




 **Name:** Programmierung von Datenbanken - SS2022

 **Kursbereiche:** SoSe 22

 **Kurskontakt:** Patrick Bonack, Emmanuela Georgoula, Lanouar Dominik Jaouani, Robert Macu, Anastasia Navodkina, Chantal Schneevoigt, Andreas Scholl, Margarita Syrtlanova, Oliver Theobald, Karsten Tolle, Adrian Welcker

Vorlesung 1

Teil:PDB-1

Dr. Karsten Tolle – PDB – SS 2022



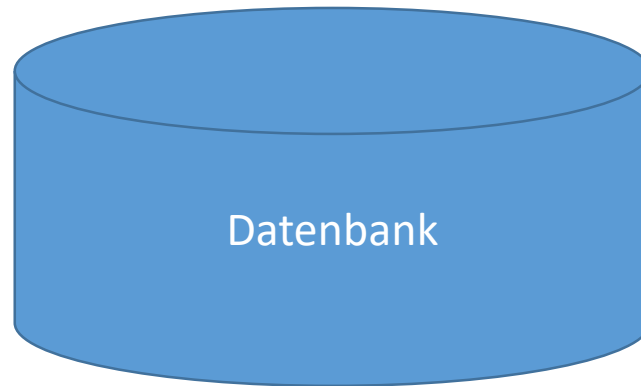
Inhalt heute ...

- Motivation und Begriffe
- Tools und Links
- SQL (survival pack)
 - create database (datenbank anlegen)
 - create table (Tabelle erzeugen)
 - insert into (Einfügen)
 - select (Anfragen)
 - Struktur
 - where-Bedingungen

Datenbank

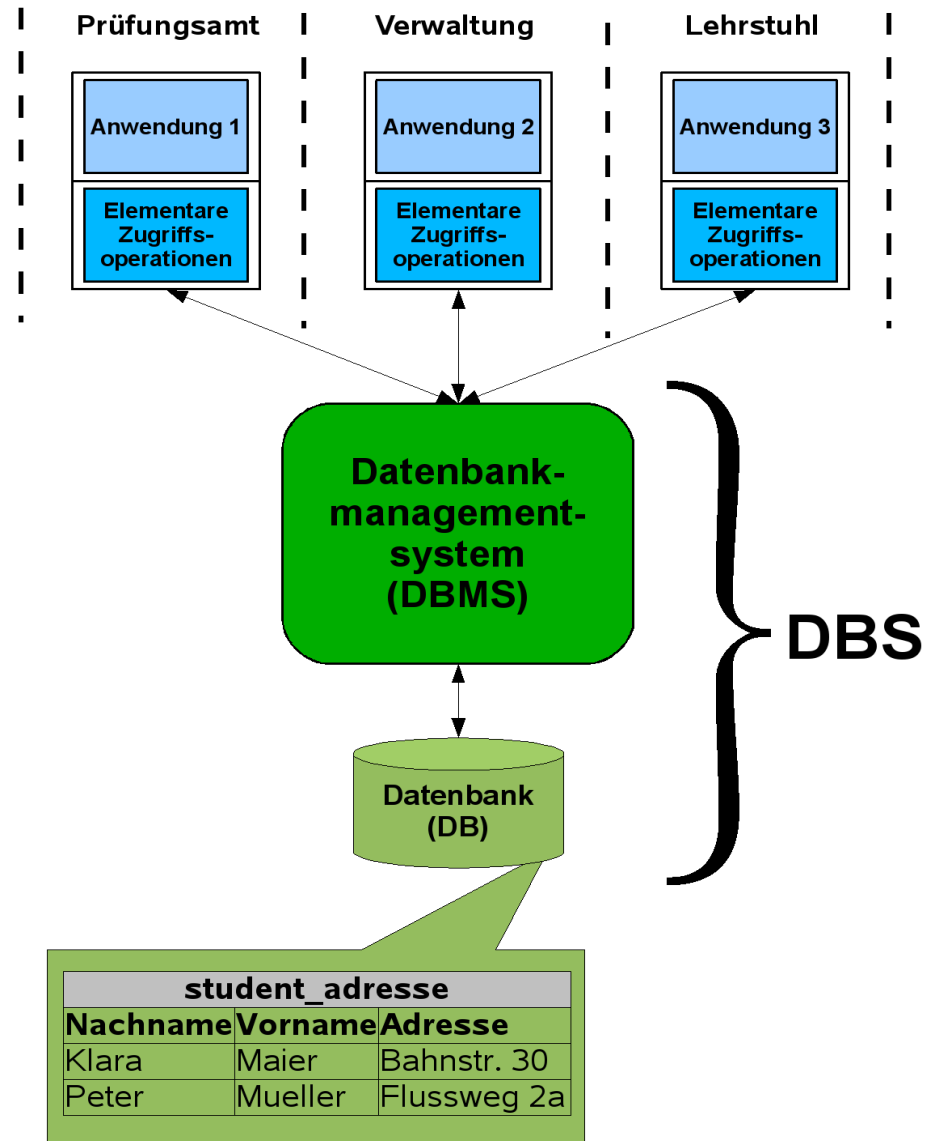
Definition (Duden):

Elektronisches System, in dem große Mengen an Daten zentral gespeichert werden können.



Was ist der Unterschied einer Datenbank zu einer Festplatte?

- Datenbank-management-system (DBMS)
- Datenbank (DB)
- Datenbanksystem (DBS)

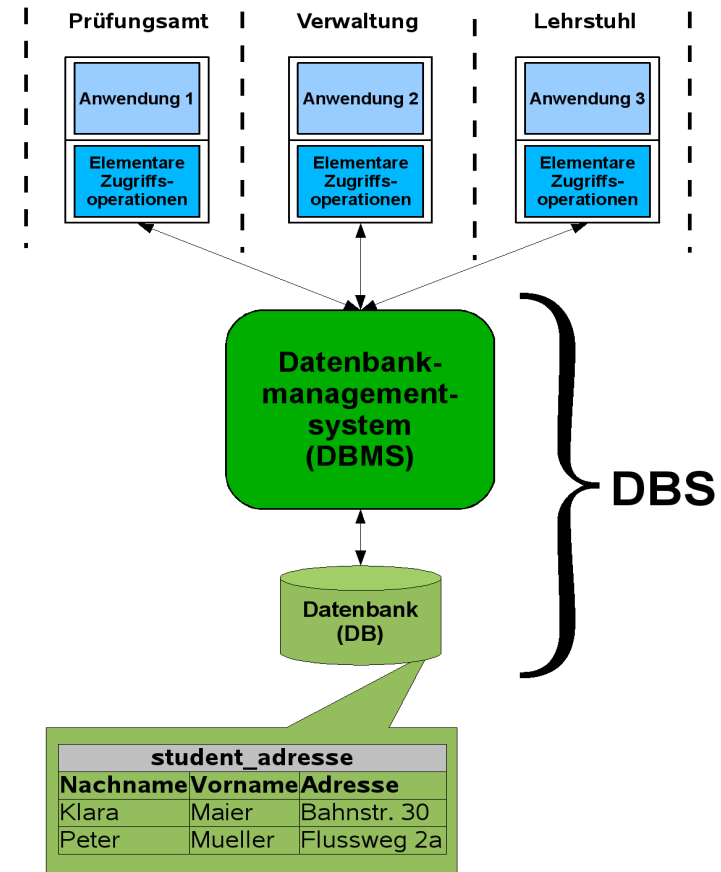


Fragen um herauszufinden, ob man ein DBMS benötigt:

1. Liegt eine **große Datenmenge** vor, die schwer zu managen ist?
 - Braucht es lange das Dokument zu öffnen?
 - Ist das Betrachten der Daten schwierig, muss man viel scrollen und ist es schwer die gesuchten Daten zu finden?
2. Arbeiten **verschiedene Personen/Anwendungen** mit den Daten?
3. Gibt es weitere Daten, die mit den gegebenen Daten in Relation stehen und ebenfalls gespeichert werden?
 - Ziehen Änderungen an einer Stellen Änderungen an anderen Stellen nach sich?
4. Werden die gleichen Daten an **unterschiedlichen Orten** verwendet?

Vorteile DBS

- **Redundanz und Inkonsistenz**
Können durch die zentrale Datenverwaltung und Datenhaltung vermieden werden.
- **Sicherheit gegen Datenmissbrauch**
Durch die zentrale Benutzerverwaltung können Zugriffsrechte gut kontrolliert werden.
- **Datenkonsistenz auch bei Ausfall**
Durch Recovery-Strategien kann sicher gestellt, dass auch nach einem unerwartetem Ausfall die Daten konsistent bleiben (Transaktionen).
- ... und weitere!



Erstellung einer Datenbank: Erster Schritt ...

- Was sind die Anforderungen?
- Was sind die Ziele?
- Ist bekannt was gespeichert werden soll?

→ **Design des *Datenmodels***

Erstellung einer Datenbank:

Zweiter Schritt ...

- Welches DBMS wird genutzt?
- Wer soll wie auf die Daten zugreifen?

→ **Umsetzung des *Datenmodels* im *DBMS***

- Wir verwenden in PDB:
 - **relationales** DBMS – insb. MariaDB
 - Zugriff über die Anfragesprache **SQL**



Relationales Datenbankmodell

E.F. Codd, 1970 (Grundbegriffe)

- **Tabellen** mit Zeilen und Spalten um die Daten darzustellen.

employee

Attribute

Tupel

EMPNO	FIRSTNME	LASTNME	PHONENO	SALARY
001	Jon	Lucas	2983	2000
003	Jon	Smith	2980	3588
103	Lucas	Jon	4444	3980
999	Jon	Smith	3987	1500

Schema bzw. Relationenschema:

employee (EMPNO, FIRSTNME, LASTNME, PHONENO, SALARY)

Downloads

- **MariaDB:** <https://mariadb.org/>
Für die Übungsblätter gilt als Grundlage
Version 10.6.7:

<https://mariadb.com/de/downloads/>

für Mac User entweder zurück zu Version 10.1.44

https://mariadb.org/download/?t=mariadb&o=true&p=mariadb&r=10.1.44&os=macosx&cpu=x86_64

oder (BESSER) über Homebrew (siehe Video „Installation von MariaDB unter macOS via Homebrew“ im Moodle-Kurs) ... dort sogar Version 10.7.3 😊



- (alternativ) MySQL: <https://dev.mysql.com/downloads/mysql/>
Community Server

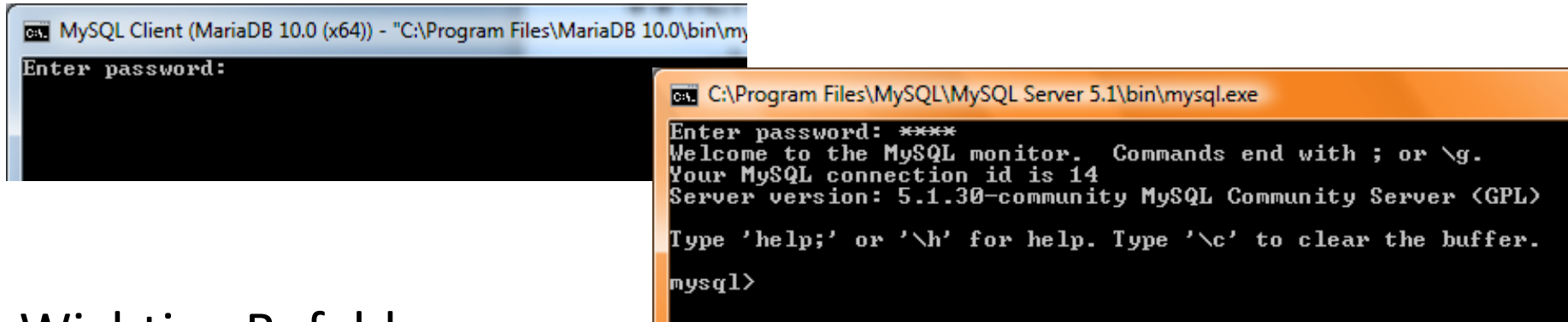
Workbench extra

<https://dev.mysql.com/downloads/workbench/>

(oder als Packet)



Command Line Client



```
MySQL Client (MariaDB 10.0 (x64)) - "C:\Program Files\MariaDB 10.0\bin\my
Enter password:

C:\Program Files\MySQL\MySQL Server 5.1\bin\mysql.exe
Enter password: *****
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 14
Server version: 5.1.30-community MySQL Community Server (GPL)
Type 'help;' or '\h' for help. Type '\c' to clear the buffer.
mysql>
```

- Wichtige Befehle:
 - show databases; -- zeigt die Datenbanken an
 - **create database <database_name>; -- erzeugt eine leere DB**
 - drop database <database_name>; -- löscht eine DB
 - use <database_name>; -- erzeugt eine Verbindung zur Datenbank
 - show tables; -- zeigt die Tabellen der Datenbank an
 - explain <table_name>; -- gibt Informationen über die Tabelle
 - show variables; -- zeigt die aktuellen Einstellungen an

HeidiSQL (geht auch mit MySQL)

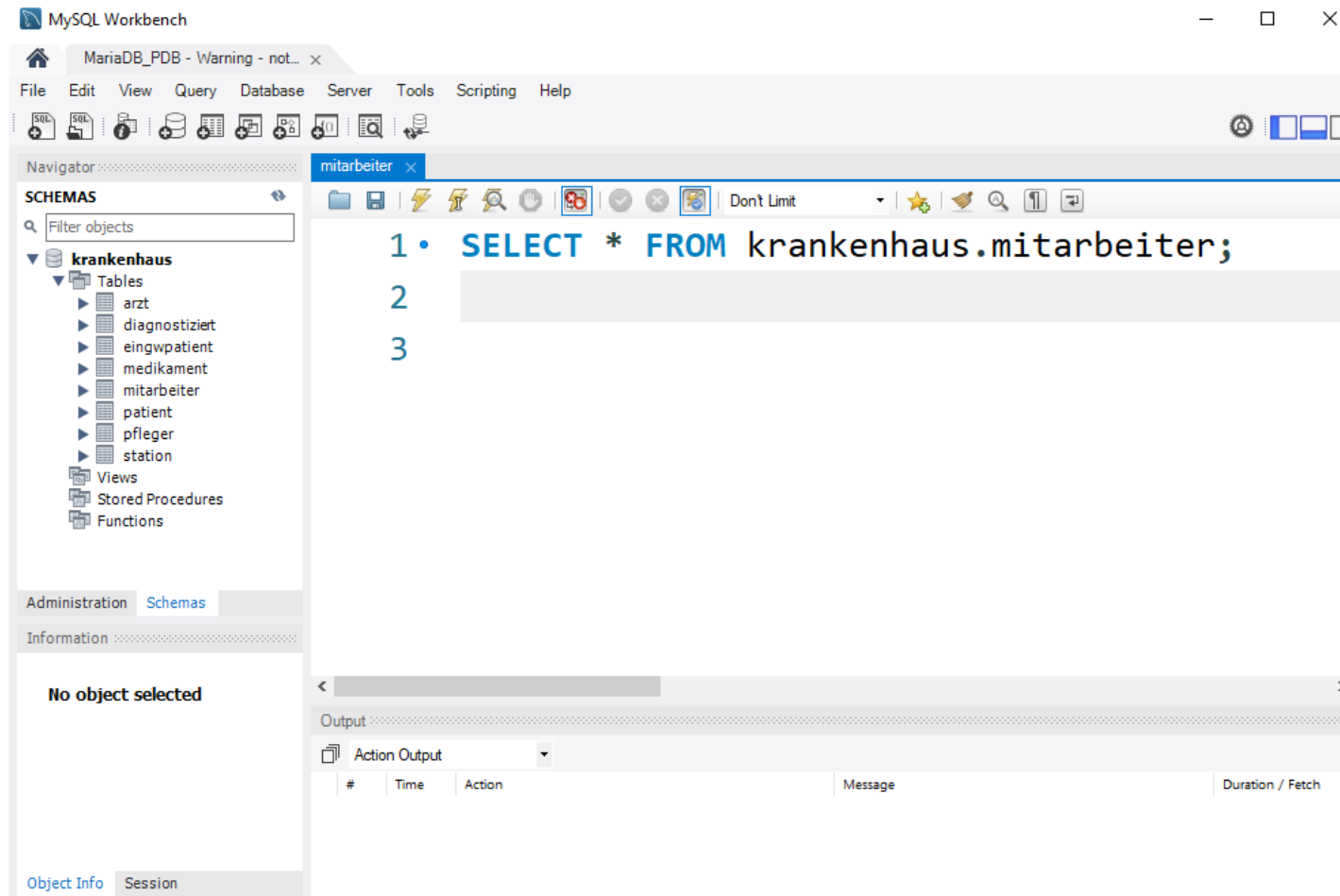
The screenshot displays the HeidiSQL application window, titled "Verbindungsmanager". The left sidebar shows a list of connections, with "SS2022_PDB" selected. The main panel shows the configuration for this connection: "Verbindungstyp" is set to "MariaDB or MySQL (TCP/IP)", "Library" is "libmariadb.dll", "Hostname / IP" is "127.0.0.1", "Benutzername" is "root", and "Passwort" is empty. The "Datenbanken" section shows a tree view with "SS2022_PDB" expanded, revealing "information_schema", "krankenhaus" (16,0 KiB), "mysql", "performance_schema", and "sys". The "krankenhaus" database is further expanded to show a "test" table (16,0 KiB). The bottom toolbar includes buttons for "Neu", "Speichern", "Löschen", and "Öffnen".

The right pane shows the SQL editor with the following code:

```
1 CREATE DATABASE krankenhaus;  
2  
3 CREATE TABLE test (id INT);
```

The status bar at the bottom indicates the current host is "127.0.0.1" and the selected database is "Datenbank: krankenhaus".

MySQL Workbench (geht auch mit MariaDB)



SQL-Online-Tutorial

- <http://sqlzoo.net/>



SELECT basics
quiz
SELECT from world
quiz
SELECT from nobel
quiz
SELECT in SELECT
quiz
SUM and COUNT
quiz
JOIN
quiz
More JOIN
quiz
Using NULL
quiz
Self JOIN
quiz

Reference

Tools

SQL Tutorial

New server in place 16th Jan 2020. Report errors to sqlzoo.qa@gmail.com The old server is still available at <https://old.sqlzoo.net/>

Tutorials: Learn SQL in stages

0 SELECT basics

Some simple queries to get you started

1 SELECT name

Some pattern matching queries

2 SELECT from World

In which we query the World country profile table.

3 SELECT from Nobel

Additional practice of the basic features using a table of Nobel Prize winners.

4 SELECT within SELECT

In which we form queries using other queries.

5 SUM and COUNT

In which we apply aggregate functions. [more the same](#)

6 JOIN

In which we join two tables; game and goals. [previously music tutorial](#)

7 More JOIN operations

In which we join actors to movies in the Movie Database.

8 Using Null

In which we look at teachers in departments. [previously Scottish Parliament](#)

8+ Numeric Examples

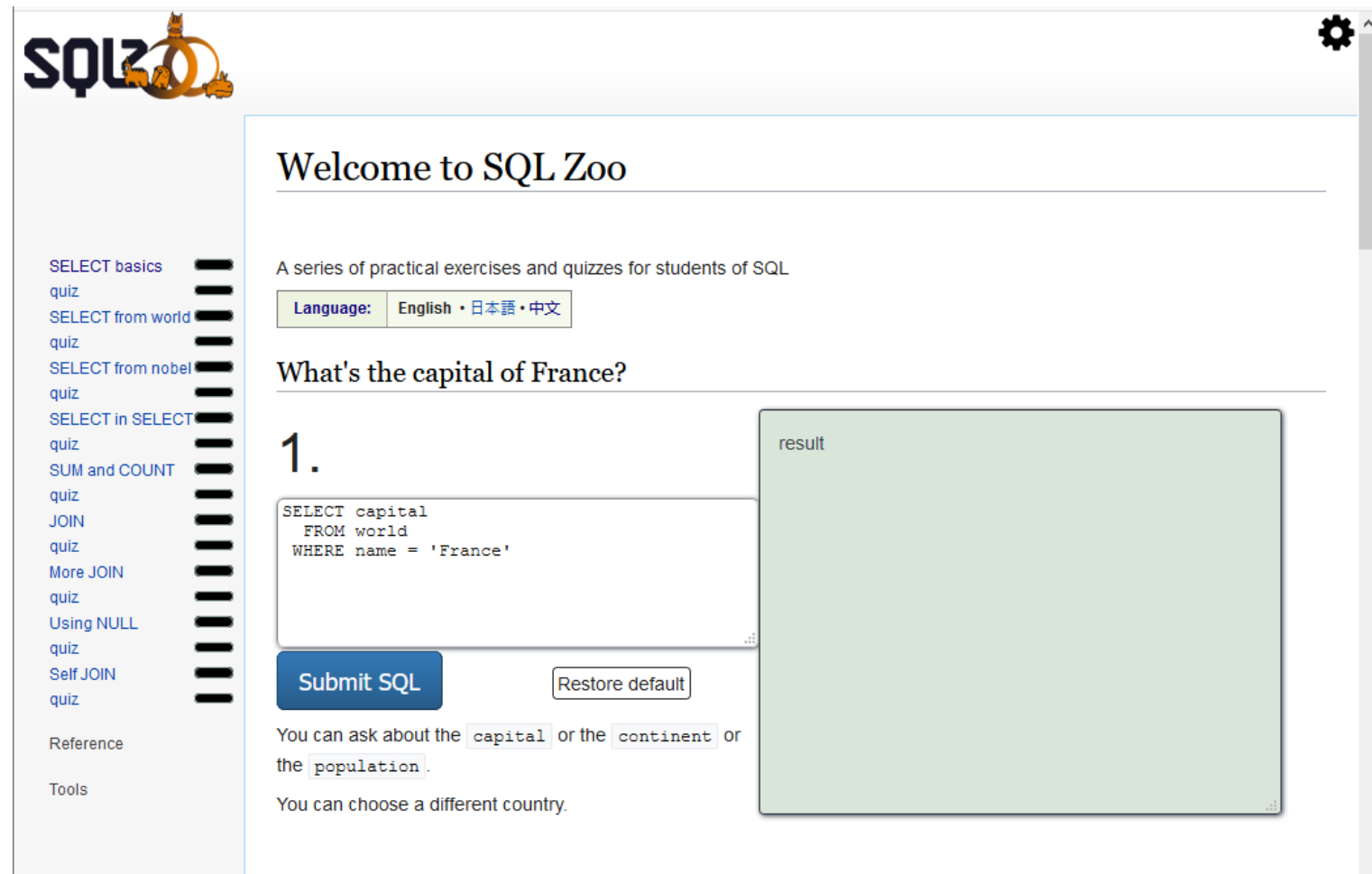
In which we look at a survey and deal with some more complex calculations.

9 Window function



SQL Online Tutorial

- Anfragen können interaktiv ausgeführt werden.



The screenshot shows the SQL Zoo website. The header features the SQL Zoo logo and a settings gear icon. A left sidebar contains a list of topics with corresponding progress bars: SELECT basics, SELECT from world, SELECT from nobel, SELECT in SELECT, SUM and COUNT, JOIN, More JOIN, Using NULL, Self JOIN, Reference, and Tools. The main content area is titled 'Welcome to SQL Zoo' and describes a series of practical exercises and quizzes. A language selector shows 'English' as the current language, with options for '日本語' and '中文'. The current quiz question is 'What's the capital of France?'. It includes a text input field with the SQL query:

```
SELECT capital
FROM world
WHERE name = 'France'
```

, a 'Submit SQL' button, and a 'Restore default' button. To the right of the input field is a large green box labeled 'result'. Below the input field, there is explanatory text: 'You can ask about the capital or the continent or the population.' and 'You can choose a different country.'



SQL – Web Links

- **SQL Online ausprobieren:** <http://www.w3schools.com/sql/>
und <http://sqlzoo.net/>
- **Mobile Apps, z.B. Learn SQL**
<https://play.google.com/store/apps/details?id=com.sololearn.sql&hl=de>
<https://itunes.apple.com/de/app/learn-sql/id953775305?mt=8>
- **Literatur – eBooks allgemein:** http://www.ub.uni-frankfurt.de/datenbanken/ebooks_gesamt.html

Structured Query Language

- SQL ist für Relationale Datenbanksysteme!

Standards:

- SQL-1 von 1986 bzw. 1989 (ca. 120 Seiten)
- SQL-2 (SQL92) von 1992 (ca. 580 Seiten)
<http://www.contrib.andrew.cmu.edu/~shadow/sql/sql1992.txt>
- SQL-3 (SQL99) von 2000 (ca. 1200 Seiten)
- SQL 2003 - ISO/IEC 9075:2003
- SQL:2006 - ISO/IEC 9075-14:2006 (SQL/XML)
- ...

Data Definition Language (DDL) und Data Manipulation Language (DML)



PRODUCTS SERVICES PRICING

Knowledge Base » MariaDB Server Documentation » SQL Statements & Structure » SQL Statements

Home

Open Questions

MariaDB Server

MariaDB MaxScale

MariaDB ColumnStore

Connectors

Ask a question here

View 9 questions

Localized Versions

- Comandi SQL [it]
- Comandos SQL [es]
- Các lệnh SQL [vi]
- Команды языка SQL [ru]
- Comandos SQL [pt]

adbd.com/kb/en/data-definition/

SQL Statements

Complete list of SQL statements for data definition, data manipulation, etc.



Account Management SQL Commands

CREATE/DROP USER, GRANT, REVOKE, SET PASSWORD etc.



Administrative SQL Statements

SQL statements for setting, flushing and displaying server variables and resources.



Data Definition

SQL commands for defining data, such as ALTER, CREATE, DROP, RENAME etc.



Data Manipulation

SQL commands for querying and manipulating data, such as SELECT, UPDATE, DELETE etc.



Prepared Statements

Prepared statements from any client using the text based prepared statement interface.



Programmatic & Compound Statements

Compound SQL statements for stored routines and in general.



Stored Routine Statements

SQL statements related to creating and using stored routines.



Table Statements

Documentation on Creating, Altering, Analyzing and Maintaining Tables.

<https://mariadb.com/kb/en/sql-statements/>

Datenbank anlegen ...

`create database <name>;`

`# alternativ: create schema <name>;`

`create database krankenhaus;`

`# create schema krankenhaus;`

Tabellen erstellen

Eine Tabelle wird im Minimalfall mit ihrem eindeutigen Namen sowie der Liste der zugehörigen Attribute samt Domänen nach folgendem Schema definiert:

```
create table Relations-Name (  
    Attribut-Name Domäne { , Attribut-Name Domäne}*   
);
```

z.B.: `create table test (id int);`

```
create table kunde (  
    name varchar(30),  
    vorname varchar(20),  
    strasse varchar(50),  
    stadt varchar(25),  
    kinder int,  
    gebdatum date  
);
```

CREATE TABLE

Syntax

```
CREATE [OR REPLACE] [TEMPORARY] TABLE [IF NOT EXISTS] tbl_name  
    (create_definition,...) [table_options]... [partition_options]  
CREATE [OR REPLACE] [TEMPORARY] TABLE [IF NOT EXISTS] tbl_name  
    [(create_definition,...)] [table_options]... [partition_options]  
    select_statement  
CREATE [OR REPLACE] [TEMPORARY] TABLE [IF NOT EXISTS] tbl_name  
    { LIKE old_table_name | (LIKE old_table_name) }  
  
select_statement:  
    [IGNORE | REPLACE] [AS] SELECT ... (Some legal select statement)
```

<https://mariadb.com/kb/en/create-table/>

die wichtigsten-SQL-Datentypen (Domänen)

- integer/int
- **double(m,d)**
- **float(m,d)**
- **decimal(m,d)**
- char(*n*)
- varchar(*n*)
- text
- date
- time
- datetime
- timestamp
- clob(*n*)
- blob(*n*)
- ...

Siehe auch: <https://mariadb.com/kb/en/data-types/>

Einfügen von Tupeln I

Um Daten einzufügen, spezifiziert man das Tupel, welches eingefügt werden soll ...

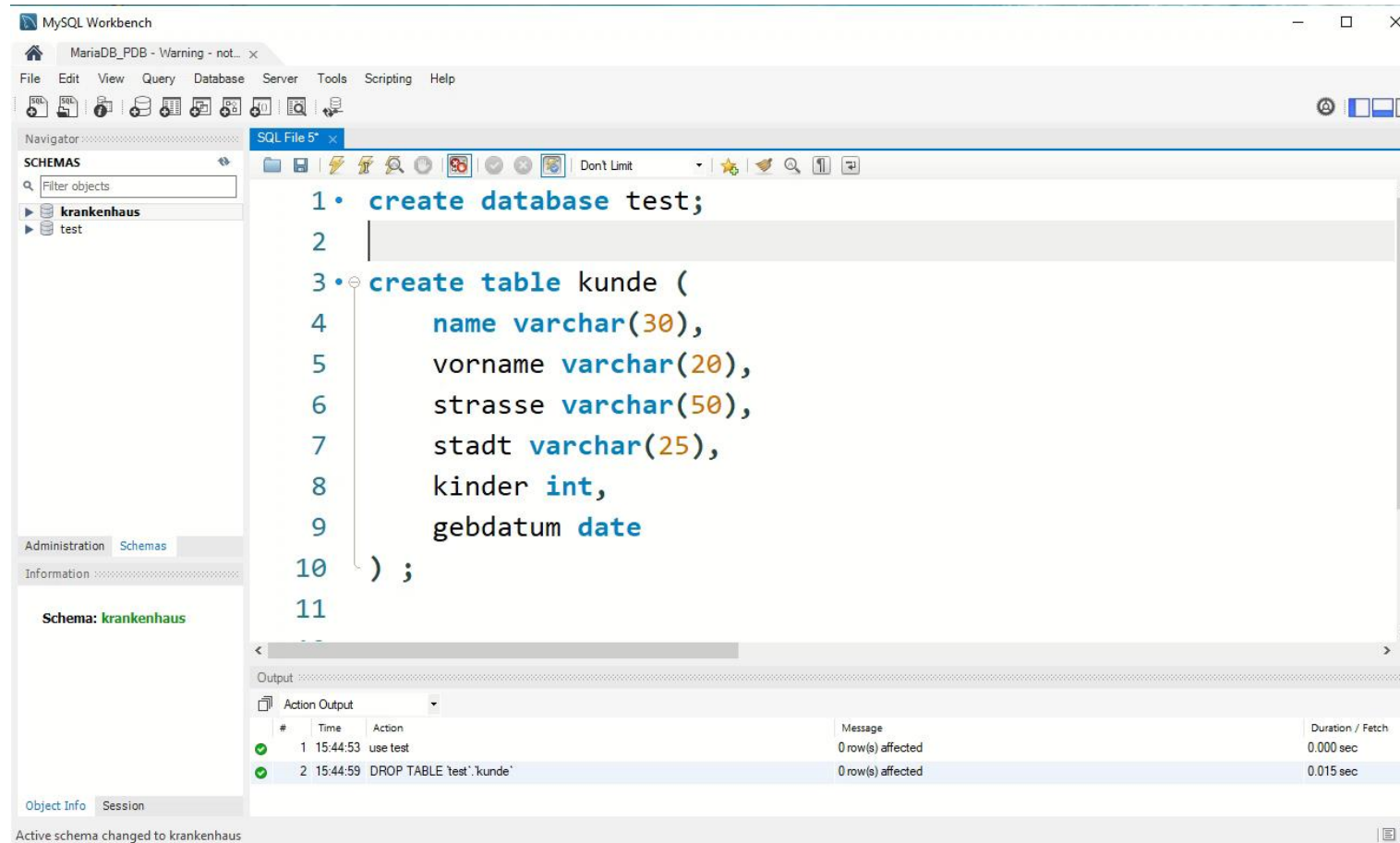
Die Werte für die Attribute der Tupel müssen aus der Domäne (Definitionsbereich) der Attribute sein.

insert into kunde

values

('Otto', 'Hans', 'Bäckerweg 12', 'Frankfurt',
3, '1970-12-01') ;

→ Der Kunde "Hans Otto" wird eingefügt.



Einfügen von Tupeln II

... unbekannte Werte können mit **NULL-Values** befüllt werden:

```
insert into kunde  
values
```

```
('Otto', 'Hans', null, 'Frankfurt', 3, '1970-12-01') ;
```

→ Der Kunde "Hans Otto" wird eingefügt.

Einfügen von Tupeln III

... alternativ kann man die Attribute angeben, die gesetzt werden sollen:

```
insert into kunde (vorname, name, gebdatum)  
values ('Hans', 'Otto', '1970-12-01') ;
```

→ Der Kunde "Hans Otto" wird eingefügt.

... welche Version ist besser?

1. **insert into** kunde
 values ('Otto', 'Hans', null, null, null, '1970-12-01') ;
2. **insert into** kunde (vorname, name, gebdatum)
 values ('Hans', 'Otto', '1970-12-01') ;

Einfügen von Tupeln IV

... es können auch mehrere Datensätze mit einem SQL-Statement eingefügt werden:

```
insert into kunde (vorname, name, gebdatum)  
values
```

```
    ('Hans', 'Otto', '1970-12-01'),
```

```
    ('Hans', 'Otto', '1925-11-24'),
```

```
    ('Hans', 'Otto', '2012-12-24') ;
```

→ Der Kunde "Hans Otto" wird 3-mal eingefügt.

Anfragen ohne Bedingungen

kunde (name, vorname, strasse, stadt)

select name, vorname **from** kunde;

select vorname, name **from** kunde;

select stadt **from** kunde;

select * from kunde;

Nicht in Programmen!

SQL verwirklicht das Prinzip der „Vielfachmenge“ (engl. multiset). In den Ergebnismengen können demnach Duplikate auftreten.

Sind keine Duplikate erwünscht, müssen sie explizit durch den Zusatz **distinct** entfernt werden.

select distinct stadt from kunde;

where-Klausel

Bezüglich der Bedingung sind Vergleiche mit den üblichen Operatoren, den logischen Verknüpfungen **and** und **or** sowie Klammerungen gestattet.

kunde (name, vorname, strasse, stadt, kinder)

```
select * from kunde  
where kinder > 0 or stadt = 'Frankfurt' and  
name = 'Otto' ;
```

kunde (name, vorname, strasse, stadt, kinder)

```
select * from kunde  
where kinder > 0 or stadt = 'Frankfurt' and  
name = 'Otto' ;
```



```
select * from kunde  
where kinder > 0 or (stadt = 'Frankfurt' and  
name = 'Otto');
```

Ausführungsreihenfolge: erst AND dann OR ... aber besser lesbar mit Klammern!