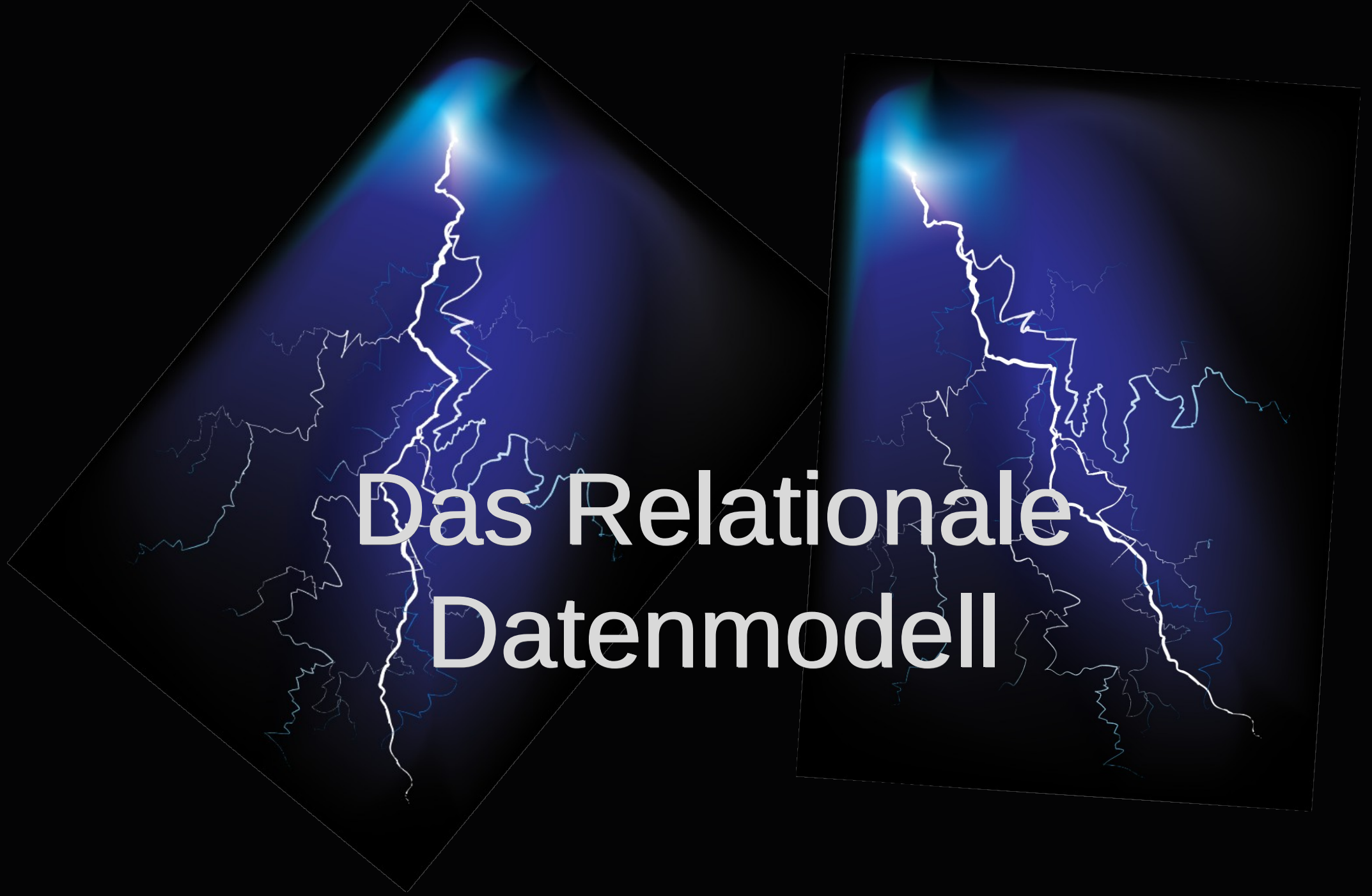
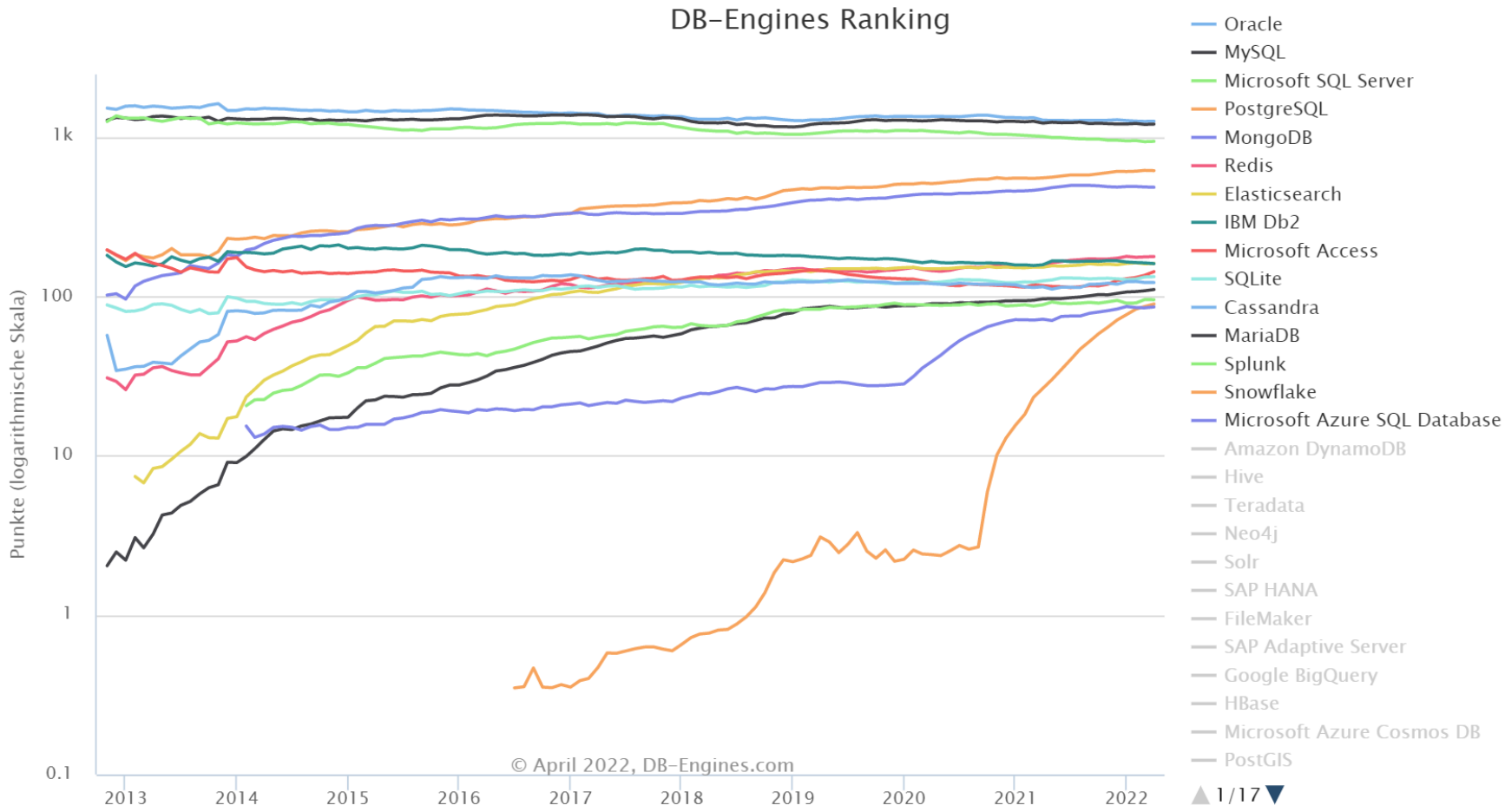


Das Relationale Datenmodell

3. Das Relationale Datenmodell

1. **Das Datenmodell**
2. Transformation von ER-Diagrammen in das Relationale Modell
3. Relationale Algebra
4. Relationaler Kalkül

Datenbankranking



https://db-engines.com/de/ranking_trend

Datenbanken – Wie machen es die “Großen”?
Beispiel: Google Spanner und Facebook TAO

Google Spanner [1][2]:

- Verteilte “NewSQL” Datenbank mit “multi-version concurrency control”.
  - “Semi-relationales” Datenmodel.
  - Unterstützt SQL Anfragen und Transaktionen.

Facebook TAO [3]:

- Verteilte Graphdatenbank – verwendet MySQL zur Persistenz.
  - Optimiert für Lesen.
  - Wertet Effizienz und Verfügbarkeit höher als Konsistenz.

TAO

[1] J.C. Corbett et al.: Spanner: Google's Globally-Distributed Database. OSDI 2012: 251-264
[2] D. F. Bacon et al: Spanner: Becoming a SQL System. SIGMOD Conference 2017: 331-343
[3] N. Bronson, et al.: TAO: Facebook's Distributed Data Store for the Social Graph. USENIX Annual Technical Conference 2013: 49-60

Einführung

- Das relationale Datenmodell nutzt das einfache Strukturierungsprinzip “Tabelle” zur Modellierung sowohl von Objekten als auch von Beziehungen.

- Das Modell geht auf *Codd* (1970) zurück und hat sich seither auf breiter Basis durchgesetzt.
  - Edgar F. Codd: *A Relational Model of Data for Large Shared Data Banks*. Commun. ACM 13(6): 377-387 (1970), see http://dblp.dagstuhl.de/pers/hd/c/Codd:E=_F=

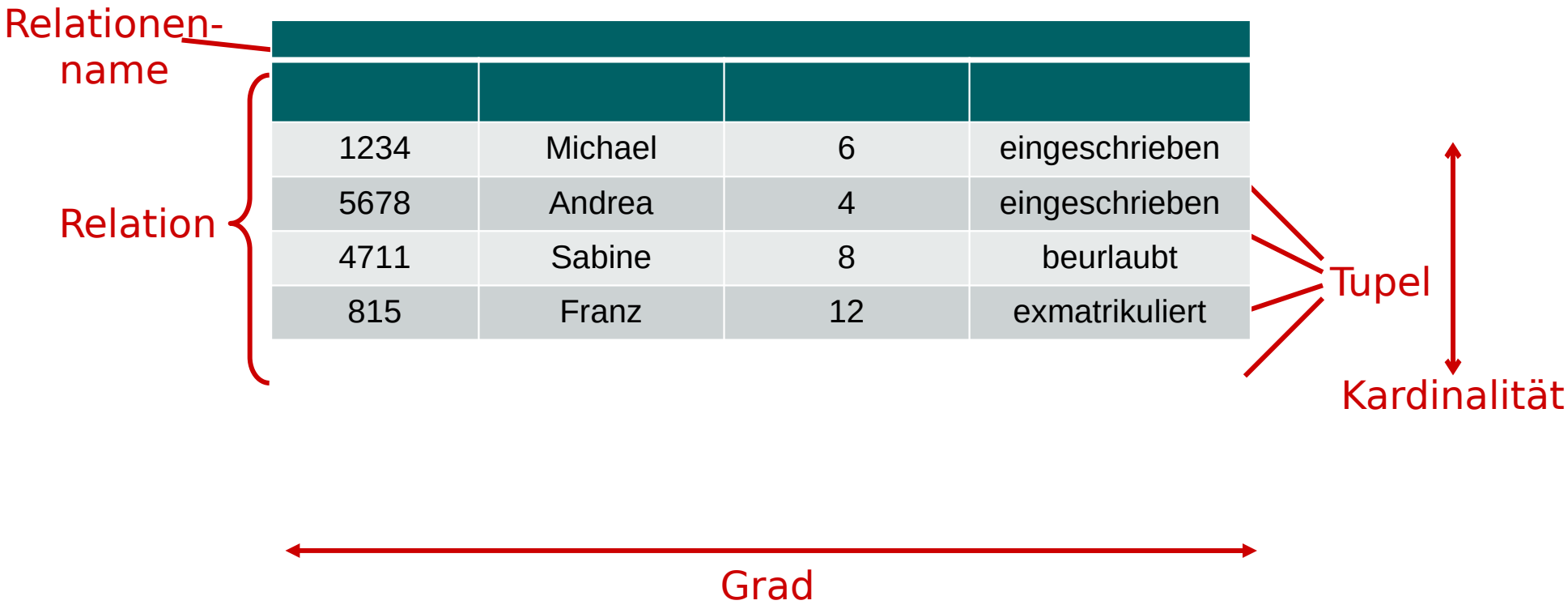
- Heute ist es Grundlage vieler kommerzieller Datenbanksysteme (Oracle, IBM DB/2, Informix, Ingres, Sybase, MS SQL-Server, MS Access, usw.) und damit wichtiger Bestandteil vieler großer Anwendungssysteme.

- Im naturwissenschaftlich-technischen Bereich dient es vielfach als Grundlage für komplexere Datenmodelle, insbesondere für sogenannte “Nichtstandard-Anwendungen”.

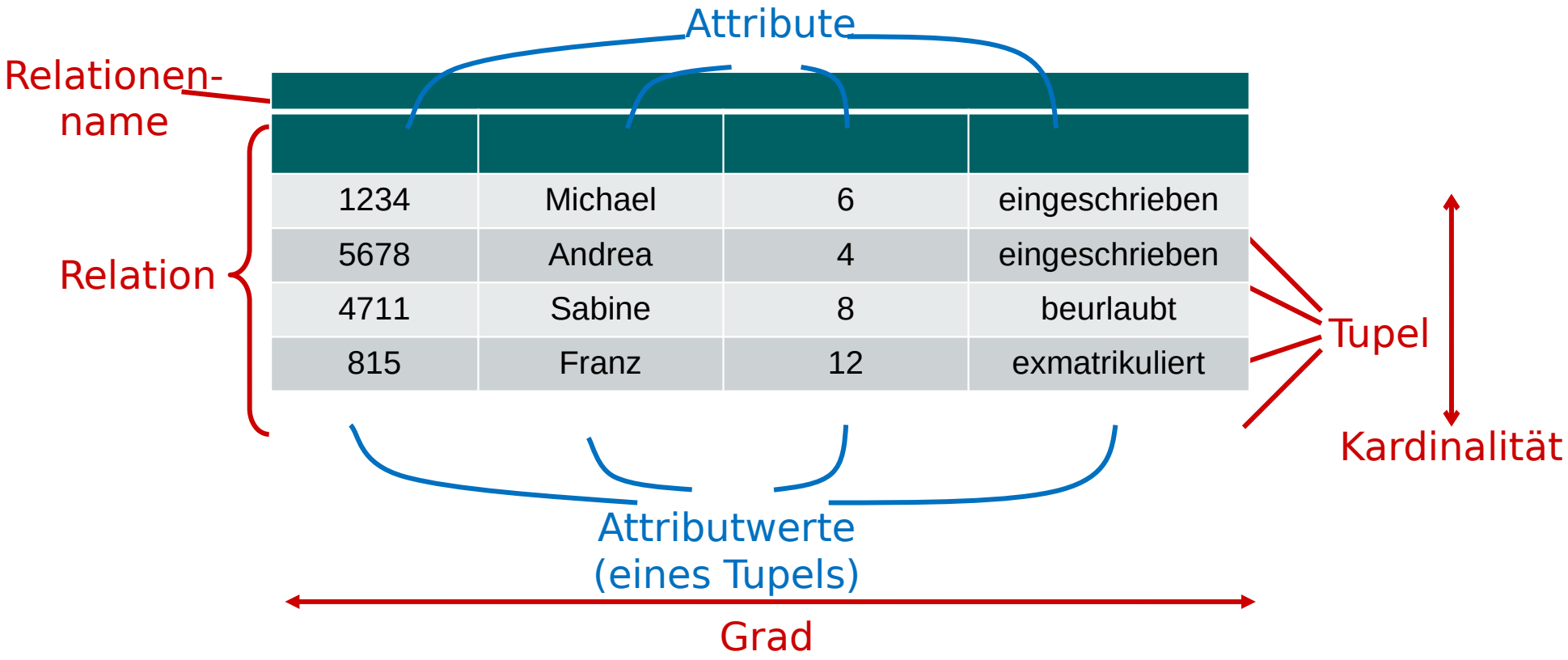
Relationenschemata Terminologie

1234	Michael	6	eingeschrieben
5678	Andrea	4	eingeschrieben
4711	Sabine	8	beurlaubt
815	Franz	12	exmatrikuliert

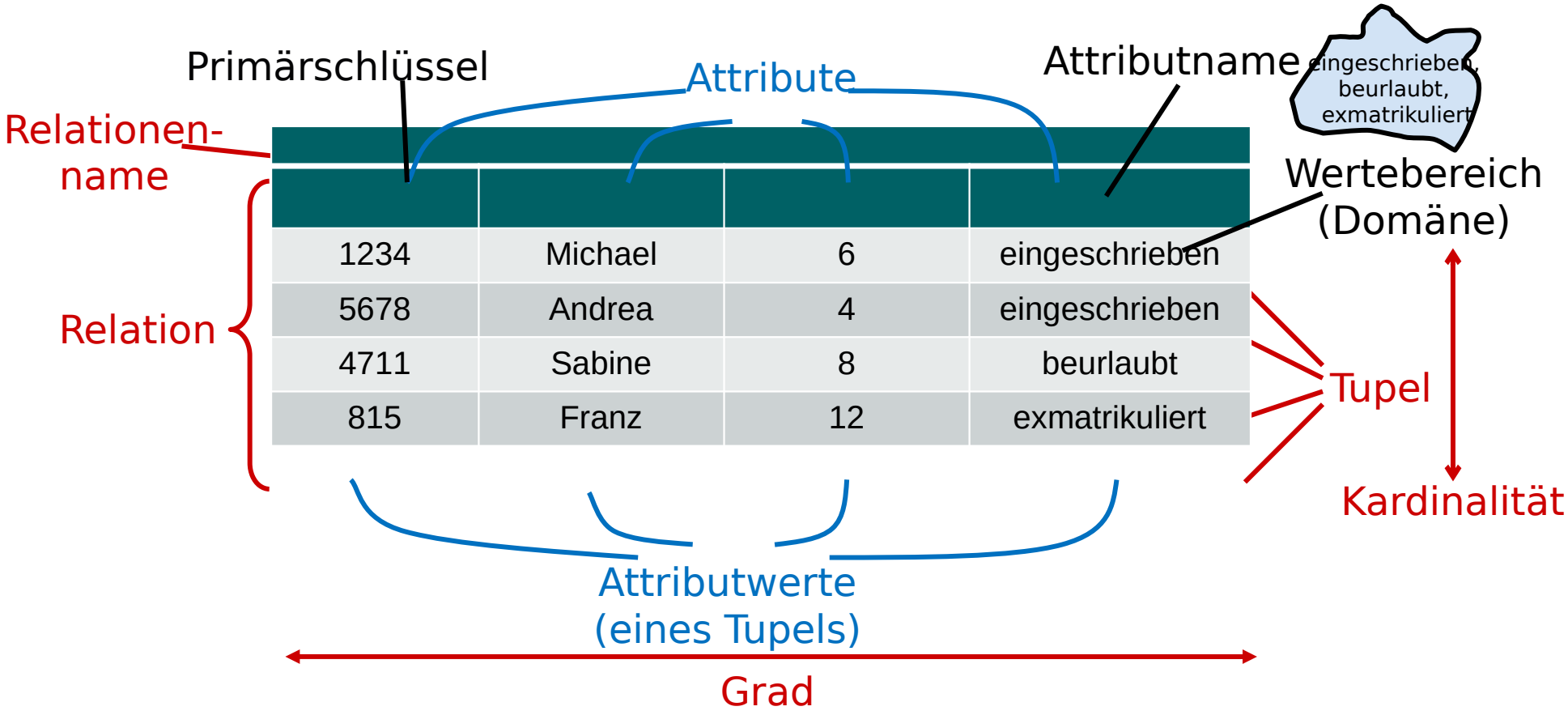
Relationenschemata Terminologie



Relationenschemata Terminologie



Relationenschemata Terminologie



1234	Michael	6	eingeschrieben
5678	Andrea	4	eingeschrieben
4711	Sabine	8	beurlaubt
815	Franz	12	exmatrikuliert

Relationen, Domänen

- Ein *Wertebereich (Domäne, Domain)* ist eine (logisch zusammengehörige) Menge von Werten, z.B. INTEGER, STRING, DATUM, {1, … , 10}, etc. Eine Domäne kann *endliche* oder *unendliche* Kardinalität haben. Die Funktion bildet Attribute auf ihren Wertebereich ab.

- Ist und eine Folge von Attributnamen mit den Wertebereichen , dann ist eine *Relation r* Teilmenge des kartesischen Produktes der Wertebereiche :

- *Beispiel*
  - Gegeben seien die Wertebereiche und . Dann gibt es unter anderem die Relationen:

Tupel, Tabellen

- Einzelne Elemente einer Relation heißen *Tupel*.
- Für heißt der *Grad* oder die *Stelligkeit* der Relation; alle Tupel in haben Komponenten.

- Relationen kann man als *Tabellen* verstehen und darstellen. Die Zeilen einer Tabelle entsprechen den Tupeln. Die Spalten heißen Attribute; sie können Namen tragen, im Beispiel *MatNr* und *Name*.

- Bemerkung: Im Datenbankbereich betrachtet man gewöhnlich nur endliche Relationen. Unendliche Relationen können nicht materialisiert werden, sie treten z.B. im Bereich der deduktiven Datenbanksysteme auf.

30487	Bob
30548	Ede
30375	Anna
30455	Pia

$$D_1 = \text{Integer} = \{1, 2, \ldots\}, D_2 = \text{String}$$

Relationenschemata: Zwei Alternativen

- **Geordnetes Relationenschema**
  - Ein *geordnetes einfaches Relationenschema* ist ein -Tupel aus Bezeichnungen für Wertebereiche (*Domains*) (). Die Komponenten eines Relationenschemas heißen Attribute; sie können benannt sein: .

- **Tupel als Abbildung**
  - Ein einfaches *Relationenschema* ist eine endliche Menge unterschiedlicher Attributsnamen . Jedem Attributsnamen () ist ein Wertebereich (*Domain*) zugeordnet: .
  - Sei . Eine *Relation* über dem Relationenschema mit Tupeln ist eine endliche Menge von Abbildungen von nach . Dabei gilt für alle , , und stets:

- Die beiden Definitionen sind prinzipiell gleichwertig
  - Auf die Attributwerte eines Tupels kann man über die Namen zugreifen: ,
  - Im geordneten Schema geht das auch über die Position: , .
  - Im folgenden verwenden wir jeweils die einfacher anzuwendende Alternative.

Relationen, Reihenfolgen, Datenbanken

- Reihenfolgeabhängigkeiten
  - Die Reihenfolge der Zeilen (= Tupel) spielt keine Rolle (Relation ist Menge).
  - Im geordneten Relationenschema ist die Reihenfolge der „Spalten“ wichtig:
  - Im ungeordneten Relationenschema (Abbildungsmodell) ist die Reihenfolge unwichtig:
  - Vergleiche hierzu das JSON-Datenformat (json.org)
    - „array“: geordnete Aufzählung
    - „object“: Abbildungsmodell (key, value)
- **Relation, Datenbank**
  - Eine *Relation* ist eine Ausprägung eines Relationenschemas.
  - Ein *Datenbankschema* ist eine Menge von Relationenschemata.
  - Eine *Datenbank* ist eine Menge der aktuellen Relationen (Ausprägungen)

Beispiel: Relation Städte

Städte		
Name	*Einwohner*	*Land*
Aachen		NRW
München	1.211.617	Bayern
Bremen	535.058	Bremen
Bonn		NRW

- Als *“geordnetes Relationenschema”*:
  - Schema: (Name: STRING, Einwohner: INTEGER, Land: STRING)
  - Ausprägung: {(München, 1.211.617, Bayern), (Bremen, 535.058, Bremen)}

- Gemäß *“Tupel als Abbildung”*:
  - Schema: {Name, Einwohner, Land} mit dom (Name) = STRING, dom (Einwohner) = INTEGER, dom (Land) = STRING
  - Ausprägung: {*t*1, *t*2} mit:
    - *t*1 (Name) = München, *t*1 (Einwohner) = 1.211.617, *t*1 (Land) = Bayern
    - *t*2 (Name) = Bremen, *t*2 (Einwohner) = 535.058, *t*2 (Land) = Bremen

Schlüssel

- Eine minimale Teilmenge der Attribute eines Relationenschemas, anhand der alle Tupel einer (möglichen) Relation unterscheidbar sind, heißt Schlüssel.

- Definition *(Schlüssel):*
  - Eine Teilmenge der Attribute eines Relationenschemas ist ein *Schlüssel* (genauer *Schlüsselkandidat*), wenn gilt:
    - (i) **Eindeutigkeit:** Keine Ausprägung von kann zwei verschiedene Tupel enthalten, die sich in allen Attributen von gleichen.
    - (ii) **Minimalität:** Keine echte Teilmenge von erfüllt die Bedingung (i).

- Formal:
  - Sei eine Relation über dem Schema und eine Teilmenge der Attribute von ; bezeichne die Einschränkung (Projektion) des Tupels auf die Attribute in .
  - ist ein Schlüssel der Relation , wenn gilt:
    - (i) Für alle möglichen Ausprägungen und Tupel gilt:
    - (ii) Für alle Attributmengen , die (i) erfüllen, gilt:

Beispiel für Schlüssel

Personal	
PNr	*Gehalt*
1	1.700 €
2	2.172 €
3	3.189 €
4	2.167 €

- *Wichtig:*
  - Die Schlüsseleigenschaft ist abhängig von der Semantik des Schemas, nicht von der aktuellen Ausprägung einer Relation!
- *Beispiel*: Relation mit Personalnummer und Gehalt
  - In der Relation *Personal* erfüllen sowohl die Attributmengen als auch die Bedingung (*i*). Schlüssel ist jedoch nur , da Personalnummern logisch stets eindeutig sind, Gehälter jedoch nicht.
  - Auch erfüllt Bedingung (*i*), kann jedoch auf reduziert werden und ist deshalb kein Schlüssel.
- Gelegentlich gibt es mehrere Schlüsselkandidaten für eine Relation, unter denen dann einer als *Primärschlüssel* ausgewählt wird (üblicherweise der mit den kürzesten Attributswerten).
- Wie bereits im ER-Diagramm werden die zum Schlüssel gehörigen Attribute im Schema unterstrichen: Mitarbeiter (PNr, Gehalt)

Das Relationale Datenmodell

1. Das Datenmodell
2. **Transformation von ER-Diagrammen in das Relationale Modell**
3. Relationale Algebra
4. Relationaler Kalkül

Informatik 5 -
Informationssysteme
und Datenbanken

RWTH AACHEN
UNIVERSITY

Datenabhängigkeiten

- Im relationalen Modell:
  - Objekte (Entities) und alle Arten von Beziehungen (Relationships) werden durch Relationen dargestellt

- keine Einschränkungen: an den Relationen können beliebige Mengen von Attributen beliebiger Objekte beteiligt sein
  - auch „ungültige“ Kombinationen können auftreten

- Integritätsbedingungen um Konsistenz der Datenbank zu gewährleisten:
  - intra- und interrelationale Abhängigkeiten
  - wichtig zur Modellierung von 1:n-, Is-A-Beziehungen, uvm.

Relationenschema

- Intrarelationale Abhängigkeiten
  - Beispiel: Schlüssel einer Relation
  - Erlaubt Aussagen über die Konsistenz/Gültigkeit einer Relation
    - Ein Schlüssel einer Relation darf in der Relation nicht doppelt vorkommen

- Formal: Erweiterung der Relationenschemata
  - Es bezeichne eine Menge von Attributen
  - bezeichnet die Menge aller Tupel über
  - ist die Menge aller Relationen über
  - eine intrarelationale Abhängigkeit über ist eine Abbildung

formal spezifiziert sie, welche der möglichen Relationen eine gegebene Bedingung erfüllen

Relationenschema

- Intrarelationale Abhängigkeiten
  - Beispiel: Schlüssel einer Relation
  - Sei . Eine Schlüsselabhängigkeit bezeichnet folgende intrarelationale Abhängigkeit

- ist ein *Relationenschema*, dabei sind
  - ein Relationssymbol
  - eine Menge von Attributen
  - eine Menge intrarelationaler Abhängigkeiten (Schlüsselabhängigkeiten) über

- eine Relation mit Relationenschema heißt *gültig* oder *konsistent*, falls sie alle intrarelationalen Abhängigkeiten erfüllt

Datenbankschema

- Interrelationale Abhängigkeiten
  - Beispiel:
  - Erlaubt Aussagen über die Konsistenz/Gültigkeit einer Datenbank
    - jede Matrikelnummer die in der Relation enthalten ist, muss auch in einem Eintrag in existieren

- Formal: Erweiterung der Datenbankschemata
  - Es bezeichne eine endliche Menge von Relationenschemata
  - Eine Datenbank ist eine Menge von Relationen (Ausprägungen)
  - bezeichnet die Menge aller Datenbanken über
  - eine interrelationale Abhängigkeit über ist eine Abbildung

formal spezifiziert sie, welche der möglichen Datenbanken die gegebene Bedingung erfüllen

Datenbankschema

- Interrelationale Abhängigkeiten
  - Beispiel:
  - Inklusionsabhängigkeit: Sei eine endliche Menge von Relationenschemata
    und , eine bijektive Abbildung sowie

Eine Inklusionsabhängigkeit:

bezeichnet folgende interrelationale Abhängigkeit

Anschaulich: jede Matrikelnummer die in der Relation enthalten ist, muss auch in einem Eintrag in existieren. ist bei gleichen Attributnamen die Identitätsfunktion, ansonsten wird die Definition von aus dem Kontext klar, da die Attributnamen aufeinander abbildet.

Datenbankschema

- Interrelationale Abhängigkeiten
  - Beispiel:
  - Zweiter Typ, Exklusionsabhängigkeit: Sei wieder eine endliche Menge von Relationenschemata, , eine bijektive Abbildung sowie . Eine Exklusionsabhängigkeit

    bezeichnet die Abhängigkeit

Datenbankschema

- ist ein *Datenbankschema*, dabei sind
  - eine Menge von Relationenschemata
  - eine Menge interrelationaler Abhängigkeiten über
- Eine Datenbank mit Datenbankschema heißt *konsistent* oder *gültig*, falls alle intra- und interrelationalen Abhängigkeiten erfüllt sind

Fremdschlüssel

- Inklusionsabhängigkeiten: Modellierung von – und Is-A – Beziehungen

- Sei ein Datenbankschema. Eine Attributmenge eines Relationenschemas heißt Fremdschlüssel, bezogen auf eine Relation , falls
  - den selben Wertebereich besitzt wie der Schlüssel von
  - die interrelationale Abhängigkeit , wobei der Schlüssel von ist, gilt in , d.h.

- Die zweite Bedingung nennt man auch *referentielle Integrität* des Fremdschlüssels
  - jedes Tupel in den Fremdschlüsselattributen verweist auf ein existierendes Tupel der referenzierten Relation oder die Fremdschlüsselattribute sind auf Null gesetzt

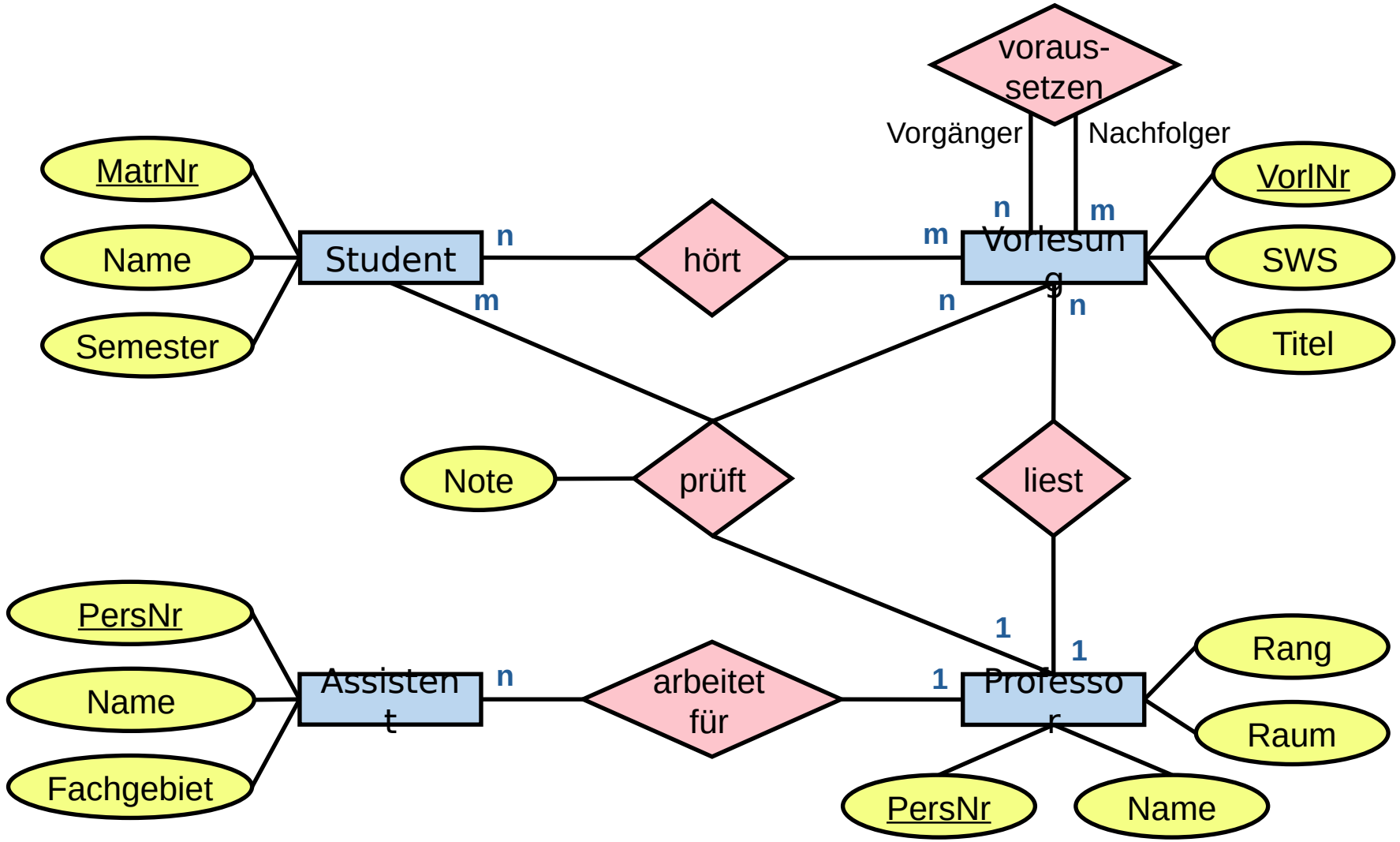
ER Modell und Relationales Modell

- konzeptueller Entwurf: ER-Modell
  - Entity-Typen
  - Beziehungs-Typen

- mathematischer Entwurf: Relationales Modell
  - nur ein Strukturierungskonzept: Relationen

- Entities und Beziehungen werden jeweils auf Relationen abgebildet

Beispiel : ER-Diagramm (Universität)



Beispiel für eine Datenbank

Professor			
PersNr	**Name**	**Rang**	**Raum**
2125	Sokrates	W3	226
2126	Russel	W3	232
2127	Kopernikus	W2	310
2133	Popper	W2	52
2134	Augustinus	W2	309
2136	Curie	W3	36
2137	Kant	W3	7

Student		
MatrNr	**Name**	**Semester**
24002	Xenokrates	18
25403	Jonas	12
26120	Fichte	10
26830	Aristoxenos	8
27550	Schopenhauer	6
28106	Carnap	3
29120	Theophrastos	2
29555	Feuerbach	2

Vorlesung			
VorlNr	**Titel**	**SWS**	**gelesenVon**
5001	Grundzüge	4	2137
5041	Ethik	4	2125
5043	Erkenntnistheorie	3	2126
5049	Mäeutik	2	2125
4052	Logik	4	2125
5052	Wissenschaftstheorie	3	2126
5216	Bioethik	2	2126
5259	Der Wiener Kreis	2	2133
5022	Glaube und Wissen	2	2134
4630	Die 3 Kritiken	4	2137

hört	
MatrNr	**VorlNr**
26120	5001
27550	5001
27550	4052
28106	5041
28106	5052
28106	5216
28106	5259
29120	5001
29120	5041
29120	5049
29555	5022
25403	5022

voraussetzen	
Vorgänger	**Nachfolger**
5001	5041
5001	5043
5001	5049
5041	5216
5043	5052
5041	5052
5052	5259

Assistent			
PersINr	**Name**	**Fachgebiet**	**Boss**
3002	Platon	Ideenlehre	2125
3003	Aristoteles	Syllogistik	2125
3004	Wittgenstein	Sprachtheorie	2126
3005	Rhetikus	Planetenbewegung	2127
3006	Newton	Keplersche Gesetze	2127
3007	Spinoza	Gott und Natur	2126

prüft			
MatrNr	**VorlNr**	**PersNr**	**Note**
28106	5001	2126	1,0
25403	5041	2125	2,0

Beispiel – Datenbankschema

- Relationenschemata (ohne Wertebereiche):
  - Student(MatrNr, Name, Semester)
  - Professor(PersNr, Name, Rang, Raum)
  - Vorlesung(VorlNr, Titel, SWS, gelesenVon)
  - Assistent(PersNr, Name, Fachgebiet, Boss)

  - Voraussetzen(Vorgänger, Nachfolger)
  - Hört(MatrNr, VorlNr)
  - Prüft(MatrNr, VorlNr, PersNr, Note)

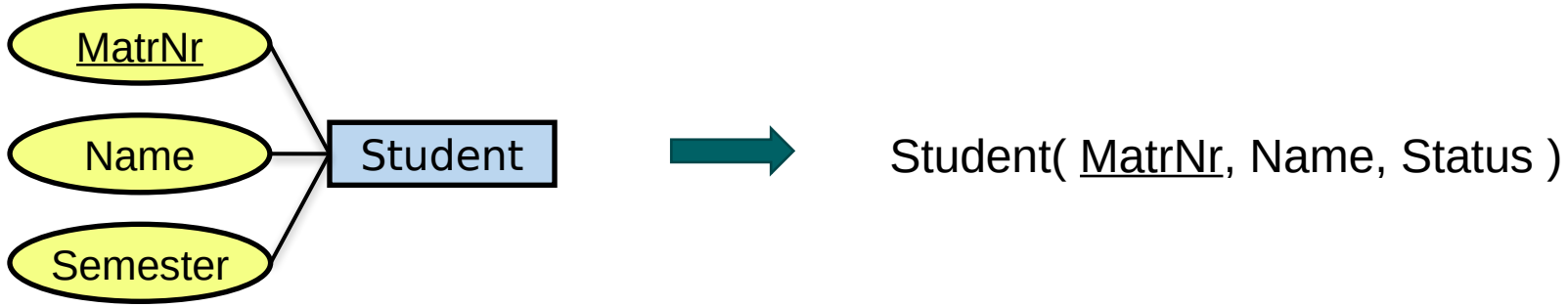
Beispiel – Datenbankschema

- Interrelationale Abhängigkeiten:

Entitytyp mit Attributen

- Jeder Entity-Typ wird zu einer Tabelle/Relation mit entsprechenden Attributen

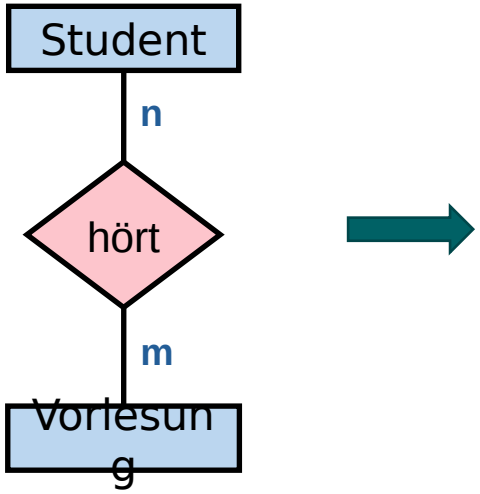
- Spezialfälle
  - Zusammengesetzte Attribute
    - Jedes (Teil-) Attribut als einzelnes Attribut
    - Oder das zusammengesetzte Attribute als Ganzes
  - Mehrwertige Attribute
    - Neue Relation mit zusammengesetztem Schlüssel (bestehend aus eigenem Schlüssel und Fremdschlüssel auf den Entity-Typ)
    - Ähnlich zu einer 1:n – Beziehung



n:m – Beziehungen

- werden durch eigene Relationen dargestellt

  - Schlüssel: besteht aus zwei Fremdschlüsseln auf die Relationen und



Entitäten:
Student(MatrNr)
Vorlesung(VorlNr)
Hört(MatrNr, VorlNr)

Interrelationale Abhängigkeiten:

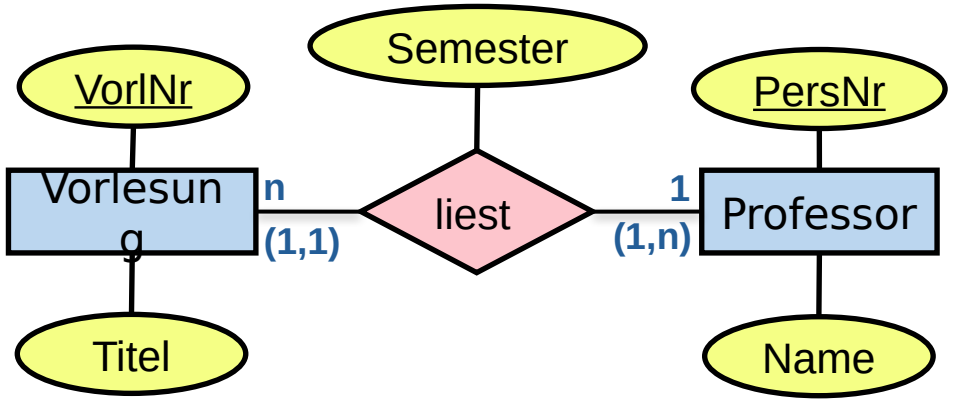
1:n – Beziehungen

- 1:n – Spezialfall von n:m
  - Schlüssel der Relation ist ein Fremdschlüssel auf (ist dann durch bestimmt)

- Praktikabel falls nur wenige Objekte aus an teilnehmen, da die hinzugefügte Relation dann klein ist.

1:n – Beziehungen

- Effizienter: die Beziehung wird in die Relation des -Entity-Typs integriert
  - bekommt Schlüssel von als Fremdschlüssel



Professor(PersNr, Name)

Vorlesung(VorlNr, Titel, Semester,
ProfessorPersNrLiest)

Vorlesung[ProfessorPersNrLiest]
Professor[PersNr]

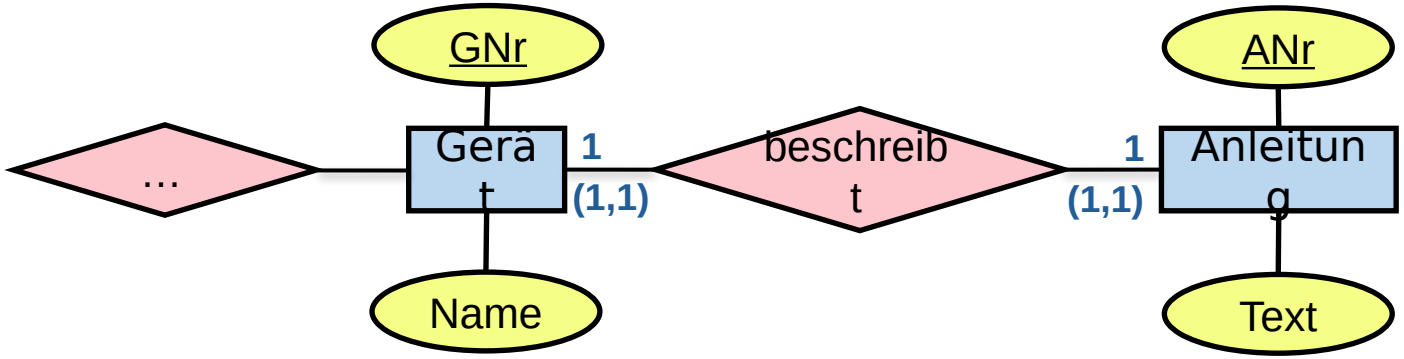
  - Vorteil: Eine zusätzliche Relation ist nicht notwendig
  - Nachteil: falls nur wenige Objekte aus an teilnehmen enthält viele null Werte
    - im Beispiel: falls Vorlesungen meistens nicht von einem Professor gehalten werden
      (andere (min,max)-Notation nötig)

1:1 – Beziehungen

- Mögliche Lösungen:
  - wie 1:n – Beziehung, Fremdschlüssel in verweist auf oder eigene Relation

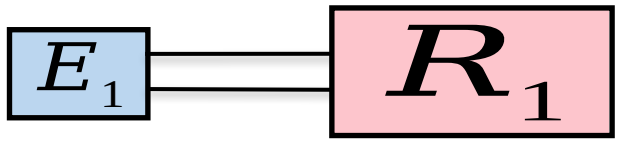
  - Effizienter: Verschmelzen der beteiligten Relationen zu einer einzigen
    - nur falls Teilnahme an total ist und und/oder nicht an anderen Beziehungen teilnehmen

- Beispiel:



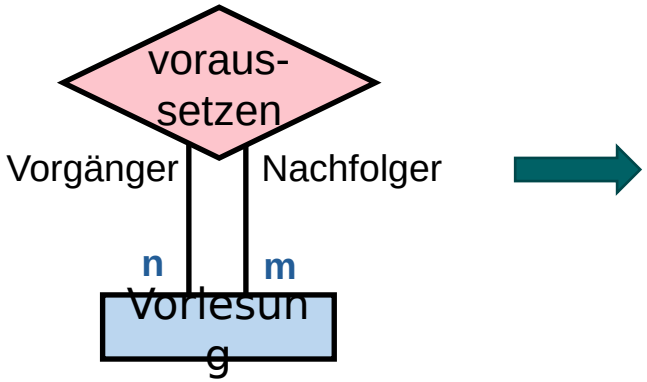
➡ Gerät(GNr, Name, ANr, Text)

Rekursive Beziehungen



- Möglichkeit für 1:1 und 1:n – Beziehung:
  - hat Fremdschlüssel auf sich selbst
  - Attribute von werden zu hinzugefügt

- m:n – Beziehung:
  - eigene Relation für

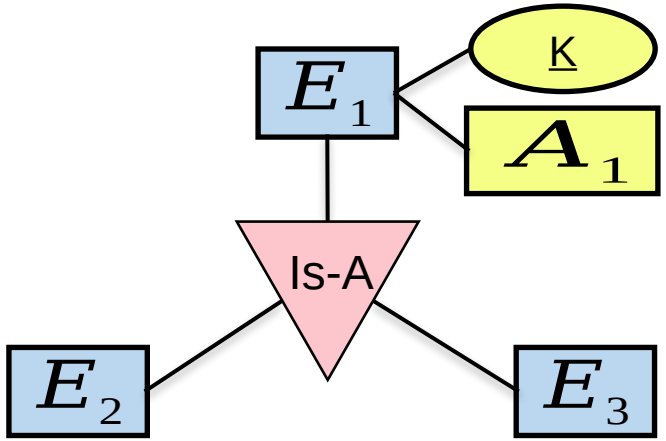


Voraussetzen(Vorgänger, Nachfolger)

Voraussetzen[Vorgänger] ⊆ Vorlesung[VorlNr]

Voraussetzen[Nachfolger] ⊆ Vorlesung[VorlNr]

Generalisierung / Spezialisierung



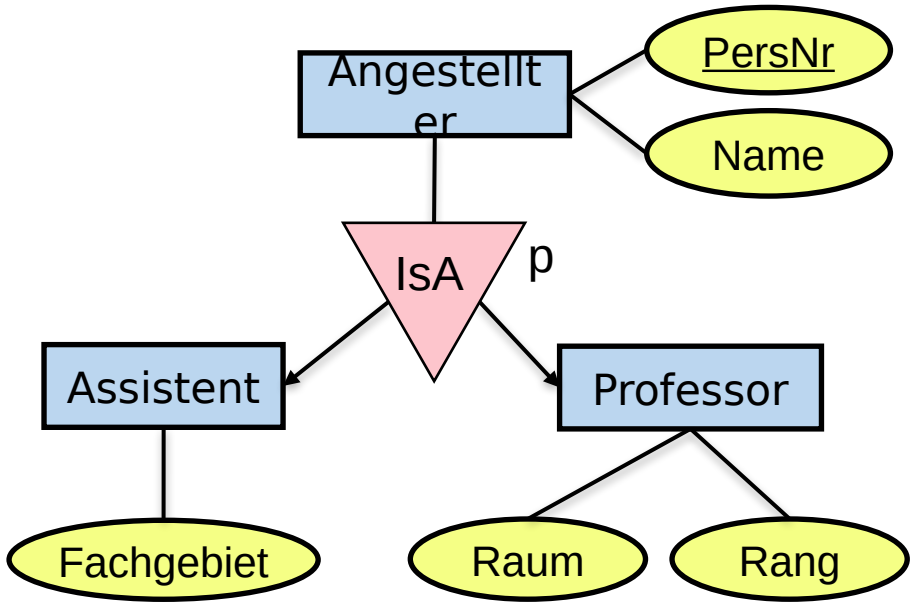
- jede Spezialisierung besitzt einen Fremdschlüssel auf die Generalisierung
  - ist die Beziehung disjunkt: Exklusionsbeziehungen
  - ist die Beziehung zusätzlich total: keine eigene Relation für

- Nachteil (außer bei totalen, disjunkten Beziehungen):
  - Volle Informationen zu und können nur durch join mit abgerufen werden
  - Views/Sichten relationaler DBMS verschaffen Abhilfe

Möglichkeiten zur Übersetzung von isA-Beziehungen

- Erzeuge Tabellen für **jeden** Entity-Typ
  - Für partielle isA-Beziehungen
- Erzeuge **nur** Tabellen **für Subtypen**
  - Bei totalen isA-Beziehungen
- Erzeuge **eine einzige Tabelle**, die alle Attribute aller Entity-Typen enthält, sowie ein **Typ-Attribut**
  - Für disjunkte isA-Beziehungen
- Erzeuge **eine einzige Tabelle**, die alle Attribute aller Entity-Typen enthält, sowie ein **boole'sches Typ-Attribut** für jeden der Entity-Typen
  - falls die Subtypen überlappen (nicht disjunkt)

Generalisierung / Spezialisierung (Partielle Beziehung)

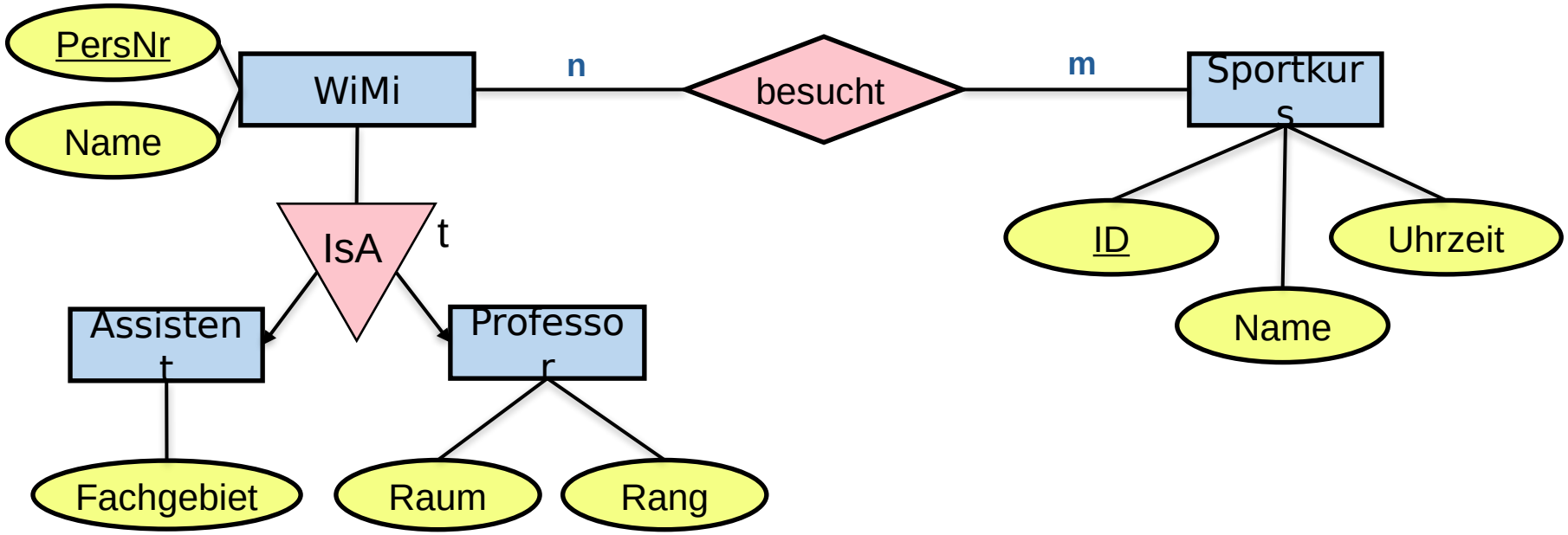


Angestellter(PersNr, Name)
Professor(PersNr, Rang, Raum)
Assistent(PersNr, Fachgebiet)

Professor[PersNr] ⊆ Angestellter[PersNr]
Assistent[PersNr] ⊆ Angestellter[PersNr]

Professor[PersNr] Assistent[PersNr]

Generalisierung / Spezialisierung (totale Beziehung)



Professor(PersNr, Name, Raum, Rang)
Assistent(PersNr, Name, Fachgebiet)
Sportkurs(ID, Name, Uhrzeit)
Besucht(PersNr, SportkursID)

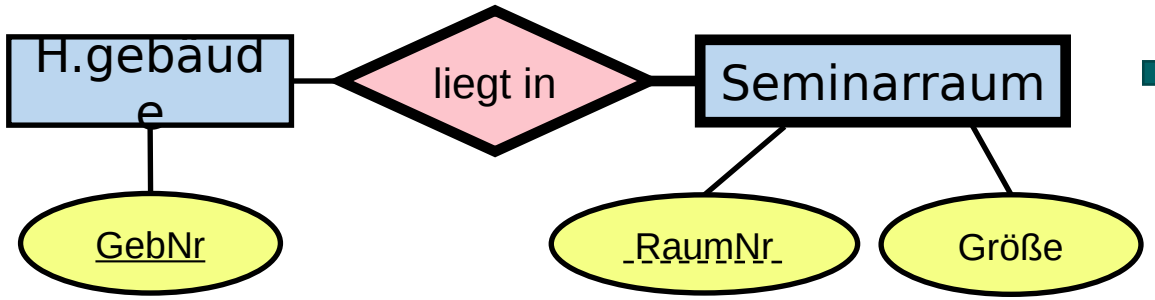
Professor[PersNr] Assistent[PersNr] =

Besucht[SportkursID] Sportkurs[ID]
Besucht[PersNr] Assistent[PersNr] Professor[PersNr]

Schwache Entity-Typen

- Schwacher Entity-Typ mit
  - Teilschlüssel
  - Übergeordneter Entity-Typ mit Schlüssel

- Transformation
  - Relationenschema für ($\underline{K_1}$, $\underline{K_2}$,)
  - Schlüssel wird gebildet aus und
  - ist ein Fremdschlüssel auf



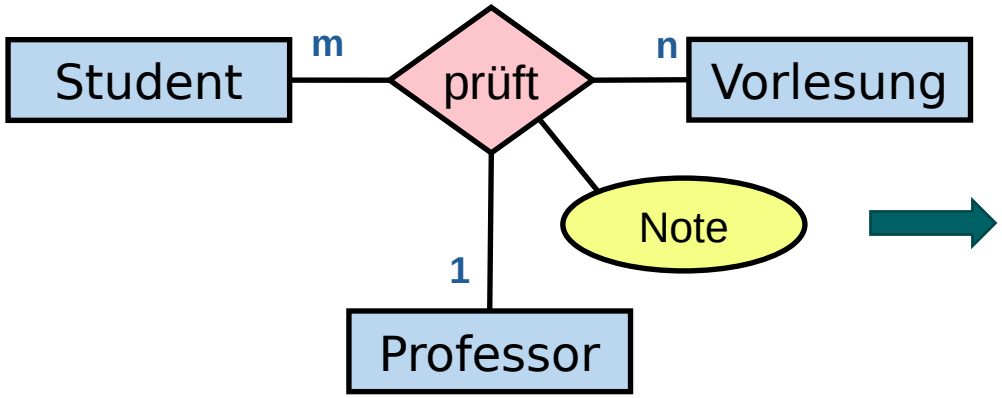
Seminarraum(RaumNr, GebNr, Größe)

Seminarraum[GebNr] H.gebäude[GebNr]

n-stellige Beziehungen

- können wie üblich durch eine eigene Relation dargestellt werden
  - Kardinalitäten geben an, wie sich der Schlüssel zusammensetzt

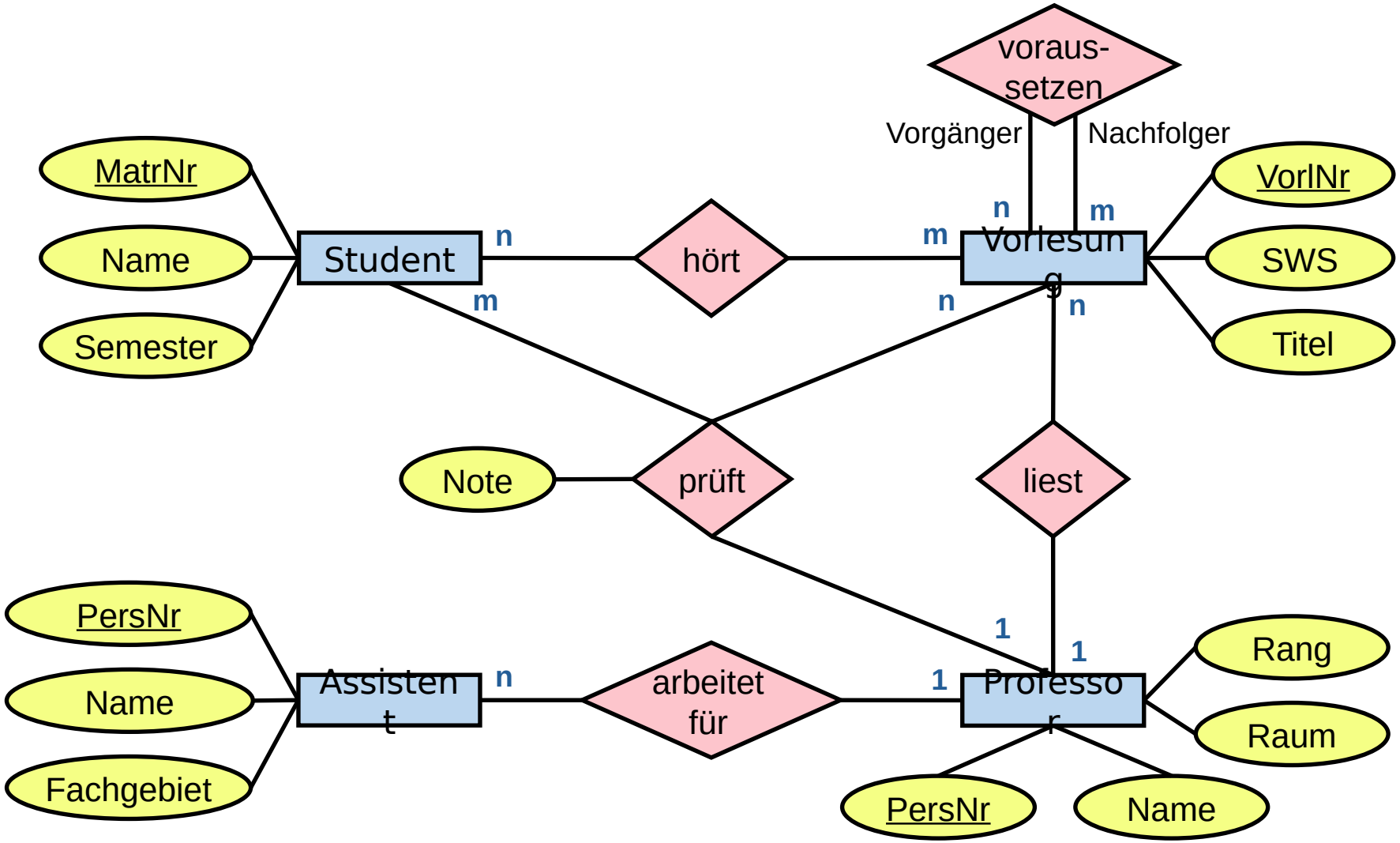
- Beispiel:



Prüft(<u>MatrNr</u>, <u>VorlNr</u>,
PersNr, Note)

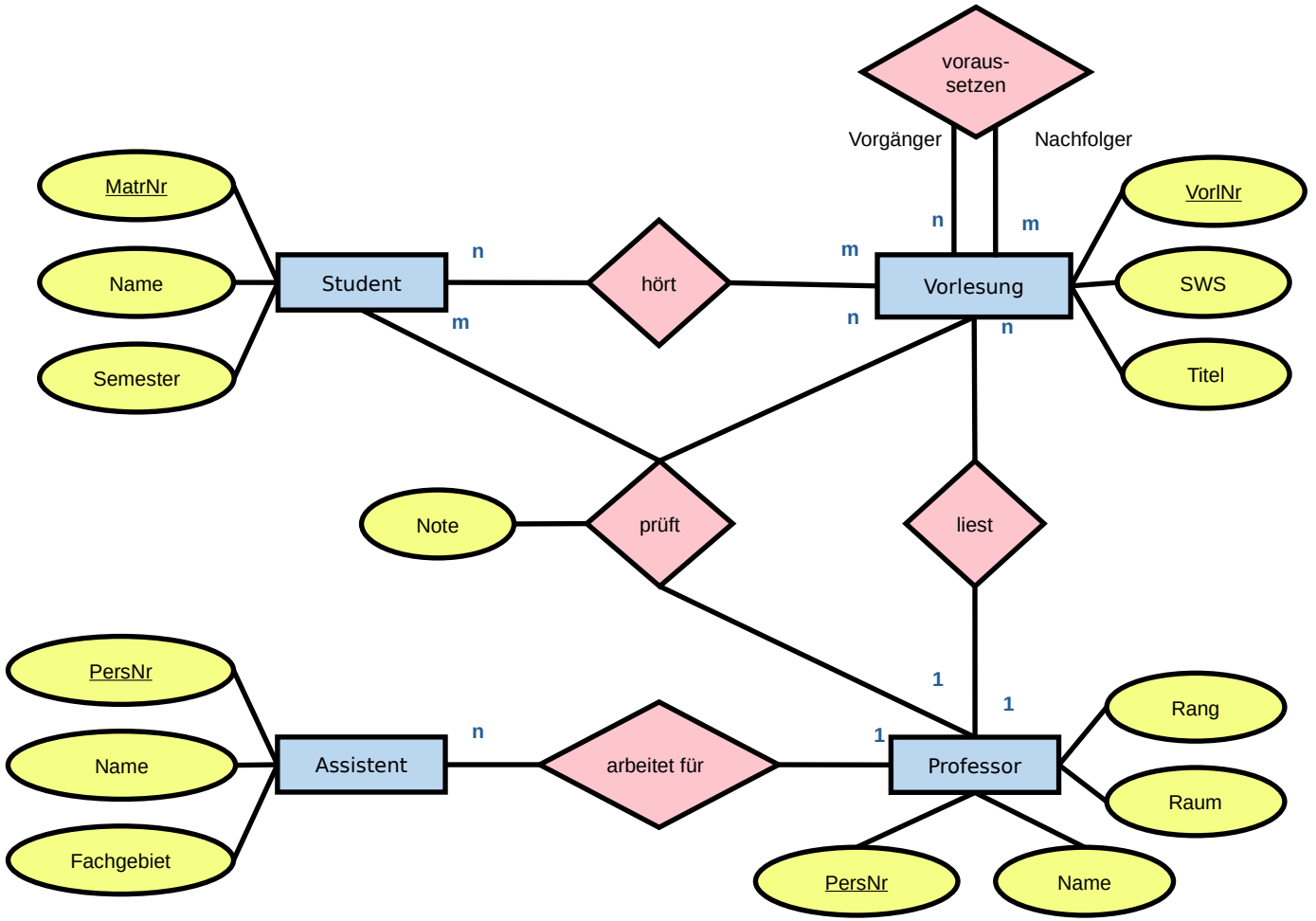
Prüft[MatrNr] ⊆ Student[MatrNr]
Prüft[VorlNr] ⊆ Vorlesung[VorlNr]
Prüft[PersNr] ⊆ Professor[PersNr]

Beispiel : ER-Diagramm (Universität)



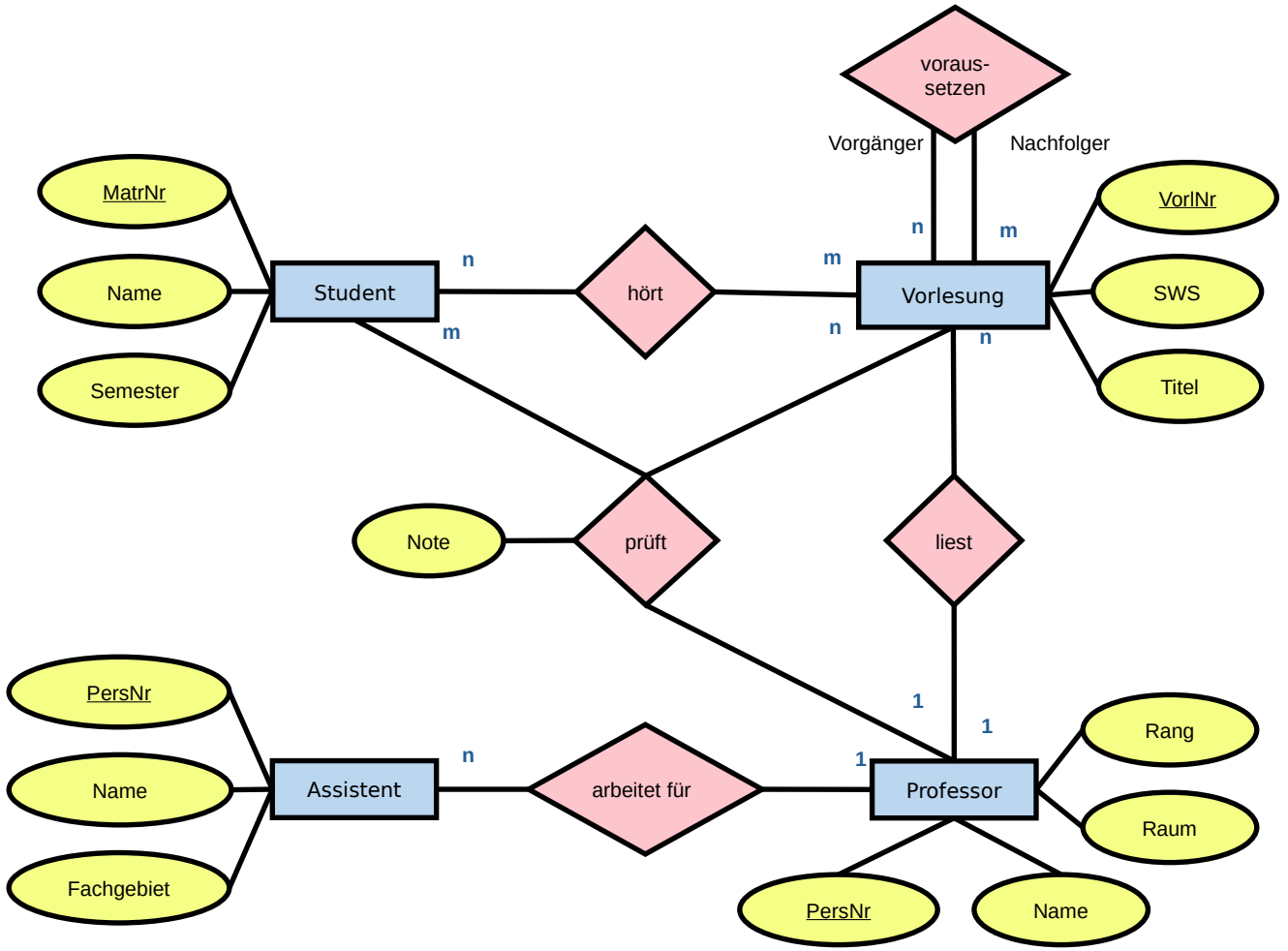
Beispiel für ein Datenbankschema

- **Relationsschema R:**
  - Student(MatrNr, Name, Semester)
  - Professor(PersNr, Name, Rang, Raum)
  - Vorlesung(VorlNr, Titel, SWS, gelesenVon)
  - Assistent(PersNr, Name, Fachgebiet, Boss)
  - Voraussetzen(Vorgänger, Nachfolger)
  - Hört(MatrNr, VorlNr)
  - Prüft(MatrNr, VorlNr, PersNr, Note)



Beispiel für ein Datenbankschema

- **Interrelationale Abhängigkeiten:**
  - Vorlesung[gelesenVon] ⊆ Professor[PersNr]
  - Assistent[Boss] ⊆ Professor[PersNr]
  - Voraussetzen[Vorgänger] ⊆ Vorlesung[VorlNr]
  - Voraussetzen[Nachfolger] ⊆ Vorlesung[VorlNr]
  - Hört[MatrNr] ⊆ Student[MatrNr]
  - Hört[VorlNr] ⊆ Vorlesung[VorlNr]
  - Prüft[MatrNr] ⊆ Student[MatrNr]
  - Prüft[VorlNr] ⊆ Vorlesung[VorlNr]
  - Prüft[PersNr] ⊆ Professor[PersNr]
  - Assistent[PersNr] ∩ Professor[PersNr] = ∅



3. Das Relationale Datenmodell

1. Das Datenmodell
2. Transformation von ER-Diagrammen in das Relationale Modell
3. **Relationale Algebra**
4. Relationaler Kalkül

Arbeiten mit Relationen

- Es gibt verschiedene formale Modelle, um mit relationalen Datenbanken zu arbeiten und Anfragen zu formulieren. Die wichtigsten Beispiele sind die *relationale Algebra* und der *relationale Kalkül*. Sie dienen als Grundlage für konkrete Anfragesprachen.

- **Relationale Algebra**
  - Algebra: Menge mit Operationen (hier: Relationen und relationale Operatoren).
  - Anfragen werden als Operationen auf den Relationen einer Datenbank ausgedrückt.
  - Die Antwortmenge wird durch die tatsächliche Anwendung der Operationen berechnet.
  - *konstruktives* (**prozedurales**) Vorgehen: Es wird angegeben, **WIE** eine Anfrage zu bearbeiten ist

- **Relationenkalkül**
  - Kalkül: Logischer Formalismus zum Ableiten von Ergebnissen.
  - Anfragen werden durch Formeln der Prädikatenlogik erster Stufe ausgedrückt.
  - Die Antwortmenge enthält genau diejenigen Tupel, welche die Formel erfüllen.
  - *deskriptives* (**deklaratives**) Vorgehen: Vorgabe des **WAS**, aber nicht des WIE einer Anfrage

Relationale Sprachen

- Die Relationale Algebra gilt als Maß für die Ausdruckskraft einer Anfragesprache.
  - Eine Sprache *L* heißt *relational vollständig* genau dann, wenn jeder Ausdruck der Relationenalgebra in *L* simuliert werden kann, d.h. es gibt Ausdrücke in *L*, mit denen dasselbe Ergebnis erreicht werden kann.
  - Eine Sprache *L* heißt *streng relational vollständig* genau dann, wenn jeder Ausdruck der Relationenalgebra durch einen einzigen Ausdruck von *L* simuliert werden kann.

- Für den Relationenkalkül unterscheiden wir zwei Arten: Tupel- und Bereichskalkül. Beide sind streng relational vollständig. Insbesondere sind Ausdrücke der Relationenalgebra, sichere Ausdrücke des Tupelkalküls und sichere Ausdrücke des Bereichskalküls äquivalent in ihrer Mächtigkeit.
  - Ein Ausdruck des Tupelkalküls bzw. des Bereichskalküls heißt sicher, wenn sich aus der syntaktischen Struktur erkennen lässt, dass bei der Auswertung einer Anfrage nur endlich viele Tupelkandidaten überprüft werden müssen.

Relationale Algebra

- Eine Algebra ist allgemein gegeben durch
  - Menge von Objekten (Wertebereich)
  - Operationen zur Verknüpfung von Objekten

- Beispiele
  - Natürliche Zahlen mit den Operationen Addition, Multiplikation, …
  - Zeichenketten mit der Operation Konkatenation (d.h. Hintereinanderschreibung)
  - Boolesche Algebra: Wahrheitswerte {wahr, falsch}, Operationen UND, ODER, NICHT, …
  - Mengenalgebra, Operationen Durchschnitt, Vereinigung, Differenz, …

- Relationale Algebra
  - *Relationen* als Wertebereich
  - *relationale Operationen*, die Relationen als Argumente haben und wiederum Relationen als Resultate liefern

- Anfragen
  - Formulierung von Anfragen als Ausdrücke der relationalen Algebra, d.h. durch (rekursive) Anwendung von Operationen auf Relationen. Die Bearbeitung der Anfragen ist durch tatsächliche Auswertung der Operationen auf den Relationen einer Datenbank möglich.

Sechs Grundoperationen der Relationalen Algebra

- Für die relationale Algebra gibt es sechs Grundoperationen, aus denen sich alle anderen Operationen nachbilden lassen: Vereinigung, Differenz, Kartesisches Produkt, Selektion, Projektion, Umbenenung

- *Vereinigung*
  - Die beiden Relationen und müssen das gleiche Schema besitzen. Dann ist definiert als die mengentheoretische Vereinigung:

    - Anschaulich: Packe alle Tupel zusammen in eine Relation (lösche dabei Duplikate)
    - Beispiel:

- *Differenz*
  - Die beiden Relationen und müssen das gleiche Schema besitzen. Dann ist definiert als die mengentheoretische Differenz:

    - Anschaulich: Nimm alle Tupel aus heraus, die es auch in gibt.
    - Beispiel:

Sechs Grundoperationen – Kartesisches Produkt

- *Kartesisches Produkt* (Kreuzprodukt)
  - Seien und der Grad von bzw. . Dann ist die Menge aller -Tupel, deren erste Komponenten ein Tupel in und deren letzte Komponenten ein Tupel in bilden, d.h.
  - Für die Anzahl der Tupel gilt:
  - Anschaulich: verknüpfe jedes Tupel aus mit jedem Tupel aus
  - Beispiel:

Sechs Grundoperationen – Selektion

- *Selektion*
  - Mit der Selektion werden diejenigen Tupel aus der Relation ausgewählt, die eine durch die logische Formel ausgedrückte Eigenschaft erfüllen.
  - Beispiel (Universität):

MatrNr	Name	Semester
24002	Xenokrates	18
25403	Jonas	12

  - Die Formel besteht aus
    - **Operanden:** Attributsnamen (Semester) oder Konstanten (11)
    - **Vergleichsoperatoren:**
    - **Boolesche Operatoren:** (and, or, not)

Sechs Grundoperationen – Selektion

MatrNr	Name	Semester
24002	Xenokrates	18
25403	Jonas	12

- Zur Bestimmung von wird die Formel für jedes Tupel betrachtet. Jedes Attribut in wird durch den Wert ersetzt.
  - eine naive Auswertungsstrategie überprüft jedes Tupel; die Verwendung von Indexstrukturen ermöglicht hier in manchen Fällen Einsparungen.

Sechs Grundoperationen – Projektion

- *Projektion*
  - Die Projektion erlaubt es Spalten (Attribute) aus einer Relation auszuwählen.
  - Sei der Grad von und eine Auswahl von paarweise verschiedenen Attributen (oder entsprechenden Indizes) aus . Dann gilt:

  - Beispiel:

Rang
W3
W2

  - Die Anzahl der Tupel kann sich durch implizite Elimination von Duplikaten verringern, solange in der Projektion keine (vollständigen) Schlüssel enthalten sind.

Sechs Grundoperationen – Umbenennung

- *Umbenennung oder*
  - Die Umbenennung kann sowohl Relationen (nach als auch Attribute einer Relation (nach in der Relation) umbenennen
  - Beispiel: Wir wollen die Voraussetzungen 2. Stufe (also die Voraussetzungen der Voraussetzungen) der Vorlesung 5216 bestimmen

  - Zwischenergebnis:

V1		V2	
Vorgänger	Nachfolger	Vorgänger	Nachfolger
5001	5041	5001	5041
…	…	…	…
5001	**5041**	**5041**	**5216**
…	…	…	…
5052	5259	5052	5259

  - Ergebnis: 5001 ist Vorgänger von 5041, diese ist wiederum Vorgänger von 5216

Beispiele - Grundoperationen

- Zwei Relationen und ihre Verknüpfung mit relationalen Operationen

R	A	B	C
	a	b	c
	d	a	f
	c	b	d

S	A	B	C
	b	g	a
	d	a	f

$R \cup S$	A	B	C
	a	b	c
	d	a	f
	c	b	d
	b	g	a

$R - S$	A	B	C
	a	b	c
	c	b	d

$R \times S$	R.A	R.B	R.C	S.A	S.B	S.C
	a	b	c	b	g	a
	a	b	c	d	a	f
	d	a	f	b	g	a
	d	a	f	d	a	f
	c	b	d	b	g	a
	c	b	d	d	a	f

$\pi_{A, C}(R)$	A	C
	a	c
	d	f
	c	d

$\sigma_{B = b}(R)$	A	B	C
	a	b	c
	c	b	d

- Mit den vorgestellten Grundoperationen lassen sich nun alle Operationen der relationalen Algebra (induktiv) definieren.

Die Operationen der relationalen Algebra

- Induktive Definition der Operationen:
  - zunächst die möglichen Basisausdrücke
  - dann Zusammensetzen eines Relationenalgebra-Ausdruck mithilfe der Grundoperationen

  - ein Basisausdruck ist:
    - eine konstante Relation
    - eine Relation der Datenbank

  - seien und Relationenalgebra-Ausdrücke. Dann sind folgendes auch gültige Ausdrücke:
    - , wobei die Schemata und der Relationen übereinstimmen müssen – also
    - , wobei hier auch gelten muss

Die Operationen der relationalen Algebra (2)

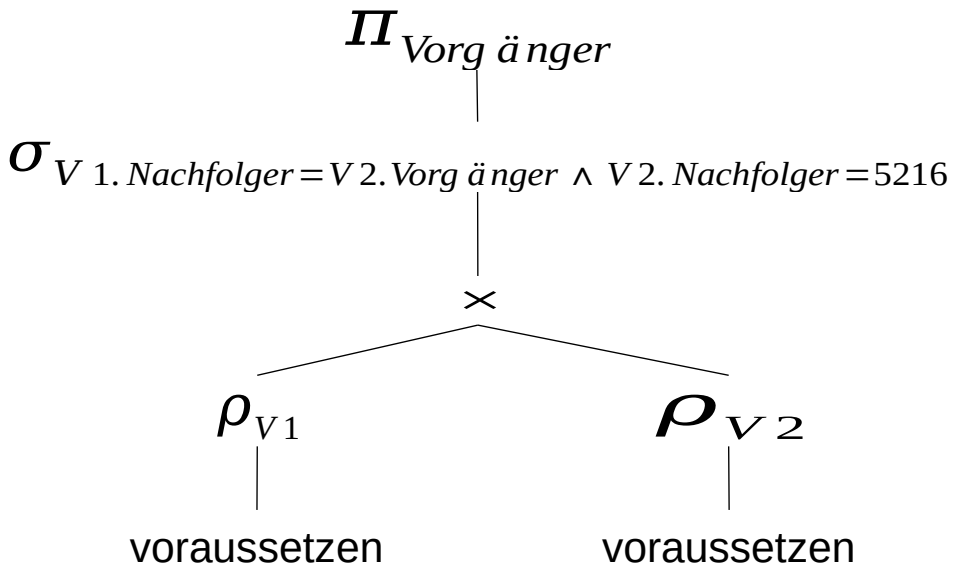
- seien und Relationenalgebra-Ausdrücke. Dann sind folgendes auch gültige Ausdrücke:
  - , mit einer Formel über den Attributen der Relation
  - , mit einer Liste von Indizes/Attributen die in vorkommen
  - und wobei ein Attributname der Relation ist und nicht schon als Attributname in vorkommt

- Mit der vorgestellten Algebra lassen sich beliebig komplexe Anfragen (Queries) formulieren, zum Beispiel die im Weiteren vorgestellten zusätzlichen Operationen

- Ausdrücke der relationalen Algebra lassen sich auch als Operatorbäume darstellen

Operatorbäume

- Jeder Ausdruck einer Algebra lässt sich auch als *Operatorbaum* darstellen.
  - Die Blätter enthalten *Relationen*.
  - Die inneren Knoten repräsentieren die *Operationen*.



Weitere Operationen – Durchschnitt

- Neben den sechs Grundoperationen existiert eine Reihe anderer nützlicher relationaler Operationen.
- *Durchschnitt*
  - Die beiden Relationen und müssen das gleiche Schema besitzen. Dann ist definiert als die Schnittmenge der beiden Relationen, d.h.
  - Beispiel: Finde die PersNr der W3-Professoren die mindestens eine Vorlesung halten.
    - Ergebnis

Vorlesung			
VorlNr	**Titel**	**SWS**	**gelesenVon**
5001	Grundzüge	4	2137
5041	Ethik	4	2125
5043	Erkenntnistheorie	3	2126
5049	Mäeutik	2	2125
4052	Logik	4	2125
5052	Wissenschaftstheorie	3	2126
5216	Bioethik	2	2126
5259	Der Wiener Kreis	2	2133
5022	Glaube und Wissen	2	2134
4630	Die 3 Kritiken	4	2137

Professor			
PersNr	**Name**	**Rang**	**Raum**
2125	Sokrates	W3	226
2126	Russel	W3	232
2127	Kopernikus	W2	310
2133	Popper	W2	52
2134	Augustinus	W2	309
2136	Curie	W3	36
2137	Kant	W3	7

PersNr

Weitere Operationen – Durchschnitt

- Neben den sechs Grundoperationen existiert eine Reihe anderer nützlicher relationaler Operationen.
- *Durchschnitt*
  - Die beiden Relationen und müssen das gleiche Schema besitzen. Dann ist definiert als die Schnittmenge der beiden Relationen, d.h.

  - Beispiel: Finde die PersNr der W3-Professoren die mindestens eine Vorlesung halten.

    - Ergebnis

PersNr
2125
2126
2137

Vorlesung

VorlNr	Titel	SWS	gelesenVon
5001	Grundzüge	4	2137
5041	Ethik	4	2125
5043	Erkenntnistheorie	3	2126
5049	Mäeutik	2	2125
4052	Logik	4	2125
5052	Wissenschaftstheorie	3	2126
5216	Bioethik	2	2126
5259	Der Wiener Kreis	2	2133
5022	Glaube und Wissen	2	2134
4630	Die 3 Kritiken	4	2137

Professor

PersNr	Name	Rang	Raum
2125	Sokrates	W3	226
2126	Russel	W3	232
2127	Kopernikus	W2	310
2133	Popper	W2	52
2134	Augustinus	W2	309
2136	Curie	W3	36
2137	Kant	W3	7

Beispiel für eine Datenbank

Professor			
PersNr	**Name**	**Rang**	**Raum**
2125	Sokrates	W3	226
2126	Russel	W3	232
2127	Kopernikus	W2	310
2133	Popper	W2	52
2134	Augustinus	W2	309
2136	Curie	W3	36
2137	Kant	W3	7

Student		
MatrNr	**Name**	**Semester**
24002	Xenokrates	18
25403	Jonas	12
26120	Fichte	10
26830	Aristoxenos	8
27550	Schopenhauer	6
28106	Carnap	3
29120	Theophrastos	2
29555	Feuerbach	2

Vorlesung			
VorlNr	**Titel**	**SWS**	**gelesenVon**
5001	Grundzüge	4	2137
5041	Ethik	4	2125
5043	Erkenntnistheorie	3	2126
5049	Mäeutik	2	2125
4052	Logik	4	2125
5052	Wissenschaftstheorie	3	2126
5216	Bioethik	2	2126
5259	Der Wiener Kreis	2	2133
5022	Glaube und Wissen	2	2134
4630	Die 3 Kritiken	4	2137

hört	
MatrNr	**VorlNr**
26120	5001
27550	5001
27550	4052
28106	5041
28106	5052
28106	5216
28106	5259
29120	5001
29120	5041
29120	5049
29555	5022
25403	5022

voraussetzen	
Vorgänger	**Nachfolger**
5001	5041
5001	5043
5001	5049
5041	5216
5043	5052
5041	5052
5052	5259

Assistent			
PersINr	**Name**	**Fachgebiet**	**Boss**
3002	Platon	Ideenlehre	2125
3003	Aristoteles	Syllogistik	2125
3004	Wittgenstein	Sprachtheorie	2126
3005	Rhetikus	Planetenbewegung	2127
3006	Newton	Keplersche Gesetze	2127
3007	Spinoza	Gott und Natur	2126

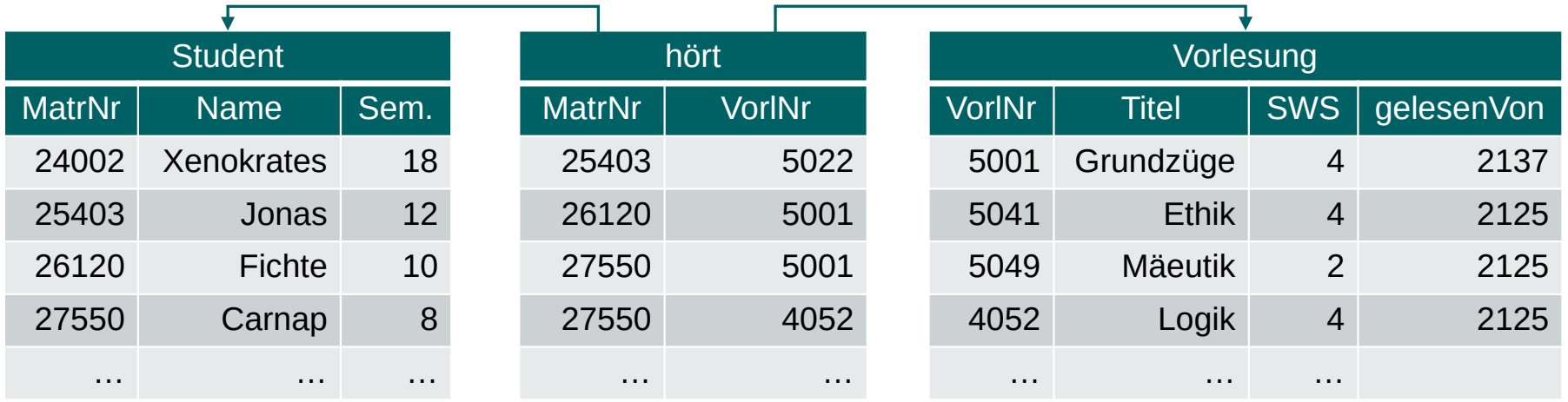
prüft			
MatrNr	**VorlNr**	**PersNr**	**Note**
28106	5001	2126	1,0
25403	5041	2125	2,0

Weitere Operationen – Natürlicher Verbund (natural join)

- Natürlicher Verbund
  - Idee: Selektiere aus dem Kreuzprodukt nur zueinander „passende“ Tupel. Es sind nicht immer alle Tupel eines Kreuzprodukts relevant.
  - Sei , d.h. und haben gemeinsame Attribute. Wir schreiben die Attribute von als und die Attribute von als . Dann ist
  - Beispiel: Zugriff auf die Informationen einer m:n-Beziehung, wie zum Beispiel

Weitere Operationen – Natürlicher Verbund (natural join) (2)

- Beispiel:



Student		
MatrNr	Name	Sem.
24002	Xenokrates	18
25403	Jonas	12
26120	Fichte	10
27550	Carnap	8
…	…	…

hört	
MatrNr	VorlNr
25403	5022
26120	5001
27550	5001
27550	4052
…	…

Vorlesung			
VorlNr	Titel	SWS	gelesenVon
5001	Grundzüge	4	2137
5041	Ethik	4	2125
5049	Mäeutik	2	2125
4052	Logik	4	2125
…	…	…	

MatrNr	Name	Semester	VorlNr	Titel	SWS	gelesenVon
25403	Jonas	12	5022	Glaube und Wissen	2	2134
26120	Fichte	10	5001	Grundzüge	4	2137
27550	Carnap	8	5001	Grundzüge	4	2137
…	…	…	…	…	…	…

Weitere Operationen – Natürlicher Verbund (natural join) (3)

- Der natürliche Join ist sowohl **assoziativ**, d.h.

  als auch **kommutativ** bis auf Vertauschung der Attributreihenfolge, zum Beispiel ist

- Da der natürliche Join nur auf Attributen mit gleichen Attributnamen arbeitet, kann es notwendig sein Attribute umzubenennen.
  - Wollen wir die Relationen *Vorlesung* und *Professor* miteinander verbinden, müssen wir eines der Attribute *Vorlesung.gelesenVon* oder *Professor.PersNr* umbenennen.

- Wird der natürliche Join auf zwei Relationen ohne gleiche Attributnamen angewandt, dann entspricht des Ergebnis dem Ergebnis des kartesischen Produktes der beiden Relationen.

Weitere Operationen – Theta-Join

- Der Theta-Join :
  - Idee: Selektiere aus dem Kreuzprodukt nur zueinander „passende“ Tupel. „Passend“ wird hierbei durch die Formel festgelegt.
  - Seien und und eine Formel über . Dann ist
  - Doppelte Attribute werden nicht aussortiert!
  - Beispiel:

Student. MatrNr	Name	Semester	hört. MatrNr	VorlNr
25403	Jonas	12	25403	5022
26120	Fichte	10	26120	5001
…	…	…		…

Weitere Operationen – Äußere Join-Operatoren

- Im Gegensatz zum natürlichen Join bleiben bei äußeren Join-Operatoren auch Tupel erhalten, die keinen “Joinpartner” gefunden haben.
  - left outer join ⟕ : Die Tupel der linken Relation bleiben erhalten.
  - right outer join ⟖ : Die Tupel der rechten Relation bleiben erhalten.
  - (full) outer join ⟗ : Die Tupel beider Relationen bleiben erhalten.

- Darüber hinaus gibt es die Semi-Join-Operatoren und . Der linke Semi-Join von mit – in Zeichen – ist definiert als

wobei die Menge der Attribute von ist.

Der rechte Semi-Join analog dazu (mit dann Menge der Attribute von)

Join-Operatoren – Beispiele

natural join

left outer join

full outer join

left semi-join with

3. Das Relationale Datenmodell

1. Das Datenmodell
2. Transformation von ER-Diagrammen in das Relationale Modell
3. Relationale Algebra
4. **Relationaler Kalkül**

Relationaler Kalkül

- Abgrenzung zur relationalen Algebra:
  - Die relationale Algebra ist eine *prozedurale* Sprache: Ein Ausdruck gibt an, unter Benutzung welcher Operationen das Ergebnis einer Anfrage berechnet werden soll (**WIE**).
  - Der relatione Kalkül ist eine *deklarative* Sprache: Ein Ausdruck des Relationenkalküls beschreibt, welche Eigenschaften die Tupel der Ergebnisrelation haben müssen (**WAS**).

- Der Relationenkalkül und die relationale Algebra sind allerdings gleich mächtig.
  - Das bedeutet, zu jedem Ausdruck der in der Relationenalgebra formuliert ist, gibt es einen Ausdruck im Relationenkalkül der die gleiche Rückgabe liefert, und andersherum.

Relationaler Kalkül

- Relationenkalkül: Prädikatenlogik erster Stufe.
  - *Syntax*: wie können gültige Ausdrücke des Kalküls gebildet werden. Dabei treten Konstanten, Variablen, Operatoren und Quantoren auf (symbolische Ebene).
  - *Semantik*: beschreibt die Bedeutung von syntaktisch korrekt gebildeten Ausdrücken. Dabei geht es um Relationen und die Erfüllung von Bedingungen (Bedeutungsebene).

- Zwei ähnliche Arten des Relationenkalküls:
  - *Tupelkalkül*: Variablen repräsentieren die Tupel einer Relation
  - *Domänenkalkül*: Variablen repräsentieren mögliche Werte eines Attributs

- Zunächst: Beispiele zu Relationenkalkül-Anfragen und Quantoren.
- Später: Formale Definition von Syntax und Semantik

Der Tupelkalkül – Beispiele

- Form der Ausdrücke im Tupelkalkül:

  Hier ist eine sogenannte *Tupelvariable* und eine *Formel*.

  - die neue Relation besteht aus den Tupeln die die Formel erfüllen
  - die Variable muss in eine *freie Variable* sein (= nicht quantifiziert, dazu gleich mehr)
  - Beispiel: Finde alle Professoren mit Rang ‘W3’.

    - Auswertung:
      - wird an jedes Tupel der Relation Professor gebunden
      - dann wird für jedes Tupel die Bedingung überprüft

Der Tupelkalkül – Quantifizierung

- Existenz- und Allquantoren
  - : es existiert ein in , sodass gilt.
  - für alle aus gilt .
  - hierbei muss in eine freie Variable sein

  - Beispiel: Finde die Studenten, die mindestens eine Vorlesung bei der Professorin Curie hören. ist in dem folgenden Prädikat die einzige freie Tupelvariable!

Der Tupelkalkül – Quantifizierung

- Existenz- und Allquantoren
  - Beispiel: Finde die Studenten, die alle von Professor Sokrates (PersNr 2125) angebotenen Vorlesungen besuchen.

    Auch hier ist die einzige freie Variable des Prädikats.

$$F \rightarrow G := \neg F \vee G$$

Beispiel für eine Datenbank

Professor			
PersNr	**Name**	**Rang**	**Raum**
2125	Sokrates	W3	226
2126	Russel	W3	232
2127	Kopernikus	W2	310
2133	Popper	W2	52
2134	Augustinus	W2	309
2136	Curie	W3	36
2137	Kant	W3	7

Student		
MatrNr	**Name**	**Semester**
24002	Xenokrates	18
25403	Jonas	12
26120	Fichte	10
26830	Aristoxenos	8
27550	Schopenhauer	6
28106	Carnap	3
29120	Theophrastos	2
29555	Feuerbach	2

Vorlesung			
VorlNr	**Titel**	**SWS**	**gelesenVon**
5001	Grundzüge	4	2137
5041	Ethik	4	2125
5043	Erkenntnistheorie	3	2126
5049	Mäeutik	2	2125
4052	Logik	4	2125
5052	Wissenschaftstheorie	3	2126
5216	Bioethik	2	2126
5259	Der Wiener Kreis	2	2133
5022	Glaube und Wissen	2	2134
4630	Die 3 Kritiken	4	2137

hört	
MatrNr	**VorlNr**
26120	5001
27550	5001
27550	4052
28106	5041
28106	5052
28106	5216
28106	5259
29120	5001
29120	5041
29120	5049
29555	5022
25403	5022

voraussetzen	
Vorgänger	**Nachfolger**
5001	5041
5001	5043
5001	5049
5041	5216
5043	5052
5041	5052
5052	5259

Assistent			
PersINr	**Name**	**Fachgebiet**	**Boss**
3002	Platon	Ideenlehre	2125
3003	Aristoteles	Syllogistik	2125
3004	Wittgenstein	Sprachtheorie	2126
3005	Rhetikus	Planetenbewegung	2127
3006	Newton	Keplersche Gesetze	2127
3007	Spinoza	Gott und Natur	2126

prüft			
MatrNr	**VorlNr**	**PersNr**	**Note**
28106	5001	2126	1,0
25403	5041	2125	2,0

Der Tupelkalkül – Konstruktion neuer Relationen

- Schema einer Tupelvariable:
  - Eine Tupelvariable besitzt ein Schema
  - ist das Schema aus dem Zusammenhang klar, so wird auf die Angabe verzichtet
  - über das Schema können neue Tupel konstruiert werden (vgl. Projektion und Join):
    - aller Professoren mit
    - Kurzschreibweise (Tupelkonstruktor, nächste Seite):

Der Tupelkalkül – Konstruktion neuer Relationen

- Konstruktion neuer Tupel:
  - Tupelkonstruktor :
    - die Attribute müssen in den Schemata der Relationen enthalten sein, an die die gebunden werden
  - Beispiel (Projektion und Join):
    - Ordne jedem Professor (Name) den ihm zugeordneten Assistenten (PersNr) zu
    - Kurzschreibweise für:

Der Tupelkalkül – Formale Definition (Syntax)

- Syntax: **Tupelvariable** Formel Ausdruck
  - Idee: Ein Ausdruck liefert als Ergebnis die Relation aller Tupel, die die Formel erfüllen.
  - Konstruktion von gültigen Tupelkalkül-Ausdrücken

  - Tupelvariablen:
    - Eine Tupelvariable besitzt ein Schema

    - Das Schema ist oft implizit festgelegt durch
      - eine Formel der Form , oder
      - die Struktur des Tupelkonstruktors

    - Eine Tupelvariable kann *frei* oder *gebunden* auftreten (siehe Syntax: Formel)

Der Tupelkalkül – Formale Definition (Syntax)

- Syntax: Tupelvariable **Formel** Ausdruck
  - Idee: Ein Ausdruck liefert als Ergebnis die Relation aller Tupel, die die Formel erfüllen.

  - Die Grundbausteine der Formeln sind die Atome
  - Tupelvariablen kommen in Atomen grundsätzlich nur *frei* vor
  - Es gibt drei Arten von Atomen:
    - ist ein Atom, wobei eine Tupelvariable und eine Relation ist
    - ist ein Atom, wobei Tupelvariablen sind, und Attributnamen von bzw. und ein Vergleichsoperator. und müssen bzg. vergleichbar sein
    - , wobei im Unterschied zu oben eine Konstante ist

Der Tupelkalkül – Formale Definition (Syntax)

- Syntax: Tupelvariable **Formel** Ausdruck
  - Idee: Ein Ausdruck liefert als Ergebnis die Relation aller Tupel, die die Formel erfüllen.
  - Der Aufbau von Formeln ist rekursiv definiert:
    - Atome: Jedes Atom ist eine Formel
    - Verknüpfungen: Seien Formeln. Dann sind auch , und Formeln
    - Quantoren: Sei eine Formel in der als *freie* Variable auftritt. Dann sind auch und Formeln. ist in dieser Formel dann eine *gebundene* Variable
  - Beispiel: In der folgenden Formel ist eine freie Variable, eine gebundene.

Der Tupelkalkül – Formale Definition (Syntax)

- Syntax: Tupelvariable Formel **Ausdruck**
  - Idee: Ein Ausdruck liefert als Ergebnis die Relation aller Tupel, die die Formel erfüllen.
  - Ausdruck: Ein Ausdruck des Tupelkalküls hat die Form
    - , ist die einzige freie Tupelvariable in der Formel

Der Tupelkalkül – Formale Definition (Semantik)

- Semantik: Tupelvariable Formel Ausdruck
  - Idee: Interpretation der Tupelkalkül-Ausdrücke
  - Analoge drei Schritte:
    - Tupelvariablen konkrete Tupel
    - Formeln (Atome, Operatoren & Quantoren) true, false
    - Ausdrücke Relationen

Der Tupelkalkül – Formale Definition (Semantik)

- Semantik: **Tupelvariable** Formel Ausdruck
  - Idee: Interpretation der Tupelkalkül-Ausdrücke
  - Belegung von Tupelvariablen: Seien
    - eine Tupelvariable mit Schema ,
    - eine Formel, die (i.A. nicht nur) als freie Tupelvariable enthält
    - ein beliebiges Tupel (muss i.A. nicht zu geg. Relation gehören)

    Bei der Belegung von mit wird jedes freie Vorkommen von in durch ersetzt:
  - Beispiel: Sei mit .

    Für ist

Der Tupelkalkül – Formale Definition (Semantik)

- Semantik: Tupelvariable **Formel** Ausdruck
  - Idee: Interpretation der Tupelkalkül-Ausdrücke
  - Interpretation einer Formel (analog zu syntaktischem Aufbau):
    - Die Formel darf keine freien Variablen mehr enthalten.
    - Atome haben dann nur noch zwei Formen:
      - , falls in enthalten ist; sonst
      - , falls in Beziehung zu steht (Bsp.:)
  - Beispiel:

Der Tupelkalkül – Formale Definition (Semantik)

- Semantik: Tupelvariable **Formel** Ausdruck
  - Idee: Interpretation der Tupelkalkül-Ausdrücke
  - Interpretation einer Formel (analog zu syntaktischem Aufbau):
    - Logische Operatoren:
      - , falls ist und umgekehrt
      - genau dann, wenn
      - , falls mindestens eines von ist
  - Beispiel:

Der Tupelkalkül – Formale Definition (Semantik)

- Semantik: Tupelvariable **Formel** Ausdruck
  - Idee: Interpretation der Tupelkalkül-Ausdrücke
  - Interpretation einer Formel (analog zu syntaktischem Aufbau):
    - Quantoren:
      - Zur Interpretation von und darf nur in frei sein
      - gdw ein existiert, sodass ist
      - gdw für alle gilt:
  - Beispiel:

Der Tupelkalkül – Formale Definition (Semantik)

- Semantik: Tupelvariable Formel **Ausdruck**
  - Idee: Interpretation der Tupelkalkül-Ausdrücke
  - Interpretation eines Ausdrucks:

    Sei oder ein Ausdruck und

    das Schema von bzw. .
    - bzw. dürfen die einzigen freien Variabeln in sein.
    - Der Wert von ist die Menge aller Tupel für die gilt

Der Domänenkalkül

- Im Unterschied zum Tupelkalkül: *Bereichsvariablen*
  - Ein Ausdruck mit Bereichsvariablen hat die Form

- Definitionen:
  - Atome:
    - Bedeutung: ist wahr, falls nach Belegung der mit das Tupel
      in enthalten ist
    - Bedeutung: ist wahr, falls nach Belegung in der Beziehung zu steht
  - Formeln: Analog zum Tupelkalkül, Quantoren beziehen sich auf Domänen
  - Ausdruck: siehe oben

Der Domänenkalkül

- Beispiele:
  - Finde MatNr und Name der Studenten, die mindestens eine Prüfung bei Professor Curie abgelegt haben.
  - Operationen der Relationalen Algebra:

Beispiele Tupelkalkül VS Domänenkalkül

- MatNr und Name der Studenten, die mindestens eine Prüfung bei Professor Curie abgelegt haben.
- Tupelkalkül
- Domänenkalkül

Kurzschreibweise

Beispiele Tupelkalkül VS Domänenkalkül

- aller Professoren mit

- Tupelkalkül

- Domänenkalkül

Beispiele Tupelkalkül VS Domänenkalkül

- Ordne jedem Professor (Name) den ihm zugeordneten Assistenten (PersNr) zu
- Tupelkalkül
- Domänenkalkül

Folie 1