# GOOGLE BIG QUERY SQL PROJECT

## 0) Create a sample query (with limit 3 rows joining 3 tables together correctly

Ans:

```
SELECT  *

FROM `bigquery-public-data.imdb.title_basics` tb
join  `bigquery-public-data.imdb.title_ratings` tr
ON tb.tconst=tr.tconst
join `bigquery-public-data.imdb.title_principals` tp
ON tr.tconst=tp.tconst
LIMIT 3
```

2nd way:

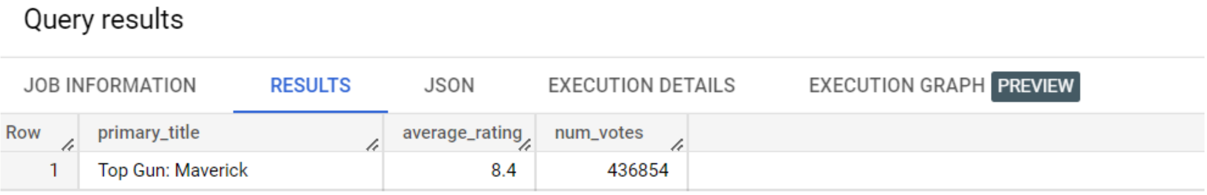
```
SELECT
tb.primary_title,
tr.* EXCEPT(tconst),
tp.* EXCEPT(TCONST)
FROM `bigquery-public-data.imdb.title_basics` tb
join  `bigquery-public-data.imdb.title_ratings` tr
ON tb.tconst=tr.tconst
join `bigquery-public-data.imdb.title_principals` tp
ON tr.tconst=tp.tconst
LIMIT 3
```

3RD WAY WITH TCOSTANTS:

```
SELECT
tb.primary_title,
tr.* ,
tp.*
FROM `bigquery-public-data.imdb.title_basics` tb
join  `bigquery-public-data.imdb.title_ratings` tr
ON tb.tconst=tr.tconst
join `bigquery-public-data.imdb.title_principals` tp
ON tr.tconst=tp.tconst
LIMIT 3
```

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

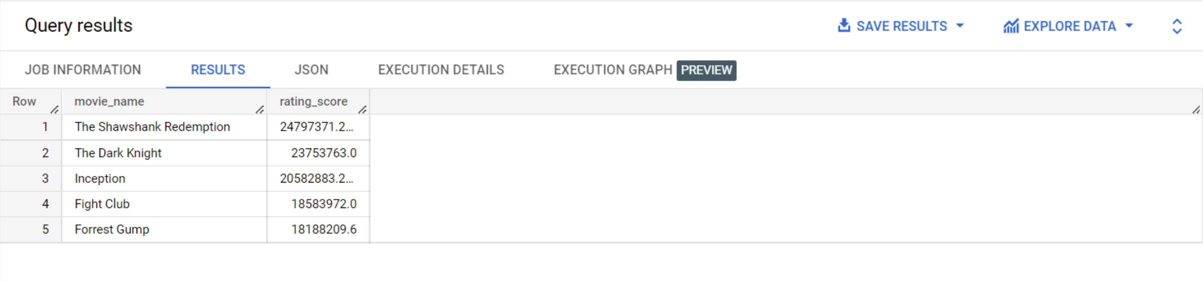
```
SELECT

      primary_title,
      average_rating,
      num_votes
      FROM `bigquery-public-data.imdb.title_basics` tb
       join  `bigquery-public-data.imdb.title_ratings` tr
      ON tb.tconst=tr.tconst
      WHERE start_year =2022
      and num_votes>100000
      and (title_type LIKE '%Movie' or title_type LIKE '%movie')
      order by 2 DESC
      LIMIT 1
```


Query results

| Row | primary_title | average_rating | num_votes |
|---|---|---|---|
| 1 | Top Gun: Maverick | 8.4 | 436854 |

## 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 ?

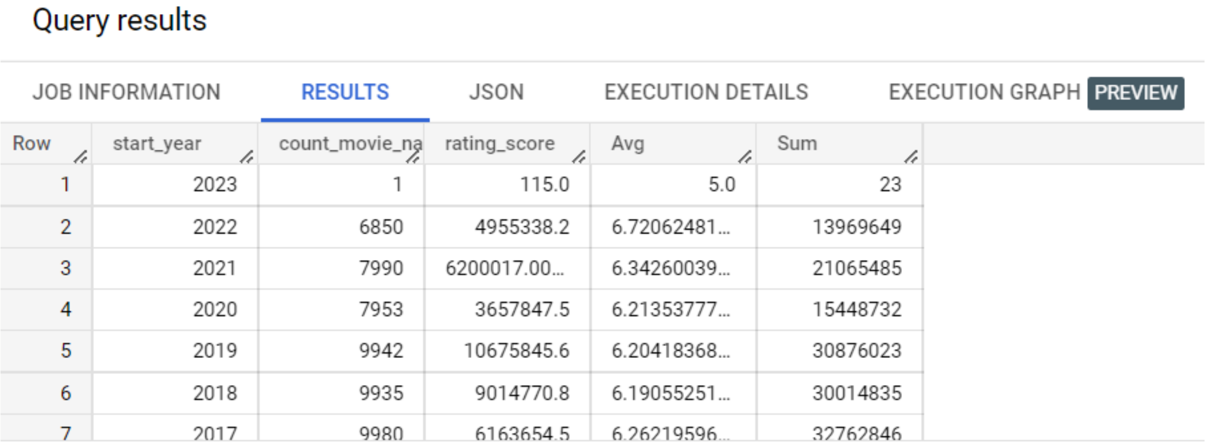
```
SELECT
primary_title as movie_name,
average_rating *num_votes as rating_score
FROM `bigquery-public-data.imdb.title_basics` tb
join  `bigquery-public-data.imdb.title_ratings` tr
ON tb.tconst=tr.tconst
WHERE title_type LIKE '%Movie' or title_type LIKE '%movie'
order by 2 desc
limit 5
```



| 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.

```
SELECT

distinct start_year,

count(distinct primary_title) as count_movie_name,
MAX(average_rating * num_votes) as rating_score,
Avg(average_rating) as Avg,
Sum(num_votes) as Sum
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 1
order by 1 desc
```

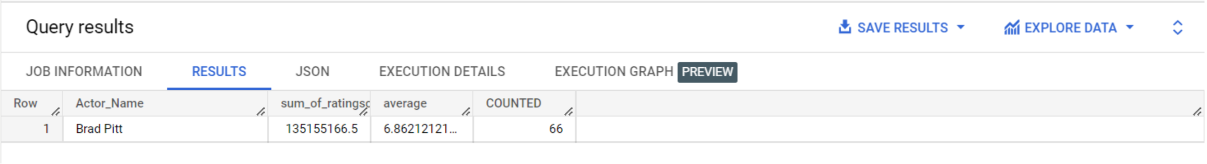


| Row | start_year | count_movie_na | rating_score | Avg | Sum |
|---|---|---|---|---|---|
| 1 | 2023 | 1 | 115.0 | 5.0 | 23 |
| 2 | 2022 | 6850 | 4955338.2 | 6.72062481… | 13969649 |
| 3 | 2021 | 7990 | 6200017.00… | 6.34260039… | 21065485 |
| 4 | 2020 | 7953 | 3657847.5 | 6.21353777… | 15448732 |
| 5 | 2019 | 9942 | 10675845.6 | 6.20418368… | 30876023 |
| 6 | 2018 | 9935 | 9014770.8 | 6.19055251… | 30014835 |
| 7 | 2017 | 9980 | 6163654.5 | 6.26219596… | 32762846 |

## 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 ??

Re: there’s 2 types of movies: movie & TV movies

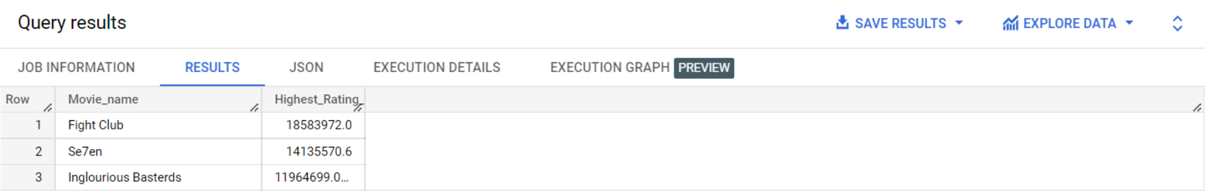
```
Select

primary_name As Actor_Name,
sum(num_votes * average_rating) as sum_of_ratingscore,
Avg(average_rating) as average,
COUNT(primary_title) as COUNTED
FROM `bigquery-public-data.imdb.title_basics` tb
JOIN `bigquery-public-data.imdb.title_ratings` tr
  ON tb.tconst = tr.tconst
JOIN `bigquery-public-data.imdb.title_principals` tp
  ON tr.tconst = tp.tconst
JOIN `bigquery-public-data.imdb.name_basics` nb
  ON tp.nconst = nb.nconst
where (title_type like "%movie" or title_type like "%Movie")  and primary_profession like"%
actor%"
GROUP BY 1
ORDER BY 2 DESC
limit 1
```



| Row | Actor_Name | sum_of_ratingsc | average | COUNTED |
|---|---|---|---|---|
| 1 | Brad Pitt | 135155166.5 | 6.86212121… | 66 |

## 5) What are the top 3 movies with highest rating_score for the actor found above *** there is one easy way more manual and one more complex but where you can answer in 1 single query *** bonus point for the complex version but probably level3 ;)

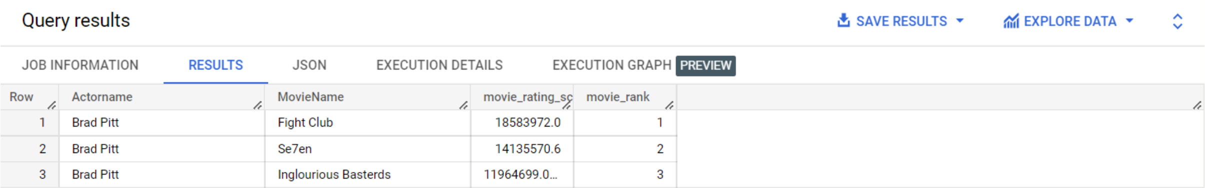
```
SELECT
primary_title AS Movie_name,
(average_rating * num_votes) AS  Highest_Rating_Score
FROM
  `bigquery-public-data.imdb.title_basics` tb
LEFT JOIN
  `bigquery-public-data.imdb.title_ratings` tr
ON
  tb.tconst = tr.tconst
LEFT JOIN
  `bigquery-public-data.imdb.title_principals` tp
ON
  tr.tconst=tp.tconst
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
  ORDER BY 2 DESC LIMIT 3
```



| Row | Movie_name | Highest_Rating |
|---|---|---|
| 1 | Fight Club | 18583972.0 |
| 2 | Se7en | 14135570.6 |
| 3 | Inglourious Basterds | 11964699.0... |

## 5) Alternate Query

```
SELECT
  primary_name AS Actorname,
  primary_title AS MovieName,
  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
```



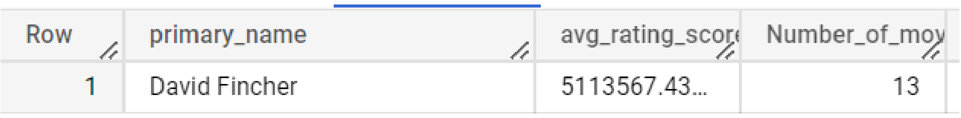
| Row | Actorname | MovieName | movie_rating_sc | movie_rank |
|---|---|---|---|---|
| 1 | Brad Pitt | Fight Club | 18583972.0 | 1 |
| 2 | Brad Pitt | Se7en | 14135570.6 | 2 |
| 3 | Brad Pitt | Inglourious Basterds | 11964699.0... | 3 |

## 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 )?

```
SELECT
  primary_name,
  AVG(num_votes * average_rating) AS avg_rating_score,
  COUNT(primary_title) AS Number_of_movies
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_movies >= 5
ORDER BY 2 DESC
LIMIT 1
```

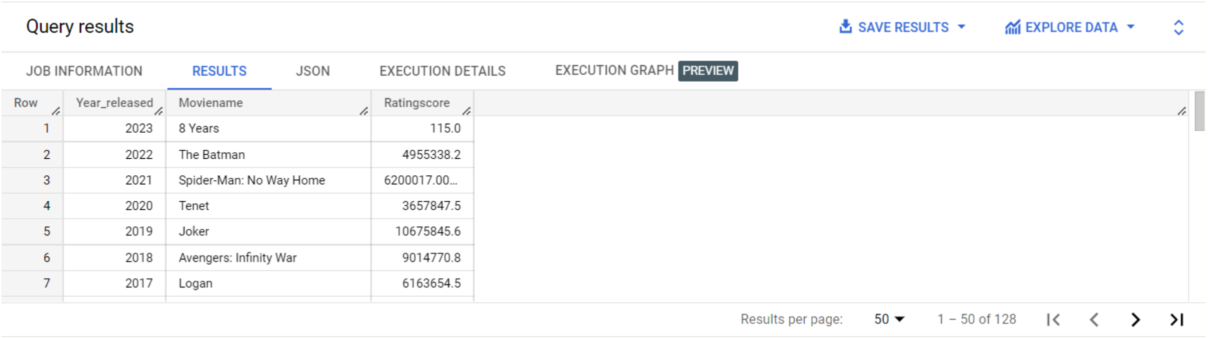
Output



| Row | primary_name | avg_rating_scor | Number_of_mov |
|---|---|---|---|
| 1 | David Fincher | 5113567.43... | 13 |

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

```
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
```

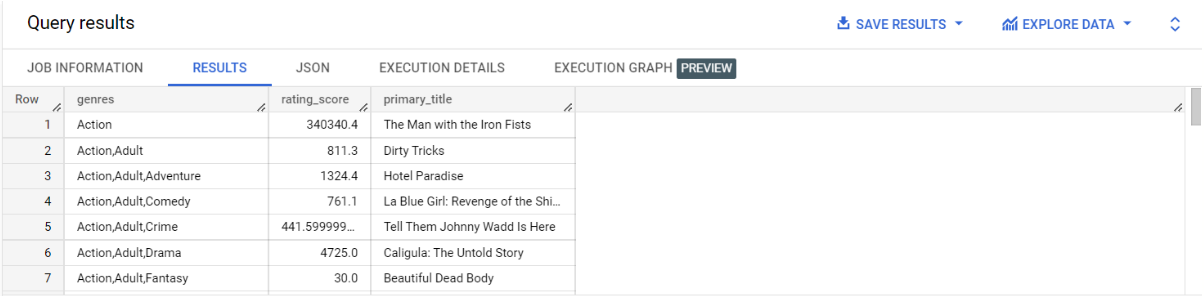


| Row | Year_released | Moviename | Ratingscore |
|---|---|---|---|
| 1 | 2023 | 8 Years | 115.0 |
| 2 | 2022 | The Batman | 4955338.2 |
| 3 | 2021 | Spider-Man: No Way Home | 6200017.00... |
| 4 | 2020 | Tenet | 3657847.5 |
| 5 | 2019 | Joker | 10675845.6 |
| 6 | 2018 | Avengers: Infinity War | 9014770.8 |
| 7 | 2017 | Logan | 6163654.5 |

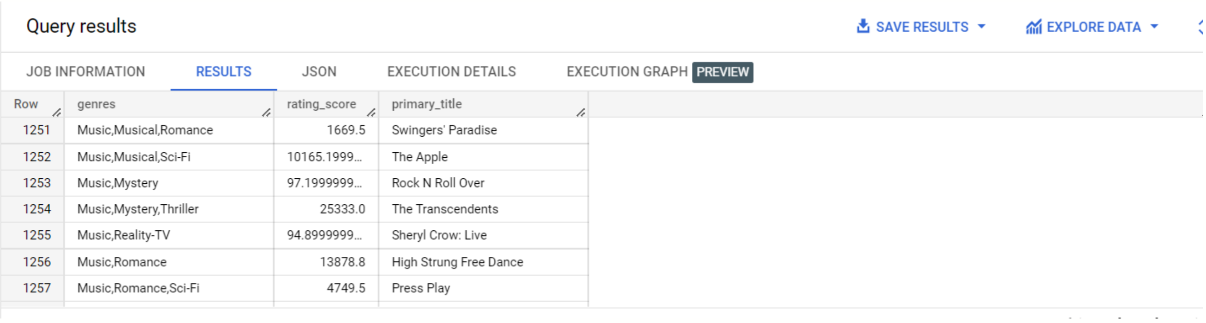
Results per page: 50 1 – 50 of 128

## 8) For each Movie Genre, for the release since 2000, give the movie title with the highest rating_score *** Same logic as above but you will use the Genre instead of the Year in the subquery and the join condition

```
SELECT
tb.genres
,tr.average_rating * tr.num_votes AS rating_score
,tb.primary_title

FROM `bigquery-public-data.imdb.title_basics` tb
JOIN `bigquery-public-data.imdb.title_ratings` tr
ON tb.tconst = tr.tconst
JOIN

  (SELECT
  tb.genres
  ,MAX(tr.average_rating * tr.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 tb.title_type LIKE "%movie%" OR title_type LIKE '%Movie'
  AND tb.start_year >= 2000
  GROUP BY 1) tmpYearMaxRatingScore
  ON tmpYearMaxRatingScore.genres = tb.genres
 AND tmpYearMaxRatingScore.rating_score = tr.average_rating * tr.num_votes

ORDER BY 1
```



| Row | genres | rating_score | primary_title |
|---|---|---|---|
| 1 | Action | 340340.4 | The Man with the Iron Fists |
| 2 | Action,Adult | 811.3 | Dirty Tricks |
| 3 | Action,Adult,Adventure | 1324.4 | Hotel Paradise |
| 4 | Action,Adult,Comedy | 761.1 | La Blue Girl: Revenge of the Shi… |
| 5 | Action,Adult,Crime | 441.599999… | Tell Them Johnny Wadd Is Here |
| 6 | Action,Adult,Drama | 4725.0 | Caligula: The Untold Story |
| 7 | Action,Adult,Fantasy | 30.0 | Beautiful Dead Body |



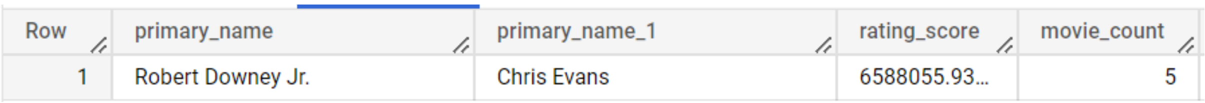
| Row | genres | rating_score | primary_title |
|---|---|---|---|
| 1251 | Music,Musical,Romance | 1669.5 | Swingers' Paradise |
| 1252 | Music,Musical,Sci-Fi | 10165.1999… | The Apple |
| 1253 | Music,Mystery | 97.1999999… | Rock N Roll Over |
| 1254 | Music,Mystery,Thriller | 25333.0 | The Transcendents |
| 1255 | Music,Reality-TV | 94.8999999… | Sheryl Crow: Live |
| 1256 | Music,Romance | 13878.8 | High Strung Free Dance |
| 1257 | Music,Romance,Sci-Fi | 4749.5 | Press Play |

## 08) 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

```
WITH actor_duo AS
(SELECT
primary_name, tb.primary_title, tp.tconst
FROM `bigquery-public-data.imdb.name_basics` nb
JOIN
`bigquery-public-data.imdb.title_principals` tp
ON nb.nconst = tp.nconst
JOIN `bigquery-public-data.imdb.title_basics`tb
ON tp.tconst = tb.tconst
WHERE tp.ordering<=2
AND title_type LIKE '%movie%'
AND primary_profession LIKE 'actor%')
SELECT
ad.primary_name,
ad1.primary_name,
AVG(average_rating*num_votes) AS rating_score,
COUNT(ad.primary_title) AS movie_count
FROM actor_duo ad
JOIN actor_duo ad1
ON ad.primary_name > ad1.primary_name
AND
ad.primary_title = ad1.primary_title
JOIN `bigquery-public-data.imdb.title_ratings`tr
ON ad.tconst = tr.tconst
GROUP BY 1,2
```

```
HAVING COUNT(ad.primary_title) >=4
ORDER BY rating_score DESC
LIMIT 1
```

Output:



| Row | primary_name | primary_name_1 | rating_score | movie_count |
|---|---|---|---|---|
| 1 | Robert Downey Jr. | Chris Evans | 6588055.93... | 5 |