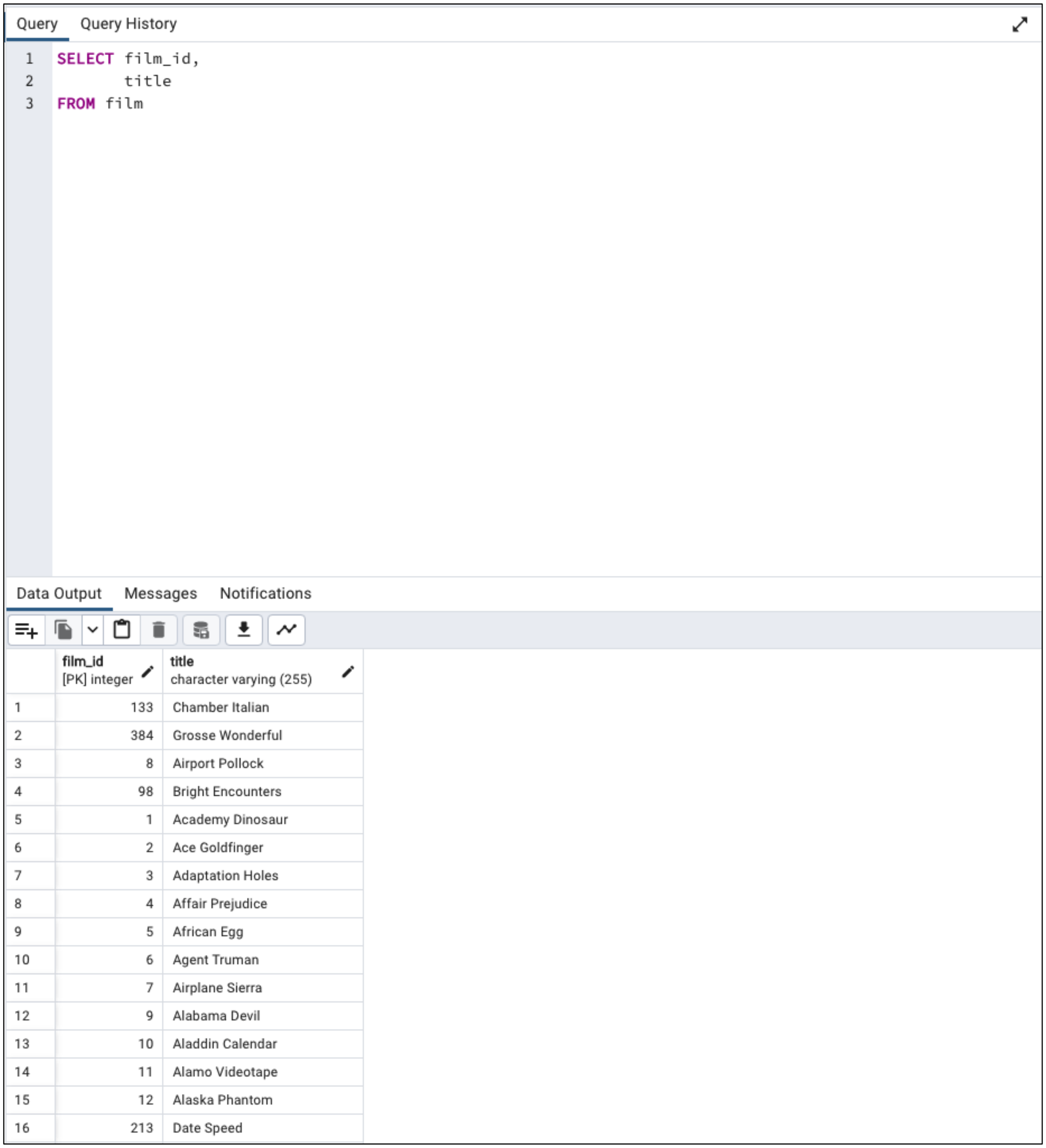
1. **Refining Your Query:** You need to get some data from the "film" table and decide to use the query SELECT * FROM film.

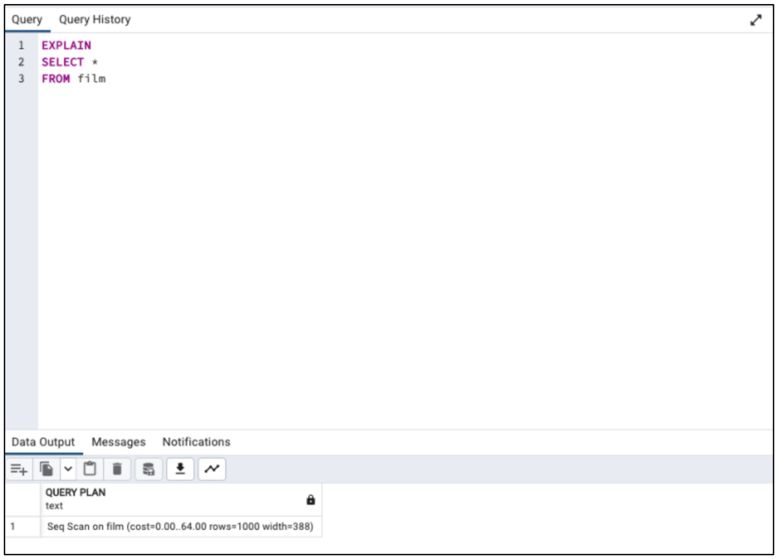
- **You realize that only the "film_id" and "title" columns are needed. Write a new query that selects only those 2 columns.**



```
SELECT film_id,
       title
FROM film
```

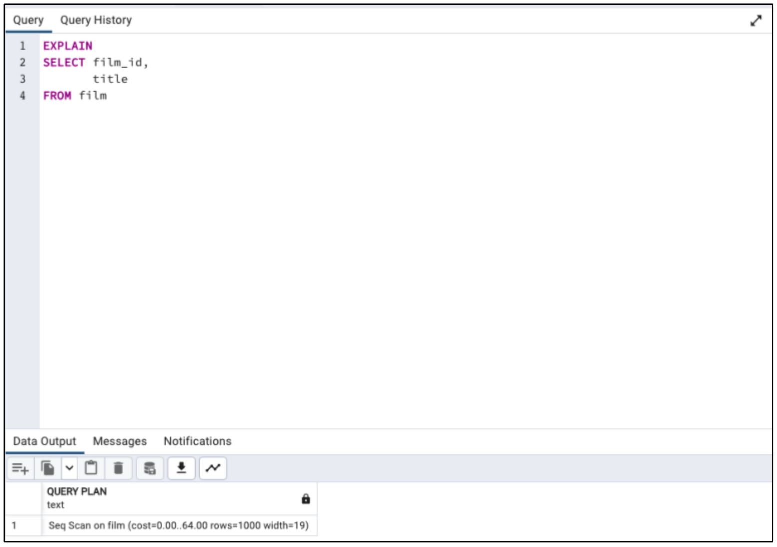
	film_id [PK] integer	title character varying (255)
1	133	Chamber Italian
2	384	Grosse Wonderful
3	8	Airport Pollock
4	98	Bright Encounters
5	1	Academy Dinosaur
6	2	Ace Goldfinger
7	3	Adaptation Holes
8	4	Affair Prejudice
9	5	African Egg
10	6	Agent Truman
11	7	Airplane Sierra
12	9	Alabama Devil
13	10	Aladdin Calendar
14	11	Alamo Videotape
15	12	Alaska Phantom
16	213	Date Speed

- **Compare the cost of the original query and the revised query, and write a few sentences explaining the comparison. Can you suggest any ways to optimize this query?**



```sql
EXPLAIN
SELECT *
FROM film
```

	QUERY PLAN text
1	Seq Scan on film (cost=0.00..64.00 rows=1000 width=388)



```sql
EXPLAIN
SELECT film_id,
       title
FROM film
```

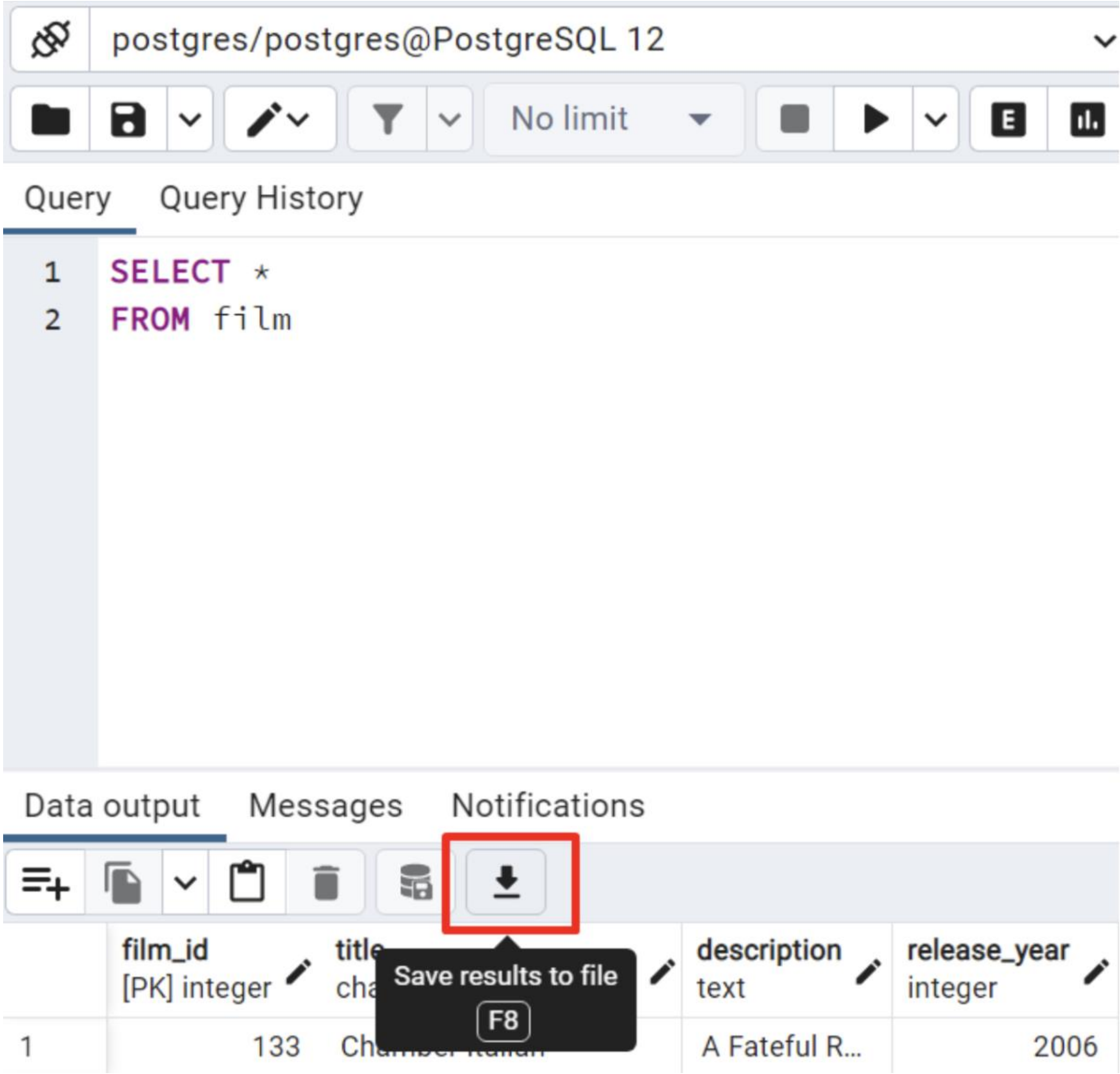
	QUERY PLAN text
1	Seq Scan on film (cost=0.00..64.00 rows=1000 width=19)

The cost or time of returning the first row is 0 for both the original and revised query, however, the cost of returning all the rows is 64. Since the cost unit does not refer to a specified second or minute, all this tells us is that a query with a cost of 64 will take longer than a query with a cost of 30 (seconds).

To optimize this query, use LIMIT to sample query results. Using a LIMIT statement prevents taxing the production database with a large query, only to discover the query requires editing or refinement. Run your query during off-peak hours when concurrent users are at their lowest numbers, which is in the middle of the night.

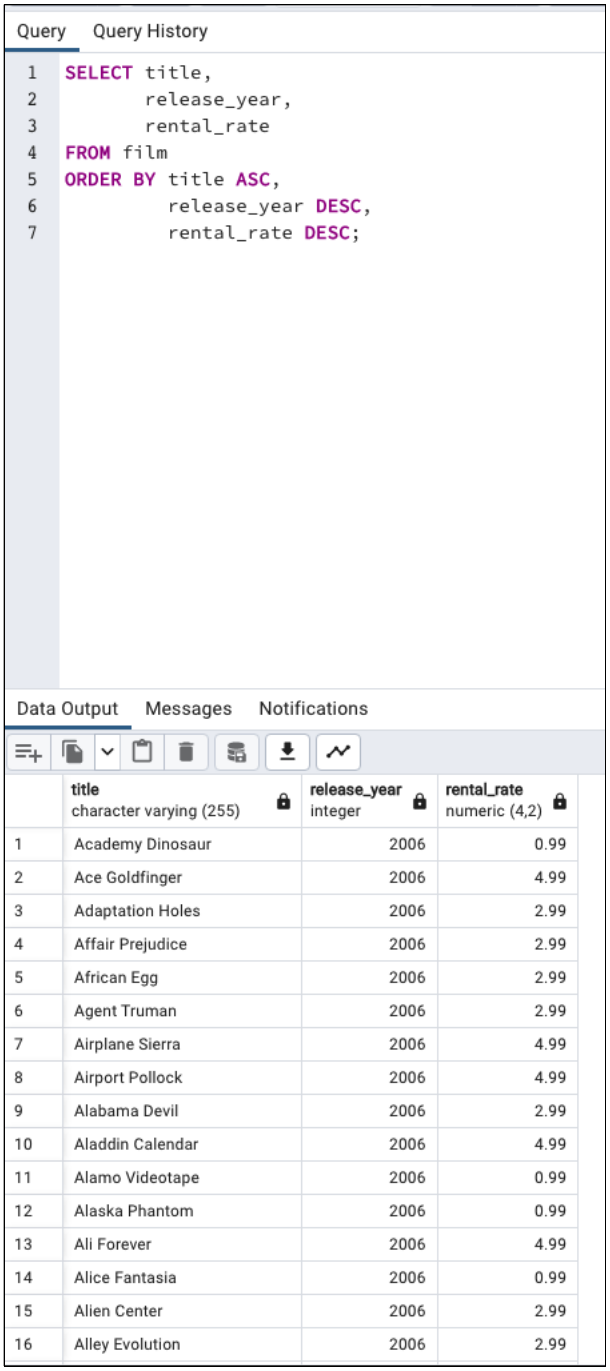
2. **Ordering the Data:**

- **In the pgAdmin Query Tool, run a query that selects every film from the "film" table, with the movies sorted by title from A to Z, then by most recent release year, and then by highest to lowest rental rate.**
- **Extract the data output of your query into a CSV file for the film collection department to analyze in Excel. To do this, click the button "Save results to file":**



```
SELECT *
FROM film
```

	film_id [PK] integer	title cha	description text	release_year integer
1	133	Ch	A Fateful R...	2006



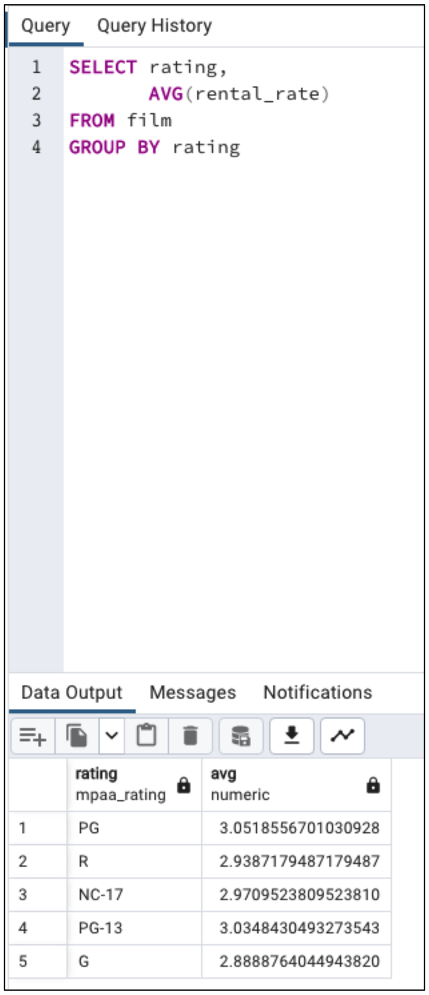
```
SELECT title,
       release_year,
       rental_rate
FROM film
ORDER BY title ASC,
         release_year DESC,
         rental_rate DESC;
```

	title character varying (255)	release_year integer	rental_rate numeric (4,2)
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
15	Alien Center	2006	2.99
16	Alley Evolution	2006	2.99

3. **Grouping Data: The strategy department has asked you the questions below. Write a SQL query to retrieve the correct answers, then extract your results as a CSV file.**

○ **What is the average rental rate for each rating category?**

We want to see the average of rental rate by rating so include "rating" after SELECT and GROUP BY.

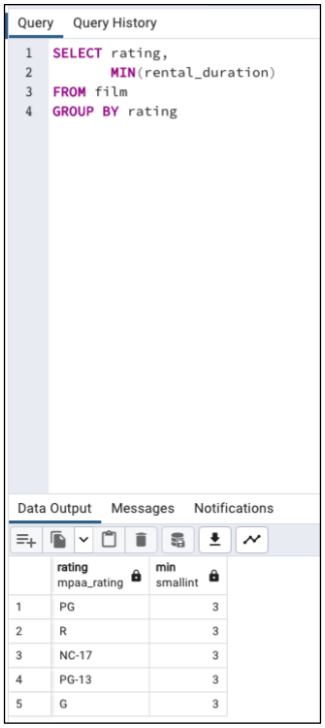


```
SELECT rating,
       AVG(rental_rate)
FROM film
GROUP BY rating
```

	rating mpaa_rating	avg numeric
1	PG	3.0518556701030928
2	R	2.9387179487179487
3	NC-17	2.9709523809523810
4	PG-13	3.0348430493273543
5	G	2.8888764044943820

○ What are the minimum and maximum rental durations for each rating category?

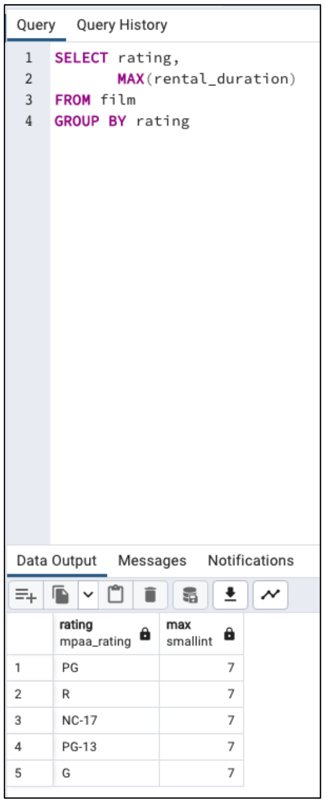
Minimum



```
SELECT rating,
       MIN(rental_duration)
FROM film
GROUP BY rating
```

	rating mpaa_rating	min smallint
1	PG	3
2	R	3
3	NC-17	3
4	PG-13	3
5	G	3

Maximum



```
SELECT rating,
       MAX(rental_duration)
FROM film
GROUP BY rating
```

	rating mpaa_rating	max smallint
1	PG	7
2	R	7
3	NC-17	7
4	PG-13	7
5	G	7

4. **Database Migration: Your team has decided to use an external tool to collect data on user behavior in the new Rockbuster Android app. Data collected from this new source will need to be loaded into the data warehouse before you can analyze it.**

- **Can you outline the procedure for migrating the data and who will be responsible for it?**

  If we want to migrate the user behavior data into the Rockbuster database, we need to follow the Extract, Transform, and Load (ETL) procedure. The data engineer will be responsible in executing the procedure. The first step involves collecting (or extract) the data from multiple data sources such as databases, applications, and flat files. The second step is transforming, the extracted data is converted into another format. The transformation process corrects the data, removes any incorrect data and fixes any errors in the data before loading it. Lastly, the transformed data is inserted or loaded into a data warehouse, where we can use it to answer important questions such as the user behavior in the new Rockbuster Android application.

- **What problems do you foresee if you start analyzing the data before it's been loaded into the data warehouse?**

  If an analyst begins to analyze data prior to being loaded into the data warehouse, then the data is inaccurate. The multiple data sources would be considered unreliable since it contains incorrect data that requires further remediation. An analyst will experience difficulties in answering the important questions such as the user behavior in the new Rockbuster Android application.