zahlen oder Alphabet).

```
MariaDB [hr]> select * from EMPLOYEES order by last_name, first_name;
```

EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EMAIL	PHONE_NUMBER	HIRE_DATE	JOB_ID	SALARY	COMMISSION_PCT
174	Ellen	Abel	EABEL	011.44.1644.429267	2004-05-11	SA_REP	11000.00	0
166	Sundar	Ande	SANDE	011.44.1346.629268	2008-03-24	SA_REP	6400.00	0
130	Mozhe	Atkinson	MATKINSO	650.124.6234	2005-10-30	ST_CLERK	2800.00	N
105	David	Austin	DAUSTIN	590.423.4569	2005-06-25	IT_PROG	4800.00	N
204	Hermann	Baer	HBAER	515.123.8888	2002-06-07	PR_REP	10000.00	N
116	Shelli	Baida	SBAIDA	515.127.4563	2005-12-24	PU_CLERK	2900.00	N
167	Amit	Banda	ABANDA	011.44.1346.729268	2008-04-21	SA_REP	6200.00	0
172	Elizabeth	Bates	EBATES	011.44.1343.529268	2007-03-24	SA_REP	7300.00	0
192	Sarah	Bell	SBELL	650.501.1876	2004-02-04	SH_CLERK	4000.00	N
151	David	Bernstein	DBERNSTE	011.44.1344.345268	2005-03-24	SA_REP	9500.00	0
129	Laura	Bissot	LBISSOT	650.124.5234	2005-08-20	ST_CLERK	3300.00	N
169	Harrison	Bloom	HBLLOOM	011.44.1343.829268	2006-03-23	SA_REP	10000.00	0

Man kann absteigend und aufsteigend sortieren. Standardmäßig sortiert man immer aufsteigend (0-9, A-Z), wenn man dies erzwingen will, schreibt man ASC hinter die „order by“-Bedingung. Wenn man absteigend sortieren will (9-0, Z-A), schreibt man DESC dahinter.

```
MariaDB [hr]> select first_name, last_name, salary from EMPLOYEES where job_id = 'IT_PROG' and salary > 5000 order by salary desc;
```

first_name	last_name	salary
Alexander	Hunold	9000.00
Bruce	Ernst	6000.00

2 rows in set (0.000 sec)

Mit der „concat()“-Methode kann man sich Daten in einen String zusammenfassen. Mit dieser Technik könnte man auch sinnvolle Sätze bilden.

```
MariaDB [hr]> select concat(first_name, last_name, salary, email) as employee_data from EMPLOYEES order by last_name, first_name;
```

employee_data
EllenAbel11000.00EABEL
SundarAnde6400.00SANDE
MozheAtkinson2800.00MATKINSO
DavidAustin4800.00DAUSTIN
HermannBaer10000.00HBAER
ShelliBaida2900.00SBAIDA
AmitBanda6200.00ABANDA
ElizabethBates7300.00EBATES
SarahBell4000.00SBELL

„Das „where“ in SQL ist wie das „if“ in C#“ sagte einmal ein weiser Mann (nicht ich 😊). Sprich: Mithilfe der „where“ Klausel kann man eine Bedingung festlegen. Zum Beispiel kann man nur Employees ausgeben, die in den Programmierer-Abteil arbeiten gehen:

```
MariaDB [hr]> select * from EMPLOYEES where job_id = 'IT_PROG';
```

EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EMAIL	PHONE_NUMBER	HIRE_DATE	JOB_ID	SALARY	COMMISSION_PCT	MANAGER_ID	DEPARTMENT_ID
103	Alexander	Hunold	AHUNOLD	590.423.4567	2006-01-03	IT_PROG	9000.00	NULL	102	60
104	Bruce	Ernst	BERNST	590.423.4568	2007-05-21	IT_PROG	6000.00	NULL	103	60
105	David	Austin	DAUSTIN	590.423.4569	2005-06-25	IT_PROG	4800.00	NULL	103	60
106	Valli	Pataballa	VPATABAL	590.423.4560	2006-02-05	IT_PROG	4800.00	NULL	103	60
107	Diana	Lorentz	DLARENTZ	590.423.5567	2007-02-07	IT_PROG	4200.00	NULL	103	60

Natürlich kann man auch mehrere Bedingungen festlegen. Genau wie in C# gibt es && → AND und || → OR.

```
MariaDB [hr]> select first_name, last_name, salary from EMPLOYEES where job_id = 'IT_PROG' and salary > 5000;
```

first_name	last_name	salary
Alexander	Hunold	9000.00
Bruce	Ernst	6000.00

2 rows in set (0.000 sec)

Diesen Befehl sollten Sie nun bereits interpretieren können:

```
select first_name, last_name, salary from EMPLOYEES where job_id = 'IT_PROG' and salary > 4000 order by last_name, first_name;
```

3.2.2 like, not, in, between, limit

Mithilfe von „like“ können Sie nach bestimmten Mustern mit der „where“-Klausel suchen. Das Prozentzeichen steht hierbei für beliebig viele Buchstaben oder Zahlen.

```
MariaDB [hr]> select first_name, last_name, salary from EMPLOYEES where last_name like 'H%';
```

first_name	last_name	salary
Alexander	Hunold	9000.00
Guy	Himuro	2600.00
Peter	Hall	9000.00
Alyssa	Hutton	8800.00
Michael	Hartstein	13000.00
Shelley	Higgins	12008.00

6 rows in set (0.000 sec)

NOT verneint eine Klausel. Ob NOT NOT die Klausel nicht verneint? 😊

```
MariaDB [hr]> select first_name, last_name, salary from EMPLOYEES where last_name like 'H%' and last_name not like '%n';
```

first_name	last_name	salary
Alexander	Hunold	9000.00
Guy	Himuro	2600.00
Peter	Hall	9000.00
Shelley	Higgins	12008.00

4 rows in set (0.000 sec)

Mithilfe von IN können Sie nach verschiedenen Zeichenketten filtern. Sozusagen ein „where“ mit ganz vielen AND nur kürzer!

```
MariaDB [hr]> select last_name, first_name, job_ID from EMPLOYEES where JOB_ID IN ('SA_MAN', 'SA_REP', 'ST_CLERK');
```

last_name	first_name	job_ID
Nayer	Julia	ST_CLERK
Mikkilineni	Irene	ST_CLERK
Landry	James	ST_CLERK
Markle	Steven	ST_CLERK

BETWEEN – AND schreibt nur Zahlen die zwischen (inklusive, inklusive) einem Bereich sind. Dass die beiden angegebenen Zahlen inklusive sind, macht laut Definition der Wörter between und and eigentlich keinen Sinn. Jedoch macht es ITlern den Job wahrscheinlich einfacher, weil man nicht 999999 sondern 1000000 schreiben muss...

```
MariaDB [hr]> select first_name, last_name, salary from EMPLOYEES where SALARY between 10001 and 16999;
```

first_name	last_name	salary
Nancy	Greenberg	12008.00
Den	Raphaely	11000.00
John	Russell	14000.00
Karen	Partners	13500.00
Alberto	Errazuriz	12000.00
Gerald	Cambrault	11000.00
Eleni	Zlotkey	10500.00
Clara	Vishney	10500.00
Lisa	Ozer	11500.00
Ellen	Abel	11000.00
Michael	Hartstein	13000.00
Shelley	Higgins	12008.00

12 rows in set (0.000 sec)

Mit LIMIT kann man einen Maximalanzahl an Reihen des Outputs festlegen (in manchen Datenbankprogrammen heißt es FETCH).

```
MariaDB [hr]> select first_name, last_name from EMPLOYEES order by last_name asc, first_name desc limit 10;
```

first_name	last_name
Ellen	Abel
Sundar	Ande
Mozhe	Atkinson
David	Austin
Hermann	Baer
Shelli	Baida
Amit	Banda
Elizabeth	Bates
Sarah	Bell
David	Bernstein

10 rows in set (0.000 sec)

3.2.3 join

Der INNER JOIN fügt zwei Tabellen nach einer Bedingung zusammen, sodass man auf Attribute von beiden Tabellen zugreifen kann.

```
MariaDB [hr]> select E.first_name, E.last_name, D.department_name from EMPLOYEES E join DEPARTMENTS D on E.department_id = D.department_id;
```

first_name	last_name	department_name
Shelley	Higgins	Accounting
William	Gietz	Accounting
Jennifer	Whalen	Administration
Steven	King	Executive
Neena	Kochhar	Executive
Lex	De Haan	Executive
Nancy	Greenberg	Finance
Daniel	Faviet	Finance
John	Chen	Finance
Ismael	Sciarra	Finance
Jose Manuel	Urman	Finance

5 rows in set (0.000 sec)

```
MariaDB [hr]> select E.first_name, E.last_name, D.department_name from EMPLOYEES E join DEPARTMENTS D on E.department_id = D.department_id where D.department_name = "IT";
```

first_name	last_name	department_name
Alexander	Hunold	IT
Bruce	Ernst	IT
David	Austin	IT
Valli	Pataballa	IT
Diana	Lorentz	IT

5 rows in set (0.000 sec)

Der NATURAL JOIN ist wie der INNER JOIN, nur dass man die ON Bedingung weglässt, weil man einfach die Schlüssel vergleicht(, was in den meisten Fällen am meisten Sinn macht).

```
MariaDB [hr]> select first_name, last_name, department_name from EMPLOYEES natural join DEPARTMENTS where department_name = "IT";
```

first_name	last_name	department_name
Alexander	Hunold	IT
Bruce	Ernst	IT
David	Austin	IT
Valli	Pataballa	IT
Diana	Lorentz	IT

5 rows in set (0.000 sec)

Man kann auch geJOINte Tabellen JOINen (REKURSION)

```
MariaDB [hr]> select E.first_name, E.last_name, D.department_name, C.country_name from EMPLOYEES E join DEPARTMENTS D on E.department_id = D.department_id join LOCATIONS L on D.location_id = L.location_id join COUNTRIES C on L.country_id = C.country_id where D.department_name = "IT";
```

first_name	last_name	department_name	country_name
Alexander	Hunold	IT	United States of America
Bruce	Ernst	IT	United States of America
David	Austin	IT	United States of America
Valli	Pataballa	IT	United States of America
Diana	Lorentz	IT	United States of America

5 rows in set (0.000 sec)

3.2.3.1 Gedankenexperiment

Wenn man obiges (letztes) Beispiel ohne der „where“-Klausel ausführt, werden insgesamt 106 Reihen, sprich Employees ausgegeben.

```
+-----+
106 rows in set (0.009 sec)
```

Insgesamt gibt es allerdings 107 Employees...

```
+-----+
107 rows in set (0.001 sec)
```

Warum werden bei dem JOIN also nur 106 Employees ausgegeben?

Antwort: Ein Employee hat keine Department_id, weil dieser Employee wahrscheinlich Chef ist, weshalb der INNER JOIN diesen Employee nicht mit in die Ausgabe packt.

Jack	Livingston	80
Kimberely	Grant	NULL
Charles	Johnson	80

Ein LEFT JOIN nimmt alle Werte der linken Tabelle (auch wenn diese NULL sind) und vergleicht diese mit den Werten der rechten Tabelle.

```
MariaDB [hr]> select e.first_name, e.last_name, e.department_id as Employee_Department_ID, d.department_id as Departments_Department_ID, d.department_name from EMPLOYEES e left join DEPARTMENTS d on e.department_id = d.department_id;
+-----+
| first_name | last_name | Employee_Department_ID | Departments_Department_ID | department_name |
+-----+
| Steven     | King     | 90                     | 90                         | Executive       |
| Neena      | Kochhar  | 90                     | 90                         | Executive       |
| Lex        | De Haan  | 90                     | 90                         | Executive       |
| Jack       | Livingston | 80                     | 80                         | Sales           |
| Kimberely  | Grant    | NULL                   | NULL                       | NULL            |
| Charles    | Johnson  | 80                     | 80                         | Sales           |
+-----+

MariaDB [hr]> select E.first_name, E.last_name, D.department_name, C.country_name from EMPLOYEES E left join DEPARTMENTS D on E.department_id = D.department_id left join LOCATIONS L on D.location_id = L.location_id left join COUNTRIES C on L.country_id = C.country_id;
+-----+
| first_name | last_name | department_name | country_name |
+-----+
| Jack       | Livingston | Sales           | United Kingdom |
| Kimberely  | Grant    | NULL            | NULL          |
| Charles    | Johnson  | Sales           | United Kingdom |
+-----+

MariaDB [hr]> select E.first_name, E.last_name, D.department_name, C.country_name from EMPLOYEES E left join DEPARTMENTS D on E.department_id = D.department_id left join LOCATIONS L on D.location_id = L.location_id left join COUNTRIES C on L.country_id = C.country_id where E.department_id is NULL;
+-----+
| first_name | last_name | department_name | country_name |
+-----+
| Kimberely  | Grant    | NULL            | NULL          |
+-----+
1 row in set (0.000 sec)
```

Ein RIGHT JOIN funktioniert genau anders herum. Lustig ist zu erwähnen, dass LEFT und RIGHT wirklich links und rechts entsprechen, nicht 1 und 2 oder so...

3.2.4 count, coalesce, min, max

Mit COUNT() kann man Einträge zählen.

```
MariaDB [hr]> select count(last_name) from EMPLOYEES;
+-----+
| count(last_name) |
+-----+
|                107 |
+-----+
1 row in set (0.004 sec)
```

Mittels COALESCE() kann man NULLWERTE ersetzen, sodass immer die angegebene Zahl dortsteht.

```
MariaDB [hr]> select first_name, last_name, COALESCE(department_id, 0) from EMPLOYEES;
+-----+-----+-----+
| first_name | last_name | department_id |
+-----+-----+-----+
| Jack      | Livingston | 80            |
| Kimberly  | Grant      | 0             |
| Charles   | Johnson    | 80            |
| Winston   | Taylor     | 50            |
```

Dies betrifft allerdings NUR die Ausgabe:

```
MariaDB [hr]> select e.first_name, e.last_name, coalesce(e.department_id, 9999), d.department_name from EMPLOYEES e join DEPARTMENTS d on coalesce(e.department_id, 60) = d.department_id;
+-----+-----+-----+-----+
| first_name | last_name | coalesce(e.department_id, 9999) | department_name |
+-----+-----+-----+-----+
| Jack      | Livingston | 80                               | Sales           |
| Kimberly  | Grant      | 9999                             | IT              |
| Charles   | Johnson    | 80                               | Sales           |
```

Mit MIN() bzw. MAX() kann man das Minimum bzw. das Maximum einer Zahl ermitteln.

```
MariaDB [hr]> select max(salary) from EMPLOYEES;
+-----+
| max(salary) |
+-----+
|      24000.00 |
+-----+
1 row in set (0.009 sec)
```

```
MariaDB [hr]> select min(salary) from EMPLOYEES;
+-----+
| min(salary) |
+-----+
|      2100.00 |
+-----+
1 row in set (0.000 sec)
```

```
ERROR 1054 (42S22): Unknown column 'salary' in 'field list'
MariaDB [hr]> select max(salary) from EMPLOYEES where department_id = 60;
+-----+
| max(salary) |
+-----+
|      9000.00 |
+-----+
1 row in set (0.004 sec)
```

3.2.5 group by, having(, max, min), avg

Mit GROUP BY kann man bestimmte Einträge gruppieren, sodass man zum Beispiel den maximalen bzw. minimalen Gehalt in jeder Abteilung berechnen kann:

```
MariaDB [hr]> select max(salary), d.department_name from EMPLOYEES natural join DEPARTMENTS d group by department_id;
+-----+-----+
| max(salary) | department_name |
+-----+-----+
|      4400.00 | Administration  |
|     13000.00 | Marketing       |
|     11000.00 | Purchasing      |
|      6500.00 | Human Resources |
|      8200.00 | Shipping        |
|      9000.00 | IT              |
|     10000.00 | Public Relations|
|     14000.00 | Sales           |
|     24000.00 | Executive       |
|     12008.00 | Finance         |
|     12008.00 | Accounting      |
+-----+-----+
11 rows in set (0.001 sec)
```

```
MariaDB [hr]> select min(salary), d.department_name from EMPLOYEES natural join DEPARTMENTS d group by department_id;
+-----+-----+
| min(salary) | department_name |
+-----+-----+
|      4400.00 | Administration  |
|      6000.00 | Marketing       |
|      2500.00 | Purchasing      |
|      6500.00 | Human Resources |
|      2100.00 | Shipping        |
|      4200.00 | IT              |
|     10000.00 | Public Relations|
|      6100.00 | Sales           |
|     17000.00 | Executive       |
|      6900.00 | Finance         |
|      8300.00 | Accounting      |
+-----+-----+
11 rows in set (0.001 sec)
```

```
MariaDB [hr]> select d.department_name, min(salary) as Minimum, max(salary) as Maximum from EMPLOYEES natural join DEPARTMENTS d group by department_id;
```

department_name	Minimum	Maximum
Administration	4400.00	4400.00
Marketing	6000.00	13000.00
Purchasing	2500.00	11000.00
Human Resources	6500.00	6500.00
Shipping	2100.00	8200.00
IT	4200.00	9000.00

Mittels AVG kann man den Durchschnitt einer Zahl berechnen:

```
MariaDB [hr]> select d.department_name as Abteilung, min(salary) as Minimum, max(salary) as Maximum, avg(salary) as Durchschnitt from EMPLOYEES natural left join DEPARTMENTS d group by department_id order by Minimum desc, Maximum desc, Durchschnitt desc;
```

Abteilung	Minimum	Maximum	Durchschnitt
Executive	17000.00	24000.00	19333.333333
Public Relations	10000.00	10000.00	10000.000000
Accounting	8300.00	12008.00	10154.000000
NULL	7000.00	7000.00	7000.000000
Finance	6900.00	12008.00	8601.333333
Human Resources	6500.00	6500.00	6500.000000
Sales	6100.00	14000.00	8955.882353
Marketing	6000.00	13000.00	9500.000000
Administration	4400.00	4400.00	4400.000000
IT	4200.00	9000.00	5760.000000
Purchasing	2500.00	11000.00	4150.000000
Shipping	2100.00	8200.00	3475.555556

12 rows in set (0.001 sec)

„Having ist das Where von Group By“ sagte einmal ein weiser Mann (diesmal wars unser Herr Professor...). Diese Aussage beschreibt Having so einfach, wie es ist. Will man nur bestimmte Einträge gruppieren, so kann man dies mit HAVING. SQL benötigt hier wahrscheinlich einen anderen Namen/Befehl, damit es unterscheiden kann, ob man nun die Gruppierung oder die Ausgabe beschränken will.

```
MariaDB [hr]> select count(*) as Anzahl, department_id from EMPLOYEES group by department_id having Anzahl > 5;
```

Anzahl	department_id
6	30
45	50
34	80
6	100

4 rows in set (0.000 sec)

```
MariaDB [hr]> select count(*) as Anzahl, d.department_name from EMPLOYEES e natural left join DEPARTMENTS d where not d.department_name = 'IT' group by department_id having Anzahl > 4;
```

Anzahl	department_name
6	Purchasing
45	Shipping
34	Sales
6	Finance

4 rows in set (0.000 sec)

3.2.6 Kombinationen

Um die Anzahl an Mitarbeiter je Abteilung auszugeben, muss man COUNT() und JOIN verwenden.

```
MariaDB [hr]> select count(*), d.department_name from EMPLOYEES natural left join DEPARTMENTS d
group by department_id;
+-----+-----+
| count(*) | department_name |
+-----+-----+
| 1 | NULL |
| 1 | Administration |
| 2 | Marketing |
| 6 | Purchasing |
| 1 | Human Resources |
| 45 | Shipping |
| 5 | IT |
| 1 | Public Relations |
| 34 | Sales |
| 3 | Executive |
| 6 | Finance |
| 2 | Accounting |
+-----+-----+
12 rows in set (0.000 sec)
```

Manchmal ist es notwendig einen SELECT Befehl in einem SELECT Befehl durchzuführen, wenn nicht sogar rekursiv noch mehr SELECT Befehle (komplexere Abfragen). Diese Art von Befehl nennt man dann SUBSELECT:

```
MariaDB [hr]> select department_name from DEPARTMENTS where department_id = (select department_id
d from EMPLOYEES group by department_id order by count(*) desc limit 1);
+-----+
| department_name |
+-----+
| Shipping |
+-----+
1 row in set (0.001 sec)
```

```
MariaDB [hr]> select country_name from COUNTRIES where country_id = (select country_id from LOCA
TIONS where location_id = (select location_id from DEPARTMENTS where department_id = (select dep
artment_id from EMPLOYEES group by department_id order by count(*) desc limit 1)));
+-----+
| country_name |
+-----+
| United States of America |
+-----+
```

```
MariaDB [hr]> select e.first_name, e.last_name, e.salary, d.department_name from EMPLOYEES e nat
ural join DEPARTMENTS d group by department_id order by max(salary) desc;
+-----+-----+-----+-----+
| first_name | last_name | salary | department_name |
+-----+-----+-----+-----+
| Steven | King | 24000.00 | Executive |
| John | Russell | 14000.00 | Sales |
| Michael | Hartstein | 13000.00 | Marketing |
| Nancy | Greenberg | 12008.00 | Finance |
| Shelley | Higgins | 12008.00 | Accounting |
| Den | Raphaely | 11000.00 | Purchasing |
| Hermann | Baer | 10000.00 | Public Relations |
| Alexander | Hunold | 9000.00 | IT |
| Matthew | Weiss | 8000.00 | Shipping |
| Susan | Mavris | 6500.00 | Human Resources |
| Jennifer | Whalen | 4400.00 | Administration |
+-----+-----+-----+-----+
11 rows in set (0.001 sec)
```

```
MariaDB [hr]> select e.first_name, e.last_name, e.salary from (select max(salary) as max_sal, de
partment_id from EMPLOYEES group by department_id) sub join EMPLOYEES e on e.department_id = sub
.department_id and e.salary = sub.max_sal;
```

first_name	last_name	salary
Steven	King	24000.00
Alexander	Hunold	9000.00
Nancy	Greenberg	12008.00
Den	Raphaely	11000.00
Adam	Fripp	8200.00
John	Russell	14000.00
Jennifer	Whalen	4400.00
Michael	Hartstein	13000.00
Susan	Mavris	6500.00
Hermann	Baer	10000.00
Shelley	Higgins	12008.00

11 rows in set (0.001 sec)

Das Schlüsselwort DISTINCT schließt doppelte Einträge aus.

```
MariaDB [hr]> select distinct job_id from EMPLOYEES;
```

job_id
AD_PRES
AD_VP
IT_PROG
FI_MGR
FI_ACCOUNT
PU_MAN
PU_CLERK
ST_MAN
ST_CLERK
SA_MAN
SA_REP
SH_CLERK
AD_ASST
MK_MAN
MK_REP
HR_REP
PR_REP
AC_MGR
AC_ACCOUNT

19 rows in set (0.001 sec)

Alle Befehle lassen sich miteinander kombinieren, um jedes mögliche Ergebnis zu erreichen, welches man möchte (Voraussetzung: die Datenbank bietet dieses Ergebnis).

```
MariaDB [hr]> select count(*), c.country_name from DEPARTMENTS d natural join LOCATIONS l natural
join COUNTRIES c where department_name not like '%Sales%' group by c.country_id having count(*)
> 3;
+-----+-----+
| count(*) | country_name |
+-----+-----+
|      21 | United States of America |
+-----+-----+
1 row in set (0.001 sec)
```

3.2.7 Zusatzaufgabe für die treuen LeserInnen

Geben Sie für jedes Land aus, wie viele Abteilungen es dort gibt! Wenn in einem Land 0 Abteilungen sind, soll 0 ausgegeben werden (bei Lösungsvorschlägen bitte melden!!)

Die Lösung des 1. Satzes ist aus Sicherheitsgründen verkehrt herum (NICHT SCHUMMEL!

😊):

```
MariaDB [hr]> select c.country_name, count(*) from DEPARTMENTS d join LOCATIONS l using(location_id) join COUNTRIES c using(country_id) group by c.country_name;
+-----+-----+
| country_name | count(*) |
+-----+-----+
| Canada       | 1         |
| Germany      | 1         |
| United Kingdom | 2         |
| United States of America | 23        |
+-----+-----+
4 rows in set (0.009 sec)
```

4 Kommentar

Dieses Protokoll beschäftigt sich hauptsächlich mit DQL (Data Query Language).