COMP6240 - Relational Databases

Assignment 1 (SQL)

Due date: 23:59, 30 August, 2023

Instructions:

- **This assignment should be done individually (no group work)**. Do not post any idea/interpretation/partial solution/result related to this assignment on the Wattle Discussion forum. Join the drop-in sessions if you need any clarifications or need any technical support for accessing the `moviedb` database.
- This assignment will count for 20% of the final grade. Each question is worth 2 marks for a total of 20 marks.
- A copy of the `moviedb` database is available on `partch` server. You should connect to the `moviedb` database by entering the following in your terminal

```
psql moviedb
```

- You must submit one file: `myqueries.sql` for all the questions on Wattle before the due date. You can download the template file from the folder "Assignment 1 (SQL) for COMP6240" on Wattle. You are welcome to run your query against the `moviedb` database one by one following previous lab instructions. You must enter your queries into the template file, and more specifically, for the submitted file `myqueries.sql`, it should be executable in the given database `moviedb`

```
moviedb=> \i myqueries.sql
```

- The correctness of queries should not depend on any database state, and the current content in `moviedb` is available for you to get familiar with the `moviedb` database. Note that partial marks may be awarded if the query only has minor issues.
- Sample SQL questions and solutions on `moviedb` are available on Wattle, which will be helpful for you to work on your assignment.
- Late submission is not granted under any circumstance. You will be marked on whatever you have submitted at the time of the deadline. Please take careful note of deadlines and adhere to them. Of course, if you find yourself in a situation beyond your control that you believe significantly affects an assessment, you should submit an Assessment Extension Request through Wattle along with the supporting documents.
- Provide references to all the resources that you have used to complete this assignment.
- **Plagiarism will attract academic penalties in accordance with the ANU guidelines. A student in this course is expected to be able to explain and defend any submitted assessment item. The course convener can conduct or initiate an additional interview about any submitted assessment item for any student. If there is a significant discrepancy between the two forms of assessment, it will be automatically treated as a case of suspected academic misconduct.**

Questions:

The relational database `moviedb` has the following database schema:

Movie(title, production_year, country, run_time, major_genre)
    primary key : {title, production_year}

Person(id, first_name, last_name, year_born)
    primary key : {id}

Award(award_name, institution, country)
    primary key : {award_name}

Restriction_Category(description, country)
    primary key : {description, country}

Director(id, title, production_year)
    primary key : {title, production_year}
    foreign keys : [title, production_year] ⊆ Movie[title, production_year]
    [id] ⊆ Person[id]

Writer(id, title, production_year, credits)
    primary key : {id, title, production_year}
    foreign keys : [title, production_year] ⊆ Movie[title, production_year]
    [id] ⊆ Person[id]

Crew(id, title, production_year, contribution)
    primary key : {id, title, production_year}
    foreign keys : [title, production_year] ⊆ Movie[title, production_year]
    [id] ⊆ Person[id]

Scene(title, production_year, scene_no, description)
    primary key : {title, production_year, scene_no}
    foreign keys : [title, production_year] ⊆ Movie[title, production_year]

Role(id, title, production_year, description, credits)
    primary key : {title, production_year, description}
    foreign keys : [title, production_year] ⊆ Movie[title, production_year]
    [id] ⊆ Person[id]

Restriction(title, production_year, description, country)
    primary key : {title, production_year, description, country}
    foreign keys : [title, production_year] ⊆ Movie[title, production_year]
    [description, country] ⊆ Restriction_Category[description, country]

Appearance(title, production_year, description, scene_no)
    primary key : {title, production_year, description, scene_no}
    foreign keys : [title, production_year, scene_no] ⊆ Scene[title, production_year, scene_no]
    [title, production_year, description] ⊆ Role[title, production_year, description]

Movie_Award(title, production_year, award_name, year_of_award, category, result)
    primary key : {title, production_year, award_name, year_of_award, category}
    foreign keys : [title, production_year] ⊆ Movie[title, production_year]
    [award_name] ⊆ Award[award_name]

Crew_Award(id, title, production_year, award_name, year_of_award, category, result)
    primary key : {id, title, production_year, award_name, year_of_award, category}
    foreign keys : [id, title, production_year] ⊆ Crew[id, title, production_year]
    [award_name] ⊆ Award[award_name]

Director_Award(title, production_year, award_name, year_of_award, category, result)
    primary key : {title, production_year, award_name, year_of_award, category}
    foreign keys : [title, production_year] ⊆ Director[title, production_year]
    [award_name] ⊆ Award[award_name]

Writer_Award(id, title, production_year, award_name, year_of_award, category, result)
    primary key : {id, title, production_year, award_name, year_of_award, category}
    foreign keys : [id, title, production_year] ⊆ Writer[id, title, production_year]
    [award_name] ⊆ Award[award_name]

Actor_Award(title, production_year, description, award_name, year_of_award, category, result)
    primary key : {title, production_year, description, award_name, year_of_award, category}
    foreign keys : [title, production_year, description] ⊆ Role[title, production_year, description]
    [award_name] ⊆ Award[award_name]

There are five different categories of awards: movie awards, crew awards, director awards, writer awards and actor awards. A movie can only win an award after being nominated for the award.

Your task is to answer the following questions using SQL queries. For each question, your answer must be a *single SQL query* that may contain subqueries, and you must write your answers into the template file `myqueries.sql`.

1. How many persons were born before 1974 whose last name ends with 'e'? List that number. Note: You do not need to count persons with an ordinal suffix, e.g., "Kaye (I)".
2. Find the average run time of movie(s) which were produced after 1991 and categorised as 'R' restriction in the USA. List the average as a decimal (round to two decimal places). Hint: in PostgreSQL, the function ROUND(x, n) can round x to n decimal places, e.g., if x=0.1129, then ROUND(x, 2) = 0.11.
3. How many movies have exactly 2 crew members? List that number.
4. Find director(s) who have never been nominated for a director award. List their count. Hint: If an award has been won, then implicitly the winner must have been nominated.
5. Of all the directors who have directed at least one 'action' movie, list the first and last name of director(s) who have directed the fewest 'action' movies. Order your result in ascending order of their first names
6. What proportion of comedy movies are produced in Australia among all comedy movies in this database? List the proportion as a decimal (round to two decimal places).
7. Of all the movies that have won both a director award and an actor award in the same year, which movie(s) have won the smallest combined total of both director and actor awards in a single year? List their title(s) and production year(s).
8. How many movies have won at least one award (including movie awards, crew awards, director awards, writer awards and actor awards)? List that number.
9. List all the pairs of movies which have won any award in the same year. List the pairs of their title and production_year. Note that the result should not contain duplicated pairs of title and production_year, e.g., {(title1, production_year1), (title2, production_year2)} and {(title2, production_year2), (title1, production_year1)} are considered as duplicated pairs and your query should only produce one of them in the result. Hint: in PostgreSQL, the function CONCAT($A_1, A_2, \ldots, A_n$) can be used to combine selected attributes.
10. Find all the writers who have only written movies with at least one other writer (i.e., have never written a movie on their own). List their ids, first and last names.