CSC 370 — Database Systems
Summer 2015
Assignment No. 3

Note 1 **This assignment is to be done individually**

Note 2 Working with other people is prohibited.

- Due date: June 5, 2015, at the beginning of the class.
- This assignment is worth 1% of your total course mark.
- Summit in paper your queries, and their corresponding relational algebra.
- Submit electronically the SQL queries in a single **text** file, including the sample results (first 10 are enough).

Objectives

After completing this assignment, you will have experience:

- Write SQL queries.

Section 2. IMDB

A. In the *imdb* database I have created a set of relations that correspond to the IMDB data. You should be able to read its relations. Look at their names and schemas, and their foreign key constraints. These are some specifics about the data:

- Productions. Correspond to all types of productions. Use the field *attr* to determine the type of production (NULL for movies, *TV* for movies-made-for-TV (they are not considered movies in the queries below), *TV-series* for tv-shows, *TV-ep* for episodes of tv-shows, *V* for "videos" (I am not sure what this is) and *VG* for video games.
- Roles. When an actor/actress appears as different characters in the same movie, the field *character* contains them all, separated by "/". For example: the dual role of Dustin Hoffman in Tootsie is encoded as *Michael Dorsey/Dorothy Michaels*.
- Persons. Information about people. *pindex* is a field that allows to distinguish people with the same firstname/lastname.
- Directors. Who is a director of a movie. *pid* refers to the key of a record in *Persons*.

B. To connect to the database:

- You need to login to one of the Linux computers in the faculty.
- To connect to the database you should use: host *studentdb.csc.uvic.ca*, database name *imdb*, your username will be posted in connex.

  `psql -h studentdb.csc.uvic.ca -U <username> imdb`
- Your password will be your uvic student-id (you can change it using SQL –see `alter user`):

- Connect to the DBMS using psql. Learn how to use psql.
- Some queries will take some time to execute. I recommend you learn to use screen.
- In psql, you can:
  - List the relations using `\d`
  - List the schema of a relation using `\d relation`
  - You can output the results of a query using `\o filename`. You stop such output using `\o`
  - Read the manual for psql for more information: `http://www.postgresql.org/docs/9.3/static/app-psql.html`
- For this assignment, use only relations that are type *relation* (output of `\d`)

Your task, should you choose to accept it

Answer the following questions, both in relational algebra, and SQL. **Relational algebra queries should match SQL**. For SQL queries provide the query and the result (if the result contains more than 10 rows, show only the first 10 and the **total number**). One query per question. Your query should only use the information provided in the question.

Remember, a production can be of different types. Use the field *attr* to determine the type of production (NULL for movies, *TV* for movies-made-for-TV (they are not considered movies in the queries below), *TV-series* for tv-shows, *TV-ep* for episodes of tv-shows, *V* for "videos" (I am not sure what this is) and *VG* for video games.

1 One of the most famous roles of Eastwood was "The Man with no Name" in the Dollar Trilogy, directed by Sergio Leone–pid *Leone, Sergio (I)*. But in each of the three movies he had a character name. For each production in which Eastwood acted, and Leone directed, list the **id** of the production, and the **character**, and the **billing.** of his role.

$$A = \Pi_{id}\sigma_{pid='Leone,Sergio(I)'}D$$

$$\Pi_{id,character,billing}\sigma_{pid='Eastwood,Clint'\wedge id\ in\ A}R$$

```
select id, character, billing from roles
   where pid = 'Eastwood, Clint' and
   id in (select id from directors where pid = 'Leone, Sergio (I)');
```

```
                  id                    | character | billing
----------------------------------------+-----------+---------
 Il buono, il brutto, il cattivo. (1966) | Blondie   |       2
 Per un pugno di dollari (1964)          | Joe       |       1
 Per qualche dollaro in piu (1965)       | Monco     |       1
(3 rows)
```

2 List the **title**, **year** and **rank** of movies (productions where attr is null) directed by the person with last-name *Hitchcock*, firstname *Alfred* and that have more than 50,000 votes. Hint: use relation ratings. **Solution:**

Simply join Productions, Directors and Ratings. That will tell us the director and ratings for each production.

$$A = P \bowtie D \bowtie R$$

Now select those by Hitchcock with at least 50k votes, and project the result.

$$\Pi_{title,year,rank}\, \sigma_{lastname='Hitchock' \text{ and } firstname='Alfred' \text{ and attr is null and votes}>50000} A$$

```
WITH A as
  (SELECT *
   FROM productions
      NATURAL JOIN persons
      NATURAL JOIN directors
    NATURAL JOIN ratings
   )
SELECT title, year, rank
FROM A
WHERE lastname = 'Hitchcock' and
      firstname = 'Alfred'
       and attr is null and votes > 50000;
```

```
        title         | year | rank
----------------------+------+------
 Dial M for Murder    | 1954 |  8.2
 North by Northwest   | 1959 |  8.4
 Notorious            | 1946 |  8.1
 Psycho               | 1960 |  8.6
 Rear Window          | 1954 |  8.6
 Rebecca              | 1940 |  8.2
 Rope                 | 1948 |  8.1
 Strangers on a Train | 1951 |  8.1
 The Birds            | 1963 |  7.8
 Vertigo              | 1958 |  8.4
```

3 One of the great pairs in movie history is Paul Newman (pid *Newman, Paul (I)*) and Robert Redford (pid *Redford, Robert (I)*). For every movie in which both acted, and their character does not include the string *Himself* (use a regular expression comparison) list the title of the movie, its year (use attribute year), its ratings rank (from relation ratings), Paul's character, and billing, and Robert's role, and billing. Result contains 3 tuples.

Solution:

What we need is to find the productions, characters, and billings of characters of Paul and then the characters of Rob. Then combine and intersect the results via a join.

Paul:

$$Paul = \Pi_{\text{id, pid, character}\to\text{paulchar, billing}\to\text{ paulbilling}} \sigma_{\text{pid = 'Newman, Paul (I)'} \wedge \text{character !~'Himself'}} R$$

Robert:

$$Rob = \Pi_{\text{id, pid, character}\to\text{robchar, billing}\to\text{robbilling}} \sigma_{\text{pid = 'Redford, Robert (I)'} \wedge \text{character !~'Himself'}} R$$

Then combine and intersect by joining Rob and Paul, and join with productions and ratings to get title, year, rank:

$$\Pi_{\text{title, year, rank, paulchar, paulbilling, robchar, robbilling}} \sigma_{\text{attr is null}} Rob \bowtie Paul \bowtie P \bowtie R$$

```
WITH Paul AS (
      SELECT id,  character as paulchar, billing AS paulbilling
      FROM roles
      WHERE pid = 'Newman, Paul (I)' and character !~ 'Himself'
   ),
   Rob AS (
      SELECT id, pid, character as robchar, billing AS robbilling
      FROM roles
      WHERE pid = 'Redford, Robert (I)' and character !~ 'Himself'
   )
SELECT title, year, rank, paulchar, paulbilling,
       robchar, robBilling
FROM
   Paul NATURAL JOIN Rob
   NATURAL JOIN productions
   NATURAL JOIN ratings
WHERE attr is null;
```

title	year	rank	paulchar	paulbilling	robchar	robbilling
Butch Cassidy and the Sundance Kid	1969	8.2	Butch Cassidy	1	The Sundance Kid	2
Mickybo and Me	2004	7.4	Butch Cassidy		The Sundance Kid	
The Sting	1973	8.4	Henry Gondorff	1	Johnny Hooker	2

4 List the **id**, **year** and **location** of any TV-series (attr *TV-series*) that was created since 2000 (inclusive) and had at least one episode filmed in *Victoria, British Columbia*. Hint: episodes are productions, the relation *episodes* links episodes with their "parent" production entry (episodeof); list the parent production, not the episodes. In the SQL report the name of the TV-series only once (use distinct).

We first find series with episodes filmed in Victoria, and project the series they are an episode-of:

$$Vi = \Pi_{episodeof,location}\sigma_{location\ =\ 'Victoria,\ BritishColumbia'}E \bowtie L$$

Then find the corresponding tuples in Productions to find if it is a series, and year >= 2000:

$$\Pi_{id,attr,year,location}\sigma_{attr='TV-series'\ and\ year>=\ 2000}Vi \bowtie P$$

```
WITH Vi AS (SELECT episodeof, location
            FROM episodes
                 NATURAL JOIN locations
            WHERE location = 'Victoria, British Columbia')
    SELECT DISTINCT p.id , p.attr, p.year, location
       FROM Vi join productions P on (Vi.episodeof = p.id)
       WHERE attr = 'TV-series'  and year >= 2000;
```

id	year	location
"Spooksville" (2013)	2013	Victoria, British Columbia
"Eaux troubles du crime" (2007)	2007	Victoria, British Columbia
"Glutton for Punishment" (2007)	2007	Victoria, British Columbia
"The Dead Zone" (2002)	2002	Victoria, British Columbia
"Improbabilia" (2013)	2013	Victoria, British Columbia
"World's Most Extreme Homes" (2006)	2006	Victoria, British Columbia
"Cedar Cove" (2013)	2013	Victoria, British Columbia
"Senior Living on Location" (2012)	2012	Victoria, British Columbia

5 Four persons directed episodes of the TV series *"Hora Marcada" (1986)* and later directed movies in English. List the **pid** of the directors, the **id** and **rank** of the movies they directed in *English*. Make sure your result includes movies without a rank. Hint: use the relations languages and episodes.

Solution:

This query has many solutions. Let us try one:

Let us first find the pid of those who directed episodes of "La hora Marcada". Call this relation A. We join Directors with Episodes:

$$A = \Pi_{pid}\sigma_{episodeof='"HoraMarcada"(1986)'}D \bowtie E$$

To find productions in English by a given director we need to join Directors, Productions and Languages. We call this relation B

$$B = D \bowtie P \bowtie L$$

Now get add the ratings, but not every movie has a rating, so make sure we include those without it:

$$C = B \bowtie_L R$$

Now all we need is to go select from this relation those directors with *pid* in A and that have NULL in *attr*, and that the language is 'English'.

$$\Pi_{pid,id,rank}\sigma_{attr\ IS\ NULL\ AND\ pid\ IN\ A\ AND\ language='English'}(C)$$

```
WITH A AS (SELECT pid FROM episodes
              NATURAL JOIN directors
              WHERE episodeof = '"Hora Marcada" (1986)'),
     B AS (SELECT * from directors NATURAL JOIN productions NATURAL JOIN languages ),
     C AS (SELECT * from B NATURAL LEFT JOIN ratings)
SELECT pid, id, rank FROM C
WHERE language = 'English' AND
      attr IS NULL AND
      pid IN (select pid from A)
order by rank desc NULLs last, id asc;
```

```
         pid          |                        id                         | rank
----------------------+---------------------------------------------------+------
 Cuarón, Alfonso      | Children of Men (2006)                            |  7.9
 Cuarón, Alfonso      | Gravity (2013)                                    |  7.9
 Cuarón, Alfonso      | Harry Potter and the Prisoner of Azkaban (2004)   |  7.8
 Cuarón, Alfonso      | A Little Princess (1995)                          |  7.7
 Cuarón, Alfonso      | Vengeance Is Mine (1983)                          |  7.4
 Cuarón, Alfonso      | Paris, je t'aime (2006)                           |  7.3
 Cuarón, Alfonso      | Sólo con tu pareja (1991)                         |  7.2
 del Toro, Guillermo  | Hellboy II: The Golden Army (2008)                |    7
 del Toro, Guillermo  | Pacific Rim (2013)                                |    7
 del Toro, Guillermo  | Cronos (1993)                                     |  6.8
 Cuarón, Alfonso      | Great Expectations (1998)                         |  6.8
 del Toro, Guillermo  | Hellboy (2004)                                    |  6.8
 del Toro, Guillermo  | Blade II (2002)                                   |  6.7
 Silva, Batan         | After Darkness (2013)                             |  6.1
 del Toro, Guillermo  | Mimic (1997)                                      |  5.9
 Gurrola, Alfredo (I) | Cabalgando con la muerte (1989)                   |  5.7
 del Toro, Guillermo  | At the Mountains of Madness (????)                |
 del Toro, Guillermo  | Crimson Peak (2015)                               |
 del Toro, Guillermo  | Hellboy 3 (????)                                  |
 del Toro, Guillermo  | Pacific Rim 2 (2017)                              |
 del Toro, Guillermo  | Pinocchio (????/II)                               |
 del Toro, Guillermo  | Saturn and the End of Days (????)                 |
 Cuarón, Alfonso      | Tales from the Hanging Head (????)                |
 del Toro, Guillermo  | The Haunted Mansion (????)                        |
 del Toro, Guillermo  | The Witches (????)                                |
(25 rows)
```

6 A common question about SQL in stackoverflow is *"What is the difference between inner and outer join"*? Make sure you can answer that question (head to stackoverflow to read the answer). However, in class we didn't talk about *inner* or *outer* joins. What is an *inner join* and what is an *outer join* from the point of view of relational algebra?

Solution: an inner join is a theta join, and an outer join a full join.

What to submit

In paper submit the Relational Algebra, the SQL queries and the results. Electronically, submit your SQL queries.