

KLEINES PRIVATES LEHRINSTITUT

## DERKSEN

GYMNASIUM

Informatik 10 - Datenbanken (Teil 1)

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

Stunde 1+2 Wdh: Klassen und Objekte Objektkarten Memory

Stunde 3+4

Wdh: Von der Klasse zur Tabelle
Wdh: Aufbau von (relationalen) Datenbanken

SQL Spickzettel Übung: SQL Island

SQL Puzzle

Wdh: SQL Basics

Stunde 7+8

#### Tabellenbeziehunger

Tabellenbeziehungen: Fremdschlüssel Tabellenbeziehungen im Klassendiagramm Kardinalitäten



Klassendiagramm Flugverspätung

Stunde 9+10



Klassendiagramm Flugverspätung SQL: Tabellen verbinden Kreuzprodukt / Join Join Beispiel

Stunde 11+12
Join Beispiel



SQL mit Kreuzprodukt und Join

#### **Outline**

Stunde 1+2

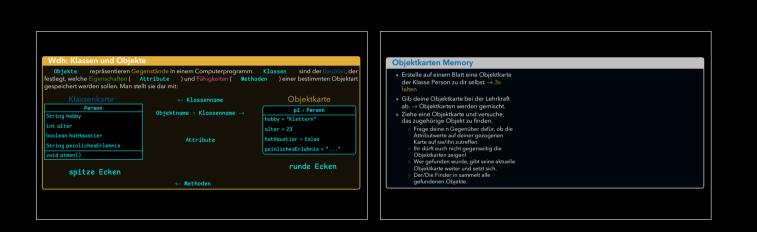
tunde 3+4

Stunde 5+6

Stunde 7+8

stunde 9+10

Stunde 11+12



#### Wdh: Klassen und Objekte **Objekte**

repräsentieren **Gegenstände** in einem Computerprogramm. ) und **Fähigkeiten** (

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

Klassenkarte

Klassen sind der Bauplan, der ) einer bestimmten Objektart

**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

Klassenkarte

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

gespeichert werden sollen. Man stellt sie dar mit:

**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()

olan, der

**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

THE GIVEN

boolean hatHaustier

String peinlichesErlebnis void atmen()

spitze Ecken

 $\leftarrow$  Klassenname

Objektname : Klassenname ightarrow

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

 $\leftarrow$  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 = "..."

runde Ecken

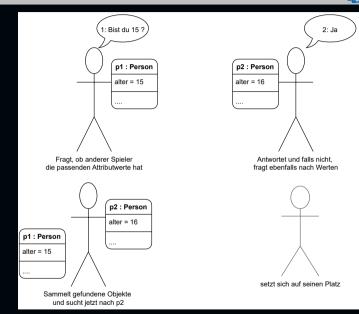
spitze Ecken

← Methoden

#### **Objektkarten Memory**

<u>\$</u>

- 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

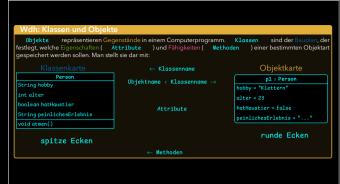
Stunde 3+4

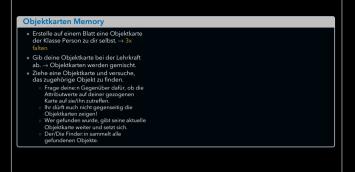
Stunde 5+6

Stunde 7+8

Stunde 9+10

tunde 11+12









SSL Spinkrantial

Figuries SSL Spinkrantian think die SSL Sprindigen der Filmen in derhijwitz ihr ber der SSL Auf

Inneren Gleif die Vorlag symptod 

G stant born in den Spinkratik in signen FOF offens.



#### Wdh: Von der Klasse zur Tabelle

Datentyp Attributwert



- 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 Zei-

#### 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	
Zell	e/ Attributw	ert			

Tabelle

Klasse/ Spaltennamen

Datensatz/Zeile/ Obiekt

Nicht verwendete Begriffe: Parameter, Methode, Board, Datentyp

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

Aufbau ist:

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 dargestellt werden. Dessen oder

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

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

auch "ID"), der Datensätze eindeutig identifiziert. Oft werden die Datensätze hiermit einfach durchnummeriert. Im <u>Tabellenschema wird er unte</u>rstrichen 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:

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 oder dargestellt werden. Dessen Aufbau ist:

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

Zum Beispiel:

<u>Datenbanken speichern Dat</u>ensätze in <u>Tabellen</u> . Die <u>Spaltenüberschriften</u> 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

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

auch "ID"), der Datensätze eindeutig identifiziert. Oft werden die Datensätze hiermit einfach durchnummeriert. Im

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

<u>Datenbanken speichern Dat</u>ensätze in <u>Tabellen</u> . Die <u>Spaltenüberschriften</u> 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 Tabellenschema dargestellt werden. Dessen Aufbau ist: TABELLENNAME(Datentyp Primärschlüssel, Datentyp Spalte1, Datentyp Spalte2, ...)

Zum Beispiel:

<u>Datenbanken speichern Dat</u>ensätze in <u>Tabellen</u> . Die <u>Spaltenüberschriften</u> 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.

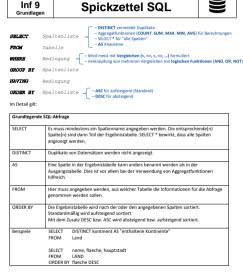
Der Aufbau einer Tabelle kann mit Klassenkarte oder Tabellenschema dargestellt werden. Dessen Aufbau ist: TABELLENNAME(Datentyp Primärschlüssel, Datentyp Spalte1, Datentyp Spalte2, ...)

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

#### **SQL Spickzettel**



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.



Auswahl von Datensätzen über Bedingungen			
WHERE	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 <, <=, =, <>, us. 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 Ausdrücke dabei sinnvoll geklammert werden			
	Beispiel WHERE iahr > 2015			
	AND laufzelt <= 90 AND NOT fsk = 18			
LIKE	Kann in einer Bedingung zur Mustererkennung von Einträgen verwendet werden. Folgende zwei Platzhalter (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%" – findet alle Titel die mit "You" beginnen Groß-/Kleinschreibung wird nicht berücksichtigt WHERE Titel I IKE "Akone%" – finder alle Titel die "Inve" enthalten			
	<ul> <li>WHERE titel LIKE "Move%" – findet alle Titel die "love" enthalten</li> <li>WHERE titel LIKE "L" – findet alle Titel die mit L beginnen und genau 4 Zeichen lang sind</li> </ul>			
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	onen		
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 "Anzahi afrikanischer Länder" FROM Land WHERE kontlinent = "Afrika"		

GROUP BY	Datensätze mit demselben Wert in der angegeben Spalte werden gruppiert.	
	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 genre1="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)

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

GEGENSTAND(String gegenstand, 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(<u>int bewohnernr</u>, 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

otunde 7±10

Stunde 11+12

### Width Vion day Klasse and Tabella 2-Zerton as another on Jahon, in der and de Opinio der Klasse Pressus serrenin bare. Trige neue based opinio error Opinio der Klasse Pressus serrenin bare. Trige neue based opinio error opi

Welfe, Affrage vom fordstonden Demendanken
Demendang verben bereiten Stelle Stelle Springerier in Stelle Stelle Springerier in Stelle Stelle Springerier in Stelle Stelle Springerier in Stelle Stelle

SOL Spickzensi

Trigorder S, Spickzensi embla die SSL Corrolagen der S Cases Br delt bjeltig im tes after SSL Aufgeben
ennann. (Der die Vorlageregebel )

eben Storet br den Spickzense jak aggeser FOF effens.



#### **SQL** Puzzle

 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

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

 Lösung:

 1) iv)
 4) i)
 7) v)

 2) viii)
 5) ix)
 8) ii)

 3) vii)
 6) iii)
 9) vi)

#### Wdh: SQL Basics

Bearbeite die Aufgabe Wdh - SOL 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, urlFROM SchwimmbadNHERE 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
	ļ., <u>,</u>		· · · · · · · · · · · · · · · · · · ·	

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

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

#### 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
				ļ <u></u>

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

#### Lösung:

- 1) iv) 4) i) 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
Wolcho COL /	\hfraga /rachta Caita\fiik	ut - u vuolokov Evaokais	taballa (linka Saita)2 C	Nedmo violetica - Lul

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

- 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
Walcha SOL	Abfraga (rachta Saita) fül	ort zu wolcher Ergebnie	taballa (linka Saita)2 C	Ordno richtia zul

ng:

		Lösur

1) 10)	4) 1)
2) viii)	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
Wolcho SOL AL	ofrago (rachta Saita) fiil	ort zu wolcher Ergebnic	taballa (linka Saita)2 C	ordno richtia zul

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

		Lösung:

T) TV)	4) <del>1</del> )	/ ) 🔻
2) viii)	5) ix)	8) ii)
3) vii)	6) iii)	



In dieser Aufga	abe geht es imm	er um die Tabelle	e land, c	deren erste Datensätze d	u 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
Wolcho SOL AL	ofrago (rachta Saita) fiil	ert zu wolcher Ergebnic	taballa (linka Saita)2 C	ordno richtia zul

ng:

		Lösun

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(\*) FROM Gemeinde

Oberbayern

09162000

WHERE regierungsbezirk= **Oberbayern** 

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

ausgibt. SELECT name, strasse, url

FROM Zoo

WHERE gemeindeschluessel = 09162000



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)

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.

SELECT regierungsbezirk, kreis, avg(flaeche)

FROM Gemeinde
GROUP BY regierungsbezirk,kreis

ORDER BY kreis



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.



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.

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.



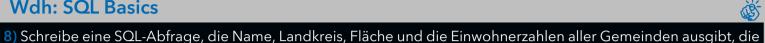
8) Schreibe eine SQL-Abfrage, die Name, Landkreis, Fläche und die Einwohnerzahlen aller Gemeinden ausgibt, die jeweils mehr als 50.000 männliche und weibliche Einwohner:innen oder eine Fläche größer als 100 km² hat.

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

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

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

#### SOI Puzzle

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.

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 1d, name, art, ur1FROM SchwimmbadMHERE art= Freibad











- 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 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 Schlachter	2	m	Metzger	4850	boese		
7	Peter Schlachter	3	m	Metzger	3250	boese		
8	Arthur 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 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 Schlachter	2	m	Metzger	4850	boese		
7	Peter Schlachter	3	m	Metzger	3250	boese		
8	Arthur Schneiderpaule	2	m	Pilot	490	gefangen		

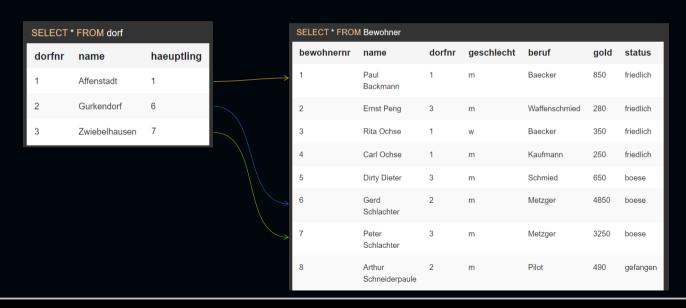


- 1. Visualisiere (mit Bleistift), wer Häuptling in welchem Dorf ist.
- 2. Überlege, wie du allgemein für diese zwei Tabellen darstellen kannst, wie sie (und ihre Spalten) miteinander in Beziehung stehen.

SELECT '	FROM dorf			SELECT * FROI	M Bewohner					
dorfnr	name	haeuptling		bewohnernr	name	dorfnr	geschlecht	beruf	gold	status
1	Affenstadt	1		1	Paul 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 Schlachter	2	m	Metzger	4850	boese
				7	Peter Schlachter	3	m	Metzger	3250	boese
				8	Arthur Schneiderpaule	2	m	Pilot	490	gefangen

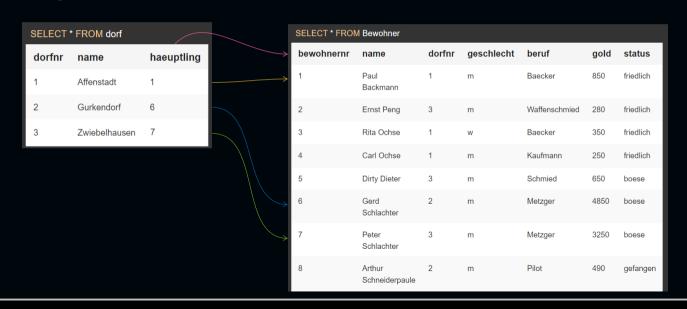


- 1. Visualisiere (mit Bleistift), wer Häuptling in welchem Dorf ist.
- 2. Überlege, wie du allgemein für diese zwei Tabellen darstellen kannst, wie sie (und ihre Spalten) miteinander in Beziehung stehen.



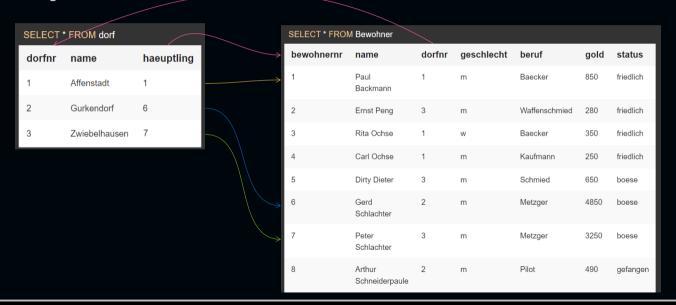


- 1. Visualisiere (mit Bleistift), wer Häuptling in welchem Dorf ist.
- 2. Überlege, wie du allgemein für diese zwei Tabellen darstellen kannst, wie sie (und ihre Spalten) miteinander in Beziehung stehen.





- 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

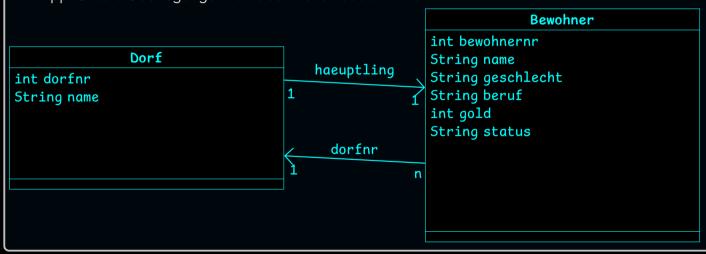
- 1. Ergänze das Klassendiagramm entsprechend den beiden Tabellen oben.
- 2. Wie kann man die Beziehungen zwischen den beiden Tabellen im Klassendiagramm darstellen? <u>Tipp: Unsere Überleg</u>ungen von oben helfen dabei.



#### 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

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

Fremdschlüssel. Im Tabellenschema werden die durch (manchmal

auch

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

Fremdschlüssel. Im **Tabellenschema** werden die Fremdschlüssel durch (manchmal

auch

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

Fremdschlüssel. Im **Tabellenschema** werden die Fremdschlüssel durch überstreichen (manchmal

auch

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).

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

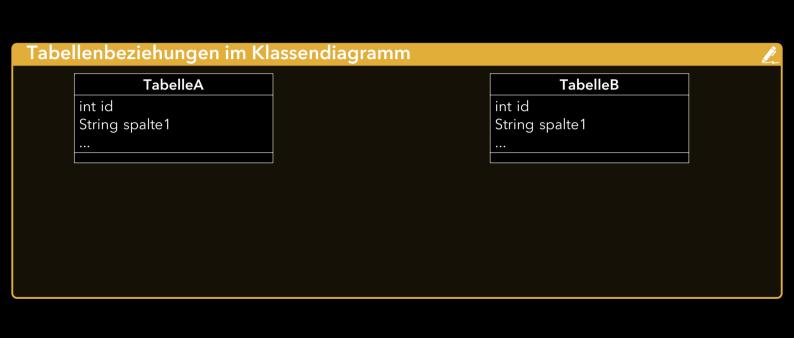
Fremdschlüssel. Im **Tabellenschema** werden die **Fremdschlüssel** 

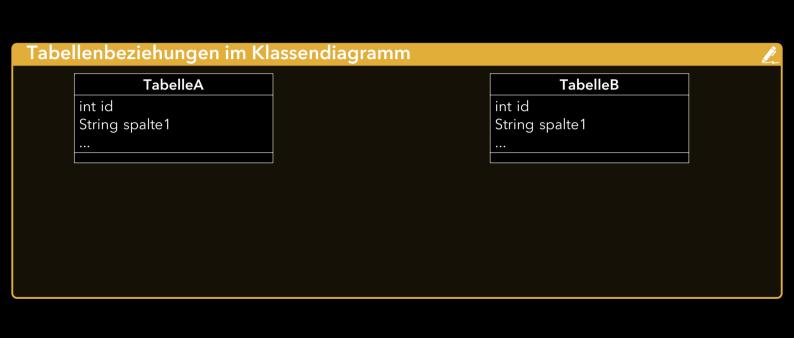
unterpunkten ) markiert. Ein Beispiel in SQL-Island ist der Häuptling eines Dorfes, der in der Tabelle auch Dorf mittels bewohnernr eingetragen wird. Die bewohnernr ist hierbei Primärschlüssel in der Tabelle **Bewohner** und in der Tabelle Dorf (heißt hier aber haeuptling).

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 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).

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

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

• m:n, z.B. Lehrer pro Schulklasse + Schulklassen pro Lehrer (in

Bewohner hat.

Datenbanken nicht direkt umsetzbar, dazu später mehr).

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

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

Datenbanken nicht direkt umsetzbar, dazu später mehr).

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

• m:n, z.B. Lehrer pro Schulklasse + Schulklassen pro Lehrer (in

• 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

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

- 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

beliebig viele Lehrer pro Schulklasse + beliebig viele Schulklassen pro Lehrer (in • m:n, z.B. Datenbanken nicht direkt umsetzbar, dazu später mehr).



## 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

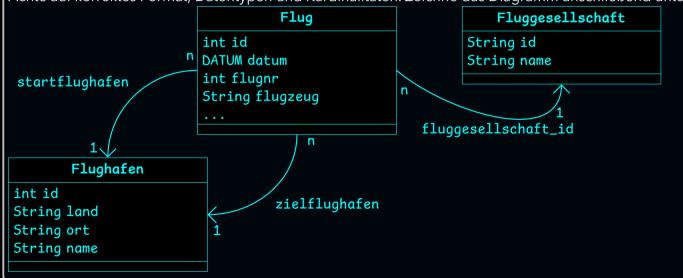


Bearbeite diese Aufgabe auf artemis.tum.de.

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

| Statutechastologye
| V. Warders to should be undustry a midster four of the first first

The black and a part of the state of the sta

And Andrew Control of Control of

Control of Control of













## 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

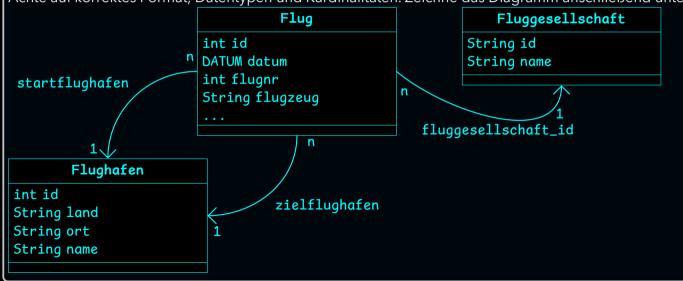


Bearbeite diese Aufgabe auf artemis.tum.de.

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



Wir kennen jetzt Tabellen, die miteinander über Fremd- und Primärschlüssel in Beziehung stehen. Nun möchten wir 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.

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

**SELECT\*** 

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

## 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 alle 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

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 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

**SELECT\*** 

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

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

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\*** 

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 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 Join

zugehörigem Primärschlüssel . Dann spricht man von einem

**SELECT\*** 

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

#### Join Beispiel Lehrkraft Schule **SELECT**\* kuerzel schule id ort FROM Lehrkraft, Schule Her MTG MTG Haidh. WHERE Lehrkraft.schule = Schule.id Ext Dante Dante Sendl. id kuerz<u>el</u> schule id ort MTG MTG Haidh. Her Ext MTG Haidh. **Dante** Her MTG Dante Sendl.

2	2   Ext	Dante	Dante	Sendl.	
Ergebnistabelle des Joins					
id	d   kuerze	el schule	id	ort	
1	1 Her	MTG	MTG	Haidh	
2	2 Ext	Dante	Dante	Sendl	

### **Outline**

Stunde 1+2

Stunde 3+4

Stunde 5+6

Stunde 7+8

Stunde 9+10

Stunde 11+12





Management of the process of the pro





#### SQL mit Kreuzprodukt und Join

Bearbeite diese Aufgabe auf art em is, 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 tu. 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. Notiere unten anschließend deine korrekten SQL-Abfragen unten.

#### Join Beispiel Lehrkraft Schule **SELECT**\* kuerzel schule id ort FROM Lehrkraft, Schule Her MTG MTG Haidh. WHERE Lehrkraft.schule = Schule.id Ext Dante Dante Sendl. id kuerzel schule id ort 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.

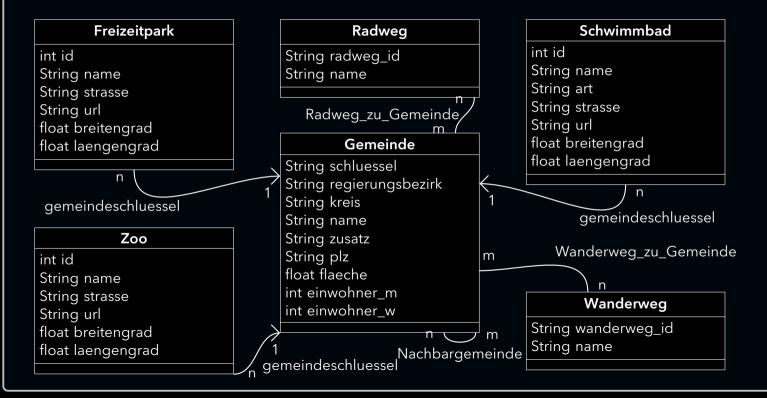
Gib immer genau die geforderten Daten aus und nicht mehr. Sortiere nicht, wenn du nicht dazu aufgefordert wirst.

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')