

Grundlagen: Datenbanken

1. Zentralübung

Harald Lang

► Ist der Prüfungstermin schon bekannt?

- Termin: **Mi. 18.02.2015, 08:00 Uhr**

TUM Technische Universität München

- ▣ Hochschulpräsidium
- ▣ Gremien
- ▣ Hochschulreferate
- ▣ Zentrale Serviceeinrichtungen
- ▣ Zentrale Verwaltung
- ▣ Fakultäten
 - ▣ Mathematik
 - ▣ Physik
 - ▣ Chemie
 - ▣ Wirtschaftswissenschaften
 - ▣ Bau Geo Umwelt
 - ▣ Architektur
 - ▣ Maschinenwesen
 - ▣ Elektrotechnik und Informatik
 - ▣ **Informatik**
 - ▣ Wissenschaftszentrum Wei
 - ▣ Medizin
 - ▣ Sport- und Gesundheitswis
 - ▣ TUM School of Education
- ▣ Integrative Research Centers
- ▣ TUM Graduate School
- ▣ Wissenschaftliche Zentralinsti
- ▣ Forschungscluster
- ▣ Beauftragte und Vertretungen
- ▣ Partnerschaftliche Einrichtung

Fakultät für Informatik

- [Homepage](#)
- [Kontakt](#)
- [Lageplan](#)
- [Dekan](#)
- [Prodekan](#)
- [Studiendekan](#)
- [Beauftragte](#)
- [Fakultätsrat](#)
- [Prüfungsausschuss Informatik](#)
- [Beirat](#)
- [Fachschaftsvertretung](#)
- [Dekanat IN](#)
- [Institut für Informatik](#)
- [Zentrale Einrichtungen des Instituts für Informatik](#)
- [Center of Doctoral Studies in Informatics and its Applications CeDoSIA](#)

Forschung & Lehre	Ressourcen
Modulhandbuch	Personen & Zuständigkeiten
Lehrveranstaltungen	
Prüfungstermine	
Studienangebot	

- ▶ **Gilt der Bonus auch für die Nachholklausur?**
 - ▶ Ja. Selbst dann, wenn die Hauptklausur nicht bestanden wurde.
- ▶ **Ich kann nicht zur ersten Klausur antreten. Muss ich zur Hauptklausur angemeldet sein, um an der Nachholklausur teilnehmen zu dürfen?**
 - ▶ Bitte nicht, das spart Papier und schont die Umwelt.
- ▶ **Ist der Termin für die Nachholklausur schon bekannt?**
 - ▶ Noch nicht. Voraussichtlich wird der Termin Ende März/Anfang April sein.

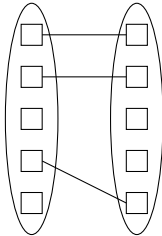
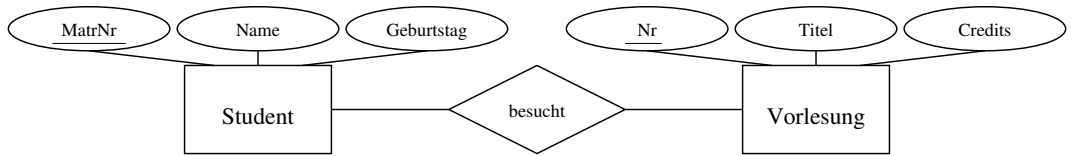
Diese Folien werden online gestellt.

Nutzen Sie die heutige Veranstaltung um Fragen zu stellen. :)

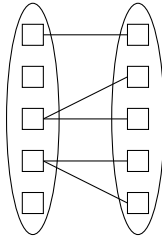
Agenda

- ▶ ER-Modellierung
- ▶ Relationales Modell
- ▶ Anfragesprachen (Algebra & Kalküle)
- ▶ SQL
- ▶ Relationale Entwurfstheorie
- ▶ (Physische Datenorganisation)

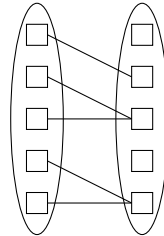
Datenbankentwurf



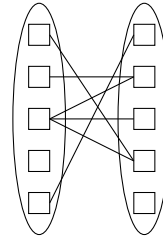
1:1



1:N

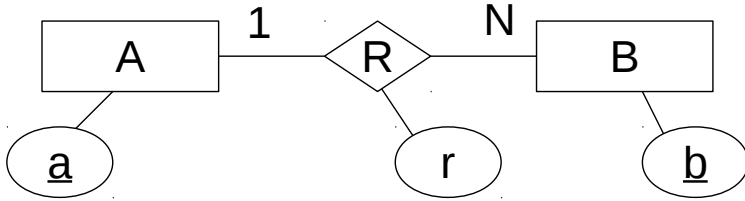


N:1

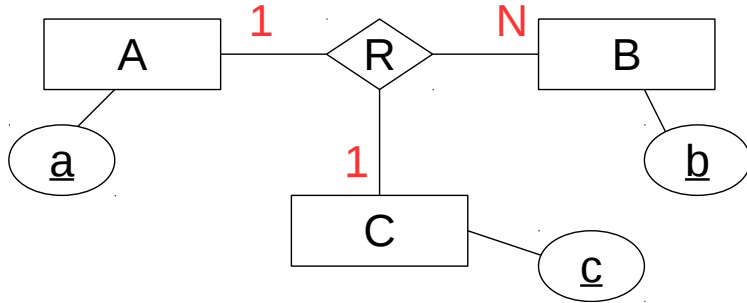


M:N

ER-Modell in Schema überführen und verfeinern



ER-Modellierung | Partielle Funktionen



Partielle Funktionen:

Relationen:

Das Relationale Modell

Definition

- ▶ Eine relationale Datenbank enthält eine Menge von Relationen
- ▶ Eine Relation R besteht aus zwei Bestandteilen:
 - ▶ Einer **Instanz** R : eine Tabelle mit Zeilen und Spalten; der *aktuelle Inhalt* der Relation (auch Ausprägung genannt)
 - ▶ Einem **Schema** \mathcal{R} : spezifiziert den *Namen der Relation* und die *Namen und Datentypen der Spalten*; legt die Struktur der Relation fest

Das Relationale Modell

Beispielausprägung:

<i>Studenten</i>		
MatrNr	Name	Semester
24002	Xenokrates	18
25403	Jonas	10
27550	Schopenhauer	6
...

Schema:

Relationale Algebra

Algebraische Operatoren:

Projektion	Π_{A_1, \dots, A_n}
Selektion	σ_p
Kreuzprodukt	\times
Verbund (Join)	$\bowtie_\theta, \Join_\theta, \ltimes_\theta, \ltimes\ltimes_\theta, \times\ltimes_\theta, \times\ltimes\ltimes_\theta, \triangleright_\theta, \triangleleft_\theta$
Mengenoperationen	\cup, \cap, \setminus
Division	\div
Gruppierung/Aggregation	$\Gamma_{A_1, \dots, A_n; a_1:f_1, \dots, a_m:f_m}$
Umbenennung	ρ_N , oder $\rho_{a_1 \leftarrow b_1, \dots, a_n \leftarrow b_n}$

Relationale Algebra

Finde Studenten (nur Namen ausgeben), die im gleichen Semester sind wie Feuerbach.

Tupelkalkül

Finde Studenten, die die Vorlesung Grundzüge hören.

Domänenkalkül

Finde Studenten, die die Vorlesung Grundzüge hören.

ALL-Quantifizierung

Finde Studenten, die ALLE Vorlesungen gehört haben.

Tupelkalkül:

ALL-Quantifizierung

Finde Studenten, die ALLE Vorlesungen gehört haben.

Domänenkalkül:

ALL-Quantifizierung

Finde Studenten, die ALLE Vorlesungen gehört haben.

Relationale Algebra:

Einschub: Relationale Division

$$R \div S = \Pi_{\mathcal{R} \setminus \mathcal{S}}(R) \setminus \Pi_{\mathcal{R} \setminus \mathcal{S}}((\Pi_{\mathcal{R} \setminus \mathcal{S}}(R) \times S) \setminus R)$$

Voraussetzung: $\mathcal{R} \supseteq \mathcal{S}$

Beispiel:

Studenten : {[Name]}

besucht : {Name, Titel}

Studenten \div *besucht*

ALL-Quantifizierung

Finde Studenten, die ALLE Vorlesungen gehört haben.

SQL:

Andere Join Arten | Semi-Join

$Q_1 : \text{Vorlesungen} \bowtie \text{hoeren}$

SQL (mit EXISTS):

SQL (ohne EXISTS):

Andere Join Arten | Outer-Join

$Q : R \bowtie S$, mit $R = \{a, b, c\}$ und $S = \{c, d, e\}$

Domänenkalkül:

Relationale Entwurftheorie

Funktionale Abhängigkeiten (kurz FDs, für functional dependencies):

- ▶ Seien α und β Attributmengen eines Schemas \mathcal{R} .
- ▶ Wenn auf \mathcal{R} die FD $\alpha \rightarrow \beta$ definiert ist, dann sind nur solche Ausprägungen R zulässig, für die folgendes gilt:
 - ▶ Für alle Paare von Tupeln $r, t \in R$ mit $r.\alpha = t.\alpha$ muss auch gelten $r.\beta = t.\beta$.

Funktionale Abhängigkeiten

Seien $\alpha, \beta, \gamma, \delta \in \mathcal{R}$

Axiome von Armstrong:

▶ *Reflexivität:*

Falls $\beta \subseteq \alpha$, dann gilt immer $\alpha \rightarrow \beta$

▶ *Verstärkung:*

Falls $\alpha \rightarrow \beta$ gilt, dann gilt auch $\alpha\gamma \rightarrow \beta\gamma$

▶ *Transitivität:*

Falls $\alpha \rightarrow \beta$ und $\beta \rightarrow \gamma$ gelten, dann gilt auch $\alpha \rightarrow \gamma$

Mithilfe dieser Axiome können alle *geltenden* FDs hergeleitet werden.

Sei F eine FD-Menge. Dann ist F^+ die Menge aller geltenden FDs (*Hülle von F*)

Funktionale Abhängigkeiten

Nützliche und vereinfachende Regeln:

▶ *Vereinigungsregel:*

Falls $\alpha \rightarrow \beta$ und $\alpha \rightarrow \gamma$ gelten, dann gilt auch $\alpha \rightarrow \beta\gamma$

▶ *Dekompositionsregel:*

Falls $\alpha \rightarrow \beta\gamma$ gilt, dann gilt auch $\alpha \rightarrow \beta$ und $\alpha \rightarrow \gamma$

▶ *Pseudotransitivitätsregel:*

Falls $\alpha \rightarrow \beta$ und $\gamma\beta \rightarrow \delta$ gelten, dann gilt auch $\gamma\alpha \rightarrow \delta$

Schlüssel

- ▶ Schlüssel identifizieren jedes Tupel einer Relation \mathcal{R} eindeutig.
- ▶ Eine Attributmenge $\alpha \subseteq \mathcal{R}$ ist ein **Superschlüssel**, gdw. $\alpha \rightarrow \mathcal{R}$
- ▶ Ist α zudem noch *minimal*, ist es auch ein **Kandidatschlüssel** (meist mit κ bezeichnet)
 - ▶ Es existiert also kein $\alpha' \subset \alpha$ für das gilt: $\alpha' \rightarrow \mathcal{R}$
- ▶ I.A. existieren mehrere Super- und Kandidatschlüssel.
- ▶ Man muss sich bei der Realisierung für einen Kandidatschlüssel entscheiden, dieser wird dann **Primärschlüssel** genannt.
- ▶ Der triviale Schlüssel $\alpha = \mathcal{R}$ existiert immer.

Schlüssel bestimmen

- ▶ Ob ein gegebenes α ein Schlüssel ist, kann mithilfe der Armstrong Axiome ermittelt werden (zu Aufwendig!)
- ▶ Besser: Die **Attributhülle** $AH(\alpha)$ bestimmen.

- ▶ Beispiel: $\mathcal{R} = \{ A , B , C , D \}$,

mit den FDs $F_{\mathcal{R}} = \{AB \rightarrow CD, B \rightarrow C, D \rightarrow B\}$

$AH(\{D\})$:

$AH(\{A, D\})$:

Mehrwertige Abhängigkeiten

multivalued dependencies (MVDs)

“Halb-formal”:

- ▶ Seien α und β disjunkte Teilmengen von \mathcal{R}
- ▶ und $\gamma = (\mathcal{R} \setminus \alpha) \setminus \beta$
- ▶ dann ist β mehrwertig abhängig von α ($\alpha \twoheadrightarrow \beta$), wenn in jeder gültigen Ausprägung von \mathcal{R} gilt:
- ▶ Bei zwei Tupeln mit gleichem α -Wert kann man die β -Werte vertauschen, und die resultierenden Tupel müssen auch in der Relation enthalten sein.

Wichtige Eigenschaften:

- ▶ Jede FD ist auch eine MVD (gilt i.A. nicht umgekehrt)
- ▶ wenn $\alpha \twoheadrightarrow \beta$, dann gilt auch $\alpha \twoheadrightarrow \gamma$ (Komplementregel)
- ▶ $\alpha \twoheadrightarrow \beta$ ist trivial, wenn $\beta \subseteq \alpha$ ODER $\alpha \cup \beta = \mathcal{R}$ (also $\gamma = \emptyset$)

Normalformen: 1NF \supset 2NF \supset 3NF \supset BCNF \supset 4NF

- ▶ **1. NF:** Attribute haben nur atomare Werte, sind also nicht mengenwertig.
- ▶ **2. NF:** Jedes Nichtschlüsselattribut (NSA) ist voll funktional abhängig von jedem Kandidatenschlüssel.
 - ▶ β hängt **voll funktional** von α ab ($\alpha \xrightarrow{\bullet} \beta$), gdw. $\alpha \rightarrow \beta$ und es existiert kein $\alpha' \subset \alpha$, so dass $\alpha' \rightarrow \beta$ gilt.
- ▶ **3. NF:** Frei von transitiven Abhängigkeiten. Alle NSAe hängen direkt von einem Kandidatenschlüssel ab.
 - ▶ für alle geltenden nicht-trivialen FDs $\alpha \rightarrow \beta$ gilt entweder
 - ▶ α ist ein Superschlüssel, oder
 - ▶ jedes Attribut in β ist in einem Kandidatenschlüssel enthalten
- ▶ **BCNF:** Die linken Seiten (α) aller geltenden nicht-trivialen FDs sind Superschlüssel.
- ▶ **4. NF:** Die linken Seiten (α) aller geltenden nicht-trivialen MVDs sind Superschlüssel.

Höchste NF bestimmen

a) $\mathcal{R} = \{A, B, C\}, F_{\mathcal{R}} = \{AB \rightarrow C, B \rightarrow C\}$

b) $\mathcal{R} = \{A, B, C\}, F_{\mathcal{R}} = \{A \rightarrow B, B \rightarrow C\}$

Höchste NF bestimmen

c) $\mathcal{R} = \{A, B, C, D\}, F_{\mathcal{R}} = \{AB \rightarrow D, B \rightarrow C, C \rightarrow B\}$

d) $\mathcal{R} = \{A, B, C, D\}, F_{\mathcal{R}} = \{AB \rightarrow CD\}$

Höchste NF bestimmen

e) $\mathcal{R} = \{A, B, C, D, E\}, F_{\mathcal{R}} = \{A \rightarrow BCDE\}, D_{\mathcal{R}} = \{BC \twoheadrightarrow E\}$

f) $\mathcal{R} = \{A, B, C, D, E\}, F_{\mathcal{R}} = \{A \rightarrow BCDE\}, D_{\mathcal{R}} = \{AB \twoheadrightarrow E\}$

Schema in 3NF bringen

Synthesealgorithmus

▶ Eingabe:

▶ **Kanonische Überdeckung** \mathcal{F}_c

- ▶ Linksreduktion
- ▶ Rechtsreduktion
- ▶ FDs der Form $\alpha \rightarrow \emptyset$ (sofern vorhanden)
- ▶ FDs mit gleicher linke Seite zusammenfassen

▶ Algorithmus:

1. Für jede FD $\alpha \rightarrow \beta$ in \mathcal{F}_c forme ein Unterschema $\mathcal{R}_\alpha = \alpha \cup \beta$, ordne \mathcal{R}_α die FDs $\mathcal{F}_\alpha := \{\alpha' \rightarrow \beta' \in \mathcal{F}_c \mid \alpha' \cup \beta' \subseteq \mathcal{R}_\alpha\}$ zu
2. Füge ein Schema \mathcal{R}_κ mit einem Kandidatenschlüssel hinzu
3. Eliminiere redundante Schemata, d.h. falls $\mathcal{R}_i \subseteq \mathcal{R}_j$, verwerfe \mathcal{R}_i

Synthesealgorithmus

$$\mathcal{R} = \{A, B, C, D, E, F, G\}, F_{\mathcal{R}} = \{A \rightarrow BCDE, E \rightarrow FB, F \rightarrow A\}$$

Schema in BCNF bringen

BCNF-Dekompositionsalgorithmus

- ▶ Starte mit $Z = \{\mathcal{R}\}$
- ▶ Solange es noch ein $\mathcal{R}_i \in Z$ gibt, das nicht in BCNF ist:
 - ▶ Finde eine FD $(\alpha \rightarrow \beta) \in F^+$ mit
 - ▶ $\alpha \cup \beta \subseteq \mathcal{R}_i$
 - ▶ $\alpha \cap \beta = \emptyset$
 - ▶ $\alpha \rightarrow \mathcal{R}_i \notin F^+$
 - ▶ Zerlege \mathcal{R}_i in $\mathcal{R}_{i.1} := \alpha \cup \beta$ und $\mathcal{R}_{i.2} := \mathcal{R}_i - \beta$
 - ▶ Entferne \mathcal{R}_i aus Z und füge $\mathcal{R}_{i.1}$ und $\mathcal{R}_{i.2}$ ein, also $Z := (Z - \{\mathcal{R}_i\}) \cup \{\mathcal{R}_{i.1}\} \cup \{\mathcal{R}_{i.2}\}$

Schema in 4NF bringen

4NF-Dekompositionsalgorithmus

- ▶ Starte mit $Z = \{\mathcal{R}\}$
- ▶ Solange es noch ein $\mathcal{R}_i \in Z$ gibt, das nicht in 4NF ist:
 - ▶ Finde eine MVD $\alpha \twoheadrightarrow \beta \in \mathcal{F}^+$ mit
 - ▶ $\alpha \cup \beta \subset \mathcal{R}_i$
 - ▶ $\alpha \cap \beta = \emptyset$
 - ▶ $\alpha \rightarrow \mathcal{R}_i \notin \mathcal{F}^+$
 - ▶ Zerlege \mathcal{R}_i in $\mathcal{R}_{i.1} := \alpha \cup \beta$ und $\mathcal{R}_{i.2} := \mathcal{R}_i - \beta$
 - ▶ Entferne \mathcal{R}_i aus Z und füge $\mathcal{R}_{i.1}$ und $\mathcal{R}_{i.2}$ ein, also $Z := (Z - \{\mathcal{R}_i\}) \cup \{\mathcal{R}_{i.1}\} \cup \{\mathcal{R}_{i.2}\}$

BCNF-Dekompositionsalgorithmus

$$\mathcal{R} = \{A, B, C, D, E, F\}, F_{\mathcal{R}} = \{B \rightarrow AD, DEF \rightarrow B, C \rightarrow AE\}$$

SQL (1)

Finde die "Wechsler".

Ausgabe: Name, MatrNr, Besuchte Gruppen (absteigend sortiert).

<i>Übungen</i>			
MatrNr	Gruppe	Punkte	Woche
24002	1	1	1
24002	1	0	2
26120	2	0	1
26120	3	1	2
...

SQL (2)

Finde Student(en) mit höchster Punktzahl.

Ausgabe: Name, MatrNr, Punktestand (absteigend sortiert).

<i>Übungen</i>			
MatrNr	Gruppe	Punkte	Woche
24002	1	1	1
24002	1	0	2
26120	2	0	1
26120	3	1	2
...

SQL (3)

Bonus verrechnen (wenn Anzahl Punkte > 1).

Ausgabe: Name, MatrNr, '[Kein] Bonus erhalten'.

<i>Übungen</i>			
MatrNr	Gruppe	Punkte	Woche
24002	1	1	1
24002	1	0	2
26120	2	0	1
26120	3	1	2
...

