Einführung in Datenbanken

Andreas Wassmer

2023-09-09

Inhalt

Kapitel 1. Einleitung

1.1. Gedanken zum Kurs

Für diesen Kurs setze ich auf die MySQL-Datenbank. Diese wurde ursprünglich als OpenSource-Software entwickelt. Die Lizenzen wurden dann von Oracle gekauft. Im Moment bietet der Datenbankriese aber immer noch eine frei verfügbare Version an, die Community Edition. Aufgrund der marktbeherrschenden Stellung von Oracle lehnen viele freie Entwickler die Verwedung vom MySQL ab. Sie nutzen dafür MariaDB, welche aus der Abspaltung vom MySQL entstanden ist. Sie bietet die gleiche Funktionalität. Der Code wird aber nicht von Oracle kontrolliert sondern unabhängig davon weiterentwickelt.

Ich habe mich aus Rücksicht auf die Apple-User für MySQL entscheiden. MariaDB lässt sich nicht korrekt auf der Applehardware mit den M1 oder M2 Prozessoren installieren. Es gibt kein vorkompiliertes Paket, welches sich über ein Diskimage installieren lässt. Auf der Homepage von MariaDB wird vorgschlagen, stattdessen den Paketmanager "Homebrew" zu verwenden. Aber auch dieser Weg funktioniert nicht. Hingegen funktionert die Installation von MySQL problemlos.

Der zweite Grund ist, dass wir im Kurs die Datenbankverwaltung "Workbench" einsetzen. Diese ist uneigeschränkt für Windows und Apple verfügbar. Diese Version meldet bei MariaDB Kompatibilitätsprobleme. Meine Erfahrungen zeigen, dass die Exportfunktion von DB eingeschränkt ist. Ich möchte mich ganz auf das Arbeiten mit der Datenbank konzentrieren. Daher habe ich den einfachsten Weg gewählt, der für alle Studierenden funktioniert.

1.2. Literatur

Wir können uns in den zur Verfügung stehenden Lektionen nur oberflächlich mit dem Thema Datenbanken auseinandersetzen. Wenn Sie nicht alles verstanden haben, verweise ich Sie auf YouTube. Dort finden vor allem Einsteiger eine Menge von Videos, auch auf Deutsch. Wenn Sie sich aber vertiefter mit relationalen Datenbanken beschäftigen möchten, kommen Sie mit den Videos nicht weiter. Dann empfehle ich Ihnen das Buch "Grundkurs Relationale Datenbanken" von René Steiner aus dem Verlag Springer Vieweg. Dieses geht deutlich über den Stoff hinaus, den ich Ihnen vermitteln kann.

Als Einstieg empfehlen kann ich auch das Video "SQL Tutorial für Anfänger" auf YouTube (Link1). Junus Ergin nutzt eine vorkonfigurierte Datenbank, die frei verfügbar ist. Zu finden ist sie unter dem Link 2.

1.3. Links

1 MSQL Tutorial für Anfänger	https://www.youtube.com/watch?v=7HZbGReAi5s https://dev.mysql.com/downloads/my
2 Online MySQL-Datenbank	https://www.w3schools.com/sql/trysql.asp?filename=trysql_asc

Kapitel 2. Installation

2.1. MySQL-Datenbank

Den MySQL-Datenbankserver gibt es für Windows, macos und Linux. Öffnen Sie die Webseite die Sie im Abschnitt Links unter [1] finden und wählen Sie die Version für Ihr Betriebssystem aus. Nehmen Sie nicht die neuste Version, sondern die stabile Version 8.0.34. Nachdem die Datei heruntergeladen wurde, installieren Sie sie auf dem PC.

Wenn Sie macos verwenden und den Packagemanager "Homebrew" verwenden, können Sie die neuste Version auch damit installieren:

```
$ brew install mysql
```

Sie erhalten dann die Version 8.1.0.

2.1.1. Alternativen

Wer mit Windows arbeitet und trotzdem lieber MariaDB einsetzt, kann diese selbstverständlich installieren und nutzen. Unsere Beispiele und Übungen werden auch funktionieren. Laden Sie den Community Server herunter. Den Link finden Sie unter [4].

2.2. Workbench

Um die Arbeit mit der MySQL-Datenbank zu vereinfachen, setzen wir das Programm MySQL-Workbench ein. Dieses erlaubt es, mithilfe eines GUIs mit der Datenbank zu interagieren. Zudem bietet es den Vorteil, eine Datenbank über ER-Diagramme (Entity Relationship) zu erzeugen. Dadurch können wir es auch für das Modelling einsetzen.

2.2.1. Alternativen

Auch hier muss ich erwähnen, dass es andere Programme für denselben Zweck gibt. Zu erwähnen ist HeidiSQL, welches nur für Windows angeboten wird, jedoch nicht den gleichen Funktionsumfang hat (siehe Link 4). Neuer ist DBeaver, welches es als frei verfügbare Communityversion gibt. Vom Umfang her ähnlich wie Workbench wäre es eine gute Alternative. Ich finde dessen Bedienung jedoch etwas umständlicher. Zudem ist Workbench nach wie vor ein Standard in der Industrie. Und es gibt eine Version für Windows und macos.

2.3. Links

1 MySQL Community Version	https://dev.mysql.com/downloads/mysql/
2 MySQL Workbench	https://www.heidisql.com/download.php
3 MariaDB	https://mariadb.com/downloads/
4 HeidiSQL	https://www.heidisql.com/download.php

Kapitel 3. SQL-Abfragesprache

3.1. SQL Datentypen

Die Sprache SQL kennt eine grosse Menge von Datentypen. Viele davon sind aus Gründen der Kompatibilität mit verschiedenen Datenbanksystemen redundant. In 90% der Fälle kommen wir mit den folgenden Typen aus:

NULL	Der NULL-Wert, verwendet für einen Wert, der nicht bekannt ist
INTEGER	Eine Ganzzahl mit Vorzeichen, z.B. -12345, 0, 23
REAL	Eine Fliesskommazahl mit Vorzeichen, z.B. -33.33, 0.001, 12345.876
TEXT	Zeichenketten, z.B. "Hallo", "Andreas"
BLOB	Daten, die genau so gespeichert werden. Verwendet z.B. für Bilder

3.2. Datenbanken und Tabellen anlegen

Bevor Sie mit dem Anlegen von Tabellen beginnen können, müssen Sie eine Datenbank anlegen. Dies geschieht mit der Anweisung

```
CREATE DATABASE company;
```

Nun lassen sich die benötigten Tabellen anlegen, z.B. eine, welche die Personalien der Angestellten enthält:

```
CREATE TABLE company.employees (
    emp_no INT PRIMARY KEY AUTO_INCREMENT,
    birth_date DATE,
    first_name VARCHAR(100),
    last_name VARCHAR(100),
    gender INT
    hire_date DATE);
```

3.3. Abfragen von Daten mit SQL

3.3.1. Informationen über die Datenbank

Alle vorhandenen Tabellen in einer Datenbank abfragen:

```
SHOW TABLES;
```

Alle Attribute einer Tabelle

Die folgende Abfrage zeigt alle Attribute der Tabelle `departments`

```
SELECT * FROM departments
```

dept_no	dept_name
d009	Customer Service
d005	Development
d002	Finance
d003	Human Resources
d001	Marketing
d004	Production
d006	Quality Management
d008	Research
d007	Sales

Nur gewisse Attribute anzeigen

Für unsere Datenbank `employees` liefert die Abfrage

```
SELECT first_name, last_name FROM employees LIMIT 5;
```

die Vor- und Nachnamen der ersten 5 Einträe in der Tabelle:

first_name	last_name
Georgi	Facello
Bezalel	Simmel
Parto	Bamford
Chirstian	Koblick
Kyoichi	Maliniak

Filtern von Abfragen

Abfrage aller Mitarbeiter deren Gehalt kleiner als $50'000 ist:

```
SELECT emp_no, salary FROM salaries WHERE salary < 50000;
```

emp_no	salary

10003	40006
10003	43616
10003	43466
10003	43636
10003	43478
…	…

3.4. Übersicht SELECT Statement

Die folgende Abbildung zeigt die Syntax des SELECT-Statements.

3.5. Daten einfügen

Das folgende Statement fügt der Tabelle `departments` ein neues Departement hinzu:

```
INSERT INTO departments VALUES('d010', 'Computing')
```

3.6. Aufräumen

3.6.1. Löschen von Tabellen und Datenbanken

Mit der Anweisung `DROP` können Sie Tabellen und Datenbanken löschen. Der Inhalt geht dabei unwiderbringlich verloren. Wenn Sie mit der Shell arbeiten gibt es auch keine Warnung.

Eine Tabelle löschen Sie mit:

```
DROP TABLE employees;
```

und die Datenbank mit

```
DROP DATABASE company;
```

Kapitel 4. Arbeiten mit Date und Time

MySQL kennt spezielle Datentypen für das Speichern eines Datums und einer Uhrzeit. Soll in einer Tabelle `member` etwas das Beitrittsdatum erfasst werden, so kann dies wie folgt geschehen

```
CREATE table member (
    id int primary key auto_increment,
    first_name varchar(100),
    family_name varchar(100),
    member_since date
)
```

Ein Datensatz wird hinzugefügt

```
insert into member (first_name, family_name, member_since) values
("Peter", "Schweizer", "1969-03-10");
```

Es ist wichtig, dass das Datum in der Form "YYYY-MM-DD" eingegeben wird. Wird anstelle des Standardformats zum Beispiel das in der Schweiz übliche Format "10.03.1969" verwendet, verweigert MySQL die Annahme mit dem Fehler `Error Code: 1292. Incorrect date value: '10.03.1969' for column 'member_since' at row 1`

Im Prinzip lässt sich ein Datum auch als `VARCHAR` speichern. Dann muss die Umwandlung in ein Datum im Code geschehen.

Die Verwednug des Datentyps `DATE` hat daher den Vorteil, dass das Arbeiten mit Daten durch Standardfunktionen des DBMS erleichtert wird. Will man z.B. wissen, welche Mitglieder im Jahr 2000 eingetreten sind, so nutzt man das Statement

```
select first_name, family_name from member where year(member_since) = 2000;
```

Mit der Funktion `CURDATE()` lässt sich das aktuelle Datum ermitteln:

```
select curdate();
```

```
+------------+
| curdate()  |
+------------+
| 2023-09-01 |
+------------+
1 row in set (0.00 sec)
```

Damit lässt sich ausrechnen, welches Mitglied bereits wie lange im Club ist:

```
select first_name, family_name, year(curdate())-year(member_since) as membership_years
from member;
```

```
+------------+-------------+------------------+
| first_name | family_name | membership_years |
+------------+-------------+------------------+
| Peter      | Schweizer   |               54 |
+------------+-------------+------------------+
1 row in set (0.00 sec)
```

Es lassen sich auch bestimmte Zeitintervalle berechnen:

```
select datediff('2023-12-25', curdate()) as days_to_xmas;
```

```
+--------------+
| days_to_xmas |
+--------------+
|          115 |
+--------------+
1 row in set (0.00 sec)
```

Das Datum lässt sich in fast jedes Format konvertieren. Benötigt man z.B. das in der Schweiz übliche Format, so gelingt dies mit

```
select first_name, family_name, date_format(member_since, "%d.%m.%Y") as membership
from member;
```

```
+------------+-------------+------------+
| first_name | family_name | membership |
+------------+-------------+------------+
| Peter      | Schweizer   | 10.03.1969 |
+------------+-------------+------------+
1 row in set (0.00 sec)
```

4.1. Der Datentyp TIME

Auch für das Speichern einer Uhrzeit gibt es einen Datentyp: `TIME`. Dieser erwartet die Daten immer in der Form "HH:MM:SS" oder "HH:MM", also z.B. "07:12:00" oder einfach "7:12". Die führende Null bei Stunden, Minuten und Sekunden kann weggelassen werden.

Der Typ `TIME` kann auch verwendet werden, um Zeitspannen abzubilden, z.B.

```
select name, ip, time_online from server where ip="192.168.1.23";
```

```
+-----------+-------------+-------------+
| name      | ip          | time_online |
+-----------+-------------+-------------+
| webserver | 192.168.1.23 | 123:45:05   |
+-----------+-------------+-------------+
1 row in set (0.00 sec)
```

Allerdings sind nur Zeitspannen bis zur maximalen Länge von '-838:59:59' oder '838:59:59' möglich. Sollen längere Zeiträume abgebildet werden, kann auf den Datentyp `VARCHAR` ausgewichen werden.

4.2. Der Datentyp DATETIME

Oft ist es nötig, das Datum und die Uhrzeit zusammen zu speichern. Man spricht dann auch von einem Zeitstempel. Für solche Fälle existiert der Datentyp `DATETIME`. Er erwartet Daten in der Form "YYYY-MM-DD HH:MM:SS". Für den Zeitteil gilt wiederum, dass die führenden Nullen nicht geschrieben werden müssen.

```
select username, last_login from players where username="king12345";
```

```
+-----------+---------------------+
| username  | last_login          |
+-----------+---------------------+
| king12345 | 2023-07-25 13:01:34 |
+-----------+---------------------+
1 row in set (0.00 sec)
```

4.3. Links

Die Doku zu den Typen DATE, TIME und DATETIME
https://dev.mysql.com/doc/refman/5.7/en/date-and-time-types.html
Eine Erklärung zu allen Datums- und Zeit-Funktionen
https://dev.mysql.com/doc/refman/5.7/en/date-and-time-functions.html

Anhang A: Arbeiten mit CSV-Dateien

Anhang B: Arbeiten mit Excel-Dateien