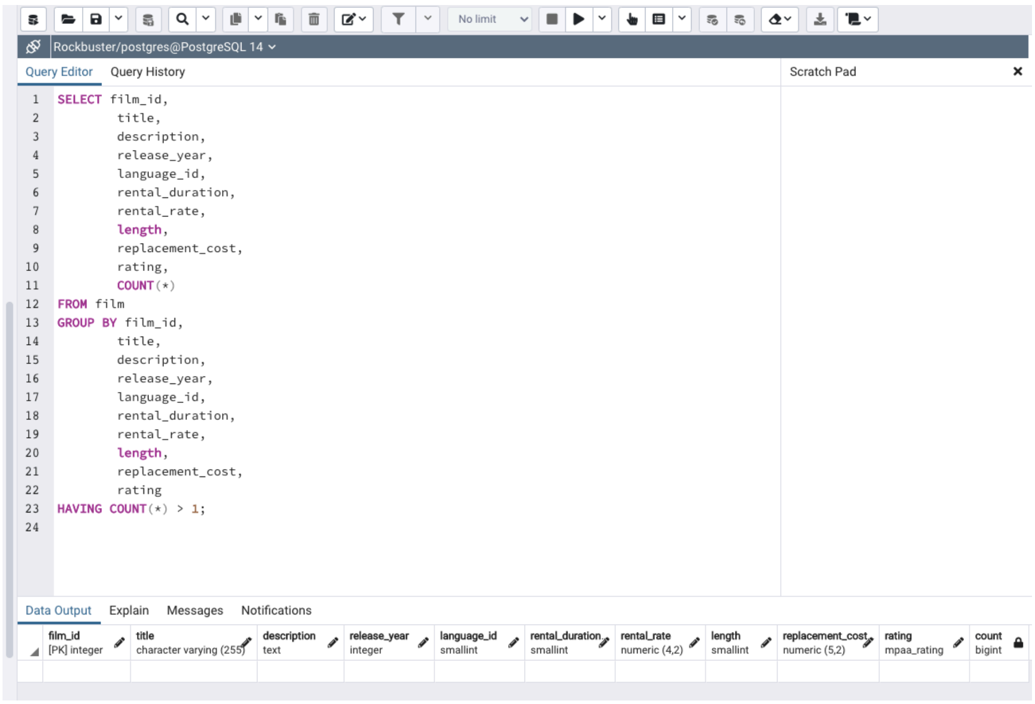
3.6

1. **Check for and clean dirty data:** Find out if the film table and the customer table contain any dirty data, specifically non-uniform or duplicate data, or missing values. Create a new "Answers 3.6" document and copy-paste your queries into it. Next to each query write 2 to 3 sentences explaining how you would clean the data (even if the data is not dirty).

Checking for duplicate value from the film table

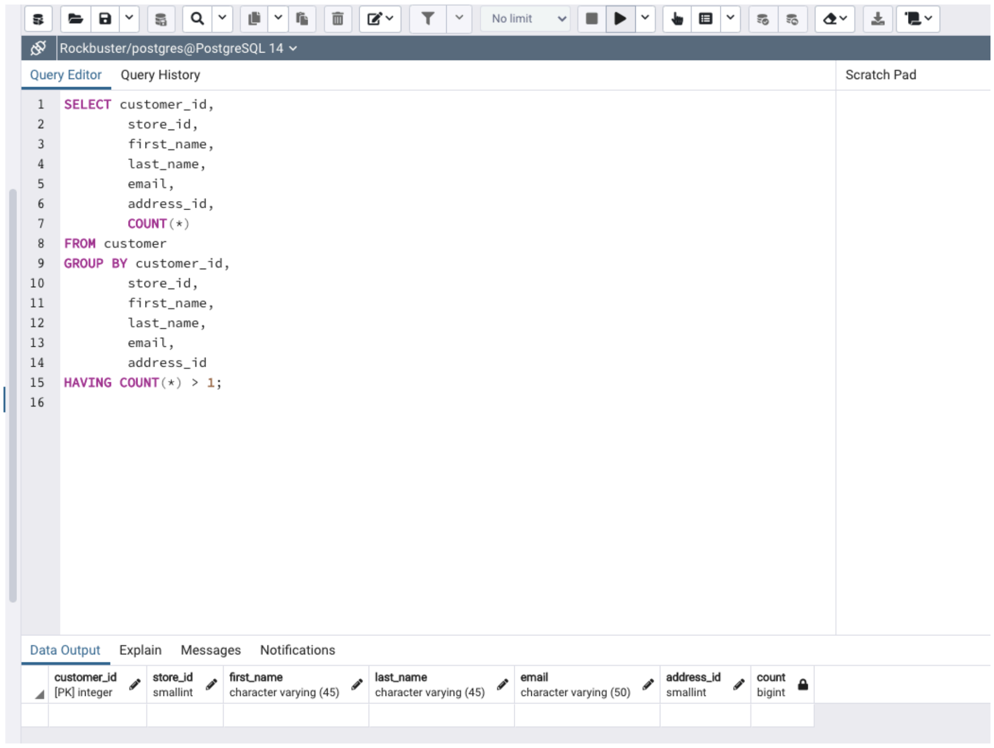


Rockbuster/postgres@PostgreSQL 14

```
SELECT film_id,
        title,
        description,
        release_year,
        language_id,
        rental_duration,
        rental_rate,
        length,
        replacement_cost,
        rating,
        COUNT(*)
FROM film
GROUP BY film_id,
        title,
        description,
        release_year,
        language_id,
        rental_duration,
        rental_rate,
        length,
        replacement_cost,
        rating
HAVING COUNT(*) > 1;
```

```
film_id       title                   description  release_year  language_id  rental_duration  rental_rate   length    replacement_cost  rating       count
[PK] integer  character varying (255) text         integer       smallint     smallint         numeric (4,2) smallint  numeric (5,2)     mpaa_rating  bigint
```

Checking for duplicate value from the customer table



Rockbuster/postgres@PostgreSQL 14

```
SELECT customer_id,
        store_id,
        first_name,
        last_name,
        email,
        address_id,
        COUNT(*)
FROM customer
GROUP BY customer_id,
        store_id,
        first_name,
        last_name,
        email,
        address_id
HAVING COUNT(*) > 1;
```

```
customer_id   store_id  first_name              last_name               email                   address_id  count
[PK] integer  smallint  character varying (45)  character varying (45)  character varying (50)  smallint    bigint
```

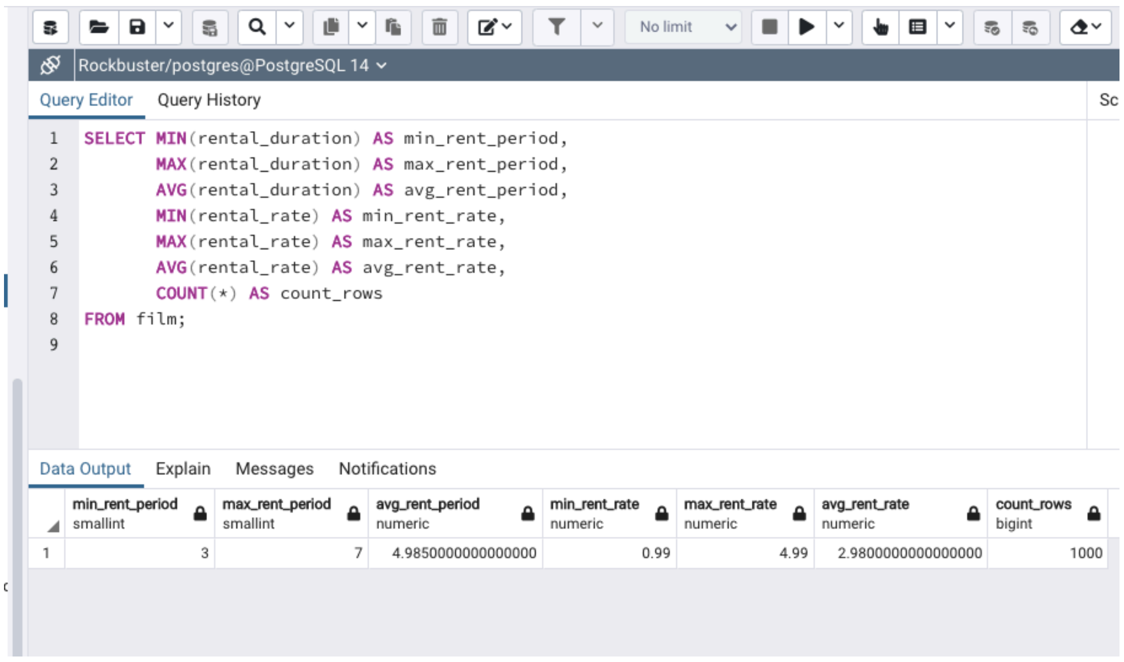
There is no returned duplicate value. Essentially, there are two ways of dealing with duplicate value if you have permission to alter the database.

- Create a virtual table “View” where unique records can be selected
- Delete duplicate record from the table or View

However, if altering table is not permitted, we can use GROUP BY or DISTINCT to select unique records.

2. **Summarize your data:** Use SQL to calculate descriptive statistics for both the film table and the customer table. For numerical columns, this means finding the minimum, maximum, and average values. For non-numerical columns, calculate the mode value. Copy-paste your SQL queries and their outputs into your answers document.

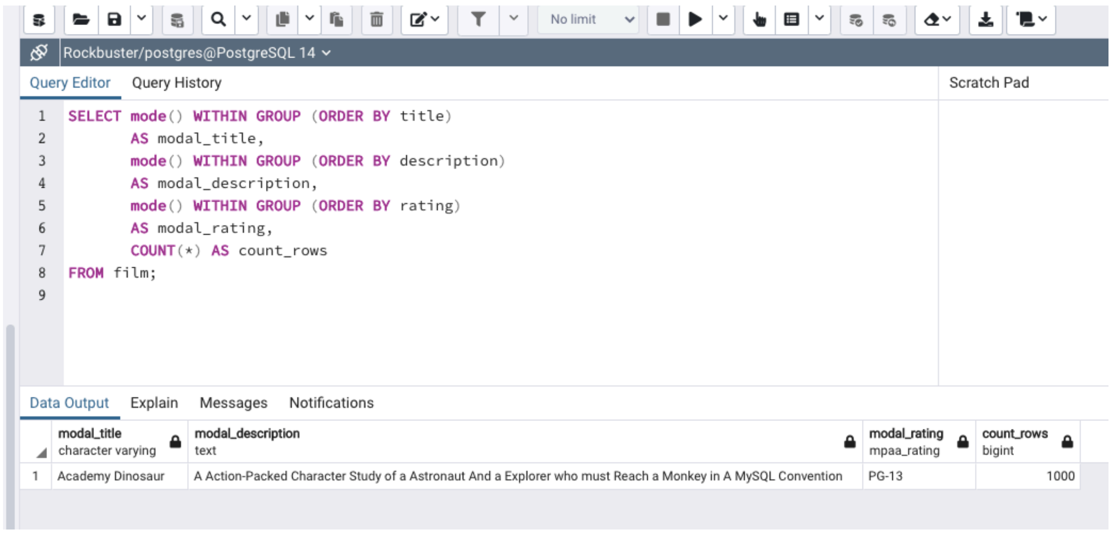
Summary for numeric columns in film table



```sql
SELECT MIN(rental_duration) AS min_rent_period,
       MAX(rental_duration) AS max_rent_period,
       AVG(rental_duration) AS avg_rent_period,
       MIN(rental_rate) AS min_rent_rate,
       MAX(rental_rate) AS max_rent_rate,
       AVG(rental_rate) AS avg_rent_rate,
       COUNT(*) AS count_rows
FROM film;
```

```
  min_rent_period  max_rent_period  avg_rent_period     min_rent_rate  max_rent_rate  avg_rent_rate       count_rows
  smallint         smallint         numeric            numeric        numeric        numeric             bigint
1               3                7  4.9850000000000000           0.99           4.99  2.9800000000000000        1000
```

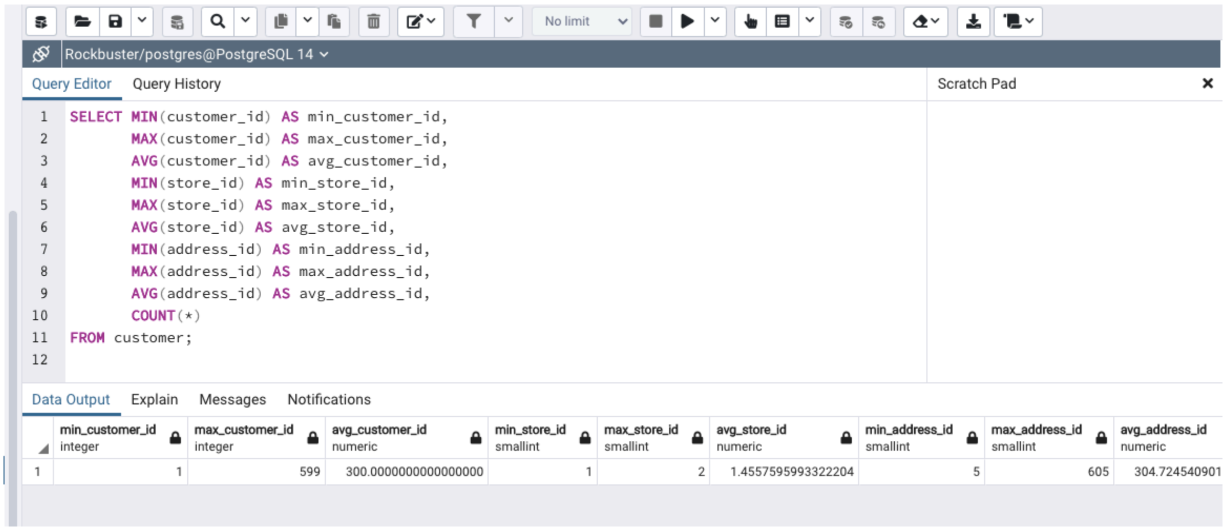
Summary for non-numeric columns in film table



```sql
SELECT mode() WITHIN GROUP (ORDER BY title)
       AS modal_title,
       mode() WITHIN GROUP (ORDER BY description)
       AS modal_description,
       mode() WITHIN GROUP (ORDER BY rating)
       AS modal_rating,
       COUNT(*) AS count_rows
FROM film;
```

```
  modal_title        modal_description                                                                                      modal_rating  count_rows
  character varying  text                                                                                                   mpaa_rating   bigint
1 Academy Dinosaur   A Action-Packed Character Study of a Astronaut And a Explorer who must Reach a Monkey in A MySQL Convention  PG-13   1000
```

Summary for numeric columns in customer table

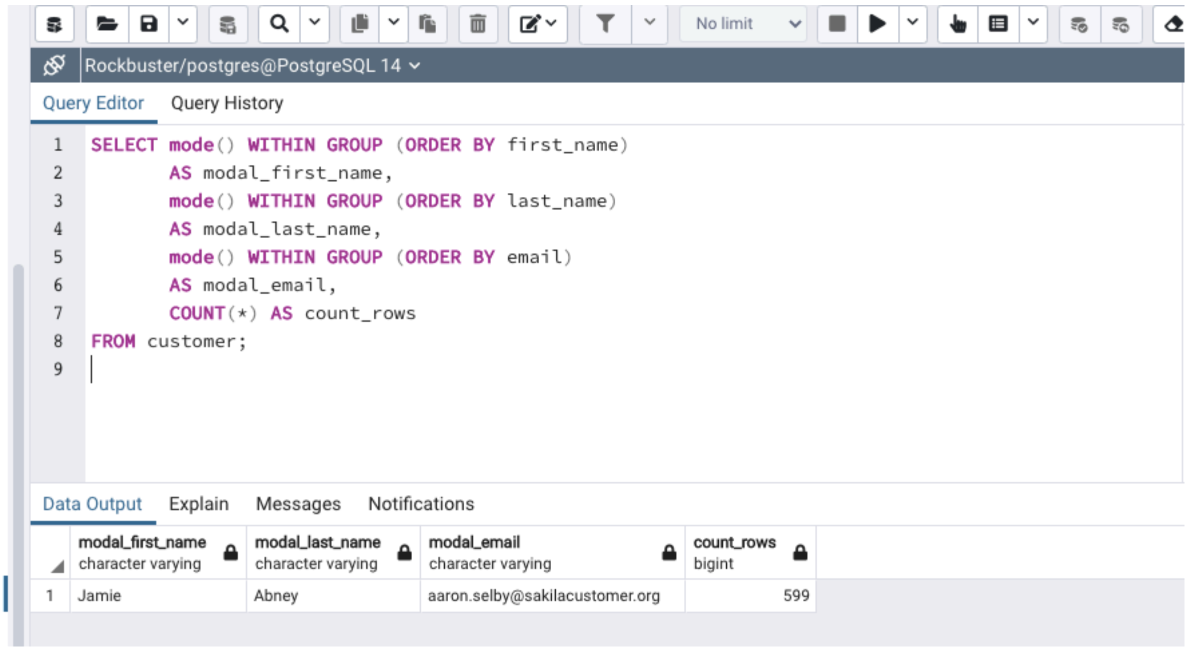


Rockbuster/postgres@PostgreSQL 14

```sql
SELECT MIN(customer_id) AS min_customer_id,
       MAX(customer_id) AS max_customer_id,
       AVG(customer_id) AS avg_customer_id,
       MIN(store_id) AS min_store_id,
       MAX(store_id) AS max_store_id,
       AVG(store_id) AS avg_store_id,
       MIN(address_id) AS min_address_id,
       MAX(address_id) AS max_address_id,
       AVG(address_id) AS avg_address_id,
       COUNT(*)
FROM customer;
```

	min_customer_id integer	max_customer_id integer	avg_customer_id numeric	min_store_id smallint	max_store_id smallint	avg_store_id numeric	min_address_id smallint	max_address_id smallint	avg_address_id numeric
1	1	599	300.0000000000000000	1	2	1.4557595993322204	5	605	304.724540901

Summary for non-numeric columns in customer table



Rockbuster/postgres@PostgreSQL 14

```sql
SELECT mode() WITHIN GROUP (ORDER BY first_name)
       AS modal_first_name,
       mode() WITHIN GROUP (ORDER BY last_name)
       AS modal_last_name,
       mode() WITHIN GROUP (ORDER BY email)
       AS modal_email,
       COUNT(*) AS count_rows
FROM customer;
```

	modal_first_name character varying	modal_last_name character varying	modal_email character varying	count_rows bigint
1	Jamie	Abney	aaron.selby@sakilacustomer.org	599

3. **Reflect on your work:** Back in Achievement 1 you learned about data profiling in Excel. Based on your previous experience, which tool (Excel or SQL) do you think is more effective for data profiling, and why? Consider their respective functions, ease of use, and speed. Write a short paragraph in the running document that you have started.

Excel works perfectly with small data size. With this it will be easy to view the data using the pivot table. However, renaming the output (Aliasing) for aggregate column would take more time in excel. Essentially, it is easy to work with huge data in SQL. Using SQL data profiling becomes much easier and faster. Specific result/query to details would be returned in a glance with the right syntax. Conclusively, SQL works perfectly for data profiling based on the speed and ability to work with huge data.