a. Joined directors table with locations table using the director's birth location ids. This returned all directors along with their birth locations. I then returned the first and last names from only those tuples where the birth location country is Canada

RESULT:

"firstname","surname"

"John","Doe"

"Jane","Smith"

b. I used the director name: "John Doe". I joined the directors table with the director_work table using the director ids, returning all directors ids and the ids of the movies they worked on. To this result I then joined the movies table using the movies id and returned only the titles of those movies where the directors firstname and lastname was "John" "Doe".

RESULT:

"title"

"Movie A"

c. I joined the actors table to the actor_work table using the actor ids. I then joined the result to the movies table using the movie ids for only those movies where the movie budget was greater than $1M. I then returned the firstname, lastname, movieTitle and the budget of the movie multiplied by the appropriate currency conversion factors:

RESULT:

"firstname","surname","title","cost_in_cad","cost_in_jpy","cost_in_rub","cost_in_eur","cost_in_chf"

"James","Wilson","Movie B","2070000.00","225480000.00","138900000.00","1410000.00","1350000.00"

"Nina","Rodriguez","Movie C","2760000.00","300640000.00","185200000.00","1880000.00","1800000.00"

"Sam","Taylor","Movie D","1656000.00","180384000.00","111120000.00","1128000.00","1080000.00"

"Ella","Garcia","Movie E","1932000.00","210448000.00","129640000.00","1316000.00","1260000.00"

"Jane","Smith","Movie C","2760000.00","300640000.00","185200000.00","1880000.00","1800000.00"

d. Using only the directors table I used surname LIKE ‘A%’ OR surname LIKE ‘D%’ to return only those tuples where the surname starts with an A or D. From the result I returned the firstname and surname.

RESULT:

"firstname","surname"

"John","Doe"

e. I used the WITH command to temporarily give names to the intermediary results. I used (EXTRACT(YEAR FROM AGE(current_date, actors."date_of_birth"))) to return all those tuples from the actors table where the actor was younger than 40. I then set under40Actors as an alias for the result.

I then used the movie_genres table and joined it with the genres table using the movie id to where the genre was 'Comedy'. From the result I return only the movie ids and set comedyMovies as an alias for the result.

Finally I joined the movies table with the comedyMovies result using the movies id and then joined this result to the actor_work table again using the movie ids. Lastly I joined this result to the under40Actors result using the actor ids to get all those actors under 40 who worked in comedy movies. I returned the movie titles from the result.

RESULT:

"title"

"Movie A"

f. I performed a self-join on the actors table such that a1.id != a2.id so no tuple contained the same actor twice. I then extracted only those tuples where the actor eye colour ='Blue' and the (a1.firstname, a2.firstname) pairs were alphabetically sorted such that a1.firstname <= a2.firstname. I used greater than or equals for the case that there were two actors with the same firstname and blue eyes but were different people. The lab instruction do not account for this situation so I decided to add it. If I did a1.firstname < a2.firstname instead I would still get the actors first names in alphabetical order but situations where there are two different actors with the same firstname and blue eyes would be excluded.

RESULT:

"firstname","firstname (1)"

"Anna","Nina"

"Anna","Ella"

"Ella","Nina"