Gruppe A

Bitte tragen Sie **sofort** und **leserlich** Namen, Studienkennzahl und Matrikelnummer ein und legen Sie Ihren Studentenausweis bereit.

PRÜFUNG AUS DATENBANKSYSTEME **VU 184.686**			7. 5. 2014
Kennnr.	Matrikelnr.	Familienname	Vorname

Arbeitszeit: 100 Minuten. Aufgaben sind auf den Angabeblättern zu lösen; Zusatzblätter werden nicht gewertet.

Aufgabe 1: (15)

Kreuzen Sie an, ob die folgenden Aussagen wahr oder falsch sind.

1. Betrachten Sie zwei Relationen $R(AB)$ und $S(AB)$. Dann gilt auf jeden Fall folgende Gleichheit: $\pi_A(R - S) = \pi_A R - \pi_A S$ wahr ◯ falsch ⊗
2. Beim Zweiphasen-Commit-Protokoll kann der Koordinator mittels Timer-Überwachung verhindern, dass beim Absturz eines Agenten die Beendigung einer Transaktion blockiert wird. wahr ⊗ falsch ◯
3. Betrachten Sie zwei Relationen $R(\underline{A}B)$ und $S(AC)$. Um den Ausdruck $R \bowtie S$ mittels Index Nested Loop Join auszuwerten, muss A in der Relation S als Fremdschlüssel definiert sein. wahr ◯ falsch ⊗
4. Eine Relation R sei an 4 Netzwerk-Knoten materialisiert mit den Gewichten 5, 13, 7 und 9. Dann sind $Q_r(R) = 17$ und $Q_w(R) = 18$ gültige Lese- bzw. Schreib-Quoren. wahr ⊗ falsch ◯
5. RAID garantiert die Eigenschaften *R*ecovery, *A*tomicity, *I*solation, und *D*urability. wahr ◯ falsch ⊗
6. Nehmen Sie an, dass eine Relation R 100 Seiten umfasst und dass die Puffergröße 102 beträgt. Dann verursacht ein Block-Nested-Loop-Join von R mit S geringere I/O-Kosten als ein Sort-Merge-Join. wahr ⊗ falsch ◯
7. Unterschiedliche Ordnungen beim Einfügen der Schlüssel-Werte in einen B+ Baum können zu unterschiedlichen Baum-Strukturen führen. wahr ⊗ falsch ◯
8. Die Historie $w_1(A)$, $r_2(B)$, $r_3(D)$, $w_1(B)$, $w_2(D)$, $w_3(D)$, c_1, c_2, c_3 ist serialisierbar aber nicht strikt. wahr ◯ falsch ⊗
9. Betrachten Sie die Relationen $R(\underline{A}B)$ mit 5000 Tupeln und $S(AC)$ mit 3000 Tupeln. Dann ergibt der Ausdruck $R \bowtie S$ maximal 3000 Tupel. wahr ⊗ falsch ◯
10. Mit SQL-92 lassen sich Anfragen formulieren, die man mit PL/pgSQL nicht formulieren könnte. wahr ◯ falsch ⊗

(Pro korrekter Antwort 1.5 Punkte, **pro inkorrekter Antwort -1.5 Punkte**, pro nicht beantworteter Frage 0 Punkte, für die gesamte Aufgabe mindestens 0 Punkte)

Aufgabe 2: (14)

Eine international tätige Firma mit Standorten in Europa und Amerika will eine verteilte Personaldatenbank erstellen. In dieser Datenbank sind folgende Tabellen enthalten:

Angestellter(SSN, name, fkt, geschlecht, geburtsdatum, standort, adresse)
Reisebudget(fkt, adresse, budget).

Die Angestelltentabelle wird (vertikal) fragmentiert in folgende zwei Tabellen:

AngestellterBasic(SSN, name, fkt, geschlecht, standort) und
AngestellterExtra(SSN, geburtsdatum, adresse).

Außerdem wird die Tabelle AngestellterBasic(SSN, name, fkt, geschlecht, standort) nach dem Attribut "standort" (horizontal) weiter fragmentiert in die zwei Tabellen AngestellterBasicEuropa und AngestellterBasicAmerika. Es ist die Anfrage

select name
from Angestellter a, Reisebudget r
where a.fkt = 'Abteilungsleiter' and r.budget > 20k and a.fkt = r.fkt and a.adresse = r.adresse

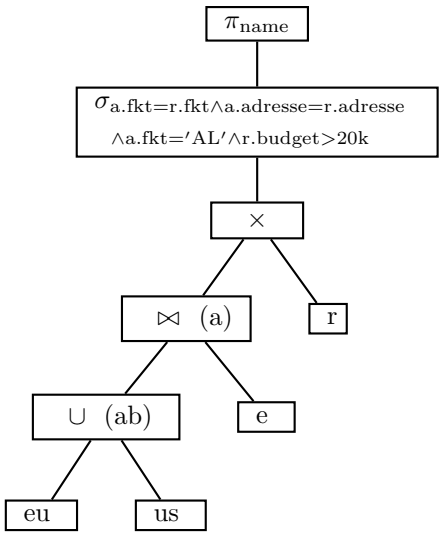
auszuführen (d.h. Informationen über Abteilungsleiter mit höherem Reisebudget).

(a) Zeichnen Sie ins erste Kästchen den Operatorbaum für die kanonische Übersetzung. Verwenden Sie für die 3 Fragmente der Tabelle Angestellter sowie für die Tabelle Reisebudget folgende Abkürzungen: **e** (AngestellterExtra), **eu** (AngestellterBasicEuropa), **us** (AngestellterBasicAmerika) und **r** (Reisebudget). Außerdem können Sie den String 'Abteilungsleiter' mit 'AL' abkürzen.

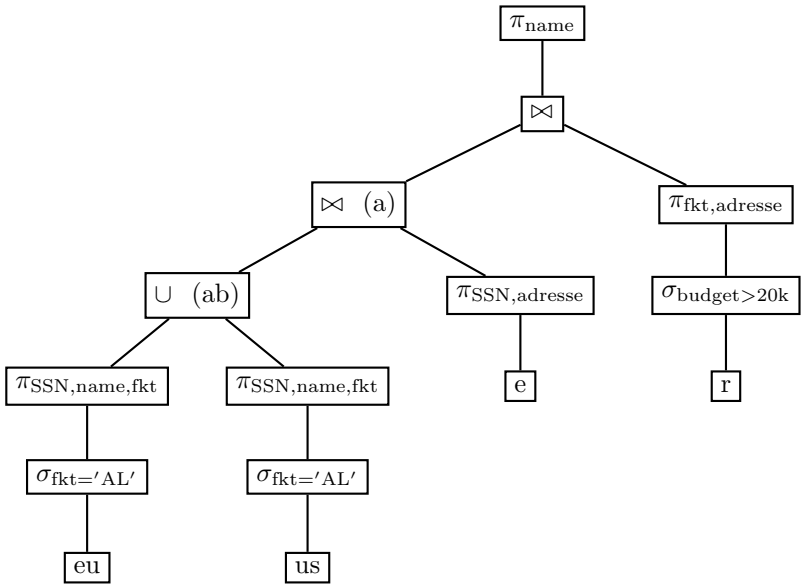
(b) Zeichnen Sie ins zweite Kästchen den Operator-Baum für den optimierten algebraischen Ausdruck. Wenden Sie für die Optimierung folgende Heuristiken an:

- Selektionen so weit wie möglich nach unten verschieben,
- Attribute, die nicht mehr benötigt werden, möglichst früh wegprojizieren,
- Kreuzprodukte durch Joins ersetzen.

(a) Kanonische Übersetzung:



(b) Optimierter Ausdruck:



Aufgabe 3: Zeitstempel-basierende Synchronisation (10)

Sie finden in der unten angeführten Tabelle Schedules von zwei Transaktionen T_1 und T_2. Es gibt zwei Datenobjekte A und B, deren Anfangswerte für readTS und writeTS jeweils 0 sind. Die beiden Zeitstempel haben die folgenden Werte: TS(T1) = 5 und TS(T2) = 10. Verwenden Sie die Regel für Zeitstempel-basierende Synchronisation wie in der Vorlesung besprochen und geben Sie nun jeweils in den rechten vier Spalte die aktuellen Werte für readTS(A), writeTS(A), readTS(B) und writeTS(B) an.

#	T_1	T_2	readTS(A)	writeTS(A)	readTS(B)	writeTS(B)
1	SELECT A INTO valueA1		5	0	0	0 .....
2	valueA = valueA1 + 10		-	-	-	-
3	UPDATE A = valueA1		5	5	0	0 .....
4		SELECT A INTO valueA2	10	5	0	0 .....
5		valueA2 = valueA2 * 2	-	-	-	-
6		UPDATE A = valueA2	10	10	0	0 .....
7	SELECT B INTO valueB1		10	10	5	0 .....
8	valueB = valueB1 + 10		-	-	-	-
9	UPDATE B = valueB1		10	10	5	5 .....
10		SELECT B INTO valueB2	10	10	10	5 .....
11		valueB2 = valueB2 - 10	-	-	-	-
12		UPDATE B = valueB2	10	10	10	10 .....

Ist der Schedule serialisierbar?
⊗ Ja ◯ Nein

Konflikt in Zeile ..... bei Transaktion ..... Verletzte Regel: .....

Aufgabe 4: Relationale Algebra (6)

Betrachten Sie folgende Operationen der relationalen Algebra: σ (Selektion), π (Projektion), $\times$ (kartesisches Produkt), ρ (Umbenennung), $\cup$ (Vereinigung) und $\setminus$ (Mengendifferenz).

Zeigen oder widerlegen Sie, dass sich aus diesen Operationen der Mengendurchschnitt ($\cap$) ausdrücken lässt.

$$R \cap S = R \setminus (R \setminus S)$$

Die folgende Datenbankbeschreibung gilt für die Aufgaben 5 – 8:

Gegeben ist folgendes stark vereinfachtes Datenbankschema, in dem die Reisebudgets für Angestellte einer Firma verwaltet werden.

`Angestellte`(ssn, name, funktion: *Reisebudget.funktion*, standort, adresse: *Reisebudget.adresse*)
`Reisebudget`(funktion, adresse, budget)

Auf der letzten Seite dieser Prüfung finden Sie eine Beispielinstanz dieses Schemas!

In der Tabelle `Angestellte` werden die Angestellten des Unternehmens gespeichert. Das Attribut `ssn` speichert der Einfachheit halber eine ganzzahlige Zahl. Treffen Sie bezüglich der Datentypen der restlichen Attribute plausible Annahmen.

In der Tabelle `Reisebudget` ordnet abhängig von `funktion` und `adresse` ein Reisebudget `budget` zu. Das `budget` soll der Einfachheit halber eine ganzzahlige Zahl sein. Treffen Sie auch hier bezüglich der Datentypen der restlichen Attribute plausible Annahmen.

Aufgabe 5: (6)

Geben Sie CREATE TABLE Statements mit allen nötigen Constraints für die beiden Tabellen an.
Achten Sie dabei insbesondere darauf, dass Ihre Statements auch in der angegebenen Reihenfolge ausführbar sind.

```
CREATE TABLE Reisebudget (
        funktion VARHCHAR(20),
        adresse VARCHAR(20),
        budget INTEGER,
        PRIMARY KEY(funktion, adresse)
);

CREATE TABLE Angestellte (
        ssn INTEGER PRIMARY KEY,
        name VARCHAR(40),
        funktion VARCHAR(20) REFERENCES Reisebudget(funktion),
        standort VARCHAR(20),
        adresse VARCHAR(40) REFERENCES Reisebudget(adresse)
);
```

Aufgabe 6: (6)

Evaluieren Sie die folgenden SQL-Statements bezüglich der Datenbankinstanz **personal** (siehe letzte Seite), und geben Sie die Ausgaben der Abfrage an. Falls mehrere Ergebnisse zurückgegeben werden, trenne sie diese durch ; (Semikolons).

```
SELECT SUM(budget) FROM Reisebudget GROUP BY adresse ORDER BY adresse;
```

7000 ; 11000; 6000

```
SELECT r.budget * COUNT(*) FROM Angestellte a, Reisebudget r
WHERE a.funktion = r.funktion AND a.adresse = r.adresse
GROUP BY a.adresse, a.funktion ORDER BY a.adresse, a.funktion;
```

4000 ; 6000 ; 4000 ; 6000 ; 1000 ; 5000

```
SELECT COUNT(*) FROM Angestellte, Reisebudget;
```

56

```
SELECT COUNT(*) FROM angestellte GROUP BY standort ORDER BY standort;
```

5 ; 3

Aufgabe 7: (8)

Erstellen Sie einen PL/pgSQL Trigger `budgetAnpassung`, der verhindert, dass das `budget` Attribut der Tabelle `Reisebudget` um mehr als 2000 reduziert wird.

Falls das Budget um mehr als 2000 reduziert wird, so soll diese Änderung nicht durchgeführt werden, und ein `NOTICE` mit dem versuchten Betrag der Reduzierung ausgegeben werden

Beispiel: Wird ein Budget versucht von 6000 auf 2000 zu reduzieren, so soll diese Änderung nicht durchgeführt werden, und 4000 ausgegeben werden.

```
CREATE OR REPLACE FUNCTION fBudgetAnpassung() RETURNS TRIGGER AS $$
DECLARE
   reduktion INTEGER;
BEGIN
   reduktion := OLD.budget - NEW.budget
   IF (reduktion > 2000) THEN
      RAISE NOTICE reduktion;
      RETURN NULL;
   END IF;
   RETURN NEW;
END;
$$ LANGUAGE plpgsql;

CREATE TRIGGER budgetAnpassung BEFORE UPDATE ON Reservierung FOR EACH ROW
EXECUTE PROCEDURE fbudgetAnpassung();
```

Aufgabe 8: (10)

Vervollständigen Sie die Java Methode `budget`, der ein `standort` übergeben wird, und die

- für jede `adresse` die Anzahl der Angestellten ausgibt
  (Beispiel bezogen auf die Instanz **personal**, wenn Standort *A* übergeben wird:
  Firmenstrasse 1: 3 Angestellte sowie *Zweigstellengasse 2: 2 Angestellte*)
- das Reisebudget an diesem Standort um 1000 erhöht
  (Beispiel bezogen auf die Instanz **personal**, wenn Standort *B* übergeben wird:
  Abteilungsleitung / Aussenstellengasse 3 wird auf `budget` 5000 und *Basis / Aussenstellengasse 3* auf 4000 gesetzt)

Verwenden Sie ausschliesslich PreparedStatements (d.h. keine anderen Arten von Statements). Legen Sie die PreparedStatements hier an. Sie können hier auf eine Connection `c` zugreifen.

```
PreparedStatement psSelect = c.prepareStatement(
   "SELECT standort, COUNT(*) FROM Angestellte WHERE standort=? GROUP BY standort");

PreparedStatement psSelect = c.prepareStatement(
   "UPDATE Reisebudget SET budget=budget+1 WHERE adresse="+
   "(SELECT adresse FROM Angestellte WHERE standort=?)");
```

Vervollständigen Sie nun die Methode `budget`. Sie können die oben angelegten PreparedStatements verwenden. Um die genaue Formatierung der Ausgabe und die Fehlerbehandlung brauchen Sie sich nicht zu kümmern. Schliessen sie geöffnete Ressourcen (jedoch nicht solche, die bei der nächsten Ausführung der Methode noch benötigt werden).

```
public void budget(String standort) throws Exception {
   psSelect.setInt(1, standort);
   ResultSet rs = psReihen.executeQuery();
   while(rs.next()) {
             System.out.println(psSelect.getString(1) + ": " +
                     psSelect.getInt(2) + " Angestellte");
   }
   rs.close();

  psUpdate.setInt(1, standort);
  psUpdate.executeUpdate();
}
```

Gesamtpunkte: 75

Sie können diese Seite abtrennen und brauchen sie nicht abgeben!

Datenbankinstanz **personal**:

Reisebudget		
funktion	adresse	budget
Management	Firmenstrasse 1	6 000
Abteilungsleitung	Firmenstrasse 1	3 000
Basis	Firmenstrasse 1	2 000
Management	Zweigstellengasse 2	5 000
Basis	Zweigstellengasse 2	1 000
Abteilungsleitung	Aussenstellengasse 3	4 000
Basis	Aussenstellengasse 3	3 000

Angestellte				
ssn	name	funktion	standort	adresse
1	Alice	Management	A	Firmenstrasse 1
2	Bob	Basis	A	Firmenstrasse 1
3	Carol	Basis	A	Firmenstrasse 1
4	Dave	Management	A	Zweigstellengasse 2
5	Erin	Basis	A	Zweigstellengasse 2
6	Frank	Abteilungsleitung	B	Aussenstellengasse 3
7	Grace	Basis	B	Aussenstellengasse 3
8	Harry	Basis	B	Aussenstellengasse 3