Aufgabenblatt 5
Transaktionen, Anfragen und XML

- Abgabetermin: **Sonntag, 07.07.13**
- Zur Prüfungszulassung muss ein Aufgabenblatt mit mind. 25% der Punkte bewertet werden und alle weiteren Aufgabenblätter mit mindestens 50% der Punkte.
- Die Aufgaben sollen in Zweiergruppen bearbeitet werden.
- Abgabesytem unter
  `https://www.hpi.uni-potsdam.de/naumann/sites/abgabe/dbs1_2013`
  - ausschließlich PDF-Dateien *im A4-Format*
  - *eine Datei pro Aufgabe* namens Aufgabe-<aufgabenNr>.pdf
  - *jedes* Blatt beschriftet mit Namen

Aufgabe 1: Konfliktserialisierbarkeit

Sind die folgenden Schedules konfliktserialisierbar?

Begründe deine Entscheidung jeweils auf zwei Wegen:

- mittels des graphbasierten Tests *und*
- durch Angabe eines konfliktäquivalenten seriellen Schedules bzw. durch Angabe nichtserialisierbarer, konfligierender Aktionskombinationen. Bei der Angabe eines konfliktäquivalenten seriellen Schedules ist es ausreichend, die Reihenfolge der Transaktionen anzugeben (z.B. T_1, T_2, T_3).

a) $S_1 = \langle r_1(X), r_2(X), w_1(X), r_2(Y), w_2(Y), r_1(Y) \rangle$ **3 P**

b) $S_2 = \langle r_1(X), r_2(X), w_1(X), r_2(Y), w_1(Y), r_2(Y) \rangle$ **3 P**

c) $S_3 = \langle r_3(X), r_1(Y), w_3(Y), w_2(X), w_2(Y) \rangle$ **3 P**

d) $S_4 = \langle w_1(X), w_2(X), w_2(Y), w_1(Y), r_2(X), w_3(Y) \rangle$ **3 P**

Aufgabe 2: Konsistenz, 2PL-Bedingung, Legalität

Betrachte den folgenden Schedule (entspricht S_1 aus Aufgabe 1 mit ergänzten $*$lock-Operationen):

$sl_1(X), r_1(X), sl_2(X), r_2(X), u_2(X), w_1(X), xl_2(Y), r_2(Y), w_2(Y), sl_1(Y), u_1(X), r_1(Y), u_1(Y), u_2(Y)$

a) Gib für beide Transaktionen an, ob sie jeweils konsistent sind, und begründe kurz. **2 P**

b) Gib für beide Transaktionen an, ob sie jeweils die 2PL-Bedingung erfüllen, und begründe kurz. **2 P**

c) Ist der Schedule legal, wenn die Locks in der dargestellten Reihenfolge ausgeführt werden? Warum bzw. warum nicht? (Anmerkung: Wir nehmen hierbei an, dass keine Aktionen zurückgestellt oder verschoben werden.) **1 P**

Aufgabe 3: Scheduler

Betrachte den folgenden Schedule:

$r_1(A), r_2(B), r_3(C), r_1(B), r_2(C), r_3(D), w_1(C), w_2(D), w_3(E)$

Überlege zunächst, wo in diesem Schedule $*$lock-Operationen eingefügt werden müssen, damit sich ein konsistenter, 2PL-konformer Schedule ergibt. Gib anschließend den tabellarischen Ablaufplan der Transaktionsausführung eines DBMS-Schedulers an. Kennzeichne darin, welche Sperranforderungen akzeptiert bzw. zunächst abgelehnt werden.

Hinweise zum Einfügen der $*$lock-Operationen:

- Füge Lock-, Shared-Lock- und Exclusive-Lock Operationen jeweils so nah wie möglich an der Aktion ein, die als zuerst den Lock benötigt.
- Füge die Unlocks einer Transaktion immer am Ende der jeweiligen Transaktion ein.

a) Nutze zur Erstellung der Ablaufpläne einfache Lock- und Unlock-Operationen. **5 P**

b) Nutze zur Erstellung der Ablaufpläne Shared-Lock-, Exclusive-Lock- und Unlock-Operationen. **5 P**

Aufgabe 4: XML

In dieser Aufgabe laden wir die `movie`-Daten der IMDb im XML-Format in DB2. Dazu erstellen wir eine Tabelle mit dem Namen `moviexml`. Jedes Tupel in dieser Tabelle stellt einen Film mit dessen ID (Attribut `mid`) und weiteren Informationen im XML-Format (Attribut `content`) dar. Das XML-Attribut eines Film-Tupels hat beispielsweise folgenden Inhalt (hier der Film mit der ID `Ghosts of the Past (1991) (TV)`):

```
<movie>
    <mid>Ghosts of the Past (1991) (TV)</mid>
    <title>Ghosts of the Past</title>
    <year>1991</year>
    <actors>
        <actor>
            <name>Davis, Carl (IV)</name>
        </actor>
        <actor>
            <name>McCartney, Paul</name>
        </actor>
    </actors>
</movie>
```

Um die Tabelle zu erzeugen und zu befüllen, stellen wir ein SQL-Script `create_and_insert_utf8.sql` auf der Übungsseite bereit. Dieses kann mit dem Befehl `db2 -tvf create_and_insert_utf8.sql` auf der Kommandozeile in der VM ausgeführt werden. Die in den vorigen Übungen angelegten Tabellen werden in dieser Aufgabe *nicht* benötigt.

Gib für die folgenden Anfragen jeweils sowohl eine geeignete SQL-Anfrage mit eingebetteten XQuery/XPath-Anteilen als auch die Ergebnisse der Anfrage an.

a) Gib ID, Titel und Jahr aus für alle Filme, in denen der Schauspieler mit dem Namen `Affleck, Ben` mitgespielt hat. **2 P**

b) Gib ID, Titel und Jahr aus für alle Filme, in denen ein Schauspieler mitgespielt hat, dessen Name `Kramer` beinhaltet. **2 P**

c) Gib ID, Titel, Jahr und zusätzlich die Anzahl der Schauspieler aus für alle Filme, in denen mehr als 50 Schauspieler mitgespielt haben. **3 P**

Aufgabe 5: Anfragebearbeitung

Gegeben sei das aus der Übung bekannte Produkt-Schema:

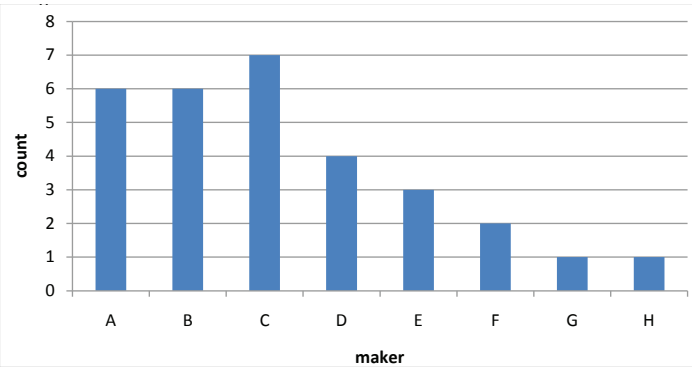
- Product(maker, model, type)
- PC(model, speed, ram, hd, rd, price)
- Laptop(model, speed, ram, hd, screen, price)
- Printer(model, color, type, price)

Für alle Relationen gilt die funktionale Abhängigkeit:

- <Relation>.model → Product.model

Nimm die folgenden Kardinalitäten/Werteverteilungen an:

- Product: 30 Tupel
- PC: 13 Tupel
- Laptop: 10 Tupel
- Printer: 7 Tupel
- maker:



Bestimme die Ergebniskardinalität (also die Anzahl der Tupel im Ergebnis) der folgenden Anfrage. Skizziere dazu den Operatorbaum und gib an jeder Kante die Anzahl der zu erwartenden Tupel an. **5 P**

$$\pi_{model,price}(\sigma_{maker='A' \vee maker='B'}(Product \bowtie (\pi_{model,price}(PC) \cup \pi_{model,price}(Laptop) \cup \pi_{model,price}(Printer))))$$