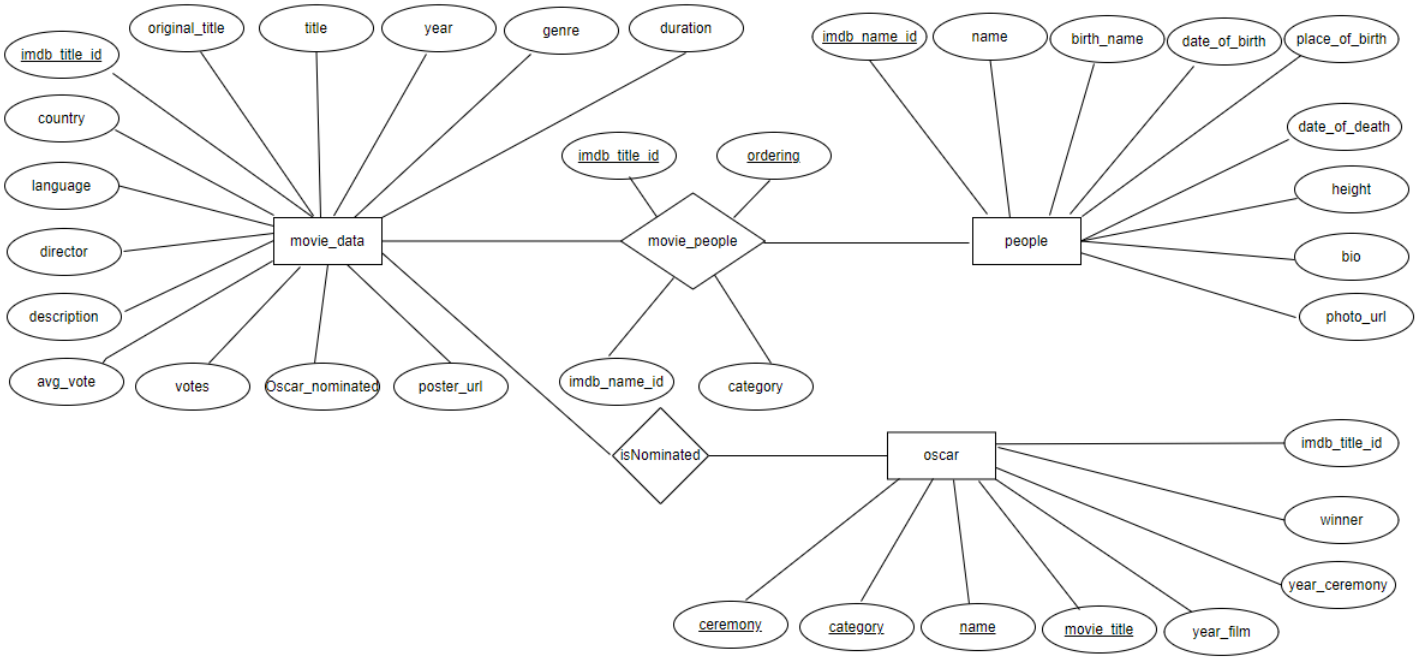
Appendix

1 DDL

```
create database MOVIE_DB;
use MOVIE_DB;

create table movie_data
(   imdb_title_id   varchar(20)  primary key,
    original_title  varchar(250) null,
    title           varchar(250) null,
    year            int          null,
    genre           varchar(250) null,
    duration        int          null,
    country         varchar(250) null,
    language        varchar(250) null,
    director        varchar(250) null,
    description     varchar(500) null,
    avg_vote        float        null,
    votes           int          null,
    Oscar_nominated tinyint(1)   null,
    poster_url      varchar(250) null);

create table movie_people
(   imdb_title_id varchar(25) not null,
    ordering      int         not null,
    imdb_name_id  varchar(25) null,
    category      varchar(25) null,
    primary key (imdb_title_id, ordering));

create table oscar
(   year_film     int         null,
    year_ceremony int         null,
    ceremony      int         not null,
    category      varchar(25) not null,
    name          varchar(25) not null,
    winner        tinyint(1)  null,
    imdb_title_id varchar(25) null,
    movie_title   varchar(25) not null,
    primary key (ceremony, category, name, movie_title),
    foreign key (imdb_title_id) references movie_data (imdb_title_id));

create table people
(   imdb_name_id   varchar(25)  not null primary key,
    name           varchar(25)  null,
    birth_name     varchar(25)  null,
    date_of_birth  varchar(25)  null,
    place_of_birth varchar(25)  null,
    date_of_death  varchar(25)  null,
    height         float        null,
    bio            varchar(1000) null,
    photo_url      varchar(150) null);
```

2 ER Diagram



3 Query

3.1 return all movies

```
SELECT title, avg_vote, poster_url,imdb_title_id
FROM movie_data
ORDER BY votes DESC, year DESC;
```

3.2 return all the information of movie

```
SELECT *
FROM movie_data
WHERE imdb_title_id = '${movie_id}'
```

3.3 return a list of movies which met the searching criteria

```
SELECT title, imdb_title_id, avg_vote, poster_url, year, director,
       genre, country, language, duration, Oscar_nominated
FROM  movie_data
WHERE title LIKE '%${title}%'
      AND (${isOscar} IS NULL OR Oscar_nominated = ${isOscar})
      AND year BETWEEN ${yearLow} AND ${yearHigh}
      AND genre LIKE '%${genre}%'
      AND country LIKE '%${country}%'
      AND language LIKE '%${language}%'
ORDER BY votes DESC, year DESC;
```

3.4 return all actors/crews of the given movie

```
SELECT P.name, P.imdb_name_id, P.photo_url
FROM people P
JOIN movie_people MP ON MP.imdb_name_id = P.imdb_name_id
WHERE MP.imdb_title_id = '${movie_id}';
```

3.5 recommend an Oscar winning movie to the user

```
SELECT md.title, md.poster_url, md.description,
       md.imdb_title_id, r.num_nominations,
       md.year, md.duration, p.name AS director_name
FROM ( SELECT imdb_title_id, COUNT(imdb_title_id) AS num_nominations
       FROM oscar
       WHERE winner = true
       GROUP BY imdb_title_id
   ) AS r
       JOIN movie_data AS md
       ON md.imdb_title_id = r.imdb_title_id
       JOIN movie_people AS mp
       ON md.imdb_title_id = mp.imdb_title_id
       AND mp.category = 'director'
       JOIN people AS p
       ON mp.imdb_name_id = p.imdb_name_id
ORDER BY RAND(DATE_FORMAT(NOW(), '%Y-%m-%d'))
LIMIT 1;
```

3.6 return most recent 50 movies with given genre

```
SELECT title, poster_url, description, imdb_title_id
FROM movie_data
WHERE genre Like '%${genre}%'
      AND title IS NOT NULL
      AND poster_url IS NOT NULL
      AND description IS NOT NULL
Order by year DESC, votes DESC
LIMIT 50
```

3.7 top 50 movies with given language

```
SELECT title, poster_url, description, imdb_title_id
FROM movie_data
WHERE language Like '%${language}%'
      AND title IS NOT NULL
      AND poster_url IS NOT NULL
      AND description IS NOT NULL
      Order by avg_vote DESC
LIMIT 50
```

3.8 return all people

```
SELECT p.imdb_name_id,p.name, p.photo_url,
       count(imdb_title_id) movies_actedIn
FROM   movie_people mp join people p on p.imdb_name_id = mp.imdb_name_id
WHERE  mp.category IN ('actor', 'director', 'producer')
group  by p.imdb_name_id
ORDER  BY movies_actedIn DESC;
```

3.9 return all information of given people

```
SELECT *
FROM people
WHERE imdb_name_id = '${person_id}'
```

3.10 movie people acted :person id

```
SELECT M.title, M.poster_url, M.imdb_title_id
FROM movie_people MP join movie_data M on  M.imdb_title_id = MP.imdb_title_id
WHERE MP.imdb_name_id = '${person_id}' and M.avg_vote > 7
order by M.avg_vote DESC
limit 20;
```

3.11 search people by name

```
SELECT *
FROM people
WHERE name LIKE '%${name}%'
```

3.12 avg vote person

```
SELECT round(AVG(M.avg_vote),2) as avg_rating
FROM movie_data M, movie_people MP, people P
WHERE M.imdb_title_id = MP.imdb_title_id
AND MP.imdb_name_id = P.imdb_name_id
AND P.imdb_name_id = '${person_id}';
```

3.13 related actors :id

```
WITH X_movies AS(
   SELECT M.imdb_title_id,M.title
   FROM movie_people MP JOIN movie_data M ON M.imdb_title_id = MP.imdb_title_id
        JOIN people P ON MP.imdb_name_id = P.imdb_name_id
   WHERE P.imdb_name_id = '${person_id}'),

coActors1 AS (
   SELECT DISTINCT P.imdb_name_id AS id, P.name AS name, P.photo_url
   FROM movie_people MP JOIN X_movies XM on XM.imdb_title_id = MP.imdb_title_id
        JOIN people P ON P.imdb_name_id = MP.imdb_name_id
   WHERE MP.imdb_name_id != '${person_id}'),

co1movies AS(
    SELECT DISTINCT MP.imdb_title_id
    FROM movie_people MP JOIN coActors1 C1 ON C1.id = MP.imdb_name_id),

coActors2 AS (
   SELECT DISTINCT P.imdb_name_id AS id, P.name AS name, P.photo_url
   FROM movie_people MP JOIN co1movies C1 ON C1.imdb_title_id = MP.imdb_title_id
        JOIN people P ON P.imdb_name_id = MP.imdb_name_id
   WHERE P.imdb_name_id != '${person_id}'
         AND
         MP.imdb_name_id NOT IN (SELECT coActors1.id FROM  coActors1)),

co2movies AS(
    SELECT DISTINCT MP.imdb_title_id
    FROM movie_people MP JOIN coActors2 C2 ON C2.id = MP.imdb_name_id),

coActors3 AS (
   SELECT DISTINCT MP.imdb_name_id AS id, MC.name AS name, MC.photo_url
   FROM movie_people MP JOIN co2movies C2 ON C2.imdb_title_id = MP.imdb_title_id
        JOIN people MC ON MC.imdb_name_id = MP.imdb_name_id
   WHERE
       MC.imdb_name_id != '${person_id}'
       AND
       MC.imdb_name_id NOT IN (SELECT coActors2.id FROM coActors2)
       AND
       MC.imdb_name_id NOT IN (SELECT coActors1.id FROM  coActors1)),
actoerTable1 AS(
   SELECT name, id, photo_url
   FROM coActors1
   UNION
   SELECT name, id, photo_url from coActors2)

SELECT name, id, photo_url
FROM actoerTable1
UNION
SELECT name, id, photo_url from coActors3
LIMIT ${offset}, ${pageSize};
```

3.14 search won

```
SELECT DISTINCT M.imdb_title_id, M.title, M.country, O.year_ceremony as year,
                M.genre, M.avg_vote, O.category, O.winner,O.name
FROM movie_data M JOIN oscar O ON M.imdb_title_id = O.imdb_title_id
ORDER BY O.year_ceremony, O.category;

SELECT DISTINCT M.imdb_title_id, M.title, M.country, O.year_ceremony as year,
                M.genre, M.avg_vote, O.category,O.winner,O.name
FROM movie_data M JOIN oscar O ON M.imdb_title_id = O.imdb_title_id
WHERE (O.year_ceremony = ${year} OR ${year} IS NULL)
ORDER BY O.year_ceremony, O.category;
```

3.15 top Oscar director

```
SELECT p.name, p.imdb_name_id, p.photo_url,
IFNULL(num_picture_nominations, 0) AS num_picture_nominations,
IFNULL(num_picture_wins, 0) AS num_picture_wins,
IFNULL(num_direction_nominations, 0) AS num_direction_nominations,
IFNULL(num_direction_wins, 0) AS num_direction_wins,
ROUND(IFNULL(m.avg_vote, 0), 1) AS avg_rating
FROM
  (
    SELECT
      mp.imdb_name_id,
      COUNT(DISTINCT CASE
        WHEN o.category IN ('Outstanding Picture', 'Outstanding Production',
        'Outstanding Motion Picture',
        'Best Motion Picture', 'Best Picture')
        THEN o.imdb_title_id ELSE NULL END) AS num_picture_nominations,
      COUNT(DISTINCT CASE
        WHEN o.category IN ('Outstanding Picture', 'Outstanding Production',
        'Outstanding Motion Picture',
        'Best Motion Picture', 'Best Picture') AND o.winner = 1
        THEN o.imdb_title_id ELSE NULL END) AS num_picture_wins,
      COUNT(DISTINCT CASE
        WHEN o.category = 'DIRECTING'
        THEN o.imdb_title_id ELSE NULL END) AS num_direction_nominations,
      COUNT(DISTINCT CASE
        WHEN o.category = 'DIRECTING' AND o.winner = 1
        THEN o.imdb_title_id ELSE NULL END) AS num_direction_wins
    FROM
      oscar o
      JOIN movie_people mp ON o.imdb_title_id = mp.imdb_title_id
    WHERE
      mp.category = 'director'
    GROUP BY
      mp.imdb_name_id
  ) AS oscar_data
  JOIN people p ON oscar_data.imdb_name_id = p.imdb_name_id
  LEFT JOIN (
    SELECT
      mp.imdb_name_id,
      AVG(m.avg_vote) AS avg_vote
    FROM
      movie_data m
      JOIN movie_people mp ON m.imdb_title_id = mp.imdb_title_id
    WHERE
      mp.category = 'director'
    GROUP BY
      mp.imdb_name_id
```

```
  ) AS m ON oscar_data.imdb_name_id = m.imdb_name_id
ORDER BY
  num_direction_wins DESC,
  num_direction_nominations DESC,
  num_picture_wins DESC,
  num_picture_nominations DESC,
  avg_rating DESC
LIMIT 10;
```

3.16 oscar decade

```
WITH oscar_movies AS (
    SELECT DISTINCT imdb_title_id, year_ceremony
    FROM oscar
  ),
  oscar_movies_with_rating AS (
    SELECT o.imdb_title_id, o.year_ceremony, m.avg_vote
    FROM movie_data m
    INNER JOIN oscar_movies o
    ON o.imdb_title_id = m.imdb_title_id
  ),
  actor_nominations AS (
    SELECT mp.imdb_name_id,
           CONCAT((om.year_ceremony DIV 10) * 10, '-',
           (om.year_ceremony DIV 10) * 10 + 9) AS decade,
           COUNT(*) AS num_nominations,
           AVG(om.avg_vote) as avg_rating
    FROM movie_people mp
    INNER JOIN oscar_movies_with_rating om
    ON mp.imdb_title_id = om.imdb_title_id
    WHERE mp.category IN ('actor','actress')
    GROUP BY mp.imdb_name_id, decade
  ),
  actor_nominations_max AS (
    SELECT MAX(num_nominations) AS max_nominations, decade
    FROM actor_nominations
    GROUP BY decade
  )
  SELECT p.name, an.max_nominations as num_nominations, an.decade,
         ROUND(a.avg_rating, 1) as avg_rating,a.imdb_name_id
  FROM actor_nominations_max an
  INNER JOIN actor_nominations a ON a.decade = an.decade
             AND a.num_nominations = an.max_nominations
  INNER JOIN people p on a.imdb_name_id = p.imdb_name_id
  WHERE an.decade <> '1920-1929'
  LIMIT ${offset}, ${pageSize};
  `;
```

3.17 oscar actress

```
WITH oscar_movies AS (
    SELECT DISTINCT imdb_title_id, year_ceremony
    FROM oscar
),movie_counts AS (
    SELECT imdb_name_id, COUNT(DISTINCT imdb_title_id) AS total_movies
    FROM movie_people
    WHERE category = 'actress'
    GROUP BY imdb_name_id
),movie_ages AS (
  SELECT mp.imdb_name_id,
         p.name,
         p.photo_url,
         STR_TO_DATE(p.date_of_birth, '%m/%d/%Y') AS date_of_birth,
         MAX(m.year - YEAR(STR_TO_DATE(p.date_of_birth, '%m/%d/%Y'))) AS max_age,
         ROUND(AVG(m.year - YEAR(STR_TO_DATE(p.date_of_birth, '%m/%d/%Y')))) AS average_age
  FROM movie_people mp
         JOIN movie_data m ON m.imdb_title_id = mp.imdb_title_id
         JOIN people p ON p.imdb_name_id = mp.imdb_name_id
  WHERE mp.category = 'actress'
  GROUP BY mp.imdb_name_id, p.name, p.photo_url, p.date_of_birth
)
SELECT mp.imdb_name_id, ma.name, mc.total_movies, ma.photo_url,
       COUNT(*) AS oscar_freq,
       ROUND(MAX(om.year_ceremony - YEAR(ma.date_of_birth))) AS max_oscar_age,
       ROUND(AVG(m.avg_vote),1) AS avg_rating,
       ma.max_age,
       ma.average_age
FROM movie_people mp
       JOIN oscar_movies om ON om.imdb_title_id = mp.imdb_title_id
       JOIN movie_counts mc ON mc.imdb_name_id = mp.imdb_name_id
       JOIN movie_ages ma ON ma.imdb_name_id = mp.imdb_name_id
       JOIN movie_data m ON m.imdb_title_id = mp.imdb_title_id
WHERE mp.category = 'actress'
GROUP BY mp.imdb_name_id, ma.name, ma.max_age,ma.average_age,mc.total_movies, ma.photo_url
ORDER BY AVG(om.year_ceremony - YEAR(ma.date_of_birth)) DESC
  LIMIT ${offset}, ${pageSize};
```