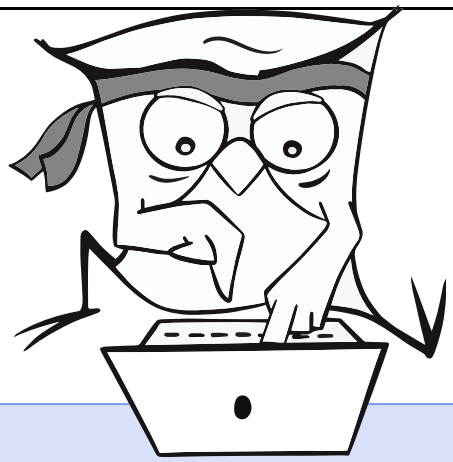
Datenbanken - Nutzung mit der Sprache „SQL"

Justus A. Weber

Was ist SQL?

SQL steht für Structured Query Language. Das ist eine Sprache, mit der man Datenbanken erstellen, verwalten und abfragen kann. Eine Datenbank ist im Grunde eine große digitale Tabelle, in der Informationen (z. B. über Schüler, Bücher oder Filme) gespeichert sind.



Warum ist SQL wichtig?

n der heutigen digitalen Welt gibt es fast überall Datenbanken: in Online-Shops, sozialen Netzwerken oder Apps. Ohne SQL könnten wir diese Daten nicht sinnvoll nutzen. Mit SQL kannst du z. B.:

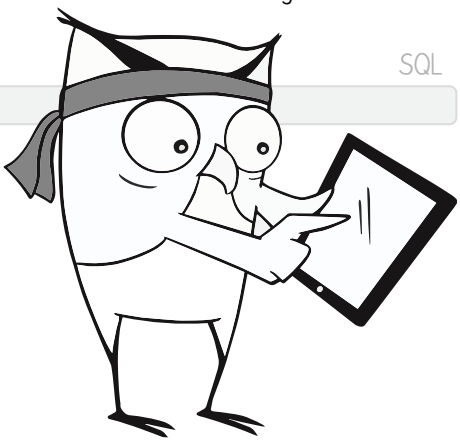
- Informationen in einer
- Datenbank suchen („Welche Filme sind ab 12 freigegeben?")
- Daten hinzufügen („Einen neuen Schüler in die Liste eintragen")
- Daten ändern („Die Telefonnummer eines Schülers aktualisieren")
- Daten löschen („Einen abgelaufenen Eintrag entfernen")

Wie funktioniert SQL?

SQL besteht aus einfachen Befehlen, um mit der Datenbank zu sprechen. Ein paar Beispiele:

- **SELECT** – Um Daten aus der Datenbank zu lesen
- **INSERT** – Um neue Daten hinzuzufügen
- **UPDATE** – Um Daten zu ändern
- **DELETE** – Um Daten zu löschen

Beispiel:
Ein einfacher SQL-Befehl, um in einer Schuldatenbank alle Schüler der Klasse 8A anzuzeigen, sieht folgendermaßen aus:



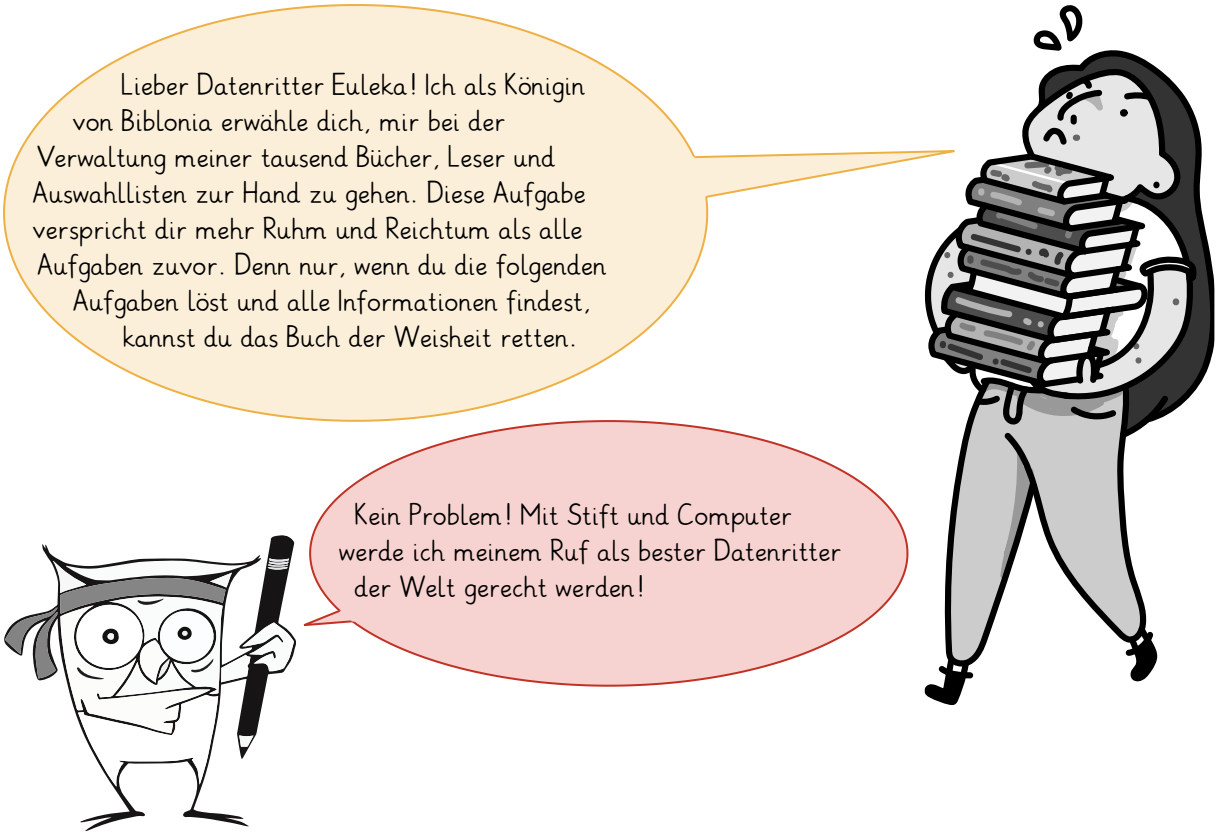
SQL

```sql
SELECT * FROM Schüler WHERE Klasse = "8A"
```

Auswahl aller Schüler der Klasse 8A

Übungssache

SQL zu lernen geht am besten, indem man es übt. Auf der nächsten Seite findest du eine kurze Geschichte mit Beispielen und danach darfst du selbst ran.



Justus A. Weber

1 Die Königin sagt: „Finde alle Bücher in meiner Bibliothek!"
Kein Problem – es wird Zeit für den **SELECT**-Zauber:

SQL

```sql
SELECT * FROM Buecher
```

Alle Bücher erscheinen auf magische Art in einer Tabelle.

2 „Zeig mir nur die Titel und das Erscheinungsjahr der Bücher", fordert die Königin.
Euleka weiß sofort, dass nur eine Projektion hier weiterhelfen kann.

SQL

```sql
SELECT Titel, Erscheinungsjahr FROM Buecher
```

Der Zauber zeigt nur auf die wichtigsten Spalten. Die Tabelle wird kleiner und übersichtlicher.

3 „Wie viele Bücher habe ich insgesamt?", wundert sich die Königin weiter.

SQL

```sql
SELECT COUNT(*) FROM Buecher
```

Der COUNT-Zauber zeigt die Gesamtzahl an. „Wow, 1234 Bücher!", staunt Euleka.

Das funktioniert ja super! Ich wusste, dass du der richtige Datenritter bist!", schwärmt die Königin.
Doch nun wird alles nochmal deutlich komplizierter.

4 „Zeig mir, wie viele Bücher pro Erscheinungsjahr vorhanden sind!", fordert die Königin nun. Euleka weiß genau, dass hier nur der mächtige **GROUP BY**-Zauber helfen kann.

SQL

```sql
SELECT Erscheinungsjahr, COUNT(*) FROM Buecher
GROUP BY Erscheinungsjahr
```

Es erscheint eine Tabelle, die anzeigt, wie viele Bücher pro Jahr erschienen sind.

5 Die Königin frohlockt: „Das wird ja immer besser! Ich bin schwer beeindruckt." und leitet daraufhin Euleka zum großen Finale. „Jetzt suche nach Verbündeten: Finde heraus, welche Leser sich welche Bücher ausgeliehen haben!"

SQL

```sql
SELECT Leser.Name, Buecher.Titel
FROM Ausleihen
JOIN Leser ON Ausleihen.LeserID = Leser.ID
JOIN Buecher ON Ausleihen.BuchID = Buecher.ID
```

Es erscheint eine Liste mit Lesern und den Büchern, die sie ausgeliehen haben. Die Königin ist von Eulekas Fähigkeiten überwältigt.

Du hast alle fünf Aufgaben mit Bravour bestanden und so das Buch der Weisheit gerettet. Ich ernenne dich zum **Hüter von Biblonia**, da du nun weißt, wie wichtig SQL für unser Wohl ist.

Keine Mittagspause, Überstunden ohne Ende und immer wieder JOIN-Befehle?! Das ist schlimmer als der Programmierjob, den ich vorher hatte!

Nachdem du die Abenteuer von Euleka miterlebt hast, bist du sicher ganz verrückt danach, selbst in die magische Welt der Datenbankabfragen mittels SQL einzusteigen. Weiter geht es damit auf der nächsten Seite.

Grundaufgaben in SQLVerine

Übungsaufgaben in SQL-Verine

Löse die neun Aufgaben unter diesem Link, um deine Fähigkeiten in SQL zu verbessern.

https://editor.sqlverine.org

Die Datenbank **Grundschule.db** hat verschiedene Tabellen, welche Information zur Schule enthalten. Mit einem Blick auf das Schema, kannst du dir den Aufbau der Datenbank anschauen und weißt dann, welche Tabellen es gibt und welche Informationen diese wohl beinhalten.

Nun wollen wir uns die Daten der Schule einmal genauer anschauen.

klassen	
id	INTEGER
name	TEXT
jahrgangsstufe	INTEGER

lehrer	
id	INTEGER
vorname	TEXT
nachname	TEXT
geburtsdatum	TEXT
besoldung	TEXT
klasse_id	INTEGER

schueler	
id	INTEGER
vorname	TEXT
nachname	TEXT
geburtsdatum	TEXT
klasse_id	INTEGER

1. Liste alle Klassen der Schule auf.

```
SELECT * FROM klassen
```

2. Liste alle Schüler der Schule auf.

```
SELECT * FROM schueler
```

3. Liste alle Lehrer der Schule auf.

```
SELECT * FROM lehrer
```

Justus A. Weber

4 Finde aus der Tabelle Schüler den Nachnamen von Paul.

```
SELECT nachname FROM schueler
WHERE vorname = 'Paul'
```

5 Sortiere alle Vornamen der Schüler nach dem Geburtsdatum.

```
SELECT vorname FROM schueler
ORDER BY geburtsdatum ASC
```

6 Suche die höchste Gehaltsstufe aller Lehrkräfte mit Hilfe der Aggregatsfunktion MAX.

```
SELECT MAX(besoldung) FROM lehrer
```

7 Lass dir neben der Gehaltsstufe zusätzlich den Vor- und den Nachnamen der Lehrkraft anzeigen.

> **Hinweis**
> Mehrere Attribute lassen sich in SQLVerine anzeigen, indem du auf das bisherige Attribut und dann auf das grüne „Plus" drückst.

```
SELECT MAX(besoldung), vorname, nachname FROM lehrer
```

8 Finde heraus, wie viele Schüler•innen in jeder Klasse sind.

```
SELECT COUNT(id) FROM schueler GROUP BY klasse_id
```

9 Verbinde die Tabelle schueler mit der Tabelle klassen, um den Namen der Klassen auszugeben.

```
SELECT COUNT(id) FROM schueler GROUP BY klasse_id
```

> **Hinweis**
> Die Verbindung von Schülern und Klasse kann im Schema der Datenbank nachgelesen werden. Dort siehst du farblich, dass der Primärschlüssel der Klassen als Fremdschlüssel bei den Schülern genutzt wird. So wird jeder Schüler genau einer Klasse zugeordnet, wobei in einer Klasse mehrere Schüler sein können. Der Verbund (JOIN) muss also über diese Attribute erfolgen.

```
SELECT COUNT(schueler.id), klassen.name FROM schueler
JOIN klassen ON schueler.klasse_id = klassen.id GROUP BY klassen.id
```

Übungsaufgaben

Du kennst das sicherlich: Ihr wollt abends zuhause einen Film schauen, wisst aber nicht genau welchen. Hilfreich ist hier oft die Möglichkeit, auf einem Streamingdienst zu suchen oder nach gewissen Kriterien zu filtern. Technisch läuft im Hintergrund dabei nichts anderes als eine clever gewählte Datenbankabfrage ab. Dieser „Datenbankprofi hinter den Kulissen" darfst du in den folgenden Aufgaben selbst sein.

① Liste alle Filme der Datenbank mit all ihren Eigenschaften auf.

```sql
SELECT * FROM Film
```

② Gib aus der Tabelle Serie nur die Spalten Titel und Erscheinungsjahr aus.

```sql
SELECT Titel, Erscheinungsjahr FROM Serie
```

③ Zeige alle Filme, die im Jahr 2020 oder später erschienen sind.

```sql
SELECT * FROM Film WHERE Erscheinungsjahr >= 2020
```

④ Gib alle Filmtitel sortiert nach dem Erscheinungsjahr aus.

```sql
SELECT Titel, Erscheinungsjahr FROM Film ORDER BY Erscheinungsjahr
```

⑤ Wie viele Serien wurden jeweils pro Jahr veröffentlicht?

```sql
SELECT Erscheinungsjahr, COUNT(*) FROM Serie
GROUP BY Erscheinungsjahr
```

⑥ Gib alle Serien aus, die der Kategorie „Drama" zugeordnet sind. (Nutze SerienKategorie und Kategorie)

```sql
SELECT Serie.Titel
FROM Serie
JOIN SerienKategorie ON Serie.SId = SerienKategorie.SerienID
JOIN Kategorie ON SerienKategorie.KategorieID = Kategorie.KNr
WHERE Kategorie.Bezeichnung = 'Drama'
```

⑦ Gib aus, wie viele Filme in der Datenbank von Christopher Nolan gedreht wurden.

```sql
SELECT COUNT(*) AS Anzahl_Filme
FROM Film
JOIN FilmRegie ON Film.FId = FilmRegie.FilmID
JOIN Regisseur ON FilmRegie.RegisseurID = Regisseur.DNr
WHERE Regisseur.Vorname = 'Christopher'
AND Regisseur.Nachname = 'Nolan'
```

Justus A. Weber