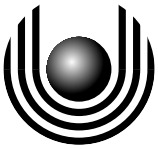
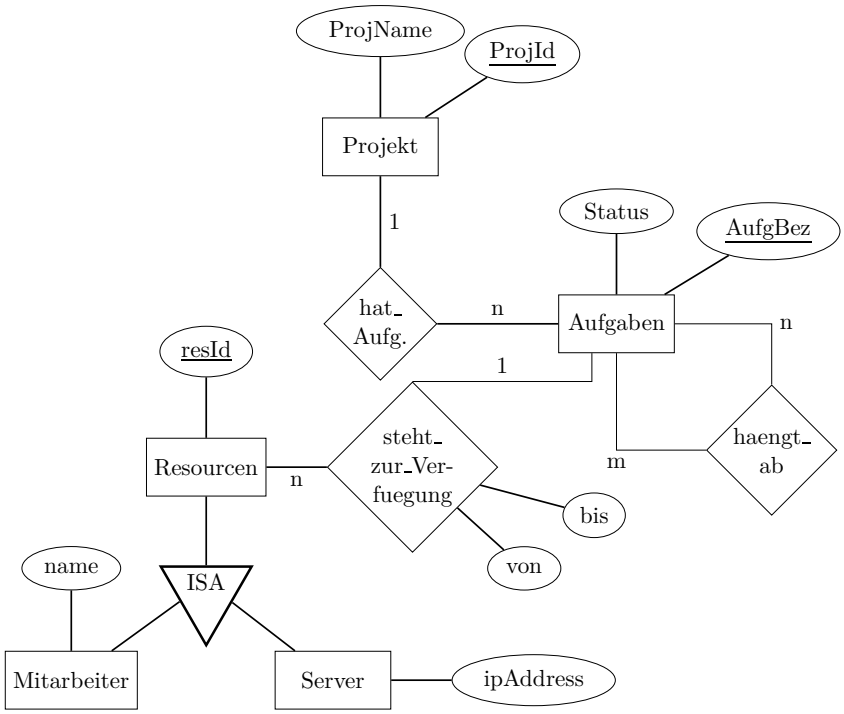


FernUniversität in Hagen

Lösungsvorschläge
zur Teilklausur
01671 „Datenbanken I“

15.09.2021

Aufgabe 1



(Punkte: Jeweils ein Punkt pro is-a-Relationship, Entity- bzw. Beziehungstyp, Kardinalität und Schlüsselattribut)
Hinweis: Für die Kardinalität der Beziehung zwischen Aufgaben und Ressourcen ist 1:n oder m:n korrekt.

Aufgabe 2

(a)

$$\pi_{TelefonNr}(Kunde - \pi_{KundenNr,TelefonNr}(Kunde \bowtie Buchung))$$

`(Hinweis: Relationen müssen vereinigungsverträglich sein)`

(b)

Um Join-Operationen möglichst effizient auszuführen, sollte überprüft werden, ob die zugehörigen Tupelmengen vor dem Join reduziert werden können. Selektionen, die sich auf eine Relation beziehen, werden daher vor dem Join ausgeführt, um die Anzahl der Tupel zu reduzieren. In der gegebenen Abfrage bezieht sich die Selektion auf die Relation Buchung. Sie wird jedoch nach dem Join ausgeführt. Daher ist die gegebene Abfrage nicht effizient. Die Abfrage wird dahingehend modifiziert, dass die Selektion vor dem Join ausgeführt wird. Der anschließende Join nutzt diese Ergebnisrelation (Ergebnis der Selektion) anstatt die gesamte Relation Buchung. Die modifizierte Abfrage sieht wie folgt aus:

$$\pi_{Ort,Kategorie}((\sigma_{Preis \geq 500 \,\wedge\, AnzahlTage \geq 3}\ Buchung) \bowtie Hotel)$$

Aufgabe 3

(a)

```
SELECT KundenID, gewerblich
FROM Kunde, Auftrag
WHERE Status = 'offen'
  AND Kunde.KundenID = Auftrag.KundenRef
```

Alternative Lösung:

```
SELECT KundenID, gewerblich
FROM Auftrag
  JOIN Kunde ON Auftrag.KundenRef = Kunde.KundenID
  AND Status = 'offen'
```

Anmerkung für obere Lösungen: Statt *KundenID*, kann auch *KundenRef* verwendet werden.
Anmerkung für die obere alternative Lösung: Die Prüfung des Wertes von *Status* kann alternativ in einem WHERE-Abschnitt erfolgen.

Weitere Lösungsmöglichkeit mittels *SUBSELECT*:

```
SELECT KundenID, gewerblich
FROM Kunde
WHERE Kunde.KundenID IN (
  SELECT KundenRef
  FROM Auftrag
  WHERE Status = 'offen'
)
```

(b)

```
SELECT KundenID
FROM Kunde
WHERE Kunde.KundenID NOT IN (
  SELECT KundenRef
  FROM Auftrag
)
```

Alternative Lösung:

```
SELECT KundenID
FROM Kunde k
  LEFT JOIN Auftrag a ON k.KundenID = a.KundenRef
WHERE a.KundenRef IS NULL
```

(c)

```
SELECT Bauteil.BauteilID, AuftragsID
FROM Bauteil
  LEFT JOIN enthaelt ON enthaelt.BauteilID = Bauteil.BauteilID
  AND Bestellmenge >= 1000
ORDER BY AuftragsID
```

(d)

```
SELECT HerstellerName
FROM (
  SELECT HerstellerName, BauteilID
  FROM Bauteil
    GROUP BY BauteilID
    HAVING Farbe = 'blau'
)
  GROUP BY HerstellerName
  HAVING COUNT( * ) >= 3
```

Hinweis: Die innere Abfrage bestimmt zunächst alle Tupel $(HerstellerName, BauteilID)$, wenn das Bauteil mit der *BauteilID* die Farbe blau hat. Die äußere Abfrage selektiert Hersteller, welche mindestens 3 unterschiedliche Bauteile produzieren.

Aufgabe 4

(a)

$F = \{\underline{LehrgebNr} \rightarrow LehrgebBez,\ \underline{LehrgebNr} \rightarrow LehrgebInNr,$
$LehrgebInNr \rightarrow LehrgebInName,\ \underline{GebNr} \rightarrow AnzLehrgeb\}$

Die Relation *Gebäude* genügt nicht der 2NF, da die Nichtschlüsselattribute nicht voll funktional von beiden Schlüsselattributen abhängen. Daher wird die Relation wie folgt zerlegt:

Lehrgebiet(LehrgebNr, GebNr, LehrgebBez, LehrgebInNr, LehrgebInName)

Gebäude_neu(GebNr, AnzLehrgeb)

(Punkte: 2 Punkte für die Fd-Menge; 2 Punkte für die Bgründung, warum die Relation nicht der 2NF genügt; 2 Punkte für die Überführung in die 2NF)

(b)

Das Ergebnis von (a) genügt nicht der 3NF, da das Nichtschlüsselattribut *LehrgebInName* transitiv von dem Schlüsselattribut *LehrgebNr* abhängt. Anders ausgedrückt, darf das Nichtschlüsselattribut *LehrgebInName* nicht voll funktional von dem Nichtschlüsselattribut *LehrgebInNr* abhängen.

Um der 3NF zu genügen muss lediglich das Attribut *LehrGebInName* entfernt werden, da sich diese Informationen bereits in der Relation mit den Daten aller Mitarbeiter befindet:

Lehrgebiet(LehrgebNr, GebNr, LehrgebBez, LehrgebInNr)

Gebäude_neu(GebNr, AnzLehrgeb)

VORGEGEBEN: Mitarbeiter(Personalnummer, Name, Dienstort)

(Punkte: 2 Punkte für die Begründung warum das Ergebnis von (a) nicht der 3NF genügt; 2 Punkte für die Überführung in die 3NF)

Hinweis: Falls die vorgegebene Relation *Mitarbeiter* übersehen und stattdessen eine weitere Relation LehrgebIn(LehrgebietInNr, LergebInName) erstellt wird, soll ein Punkt abgezogen werden.

(c)

Einfüge-Anomalie (Insertion Anomaly): Ein neuer Artikel wird aufgenommen (*ArtikelNr*=1004, *ArtikelKategorie*=“Dekoration”), ist aber noch in keinem Markt erhältlich.

Dies ist nicht ohne weiteres möglich, da ein Null-Wert für das Schlüsselattribut *Verkaufsmarkt* verwendet werden müsste.

Änderungs-Anomalie (Update Anomaly): Der Verkaufsmarkt *EasyStore* wird von *Mainz* nach *Wiesbaden* verlegt.

Die gesamte Relation muss durchsucht werden, da diese Änderung an mehreren Stellen vorgenommen werden muss, obwohl nur ein “Tatbestand” geändert wird.

Lösch-Anomalie (Delete Anomaly): Das Produkt mit der *ArtikelNr* 1003 soll aus dem Programm genommen werden.

Hierbei gehen unter Umständen weitere Daten wie in diesem Fall die Verkaufsmärkte *MegaMarkt* und *SparMarkt*, sowie deren Marktstandorte verloren.

(Punkte: 2,5 je Anomalie und Beispiel. Die Angabe zweier Anomalien ist ausreichend)

Hinweis: Von der Musterlösung abweichende Beispiele können ebenfalls korrekt sein.