ETL Project Writeup - Group 1

Alex Arnold, Juveriya Baig, Samantha Lane, Angie Tran

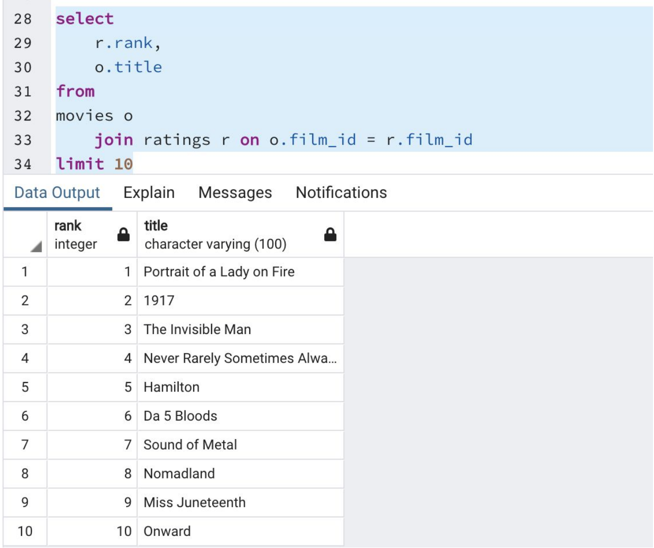
Extract: For this project, we chose to extract data from Rotten Tomatoes, specifically a table of the Top 100 Movies of 2020. With this data, we ran these movies through the OMDB API to pull relevant information. We scraped the Rotten Tomatoes website using Pandas and created a table. We then used the OMDB API to create a more detailed table using the information from Rotten Tomatoes. This new table was exported into a CSV.

Transform: In order to clean our data, a variety of things were required. We first cleaned our data from Rotten Tomatoes. We removed parenthesis' from movie titles to give the titles more consistency and then removed the percent sign from ratings to make the column an integer. The rank column was a float due to the numbers being decimals so we converted that column into integers as well.

Next, we transformed the data collected from the OMDB API. We initialized the lists of columns we wanted to add to our cleaned Rotten Tomatoes table. We then created a loop that would parse the responses and append the information to our list of columns. If information was missing from the API, we used a Try/Except to move onto the next movie. Next, columns were renamed and data was exported to a CSV. The final thing we did to our API data was to go through columns we wanted to write queries for in pgAdmin 4. The columns we chose were Genre, Language, Actor, Country, Director, and Movies. From these columns, we created queries to match our ERD diagram and used pgAdmin 4 to extract specific insights we wanted from our data.

Load: We also created a database that we could each access using Google Console. We connected this database to pgAdmin and were able to create each table. We connected our Jupyter Notebook to our database and were able to load each data frame into its designated table. We then chose five queries to run on pgAdmin 4. Shown below are the outputs of the queries:

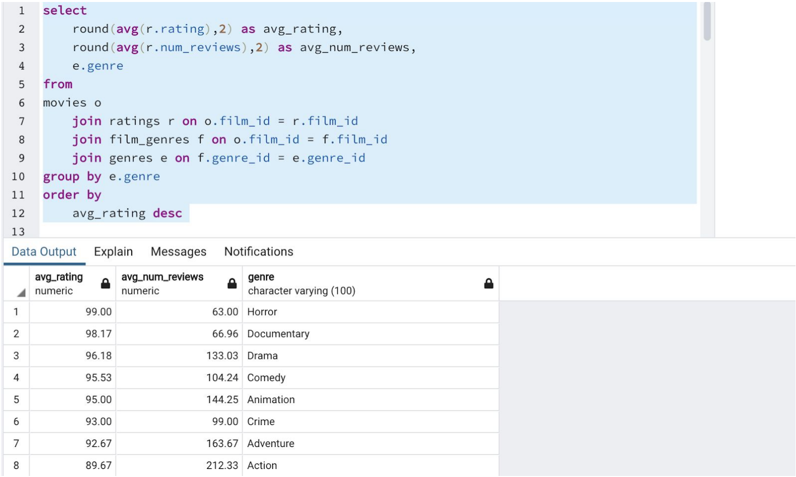
1) Top 10 Movies - we wanted to see the top 10 movies in 2020.



```sql
select
    r.rank,
    o.title
from
movies o
    join ratings r on o.film_id = r.film_id
limit 10
```

	rank integer	title character varying (100)
1	1	Portrait of a Lady on Fire
2	2	1917
3	3	The Invisible Man
4	4	Never Rarely Sometimes Alwa...
5	5	Hamilton
6	6	Da 5 Bloods
7	7	Sound of Metal
8	8	Nomadland
9	9	Miss Juneteenth
10	10	Onward

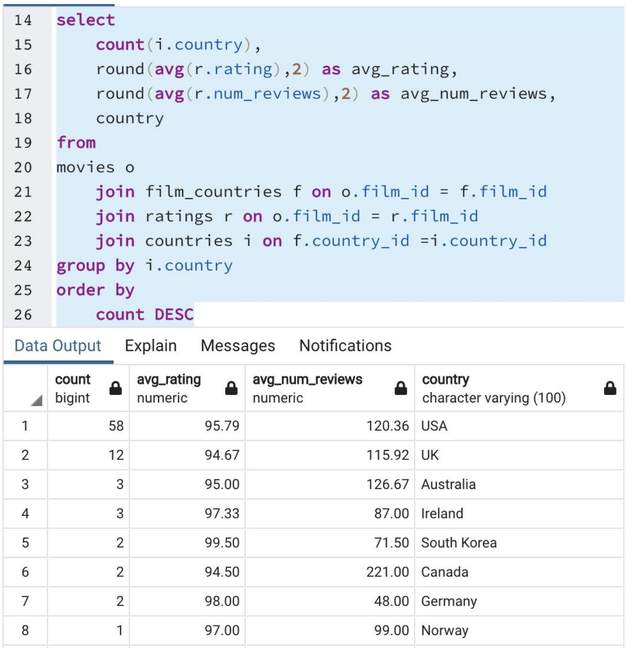
2) Average Rating by Genre - we looked for what genre had the highest rating based on the number of reviews. As seen below, Horror movies had the highest rating but the least number of reviews, which was an interesting insight.



```sql
select
    round(avg(r.rating),2) as avg_rating,
    round(avg(r.num_reviews),2) as avg_num_reviews,
    e.genre
from
movies o
    join ratings r on o.film_id = r.film_id
    join film_genres f on o.film_id = f.film_id
    join genres e on f.genre_id = e.genre_id
group by e.genre
order by
    avg_rating desc
```

	avg_rating numeric	avg_num_reviews numeric	genre character varying (100)
1	99.00	63.00	Horror
2	98.17	66.96	Documentary
3	96.18	133.03	Drama
4	95.53	104.24	Comedy
5	95.00	144.25	Animation
6	93.00	99.00	Crime
7	92.67	163.67	Adventure
8	89.67	212.33	Action

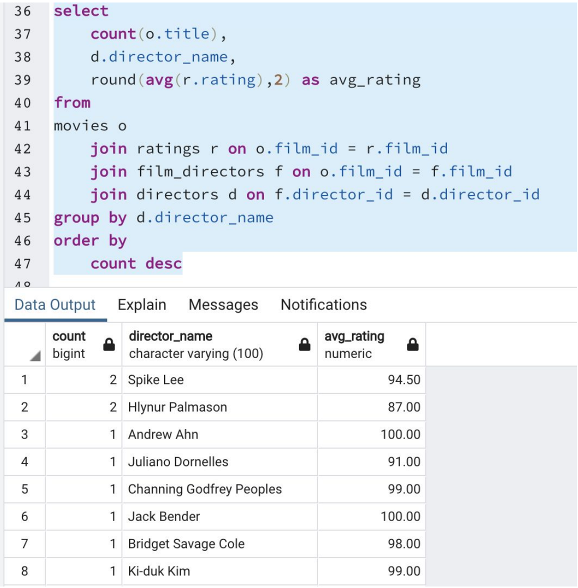
3) Count of Number of Films per Country - This showed us which country had the most movies shown for the top 100 2020 movies. The USA had the highest count by far.



```
select
    count(i.country),
    round(avg(r.rating),2) as avg_rating,
    round(avg(r.num_reviews),2) as avg_num_reviews,
    country
from
movies o
    join film_countries f on o.film_id = f.film_id
    join ratings r on o.film_id = r.film_id
    join countries i on f.country_id =i.country_id
group by i.country
order by
    count DESC
```

	count bigint	avg_rating numeric	avg_num_reviews numeric	country character varying (100)
1	58	95.79	120.36	USA
2	12	94.67	115.92	UK
3	3	95.00	126.67	Australia
4	3	97.33	87.00	Ireland
5	2	99.50	71.50	South Korea
6	2	94.50	221.00	Canada
7	2	98.00	48.00	Germany
8	1	97.00	99.00	Norway

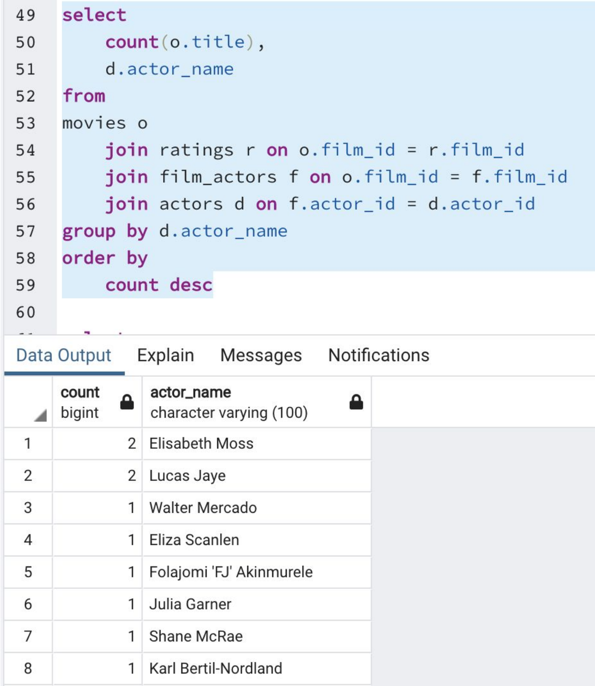
4) Count of films per director with their average rating and number of reviews - This showed us which director had produced the most movies in the top 100 movies of 2020 and what the average rating for those movies was.



```
select
    count(o.title),
    d.director_name,
    round(avg(r.rating),2) as avg_rating
from
movies o
    join ratings r on o.film_id = r.film_id
    join film_directors f on o.film_id = f.film_id
    join directors d on f.director_id = d.director_id
group by d.director_name
order by
    count desc
```

	count bigint	director_name character varying (100)	avg_rating numeric
1	2	Spike Lee	94.50
2	2	Hlynur Palmason	87.00
3	1	Andrew Ahn	100.00
4	1	Juliano Dornelles	91.00
5	1	Channing Godfrey Peoples	99.00
6	1	Jack Bender	100.00
7	1	Bridget Savage Cole	98.00
8	1	Ki-duk Kim	99.00

5) Count of number of films by actor - Finally, we wanted to see the number of movies actors were in for the top 100 movies and if there were actors in multiple top movies.



```
select
    count(o.title),
    d.actor_name
from
movies o
    join ratings r on o.film_id = r.film_id
    join film_actors f on o.film_id = f.film_id
    join actors d on f.actor_id = d.actor_id
group by d.actor_name
order by
    count desc
```

	count bigint	actor_name character varying (100)
1	2	Elisabeth Moss
2	2	Lucas Jaye
3	1	Walter Mercado
4	1	Eliza Scanlen
5	1	Folajomi 'FJ' Akinmurele
6	1	Julia Garner
7	1	Shane McRae
8	1	Karl Bertil-Nordland

As a bonus, we made a Flask App on VS Code to show our data as well.