Dr. Stefan Brass
Institut für Informatik
Universität Giessen

13. Dezember 2000

Vorlesung Datenbanksysteme
— Zwischenklausur —

Name: ______________________________

Aufgabe	Punkte	Maximale Punktzahl
1 (ER Design)		12
2 (ER-Diagramm → Tabellen)		10
3 (Relationale Algebra)		9
4 (Relationale Algebra)		4
4 (SQL)		9
Summe		44

Anleitung

- Sie haben Zeit bis 11^{45} (d.h. ca.1:20).
- Bitte schreiben Sie die Antworten in den vorgegebenen Platz. Falls der Platz nicht ausreicht, können Sie weitere Blätter oder die Rückseite verwenden, aber bitte markieren Sie klar, daß es noch eine Fortsetzung gibt. Schreiben Sie Ihren Namen und die Nummer der Aufgabe auf alle zusätzlichen Blätter.
- Bitte fragen Sie, wenn Sie die Aufgabe nicht verstehen, oder die Aufgabe Ihnen mehrdeutig erscheint.
- Die Klausur hat 10 Seiten. Bitte prüfen Sie die Vollständigkeit.
- Bitte nehmen Sie keine Klausur mit und auch keine Notizen über die Aufgaben. Sie bekommen die Klausur zurück sobald sie korrigiert ist.
- Sie können alle schriftlichen Aufzeichnungen verwenden, die vor Beginn der Klausur entstanden sind. Dagegen sind Computer (Laptops etc.) leider nicht zulässig.
- Bitte vermeiden Sie jedes Verhalten, das als Täuschungsversuch mißverstanden werden könnte. Schauen Sie nicht in Richtung der Klausuren anderer Teilnehmer und sprechen Sie nicht mit anderen Teilnehmern, bis alle Klausuren abgegeben sind.
- Bitte stellen Sie sicher, daß ich Ihre Handschrift lesen kann. Verwenden Sie keine rote Tinte.
- Viel Erfolg!

Aufgabe 1 (ER Entwurf) 12 Punkte

Es ist eine Datenbank zu entwickeln, in der Busfahrpläne abgespeichert werden können. Folgende Daten sollen gespeichert werden:

- Haltestellen: Für Haltestellen muß der Name und die Gemeinde abgespeichert werden. Nur beide zusammen identifizieren die Haltestelle eindeutig (z.B. gibt es „Heuchelheim Ost" und „Waldgirmes Ost").
- Linien: Es müssen verschiedene Streckenführungen der gleichen Linie unterschieden werden, z.B. fährt die Linie 24 manchmal bis Kinzenbach und manchmal bis Wetzlar. Sie startet auch nicht immer in Gießen Bahnhof. Schließlich müssen Hin- und Rückrichtung unterschieden werden. Zur Vereinfachung soll in der Datenbank eine Linie eine genau bestimmte Folge von Haltestellen sein. Es gibt dann also eine ganze Reihe von Linien mit der gleichen Nummer. Sie sollen über einen zusätzlichen Buchstaben unterschieden werden (also Linie 24a, 24b, etc.). Es muß aber möglich sein, auch nur auf die Nummer alleine (24) zuzugreifen. Zu jeder Linie ist auch die Gesamtlänge in Kilometern zu speichern.
- Zu jeder Linie muß festgehalten werden, welche Haltestellen sie anfährt. Für jede Haltstelle und Linie ist abzuspeichern, wieviele Minuten Busse der Linie von der Starthaltestelle bis zu dieser Haltestelle benötigen.
- Schließlich sollen Fahrten eingetragen werden, also wann ein Bus einer Linie von der jeweiligen Starthaltestelle abfährt.

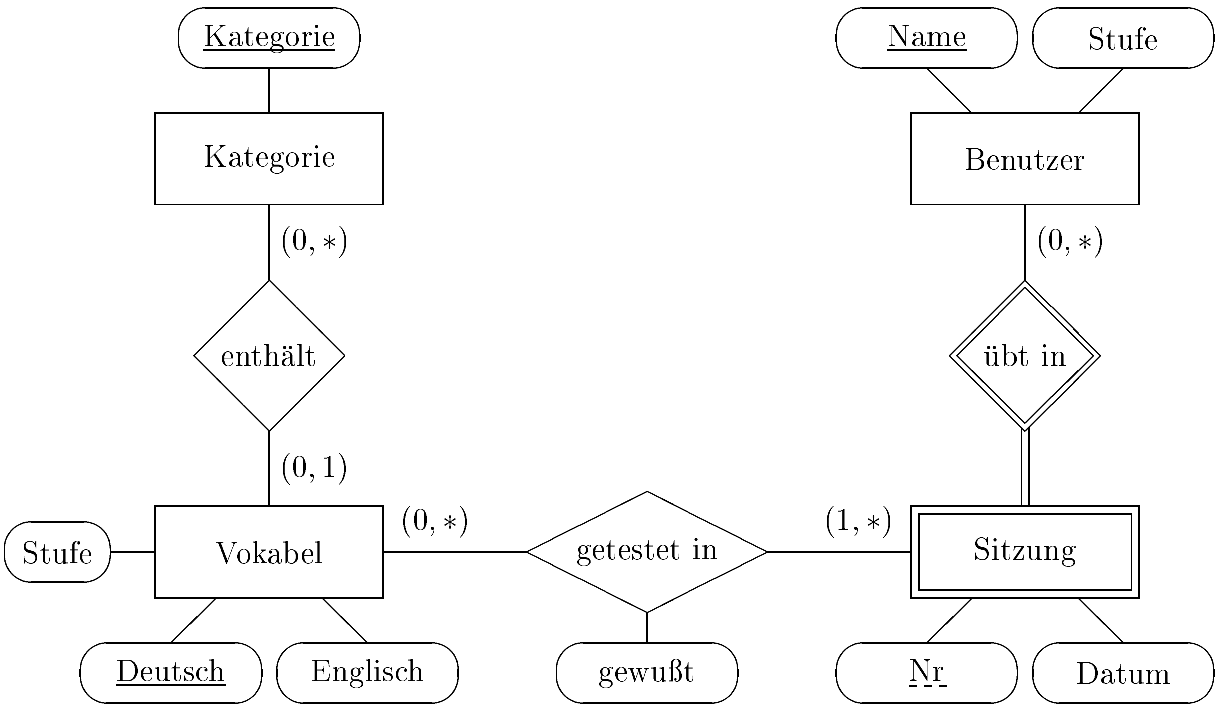
Bitte definieren Sie Schlüssel und Kardinalitäten. Es soll möglich sein, Haltestellen ohne Linien und Linien (zunächst) ohne Haltestellen in die Datenbank einzutragen. Die gleiche Haltestelle kann von mehreren Linien angefahren werden. Sie brauchen keine zusätzlichen Integritätsbedingungen zu definieren, sollen aber die ER-Konstrukte so einsetzen, daß möglichst wenig zusätzliche Integritätsbedingungen nötig wären. Im Zweifelsfall geben Sie lieber Integritätsbedingungen an.

Platz für die Lösung der Aufgabe ist auf der nächsten Seite.

Lösung zu Aufgabe 1

Aufgabe 2 (Übersetzung ER-Diagramm in Tabellen) 10 Punkte

Das folgende ER-Diagramm beschreibt die Daten eines einfachen Vokabeltrainers:



Der Vokabeltrainer kann von mehreren verschiedenen Benutzern verwendet werden. Jeder Benutzer hat einen Namen und eine Stufe (wie fortgeschritten er/sie in der Sprache ist). Wenn die Benutzer den Vokabeltrainer starten (und mindestens eine Vokabel testen), erzeugen sie damit eine Sitzung. Sitzungen werden für jeden Benutzer laufend durchnumeriert. Es wird auch das Datum jeder Sitzung gespeichert. In den Sitzungen werden dem Benutzer Vokabeln auf Deutsch angezeigt, und er/sie muß das entsprechende Gegenstück in Englisch eingeben (zur Vereinfachung berücksichtigt der Vokabeltrainer nicht, daß es mehrere englische Entsprechungen geben kann). Der Vokabeltrainer speichert für jede in der Sitzung getestete Vokabel ab, ob sie gewußt wurde oder nicht (d.h. ob die korrekte Englische Übersetzung eingeben wurde). Für Vokabeln werden auch Schwierigkeitsstufen abgespeichert. Manche Vokabeln (aber nicht alle) gehören zu einer Kategorie (etwa Farben, Lebensmittel, Büromaterial, etc.). Jede Vokabel gehört zu maximal einer Kategorie. Es kann in der Datenbank schon Kategorien geben, für die noch keine Vokabeln eingegeben wurden.

Aufgabe 2 (ER-Diagramm → Tabellen) — Fortsetzung —

Bitte übersetzen Sie das obige ER-Diagram in das Relationenmodell. Geben Sie Schlüssel und Fremdschlüssel an und markieren Sie Attribute, die Null sein können (falls es solche Attribute gibt). Falls zusätzliche Integritätsbedingungen notwendig sein sollten, um das relationale Schema äquivalent zu dem gegebenen ER-Schema zu machen, geben Sie sie bitte an (informell reicht).

Beispiel-Datenbank für Aufgaben 3, 4 und 5

In der folgenden relationalen Datenbank sollen Informationen über ein größeres Softwareprojekt gespeichert werden. In der Tabelle `MODULES(MNAME, PROGRAMMER, LINES)` werden Informationen über die Module (Quelldateien) des Projektes gespeichert. Für jedes Modul werden Name (`NAME`), Programmierer (`PROGRAMMER`) und Zeilenzahl (`LINES`) festgehalten.

MODULES		
MNAME	PROGRAMMER	LINES
db	Stefan Brass	600
cgi	Nina Brass	400
main	Nina Brass	100
⋮	⋮	⋮

Die Tabelle `PROCS(NO, PNAME, MNAME→MODULES)` listet alle Prozeduren auf, die in einem Modul definiert sind. Zur Vereinfachung werden Prozeduren über eindeutige Nummern (`NO`) identifiziert, es werden aber auch Name (`PNAME`) und Modul (`MNAME`) abgespeichert.

PROCS		
NO	PNAME	MNAME
1	ins_student	db
2	get_stud_data	db
3	ins_exercise	db
4	get_ex_data	db
5	ins_result	db
6	get_result	db
7	parse_cgi_input	cgi
8	register_student	main
9	recursive_example	main
⋮	⋮	⋮

Schließlich gibt es noch eine Tabelle `CALLS(CALLER→PROCS, CALLEE→PROC)`, die die Information enthält, welche Prozedur (`CALLER`) welche andere Prozedur (`CALLEE`) aufruft:

CALLS	
CALLER	CALLEE
8	7
8	1
9	9
⋮	⋮

Alle Attribute sind nicht null.

Aufgabe 3 (Relationale Algebra) 9 Punkte

Formulieren Sie die folgenden Anfragen in der relationalen Algebra. Ihre Anfragen müssen mit beliebigen Datenbank-Zuständen funktionieren, nicht nur mit den Beispieldaten. Sie bekommen drei Punkte für jede korrekte Anfrage. Vermeiden Sie unnötige Komplikationen (z.B. unnötige Joins oder Projektionen auf alle Spalten). Sie brauchen nicht möglichst effiziente Lösungen anzugeben (Sie bekommen nicht mehr Punkte, wenn Sie die Selektion vor einem Join ausführen.).

a) Welche Module haben über 500 Zeilen? Drucken Sie Name des Moduls, den Programmierer, und die Zeilenzahl aus.

b) Wer hat die Prozedur „`ins_Student`“ programmiert?

c) Welche Prozeduren werden von der Prozedur „`register_Student`“ aufgerufen? Geben Sie jeweils den Namen der aufgerufenen Prozedur aus.

Aufgabe 4 (Relationale Algebra) 4 Punkte

Bitte kreuzen Sie die jeweils richtige Lösung an. Es sollte nur genau eine richtige Lösung pro Teilaufgabe geben. Die falschen Lösungen können Syntaxfehler enthalten oder das falsche Ergebnis liefern. Sie bekommen jeweils zwei Punkte, falls Sie die richtige Lösung angekreuzt haben.

a) Welche Prozeduren werden nicht aufgerufen? Gesucht sind jeweils der Name der Prozedur und ihr Modul.

- ☐ $\pi_{\mathsf{PNAME,\,MNAME}}\Big(\mathsf{PROCS}\ -\ \pi_{\mathsf{NO}\leftarrow\mathsf{CALLEE}}(\mathsf{CALLS})\Big)$
- ☐ $\pi_{\mathsf{PNAME,MNAME}}\Big(\mathsf{PROCS}\ \underset{\mathsf{NO}\neq\mathsf{CALLEE}}{\bowtie}\ \mathsf{CALLS}\Big)$
- ☐ $\pi_{\mathsf{PNAME,MNAME}}\Big(\mathsf{PROCS}\bowtie\Big(\pi_{\mathsf{NO}}(\mathsf{PROCS})\ -\ \pi_{\mathsf{NO}\leftarrow\mathsf{CALLEE}}(\mathsf{CALLS})\Big)\Big)$

b) Gegen Sie Nummer und Name aller Prozeduren in den Modulen `db` und `cgi` aus.

- ☐ $\pi_{\mathsf{NO,\,PNAME}}\Big(\sigma_{\mathsf{MNAME}=\text{'db'}}(\mathsf{PROCS})\Big)\cup\pi_{\mathsf{NO,\,PNAME}}\Big(\sigma_{\mathsf{MNAME}=\text{'cgi'}}(\mathsf{PROCS})\Big)$
- ☐ $\pi_{\mathsf{NO,\,PNAME}}\Big(\sigma_{\mathsf{MNAME}=\text{'db'}\,\wedge\,\mathsf{MNAME}=\text{'cgi'}}(\mathsf{PROCS})\Big)$
- ☐ $\sigma_{\mathsf{NO,\,PNAME}}\Big(\sigma_{\mathsf{MNAME}=\text{'db'}\,\vee\,\mathsf{MNAME}=\text{'cgi'}}(\mathsf{PROCS})\Big)$

Aufgabe 5 (SQL) 9 Punkte

Formulieren Sie die folgenden Anfragen in SQL. Beachten Sie, daß die Anfragen nicht identisch zu den Anfragen aus Aufgabe 3 sind. Sie bekommen drei Punkte für jede korrekte Anfrage.

a) Geben Sie die Nummern der Prozeduren aus, die sich selbst aufrufen (direkte Rekursion). Im Beispiel wäre das Prozedur Nummer 9.

b) Welche Module haben zwischen 200 und 500 Zeilen? Geben Sie Name des Moduls, den Programmierer, und die Zeilenzahl aus.

c) Gesucht sind alle Prozeduren, deren Name mit „`get`“ beginnt, und die in den Modulen „`db`“ und „`cgi`“ definiert sind.