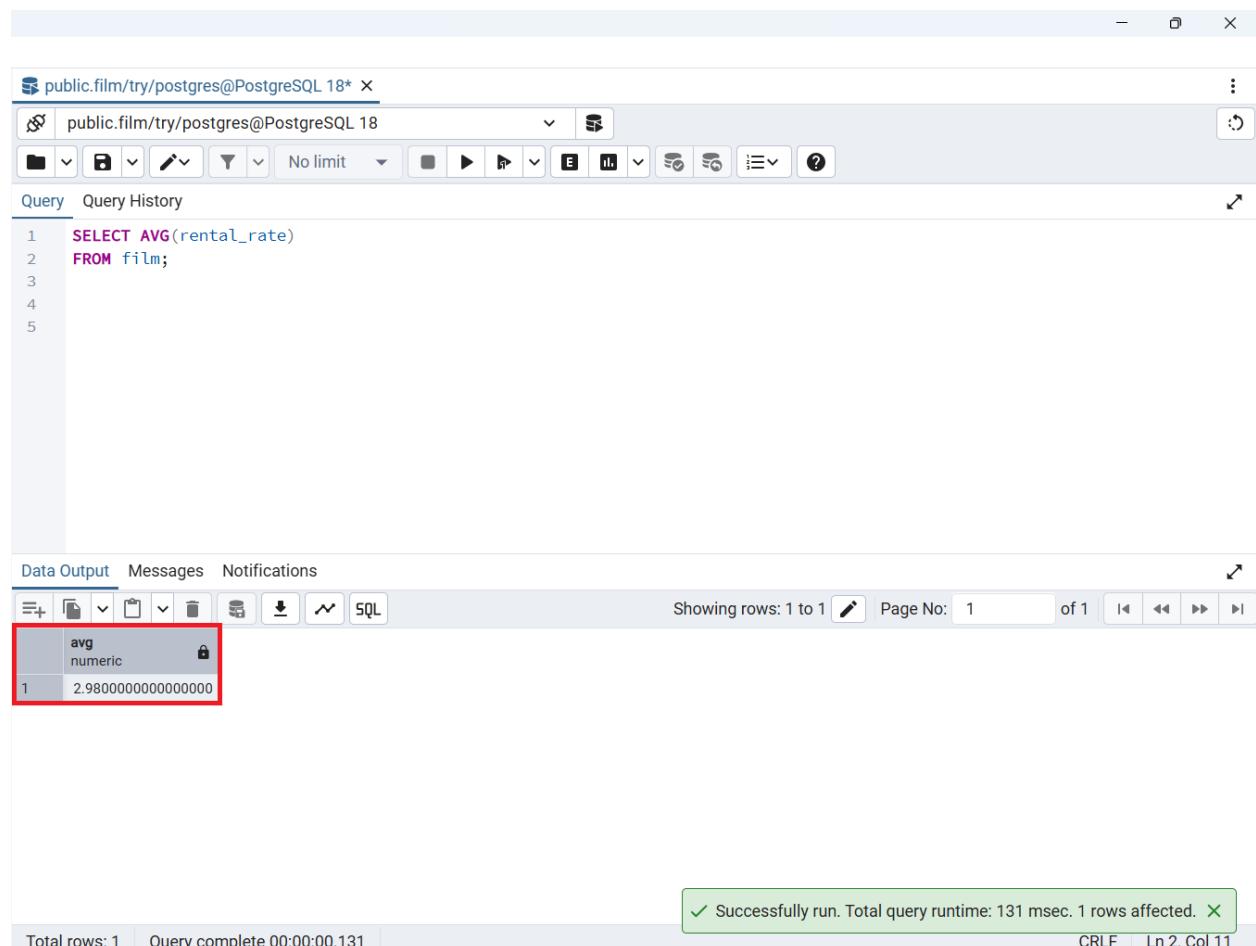


Assignment 6: DVDRental Queries

This query focuses on identifying and counting distinct values in the DVDRental database. Specifically, it retrieves the number of unique replacement_cost values for films that have a length greater than 150 minutes. The task emphasizes the use of conditional filtering with the WHERE clause and the COUNT(DISTINCT ...) aggregate function to ensure that only unique cost values are counted. This exercise helps strengthen understanding of data uniqueness, filtering conditions, and aggregate functions in SQL.

Query 1:

Description: This query finds the average rental_rate in the film table, demonstrating how to use the AVG() function to summarize numerical data in SQL.



The screenshot shows a PostgreSQL client interface with the following details:

- Connection:** public.film/try/postgres@PostgreSQL 18*
- Toolbar:** Includes icons for connection, refresh, search, and various database operations.
- Query Tab:** Contains the SQL query:

```
1 SELECT AVG(rental_rate)
2 FROM film;
```
- Data Output Tab:** Shows the results of the query:

	avg	numeric
1	2.980000000000000	2.980000000000000

The result row is highlighted with a red box.
- Status Bar:** Shows "Total rows: 1" and "Query complete 00:00:00.131".
- Message Bar:** A green message box indicates "Successfully run. Total query runtime: 131 msec. 1 rows affected."
- Bottom Right:** CRLF | Ln 2, Col 11

Query 2:

Description: This query counts the number of films in the film table whose titles start with the letter 'C', demonstrating the use of COUNT() with the LIKE operator to filter data in SQL.

The screenshot shows a PostgreSQL client interface with the following details:

- Query Editor:** The title bar says "public.film/try/postgres@PostgreSQL 18*". The toolbar includes icons for file operations, search, and database management. The query history tab is selected.
- Query Text:** The query is:

```
1 SELECT COUNT(*)
2 FROM film
3 WHERE title LIKE 'C%';
4
```
- Data Output:** The "Data Output" tab is selected. It shows a table with one row:

	count
1	92

The first column is labeled "count" and has a type of "bigint". The value "92" is highlighted with a red box.
- Status Bar:** At the bottom, it says "Total rows: 1" and "Query complete 00:00:00.103". On the right, it shows "CRLF" and "Ln 4, Col 1".

Query 3:

Description: This query finds the longest film (length) with a rental_rate of 0.99 in the film table, using the MAX() function combined with a WHERE filter to analyze specific data in SQL.

The screenshot shows the pgAdmin 4 interface. The top bar displays the connection information: public.film/try/postgres@PostgreSQL 18*. Below the toolbar, there are tabs for 'Query' and 'Query History'. The main area contains the following SQL code:

```
1 SELECT MAX(length) FROM film
2 WHERE rental_rate = 0.99;
```

The results are displayed in a table under the 'Data Output' tab. The table has two columns: 'max' and 'smallint'. The first row shows a value of 184. The entire first row is highlighted with a red box.

max	smallint
1	184

At the bottom of the interface, status bars show 'Total rows: 1' and 'Query complete 00:00:00.294'.

Query 4:

Description: In this query counts the number of unique replacement_cost values for films longer than 150 minutes, demonstrating the use of COUNT(DISTINCT ...) with a WHERE filter to analyze distinct data in SQL.

The screenshot shows the pgAdmin 4 interface with a single query window open. The query is:

```
1 SELECT COUNT(DISTINCT replacement_cost)
2 FROM film
3 WHERE length > 150;
```

The results pane shows a single row of data:

count	bigint
1	21

A red box highlights the first column of the result table. At the bottom of the interface, status messages indicate "Total rows: 1" and "Query complete 00:00:00.058".