Aufgabenblatt 4
SQL

- Abgabetermin: **Mittwoch, 12.12.07, 13:30 Uhr**
- 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.
- Bitte verwendet für jede Aufgabe ein separates Blatt und beschriftet *jedes* Blatt der Abgabe mit Namen (leserlich!), Matrikelnummern und Seitenzahl – sonst gibt es je einen Punkt Abzug! **Wir korrigieren die Abgaben aufgabenweise und nehmen sie dafür auseinander.**
- Abgabe: Auf Papier im Fach „Datenbanksysteme I“ im Foyer oder per E-Mail an dbs1-2007@hpi.uni-potsdam.de mit Subject „Abgabe DBS I: Aufgabenblatt n [Namen | Matrikelnummern]“ und Anhang als ***eine* *pdf* Datei**, deren Bezeichnung eure Namen enthält.

- **Die DB2-Dokumentation findest du unter:**
  `http://publib.boulder.ibm.com/infocenter/db2luw/v9/index.jsp`
- **SQL Reference**
  `http://www.ibm.com/software/data/db2/udb/support/manualsv9.html`

Aufgabe 1: Text → SQL

Nenne für jede der folgenden natürlichsprachlichen Fragen eine geeignete SQL-Anfrage. Begründe, falls das nicht möglich ist. Führe die Anfrage auf den Daten der IMDB aus und beantworte die jeweils in der Teilaufgabe gestellte Frage.

Hinweise:

- Benenne aggregierte Spalten so um, dass sinnvolle Spaltennamen ausgegeben werden.
- Einige der Tabellen enthalten eine Spalte mit dem Namen ID. Hierbei handelt es sich jedoch um Zeilennummern und nicht um Werte, die etwa einen Film oder eine Person eindeutig identifizieren müssen.
- Treffe *soweit nötig* geeignete Annahmen bezüglich des DB-Schemas.

a) Gib alle Genres aus, die keine zugehörigen Einträge in der Filmtabelle haben (nach Genrenamen sortiert)! Jedes Genre soll dabei nur einmal ausgegeben werden. **3 P**

b) Wieviele Schauspielerinnen gibt es? **2 P**

c) Wieviele Schauspieler gibt es, die in mindestens einem Film mitgespielt haben, der einen entsprechenden Eintrag in der Filmtabelle hat? Formuliere die Anfrage einmal mit und einmal ohne Subanfrage. **4 P**

d) Gib die Titel aller Filmpaare aus, in denen mindestens ein gemeinsamer Schauspieler mitspielt! Sortiere das Ergebnis nach dem Titel des zweiten Films. **3 P**

e) Gib alle Filme (Titel, Jahr) aus, deren Titel mit “Tatort“ beginnen, und zusätzlich wie viele Schauspielerinnen an diesen Folgen jeweils beteiligt sind. **3 P**

f) Gib die Schauspieler und Produzenten an, die am Film „Adventures of Lano & Woodley, The“ beteiligt waren, und zwar einmal nach Mengen- und ein weiteres Mal nach Multimengensemantik. (Hinweis: UNION benutzen) **4 P**

g) Erstelle eine Top-3 Liste der Schauspieler und Schauspielerinnen (Name) mit den meisten Filmen! Sortiere entsprechend. Hinweis: Recherchiere hierzu die `FETCH FIRST` Klausel.
4 P

h) Formuliere eine Anfrage, die die Jahreszahl und die Anzahl der in diesem Jahr veröffentlichten Filme abfragt für

- das höchste vorkommende Jahr
- das Jahr mit den meisten veröffentlichten Filmen

5 P

Aufgabe 2: SQL → Text

Gib natürlichsprachlich wieder, wonach die folgenden SQL-Queries suchen.

a)
```
WITH ProdAct AS
    (
    SELECT Prod.Name AS PName,
        Act.Name AS AName,
        COUNT(Act.MOVIE_ID) AS CountM
    FROM (
        SELECT NAME, MOVIE_ID FROM ACTOR
        UNION ALL
        SELECT NAME, MOVIE_ID FROM ACTRESS
    ) AS Act
    INNER JOIN MOVIE AS Mov ON Act.MOVIE_ID = Mov.MID
    INNER JOIN PRODUCER AS Prod ON Mov.MID = Prod.MOVIE_ID
    GROUP BY Prod.Name, Act.Name
    ORDER BY Prod.Name, Act.Name
)

SELECT ProdAct.PName, ProdAct.AName, CountM
FROM ProdAct,
    (
    SELECT PName, MAX(countM) AS maxValue
    FROM ProdAct GROUP BY PName
    ) AS maxCount
WHERE ProdAct.PName = maxCount.PName AND
    ProdAct.CountM = maxCount.maxValue
```

5 P

b)
```
SELECT Act.NAME
FROM ACTOR AS Act
INNER JOIN MOVIE AS Mov ON Act.MOVIE_ID = Mov.MID
GROUP BY Act.NAME
HAVING COUNT(Act.MOVIE_ID) > 1
ORDER BY Act.NAME;
```

2 P

c) Hinweis: Der Operator `TABLESAMPLE BERNOULLI (prozent)` ist IBM UDB-spezifisch und gibt zufällig ausgewählte Tupel der Ergebnismenge zurück.

```
SELECT *
FROM ACTOR AS a WHERE a.MOVIE_ID IN
    (       SELECT DISTINCT m.MID
            FROM MOVIE AS m
            TABLESAMPLE BERNOULLI (0.10)
            INNER JOIN GENRE AS g ON g.MOVIE_ID = m.MID
            WHERE g.GENRE NOT LIKE '%Adult%'
        UNION
            SELECT a.MOVIE_ID AS mid
            FROM ACTOR AS a
            TABLESAMPLE BERNOULLI (0.01)
            INNER JOIN PRODUCER AS p ON p.NAME = a.NAME
            WHERE a.MOVIE_ID = p.MOVIE_ID
    )
UNION
    SELECT *
    FROM ACTOR AS a
    TABLESAMPLE BERNOULLI(2)
```

Gib zusätzlich an ob Tupel doppelt ausgegeben werden und begründe deine Antwort! **6 P**

Aufgabe 3: Relationale Algebra → SQL

Formuliere die folgenden drei Anfragen der relationalen Algebra als SQL-Anfragen!

Verwendetes Schema:

- Stadt (StadtName, LandID, p1950, p2000, p2015)
- Land (LandID, Name, Kontinent, Hauptstadt, Bevoelkerung)
- Geographie (LandID, Landfläche, Wasserfläche, Küstenlänge, urbar)

a) $\pi_{Name,Kontinent}(\sigma_{Bevoelkerung>200.000.000}(Land))$ **2 P**

b) $\pi_{Name}(\sigma_{(Bevoelkerung<2*p1950)\vee(Bevoelkerung<4*p2000)}(\sigma_{StadtName=Hauptstadt}(Stadt \bowtie Land)))$ **3 P**

c) $\pi_{Name}(Land \bowtie Geographie) -$
$\pi_{Name}(Land \bowtie (\sigma_{G1.urbar<G2.urbar}(\rho_{G1}(Geographie) \times \pi_{urbar}(\rho_{G2}(Geographie)))))$ **4 P**

Aufgabe 4: Sichten

a) Erstelle auf Basis der IMDB-Relationen einen View mit den folgenden Informationen:

- die Namen aller Schauspieler und Schauspielerinnen
- deren Geschlecht ('m'/'w')
- Anzahl ihrer Filme
- Anzahl der Genres, in denen sie gespielt haben

Das Schema würde also wie folgt aussehen: V(Name, NrMovies, NrGenres, Geschlecht). **4 P**

b) Welche Anfragen aus Aufgabe 2 lassen sich mit dem in a.) erstellten View (sinnvoll) beantworten? Gib diese Anfragen an. **4 P**