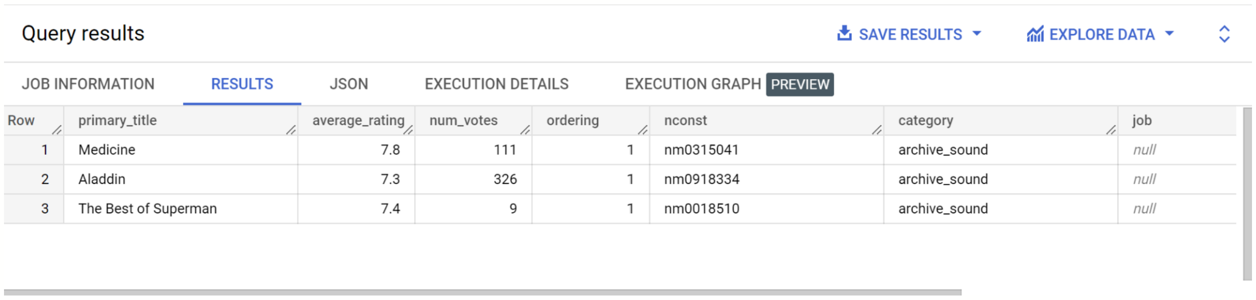
EXERCISE LEVEL 0

Explore the imdb dataset Create a sample query (with limit 3 rows joining 3 tables together correctly

Solution:

```
SELECT
tb.primary_title,
tr.* EXCEPT (tconst), -- we dont need tconst to appear again so we use
EXCEPT to avoid redundancies.
tp.* EXCEPT (tconst)
FROM
 `bigquery-public-data.imdb.title_basics` tb
INNER JOIN
 `bigquery-public-data.imdb.title_ratings` tr
ON
  tb.tconst=tr.tconst
INNER JOIN
  `bigquery-public-data.imdb.title_principals` tp
ON
  tr.tconst=tp.tconst
LIMIT 3
```



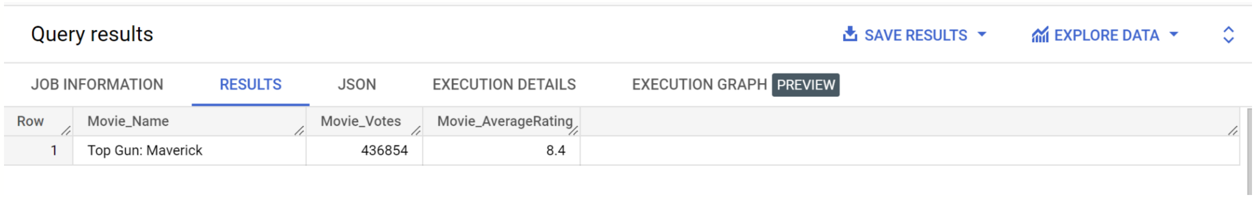
Row	primary_title	average_rating	num_votes	ordering	nconst	category	job
1	Medicine	7.8	111	1	nm0315041	archive_sound	*null*
2	Aladdin	7.3	326	1	nm0918334	archive_sound	*null*
3	The Best of Superman	7.4	9	1	nm0018510	archive_sound	*null*

EXERCISE LEVEL 1

1. What is the movie with the highest average_rating of the year 2022- in case of same give the one with more - with more than 100,000 ?

Solution:

```
SELECT
  primary_title AS Movie_Name,
  num_votes AS Movie_Votes,
  average_rating AS Movie_AverageRating
FROM `bigquery-public-data.imdb.title_basics` tb
INNER JOIN `bigquery-public-data.imdb.title_ratings` tr --To get common
elements only
  ON tb.tconst = tr.tconst
WHERE (start_year = 2022) AND (title_type LIKE '%Movie' or title_type LIKE
'%movie') AND (num_votes > 100000)
-- here we are considering both movie and tvMovie. Also we are applying the
condition for the year and num_votes.
ORDER BY 3 DESC, 2 DESC
LIMIT 1
```

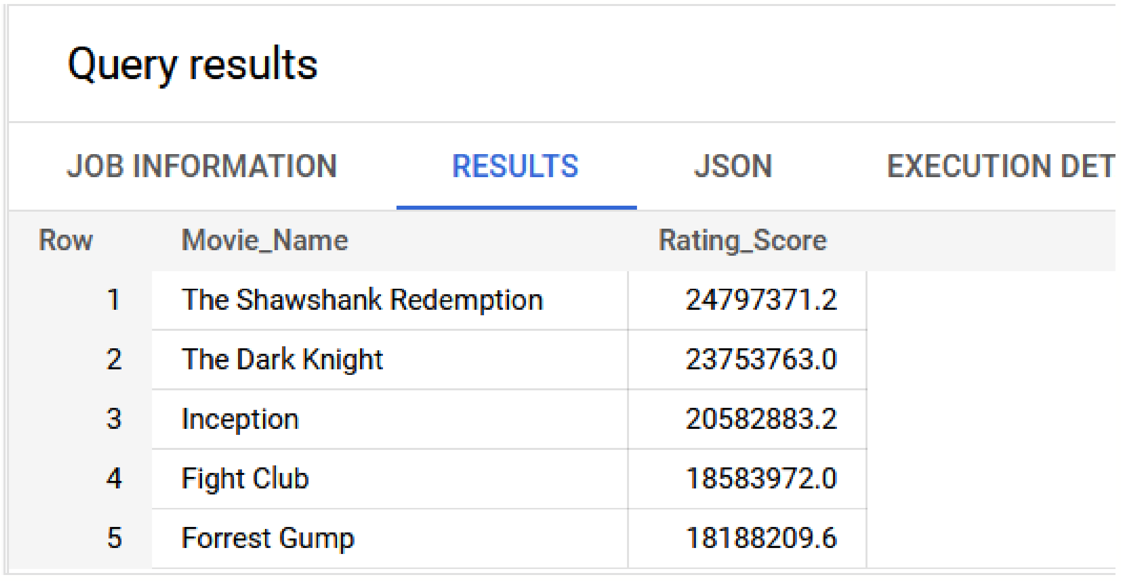


Row	Movie_Name	Movie_Votes	Movie_AverageRating
1	Top Gun: Maverick	436854	8.4

2. In order to have 1 metric to find the best movies, we will use average_rating * num_votes that we will call rating_score - What are the top 5 movies with the highest rating_score ever ?

Solution:

```
SELECT
primary_title as Movie_Name,
ROUND(average_rating *num_votes,2) AS Rating_Score -- to avoid long
decimal values
FROM `bigquery-public-data.imdb.title_basics` tb
INNER JOIN `bigquery-public-data.imdb.title_ratings` tr
ON tb.tconst = tr.tconst
where title_type LIKE '%Movie' or title_type LIKE '%movie'
---considering both movie and tvMovie
ORDER BY 2 DESC  --- To find the highest value based on Rating_Score
LIMIT 5
```

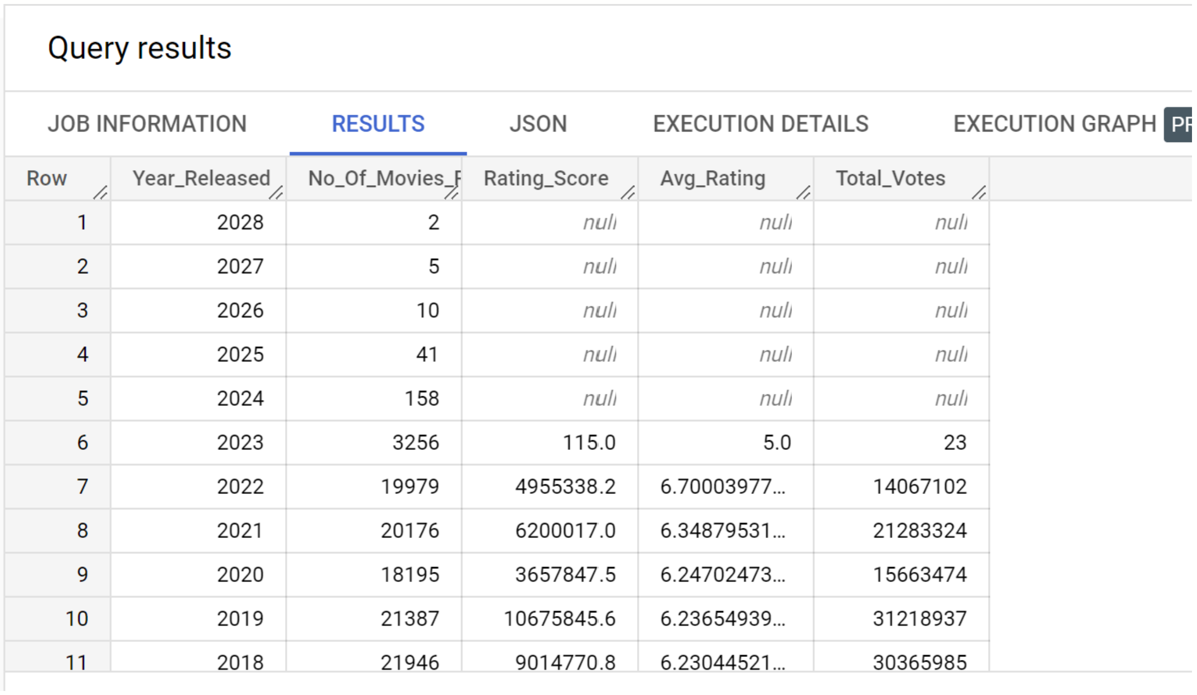


Row	Movie_Name	Rating_Score
1	The Shawshank Redemption	24797371.2
2	The Dark Knight	23753763.0
3	Inception	20582883.2
4	Fight Club	18583972.0
5	Forrest Gump	18188209.6

3. Get a query to have for each year, the number of movies released, the highest rating_score for the year and the average of average_rating for the year as well as the sum of num_votes - order by year descending

Solution:

```
SELECT
start_year as Year_Released,
COUNT(DISTINCT tb.tconst ) as No_Of_Movies_Released,-- using DISTINCT to
avoid duplicate values
MAX(ROUND((average_rating*num_votes),2)) as Rating_Score,--- to get highest
rating score for the year
AVG(average_rating) as Avg_Rating,
SUM(num_votes) as Total_Votes
FROM `bigquery-public-data.imdb.title_basics` tb
LEFT JOIN
`bigquery-public-data.imdb.title_ratings` tr
  ON tr.tconst = tb.tconst
WHERE title_type LIKE '%movie'OR title_type LIKE '%Movie'
GROUP BY start_year--- to find the data per year
ORDER BY 1 DESC---- as per the question descending the year
```


Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS | EXECUTION GRAPH

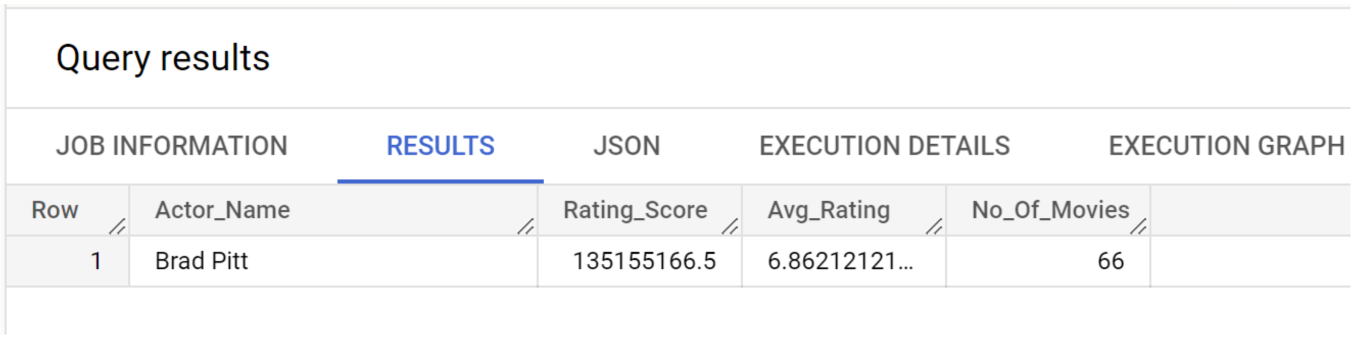
Row	Year_Released	No_Of_Movies_R	Rating_Score	Avg_Rating	Total_Votes
1	2028	2	null	null	null
2	2027	5	null	null	null
3	2026	10	null	null	null
4	2025	41	null	null	null
5	2024	158	null	null	null
6	2023	3256	115.0	5.0	23
7	2022	19979	4955338.2	6.70003977...	14067102
8	2021	20176	6200017.0	6.34879531...	21283324
9	2020	18195	3657847.5	6.24702473...	15663474
10	2019	21387	10675845.6	6.23654939...	31218937
11	2018	21946	9014770.8	6.23044521...	30365985

EXERCISE LEVEL 2

4. Who is the actor who played in movies that has the biggest sum of rating_score - provide also his average of average_rating and his number of movies ??

Solution:

```
SELECT
  primary_name as Actor_Name,
  SUM(ROUND((average_rating*num_votes),2)) as Rating_Score,
  AVG(average_rating) as Avg_Rating,
  COUNT(DISTINCT primary_title) as No_Of_Movies
FROM
  `bigquery-public-data.imdb.title_basics` tb
INNER JOIN
  `bigquery-public-data.imdb.title_ratings` tr
ON
  tb.tconst = tr.tconst
INNER JOIN
  `bigquery-public-data.imdb.title_principals` tp
ON
  tr.tconst=tp.tconst
INNER JOIN
  `bigquery-public-data.imdb.name_basics` nb
ON
  nb.nconst=tp.nconst
WHERE (title_type LIKE '%Movie' or title_type LIKE '%movie' ) AND
primary_profession LIKE '%actor%'---applying filter for actor and we get
count of movies as 66 since we consider tvMovie and movie
GROUP BY 1
ORDER BY 2 DESC
LIMIT 1
```



Query results

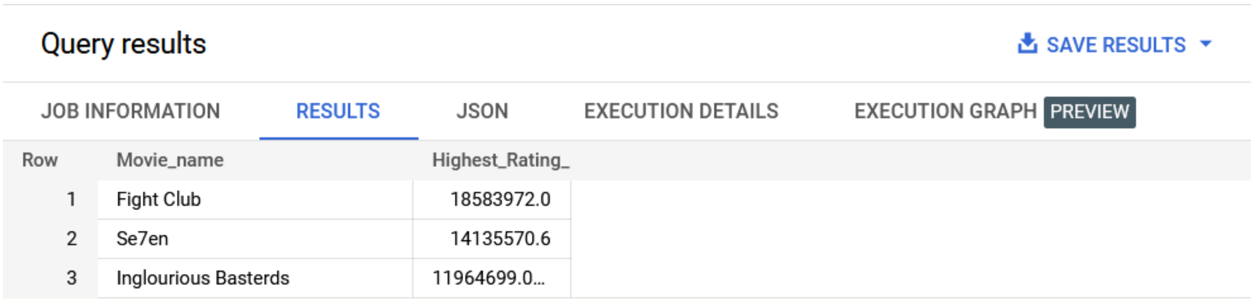
Row	Actor_Name	Rating_Score	Avg_Rating	No_Of_Movies
1	Brad Pitt	135155166.5	6.86212121...	66

5. What are the top 3 movies with highest rating_score for the actor found above

Solution:

5.1. Manual method

```
SELECT
  primary_title AS Movie_name,
  sum(average_rating * num_votes) AS  Highest_Rating_Score
FROM
  `bigquery-public-data.imdb.title_basics` tb
LEFT JOIN
  `bigquery-public-data.imdb.title_ratings` tr
ON
  tb.tconst = tr.tconst
LEFT JOIN
  `bigquery-public-data.imdb.title_principals` tp
ON
  tr.tconst=tp.tconst
LEFT JOIN
  `bigquery-public-data.imdb.name_basics` nb
ON
  nb.nconst=tp.nconst WHERE  primary_name = "Brad Pitt" AND (title_type
LIKE '%movie' or title_type LIKE '%Movie' )-- Here we are manually adding
Brad pitt based on answer in Question 4
GROUP BY 1
ORDER BY 2 DESC
LIMIT 3
```


Query results

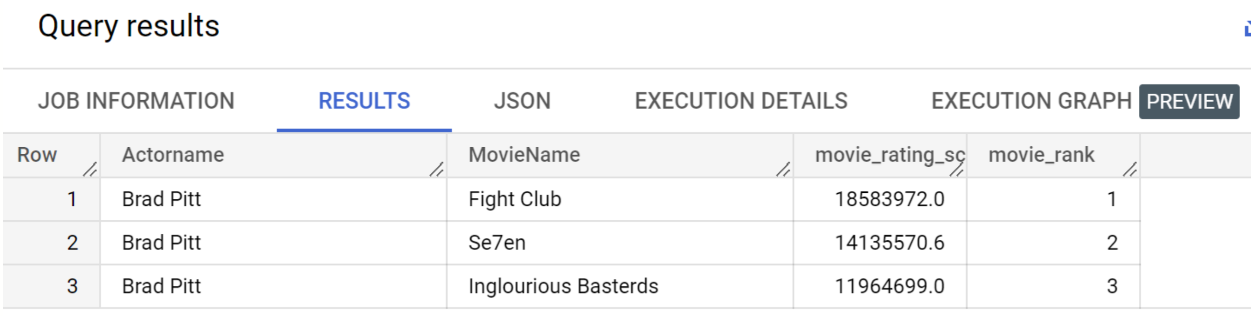
SAVE RESULTS

JOB INFORMATION RESULTS JSON EXECUTION DETAILS EXECUTION GRAPH PREVIEW

Row	Movie_name	Highest_Rating_
1	Fight Club	18583972.0
2	Se7en	14135570.6
3	Inglourious Basterds	11964699.0...

5.2 Partition Rank Method

```
SELECT
  primary_name AS Actorname,
  primary_title AS MovieName,
  ROUND(movie_rating_score,2) AS movie_rating_score,
  RANK()OVER(PARTITION BY primary_name ORDER BY movie_rating_score DESC)
movie_rank-- here we applying partition method where we grouping
movie_rating_score field for each actor and displaying it

FROM(
    SELECT
      primary_name,
      primary_title,
      average_rating * num_votes AS movie_rating_score
    FROM `bigquery-public-data.imdb.title_principals` tp
    JOIN `bigquery-public-data.imdb.title_basics` tb
      ON tp.tconst = tb.tconst
    JOIN `bigquery-public-data.imdb.name_basics` nb
      ON tp.nconst = nb.nconst
    JOIN `bigquery-public-data.imdb.title_ratings` tr
      ON tp.tconst = tr.tconst
    WHERE (title_type LIKE '%movie'OR title_type LIKE '%Movie') AND
primary_profession LIKE '%actor%'
    ) rating_table
ORDER BY SUM(movie_rating_score) OVER (PARTITION BY primary_name) DESC--
ordering based on sum of movie rating score for each actor
LIMIT 3
```


Query results

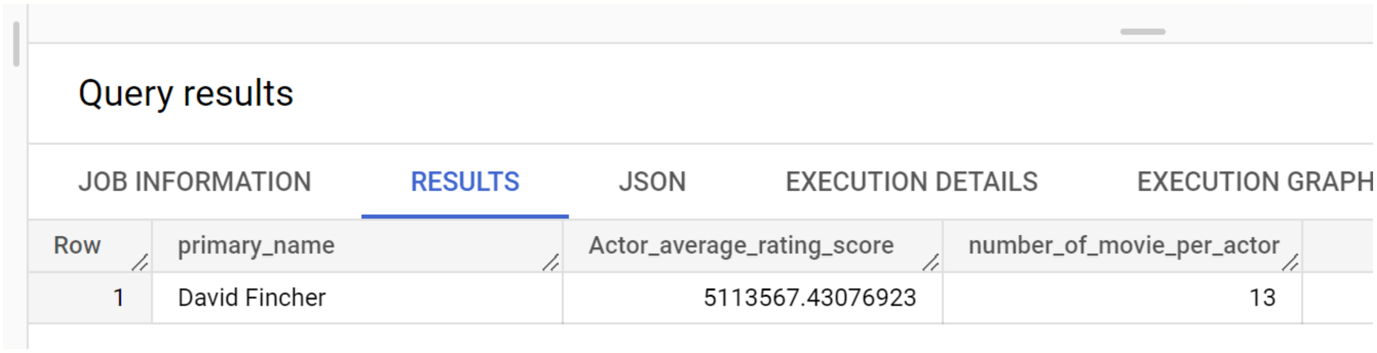
JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS | EXECUTION GRAPH PREVIEW

Row	Actorname	MovieName	movie_rating_sc	movie_rank
1	Brad Pitt	Fight Club	18583972.0	1
2	Brad Pitt	Se7en	14135570.6	2
3	Brad Pitt	Inglourious Basterds	11964699.0	3

6. Who is the actor who played in at least 5 movies with the highest average rating_score per movie (what is his average rating_score)?

Solution:

```
SELECT
  primary_name,
  (Sum_average_rating/number_of_movie_per_actor) AS Actor_average_rating_score,
  number_of_movie_per_actor
FROM --- here we are finding actor name and his average score
 (SELECT----- creating subquery TO get actor who played IN AT least 5 movies
with the highest average rating_score per movie
    primary_name,
    COUNT(DISTINCT tb.primary_title) AS number_of_movie_per_actor,
    AVG(average_rating * num_votes) AS AVG_rating_actor,
    SUM(average_rating * num_votes) AS Sum_average_rating
  FROM `bigquery-public-data.imdb.title_basics` tb
  JOIN `bigquery-public-data.imdb.title_principals` tp
    ON tb.tconst = tp.tconst
  JOIN `bigquery-public-data.imdb.name_basics` nb
    ON tp.nconst = nb.nconst
  JOIN `bigquery-public-data.imdb.title_ratings` tr
    ON tb.tconst = tr.tconst
  WHERE (title_type LIKE "%movie" OR title_type LIKE '%Movie')AND
primary_profession LIKE '%actor%'
  GROUP BY 1
  HAVING number_of_movie_per_actor >= 5
  ORDER BY 3 DESC
  LIMIT 1)
```



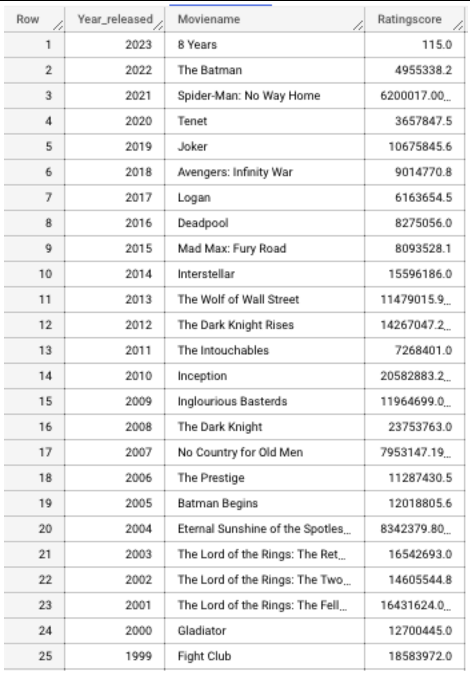
Row	primary_name	Actor_average_rating_score	number_of_movie_per_actor
1	David Fincher	5113567.43076923	13

EXERCISE LEVEL 3

7. Create a Query to get the top movie (highest rating_score) for each year

Solution:

```sql
SELECT DISTINCT start_year AS Year_released ,primary_title AS
Moviename,(average_rating*num_votes) AS Ratingscore FROM
`bigquery-public-data.imdb.title_basics` tb
JOIN `bigquery-public-data.imdb.title_ratings` tr
 ON tb.tconst = tr.tconst
JOIN
(SELECT DISTINCT
start_year as Year_Released,
MAX(average_rating*num_votes) as Rating_Score
FROM `bigquery-public-data.imdb.title_basics` tb
JOIN `bigquery-public-data.imdb.title_ratings` tr
 ON tb.tconst = tr.tconst
WHERE title_type LIKE '%Movie' or title_type LIKE '%movie'
GROUP BY start_year) Z
ON Z.Rating_Score = (tr.average_rating*tr.num_votes) AND Z.Year_Released =
tb.start_year
ORDER BY 1 DESC
```

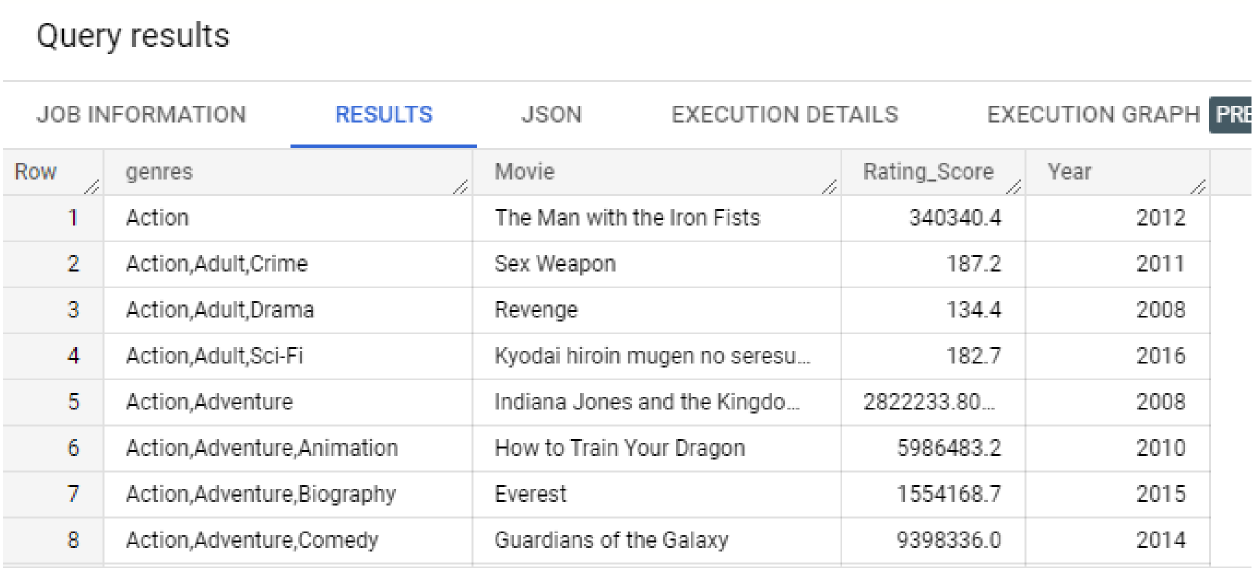


Row	Year_released	Moviename	Ratingscore
1	2023	8 Years	115.0
2	2022	The Batman	4955338.2
3	2021	Spider-Man: No Way Home	6200017.00...
4	2020	Tenet	3657847.5
5	2019	Joker	10675845.6
6	2018	Avengers: Infinity War	9014770.8
7	2017	Logan	6163654.5
8	2016	Deadpool	8275056.0
9	2015	Mad Max: Fury Road	8093528.1
10	2014	Interstellar	15596186.0
11	2013	The Wolf of Wall Street	11479015.9...
12	2012	The Dark Knight Rises	14267047.2...
13	2011	The Intouchables	7268401.0
14	2010	Inception	20582883.2...
15	2009	Inglourious Basterds	11964699.0...
16	2008	The Dark Knight	23753763.0
17	2007	No Country for Old Men	7953147.19...
18	2006	The Prestige	11287430.5
19	2005	Batman Begins	12018805.6
20	2004	Eternal Sunshine of the Spotles...	8342379.80...
21	2003	The Lord of the Rings: The Ret...	16542693.0
22	2002	The Lord of the Rings: The Two...	14605544.8
23	2001	The Lord of the Rings: The Fell...	16431624.0...
24	2000	Gladiator	12700445.0
25	1999	Fight Club	18583972.0

8. For each Movie Genre, for the release since 2000, give the movie title with the highest rating_score

Solution:

```
SELECT
  tb.genres,
  primary_title AS Movie,
 (average_rating*num_votes) AS Rating_Score,
  start_year AS Year
FROM `bigquery-public-data.imdb.title_basics` tb
JOIN
  `bigquery-public-data.imdb.title_ratings` tr
ON
  tb.tconst = tr.tconst
JOIN(
  SELECT
    genres,
    MAX(average_rating*num_votes) AS Rating_score
  FROM
    `bigquery-public-data.imdb.title_basics` tb
  JOIN
    `bigquery-public-data.imdb.title_ratings` tr
  ON
    tb.tconst = tr.tconst
  WHERE
    title_type LIKE '%Movie'
    OR title_type LIKE '%movie'
    AND start_year >= 2000
  GROUP BY
    genres) highest_ratingscore
ON
  highest_ratingscore.Rating_score = (tr.average_rating*tr.num_votes) AND
highest_ratingscore.genres = tb.genres
WHERE
    title_type LIKE '%Movie'
    OR title_type LIKE '%movie'
    AND start_year >= 2000
GROUP BY
  1,2,3,4
ORDER BY genres
```


Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS | EXECUTION GRAPH | PRE

Row	genres	Movie	Rating_Score	Year
1	Action	The Man with the Iron Fists	340340.4	2012
2	Action,Adult,Crime	Sex Weapon	187.2	2011
3	Action,Adult,Drama	Revenge	134.4	2008
4	Action,Adult,Sci-Fi	Kyodai hiroin mugen no seresu...	182.7	2016
5	Action,Adventure	Indiana Jones and the Kingdo...	2822233.80...	2008
6	Action,Adventure,Animation	How to Train Your Dragon	5986483.2	2010
7	Action,Adventure,Biography	Everest	1554168.7	2015
8	Action,Adventure,Comedy	Guardians of the Galaxy	9398336.0	2014

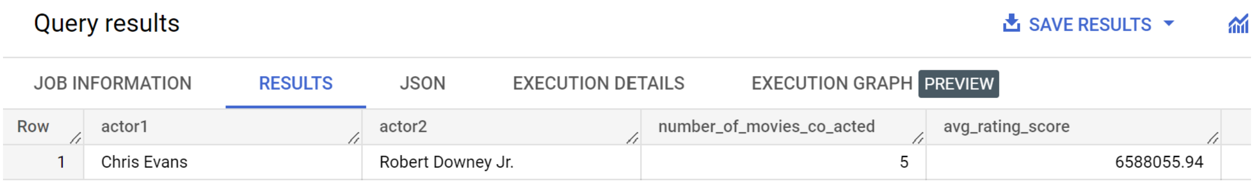
EXERCISE LEVEL 4

8. Duo? Find the actors duo that get the highest average rating_score per movie together in ordering 1 or 2, with at least 4 movies together

Solution:

```
WITH actor_selection AS --- as first step  we are creating a table with
necessary fields and limiting to the ones with ordering <= 2

(SELECT
  tp.tconst,
  primary_title,
  ordering,
  primary_name
  FROM `bigquery-public-data.imdb.title_principals` tp
  JOIN `bigquery-public-data.imdb.title_basics` tb
    ON tp.tconst = tb.tconst
  JOIN `bigquery-public-data.imdb.name_basics` nb
    ON tp.nconst = nb.nconst
  WHERE ordering <= 2 AND (title_type LIKE '%movie' OR title_type LIKE
'%Movie') AND primary_profession LIKE '%actor%')

SELECT ----  doing self join of the above table for 2 actors
```

```
  se.primary_name AS actor1,
  se1.primary_name AS actor2,
  COUNT(*) AS number_of_movies_co_acted,
  AVG(average_rating * num_votes) AS avg_rating_score
FROM actor_selection se
JOIN actor_selection se1
  ON se.tconst = se1.tconst
JOIN `bigquery-public-data.imdb.title_ratings` tr
  ON se.tconst = tr.tconst
WHERE se.primary_title = se1.primary_title --checking the condition where
they acted in same movie
GROUP BY 1,2
HAVING COUNT(*) >= 4 --we are applying count of movies >=4
ORDER BY 4 DESC -- inorder to find highest average rating score
Limit 1
```



Row	actor1	actor2	number_of_movies_co_acted	avg_rating_score
1	Chris Evans	Robert Downey Jr.	5	6588055.94

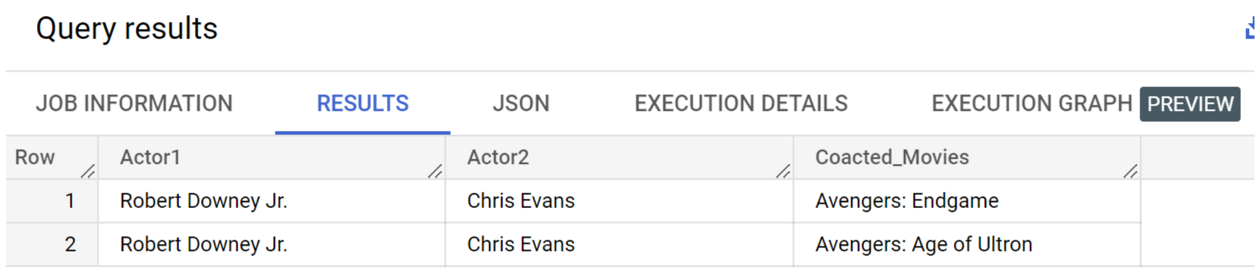
9.Extra - Bonus question if you manage a smart way to show 2 example of their movies together and then to limit the cases where the movies titles very similar looking like they belong to a sequel…

Solution:

```
 WITH actor_selection AS --- as first step  we are creating a table with necessary
fields and limiting to the ones with ordering <= 2

(SELECT

  tp.tconst,

  primary_title,

  ordering,

  primary_name,

  category

  FROM `bigquery-public-data.imdb.title_principals` tp

  JOIN `bigquery-public-data.imdb.title_basics` tb

    ON tp.tconst = tb.tconst

  JOIN `bigquery-public-data.imdb.name_basics` nb

    ON tp.nconst = nb.nconst

  WHERE ordering <= 2 AND (title_type LIKE '%movie' OR title_type LIKE '%Movie') AND
primary_profession LIKE '%actor%')

 SELECT ----  doing self join of the above table for 2 actors

  z.primary_name  Actor1,

  z1.primary_name Actor2,

  z.primary_title Coacted_Movies
```

```
FROM actor_selection z

JOIN actor_selection z1

  ON z.tconst = z1.tconst

 JOIN

(SELECT ----  doing self join of the above table for 2 actors

  se.primary_name AS actor1,

  se1.primary_name AS actor2,

  COUNT(*) AS number_of_movies_co_acted,

  AVG(average_rating * num_votes) AS avg_rating_score

FROM actor_selection se

JOIN actor_selection se1

  ON se.tconst = se1.tconst

JOIN `bigquery-public-data.imdb.title_ratings` tr

  ON se.tconst = tr.tconst

WHERE se.primary_title = se1.primary_title --checking the condition where they acted
in same movie

GROUP BY 1,2

HAVING COUNT(*) >= 4 --we are applying count of movies >=4

ORDER BY 4 DESC -- inorder to find highest average rating score

Limit 1)

ON

actor1= z.primary_name AND actor2 =z1.primary_name
```

```
WHERE z.primary_title LIKE "%:%" LIMIT 2  ---- to show the sequel conditions most
sequels have":" so we put that logic
```



Row	Actor1	Actor2	Coacted_Movies
1	Robert Downey Jr.	Chris Evans	Avengers: Endgame
2	Robert Downey Jr.	Chris Evans	Avengers: Age of Ultron