MySQL Einsteiger

Agenda

1. Technical Background
    - Technical Structure
    - Was ist SQL ?
2. Grundlagen
    - Was sind Datenobjekte und welche gibt es ?
    - SQL-Syntax
    - Datenbank-und-Tabellen-verwenden
    - Struktur und Indizes von Tabellen auslesen
3. SELECT's
    - Alle Datensätze abfragen und alias für Spalte setzen
    - Rechnen mit SELECT - Beispiele
    - Beispiel und Übung - Berechnen aus Feldern
    - Beispiel mit Select where
    - Beispiel mit select where in
    - Beispiel mit select und like
    - Beispiele mit select und range
    - Beispiel mit select und not in
    - Beispiel mit select und SUBSTR,LOWER,UPPER,CONCAT
4. IS NULL / IS NOT NULL
    - Beispiel und Übung mit IS NULL / IS NOT NULL
5. DATUM (DATETIME oder dier Unix Timestamp)
    - Beispiel mit select by date
    - Schlechtes Beispiel mit select by date
    - Beispiel mit Unix Timestamp (Umwandeln)
    - Beispiel mit Dateformat
6. COUNT / STATISTIK
    - Zählen von Datensätzen
    - Beispiel für Statistikabfrage (MAX,MIN)
7. NUMERISCHE FUNKTIONEN
    - Abrunden / Aufrunden / Kaufmännisch Runden
8. ORDER BY (Sortierung)
    - Beispiel und Übung order by
9. LIMIT
    - Kombiniertes Beispiel mit Order By und Limit + Übung
10. GROUP BY
    - Example GROUP BY / HAVING

11. JOINS

- Überblick über JOINS
- Example join - 2 tables - 1:1
- Beispiel Join über 3 Tabellen

12. ÜBUNGEN

- Uebung - Berechnung aus Feld
- Übung order by
- Übung mit Order By und Limit
- Übung mit Select where
- Übung mit select where in
- Übung mit select und like
- Übung 1+2 - Select Range
- Übung mit select und not in
- Übung mit select by date
- Übung mit select by day
- Übung - Zählen vpon Datensätzen
- Übung - Statistik AVG
- Übung - NOT NULL / NULL
- Übung mit select und SUBSTR,LOWER,UPPER,CONCAT
- Übung mit unix timestamp
- Übung mit Dateformat
- Übung join - 2 tables - 1:1
- Übung - Refresher Tag 2 - morgens - SELECT
- Übung - Refresher Tag 3 - morgens - SELECT

13. TIPPS & TRICKS

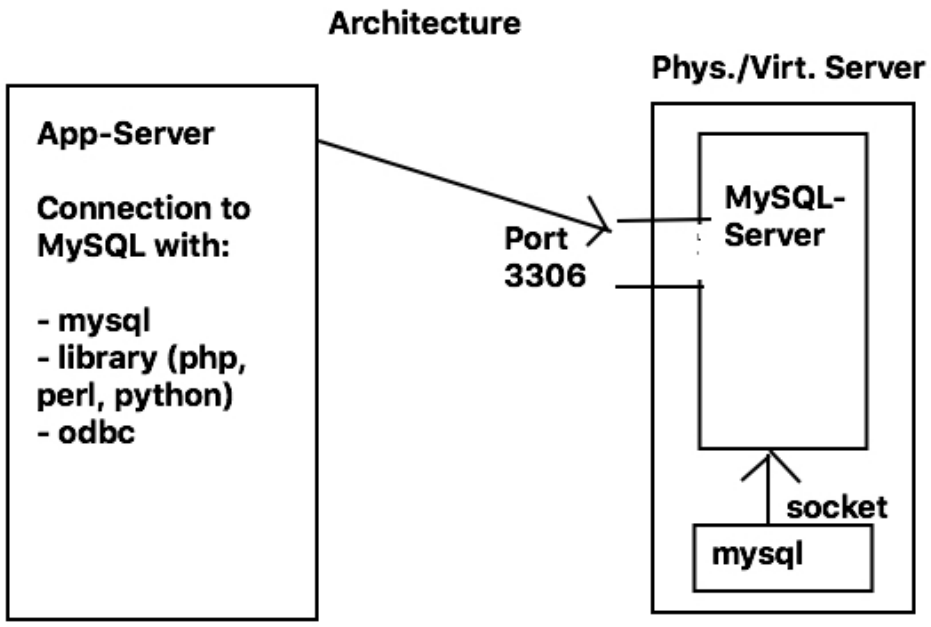
- Cheatsheet - Auf dem System zurechtfinden - 1. Sichtung
- Cheatsheet für Selects

14. Documentation

- https://dev.mysql.com/downloads/windows/installer/8.0.html

Technical Background

Technical Structure



Was ist SQL ?

```
SQL - Structured Query Language -> Strukturierte Abfragesprache
Es ist eine standardisierte Programmiersprache
die zur Verwaltung relationale Datenbanken
und zum Durchführen verschiedener mit den darin
enthaltenen Daten verwendet wird.
```

```
Will ich Daten aus eine SQL-Datenbank brauche ich die Abfrage
sprache SQL
```

Grundlagen

Was sind Datenobjekte und welche gibt es ?

Was sind Datenbankobjekte

```
Bilden eine Strutkur um Daten zu speichern
```

Welche gibt es ? (Klassiker)

Ebene 1 (oberste Ebene) Datenbank (databases/schemas)

```
eine Organisationseinheit: die Datenbank
wie ein Behältnis
unter der Haube: Verzeichnis

z.B. Datenbank sakila
```

Ebene 2 - Tabellen

```
Eine Datenbank kann mehrere Tabellen haben.
Ähnlich eines Vorratsschrankes.
Im Filesystem finden man diese unterhalb der Datenbank mit
dem entsprechenden der Tabelle:
z.B. sakila\actor.ibd

Jede Tabelle hat eine Struktur -> Columns (Felder)
```

Ebene 3 - Felder / Columns

```
Die Tabelle hat eine Struktur, die bestimmt wird durch die Columns (Felder)
Jedes Feld (Column) hat einen Datentyp, der bestimmt welcher Daten dort rein dürfen.
z.B. Strings (varchar), oder Zahlen (z.B. INT -> Integer)
```

Daten -> Zeilen (ROW)

```
Daten werden zeilenweise in Tabellen geschrieben
Jede Zeile hat ein eindeutiges Merkmale (eine eindeutige Nummer) -> Primärschlüssel
(Primary Key)
```

Weitere Datenbankobjekte

```
Views - Schadpotenzial -> 0%
Trigger - Schadpotential -> 50%

-> hier unkritisch, da nur neu eintrag in der Datenbank
CREATE TRIGGER before_employee_update
    BEFORE UPDATE ON employees
    FOR EACH ROW
 INSERT INTO employees_audit
 SET action = 'update',
     employeeNumber = OLD.employeeNumber,
     lastname = OLD.lastname,
     changedat = NOW();

Procedures -
CALL film_in_stock(1,1,@ausgabe);
select @ausgabe;
```

```
Function (wie systemfunktionen, nur selbst erstellt)
use sakila;
select get_customer_balance(1,'2015-06-01 12:55:12');

Events (zeitgesteuerte Ereignisse) -
Schadcodepotenzial when aktiviert -> Procedures
https://www.mysqltutorial.org/mysql-triggers/working-mysql-scheduled-event/

Wie weiss ich, dass events generell ausgeführt auf meinem System, wenn vorhanden
```

- Werden events ausgeführt

SQL-Syntax

Kommentare

```
-- Das ist ein Kommentar, der Text beschreibt etwas aber wir nicht vom SQL-Server
ausgeführt
```

SQL-Schlüsselworte für Operationen

```
DDL - Daten Definition Language
DML - Data Modelling Language

DDL ändern die Struktur
=======================
CREATE DATABASE #
CREATE SCHEMA  # Datenbank
CREATE TABLE
ALTER TABLE # VErändere die Struktur
DROP

CREATE - erstellen
DROP - Löschen (komplettes Objekt)
ALTER - verändern

DML - Ändert die Inhalte und macht Abfragen
====
SELECT
DELETE - löscht Daten
INSERT - einfügen vpn Daten
UPDATE - Updaten von Daten
```

Datenbank-und-Tabellen-verwenden

```
-- Datenbanken anzeigen
show databases;
-- or:
show schemas;

-- Datenbank verwenden / wechseln
use sakila;
```

```
-- Tabellen für ausgewählte Datenbanken anzeigen
show tables;

### Struktur und Indizes von Tabellen auslesen
```

show fields from actor; show create table actor; show indexes from actor;

-- Empfehlung, was ir können solltet describe actor; show indexes from actor;

```
## SELECT's

### Alle Datensätze abfragen und alias für Spalte setzen
```

select first_name as Vorname,last_name as Nachname from actor;

```
### Rechnen mit SELECT - Beispiele
```

-- erlaubt sind + - / * DIV (Integer Division) -- Integer-Division: 15/7 = 2 (nur Ganzzahl wird als Ergebnis zurückgegeben)

-- + - * / DIV(Integer) SELECT 15 DIV 2 as Ergebnis;

```
### Beispiel und Übung - Berechnen aus Feldern


### Beispiel
```

SELECT amount,1 as 'Preiserhoehung um',amount + 1 as preiserhoehung FROM payment;

```
### Übung (sakila)
```

o Lese aus der Tabelle film, den rental_rate aus. o Wir nehmen an dass dieser Netto ist. o Berechne den Brutto-Preis rental_rate * 1,19 o Gebe Nettopreis, Steuersatz und Bruttopreis aus (das sind die Überschriften)

```
### Beispiel mit Select where


### Example (Simple)
```

SELECT * FROM actor WHERE last_name = 'AKROYD' or last_name = 'GABLE';

```
### Example (mit Klammern)
```

SELECT * FROM actor WHERE (last_name = 'Akroyd' and first_name = 'Christian') or (last_name = 'Gable' and first_name = 'Christian');

```
### Übung
```

- Alle Datensätze aus actor anzeigen bei dennen der Nachname Akroyd oder der Vorname Christian ist.

```
### Beispiel mit select where in


### Example

SELECT * FROM actor WHERE first_name IN ('JOE','ED','JENNIFER');

### Übung
```

- Sucht Euch 3 Citys und last die Datensätze aus city dazu ausgeben
- Frage: Welche Feld verwenden ?
- mit where ... IN umsetzen

```
### Beispiel mit select und like


### Example
```

SELECT * FROM actor WHERE last_name like 'D%';

```
### Exercise
```

Lass Euch alle actor ausgeben, deren Nachname auf N endet.

```
### Beispiele mit select und range


### Beispiel 1
```

SELECT * FROM actor where actor_id > 100 and actor_id < 150;

10 und 50 inkludiert

select * from actor where actor_id >= 10 and actor_id <= 50;

```
### Beispiel 3
```

SELECT * FROM actor where actor_id between 100 and 150;

```
### Übung 1
```

- Gebe alle Adressen (address) aus, wo der postal_code > 10000 und postal_code < 20000 ist oder < 90000 und > 50000
- sortiert nach district
- gebe 20 Einträge aus (offset 0)

Übung 2

```
- Lasst euch Filme (film) anzeigen deren Wiederbeschaffungspreis (rental ... )
zwischen 18.99 und 20.99 ist. (inkl)
```

Übung 3

```
Lass Euch alle Eitnräge aus dem inventory mit between im Bereich 10 bis 100 anzeigen
(inkl 10 und 100)
```

Beispiel mit select und not in

Beispiel:

```
SELECT * FROM actor WHERE last_name NOT IN ('AKROYD','JONES');
```

Übung:

```
Lasst Euch alle Adressen anzeigen, die nicht im District QLD oder Alberta sind
```

Beispiel mit select und SUBSTR,LOWER,UPPER,CONCAT

Example

```
SELECT concat(title,' ',release_year,' ',description) as listeneintrag FROM film;
SELECT concat ('hans und lotta',' ','haben glueck');
SELECT concat('Ausgabe: ',substr(description,1,20)) as listeneintrag FROM film;
SELECT upper(substr(description,1,20)) from film;
SELECT upper(substr(description,1,20)) from film;
SELECT lower(substr(description,1,20)) from film;
```

Exercise

```
-- Db: sakila
-- Tabelle: actor
Gebt folgende Felder aus (in einer Abfrage)

o last_name gross geschrieben
o first_name klein geschrieben
o ausgabe mit zusammengeklebt last_name + ' ' + first_name (ohne +)
```

```
o erste beiden Buchstaben vom Nachnamen.

UPPER, LOWER, CONCAT, SUBSTR
```

IS NULL / IS NOT NULL

Beispiel und Übung mit IS NULL / IS NOT NULL

Example

```
SELECT * FROM rental WHERE return_date IS NOT NULL;
SELECT * FROM rental WHERE return_date IS NULL;
```

Exercise

```
## Database: sakila
## Tabelle: staff

Übung 1:

a) Gebt alle Datensätze aus in denen das Passwort NICHT GESETZT ist

b) Gebt alle Datensätze aus in denen das Passwort GESETZT ist
```

DATUM (DATETIME oder dier Unix Timestamp)

Beispiel mit select by date

Example

```
SELECT * FROM sakila.payment WHERE payment_date > '2005-05-24 22:00:01' and
payment_date < '2005-05-25 22:00:01'
```

Übung:

```
Gebt alle Eintrage aus der Tabelle rental aus, bei der die Rückgabe zwischen
>= 2005-03-06 00:00:01   und <  2005-08-01
der letzte eintrag wäre 2005-07-31 23:59:59
return_date
```

Schlechtes Beispiel mit select by date

Example

```
select YEAR(return_date),return_date from rental where YEAR(return_date) <= 2005;
```

Example 2:

```
select YEAR(return_date),MONTH(return_date),DAY(return_date) from rental;
```

Exercise

```
Lasst euch alle Rückgaben aus Rental anzeigen, die zwischen (inkl) dem 27. und 30.
zurückgegeben wurden
(egal welchen Monats)
```

Beispiel mit Unix Timestamp (Umwandeln)

Was ist das ?

- Das Datum wird dargestellt als Sekunden seit 01.01.1970

Warum ?

- Mit dem Unix Timestamp lassen sich einfache Vergleiche machen
- und Auswahlen treffen (z.B. Range - Abfrage)

Beispiele

```
SELECT unix_timestamp('2022-01-04');
-- Datum auslesen und in unix timestamp umwandeln
SELECT last_update,unix_timestamp(last_update) from actor;
-- Aktuelles Datum als unix timestam (inkl Uhrzeit) - Systemzeit des Servers/Rechners
-- Rechner auf dem MySQL - Server läuft
select unix_timestamp();

-- Unix timestamp in einer Datum umwandeln
SELECT from_unixtime('1648202962');
```

Exercise

```
Frage aus rental alle Datensätze ab beidenen
o Das return_date nicht null (NOT NULL) ist
o Gib als einziges Feld das return_date umgewandelt zum Unix timestamp an.
o HINT: UNIX_TIMESTAMP()
```

Beispiel mit Dateformat

Example

```
SELECT date_format(last_update,'%M %Y') as meindatum FROM actor;
```

Exercise

```
Gib für rental_date aus der tabelle  rental das rental_date für jeden Datensatz wie
folgt aus:
Tag-der-Woche Jahr-2-stellig
```

```
Weitere Spalten werden nicht benötigt.
```

- https://www.w3schools.com/sql/func_mysql_date_format.asp

COUNT / STATISTIK

Zählen von Datensätzen

Example

```
SELECT count(*) as howmany FROM actor;
select count(*) from actor where last_name like 'D%';
```

Exercise

```
Wieviele Filme mit der film_id 15 gibt es im Inventar (über alle Stores hinweg)
in der tabelle inventory
```

Beispiel für Statistikabfrage (MAX,MIN)

Beispiel

```
SELECT MAX(replacement_cost) as am_teuersten,MIN(replacement_cost) FROM sakila.film;
```

Exercise

```
Was sind die durchnittlichen Leihgebühren für alle Film, die mit dem Titel "T"
anfangen.
AVG() wird hier benötigt
```

NUMERISCHE FUNKTIONEN

Abrunden / Aufrunden / Kaufmännisch Runden

Example

```
-- Kaufmännisch runden
SELECT ROUND(1.56,1);
-- Abschneiden
SELECT truncate(1.56,1);
-- FLOOR() -> abrunden zum nächsten Integer
SELECT FLOOR(12.56);
-- CEIL() - Aufrunden zum nächsten Integer
SELECT CEIL(12.3);
```

ORDER BY (Sortierung)

Beispiel und Übung order by

Beispiel

```
SELECT first_name,last_name FROM actor ORDER BY last_name DESC, first_name;
```

Übung (schlechtes Beispiel, anderes wäre besser)

```
1. Wir sortieren alle Einträge film - Tabelle
2. rental_rate absteigend, Titel aufsteigend, release_year absteigend
```

LIMIT

Kombiniertes Beispiel mit Order By und Limit + Übung

What does it do ?

```
From your results of your query only shows a subset
```

Example

```
SELECT * from actor ORDER BY last_name DESC limit 3;
```

Übung

```
1. Zeige von den Filmen (sakila.film) die teuersten (rental_rate) zuerst, davon

Variante 1:
- Zeige die ersten 20

Variante 2: Zeige 10 ab dem 11. Film
```

```
## GROUP BY

### Example GROUP BY / HAVING


### What does it do ?
```

Group By aggregates data (data sets) Example: So if something appears multiple times, e.g a first name, it will only appears once.

```
### Example GROUP BY
```

SELECT last_name,COUNT(last_name) as cnt FROM actor GROUP BY last_name;

```
### Exercise
```

DB: sakila Table: film

Gruppiere alle filme nach rental_rate und gibt die jeweilige anzahl aus d.h. z.B. (wert nicht korrekt - weil nicht überprüft

preis. anzahl 0.99. 4 2.99. 50

```
### What does HAVING do
```

Step 1: First MySQL reads all the data based on WHERE

Step 2: Then it aggregates the data (GROUP BY) and filters it by HAVING

```
### Example Having


  * Simple: WHERE for GroupBy (because where does not work here)
  * Example
```

SELECT last_name, COUNT(last_name) FROM sakila.actor GROUP BY last_name HAVING count(last_name) > 2

```
### Exercise
```

Gruppiere alle actor o mit nachname, und anzahl nachname (felder die wir ausgeben) o Gebe nur die actor aus, deren nachname mit J anfängt

```
## JOINS

### Überblick über JOINS


### What is a JOIN for ?

 * combines rows from two or more tables
 * based on a related column between them.

### MySQL/MariaDB (Inner) Join

![Inner Join](/images/img_innerjoin.gif)

### MySQL/MariaDB (Inner) Join (explained)

  * Inner Join and Join are the same
  * Returns records that have matching values in both tables
  * Inner Join, Cross Join and Join
    * are the same in MySQL

### In Detail: [INNER] JOIN

  * Return rows when there
```

```
    * is a match in both tables
  * Example
```

SELECT actor.first_name, actor.last_name, film.title FROM film_actor INNER JOIN actor ON film_actor.actor_id = actor.actor_id INNER JOIN film ON film_actor.film_id = film.film_id;

```
### MySQL/MariaDB Left Join

![Image Left Join](/images/img_leftjoin.gif)

### MySQL/MariaDB Left (outer) Join (explained)

  * Return all records from the left table
  * _AND_ the matched records from the right table
  * The result is NULL on the right side
    * if there are no matched columns on the right
  * Left Join and Left Outer Join are the same


### In Detail: Left Join

  * Return all rows from the left side
    * even if there is not result on the right side
  * Example
```

SELECT c.customer_id, c.first_name, c.last_name, a.actor_id, a.first_name, a.last_name FROM customer c LEFT JOIN actor a ON c.last_name = a.last_name ORDER BY c.last_name;

```
### Example join - 2 tables - 1:1


### Example Join über 2 Tabellen (1:1)
```

SELECT a.*,c.* FROM customer c JOIN address a ON c.address_id = a.address_id; SELECT c.first_name,c.last_name,a.postal_code FROM customer c JOIN address a ON c.address_id = a.address_id; -- Komplexe Alternative, ich würde davon abraten, weil zuviel Schreibarbeit SELECT customer.first_name,customer.last_name,address.postal_code FROM customer JOIN address ON customer.address_id = address.address_id;

```
### Exercise über 2 Tabellen (1:1)
```

DB: sakila Tables: store,address
Feld: address_id Zeige alle Adressen der Stores an ? (Alle Felder von store und von address

```
### Beispiel Join über 3 Tabellen


### Example
```

-- Schritt 1 - Erstmal JOIN über 2 Tabellen

select a.last_name,fa.film_id from actor a JOIN film_actor fa ON a.actor_id = fa.actor_id;

-- Schritt 2 (Schritt 1 erweitert) -- Jetzt der JOIN über 3 Tabellen SELECT a.last_name,fa.film_id,f.title FROM actor a JOIN film_actor fa ON a.actor_id = fa.actor_id JOIN film f ON fa.film_id = f.film_id;

```
## ÜBUNGEN

### Uebung - Berechnung aus Feld

### Übung order by


### Beispiel
```

SELECT first_name,last_name FROM actor ORDER BY last_name DESC, first_name;

```
### Übung (schlechtes Beispiel, anderes wäre besser)
```

1. Wir sortieren alle Einträge film - Tabelle
2. rental_rate absteigend, Titel aufsteigend, release_year absteigend

Übung mit Order By und Limit

What does it do ?

```
From your results of your query only shows a subset
```

Example

```
SELECT * from actor ORDER BY last_name DESC limit 3;
```

Übung

```
1. Zeige von den Filmen (sakila.film) die teuersten (rental_rate) zuerst, davon

Variante 1:
- Zeige die ersten 20

Variante 2: Zeige 10 ab dem 11. Film
```

```
### Übung mit Select where


### Example (Simple)
```

SELECT * FROM actor WHERE last_name = 'AKROYD' or last_name = 'GABLE';

```
### Example (mit Klammern)
```

SELECT * FROM actor WHERE (last_name = 'Akroyd' and first_name = 'Christian') or (last_name = 'Gable' and first_name = 'Christian');

```
### Übung
```

- Alle Datensätze aus actor anzeigen bei dennen der Nachname Akroyd oder der Vorname Christian ist.

```
### Übung mit select where in


### Example

SELECT * FROM actor WHERE first_name IN ('JOE','ED','JENNIFER');

### Übung
```

- Sucht Euch 3 Citys und last die Datensätze aus city dazu ausgeben
- Frage: Welche Feld verwenden ?
- mit where ... IN umsetzen

```
### Übung mit select und like


### Example
```

SELECT * FROM actor WHERE last_name like 'D%';

```
### Exercise
```

Lass Euch alle actor ausgeben, deren Nachname auf N endet.

```
### Übung 1+2 - Select Range


### Beispiel 1
```

SELECT * FROM actor where actor_id > 100 and actor_id < 150;

10 und 50 inkludiert

select * from actor where actor_id >= 10 and actor_id <= 50;

```
### Beispiel 3
```

SELECT * FROM actor where actor_id between 100 and 150;

```
### Übung 1
```

- Gebe alle Adressen (address) aus, wo der postal_code > 10000 und postal_code < 20000 ist oder < 90000 und > 50000
- sortiert nach district
- gebe 20 Einträge aus (offset 0)

Übung 2

```
- Lasst euch Filme (film) anzeigen deren Wiederbeschaffungspreis (rental ... )
zwischen 18.99 und 20.99 ist. (inkl)
```

Übung 3

```
Lass Euch alle Eitnräge aus dem inventory mit between im Bereich 10 bis 100 anzeigen
(inkl 10 und 100)
```

Übung mit select und not in

Beispiel:

```
SELECT * FROM actor WHERE last_name NOT IN ('AKROYD','JONES');
```

Übung:

```
Lasst Euch alle Adressen anzeigen, die nicht im District QLD oder Alberta sind
```

Übung mit select by date

Example

```
SELECT * FROM sakila.payment WHERE payment_date > '2005-05-24 22:00:01' and
payment_date < '2005-05-25 22:00:01'
```

Übung:

```
Gebt alle Eintrage aus der Tabelle rental aus, bei der die Rückgabe zwischen
>= 2005-03-06 00:00:01   und <  2005-08-01
der letzte eintrag wäre 2005-07-31 23:59:59
return_date
```

Übung mit select by day

Example

```
select YEAR(return_date),return_date from rental where YEAR(return_date) <= 2005;
```

Example 2:

```
select YEAR(return_date),MONTH(return_date),DAY(return_date) from rental;
```

Exercise

```
Lasst euch alle Rückgaben aus Rental anzeigen, die zwischen (inkl) dem 27. und 30.
zurückgegeben wurden
(egal welchen Monats)
```

Übung - Zählen vpon Datensätzen

Example

```
SELECT count(*) as howmany FROM actor;
select count(*) from actor where last_name like 'D%';
```

Exercise

```
Wieviele Filme mit der film_id 15 gibt es im Inventar (über alle Stores hinweg)
in der tabelle inventory
```

Übung - Statistik AVG

Beispiel

```
SELECT MAX(replacement_cost) as am_teuersten,MIN(replacement_cost) FROM sakila.film;
```

Exercise

```
Was sind die durchnittlichen Leihgebühren für alle Film, die mit dem Titel "T"
anfangen.
AVG() wird hier benötigt
```

Übung - NOT NULL / NULL

Example

```
SELECT * FROM rental WHERE return_date IS NOT NULL;
SELECT * FROM rental WHERE return_date IS NULL;
```

Exercise

```
## Database: sakila
## Tabelle: staff

Übung 1:

a) Gebt alle Datensätze aus in denen das Passwort NICHT GESETZT ist

b) Gebt alle Datensätze aus in denen das Passwort GESETZT ist
```

Übung mit select und SUBSTR,LOWER,UPPER,CONCAT

Example

```
SELECT concat(title,' ',release_year,' ',description) as listeneintrag FROM film;
SELECT concat ('hans und lotta',' ','haben glueck');
SELECT concat('Ausgabe: ',substr(description,1,20)) as listeneintrag FROM film;
SELECT upper(substr(description,1,20)) from film;
SELECT upper(substr(description,1,20)) from film;
SELECT lower(substr(description,1,20)) from film;
```

Exercise

```
-- Db: sakila
-- Tabelle: actor
Gebt folgende Felder aus (in einer Abfrage)

o last_name gross geschrieben
o first_name klein geschrieben
o ausgabe mit zusammengeklebt last_name + ' ' + first_name (ohne +)
o erste beiden Buchstaben vom Nachnamen.

UPPER, LOWER, CONCAT, SUBSTR
```

Übung mit unix timestamp

Was ist das ?

- Das Datum wird dargestellt als Sekunden seit 01.01.1970

Warum ?

- Mit dem Unix Timestamp lassen sich einfache Vergleiche machen
- und Auswahlen treffen (z.B. Range - Abfrage)

Beispiele

```
SELECT unix_timestamp('2022-01-04');
-- Datum auslesen und in unix timestamp umwandeln
SELECT last_update,unix_timestamp(last_update) from actor;
-- Aktuelles Datum als unix timestam (inkl Uhrzeit) - Systemzeit des Servers/Rechners
-- Rechner auf dem MySQL - Server läuft
```

```
select unix_timestamp();

-- Unix timestamp in einer Datum umwandeln
SELECT from_unixtime('1648202962');
```

Exercise

```
Frage aus rental alle Datensätze ab beidenen
o Das return_date nicht null (NOT NULL) ist
o Gib als einziges Feld das return_date umgewandelt zum Unix timestamp an.
o HINT: UNIX_TIMESTAMP()
```

Übung mit Dateformat

Example

```
SELECT date_format(last_update,'%M %Y') as meindatum FROM actor;
```

Exercise

```
Gib für rental_date aus der tabelle  rental das rental_date für jeden Datensatz wie
folgt aus:
Tag-der-Woche Jahr-2-stellig

Weitere Spalten werden nicht benötigt.
```

- https://www.w3schools.com/sql/func_mysql_date_format.asp

Übung join - 2 tables - 1:1

Example Join über 2 Tabellen (1:1)

```
SELECT a.*,c.* FROM customer c JOIN address a ON c.address_id = a.address_id;
SELECT c.first_name,c.last_name,a.postal_code FROM customer c JOIN address a ON
c.address_id = a.address_id;
-- Komplexe Alternative, ich würde davon abraten, weil zuviel Schreibarbeit
SELECT customer.first_name,customer.last_name,address.postal_code FROM customer JOIN
address ON customer.address_id = address.address_id;
```

Exercise über 2 Tabellen (1:1)

```
DB: sakila
Tables: store,address
Feld: address_id
Zeige alle Adressen der Stores an ? (Alle Felder von store und von address
```

Übung - Refresher Tag 2 - morgens - SELECT

Übung 1:

```
actor: Datensätze ausgeben, wo der Vorname mit C beginnt
und hier nur die ersten 10 sortiert nach Nachname
- nur die Felder Vorname, Nachname ausgeben
```

Übung 2:

```
addresses:
- Alle Adressen die im district QLD, Alberta, Queens sind
- Sortierung nach postal_code
- nur die ersten 10 davon anziegen
```

Übung - Refresher Tag 3 - morgens - SELECT

Übung 1:

```
Datenbank: sakila
- Gebe den höchsten, den niedrigsten und den mittleren Zahlbetrag aus
für alle Zahlungen (payment_date) vor dem 01.07.2005 (2005-07-01)
RUNDE den mittleren Zahlbetrag kaufmännisch (ROUND) mit 2 Nachkommastellen.


Hinweis:
AVG, MIN, MAX, ROUND
und die Tabelle payment soll verwendet werden.
```

- https://www.w3schools.com/sql/func_mysql_round.asp

Übung 2:

```
Datenbank: sakila.
GEBE alle Sprachen aus, die nicht deutsch oder englisch sind.

Tabelle: language
```

TIPPS & TRICKS

Cheatsheet - Auf dem System zurechtfinden - 1. Sichtung

Zurechtfinden

```
-- Welche Datenbanken gibt es:
show databases;

-- Welche Tabelle gibt es in einer Datenbank
-- use <datenbankname>
use sakila;
show tables;

## Wie ist die Struktur eine Tabelle / welche Felder
```

```
-- describe <tabellenname> z.B. actor
describe actor;
## Welche Indizies gibt es in einer Tabelle ?
show indexes from actor;

## Alle procedures/functions einer Datenbank auslesen
select * from information_schema.routines where routine_schema = 'sakila';
```

Cheatsheet für Selects

SELECT .. FROM .. WHERE .. ORDER BY .. LIMIT

```
select * from actor;
select feld1,feld2,feld3 from actor;

-- Mit Feldalias für die Ausgabe
select feld1 as ueberschrift_ausgabe,feld as felduebersch rift from actor;
```

WHERE

```
-- WHERE feldname =/like
select * from actor where last_name = 'AKROYD';
select * from actor where last_name like 'A%';
select * from actor where last_name like 'A%' and first_name like 'C%';
select * from actor where (last_name = 'Akroyd' and first_name = 'Christian') or
(last_name = 'Gable' and first_name = 'Christian');

-- WHERE feldname IN / NOT IN
SELECT * FROM actor WHERE first_name IN ('JOE','ED','JENNIFER');
SELECT * FROM actor WHERE last_name NOT IN ('AKROYD','JONES');

-- WHERE feldname < > <= >= BETWEEN

SELECT * FROM actor where actor_id > 100 and actor_id < 150;
select * from actor where actor_id >= 10 and actor_id <= 50;
SELECT * FROM actor where actor_id between 100 and 150;

-- WHERE feldname IS NULL / IS NOT NULL
SELECT * FROM rental WHERE return_date IS NOT NULL;
SELECT * FROM rental WHERE return_date IS NULL;

-- WHERE feldname <= '2022-01-02 10:00:01' (DATUM)
SELECT * FROM sakila.payment WHERE payment_date > '2005-05-24 22:00:01' and
payment_date < '2005-05-25 22:00:01'
```

WHERE -> DATUMSABFRAGE (NUR BEI WENIG DATEN < 1.000.0000 als Richtwert)

```
-- möglichst im where keine Funktion für Feldname
select YEAR(return_date),return_date from rental where YEAR(return_date) <= 2005;
```

DATUMS-FUNKTIONEN

```
select YEAR(return_date),MONTH(return_date),DAY(return_date) from rental;
SELECT date_format(last_update,'%M %Y') as meindatum FROM actor;

-- Unix timestamp
SELECT unix_timestamp('2022-01-04');
SELECT last_update,unix_timestamp(last_update) from actor;
SELECT unix_timestamp(); -- aktuelles Datum als unix timestamp
```

STRING-FUNKTIONEN

```
SELECT concat(title,' ',release_year,' ',description) as listeneintrag FROM film;
SELECT upper(title) from film;
SELECT lower(title) from film;
SELECT lower(substr(description,1,20)) from film;
```

ZÄHLEN / STATISTIK

```
SELECT count(*) as howmany FROM actor;
select count(*) from actor where last_name like 'D%';

-- Höchste Kosten (MAX), Geringste Kosten (MIN) anzeigen
SELECT MAX(replacement_cost) as am_teuersten,MIN(replacement_cost) FROM sakila.film;
```

SORTIERUNG / LIMIT

```
-- Mit Bedingung und Sortierreihenfolge
select FELD from TABELLE WHERE BEDINGUNG ORDER BY FELD1,FELD2
select first_name,last_name from actor where last_name like 'A%' order by
first_name,last_name

-- Nur Limit
select FELD from tabelle limit 0,30; -- Ab Datensatz 1 der Ergebnismenge
select * from actor limit 30 -- Ab Datensatz

-- WHERE/ORDER BY/LIMIT
SELECT feld FROM tabelle WHERE bedingung ORDER BY FELD1,FELD2 LIMIT 0,10
SELECT feld FROM tabelle WHERE bedingung ORDER BY FELD1,FELD2 LIMIT 5,10 -- Ab dem 5.
Datensatz der Ergebnismenge
```

RECHNEN / RUNDEN

```
SELECT amount,1 as 'Preiserhoehung um',amount + 1 as preiserhoehung FROM payment;

-- Kaufmännisch runden
SELECT ROUND(1.56,1);
-- Abschneiden
SELECT truncate(1.56,1);
-- FLOOR() -> abrunden zum nächsten Integer
SELECT FLOOR(12.56);
-- CEIL() - Aufrunden zum nächsten Integer
SELECT CEIL(12.3);
```

GROUP (GRUPPIEREN)

```
SELECT last_name,COUNT(last_name) as cnt FROM actor GROUP BY last_name;

-- Nachnamen Gruppieren und alle Nachnamen anzeigen die mehr als 2 Mal vorkommen
SELECT last_name, COUNT(last_name)
FROM sakila.actor
GROUP BY last_name
HAVING count(last_name) > 2
```

JOINS

```
-- JOIN über 2 Tabellen
SELECT a.*,c.* FROM customer c JOIN address a ON c.address_id = a.address_id;
SELECT c.first_name,c.last_name,a.postal_code FROM customer c JOIN address a ON
c.address_id = a.address_id;

-- JOIN über 3 Tabellen

-- Schritt 1: 2 Tabellen
SELECT a.last_name,fa.film_id FROM actor a JOIN film_actor fa ON a.actor_id =
fa.actor_id;

-- Schritt 2: auf 3 Tabellen erweitern
SELECT a.last_name,fa.film_id,f.title
FROM actor a
JOIN film_actor fa
ON a.actor_id = fa.actor_id
JOIN film f
ON fa.film_id = f.film_id;
```

Documentation