

## Einfache Datenbankabfragen

Melde dich in deinem InstaHub (*hubname.instahub.org*) als admin an. Bearbeite anschließend die Aufgaben und notiere die gesuchten SQL-Abfragen in deinem Heft bzw. einem Textdokument.

### Projektion

1. Zeige alle Einträge der Tabelle *users* an.

`SELECT * FROM users`

2. Gib alle Benutzernamen (*username*) aus.

`SELECT username FROM users`

3. Jeder registrierte Nutzer besitzt bestimmte Rechte. Hierfür werden Rollen zugewiesen.  
Welche Rollen gibt es?

`SELECT role FROM users` (genauer s. Aufgabe 9)

Es gibt die Rollen *user* und *dba* (Datenbankadmin)

4. Aus welchen Ländern stammen die Mitglieder?

Gib hierfür neben dem Land auch den jeweiligen Namen (*name*) aus.

`SELECT country FROM users`

### Sortierung – ORDER BY

In vielen Fällen müssen die ausgegebenen Daten auf eine bestimmte Weise sortiert werden. Dies könnte beispielsweise in aufsteigender oder absteigender Reihenfolge oder auf der Grundlage eines Zahlenwerts oder Textwerts geschehen. In solchen Fällen kann man das Schlüsselwort `ORDER BY` einsetzen.

```
SELECT      Spalte1, Spalte2, ...
FROM        Tabellenname
ORDER BY    Spaltenname [ASC | DESC]
```

Optionale Angaben sind in eckige Klammern `[]` gefasst. Das Zeichen `|` steht für *oder*, d.h. man kann zwischen folgenden Möglichkeiten der Sortierung wählen:

- `ASC` bedeutet, dass die Ergebnisse in **aufsteigender** Reihenfolge angezeigt werden.
- `DESC` sortiert in **absteigender** Reihenfolge.

Wird auf die Angabe der Sortierung verzichtet, so wird die Voreinstellung `ASC` verwendet.

[Hier](#) ist eine ausführliche Erklärung mit Beispielen zu finden.

5. Ordne alle Mitglieder nach ihrem Benutzernamen in alphabetischer Reihenfolge.

`SELECT * FROM users`

`ORDER BY username`

6. Ordne die Mitglieder nach deren Größe. Gib hierzu den Benutzernamen, Namen und die Größe aus und sortiere so, dass das größte Mitglied die Liste anführt.

`SELECT username, name, centimeters FROM users`

`ORDER BY centimeters DESC`

7. Ordne die Mitglieder aufsteigend nach ihrem Geburtsdatum. Wer und wie alt ist der jüngste Nutzer?

```
SELECT * FROM users  
ORDER BY birthday
```

8. Kehre die Reihenfolge der Tabelle *users* um.

```
SELECT * FROM users  
ORDER BY id DESC
```

#### Vermeidung von Duplikaten – DISTINCT

In einer Datenbanktabelle enthält eine Spalte oftmals viele gleiche Werte – z.B. die Spalte *role*, welche die Benutzerrollen speichert. Mit **DISTINCT** werden ausschließlich unterschiedliche Werte angezeigt.

```
SELECT DISTINCT Spalte1, Spalte2, ...  
FROM Tabellenname
```

9. Gib jede Benutzerrolle nur einmal aus. Welche Benutzerrolle hat *admin*?

```
SELECT DISTINCT role  
FROM users
```

10. Aus welchen unterschiedlichen Wohnorten stammen die Mitglieder?

```
SELECT DISTINCT city  
FROM users
```

11. Welche Werte werden in der Spalte *is\_active* gespeichert? Was könnte dies bedeuten?

```
SELECT DISTINCT is_active  
FROM users
```

#### Beschränkung der Zeilen der Ergebnistabelle – LIMIT

Wenn das Netzwerk sehr langsam ist dauert es eine Weile bis alle Mitglieder angezeigt werden.

Mit dem **LIMIT** Befehl kann festgelegt werden, wie viele Datensätze ausgegeben werden sollen. Der Befehl steht immer am Ende der SQL-Abfrage.

**Beispiel:** Anzeige der ersten 25 Benutzernamen

```
SELECT username FROM users LIMIT 25
```

12. Zeige nur 3 Mitglieder an.

```
SELECT * FROM users  
LIMIT 3
```

13. Zeige nur die 4 jüngsten Mitglieder an.

```
SELECT * FROM users  
ORDER BY birthday  
LIMIT 4
```

14. Zeige nur die 5 größten Mitglieder an.

```
SELECT * FROM users  
ORDER BY centimeters DESC  
LIMIT 5
```