

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

Stunde 5+6

SQL Puzzle

Wdh: SQL Basics

Stunde 7+8
Tabellenbeziehungen
Tabellenbeziehungen: Fremdschlüssel

Stunde 9+10
Tabellenbeziehungen im Klassendiagramm
Kardinalitäten

Klassendiagramm Flugverspätung SQL: Tabellen verbinden Kreuzprodukt / Join

Stunde 11+12
Join Beispiel



SQL mit Kreuzprodukt und Join

Outline

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

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



spitze Ecken

runde Fcken

Objektkarten Memory Erstelle auf einem Blatt eine Objektkarte der Klasse Person zu dir selbst. → 3x falten Gib deine Obiektkarte 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 Der/Die Finder:in sammelt alle gefundenen Objekte.



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

Objektart gespeichert werden sollen. Man stellt sie dar mit:

Klassenkarte



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

Klassenkarte

Objektkarte

sind der Bauplan,



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

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



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



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()



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



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

 \leftarrow Methoden

spitze Ecken



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()

← Klassenname

 ${\sf Objektname}: {\sf Klassenname} \to$

Attribute

 \leftarrow Methoden

Objektkarte

p1 : Person

hobby = "Klettern"

alter = 23

hatHaustier = false

peinlichesErlebnis = "..."

spitze Ecken



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

 ${\sf Objektname}: {\sf Klassenname} \rightarrow$

Attribute

 \leftarrow Methoden

Objektkarte

p1: Person

hobby = "Klettern"

alter = 23

hatHaustier = false

peinlichesErlebnis = "..."

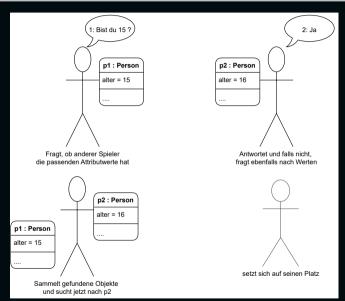
runde Ecken

spitze Ecken

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

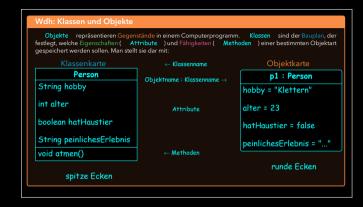
Stunde 3+4

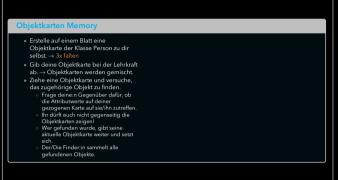
e 5+6

tunde 7+8

stunde 9+10

tunde 11+12









SOL Spirksvettell

Trippinger SSL Spirksvettell smithel side SSL Grandingen der P. Gauss. für dicht (sahr) der bes allen SSL

Anglaten benutzen Ghar des Vorlagensyntest (j), oben körnet fir den Spirksvett en segnen TSF deben.



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:



Nicht verwendete Begriffe: Parameter, Methode, Board, Datentyp

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

Der Aufbau einer Tabelle kann mit

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

oder

dargestellt werden. Dessen

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 unterstrichen und im Klassendiagramm imm</u>er als erstes Attribut aufgelistet.

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

Datenbanken speichern Datensätze in Tabellen repräsentieren die (=Zeilen) entsprechen

(oft auch

Attribute (Synonym: Feld) und bilden zusammen eine Klasse. Die Objekten und in den Spalten stehen die Attributwerte. Jede Tabelle hat einen

"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

Aufbau ist: TABELLENNAME(Datentyp Primärschlüssel, Datentyp Spaltel, Datentyp Spaltel, ...)

Aufbau ist:

Datenbanken speichern Datensätze in **Tabellen** . Die Spaltenüberschriften repräsentieren die

Attribute (Synonym: Feld) und bilden zusammen eine Klasse. Die (=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

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 Spaltel, Datentyp Spaltel, ...)

Der Aufbau einer Tabelle kann mit

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

oder

(oft auch

dargestellt werden. Dessen

Attribute (Synonym: Feld) und bilden zusammen eine Klasse. Die Datensätze Objekten und in den Spalten stehen die Attributwerte. Jede Tabelle hat einen

"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.

Aufbau ist: TABELLENNAME(Datentyp Primärschlüssel, Datentyp Spaltel, <u>Datentyp Spaltel</u>, ...)

Zum Beispiel: Jatentyp Primarschlussel, Datentyp Spaltel, Datentyp

Aufbau ist:

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

TABELLENNAME(Datentyp Primärschlüssel, Datentyp Spaltel, Datentyp Spaltel, ...)

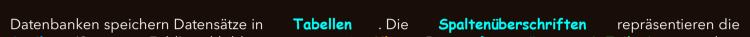
Der Aufbau einer Tabelle kann mit Klassenkarte oder



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.

dargestellt werden. Dessen

Aufbau ist: TABELLENNAME(Datentyp Primärschlüssel , Datentyp Spalte1, Datentyp Spalte2, ...)



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, ...)

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

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.

Inf 9	Spickzettel SQL				
SELECT	paltenliste - Aggregatiunktionen (COUNT, SUM, MAX, MIN, AVG) für Berechnunger - SelECT* für 'alle Spalten' - AA Allasmannen				
WHERE GROUP BY	Bedingung				
HAVING	cedingung -ASC für aufsteigend (Standard) OESC für absteigend				
Im Detail gilt:					
SELECT	Es muss mindestens ein Spaltenname angegeben werden. Die entsprechende(n) Spalte(n) sind dann Teil der Ergebnistabelle. SELECT * bewirkt, dass alle Spalten angezeigt werden.				
DISTINCT	Duplikate von Datensätzen werden nicht angezeigt.				
AS	Eine Spalte in der Ergebnistabelle kann anders benannt werden als in der Ausgangstabelle. Dies ist vor allem bei der Verwendung von Aggregatfunktionen hilfreich.				
FROM	Hier muss angegeben werden, aus welcher Tabelle die Informationen für die Abfrage genommen werden sollen.				
ORDER BY	Die Ergebnistabelle wird nach der oder den angegebenen Spalten sortiert. Standardmäßig wird aufsteigend sortiert. Mit dem Zusatz DESC bzw. ASC wird absteigend bzw. aufsteigend sortiert.				
Beispiele	SELECT DISTINCT kontinent AS "enthaltene Kontinente" FROM Land				
	SELECT name, flaeche, hauptstadt FROM LAND ORDER BY flaeche DESC				

Auswahl von Datensätzen über Bedingungen			
WHERE	In der Ergebnistabelle werden nur die Datensätze (Zellen) 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 LIKE wichtig. Mehrere Vergleiche können durch die logischen		

	Facktsonen AND, OR und HOT verknight werden. Ggf. müssen die einzelnen Ausdrücke dabei sinnvoll gektammert werden Beispiel WHERE 2015 MICHAEL 2015 MICH
LIKE	Kann in einer Bedingung zur Mustererkennung vom Einträgen verwendet werden. Fölgende zwei Parhahter (wildcard) werden häufig eingesetzt: * % steht für beliebig viele Zeichen, auch keines (* bei MS Acces) _ für genau ein beliebiges Zeichen (* bei MS Acces) Belspiele: * WHERE thel LIKE "You%" – findet alle Titel die mit "You" beginnen Groß, // Keinchveibung wird nicht berücksichtigt * WHERE Teile LIKE "Slowes" – Indet alle Titel die mit 1 beginnen und genau 4 Zeichen lags 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.

Aggregatfunktionen			
AVG	Berechnet (den Durchschnitt aller Werte einer Spalte.	
COUNT	Gibt die An	zahl 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		COUNT(*) AS "Anzahl afrikanischer Länder"	
	FROM	Land	
	WHERE	kontinent = "Afrika"	

Gruppierung	
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!**

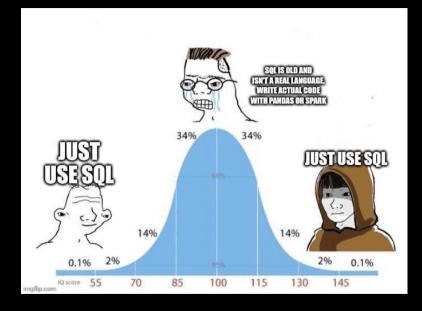


SELECT name, id FROM products WHERE discount = 0 ORDER BY price ASC;



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.

3. Spielt SQL Island, der SQL-Spickzettel hilft euch dabei.

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.

3. Spielt SQL Island, der SQL-Spickzettel hilft euch dabei.

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 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.

BEWOHNER

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 hacuptling

3. 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

tunde 11+12

Desembenken peschen Desemben in Tabellen. De Spittenberschriften reprisenten die Ambünd (Springer Hell) und bilden zusammen weise Vallen. Desemben 1. Stellen in retrigerbeiter (Springer Hell) und bilden zu der der Vallen der Vallen von der Vallen

TABELLENNAME(Datestyp Primirschlüssel , Datestyp Spaltel, Datestyp Spalte2, ...)

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



In dieser Aufgahe geht es immer um die Tahelle land, deren erste Datensätze du hier siehet

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

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 - 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. 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!

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! 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

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

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!

1) iv)		4)
2) viii)		5) i
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!

1) 10)	•	4)
2) viii)	5	5)
3) vii)	6	5)



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!

1) iv)	4) i)	7) v)
2) viii)	5) ix)	
3) vii)	6) iii)	



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

schland	83.24	0.50	
0 0 0 0 .	03.24	358	Berlin
ıkreich	67.39	544	Paris
ısilien	212.60	8516	Rio de Janeiro
	nkreich asilien 	asilien 212.60	asilien 212.60 8516

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

1) iv)	4) i)	7) 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
Wolcho SOL A	hfraga (rachta Saita) füh	rt zu wolcher Ergebai	stabollo (linko Soito)?	Ordno richtia zul

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

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 '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 '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.

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



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.



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.

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.



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.

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,

 $\label{eq:continuous} \mbox{die jeweils mehr als 50.000 m\"{a}nnliche und weibliche Einwohner:} innen oder eine Fl\"{a}che gr\"{o}\mbox{Ber als 100 km}^2 \mbox{ hat.}$

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.



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. 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

tunde 11+12

SQL Puzzlo

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
	Frankreich	67.39	544	Paris
	Brasilien	212.60	8516	Rio de Janeiro

l ösuna:

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

Wdh: SQL Basics

Bearbeite die Aufgabe Wdh - SQL Basics auf anz temis . 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. 1 ink / Skrimplec lub - group - sort - aggregat

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

Tabellenbeziehunger

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

Tabellenbeziehungen: Fremdschlüssel

Wenn Datensätze mittels Primärschlüssel in einer anderen Tabelle verwendet werden, <u>spricht man dort von</u> einem Fremdschlüssel. Im Tabellenschens werden die Frendschlüssel durch <u>überstreichen</u> (manchmal auch <u>unterpunkten</u>) markiert. Ein Beispiel in SQL Island ist der Häuptling eines Dorfes, der in der Tabelle Dorf mittels bewohnerm eingetragen wird. Die bewohnerm ist hierbei <u>Primärschlüssel</u> in der Tabelle Bewohner mit Stein ber Stein ber sein bei der Tabelle Bewohner und Fremdschlüssel in der Tabelle Dorf (helbt hier aber hauptling).



- 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		



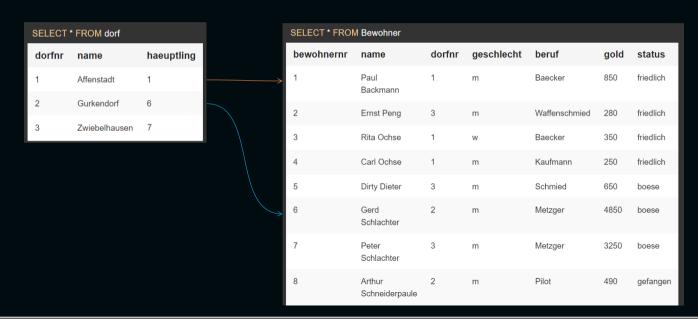
- 1. Visualisiere (mit Bleistift), wer Häuptling in welchem Dorf ist.
- 2. Überlege, wie du allgemein für diese zwei Tabellen darstellen kannst, wie sie (und ihre Spalten) miteinander in Beziehung stehen.

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	

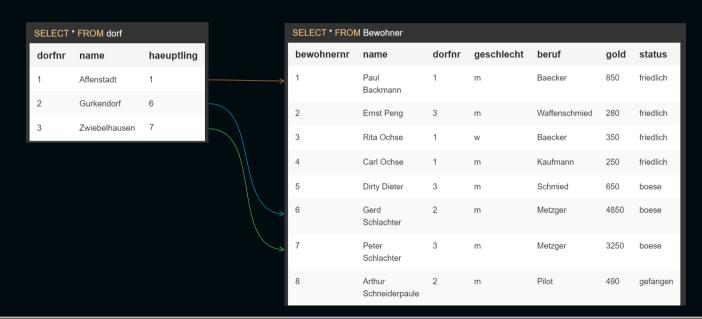


- 1. Visualisiere (mit Bleistift), wer Häuptling in welchem Dorf ist.
- 2. Überlege, wie du allgemein für diese zwei Tabellen darstellen kannst, wie sie (und ihre Spalten) miteinander in Beziehung stehen.



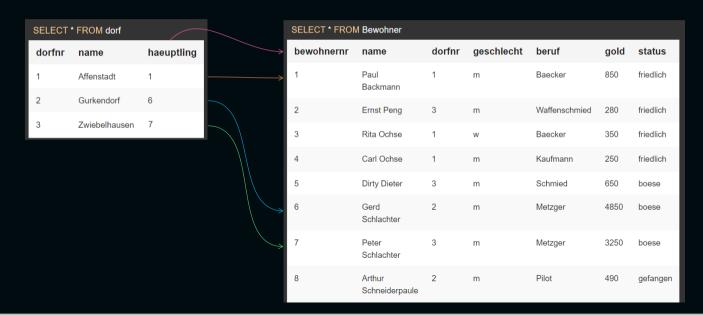


- 1. Visualisiere (mit Bleistift), wer Häuptling in welchem Dorf ist.
- 2. Überlege, wie du allgemein für diese zwei Tabellen darstellen kannst, wie sie (und ihre Spalten) miteinander in Beziehung stehen.





- 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.

SELECT	FROM dorf		SELECT * FROM	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
			6	Gerd Schlachter	2	m	Metzger	4850	boese
			7	Peter Schlachter	3	m	Metzger	3250	boese
			8	Arthur Schneiderpaule	2	m	Pilot	490	gefangen

Tabellenbeziehung im Klassendiagramm



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

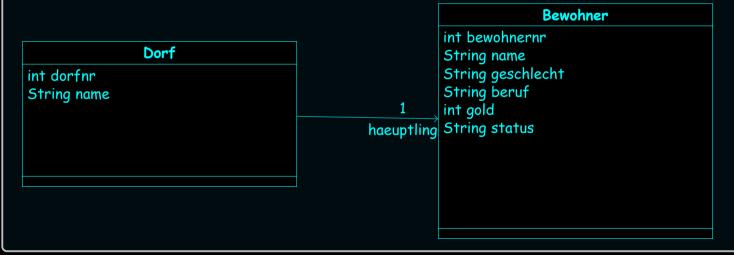
Dorf
int dorfnr
String name

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

Tabellenbeziehung im Klassendiagramm



- 1. Ergänze das Klassendiagramm entsprechend der 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 .

Tabellenbeziehungen: Fremdschlüssel

(manchmal auch

Wenn Datensätze mittels Primärschlüssel in einer anderen Tabelle verwendet werden, spricht man dort von einem Fremdschlüssel. Im Tabellenschema werden die

Fremdschlüssel

durch

Tabellenbeziehungen: Fremdschlüssel

(manchmal auch

einem Fremdschlüssel. Im Tabellenschema werden die

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

Fremdschlüssel

durch

überstreichen

Tabellenbeziehungen: Fremdschlüssel

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

Outline

Stunde 1+2

de 5+6

Stunde 7+8

Stunde 9+10

tunde 11+12

Tabellenbeziehungen

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

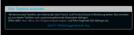
Tabellenbeziehungen: Fremdschlüssel

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 "unfterpunkten"), markiert. Ein Beispiel in SQL-Island ist der Häuptling eines Dorfes, der in der Tabelle Dorf mittels bewohnern eingetragen wird. Die bewohnern ist hierbei Primärschlüssel in der Tabelle Bewohner und Fremdschlüssel in der Tabelle Office in der Practiculation in der Tabelle Bewohner und Fremdschlüssel in der Tabelle Dorf mittels bewohner und Fremdschlüssel in der Tabelle Dorfes in der Practiculation in der Tabelle Bewohner und Fremdschlüssel in der Tabelle Dorfes in der Dorfes der Bewohner und Fremdschlüssel in der Tabelle Dorfes in der Bewohner und Fremdschlüssel in der Tabelle Dorfes in der Bewohner und Fremdschlüssel in der Tabelle Bewohner und Fremds









Companyable 2 into
Within an alloware can be the first ordinately assessed in this year, given be for Tables and commission of the commis







Tabellenbeziehungen im Klassendiagramm





- 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:

• 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

s aber Bewohner hat.

Tim, 2.B. jeder bewormer wormern einem borr, das aber

Schulklassen pro Lehrer (in

Datenbanken nicht direkt umsetzbar, dazu später mehr).

• m:n, z.B.

Schulklassen pro Lehrer (in

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

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

Lehrer pro Schulklasse +

Datenbanken nicht direkt umsetzbar, dazu später mehr).

• m:n, z.B.



Schulklassen pro Lehrer (in

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

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

Lehrer pro Schulklasse +

Datenbanken nicht direkt umsetzbar, dazu später mehr).

• m:n, z.B.

Schulklassen pro Lehrer (in

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

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

Lehrer pro Schulklasse +

Datenbanken nicht direkt umsetzbar, dazu später mehr).

beliebig viele

• m:n, z.B.



Schulklassen pro Lehrer (in

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

beliebig viele

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

Lehrer pro Schulklasse +

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

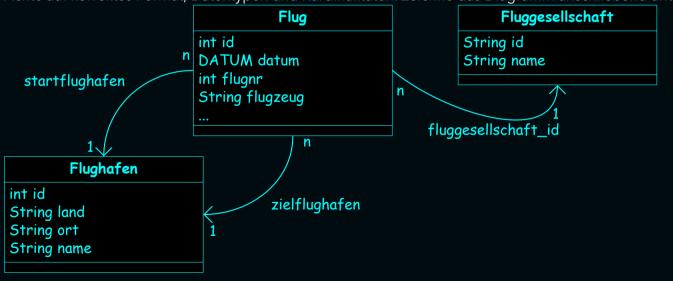


Bearbeite diese Aufgabe auf artemis.tum.de.

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:



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



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

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 . 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:).

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 . 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 **Primärschlüssel** Dann spricht man von einem

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

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 **Primärschlüssel** . Dann spricht man von einem **Join** . Zum Beispiel kann man in SQL-Island die Daten aller Dörfer und ihrer zugehörigen Häuptlinge so ausgeben:

SELECT *

FROM Dorf, Bewohner

Outline

Stunde 1+2

3+4

e 5+6

Stunde 7+8

stunde 9+10

Stunde 11+12









Kenampudak Jain Milliam and Tandan und Tanda



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 Datenhahr 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.

Notiere unten anschließend deine korrekten SQL-Abfragen unten

Jo	oin E	Beispiel	
		Lehrkra	aft
	id	kuerzel	S
	1	Her	

Ext

schule MTG Dante

SELECT * FROM Lehrkraft, Schule

kuerzel

Her

Ext

Her

Ext

kuerzel

Her

Ext

id

2

id

WHERE Lehrkraft.schule = Schule.id

schule

MTG

Dante

MTG

Dante

schule

MTG

Dante

MTG MTG

Dante

Dante

id

MTG

Dante

Ergebnistabelle des Kreuzprodukts:

id

ort Haidh. Schule

ort

Haidh.

Sendl.

id

MTG

Dante

Haidh.

Sendl.

Sendl.

Ergebnistabelle des Joins

ort	
laidh.	
endl.	



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.

Notiere unten anschließend deine korrekten SQL-Abfragen unten.



SQL mit Kreuzprodukt und Join







★ 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



★ 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 ,Gemeinde.regierungsbezirk, Zoo.url

FROM Zoo, Gemeinde



► 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 ,Gemeinde.regierungsbezirk, Zoo.url

FROM Zoo, Gemeinde

WHERE Zoo.gemeindeschluessel = Gemeinde.schluessel



SQL mit Kreuzprodukt und Join



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



SQL mit Kreuzprodukt und Join



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, Gemeinde.einwohner_w

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(*)

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, 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!

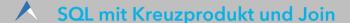
SELECT Gemeinde.name

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')