Milestone 3

Credentials:
db_config = {
Username: shelly,
Host: database-1.cguuzxmvlnvo.us-east-2.rds.amazonaws.com,
Port: 3306,
Password: Renzilin10
}

1. Users will be able to search based on the videos' titles. The videos will be returned if the title is a match or the videos'n Akas is a match to the searched titles.

```
SELECT DISTINCT title
FROM
(
        (
        SELECT titleId AS tconst, title AS title
        FROM akas
        WHERE X LIKE 'title%'
        )
        UNION
        (
        SELECT tconst, primaryTitle AS title
        FROM movie_basics
        WHERE X LIKE 'title%')
)
```

2. Users will be able to see all the videos in different types (e.g., movies, TV series, TV episodes, etc.)

titleType = 'movie', startYear = 2022

```
SELECT tconst
FROM movie_basics
WHERE titleType = 'movie' AND startYear = 2022;
```

3. Users will be able to filter the videos in certain types based on isAdult, startYear, endYear, runtimeMinutes, genres, etc.

For example: isAdult = 1, startYear > 2020, runtimeMinutes >= 50

```
SELECT tconst
FROM movie_basics
WHERE isAdult = 1 AND startYear > 2020 AND runtimeMinutes >= 50
```

4. Users will be able to check the whole information of one certain video including StartYear, EndYear, all AKA titles, language, isAdult, type, etc.

```
SELECT B.StartYear, B.EndYear, A.title, A.language, B.isAdult, B.titleType AS type
FROM movie_basics B JOIN aka A
ON B.tconst = A.titleId
```

5. Users will be able to check the whole information of crews including actors and directors given a media name

INPUT(X) -> X some tconst

```
SELECT B.category, B.characters, C.primaryName, C.birthYear, C.deathYear
FROM movie_basics A
JOIN principals B ON A.tconst = B.tconst
JOIN name_basics C ON B.nconst = C.nconst
WHERE  A.tconst = 'tt1241118'
```

6. Top 5 rating movies in a specific year (e.g. 2021).

```
SELECT mb.primaryTitle, rt.averageRating, rt.numVotes
FROM movie_basics mb
JOIN rating rt ON mb.tconst = rt.tconst
WHERE mb.startYear = 2021 AND mb.titleType = 'movie'
ORDER BY rt.averageRating DESC
```

```
LIMIT 5
```

7. Users are able to get the TOP 1000 titles with the highest averageRating score for the category and the startYear(releaseYear) they search, with the principal crew and directors information provided.

```
WITH C AS(
SELECT B.*, A.averageRating, A.numVotes
FROM movie_basics B
INNER JOIN ratings A
ON B.tconst = A.tconst),

rs AS(
   SELECT C.tconst, C.primaryTitle, C.titleType, C.averageRating, C.startYear, Rank()
   OVER(
   Partition BY C.titleType, C.startYear
   ORDER BY C.averageRating DESC) AS rn
   FROM C)

SELECT rs.tconst, rs.primaryTitle, rs.titleType, rs.averageRating, rs.startYear, pr.nconst,
pr.category
FROM rs
JOIN principals pr
ON rs.tconst = pr.tconst
WHERE rs.rn <= 1000
```

8. Users are able to get medias, average rating, all the main actor and actress and their average ages given rating larger than 5. For calculating the actors and actresses' age, if there is a death year, the age will be deathYear - birthYear, otherwise it would be 2023 - birthYear.

```
WITH TMP_TABLE AS(
   SELECT T2.primaryTitle AS MediaTitle, D.averageRating, T2.People, T2.PersonName,
T2.Age
   FROM
```

```
      (SELECT T1.tconst, T1.primaryTitle, T1.isAdult, T1.category AS People, C.primaryName
AS PersonName,
            IF(C.deathYear IS NULL, 2023 - C.birthYear, C.deathYear - C.birthYear) AS age
      FROM
        (SELECT A.tconst, A.primaryTitle, A.isAdult, A.startYear, B.category, B.nconst
        FROM movie_basics A
        JOIN principals B ON A.tconst = B.tconst) T1
      JOIN name_basics C ON C.nconst = T1.nconst
      WHERE T1.category IN ('actress', 'actor') AND T1.isAdult = 0) T2
   JOIN ratings D ON D.tconst = T2.tconst
   WHERE D.averageRating > 5)
SELECT MediaTitle, averageRating, GROUP_CONCAT(PersonName SEPARATOR '; ') AS
MainActorActress, AVG(age) AS Avg_age
FROM TMP_TABLE
GROUP BY MediaTitle, averageRating;
```

9. Users are able to get the director information and title information for the most welcome (indicated by numVotes) in each titleType and language. This is very useful and common in real search operations (for example: users would like to know the most popular English movie and who is the corresponding director).

```
WITH AA AS(
SELECT mov.*, r.averageRating, r.numVotes
FROM movie_basics mov
JOIN ratings r on mov.tconst = r.tconst
ORDER BY r.numVotes DESC
),

BB AS (SELECT AA.*, a.language, a.region, a.title
FROM AA
JOIN akas a on AA.tconst = a.titleID
GROUP BY a.language, AA.tconst, AA.numVotes, AA.titleType
ORDER BY a.language, AA.titleType, AA.numVotes DESC)

SELECT BB.*, pr.nconst, pr.category, nb.primaryName, nb.birthYear, nb.deathYear,
nb.primaryProfession, nb.knownForTitles
```

```
FROM BB
JOIN principals pr ON BB.tconst = pr.tconst
JOIN name_basics nb ON pr.nconst = nb.nconst
WHERE pr.category = 'director'
```

10. Find actors and actress who played in shows with displaying area greater than or equal to 5 and all the movie/tv series names they've played with rating greater than 6

```
SELECT primaryName, GROUP_CONCAT(primaryTitle SEPARATOR '; ') AS all_shows
FROM (
WITH actorActress AS (
SELECT p.nconst, p.category, nb.primaryName
FROM principals p JOIN name_basics nb on p.nconst = nb.nconst
WHERE tconst IN(
SELECT titleId as tconst
FROM(
   SELECT titleId, COUNT(*) as numArea
   FROM akas
   GROUP BY titleId
   ) T1
WHERE numArea >= 5) AND category IN ('actress', 'actor'))

SELECT A.*, B.tconst, mb.primaryTitle, r.averageRating
FROM actorActress A JOIN principals B
ON A.nconst = B.nconst
JOIN movie_basics mb on B.tconst = mb.tconst
JOIN ratings r on mb.tconst = r.tconst
WHERE r.averageRating > 6) A
GROUP BY primaryName;
```