

Task_4

1.

The screenshot shows the pgAdmin interface with the following components:

- Browser:** A tree view on the left showing the database structure. It includes 'Databases (2)', 'Rockbuster', and 'Schemas (1)' with a 'public' schema containing various objects like Collations, Domains, FTS Configurations, etc.
- Query Editor:** The main area on the right where the SQL query is entered. The query is:

```
1 EXPLAIN
2 SELECT *
3 FROM film
```
- Data Output:** A tab at the bottom showing the query plan. It indicates a 'Seq Scan on film' with a cost of 0.00..64.00, 1000 rows, and a width of 388.

The screenshot shows the pgAdmin interface with the following components:

- Browser:** A tree view on the left showing the database structure. It includes 'Databases (2)', 'Rockbuster', and 'Schemas (1)' with a 'public' schema containing various objects like Collations, Domains, FTS Configurations, etc.
- Query Editor:** The main area on the right where the SQL query is entered. The query is:

```
1 EXPLAIN
2 SELECT film_id, title
3 FROM film
```
- Data Output:** A tab at the bottom showing the query plan. It indicates a 'Seq Scan on film' with a cost of 0.00..64.00, 1000 rows, and a width of 19.

The cost for both queries is same. Only the width differs in both results. According to our results width does not have any effect on cost. However, we can optimize our results limiting the rows if we know what exactly we need.

2.

The screenshot shows the pgAdmin 4 web interface. On the left is the 'Browser' pane showing the database structure. The 'public' schema is expanded, showing tables. The 'category' table is selected, and its columns are visible: 'category_id', 'name', and 'last_update'. The main pane shows the 'Query Editor' with a SQL query:

```
1 SELECT title, release_year, rental_rate FROM film
2 ORDER BY title ASC,
3 release_year DESC,
4 rental_rate DESC
```

Below the query editor is the 'Data Output' tab, which displays the results of the query in a table. The table has four columns: 'id', 'title', 'release_year', and 'rental_rate'. The results are sorted by title in ascending order, then by release_year in descending order, and finally by rental_rate in descending order.

	title	release_year	rental_rate
1	Academy Dinosaur	2006	0.99
2	Ace Goldfinger	2006	4.99
3	Adaptation Holes	2006	2.99
4	Affair Prejudice	2006	2.99
5	African Egg	2006	2.99
6	Agent Truman	2006	2.99
7	Airplane Sierra	2006	4.99
8	Airport Pollock	2006	4.99
9	Alabama Devil	2006	2.99
10	Aladdin Calendar	2006	4.99
11	Alamo Videotape	2006	0.99
12	Alaska Phantom	2006	0.99
13	Ali Forever	2006	4.99
14	Alice Fantasia	2006	0.99

3.

The screenshot shows the pgAdmin 4 interface. On the left, the 'Databases (2)' tree is expanded, showing 'Rockbuster' and 'public' schemas. The 'public' schema is selected, and the 'category' table is highlighted. The 'Query Editor' tab is active, displaying the following SQL query:

```
1 SELECT rating, ROUND(AVG(rental_rate),2) AS "average rental rate" FROM film
2 GROUP BY rating
```

The 'Data Output' tab shows the results of the query in a table with two columns: 'rating' (mpaa_rating) and 'average rental rate' (numeric). The results are as follows:

rating	average rental rate
1 R	2.94
2 NC-17	2.97
3 G	2.89
4 PG	3.05
5 PG-13	3.03

The screenshot shows the pgAdmin 4 interface. On the left, the 'Databases (2)' tree is expanded, showing 'Rockbuster' and 'public' schemas. The 'public' schema is selected, and the 'category' table is highlighted. The 'Query Editor' tab is active, displaying the following SQL query:

```
1 SELECT rating, MIN(rental_duration) AS "min rental rate", MAX(rental_duration) AS "max rental rate"
2 FROM film
3 GROUP BY rating
```

The 'Data Output' tab shows the results of the query in a table with three columns: 'rating' (mpaa_rating), 'min rental rate' (smallint), and 'max rental rate' (smallint). The results are as follows:

rating	min rental rate	max rental rate
1 R	3	7
2 NC-17	3	7
3 G	3	7
4 PG	3	7
5 PG-13	3	7

4.

- Firstly, we **Extract** the necessary information from the app (Performed by data engineers, cloud engineers)
- We **Transform** the data into suitable format (Performed by data engineers also possible data analysts and scientist)
- Loading** the data into new database

If data is analyzed before migrating, firstly there might be a “clean data” issue. However, it is also possible that the data that analyzed before migrating, will not be match our current inventory data.