

Informatik 10 - Datenbanken (Teil 1)

#### Informatik 10 - Datenbanken (Teil 1)

Stunde 1+2 Wdh: Klassen und Objekte

Stunde 3+4

Wdh: Aufbau von (relationalen) Datenbanken

**SQL Spickzettel** 

Übung: SQL Island

Stunde 5+6

SQL Puzzle

Wdh: SQL Basics

Stunde 7+8

Tabellenbeziehungen: Fremdschlüssel Tabellenbeziehungen im Klassendiagramm Kardinalitäten



Stunde 9+10



SQL: Tabellen verbinden Kreuzprodukt / Join Join Beispiel

Stunde 11+12 Join Beispiel



SQL mit Kreuzprodukt und Join

#### Outline

Stunde 1+2

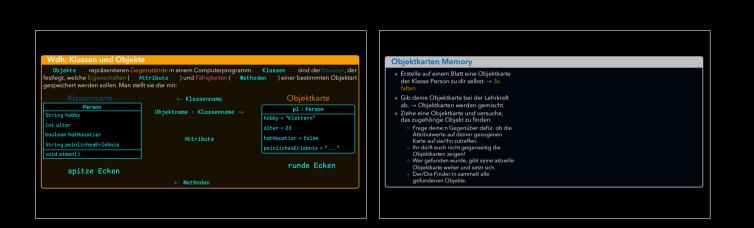
tunde 3+4

Stunde 5+6

Stunde 7+8

stunde 9+10

Stunde 11+12



**Objekte** 

repräsentieren **Gegenstände** in einem Computerprogramm. **Klassen** sind der <mark>Bauplan</mark>, der **Eigenschaften** ( ) einer bestimmten Objektart

festlegt, welche **Eigenschaften** ( gespeichert werden sollen. Man stellt sie dar mit:

erden sonen. Man stent sie dar inn

Klassenkarte

**Objekte** 

repräsentieren **Gegenstände** in einem Computerprogramm. Klassen sind der Bauplan, der festlegt, welche Eigenschaften ( Attribute ) und Fähigkeiten ( ) einer bestimmten Objektart

gespeichert werden sollen. Man stellt sie dar mit:

Klassenkarte

L

**Objekte** repräsentieren **Gegenstände** in einem Computerprogramm. **Klassen** sind der **Bauplan**, der festlegt, welche **Eigenschaften** ( **Attribute** ) und **Fähigkeiten** ( **Methoden** ) einer bestimmten Objektart gespeichert werden sollen. Man stellt sie dar mit:

Klassenkarte

L

**Objekte** repräsentieren **Gegenstände** in einem Computerprogramm. **Klassen** sind der **Bauplan**, der festlegt, welche **Eigenschaften** ( **Attribute** ) und **Fähigkeiten** ( **Methoden** ) einer bestimmten Objektart gespeichert werden sollen. Man stellt sie dar mit:

#### Klassenkarte

# Person String hobby int alter boolean hatHaustier String peinlichesErlebnis void atmen()

L

**Objekte** repräsentieren **Gegenstände** in einem Computerprogramm. **Klassen** sind der **Bauplan**, der festlegt, welche **Eigenschaften** ( **Attribute** ) und **Fähigkeiten** ( **Methoden** ) einer bestimmten Objektart gespeichert werden sollen. Man stellt sie dar mit:

#### Klassenkarte

# Person String hobby int alter boolean hatHaustier String peinlichesErlebnis void atmen()

spitze Ecken

**Objektkarte** 

**Objekte** repräsentieren **Gegenstände** in einem Computerprogramm. **Klassen** sind der **Bauplan**, der festlegt, welche **Eigenschaften** ( **Attribute** ) und **Fähigkeiten** ( **Methoden** ) einer bestimmten Objektart gespeichert werden sollen. Man stellt sie dar mit:

#### Klassenkarte

#### Person

String hobby

int alter

void atmen()

boolean hatHaustier

String peinlichesErlebnis

spitze Ecken

 $\leftarrow$  Klassenname

 $\overline{\text{Objektname}}: \mathsf{Klassenname} \to$ 

Attribute

Attribute

 $\leftarrow$  Methoden

**Objekte** repräsentieren **Gegenstände** in einem Computerprogramm. **Klassen** sind der **Bauplan**, der festlegt, welche **Eigenschaften** ( **Attribute** ) und **Fähigkeiten** ( **Methoden** ) einer bestimmten Objektart gespeichert werden sollen. Man stellt sie dar mit:

#### Klassenkarte

# Person String hobby int alter boolean hatHaustier String peinlichesErlebnis void atmen()

 $\leftarrow$  Klassenname

Objektname : Klassenname  $\rightarrow$ 

Attribute

## **Objektkarte**

p1 : Person hobby = "Klettern"

alter = 23

hatHaustier = false

peinlichesErlebnis = "..."

spitze Ecken

← Methoden

**Objekte** repräsentieren **Gegenstände** in einem Computerprogramm. **Klassen** sind der Bauplan, der festlegt, welche Eigenschaften ( Attribute ) und Fähigkeiten ( Methoden ) einer bestimmten Objektart gespeichert werden sollen. Man stellt sie dar mit:

#### Klassenkarte

# Person String hobby int alter boolean hatHaustier String peinlichesErlebnis void atmen()

 $\leftarrow$  Klassenname

Objektname : Klassenname  $\rightarrow$ 

Attribute

# **Objektkarte**

p1 : Person

hobby = "Klettern"

alter = 23

hatHaustier = false

peinlichesErlebnis = "..."

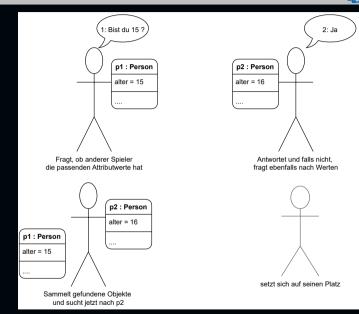
runde Ecken

spitze Ecken

← Methoden

# **Objektkarten Memory**

- Erstelle auf einem Blatt eine Objektkarte der Klasse Person zu dir selbst. → 3x
   falten
- Gib deine Objektkarte bei der Lehrkraft ab. → Objektkarten werden gemischt.
- Ziehe eine Objektkarte und versuche, das zugehörige Objekt zu finden.
  - Frage deine:n Gegenüber dafür, ob die Attributwerte auf deiner gezogenen Karte auf sie/ihn zutreffen.
  - Ihr dürft euch nicht gegenseitig die Objektkarten zeigen!
  - Wer gefunden wurde, gibt seine aktuelle Objektkarte weiter und setzt sich.
  - Der/Die Finder:in sammelt alle gefundenen Objekte.



#### **Outline**

Stunde 1+2

Stunde 3+4

Stunde 5+6

Stunde 7+8

Stunde 9+10

tunde 11+12









SQL Spickwentel

Folgonier SQ Spickwentel erstellt als 102 Goodlegen der 1 Status for der frankt fin fin der Sch Aufgeben

Seiner Ger und Vorligenspiele (E) dem beseich der Spickwent des spires 197 Filmen.



#### Wdh: Von der Klasse zur Tabelle



- Zeichnet zu zweit eine Tabelle, in der man alle Objekte der Klasse Person sammeln kann.
- Tragt eure beiden Objekte (vom Objektkarten-Memory) in die Tabelle ein.
- Ordnet die folgenden Begriffe den Teilen der Tabelle zu.
   Achtung: Nicht alle Begriffe passen und manches hat mehrere Begriffe!
   Datensatz Tabelle Zelle Klasse Objekt Parameter Attribut Spalte Feld Methode Board Zeile Datentyp Attributwert

#### Wdh: Von der Klasse zur Tabelle



- Zeichnet zu zweit eine Tabelle, in der man alle Objekte der Klasse Person sammeln kann.
- Tragt eure beiden Objekte (vom Objektkarten-Memory) in die Tabelle ein.
- Ordnet die folgenden Begriffe den Teilen der Tabelle zu.

Achtung: Nicht alle Begriffe passen und manches hat mehrere Begriffe!

Datensatz Tabelle Zelle Klasse Objekt Parameter Attribut Spalte Feld Methode Board Zeile Datentyp Attributwert

#### Lösung:

# Attribut/ Feld/

#### **Spaltenname**

| name                | alter | groesse | geschlecht | brille | ••• |
|---------------------|-------|---------|------------|--------|-----|
| Herrmann            | 24    | 1.62    | m          | false  |     |
| Zelle/ Attributwert |       |         |            |        |     |

Tabelle

Klasse/ Spaltennamen

Datensatz/Zeile/ Objekt

Nicht verwendete Begriffe: Parameter, Methode, Board, Datentyp

 $Feld: \ Wird \ oft \ synonym \ zu \ Attribut \ verwendet, \ v.a. \ in \ Programmen \ wie \ LibreOffice \ Base \ oder \ MS \ Access.$ 

Datenbanken speichern Datensätze in Die repräsentieren die Attribute (Synonym: Feld) und bilden zusammen eine Klasse. Die (=Zeilen) entsprechen

(oft

Objekten und in den Spalten stehen die Attributwerte. Jede Tabelle hat einen auch "ID"), der Datensätze eindeutig identifiziert. Oft werden die Datensätze hiermit einfach durchnummeriert. Im Tabellenschema wird er unterstrichen und im Klassendiagramm immer als erstes Attribut aufgelistet.

Der Aufbau einer Tabelle kann mit oder dargestellt werden. Dessen Aufbau ist: TABELLENNAME(Datentyp Primärschlüssel, Datentyp Spalte1, Datentyp Spalte2, ...)

Zum Beispiel:

Datenbanken speichern Datensätze in Tabellen . Die repräsentieren die Attribute (Synonym: Feld) und bilden zusammen eine Klasse. Die (=Zeilen) entsprechen

(oft

Objekten und in den Spalten stehen die Attributwerte. Jede Tabelle hat einen auch "ID"), der Datensätze eindeutig identifiziert. Oft werden die Datensätze hiermit einfach durchnummeriert. Im

Tabellenschema wird er unterstrichen und im Klassendiagramm immer als erstes Attribut aufgelistet. Der Aufbau einer Tabelle kann mit oder dargestellt werden. Dessen Aufbau ist:

TABELLENNAME(Datentyp Primärschlüssel, Datentyp Spalte1, Datentyp Spalte2, ...) Zum Beispiel:

Datenbanken speichern Datensätze in **Tabellen** . Die **Spaltenüberschriften** repräsentieren die **Attribute** (Synonym: Feld) und bilden zusammen eine **Klasse**. Die (=Zeilen) entsprechen

(oft

**auch "ID")**, der Datensätze eindeutig identifiziert. Oft werden die Datensätze hiermit einfach durchnummeriert. Im Tabellenschema wird er unterstrichen und im Klassendiagramm immer als erstes Attribut aufgelistet.

Der Aufbau einer Tabelle kann mit oder dargestellt werden. Dessen

Aufbau ist:

TABELLENNAME(<u>Datentyp Primärschlüssel</u>, Datentyp Spalte1, Datentyp Spalte2, ...)

Zum Beispiel:

Objekten und in den Spalten stehen die Attributwerte. Jede Tabelle hat einen

Datenbanken speichern Datensätze in **Tabellen** . Die Spaltenüberschriften repräsentieren die Attribute (Synonym: Feld) und bilden zusammen eine Klasse. Die Datensätze (=Zeilen) entsprechen

(oft

Objekten und in den Spalten stehen die Attributwerte. Jede Tabelle hat einen auch "ID"), der Datensätze eindeutig identifiziert. Oft werden die Datensätze hiermit einfach durchnummeriert. Im Tabellenschema wird er unterstrichen und im Klassendiagramm immer als erstes Attribut aufgelistet.

Der Aufbau einer Tabelle kann mit oder dargestellt werden. Dessen Aufbau ist:

TABELLENNAME(Datentyp Primärschlüssel, Datentyp Spalte1, Datentyp Spalte2, ...) Zum Beispiel:

Datenbanken speichern Datensätze in **Tabellen** . Die **Spaltenüberschriften** repräsentieren die **Attribute** (Synonym: Feld) und bilden zusammen eine **Klasse**. Die **Datensätze** (=**Zeilen**) entsprechen

(oft

**auch** "ID"), der Datensätze eindeutig identifiziert. Oft werden die Datensätze hiermit einfach durchnummeriert. Im Tabellenschema wird er unterstrichen und im Klassendiagramm immer als erstes Attribut aufgelistet.

Der Aufbau einer Tabelle kann mit

oder

dargestellt werden. Dessen

Objekten und in den Spalten stehen die Attributwerte. Jede Tabelle hat einen Primärschlüssel

Aufbau ist:

TABELLENNAME(<u>Datentyp Primärschlüssel</u>, Datentyp Spalte1, Datentyp Spalte2, ...)

Zum Beispiel:

Datenbanken speichern Datensätze in **Tabellen** . Die **Spaltenüberschriften** repräsentieren die Attribute (Synonym: Feld) und bilden zusammen eine Klasse. Die Datensätze (=Zeilen) entsprechen

(oft

Objekten und in den Spalten stehen die Attributwerte. Jede Tabelle hat einen Primärschlüssel auch "ID"), der Datensätze eindeutig identifiziert. Oft werden die Datensätze hiermit einfach durchnummeriert. Im Tabellenschema wird er unterstrichen und im Klassendiagramm immer als erstes Attribut aufgelistet.

Der Aufbau einer Tabelle kann mit Klassenkarte oder dargestellt werden. Dessen Aufbau ist:

TABELLENNAME(Datentyp Primärschlüssel, Datentyp Spalte1, Datentyp Spalte2, ...) Zum Beispiel:

Datenbanken speichern Datensätze in **Tabellen** . Die **Spaltenüberschriften** repräsentieren die **Attribute** (Synonym: Feld) und bilden zusammen eine **Klasse**. Die **Datensätze** (=**Zeilen**) entsprechen

(oft

auch "ID"), der Datensätze eindeutig identifiziert. Oft werden die Datensätze hiermit einfach durchnummeriert. Im Tabellenschema wird er unterstrichen und im Klassendiagramm immer als erstes Attribut aufgelistet.

Der Aufbau einer Tabelle kann mit **Klassenkarte** oder **Tabellenschema** dargestellt werden. Dessen Aufbau ist:

TABELLENNAME (Datentyn Primärschlüssel Datentyn Spalte) Datentyn Spalte?

Objekten und in den Spalten stehen die Attributwerte. Jede Tabelle hat einen Primärschlüssel

TABELLENNAME(<u>Datentyp Primärschlüssel</u>, Datentyp Spalte1, Datentyp Spalte2, ...) Zum Beispiel:

TABELLENNAME(Datentyp Primärschlüssel, Datentyp Spalte1, Datentyp Spalte2, ...)

Datenbanken speichern Datensätze in **Tabellen** . Die **Spaltenüberschriften** repräsentieren die **Attribute** (Synonym: Feld) und bilden zusammen eine **Klasse**. Die **Datensätze** (=**Zeilen**) entsprechen **Objekten** und in den Spalten stehen die Attributwerte. Jede Tabelle hat einen **Primärschlüssel** (oft

**auch "ID")**, der Datensätze eindeutig identifiziert. Oft werden die Datensätze hiermit einfach durchnummeriert. Im Tabellenschema wird er unterstrichen und im Klassendiagramm immer als erstes Attribut aufgelistet.

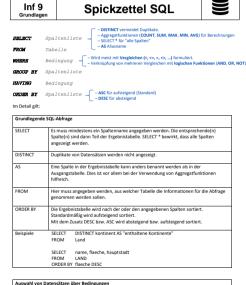
Der Aufbau einer Tabelle kann mit **Klassenkarte** oder **Tabellenschema** dargestellt werden. Dessen Aufbau ist:

Zum Beispiel:
Person(int id, String name, int alter, ...)





Folgender SQL-Spickzettel enthält alle SQL-Grundlagen der 9. Klasse. Ihr dürft (sollt!) ihn bei allen SQL-Aufgaben benutzen. Über das Vorlagensymbol 🛕 oben könnt ihr den Spickzettel als eigenes PDF öffnen.



In der Ergebnistabelle werden nur die Datensätze (Zeilen) angezeigt, welche die angegebene Bedingung erfüllen. Eine Bedingung wird mit einem Vergleich formuliert. Neben den typischen Vergleichsoperatoren wie <, <=, =, <>, usw. sind insbesondere auch IS NULL und UKE wichtig. Mehrere Vergleiche können durch die logischen

|         | Funktionen AND, OR und NOT verknüpft werden. Ggf. müssen die einzelnen<br>Ausdrücke dabei sinnvoll geklammert werden  |  |  |  |
|---------|---|--|--|--|
| LIKE    | Selgiel WHERE jahr > 2015 AND suffset cs 0 AND NO Tisks 18 Kann in einer Bedingung zur Wustererkennung von Einträgen verwendet werden. Folgende zwei Pathahlere (wildcards) werden häufig eingesetzt:   |  |  |  |
|         | % steht für beliebig viele Zeichen, auch keines (* bei MS Access)     _ für genau ein beliebiges Zeichen (? bei MS Access) Beispiele:   |  |  |  |
|         | WHERE titel LIKE "You's" – findet alle Titel die mit "You" beginnen<br>Groß-/Geinschreibung wird nicht berückschtigt WHERE titel LIKE "Sdowes" – findet alle Titel die "You" enthalten WHERE titel LIKE "\_" – findet alle Titel die mit 1 beginnen und genau 4 Zeichen lang sind |  |  |  |
| NULL    | Bedeutet, dass kein Wert in einer Zelle eingetragen ist.  |  |  |  |
| IS NULL | Überprüft (in einer Bedingung), ob kein Wert in einer Zelle eingetragen ist.  |  |  |  |

| Aggregatfunktio | nen   |  |  |
|-----------------|---|--|--|
| AVG             | Berechnet den Durchschnitt aller Werte einer Spalte.                                  |  |  |
| COUNT           | Gibt die Anzahl der Einträge einer Spalte aus.  |  |  |
| MAX bzw. MIN    | Gibt das Maximum bzw. Minimum aller Werte einer Spalte aus.                           |  |  |
| SUM             | Berechnet die Summe aller Werte einer Spalte.   |  |  |
| Beispiel        | SELECT COUNT(*) AS "Anzahl afrikanischer Länder" FROM Land WHERE kontinent = "Afrika" |  |  |

| Gruppierung |   |
|-------------|---|
| GROUP BY    | Datensätze mit demseiben Wert in der angegeben Spalte werden gruppiert.<br>Gruppierungen sind nur in Kombination mit Aggregatfunktionen sinnvoll. |
| HAVING      | An gruppierte Datensätze werden Bedingungen mit HAVING formuliert.  |
| Beispiel    | SELECT fsk, MIN(laufzeit) FROM Film WHERE genere1="filmkomödie" OR genre2="Filmkomödie" GROUP BY fsk HAVING Fsk-16                                |

# SQL keywords should be in **lower case!**



select name, id
from products
where discount = 0
order by price asc;

# Noooo, they must be in **upper case!**

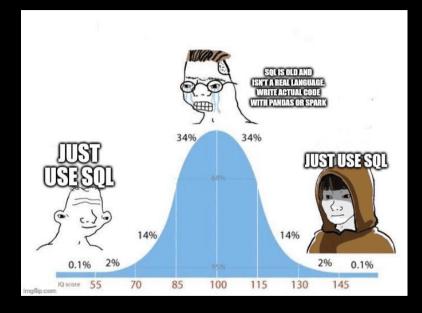


SELECT name, id FROM products WHERE discount = 0 ORDER BY price ASC;



sElEcT nAmE, iD fRoM PrOdUcTs WhErE dIsCoUnT = 0 OrDeR bY pRiCe AsC;

'Sarcastic Query Language' • by u/casperdewith



# Übung: SQL Island



sql-island.informatik.uni-kl.de/

- 1. Was sind die Primärschlüssel der Tabellen, die die einzelnen Objekte eindeutig identifizieren?
- → Notiert das vollständige Tabellenschema der Datenbank von SQL Island (mit Datentypen und Markierung der Primärschlüssel)

2. Stellt die Tabellen der Datenbank mit Klassenkarten dar.

# Übung: SQL Island



sql-island.informatik.uni-kl.de/

- 1. Was sind die Primärschlüssel der Tabellen, die die einzelnen Objekte eindeutig identifizieren?
  - → Notiert das vollständige Tabellenschema der Datenbank von SQL Island (mit Datentypen und Markierung der Primärschlüssel)

 $BEWOHNER(\underline{int\ bewohnernr}\ ,\ String\ name,\ int\ dorfnr,\ String\ geschlecht,\ String\ beruf,\ int\ gold,\ String\ status)$ 

 ${\tt GEGENSTAND}(\underline{{\tt String~gegenstand}},\, {\tt int~besitzer})$ 

DORF(int dorfnr, String name, int haeuptling)

2. Stellt die Tabellen der Datenbank mit Klassenkarten dar.

# Übung: SQL Island



#### sql-island.informatik.uni-kl.de/

- 1. Was sind die Primärschlüssel der Tabellen, die die einzelnen Objekte eindeutig identifizieren?
  - → Notiert das vollständige Tabellenschema der Datenbank von SQL Island (mit Datentypen und Markierung der Primärschlüssel)

BEWOHNER(int bewohnern , String name, int dorfnr, String geschlecht, String beruf, int gold, String status) GEGENSTAND(String gegenstand, int besitzer)

DORF(int dorfnr, String name, int haeuptling)

2. Stellt die Tabellen der Datenbank mit Klassenkarten dar.

| BEWOHNER          |  |  |
|-------------------|--|--|
| int bewohnernr    |  |  |
| String name       |  |  |
| int dorfnr        |  |  |
| String geschlecht |  |  |
| String beruf      |  |  |
| int gold          |  |  |
| String status     |  |  |
|                   |  |  |



| DORF           |  |
|----------------|--|
| int dorfnr     |  |
| String name    |  |
| int haeuptling |  |
|                |  |

# Für Schnelle



Für Schnelle: Spielt SQL Island, der SQL-Spickzettel hilft euch dabei.

#### **Outline**

Stunde 1+2

Stunde 3+4

Stunde 5+6

Stunde 7+8

stunde 9+10

Stunde 11+12

# Zeichnet zu zweit eine Tabelle, in der man alle Obiekte der Klasse Person sammeln kann.

Datembare in perceive in usernade in roots Len. Die Spottenburgsriften representeren in Ammute Spotten stellen die Ersten einer Klasse. Die Datenstätze (-Zeiter) entgeschen Objekten und in den Spotten stehen die Attributwerte. Jede Tabelle hat einem Prästrachtlüssel (oft auch JD<sup>2</sup>), der Datenstätze eindestig identifisiert. Oft werden die Obstenstäte hiermit entlich durchnummenter im Tabellensbericht. Usersusze endeutgigientmiert. Off werden die Usersatze niermt einsten durchtummenert. Im Isbesenschem wird er unterstrichen und im Klassendagramm immer als entes Attibut aufglistet.
Dar Aufbau einer Tabelle kann mit. Klassenkerte. oder Tabellenschese deroestellt werden. Desse FABELLENNAME(Datentyp Primärschlüssel , Datentyp Spaltel, Datentyp Spaltel, ...)

Folgender SCL-Spickzettel enrhält alle SCL-Grundlagen der 9. Klasse. Ihr dürft (sollt) ihn bei allen SCL-Aufgaben berutzen. Über das Vorlagersymbol. [2] oben közert ihr den Spickzettel als eigenes PDF öffinen.



#### SOL Puzzle In dieser Aufgabe geht es immer um die Tabelle land, deren erste Datensätze du hier siehst: einwohner flaeche hauptstadt name 83.24 358 Berlin Deutschland Frankreich 544 Paris Brasilien 212.60 Rio de Janeiro

Lösung:

#### Wdh: SQL Basics

Bearbeite die Aufgabe Wdh - SQL Basics auf artemis.tum.de. Artemis gibt dir immer, wenn du auf Submit drückst, die ersten Zeilen der Ergebnistabelle und ob deine SQL-Abfrage (bzw. welche Teile von ihr) richtig sind, Wenn du eine Abfrage richtig hast, notiere sie unten im Skript.

Falls du bei Gruppierung und Aggregatfunktionen Schwierigkeiten hast, hilft dir dieses Video (bitte Kopfhörer verwenden!): bycs.link/simpleclub-group-sort-aggregat

) Vervollständige die SQL-Abfrage so, dass sie ID, Name, Art und URL aller Freibäder ausgibt. SELECT id, name, art, urlFROM SchwimmbadWHERE art=

Freibad

## **SQL** Puzzle



In dieser Aufgabe geht es immer um die Tabelle **land**, deren erste Datensätze du hier siehst:

| id | name        | einwohner | flaeche | hauptstadt     |
|----|-------------|-----------|---------|----------------|
| 1  | Deutschland | 83.24     | 358     | Berlin         |
| 2  | Frankreich  | 67.39     | 544     | Paris          |
| 3  | Brasilien   | 212.60    | 8516    | Rio de Janeiro |
|    |             |           |         |                |

Welche SQL-Abfrage (rechte Seite) führt zu welcher Ergebnistabelle (linke Seite)? Ordne richtig zu



In dieser Aufgabe geht es immer um die Tabelle land, deren erste Datensätze du hier siehst:

| id | name        | einwohner | flaeche | hauptstadt     |
|----|-------------|-----------|---------|----------------|
| 1  | Deutschland | 83.24     | 358     | Berlin         |
| 2  | Frankreich  | 67.39     | 544     | Paris          |
| 3  | Brasilien   | 212.60    | 8516    | Rio de Janeiro |
|    |             |           |         |                |

Welche SQL-Abfrage (rechte Seite) führt zu welcher Ergebnistabelle (linke Seite)? Ordne richtig zu!

## Lösung:

- 1) iv)
- 2) viii)



In dieser Aufgabe geht es immer um die Tabelle land, deren erste Datensätze du hier siehst:

| id            | name        | einwohner | flaeche | hauptstadt     |
|---------------|-------------|-----------|---------|----------------|
| 1             | Deutschland | 83.24     | 358     | Berlin         |
| 2             | Frankreich  | 67.39     | 544     | Paris          |
| 3             | Brasilien   | 212.60    | 8516    | Rio de Janeiro |
| <del></del> . | ļ.,         |           |         |                |

Welche SQL-Abfrage (rechte Seite) führt zu welcher Ergebnistabelle (linke Seite)? Ordne richtig zu! Lösung:

#### 1) iv)

- 2) viii)
  - V T T T ,
- 3) vii)



In dieser Aufgabe geht es immer um die Tabelle land, deren erste Datensätze du hier siehst:

| id | name        | einwohner | flaeche | hauptstadt     |
|----|-------------|-----------|---------|----------------|
| 1  | Deutschland | 83.24     | 358     | Berlin         |
| 2  | Frankreich  | 67.39     | 544     | Paris          |
| 3  | Brasilien   | 212.60    | 8516    | Rio de Janeiro |
|    |             |           |         |                |

Welche SQL-Abfrage (rechte Seite) führt zu welcher Ergebnistabelle (linke Seite)? Ordne richtig zu!

## Lösung:

4) i)

- 1) iv)
  - 2) viii)
  - 3) vii)



In dieser Aufgabe geht es immer um die Tabelle **land**, deren erste Datensätze du hier siehst:

| id | name        | einwohner | flaeche | hauptstadt     |
|----|-------------|-----------|---------|----------------|
| 1  | Deutschland | 83.24     | 358     | Berlin         |
| 2  | Frankreich  | 67.39     | 544     | Paris          |
| 3  | Brasilien   | 212.60    | 8516    | Rio de Janeiro |
|    |             |           |         |                |

Welche SQL-Abfrage (rechte Seite) führt zu welcher Ergebnistabelle (linke Seite)? Ordne richtig zu! Lösung:

- 1) iv) 4) i) 2) viii) 5) ix)
- 2) viii) 5) ix 3) vii)



In dieser Aufgabe geht es immer um die Tabelle land, deren erste Datensätze du hier siehst:

| id | name        | einwohner | flaeche | hauptstadt     |
|----|-------------|-----------|---------|----------------|
| 1  | Deutschland | 83.24     | 358     | Berlin         |
| 2  | Frankreich  | 67.39     | 544     | Paris          |
| 3  | Brasilien   | 212.60    | 8516    | Rio de Janeiro |
|    |             |           |         |                |

Welche SQL-Abfrage (rechte Seite) führt zu welcher Ergebnistabelle (linke Seite)? Ordne richtig zu! Lösung:

| 1) iv | ) |  | 4 | ) i) |
|-------|---|--|---|------|

- 1) iv) 4) i) 2) viii) 5) ix
- 2) vii) 5) ix) 3) vii) 6) iii)



In dieser Aufgabe geht es immer um die Tabelle land, deren erste Datensätze du hier siehst:

| id | name        | einwohner | flaeche | hauptstadt     |
|----|-------------|-----------|---------|----------------|
| 1  | Deutschland | 83.24     | 358     | Berlin         |
| 2  | Frankreich  | 67.39     | 544     | Paris          |
| 3  | Brasilien   | 212.60    | 8516    | Rio de Janeiro |
|    |             |           |         |                |

Welche SQL-Abfrage (rechte Seite) führt zu welcher Ergebnistabelle (linke Seite)? Ordne richtig zu!

|  | Lösung: |
|--|---------|
|  |         |

| 1) 10)   | 7) 1)   |
|----------|---------|
| 2) viii) | 5) ix)  |
| 3) vii)  | 6) iii) |



| In dieser Aufgabe geht es immer um die Tabelle <b>land</b> , deren erste Datensätze du hier sieh | atensätze du hier siehst: |
|--|---------------------------|
|--|---------------------------|

| name        | einwohner                              | flaeche   | hauptstadt   |
|-------------|--|---|--|
| Deutschland | 83.24                                  | 358   | Berlin   |
| Frankreich  | 67.39                                  | 544   | Paris  |
| Brasilien   | 212.60                                 | 8516  | Rio de Janeiro   |
|             |  |   |  |
|             | Deutschland<br>Frankreich<br>Brasilien | Deutschland 83.24 Frankreich 67.39 Brasilien 212.60 | Deutschland       83.24       358         Frankreich       67.39       544         Brasilien       212.60       8516 |

Welche SQL-Abfrage (rechte Seite) führt zu welcher Ergebnistabelle (linke Seite)? Ordne richtig zu!

|  | Lösur |
|--|-------|
|  |       |

| 1) 10)   | 4) 1)   | /) V)  |
|----------|---------|--------|
| 2) viii) | 5) ix)  | 8) ii) |
| 3) vii)  | 6) iii) |        |



In dieser Aufgabe geht es immer um die Tabelle **land**, deren erste Datensätze du hier siehst:

| id            | name                     | einwohner               | flaeche                  | hauptstadt       |
|---------------|--------------------------|-------------------------|--------------------------|------------------|
| 1             | Deutschland              | 83.24                   | 358                      | Berlin           |
| 2             | Frankreich               | 67.39                   | 544                      | Paris            |
| 3             | Brasilien                | 212.60                  | 8516                     | Rio de Janeiro   |
|               |                          |                         |                          |                  |
| Walcha SOL-Ah | frage (rechte Seite) füh | ort zu welcher Fraehnis | tabelle (linke Seite)? C | rdne richtig zul |

ng:

|  |  | Lösur |
|--|--|-------|
|  |  |       |

| 1) iv)   | 4) i)   | 7) v)  |
|----------|---------|--------|
| 2) viii) | 5) ix)  | 8) ii) |
| 3) vii)  | 6) iii) | 9) vi) |
|          |         |        |



Bearbeite die Aufgabe Wdh - SQL Basics auf artemis.tum.de. Artemis gibt dir immer, wenn du auf Submit drückst, die ersten Zeilen der Ergebnistabelle und ob deine SQL-Abfrage (bzw. welche Teile von ihr) richtig sind, aus.

Wenn du eine Abfrage richtig hast, notiere sie unten im Skript.

Falls du bei Gruppierung und Aggregatfunktionen Schwierigkeiten hast, hilft dir dieses Video (bitte Kopfhörer

verwenden!): bycs.link/simpleclub-group-sort-aggregat

1) Vervollständige die SQL-Abfrage so, dass sie ID, Name, Art und URL aller Freibäder ausgibt.



Bearbeite die Aufgabe Wdh - SQL Basics auf artemis.tum.de. Artemis gibt dir immer, wenn du auf Submit drückst, die ersten Zeilen der Ergebnistabelle und ob deine SQL-Abfrage (bzw. welche Teile von ihr) richtig sind, aus.

Wenn du eine Abfrage richtig hast, notiere sie unten im Skript.

Falls du bei Gruppierung und Aggregatfunktionen Schwierigkeiten hast, hilft dir dieses Video (bitte Kopfhörer

verwenden!): bycs.link/simpleclub-group-sort-aggregat

1) Vervollständige die SQL-Abfrage so, dass sie ID, Name, Art und URL aller Freibäder ausgibt.

SELECT id, name, art, url

FROM Schwimmbad WHERE art= Freibad



2) Schreibe eine SQL-Abfrage, die ausgibt, wie viele Gemeinden es im Regierungsbezirk

Oberbayern gibt.

3) Schreibe eine SQL-Abfrage, die Name, Straße und URL (also die Internetadresse) alle Zoos in der Gemeinde mit Schluessel

Schluessel 09162000 ausgibt.



2) Schreibe eine SQL-Abfrage, die ausgibt, wie viele Gemeinden es im Regierungsbezirk

Oberbayern gibt. SELECT COUNT(\*) FROM Gemeinde WHERE regierungsbezirk= **Oberbayern** 

3) Schreibe eine SQL-Abfrage, die Name, Straße und URL (also die Internetadresse) alle Zoos in der Gemeinde mit

Schluessel 09162000 ausgibt.



2) Schreibe eine SQL-Abfrage, die ausgibt, wie viele Gemeinden es im Regierungsbezirk

gibt. SELECT COUNT(\*)

Oberbayern

FROM Gemeinde

WHERE regierungsbezirk=

**Oberbayern** 

3) Schreibe eine SQL-Abfrage, die Name, Straße und URL (also die Internetadresse) alle Zoos in der Gemeinde mit

Schluessel 09162000

SELECT name, strasse, url

WHERE gemeindeschluessel =

09162000

ausgibt.

FROM Zoo



4) Schreibe eine SQL-Abfrage, die die Summe aller weiblichen Einwohnerinnen und die Summe aller männlichen

Einwohner gruppiert nach Regierungsbezirk und den Namen des jeweiligen Regierungsbezirks ausgibt.

5) Schreibe eine SQL-Abfrage, die die durchschnittliche Fläche der Gemeinde eines Kreises (=Landkreis) und den Namen und Regierungsbezirk des jeweiligen Landkreises anzeigt. Sortiere die Ausgabe nach Name des Landkreises.



4) Schreibe eine SQL-Abfrage, die die Summe aller weiblichen Einwohnerinnen und die Summe aller männlichen

Einwohner gruppiert nach Regierungsbezirk und den Namen des jeweiligen Regierungsbezirks ausgibt.

#### SELECT regierungsbezirk, SUM(einwohner\_w), SUM(einwohner\_m)

FROM gemeinde GROUP BY regierungsbezirk

5) Schreibe eine SQL-Abfrage, die die durchschnittliche Fläche der Gemeinde eines Kreises (=Landkreis) und den Namen und Regierungsbezirk des jeweiligen Landkreises anzeigt. Sortiere die Ausgabe nach Name des Landkreises.



4) Schreibe eine SQL-Abfrage, die die Summe aller weiblichen Einwohnerinnen und die Summe aller männlichen

Einwohner gruppiert nach Regierungsbezirk und den Namen des jeweiligen Regierungsbezirks ausgibt.

SELECT regierungsbezirk, SUM(einwohner\_w), SUM(einwohner\_m)

GROUP BY regierungsbezirk

5) Schreibe eine SQL-Abfrage, die die durchschnittliche Fläche der Gemeinde eines Kreises (=Landkreis) und den Namen und Regierungsbezirk des jeweiligen Landkreises anzeigt. Sortiere die Ausgabe nach Name des Landkreises.

SELECT regierungsbezirk, kreis, avg(flaeche)

FROM Gemeinde
GROUP BY regierungsbezirk, kreis

ORDER BY kreis

FROM gemeinde



6) Schreibe eine SQL-Abfrage, die die Namen und Einwohnerzahlen aller Gemeinde, die mehr als 100.000 männliche und mehr als 100.000 weibliche Einwohner:innen haben, ausgibt.

7) Schreibe eine SQL-Abfrage, die die Namen und Einwohnerzahlen aller Gemeinde, die mehr als 75.000 männliche oder mehr als 75.000 weibliche Einwohner:innen haben, ausgibt.



männliche und mehr als 100.000 weibliche Einwohner:innen haben, ausgibt.

SELECT name, einwohner\_m, einwohner\_w

FROM Gemeinde WHERE einwohner\_m > 100000

AND einwohner\_w > 100000

7) Schreibe eine SQL-Abfrage, die die Namen und Einwohnerzahlen aller Gemeinde, die mehr als 75.000 männliche oder mehr als 75.000 weibliche Einwohner:innen haben, ausgibt.

FROM Gemeinde



männliche und mehr als 100.000 weibliche Einwohner:innen haben, ausgibt.

SELECT name, einwohner\_m, einwohner\_w

WHERE einwohner\_m > 100000

AND einwohner\_w > 100000

7) Schreibe eine SQL-Abfrage, die die Namen und Einwohnerzahlen aller Gemeinde, die mehr als 75.000 männliche oder mehr als 75.000 weibliche Einwohner:innen haben, ausgibt.

SELECT name, einwohner\_m, einwohner\_w

FROM Gemeinde WHERE einwohner\_m > 75000

OR einwohner\_w > 75000



8) Schreibe eine SQL-Abfrage, die Name, Landkreis, Fläche und die Einwohnerzahlen aller Gemeinden ausgibt, die jeweils mehr als 50.000 männliche und weibliche Einwohner:innen oder eine Fläche größer als 100 km² hat.

9) Schreibe eine SQL-Abfrage, die die durchschnittlichen männlichen und weiblichen Einwohnerzahlen aller Gemeinde mit mehr als 100 km² Fläche pro Landkreis und den Namen des jeweiligen Landkreises ausgibt.



8) Schreibe eine SQL-Abfrage, die Name, Landkreis, Fläche und die Einwohnerzahlen aller Gemeinden ausgibt, die jeweils mehr als 50.000 männliche und weibliche Einwohner:innen oder eine Fläche größer als 100 km² hat.

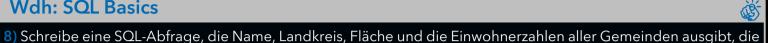
SELECT name, kreis, flaeche, einwohner\_m, einwohner\_w

FROM Gemeinde

WHERE (einwohner\_m > 50000 AND einwohner\_w > 50000) OR flaeche > 100

9) Schreibe eine SQL-Abfrage, die die durchschnittlichen männlichen und weiblichen Einwohnerzahlen aller

Gemeinde mit mehr als 100 km² Fläche pro Landkreis und den Namen des jeweiligen Landkreises ausgibt.



jeweils mehr als 50.000 männliche und weibliche Einwohner:innen oder eine Fläche größer als 100 km² hat.

SELECT name, kreis, flaeche, einwohner\_m, einwohner\_w

FROM Gemeinde WHERE (einwohner\_m > 50000 AND einwohner\_w > 50000)

OR flaeche > 100

9) Schreibe eine SQL-Abfrage, die die durchschnittlichen männlichen und weiblichen Einwohnerzahlen aller Gemeinde mit mehr als 100 km² Fläche pro Landkreis und den Namen des jeweiligen Landkreises ausgibt.

SELECT kreis, AVG(einwohner\_m), AVG(einwohner\_w)

FROM Gemeinde WHERE flaeche > 100 GROUP BY kreis



10) Schreibe eine SQL-Abfrage, die die Anzahl von Wanderwegen, die zu einer Gemeinde führen in einer Spalte

Anzahl und den jeweiligen Gemeindeschlüssel absteigend nach Anzahl sortiert, ausgibt.



10) Schreibe eine SQL-Abfrage, die die Anzahl von Wanderwegen, die zu einer Gemeinde führen in einer Spalte

Anzahl und den jeweiligen Gemeindeschlüssel absteigend nach Anzahl sortiert, ausgibt.

SELECT gemeindeschluessel, COUNT(\*) as Anzahl

FROM Wanderweg\_zu\_Gemeinde

GROUP BY gemeindeschluessel ORDER BY Anzahl DESC

#### **Outline**

- Stunde 1+2
  - Stunde 3+4
- Stunde 5+6

- Stunde 7+8
- stunde 9+10
- Stunde 11+12

In dieser Aufgabe geht es immer um die Tabelle land, deren erste Datensätze du hier siehst: einwohner flaeche hauptstadt name Deutschland 83.24 358 Berlin 544 Frankreich Paris Brasilien 212.60 Rio de Janeiro Lösuna:

#### Wdh: SQL Basics

Bearbeite die Aufgabe Wdh - SQL Basics auf artemis . tum. de. Artemis gibt dir immer, wenn du auf Submit drückst, die ersten Zeilen der Ergebnistabelle und ob deine SQL-Abfrage (bzw. welche Teile von ihr) richtig sind, aus.

Wenn du eine Abfrage richtig hast, notiere sie unten im Skript.

Fälls du bei Gruppierung und Aggregatfunktionen Schwierigkeiten hast, hilft dir dieses Video (bitte Kopfhörer verwenden!): bycs.link/simpleclub-group-sort-aggregat

Tabulankariahangan

1. Vasitatiera ibi Birdilly war skulpting a warden Birdill

1. Outige are dia diperce for data are failed international birdill

1. Outige are dia diperce for data are failed international birdilly are skulpting administration in Basinday administration.











- 1. Visualisiere (mit Bleistift), wer Häuptling in welchem Dorf ist.
- 2. Überlege, wie du allgemein für diese zwei Tabellen darstellen kannst, wie sie (und ihre Spalten) miteinander in Beziehung stehen.

| SELECT * FROM dorf |               |            |  |  |  |  |
|--------------------|---------------|------------|--|--|--|--|
| dorfnr             | name          | haeuptling |  |  |  |  |
| 1                  | Affenstadt    | 1          |  |  |  |  |
| 2                  | Gurkendorf    | 6          |  |  |  |  |
| 3                  | Zwiebelhausen | 7          |  |  |  |  |

| SELECT * FROM Bewohner |                          |        |            |               |      |           |  |  |
|------------------------|--------------------------|--------|------------|---------------|------|-----------|--|--|
| bewohnernr             | name                     | dorfnr | geschlecht | beruf         | gold | status    |  |  |
| 1                      | Paul<br>Backmann         | 1      | m          | Baecker       | 850  | friedlich |  |  |
| 2                      | Ernst Peng               | 3      | m          | Waffenschmied | 280  | friedlich |  |  |
| 3                      | Rita Ochse               | 1      | W          | Baecker       | 350  | friedlich |  |  |
| 4                      | Carl Ochse               | 1      | m          | Kaufmann      | 250  | friedlich |  |  |
| 5                      | Dirty Dieter             | 3      | m          | Schmied       | 650  | boese     |  |  |
| 6                      | Gerd<br>Schlachter       | 2      | m          | Metzger       | 4850 | boese     |  |  |
| 7                      | Peter<br>Schlachter      | 3      | m          | Metzger       | 3250 | boese     |  |  |
| 8                      | Arthur<br>Schneiderpaule | 2      | m          | Pilot         | 490  | gefangen  |  |  |



- 1. Visualisiere (mit Bleistift), wer Häuptling in welchem Dorf ist.
- 2. Überlege, wie du allgemein für diese zwei Tabellen darstellen kannst, wie sie (und ihre Spalten) miteinander in Beziehung stehen.

| SELECT * FROM dorf |               |            |  |  |  |  |
|--------------------|---------------|------------|--|--|--|--|
| dorfnr             | name          | haeuptling |  |  |  |  |
| 1                  | Affenstadt    | 1          |  |  |  |  |
| 2                  | Gurkendorf    | 6          |  |  |  |  |
| 3                  | Zwiebelhausen | 7          |  |  |  |  |

| SELECT * FROM Bewohner |                          |        |            |               |      |           |  |
|------------------------|--------------------------|--------|------------|---------------|------|-----------|--|
| bewohnernr             | name                     | dorfnr | geschlecht | beruf         | gold | status    |  |
| 1                      | Paul<br>Backmann         | 1      | m          | Baecker       | 850  | friedlich |  |
| 2                      | Ernst Peng               | 3      | m          | Waffenschmied | 280  | friedlich |  |
| 3                      | Rita Ochse               | 1      | w          | Baecker       | 350  | friedlich |  |
| 4                      | Carl Ochse               | 1      | m          | Kaufmann      | 250  | friedlich |  |
| 5                      | Dirty Dieter             | 3      | m          | Schmied       | 650  | boese     |  |
| 6                      | Gerd<br>Schlachter       | 2      | m          | Metzger       | 4850 | boese     |  |
| 7                      | Peter<br>Schlachter      | 3      | m          | Metzger       | 3250 | boese     |  |
| 8                      | Arthur<br>Schneiderpaule | 2      | m          | Pilot         | 490  | gefangen  |  |

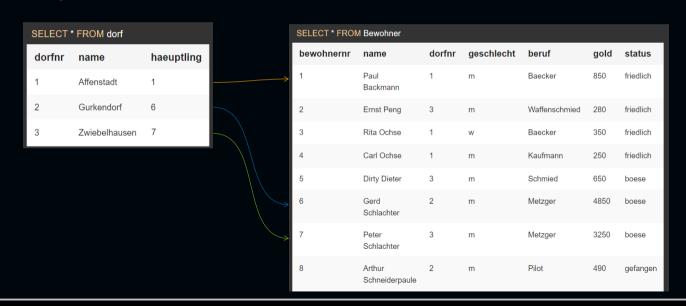


- 1. Visualisiere (mit Bleistift), wer Häuptling in welchem Dorf ist.
- 2. Überlege, wie du allgemein für diese zwei Tabellen darstellen kannst, wie sie (und ihre Spalten) miteinander in Beziehung stehen.

| SELECT* | FROM dorf     |            |          | SELECT * FROI | M Bewohner               |        |            |               |      |           |
|---------|---------------|------------|----------|---------------|--------------------------|--------|------------|---------------|------|-----------|
| dorfnr  | name          | haeuptling |          | bewohnernr    | name                     | dorfnr | geschlecht | beruf         | gold | status    |
| 1       | Affenstadt    | 1          |          | 1             | Paul<br>Backmann         | 1      | m          | Baecker       | 850  | friedlich |
| 2       | Gurkendorf    | 6          |          | 2             | Ernst Peng               | 3      | m          | Waffenschmied | 280  | friedlich |
| 3       | Zwiebelhausen | 7          |          | 3             | Rita Ochse               | 1      | w          | Baecker       | 350  | friedlich |
|         |               |            |          | 4             | Carl Ochse               | 1      | m          | Kaufmann      | 250  | friedlich |
|         |               |            |          | 5             | Dirty Dieter             | 3      | m          | Schmied       | 650  | boese     |
|         |               |            | <b>\</b> | 6             | Gerd<br>Schlachter       | 2      | m          | Metzger       | 4850 | boese     |
|         |               |            |          | 7             | Peter<br>Schlachter      | 3      | m          | Metzger       | 3250 | boese     |
|         |               |            |          | 8             | Arthur<br>Schneiderpaule | 2      | m          | Pilot         | 490  | gefangen  |

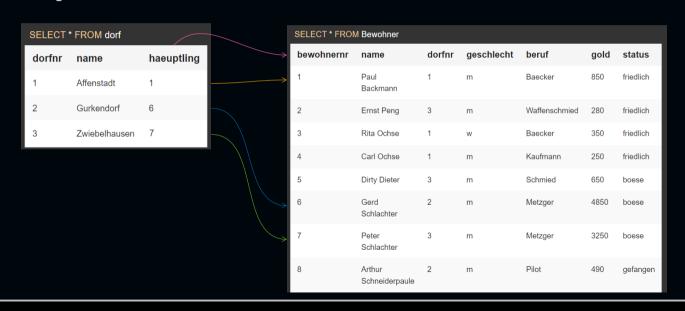


- 1. Visualisiere (mit Bleistift), wer Häuptling in welchem Dorf ist.
- 2. Überlege, wie du allgemein für diese zwei Tabellen darstellen kannst, wie sie (und ihre Spalten) miteinander in Beziehung stehen.



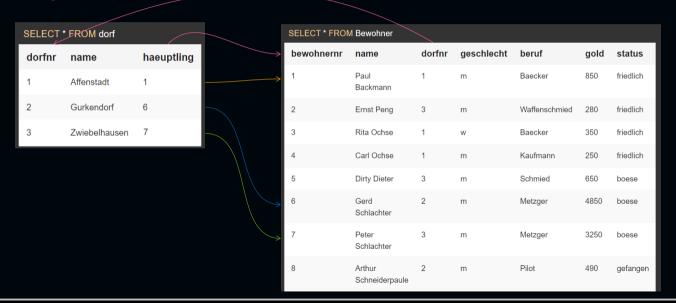


- 1. Visualisiere (mit Bleistift), wer Häuptling in welchem Dorf ist.
- 2. Überlege, wie du allgemein für diese zwei Tabellen darstellen kannst, wie sie (und ihre Spalten) miteinander in Beziehung stehen.





- 1. Visualisiere (mit Bleistift), wer Häuptling in welchem Dorf ist.
- 2. Überlege, wie du allgemein für diese zwei Tabellen darstellen kannst, wie sie (und ihre Spalten) miteinander in Beziehung stehen.



## Tabellenbeziehung im Klassendiagramm



- 1. Ergänze das Klassendiagramm entsprechend den beiden Tabellen oben.
- 2. Wie kann man die Beziehungen zwischen den beiden Tabellen im Klassendiagramm darstellen? <u>Tipp: Unsere Überleg</u>ungen von oben helfen dabei.

Dorf int dorfnr String name int bewohnernr
String name
String geschlecht
String beruf
int gold
String status

## Tabellenbeziehung im Klassendiagramm

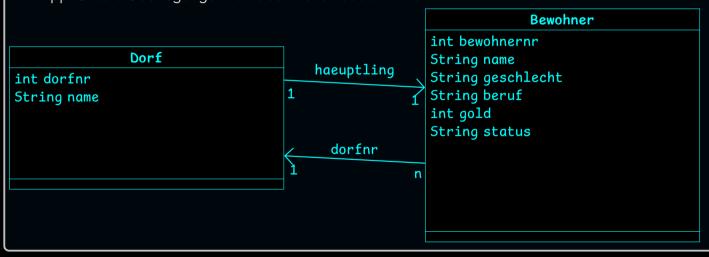
- 1. Ergänze das Klassendiagramm entsprechend den beiden Tabellen oben.
- 2. Wie kann man die Beziehungen zwischen den beiden Tabellen im Klassendiagramm darstellen? <u>Tipp: Unsere Überleg</u>ungen von oben helfen dabei.



## Tabellenbeziehung im Klassendiagramm



- 1. Ergänze das Klassendiagramm entsprechend den beiden Tabellen oben.
- 2. Wie kann man die Beziehungen zwischen den beiden Tabellen im Klassendiagramm darstellen? Tipp: Unsere Überlegungen von oben helfen dabei.



# Tabellenbeziehungen: Fremdschlüssel

durch

(manchmal

Wenn Datensätze mittels Primärschlüssel in einer anderen Tabelle verwendet werden, spricht man dort von einem

Fremdschlüssel. Im Tabellenschema werden die

auch

#### Taballanka-Sakumanan Fusan daskilisasal

Tabellenbeziehungen: Fremdschlüssel

Wenn Datensätze mittels Primärschlüssel in einer anderen Tabelle verwendet werden, spricht man dort von einem

Fremdschlüssel

durch

(manchmal

Fremdschlüssel. Im **Tabellenschema** werden die

auch

(manchmal

Tabellenbeziehungen: Fremdschlüssel

Fremdschlüssel

durch

überstreichen

Fremdschlüssel. Im **Tabellenschema** werden die

auch

Wenn Datensätze mittels Primärschlüssel in einer anderen Tabelle verwendet werden, spricht man dort von einem

# Tabellenbeziehungen: Fremdschlüssel

Wenn Datensätze mittels Primärschlüssel in einer anderen Tabelle verwendet werden, spricht man dort von einem durch überstreichen (manchmal

Fremdschlüssel. Im **Tabellenschema** werden die Fremdschlüssel

unterpunkten ) markiert. Ein Beispiel in SQL-Island ist der Häuptling eines Dorfes, der in der Tabelle auch Dorf mittels bewohnernr eingetragen wird. Die bewohnernr ist hierbei in der Tabelle **Bewohner** und in der Tabelle Dorf (heißt hier aber haeuptling).

# Tabellenbeziehungen: Fremdschlüssel

Wenn Datensätze mittels Primärschlüssel in einer anderen Tabelle verwendet werden, spricht man dort von einem durch überstreichen (manchmal

Fremdschlüssel. Im **Tabellenschema** werden die **Fremdschlüssel** 

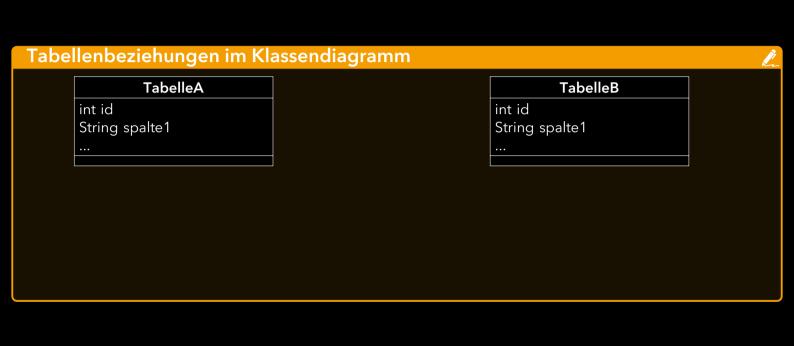
unterpunkten ) markiert. Ein Beispiel in SQL-Island ist der Häuptling eines Dorfes, der in der Tabelle auch Dorf mittels bewohnernr eingetragen wird. Die bewohnernr ist hierbei Primärschlüssel in der Tabelle **Bewohner** und in der Tabelle Dorf (heißt hier aber haeuptling).

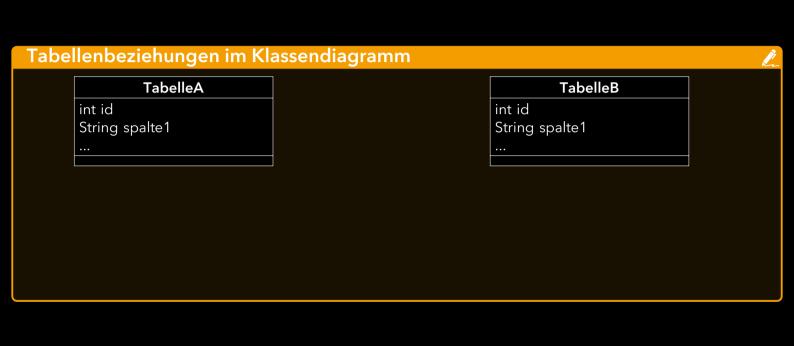
# Tabellenbeziehungen: Fremdschlüssel

Wenn Datensätze mittels Primärschlüssel in einer anderen Tabelle verwendet werden, spricht man dort von einem durch überstreichen (manchmal

Fremdschlüssel. Im **Tabellenschema** werden die Fremdschlüssel

unterpunkten ) markiert. Ein Beispiel in SQL-Island ist der Häuptling eines Dorfes, der in der Tabelle auch Dorf mittels bewohnernr eingetragen wird. Die bewohnernr ist hierbei Primärschlüssel in der Tabelle Bewohner und **Fremdschlüssel** in der Tabelle Dorf (heißt hier aber haeuptling).









- Beziehungspfeil immer vom Fremd- zum Primärschlüssel.
- 'fremdschluessel' ist eine Spalte der TabelleA, wird dort aber nicht eingetragen.
- Die Form der Pfeilspitze ist wichtig und muss genau so sein, da andere Spitzen andere Bedeutungen haben!
- Kardinalität an der Pfeilspitze ist immer 1 (bei Datenbanken), da in einer Spalte (eines Datensatzes) immer nur ein Wert stehen kann.

• m:n, z.B.

Die Kardinalität beschreibt, wie viele Objekte auf jeder Seite einer Beziehung stehen können. Es gibt folgende Arten:

Bewohner hat.

Schulklassen pro Lehrer (in

• 1:1, z.B. Häuptling pro Dorf, der auch nur in einem Dorf Häuptling ist.

Lehrer pro Schulklasse +

- 1:n, z.B. jeder Bewohner wohnt in einem Dorf, das aber
- 1:n, z.B. jeder Bewohner wohnt in einem Dorf, das aber

Datenbanken nicht direkt umsetzbar, dazu später mehr).

• m:n, z.B.

Die Kardinalität beschreibt, wie viele Objekte auf jeder Seite einer Beziehung stehen können. Es gibt folgende Arten:

Bewohner hat.

Schulklassen pro Lehrer (in

• 1:1, z.B. Häuptling pro Dorf, der auch nur in einem Dorf Häuptling ist. ein

Lehrer pro Schulklasse +

- 1:n, z.B. jeder Bewohner wohnt in einem Dorf, das aber

Datenbanken nicht direkt umsetzbar, dazu später mehr).

• m:n, z.B.

Die Kardinalität beschreibt, wie viele Objekte auf jeder Seite einer Beziehung stehen können. Es gibt folgende Arten:

Schulklassen pro Lehrer (in

- 1:1, z.B. Häuptling pro Dorf, der auch nur in einem Dorf Häuptling ist. ein
- Bewohner hat.
- 1:n, z.B. jeder Bewohner wohnt in einem Dorf, das aber mehrere

Lehrer pro Schulklasse +

Datenbanken nicht direkt umsetzbar, dazu später mehr).

• m:n, z.B.

Die Kardinalität beschreibt, wie viele Objekte auf jeder Seite einer Beziehung stehen können. Es gibt folgende Arten:

Schulklassen pro Lehrer (in

- 1:1, z.B. Häuptling pro Dorf, der auch nur in einem Dorf Häuptling ist. ein
- Bewohner hat.
- 1:n, z.B. jeder Bewohner wohnt in einem Dorf, das aber mehrere

Lehrer pro Schulklasse +

Datenbanken nicht direkt umsetzbar, dazu später mehr).

beliebig viele

• m:n, z.B.

Die Kardinalität beschreibt, wie viele Objekte auf jeder Seite einer Beziehung stehen können. Es gibt folgende Arten:

beliebig viele

Schulklassen pro Lehrer (in

- 1:1, z.B. Häuptling pro Dorf, der auch nur in einem Dorf Häuptling ist. ein
- Bewohner hat.

Lehrer pro Schulklasse +

1:n, z.B. jeder Bewohner wohnt in einem Dorf, das aber mehrere

Datenbanken nicht direkt umsetzbar, dazu später mehr).

beliebig viele



# Klassendiagramm Flugverspätung



Bearbeite diese Aufgabe auf artemis.tum.de.

Erstelle ein Klassendiagramm für die Datenbank unter dbiu.de/flugverspaetungen/.

Damit du weniger schreiben musst, kannst du die letzten 6 Spalten der Tabelle Flug durch ... ersetzen.

Achte auf korrektes Format, Datentypen und Kardinalitäten. Zeichne das Diagramm anschließend unten auf:



# Klassendiagramm Flugverspätung

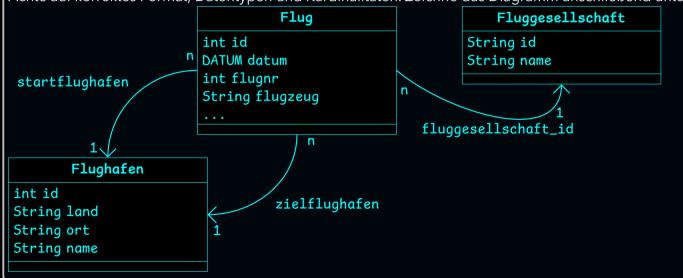


Bearbeite diese Aufgabe auf artemis.tum.de.

Erstelle ein Klassendiagramm für die Datenbank unter dbiu.de/flugverspaetungen/.

Damit du weniger schreiben musst, kannst du die letzten 6 Spalten der Tabelle Flug durch ... ersetzen.

Achte auf korrektes Format, Datentypen und Kardinalitäten. Zeichne das Diagramm anschließend unten auf:



#### Outline

Stunde 1+2

nde 3+4

Stunde 5+6

Stunde 7+8

Stunde 9+10

tunde 11+12

This indicates the property of the property of

Sent Section Section (Section Section Section



For any and the second control of the second













# Klassendiagramm Flugverspätung



Bearbeite diese Aufgabe auf artemis.tum.de.

Erstelle ein Klassendiagramm für die Datenbank unter dbiu.de/flugverspaetungen/.

Damit du weniger schreiben musst, kannst du die letzten 6 Spalten der Tabelle Flug durch ... ersetzen.

Achte auf korrektes Format, Datentypen und Kardinalitäten. Zeichne das Diagramm anschließend unten auf:



# Klassendiagramm Flugverspätung

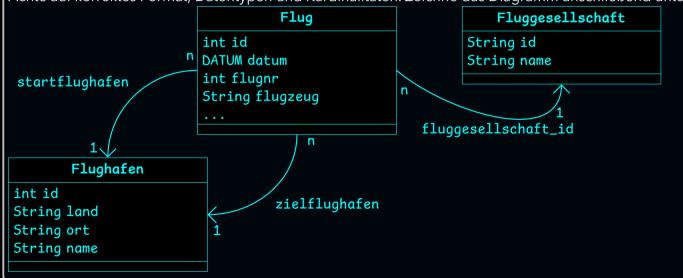


Bearbeite diese Aufgabe auf artemis.tum.de.

Erstelle ein Klassendiagramm für die Datenbank unter dbiu.de/flugverspaetungen/.

Damit du weniger schreiben musst, kannst du die letzten 6 Spalten der Tabelle Flug durch ... ersetzen.

Achte auf korrektes Format, Datentypen und Kardinalitäten. Zeichne das Diagramm anschließend unten auf:



#### **SQL:** Tabellen verbinden



aus diesen Tabellen auch zusammengehörende Datensätze abfragen. Öffne dafür www.dbiu.de/flugverspaetungen und führe folgende SQL-Abfrage aus:

**SELECT \*** 

FROM Fluggesellschaft, Flug



### **SQL:** Tabellen verbinden



werden die beiden Tabellen miteinander kombiniert?

Was beobachtest du? Werden nur zusammengehörende Datensätze angezeigt? Falls nicht, nach welchem Muster

#### **SQL:** Tabellen verbinden



Was beobachtest du? Werden nur zusammengehörende Datensätze angezeigt? Falls nicht, nach welchem Muster werden die beiden Tabellen miteinander kombiniert?

Nein, es werden alle Datensätze aus einer mit allen Datensätzen aus der anderen kombiniert und die Spalten einfach hintereinander aufgereiht.

omma

Möchte man Daten aus zwei Tabellen mit Beziehung zueinander abfragen, gibt man beide Tabellen mit Komma getrennt nach FROM an.

Die SQL-Abfrage bildet dann das der Tabellen. Die Ergebnistabelle enthält von Datensätzen beider Tabellen (Merkregel: ).

Um nur zusammengehörige Datensätze (also solche, die miteinenader in Beziehung stehen, z.B. eine Bewohner mit seinem Dorf) auszuwählen, ergänzt man als **Selektion** eine **Gleichheitsbedingung** zwischen Fremd- und zugehörigem . Dann spricht man von einem .

Zum Beispiel kann man in SQL-Island die Daten aller Dörfer und ihrer zugehörigen Häuptlinge so ausgeben:

SELECT \*
FROM Dorf, Bewohner

Möchte man Daten aus zwei Tabellen mit Beziehung zueinander abfragen, gibt man beide Tabellen mit Komma

getrennt nach FROM an. Die SQL-Abfrage bildet dann das Kreuzprodukt der Tabellen. Die Ergebnistabelle enthält

von Datensätzen beider Tabellen (Merkregel: Um nur zusammengehörige Datensätze (also solche, die miteinenader in Beziehung stehen, z.B. eine Bewohner mit seinem Dorf) auszuwählen, ergänzt man als Selektion eine Gleichheitsbedingung zwischen Fremd- und

zugehörigem . Dann spricht man von einem

Zum Beispiel kann man in SQL-Island die Daten aller Dörfer und ihrer zugehörigen Häuptlinge so ausgeben:

FROM Dorf, Bewohner

WHERE Dorf.haeuptling = Bewohner.bewohnernr

**SELECT\*** 

alle

Möchte man Daten aus zwei Tabellen mit Beziehung zueinander abfragen, gibt man beide Tabellen mit Komma getrennt nach FROM an.

Die SQL-Abfrage bildet dann das Kreuzprodukt der Tabellen. Die Ergebnistabelle enthält Kombinationen von Datensätzen beider Tabellen (Merkregel: Um nur zusammengehörige Datensätze (also solche, die miteinenader in Beziehung stehen, z.B. eine Bewohner

mit seinem Dorf) auszuwählen, ergänzt man als Selektion eine Gleichheitsbedingung zwischen Fremd- und zugehörigem . Dann spricht man von einem

Zum Beispiel kann man in SQL-Island die Daten aller Dörfer und ihrer zugehörigen Häuptlinge so ausgeben:

**SELECT\*** FROM Dorf, Bewohner

amma

Möchte man Daten aus zwei Tabellen mit Beziehung zueinander abfragen, gibt man beide Tabellen mit Kommagetrennt nach FROM an.

Die SQL-Abfrage bildet dann das Kreuzprodukt der Tabellen. Die Ergebnistabelle enthält alle Kombinationen von Datensätzen beider Tabellen (Merkregel: Jeder mit Jedem ).
Um nur zusammengehörige Datensätze (also solche, die miteinenader in Beziehung stehen, z.B. eine Bewohner mit seinem Dorf) auszuwählen, ergänzt man als Selektion eine Gleichheitsbedingung zwischen Fremd- und

zugehörigem . Dann spricht man von einem

Zum Beispiel kann man in SQL-Island die Daten aller Dörfer und ihrer zugehörigen Häuptlinge so ausgeben:

SELECT \*
FROM Dorf, Bewohner

alle

Möchte man Daten aus zwei Tabellen mit Beziehung zueinander abfragen, gibt man beide Tabellen mit Komma getrennt nach FROM an.

Die SQL-Abfrage bildet dann das Kreuzprodukt der Tabellen. Die Ergebnistabelle enthält von Datensätzen beider Tabellen (Merkregel: Jeder mit Jedem

Kombinationen Um nur zusammengehörige Datensätze (also solche, die miteinenader in Beziehung stehen, z.B. eine Bewohner mit seinem Dorf) auszuwählen, ergänzt man als Selektion eine Gleichheitsbedingung zwischen Fremd- und zugehörigem Primärschlüssel . Dann spricht man von einem

Zum Beispiel kann man in SQL-Island die Daten aller Dörfer und ihrer zugehörigen Häuptlinge so ausgeben:

SELECT \* FROM Dorf, Bewohner

alle

Möchte man Daten aus zwei Tabellen mit Beziehung zueinander abfragen, gibt man beide Tabellen mit Komma getrennt nach FROM an.

Die SQL-Abfrage bildet dann das Kreuzprodukt der Tabellen. Die Ergebnistabelle enthält Kombinationen von Datensätzen beider Tabellen (Merkregel: Jeder mit Jedem Um nur zusammengehörige Datensätze (also solche, die miteinenader in Beziehung stehen, z.B. eine Bewohner mit seinem Dorf) auszuwählen, ergänzt man als Selektion eine Gleichheitsbedingung zwischen Fremd- und

zugehörigem Primärschlüssel . Dann spricht man von einem Join

**SELECT\*** 

Zum Beispiel kann man in SQL-Island die Daten aller Dörfer und ihrer zugehörigen Häuptlinge so ausgeben:

FROM Dorf, Bewohner

# Join Beispiel Lehrkraft kuerzel Her Ext

schule MTG

Dante

**SELECT**\* FROM Lehrkraft, Schule

WHERE Lehrkraft.schule = Schule.id

ort Haidh Haidh. Sendl.

ort

Schule

id

MTG

Dante

id kuerzel schule id

| •                         | 1101 |       |       | i idiaii. |  |  |  |  |
|---------------------------|------|-------|-------|-----------|--|--|--|--|
| 2                         | Ext  | Dante | MTG   | Haidh.    |  |  |  |  |
| 1                         | Her  | MTG   | Dante | Sendl.    |  |  |  |  |
| 2                         | Ext  | Dante | Dante | Sendl.    |  |  |  |  |
| Ergebnistabelle des Joins |      |       |       |           |  |  |  |  |

| Ergebnistabelle des Joins |         |        |       |        |  |  |  |
|---------------------------|---------|--------|-------|--------|--|--|--|
| id                        | kuerzel | schule | id    | ort    |  |  |  |
| 1                         | Her     | MTG    | MTG   | Haidh. |  |  |  |
| 2                         | Ext     | Dante  | Dante | Sendl. |  |  |  |
|                           |         |        |       |        |  |  |  |

#### **Outline**

Stunde 1+2

Stunde 3+4

Stunde 5+6

Stunde 7+8

Stunde 9+10

Stunde 11+12





Knozprodaki Julia

Monoprodaki J





#### SQL mit Kreuzprodukt und Join

Bearbeite diese Aufgabe auf ar temis, tum, de. Du bekommsteine automatische Rückmeldung, ob deine Abgabe korrekt ist Alle Aufgaben beziehen sich auf die Datenbank mit untem stehendem Klassendiagramm. Eine Online-Version gibt es unter www. dib in. de /bayern/, dors ist auch das Tabellenschema zu finden.

Gib immer genau die geforderten Daten aus und nicht mehr. Sortiere nicht, wenn du nicht dazu aufgefordert wirst. Notiere unten anschließend deine korrekten SQL-Abfragen unten.

# Join Beispiel Lehrkraft id | kuerzel | s 1 | Her | 1 2 | Ext | [

schule

MTG

Dante

# SELECT \* FROM Lehrkraft, Schule

WHERE Lehrkraft.schule = Schule.id

ort

id ort
MTG Haidh.
Dante Sendl.

Schule

# id kuerzel schule id

MTG MTG Haidh. Her Ext MTG Haidh. **Dante** Her MTG Sendl. **Dante** Ext Sendl. **Dante** Dante

| Ergebnistabelle des Joins |         |        |       |        |  |  |  |
|---------------------------|---------|--------|-------|--------|--|--|--|
| id                        | kuerzel | schule | id    | ort    |  |  |  |
| 1                         | Her     | MTG    | MTG   | Haidh. |  |  |  |
| 2                         | Ext     | Dante  | Dante | Sendl. |  |  |  |
|                           |         |        |       |        |  |  |  |

# SQL mit Kreuzprodukt und Join



Bearbeite diese Aufgabe auf artemis.tum.de. Du bekommst eine automatische Rückmeldung, ob deine Abgabe korrekt ist. Alle Aufgaben beziehen sich auf die Datenbank mit untem stehendem Klassendiagramm. Eine Online-Version gibt es unter www.dbiu.de/bayern/, dort ist auch das Tabellenschema zu finden. Gib immer genau die geforderten Daten aus und nicht mehr. Sortiere nicht, wenn du nicht dazu aufgefordert wirst.

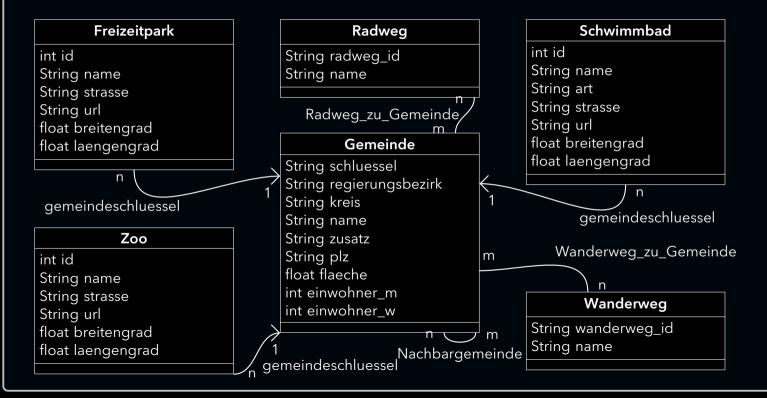
Gib immer genau die geforderten Daten aus und nicht mehr. Sortiere nicht, wenn du nicht dazu aufgefordert wirst.

Notiere unten anschließend deine korrekten SQL-Abfragen unten.



# **SQL** mit Kreuzprodukt und Join









Verändere die SQL-Abfrage so, dass die Namen und Internetadressen (=url) aller Zoos und der Name und Regierungsbezirk der jeweiligen Gemeinde ausgegeben wird:

SELECT Zoo.name, Gemeinde.name

FROM Zoo, Gemeinde





Verändere die SQL-Abfrage so, dass die Namen und Internetadressen (=url) aller Zoos und der Name und Regierungsbezirk der jeweiligen Gemeinde ausgegeben wird:

SELECT Zoo.name, Gemeinde.name , Gemeinde.regierungsbezirk, Zoo.url

FROM Zoo, Gemeinde





Verändere die SQL-Abfrage so, dass die Namen und Internetadressen (=url) aller Zoos und der Name und Regierungsbezirk der jeweiligen Gemeinde ausgegeben wird:

SELECT Zoo.name, Gemeinde.name , Gemeinde.regierungsbezirk, Zoo.url

FROM Zoo, Gemeinde

WHERE Zoo.gemeindeschluessel = Gemeinde.schluessel





Verändere die SQL-Abfrage so, dass die Namen und Straßen aller Freizeitparks und die Namen der jeweils zugehörigen Gemeinde ausgegeben wird.

SELECT Freizeitpark.name, Gemeinde.name

FROM Freizeitpark, Gemeinde





Verändere die SQL-Abfrage so, dass die Namen und Straßen aller Freizeitparks und die Namen der jeweils zugehörigen Gemeinde ausgegeben wird.

SELECT Freizeitpark.name, Gemeinde.name , Freizeitpark.strasse

FROM Freizeitpark, Gemeinde



Verändere die SQL-Abfrage so, dass die Namen und Straßen aller Freizeitparks und die Namen der jeweils zugehörigen Gemeinde ausgegeben wird.

SELECT Freizeitpark.name, Gemeinde.name , Freizeitpark.strasse

FROM Freizeitpark, Gemeinde

WHERE Gemeinde.schluessel = Freizeitpark.gemeindeschluessel





Schreibe eine SQL-Abfrage, die Namen und Art aller Schwimmbäder und den Namen und alle Einwohnerzahlen der zugehörigen Gemeinden ausgibt.





Schreibe eine SQL-Abfrage, die Namen und Art aller Schwimmbäder und den Namen und alle Einwohnerzahlen der zugehörigen Gemeinden ausgibt.

SELECT Schwimmbad.name, Schwimmbad.art,

Gemeinde.name, Gemeinde.einwohner\_m, Gemein<u>de.einwohner\_w</u>

FROM Schwimmbad, Gemeinde

WHERE Gemeinde.schluessel = Schwimmbad.gemeindeschluessel





Schreibe eine SQL-Abfrage, die die Anzahl an Schwimmbädern in Gemeinden mit **mehr** als 1000 weiblichen Einwohnerinnen ausgibt.

Tipp: Hier brauchst du mehrere verknüpfte Bedingungen





Schreibe eine SQL-Abfrage, die die Anzahl an Schwimmbädern in Gemeinden mit **mehr** als 1000 weiblichen Einwohnerinnen ausgibt.

Tipp: Hier brauchst du mehrere verknüpfte Bedingungen

SELECT COUNT(\*)

EDOM Sobwimmbod Comoins

FROM Schwimmbad, Gemeinde WHERE Gemeinde.schluessel = Schwimmbad.gemeindeschluessel

AND Gemeinde.einwohner\_w > 1000





Schreibe eine SQL-Abfrage, die die Namen aller Gemeinde in Oberbayern oder Niederbayern, zu denen ein Wanderweg führt, ausgibt. Dopplungen dürfen auftreten und sollte nicht entfernt werden!

Tipp: Hier brauchst du wieder mehrere verknüpfte Bedingungen. Überlege bei der Verknüpfung von Bedingungen.

Tipp: Hier brauchst du wieder mehrere verknüpfte Bedingungen. Überlege bei der Verknüpfung von Bedingungen, ob du Klammern setzen musst!





Schreibe eine SQL-Abfrage, die die Namen aller Gemeinde in Oberbayern oder Niederbayern, zu denen ein Wanderweg führt, ausgibt. Dopplungen dürfen auftreten und sollte nicht entfernt werden!

Tipp: Hier brauchst du wieder mehrere verknüpfte Bedingungen. Überlege bei der Verknüpfung von Bedingungen, ob du Klammern setzen musst!

SELECT Gemeinde.name

FROM Gemeinde, Wanderweg\_zu\_Gemeinde

WHERE Gemeinde.schluessel = Wanderweg\_zu\_Gemeinde.gemeindeschluessel

AND (Gemeinde.regierungsbezirk='Oberbayern'

OR Gemeinde.regierungsbezirk='Niederbayern')





Schreibe eine SQL-Abfrage, die aus den Tabellen Gemeinde und Wanderweg\_zu\_Gemeinde die Anzahl der Wanderwege, die zu Gemeinden mit mehr als 500 000 männlichen Einwohnern führen, ausgibt.





Schreibe eine SQL-Abfrage, die aus den Tabellen Gemeinde und Wanderweg\_zu\_Gemeinde die Anzahl der Wanderwege, die zu Gemeinden mit mehr als 500 000 männlichen Einwohnern führen, ausgibt.

SELECT COUNT(\*)

FROM Gemeinde, Wanderweg\_zu\_Gemeinde

WHERE Gemeinde.schluessel = Wanderweg\_zu\_Gemeinde.gemeindeschluessel

AND einwohner\_m > 500000





Schreibe eine SQL-Abfrage, die eine Liste mit den Namen aller Gemeinden, die ein 'Freibad' haben, und die Namen der jeweiligen Freibäder ausgibt.





Schreibe eine SQL-Abfrage, die eine Liste mit den Namen aller Gemeinden, die ein 'Freibad' haben, und die Namen der jeweiligen Freibäder ausgibt.

SELECT Gemeinde.name, Schwimmbad.name

FROM Gemeinde, Schwimmbad

WHERE Gemeinde.schluessel=Schwimmbad.gemeindeschluessel

AND Schwimmbad.art='Freibad'





Schreibe eine SQL-Abfrage, die die Anzahl an Radwegen, die an Gemeinden im PLZ-Bereich **größer** als 96400 angrenzen, ausgibt.





Schreibe eine SQL-Abfrage, die die Anzahl an Radwegen, die an Gemeinden im PLZ-Bereich **größer** als 96400 angrenzen, ausgibt.

SELECT COUNT(\*)

FROM Gemeinde, Radweg\_zu\_Gemeinde

WHERE Gemeinde.schluessel=Radweg\_zu\_Gemeinde.gemeindeschluessel

AND Gemeinde.plz > 96400





Schreibe eine SQL-Abfrage, die die Namen aller Zoos in einer Gemeinde namens 'Erlangen' ausgibt.





Schreibe eine SQL-Abfrage, die die Namen aller Zoos in einer Gemeinde namens 'Erlangen' ausgibt. SELECT Zoo.name

FROM Zoo, Gemeinde

WHERE Zoo.gemeindeschluessel = Gemeinde.schluessel

AND Gemeinde.name='Erlangen'





Schreibe eine SQL-Abfrage, die die IDs aller Radwege, die zu Gemeinden in Oberfranken oder Unterfranken führen, ausgibt. Dopplungen sollen nicht entfernt werden.





Schreibe eine SQL-Abfrage, die die IDs aller Radwege, die zu Gemeinden in Oberfranken oder Unterfranken führen, ausgibt. Dopplungen sollen nicht entfernt werden.

SELECT Radweg\_zu\_Gemeinde.radweg\_id

FROM Radweg\_zu\_Gemeinde, Gemeinde

WHERE Gemeinde.schluessel = Radweg\_zu\_Gemeinde.gemeindeschluessel

AND (Gemeinde.regierungsbezirk = 'Oberfranken'

OR Gemeinde.regierungsbezirk='Unterfranken')