Pflichtmodul Informationssysteme (SS 2020)

Prof. Dr. Jens Teubner

Leitung der Übungen: Thomas Lindemann, Christoph Stahl

Lösungsskizze zu Übungsblatt Nr. 12

Dieses Übungsblatt bildet eine Sammlung von Aufgaben, die teilweise Aufgaben in vergangenen Klausuren ähnlich sind. Es dient zur Selbstkontrolle und soll einen Überblick über einiger der über das Semester hinweg behandelten Themen verschaffen. Zudem wurde jede Aufgabe mit Punkten annotiert. Diese geben an, wieviel Zeit für die Bearbeitung der Aufgabe vorgesehen wird. Ein Punkt entspricht einer Minute Bearbeitungszeit. Insgesamt gibt es 60 Punkte zu verteilen, für das Blatt sind somit 60 Minuten vorgesehen.

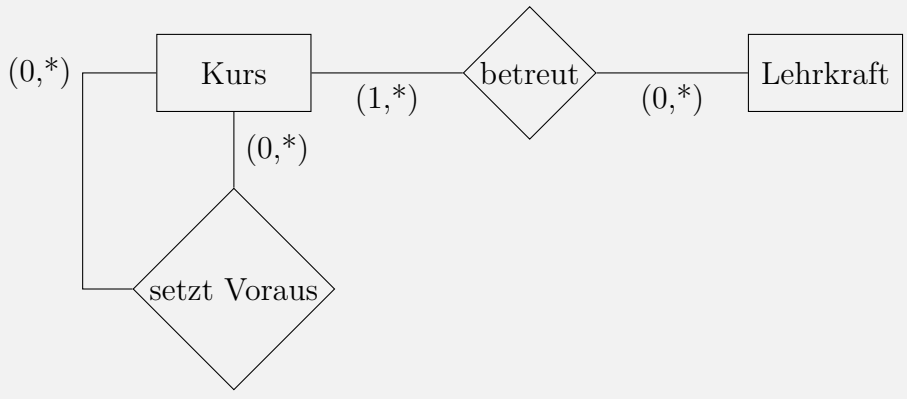
Der Stoff dieses Übungsblattes umfasst zwar ein großen Teil des Stoffes der Veranstaltung, soll aber keineswegs als vollständige Liste an klausurrelevanten Themen verstanden werden.

Aufgabe 1 (ER-Modellierung (*Entity Relation Models*) (3 + 4 + 3 Punkte))

In einem Schulungszentrum werden verschiedene Kurse angeboten, wobei ausgewählte Kurse den Inhalt anderer Kurse voraussetzen. Je Kurs gibt mindestens eine Lehrkraft, die auch verschiedene Kurse betreuen kann.

1. Erstellen Sie für diesen Diskursbereich ein ER-Diagramm. Verwenden Sie für die Funktionalität der Beziehungstypen die Min-/Max-Notation. *(3 Punkte)*

Lösung:



2. Überführen Sie das erhaltene ER-Diagram mittels `CREATE TABLE`-Statements in SQL Tabelle und geben Sie die Primär- und Fremdschlüsselbedingungen an. Überlegen Sie sich dazu geeignete Attributnamen. *(4 Punkte)*

Lösung:

```
CREATE TABLE Kurs (
    Kursnummer  INTEGER PRIMARY KEY,
    Bezeichnung VARCHAR2(255)
);
CREATE TABLE Lehrkraft (
    Personalnummer INTEGER PRIMARY KEY,
    Name VARCHAR2(255)
);
CREATE TABLE SetztVoraus (
    Kursnummer1 INTEGER,
    Kursnummer2 INTEGER,
    FOREIGN KEY (Kursnummer1) REFERENCES Kurs(Kursnummer),
    FOREIGN KEY (Kursnummer2) REFERENCES Kurs(Kursnummer),
    PRIMARY KEY (Kursnummer1, Kursnummer2)
);
CREATE TABLE Betreut (
    Kursnummer INTEGER PRIMARY KEY,
    Personalnummer INTEGER PRIMARY KEY,
    FOREIGN KEY (Kursnummer) REFERNCES Kurs(Kursnummer),
    FOREIGN KEY (Personalnummer) REFERENCES Lehrkraft(Personalnummer)
);
```

3. Die Schulungsleitung entscheidet nun, dass jeder Kurs von genau einer Lehrkraft betreut werden soll und diese auch nur genau einen Kurs betreuen sollen. Beschreiben Sie kurz, wie sich das ER-Modell ändert, und welche Optimierungen an der SQL-Repräsentation vorgenommen werden können. *(3 Punkte)*

Lösung:

- Die Kardinalität zwischen *betreut* und *Lehrkraft* sowie zwischen *betreut* und *Kurs* ändert sich auf $(1, 1)$.
- Da jeder Lehrende nun genau einem Kurs zugeordnet werden kann, kann die entsprechende Personalnummer direkt als Attribut von `Kurs` gespeichert werden. Die Relation `Betreut` entfällt somit.

Aufgabe 2 (Relationen-Algebra (*Relational Algebra*) (6 + 8 Punkte))

1. Gegeben sei folgende Datenbank.

R		
A	B	C
1	1	x
1	4	y
5	1	x

S		
A	B	C
1	3	x
1	1	x
4	3	y

T		
A	B	C
1	3	y
4	3	y
1	1	x

Werten Sie folgende Anfragen in relationaler Algebra aus. *(3 + 3 Punkte)*

(a) $R \div \pi_{B \leftarrow A,C}(S)$

Lösung:

$R \div \pi_{B \leftarrow A,C}(S)$
A
1

(b) $(S - R) \bowtie T$

Lösung:

$(S - R) \bowtie T$		
A	B	C
4	3	y

2. Der folgende Auschnitt eines Datenbankschemas modeliert eine Datenbank an Schachspielenden. Die Relation *Player* modeliert einen Schachspielenden, die Relation *Game* ein Spiel zwischen zwei Schachspielenden. Das Attribut *Remis* gibt an, ob es sich bei dem Spiel um ein Unendschieden handelt, es kann *true* oder *false* sein.

$$\mathsf{sch}(\text{Player}) = (\underline{\text{PID}}, \text{Vorname}, \text{Nachname}, \text{Geburtsdatum})$$
$$\mathsf{sch}(\text{Game}) = (\underline{\text{GID}}, \text{SiegerID}, \text{VerliererID}, \text{Remis})$$

Hierbei ist *SiegerID* und *VerliererID* ein Fremdschlüssel für Player.

Geben Sie die folgenden natürlichsprachlichen Anfragen als Anfragen der relationalen Algebra an. *(2 + 2 + 4 Punkte)*

(a) Welche Schachspielenden haben mindestens einmal gewonnen?. (Ausgabe: *Vorname, Nachname*)

Lösung:

$$\pi_{\text{Vorname,Nachname}}(\text{Player} \bowtie_{\text{SiegerID=PID}} \sigma_{\text{Remis}=\textit{false}}(\text{Game}))$$

(b) Welche *verschiedenen* Schachspielende wurden am selben Tag geboren? (Ausgabe: PID_1, PID_2)

Lösung:

$$\pi_{\text{PID}_1,\text{PID}_2}(\sigma_{G_1=G_2 \wedge \text{PID}_1 \neq \text{PID}_2}(\pi_{G_1 \leftarrow \text{Geburtsdatum},\text{PID}_1 \leftarrow \text{PID}}(\text{Player}) \times \pi_{G_2 \leftarrow \text{Geburtsdatum},\text{PID}_2 \leftarrow \text{PID}}(\text{Player})))$$

(c) Welche Schachspielende haben in mindestens zwei Spielen gewonnen? (Ausgabe: *SiegerID*)

Lösung:

$$\pi_{\text{SiegerID} \leftarrow \text{PID}_1}(\sigma_{\text{PID}_1=\text{PID}_2 \wedge \text{GID}_1 \neq \text{GID}_2}(\pi_{\text{PID}_1 \leftarrow \text{SiegerID},\text{GID}_1 \leftarrow \text{GID}}(\sigma_{\text{Remis}=\mathit{false}}(\text{Game})) \times \pi_{\text{PID}_2 \leftarrow \text{SiegerID},\text{GID}_2 \leftarrow \text{GID}}(\sigma_{\text{Remis}=\mathit{false}}(\text{Game}))))$$

Aufgabe 3 (Tupel-Relationen-Kalkül (*Tuple Relational Calculus*) (3 + 3 + 3 Pt))

Übersetzen Sie die folgenden Ausdrücke der relationalen Algebra mit Hilfe der gegebenen Übersetzungsregeln im Anhang des Übungsblattes in äquivalente Ausdrücke des Safe TRC. Dabei soll jeder einzelne Übersetzungsschritt klar erkennbar sein.

1. $\sigma_{A=5}(R)$ für $\mathsf{sch}(R) = (A, B, C)$

Lösung:

$$\begin{aligned} \mathbb{T}(v, \sigma_{A=5}(R)) &= \mathbb{T}(v, R) \wedge v.A = 5 \\ &= v \in R \wedge v.A = 5 \end{aligned}$$

2. $\pi_{A,B}(R \cup S)$ für $\mathsf{sch}(R) = (A, B, C)$ und $\mathsf{sch}(S) = (A, B, C)$

Lösung:

$$\begin{aligned} \mathbb{T}(v, \pi_{A,B}(R \cup S)) &= \exists u : (\mathbb{T}(u, R \cup S)) \wedge v \leftarrow \langle u.A, u.B \rangle \\ &= \exists u : (\mathbb{T}(u, R) \vee \mathbb{T}(u, S)) \wedge v \leftarrow \langle u.A, u.B \rangle \\ &= \exists u : (u \in R \vee u \in S) \wedge v \leftarrow \langle u.A, u.B \rangle \end{aligned}$$

3. $R \bowtie S$ für $\mathsf{sch}(R) = (A, B, C)$ und $\mathsf{sch}(S) = (A, D, E)$

Lösung:

$$
\begin{aligned}
&\mathbb{T}(v, \pi_{R.A,R.B,R.C,S.D,S.E}(\sigma_{R.A=S.A}(R \times S))) \\
&= \exists u : \mathbb{T}(u, \sigma_{R.A=S.A}(R \times S)) \wedge v \leftarrow \langle u.A, u.B, u.C, u.D, u.E \rangle \\
&= \exists u : \mathbb{T}(u, (R \times S)) \wedge u.R.A = u.S.A \wedge v \leftarrow \langle u.A, u.B, u.C, u.D, u.E \rangle \\
&= \exists w : \mathbb{T}(w, R) \wedge \exists z : \mathbb{T}(z, S) \wedge w.A = z.A \wedge v \leftarrow \langle w.A, w.B, w.C, z.D, z.E \rangle \\
&= \exists w : w \in R \wedge \exists z : z \in S \wedge w.A = z.A \wedge v \leftarrow \langle w.A, w.B, w.C, z.D, z.E \rangle
\end{aligned}
$$

Aufgabe 4 (SQL (*Structured Query Language*) (5 + 5 + 7 Punkte))

1. Im Folgenden werden Ihnen drei Momentaufnahmen von Datenbanken gegeben. Geben Sie an, wie Sie mit möglichst wenigen und möglichst kurzen SQL-Statements diese in einaner überführen können. Hierbei stellt eine Tabelle mit hellgrauem Kopf eine View dar. *(3 + 2 Punkte)*

Von:

	R			S			T	
A	B	C	D	E	F	G	H	I
1	1	x	1	3	x	1	3	y
1	4	y	1	1	x	4	3	y
5	1	x	4	3	y	1	1	x

Über:

	R			S			T	
A	B	C	D	E	F	G	H	I
1	1	x	1	3	x	1	3	y
1	4	y	1	1	x	4	3	y
5	1	x	4	3	y	1	1	x

			$View$			
A	B	C	D	E	G	H
1	1	x	1	1	1	1
1	1	x	1	3	1	1
1	4	y	4	3	1	3
1	4	y	4	3	4	3
5	1	x	1	1	1	1
5	1	x	1	3	1	1

Nach:

	R			S			T	
A	B	C	D	E	F	G	H	I
1	1	x	1	3	x	1	3	y
1	4	y	4	3	y	1	1	x

			$View$			
A	B	C	D	E	G	H
1	1	x	1	3	1	1
1	4	y	4	3	1	3

Lösung:

Hinweis: Views verändern des Zustand der Datenbank, sie sollten nicht eingeführt werden, um einen einzelnen Query zu lösen, sondern um eine häufig benötigte Sicht auf die Daten bereitzustellen.

Von → Über:
```
CREATE VIEW View FROM
    SELECT A, B, C, D, E, G, H
      FROM (R JOIN S ON C = F) JOIN T ON I = F;
```

Über → Nach:
```
DELETE FROM R WHERE A = 5;
DELETE FROM S WHERE E = 1;
DELETE FROM T WHERE G = 4;
```

2. Betrachten Sie den Semijoin Operator $\ltimes$. Gehen Sie davon aus, dass R und S die folgenden Schemata haben.

   - $\mathsf{sch}(R) = \{A, B, C\}$
   - $\mathsf{sch}(S) = \{A, D, E\}$

   (a) Geben Sie $R \ltimes S$ nur mit Basisoperatoren der relationalen Algebra an. *(3 Punkte)*

   Lösung:

   $R \ltimes S \equiv \pi_{A,B,C}(\sigma_{A'=A}(R \times \pi_{A' \leftarrow A,D,E}(S)))$

   (b) Geben Sie ein zu $R \ltimes S$ äquivalentes SQL-Statement an. *(2 Punkte)*
   Hinweis: Es gibt kein Keyword `SEMI JOIN` in SQL.

   Lösung:

   ```
   SELECT A, B, C
     FROM R, S
     WHERE R.A = S.A
   ```

3. Der folgende Auschnitt eines Datenbankschemas modeliert eine Datenbank an Schachspielenden. Die Relation *Player* modeliert einen Schachspielenden, die Relation *Game* ein Spiel zwischen zwei Schachspielenden. Das Attribut *Remis* gibt an, ob es sich bei dem Spiel um ein Unendschieden handelt, es kann *true* oder *false* sein.

   $\mathsf{sch}(\text{Player}) = (\underline{\text{PID}}, \text{Vorname}, \text{Nachname}, \text{Geburtsdatum})$
   $\mathsf{sch}(\text{Game}) = (\underline{\text{GID}}, \text{SiegerID}, \text{VerliererID}, \text{Remis})$

Geben Sie die folgenden natürlichsprachlichen Anfragen als SQL-Anfragen an.

(a) Wieviel Spiele hat *Garri Kasparow* gewonnen? (Ausgabe: *Anzahl*) *(2 Punkte)*

Lösung:

```
SELECT COUNT(*) AS Anzahl
  FROM Player JOIN Game ON PID = SiegerID
  WHERE Remis = FALSE
    AND Vorname = 'Garri' AND Nachname='Kasparow';
```

(b) Welche Schachspielende haben noch nie gewonnen? (Ausgabe: *Vorname, Nachname*) *(2 Punkte)*

Lösung:

```
SELECT Vorname, Nachname
  FROM PLAYER
  WHERE PID NOT IN (
    SELECT SiegerID FROM GAME
      WHERE Remis = FALSE
  );
```

(c) Wer ist der jüngste Spielende? (Ausgabe: *Vorname, Nachname*) *(3 Punkte)*
Hinweis: Verwenden Sie hier nicht `LIMIT`.

Lösung:

```
SELECT P.Vorname, P.Nachname
  FROM PLAYER P
  WHERE NOT EXISTS (
    SELECT * FROM PLAYER P2
      WHERE P2.Geburtsdatum < P.Geburtsdatum
  );
```

Aufgabe 5 (Schemanormalisierung (*Normalization*) (4 + 6 Punkte))

Gegeben seien das Relationenschema

$$\mathsf{sch}(R) = DBINFO$$

sowie die Menge

$$\mathcal{F} = \{DB \to INFO, \quad I \to NFO, \quad DI \to B, \quad F \to O, \quad N \to D\}$$

von funktionalen Abhängigkeiten.

1. Zeigen Sie, dass das gegebene Schema von R mit den gegebenen funktionalen Abhängigkeiten $\mathcal{F}$ **die Boyce-Codd Normalform verletzt**. *(4 Punkte)*

Lösung:

Es gilt:

- $DB \rightarrow INFO \in \mathcal{F}^+$ verletzt BCNF Bedindung nicht, denn es gilt zwar $INFO \notin \{DB\}$, aber DB ist Superkey für R.
- $I \rightarrow NFO \in \mathcal{F}^+$ verletzt BCNF Bedindung nicht, denn es gilt zwar $NFO \notin \{I\}$, aber I ist Schlüssel für R.
- $DI \rightarrow B \in \mathcal{F}^+$ verletzt BCNF Bedindung nicht, denn es gilt zwar $B \notin \{DI\}$, aber DI ist Superkey für R.
- $F \rightarrow O \in \mathcal{F}^+$ und $N \rightarrow D \in \mathcal{F}^+$ verletzen die BCNF Bedindung, nach diesen muss zerlegt werden, die Reihenfolge ist beliebig.

2. **Zerlegen Sie R mit Hilfe des BCNF-Zerlegungsalgorithmus aus der Vorlesung, sodass alle resultierenden Relationen in BCNF vorliegen.** *(6 Punkte)*

   Geben Sie dabei jeweils an, entlang welcher funktionalen Abhängigkeit Sie das Schema zerlegen. Machen Sie nach Ablauf des Algorithmus die Schemata aller erzeugten Tabellen, die zu dem zerlegten Relationenschema gehören und alle zugehörigen funktionalen Abhängigkeiten kenntlich.

Lösung:

Eine mögliche Zerlegung:

(a) Wähle ein Funktionale Abhängigkeit, für die $\mathsf{sch}(R)$ nicht in BCNF ist:
$F \rightarrow O \in \mathcal{F}^+$, aber es gilt weder $O \in \{F\}$ noch ist F Schlüssel für R.
$(F^+_{\mathcal{F}} = \{F, O\} \neq \mathsf{sch}(R))$.

(b) Wir nutzen den Algorithmus zur Zerlegung des Schemas nach $F \rightarrow O \in \mathcal{F}^+$:
$R_1(FO)$, $\mathcal{F}_1 = \{F \rightarrow O\}^+$
$R_2(FDBIN)$, $\mathcal{F}_2 = \{DB \rightarrow INF, I \rightarrow NF, DI \rightarrow B, N \rightarrow D\}^+$

$N^+_{\mathcal{F}_2} = \{N, D\}^+ \neq \mathsf{sch}(R_2)$
Damit ist N kein Schlüssel für R_2. Es wird weiter zerlegt nach $N \rightarrow D \in \mathcal{F}^+$.

$R_{2.1}(ND)$, $\mathcal{F}_{2.1} = \{N \rightarrow D\}^+$
$R_{2.2}(NFBI)$, $\mathcal{F}_{2.2} = \{I \rightarrow NF, NI \rightarrow B, NB \rightarrow IF\}^+$

Lösung:

$I^+_{\mathcal{F}_{2.2}} = \{I, N, F, B\}^+ \equiv \mathsf{sch}(R_{2.2})$
Damit ist I also ein Schlüssel für $R_{2.2}$.

$NI^+_{\mathcal{F}_{2.2}} = \{N, I, B, F\}^+ \equiv \mathsf{sch}(R_{2.2})$
Damit ist NI also ein Superkey für $R_{2.2}$.

$NB^+_{\mathcal{F}_{2.2}} = \{N, B, I, F\}^+ \equiv \mathsf{sch}(R_{2.2})$
Damit ist NB also ein Superkey für $R_{2.2}$.

Alle Relationenschemata R_1, $R_{2.1}$ und $R_{2.2}$ sind jetzt in BCNF.