Prof. Dr. Stefan Brass, Dr. Alexander Hinneburg
Institut für Informatik
Martin-Luther-Universität Halle-Wittenberg

Februar 2013

Modul Datenbanken I
— Klausur —

Name: _____

Studiengang: _____

Matrikelnummer: _____

Fühlen Sie sich gesundheitlich in der Lage die Prüfung abzulegen? ☐ Ja, ☐ Nein

Unterschrift: _____

(Diese Daten werden zur Ausstellung des Leistungsnachweises benötigt.)

Aufgabe	Punkte	Max. Punkte	Zeit
1 (Logik)		3	10 min
2 (Relationale Algebra)		2	5 min
3 (SQL-Anfragen)		18	60 min
4 (ER-Entwurf)		4	15 min
5 (Relationales Modell)		4	15 min
6 (Normalformen)		3	10 min
7 (Serialisierbarkeit)		2	5 min
Summe		36	120 min

Hinweise:

- Bearbeitungsdauer: 120 Minuten
- Skript, Bücher, Notizen sind erlaubt.
  Notebooks, PDAs, Handys etc. dürfen nicht verwendet werden.
- Die Klausur hat 16 Seiten. Bitte prüfen Sie die Vollständigkeit.
- Verwenden Sie weder Rot- noch Bleistift zum Bearbeiten der Aufgaben.
- Bitte benutzen Sie den vorgegebenen Platz. Wenn Sie auf die Rückseite ausweichen müssen, markieren Sie klar, dass es eine Fortsetzung gibt.

Beispiel-Datenbank

Die folgende Datenbank dient der HAVAG zum Verwalten der Straßenbahnlinien in Halle. Es werden die Stationen der einzelnen Linien und die möglichen Fahrten gespeichert. Das Schema der Datenbank ist

- STATION(Name, Innenstadt)
- LINIE(Nummer, Endhaltestelle→STATION, Typ)
- FAHRT((Nummer, Endhaltestelle)→LINIE,Name→STATION, Position, Dauer°)

Die Tabellen sind mit den folgenden Inhalten gefüllt. Die Inhalte dienen als Beispiele, um die Beziehungen zwischen den Tabellen zu illustrieren. Die Beispieldaten spiegeln die Wirklichkeit nicht wieder.

- Die Tabelle `LINIE` beschreibt die Straßenbahnlinien. Das Attribut Typ sagt, ob eine Linie eine Tag- oder Nachtlinie ist:

LINIE		
Nummer	Endhaltestelle	Typ
4	Kröllwitz	Tag
4	Hauptbahnhof	Tag
5	Kröllwitz	Tag
5	Ammendorf	Tag
94	Kröllwitz	Nacht
94	Marktplatz	Nacht

- Die Tabelle `STATION` beschreibt, welche Haltestellen im Straßenbahnnetz angesteuert werden. Das Attribut Innenstadt gibt an, ob eine Station zum Innenstadtbereich gehört:

STATION	
Name	Innenstadt
Kröllwitz	Nein
Hauptbahnhof	Nein
Ammendorf	Nein
Marktplatz	Ja
Rennbahnkreuz	Nein
Heide	Nein
Franckeplatz	Ja

- Die Tabelle `FAHRT` beschreibt, welche Stationen von einer Linie bedient werden. Das Attribut Position gibt an, als wievielte Station eine bestimmte Haltestelle von einer Linie angefahren wird. Das Attribut Dauer sagt in Minuten, wie lange die Fahrt zur nächsten Station mit dieser Linie dauert. Das Attribut ist NULL, wenn eine Station die letzte Station dieser Linie ist:

FAHRT				
Nummer	Endhaltestelle	Name	Position	Dauer
4	Hauptbahnhof	Kröllwitz	1	3
4	Hauptbahnhof	Heide	2	5
4	Hauptbahnhof	Rennbahnkreuz	3	5
4	Hauptbahnhof	Franckeplatz	4	5
4	Hauptbahnhof	Hauptbahnhof	5	NULL
4	Kröllwitz	Hauptbahnhof	1	5
4	Kröllwitz	Franckeplatz	2	5
4	Kröllwitz	Rennbahnkreuz	3	5
4	Kröllwitz	Heide	4	3
4	Kröllwitz	Kröllwitz	5	NULL
5	Kröllwitz	Ammendorf	1	19
5	Kröllwitz	Hauptbahnhof	2	10
5	Kröllwitz	Marktplatz	3	6
5	Kröllwitz	Rennbahnkreuz	4	5
5	Kröllwitz	Heide	5	4
5	Kröllwitz	Kröllwitz	6	NULL
5	Ammendorf	Kröllwitz	1	4
5	Ammendorf	Heide	2	5
5	Ammendorf	Rennbahnkreuz	3	6
5	Ammendorf	Marktplatz	4	10
5	Ammendorf	Hauptbahnhof	5	19
5	Ammendorf	Ammendorf	6	NULL
94	Kröllwitz	Marktplatz	1	6
94	Kröllwitz	Rennbahnkreuz	2	5
94	Kröllwitz	Heide	3	3
94	Kröllwitz	Kröllwitz	4	NULL
94	Marktplatz	Kröllwitz	1	3
94	Marktplatz	Heide	2	5
94	Marktplatz	Rennbahnkreuz	3	6
94	Marktplatz	Marktplatz	4	NULL

Aufgabe 1 (Logik) 3 Punkte

Die Informationen über die Linien, Stationen und Fahrten können durch folgende Signatur beschrieben werden. Gehen Sie davon aus, dass alle Primär- und Fremdschlüssel aus den Tabellen auf Seite 2 auch hier erfüllt sind.

- Sorten $S = \{\mathsf{int}, \mathsf{string}, \mathsf{linie}, \mathsf{station}, \mathsf{fahrt}\}$
- Funktionen der Sorte station:
  - name(station): string
  - innenstadt(station): string
- Funktionen der Sorte linie:
  - nummer(linie): int
  - endhaltestelle(linie): string
  - typ(linie): string
- Funktionen der Sorte fahrt:
  - nummer(fahrt): int
  - endhaltestelle(fahrt): string
  - name(fahrt): string
  - position(fahrt): int
  - dauer(fahrt): int
- Für die Sorten int und string gibt es die üblichen Datentyp-Funktionen/Prädikate.
- Sie können mit der Funktion is_null für int- und string-Werte abfragen, ob ein NULL-Wert vorhanden ist.

a) Formulieren Sie folgende Integritätsbedingung als logische Formel im Tupelkalkül (d.h. bezüglich obiger Signatur): Eine Fahrt darf nicht eine andere Station als die Endhaltestelle der entsprechenden Linie als letzte Station haben. $\forall linieX, fahrtY : X.nummer = Y.nummer \wedge X.endhaltestelle = Y.endhaltestelle \wedge is\_null(Y.dauer) \rightarrow X.endhaltestelle = Y.name$

b) Formulieren Sie folgende Anfrage in der Notation der Logik: Welche Paare von Linien erlauben prinzipiell ein Umsteigen am Tag, weil sie an mindestens einer gemeinsamen Station halten? Geben Sie keine symmetrischen Paare aus, d.h. nur (A, B) und nicht (B, A). Geben Sie auch keine Paare aus, die die gleiche Nummer oder die gleiche Endhaltestelle haben.

Ergebnis			
A.Nummer	A.Endhaltestelle	B.Nummer	B.Endhaltestelle
4	Hauptbahnhof	5	Kröllwitz
4	Hauptbahnhof	5	Ammendorf
4	Kröllwitz	5	Ammendorf

$\{A.Nummer, A.Endhaltestelle, B.Nummer, B.Endhaltestelle[fahrtA, fahrtB] | A.name = B.name \wedge A.nummer < B.nummer \wedge A.endhaltestelle \neq B.endhaltestelle \wedge (\exists lineC, linieD : C.nummer = A.nummer \wedge C.endhaltestelle = A.endhaltestelle \wedge$

$B.nummer = D.nummer \wedge B.endhaltestelle = D.endhaltestelle \wedge C.typ = 'Tag' \wedge D.typ = 'Tag'\}$

Platz zur Lösung von Aufgabe 1

Aufgabe 2 (Relationale Algebra) 2 Punkte

Geben Sie eine Anfrage in relationaler Algebra für die folgende Aufgabe an. Die Primär- und Fremdschlüsselbedingungen aus dem relationalen Schema gelten ebenfalls.

Geben Sie für alle Innenstadt-Stationen aus, von welchen Linien sie bedient werden.

`Ergebnis`		
Name	Nummer	Endhaltestelle
Marktplatz	5	Kröllwitz
Marktplatz	5	Ammendorf
Marktplatz	94	Kröllwitz
Marktplatz	94	Marktplatz
Franckeplatz	4	Kröllwitz
Franckeplatz	4	Hauptbahnhof

$\pi_{Name,Nummer,Endhaltestelle}(\sigma_{Innenstadt='Ja'}(Station \bowtie Fahrt))$

Aufgabe 3 (SQL-Anfragen) 18 Punkte

Formulieren Sie die folgenden Anfragen in SQL. Sie bekommen drei Punkte für jede korrekte Anfrage. Natürlich sollen Ihre Anfragen nicht nur mit den obigen Beispiel-Daten funktionieren, sondern für beliebige Tabelleninhalte. Beachten Sie, daß auch für unnötige Komplikationen Punkte abgezogen werden können. Die Anfragen sollen keine Duplikate liefern, aber für ein unnötiges `DISTINCT` oder ein unnötiges `UNION` statt `UNION ALL` werden auch Punkte abgezogen. Um Missverständnisse zu vermeiden, sind bei Aufgabe 3 jeweils die Ergebnisse der gesuchten Anfragen im Beispiel-Zustand angegeben. Falls nicht ausdrücklich eine bestimmte Spaltenüberschrift verlangt ist, müssen die Spaltenüberschriften Ihrer Anfrage nicht unbedingt mit dem Beispiel übereinstimmen.

a) Geben Sie die Nummern der Tages-Linien aus, die an Station halten, die im Namen irgend einen Platz erwähnen.

Ergebnis
Nummer
4
5

```
select distinct Nummer
from fahrt natural join linie
where type='Tag' and lower(name) like '%platz%'
```

b) Welche Tages-Linien fahren zuerst am Hauptbahnhof und danach am Franckeplatz vorbei?

Ergebnis	
Nummer	Endhaltestelle
4	Kröllwitz

```
select A.nummer, A.endhaltestelle
from linie l, fahrt A, fahrt B
where l.nummer = A.nummer and l.nummer=B.nummer and
l.endhaltestelle = A.endhaltestelle and
l.endhaltestelle = B.endhaltestelle and
A.position < B.position and
A.name='Hauptbahnhof' and B.name='Franckeplatz'
```

c) Welche Innenstadt-Stationen werden nicht von Nachtlinien bedient?

Ergebnis
Name
Franckeplatz

```
select name
from station
where Innenstadt='Ja' and
name not in
(
select name
from fahrt f, linie l
where f.nummer=l.nummer and
f.endhaltestelle = l.endhaltestelle and
l.typ='Nacht'
)
```

d) Geben Sie für jede Linien-Nummer aus, wie viele Minuten sie unterwegs ist und wie viele Stationen sie hat. Für diese Anfrage gilt die Annahme, dass es zu einer Linien-Nummer immer eine Hin- und Rückfahrt gibt und die Hin- und Rückfahrt einer Linien-Nummer die gleiche Zeit dauert und die gleichen Stationen hat. Beachten Sie, dass Wert + NULL = NULL ist. Sie können mit der Funktion isnull(Attribut,0) NULL-Werte in numerische Nullen umwandeln, so dass Wert + 0 = Wert ist.

Ergebnis		
Nummer	Gesamtdauer	AnzahlStationen
4	18	5
5	44	6
94	14	4

```
select f.nummer,
       sum(isnull(dauer,0))/2.0 as Gesamtdauer,
       count(*)/2.0 as AnzahlStationen
from Fahrt f
group by f.nummer
```

e) Geben Sie für jede Innenstadtstation, die von mehreren Linien mit unterschiedlichen Nummern besucht wird, die Anzahl der verschiedenen Liniennummern aus, von denen die Station bedient wird.

Ergebnis	
Name	AnzahlLinienNummern
Markt	2

```
select nummer, count(distinct nummer) as AnzahlLinienNummern
from station natural join fahrt
where innenstadt='Ja'
group by Name
having count(distinct nummer)>1
```

f) Geben Sie alle Stationen aus, die sich vom Franckeplatz direkt oder mit einmal Umsteigen mit beliebiger Wartezeit erreichen lassen. Geben Sie bei jeder Station in einem neuen Attribut Reiseart aus, ob die Fahrt direkt oder mit Umsteigen ist. Sortieren Sie die Ausgabe alphabetisch nach Stationsname.

Ergebnis	
Name	Reiseart
Ammendorf	umsteigen
Franckeplatz	direkt
Hauptbahnhof	direkt
Heide	direkt
Kröllwitz	direkt
Marktplatz	umsteigen
Rennbahnkreuz	direkt

```
select distinct fe.name, 'direkt' as Reiseart
from fahrt fs, fahrt fe
where
fs.name='Franckeplatz' and
fe.nummer = fs.nummer and fe.endhaltestelle = fs.endhaltestelle

union all

select distinct fe.name, 'umsteigen' as Reiseart
from fahrt fs, fahrt fu1, fahrt fu2, fahrt fe
where
fs.name='Franckeplatz' and
fu.nummer = fs.nummer and fu.endhaltestelle = fs.endhaltestelle and
fu.name = fu2.name and
fe.nummer = fu2.nummer and fe.endhaltestelle = fu2.endhaltestelle
and fe.name not in
(
select distinct fe.name, 'direkt' as Reiseart
from fahrt fs, fahrt fe
where
fs.name='Franckeplatz' and
fe.nummer = fs.nummer and fe.endhaltestelle = fs.endhaltestelle
)
order by name
```

Aufgabe 4 (ER-Entwurf) 4 Punkte

Die Havag verwaltet Kundenkarten, die für bestimmte Zeitabschnitte gelten. Es werden auch die Bearbeitungsgebühren und Ordnungsstrafen für Schwarzfahrer erfasst. Dafür sind folgende Fakten gegeben:

- Es gibt Havag-Kunden, die eine Kundenkarte haben und durch eine Kundennummer identifiziert werden. Zu einem Kunden wird der Name und die Anschrift gespeichert. Falls die Karte eine Monatsabonnement-Karte ist, wird festgehalten, bis wann das Abo noch läuft.
- Es gibt Zeitabschnitte, die durch die Kombination aus Start- und Enddatum und dem Typ identifiziert werden. Der Typ bezeichnet, ob ein Zeitabschnitt ein Monat, ein Halb-Jahr oder ein Jahr ist.
- Eine Kundenkarte ist zur Zeit für keinen oder einen Zeitabschnitt gültig. Ein Zeitabschnitt kann für keine oder viele Kundenkarten zur Zeit gültig sein.
- Wenn ein Kunde seine Karte vergessen hat und er von einem Kontrolleur beim Schwarzfahren erwischt wurde, wird von diesem Kunden eine Bearbeitungsgebühr verlangt, wenn er innerhalb einer Woche eine gültige Karte vorweisen kann. Ansonsten wird ein Ordnungsstrafe erhoben. Eine Schwarzfahrt wird durch die Kundennummer und ein laufende Nummer identifiziert. Außerdem wird ein Datum und eventuell eine Bearbeitungsgebühr und eventuell eine Ordnungsstrafe gespeichert. Ein Kunde hat keine oder viele Schwarzfahrten und eine Schwarzfahrt hat genau einen Kunden.
- Für spätere Auswertungen wird gespeichert, ob eine Kundenkarte für keinen oder viele vergangene Zeitabschnitte galt. Ein Zeitabschnitt kann mit keiner oder vielen Kundenkarten in Beziehung stehen, die für diesen Zeitabschnitt mal gegolten haben.

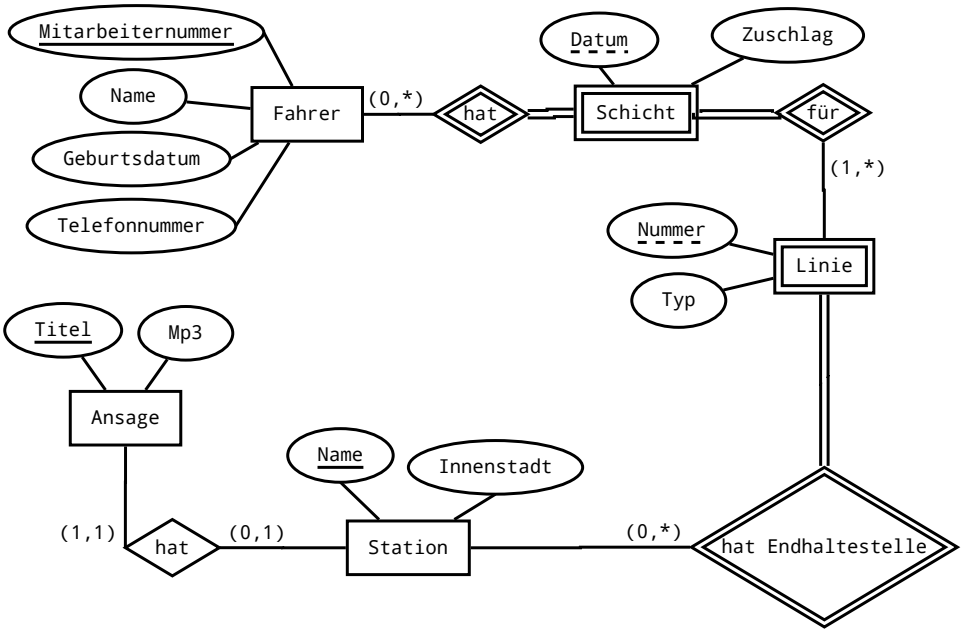
Entwerfen Sie ein Entity-Relationship-Diagramm in der Min-Max-Notation aus der DB I Vorlesung. Geben Sie die Primärschlüssel der Entities und die minimalen bzw. maximalen Kardinalitäten der Relationships an!

Lösung

Aufgabe 5 (Relationales Modell) 4 Punkte

Das folgende ER-Diagramm beschreibt die Verwaltung für die Schichten der Straßenbahnfahrer und deren Zuordnung zu den Linien, sowie die Verwaltung der Ansagen für die Stationen.

Ergänzen Sie die Übersetzung des gegebenen ER-Schemas ins Relationale Modell. Geben Sie das relationale Schema in Kurznotation (Tabelle, Attribute, Primär- und Fremdschlüssel) an. Falls es eine Spezifikation aus dem gegebenen ER-Schema gibt, die das relationales Schema nicht garantiert, geben Sie bitte explizit an, was noch geprüft werden müsste (natürliche Sprache reicht).



- Station(Name,Innenstadt)
- Linie(Nummer,Endhaltestelle→Station,Typ)

Lösung

- *Fahrer(Mitarbeiternummer, Name, Geburtsdatum, Telefonnummer)*
- *Schicht(Mitarbeiternummer→Fahrer,(Nummer,Endhaltestelle)→Line, Datum, Zuschlag)*
- *Ansage(Titel,MP3, Name→Station)*

Zusatzbedingungen

- *Name ist in Ansage ein Alternativschlüssel.*
- *Jede Line hat mindestens eine Schicht.*

Aufgabe 6 (Normalformen) 3 Punkte

Gegeben ist eine Relation $R(A, B, C, D, E, F)$ mit den funktionalen Abhängigkeiten

- $A \to C, D$
- $B \to E, F$
- $D \to B, C$
- $C, F \to A$

a) Bestimmen Sie mindestens zwei minimale Schlüssel.

- $\{D\}$
- $\{A\}$
- $\{C, B\}$
- $\{C, F\}$

b) Ist die Relation in BCNF? Wenn ja, begründen Sie ihre Antwort. Wenn nein, begründen Sie ihre Antwort und spalten Sie die Relation auf, so dass die neuen Relationen in BCNF sind. Geben Sie Primär- und Fremdschlüssel an.

Aufgabe 7 (Serialisierbarkeit) 2 Punkte

Gegeben sei folgender Schedule:

T1	T2	T3	T4
		read(Y)	
			read(Z)
			write(Z)
read(X)			
write(X)			
		write(Z)	
read(Y)			
		write(Y)	
	read(Z)		
	read(Y)		
	write(Y)		
	read(X)		
	write(X)		
commit			
	commit		
		commit	
			commit

a) Geben Sie den Konfliktgraphen für den Schedule an. *T3 → T2; T4 → T3; T4 → T2; T1 → T2; T1 → T3*

b) Ist der Schedule serialisierbar? *Ja. T1 T4 T3 T2*