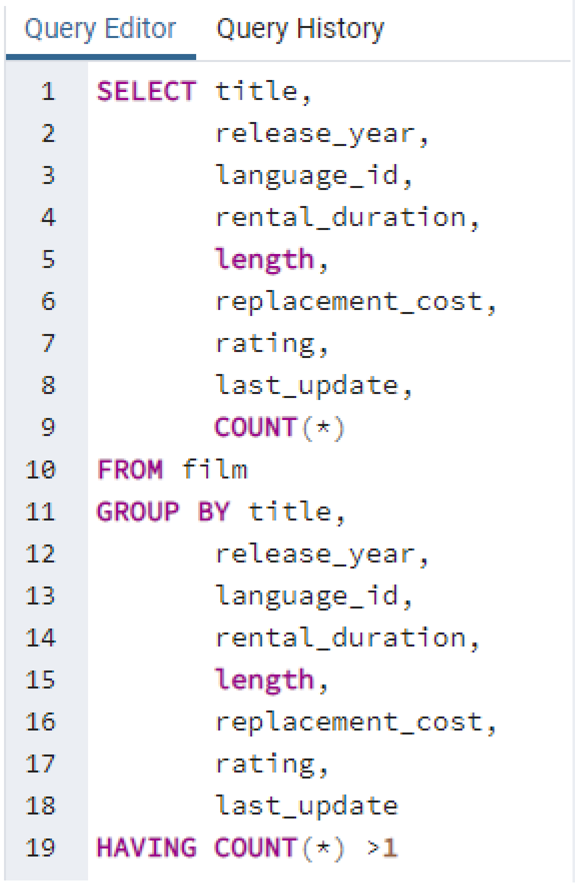
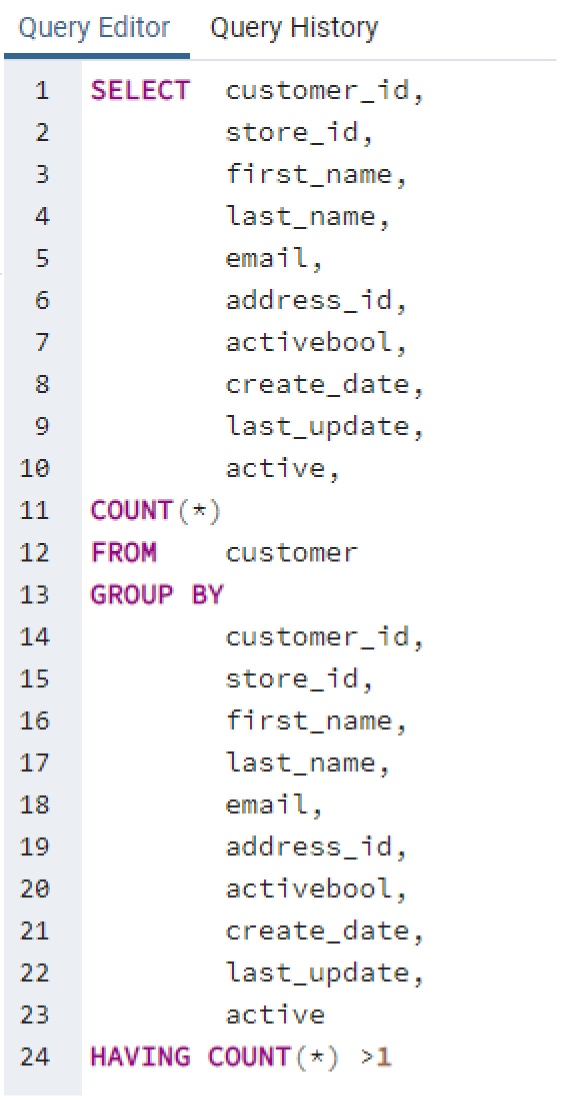


```
SELECT title,
       release_year,
       language_id,
       rental_duration,
       length,
       replacement_cost,
       rating,
       last_update,
       COUNT(*)
FROM film
GROUP BY title,
       release_year,
       language_id,
       rental_duration,
       length,
       replacement_cost,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

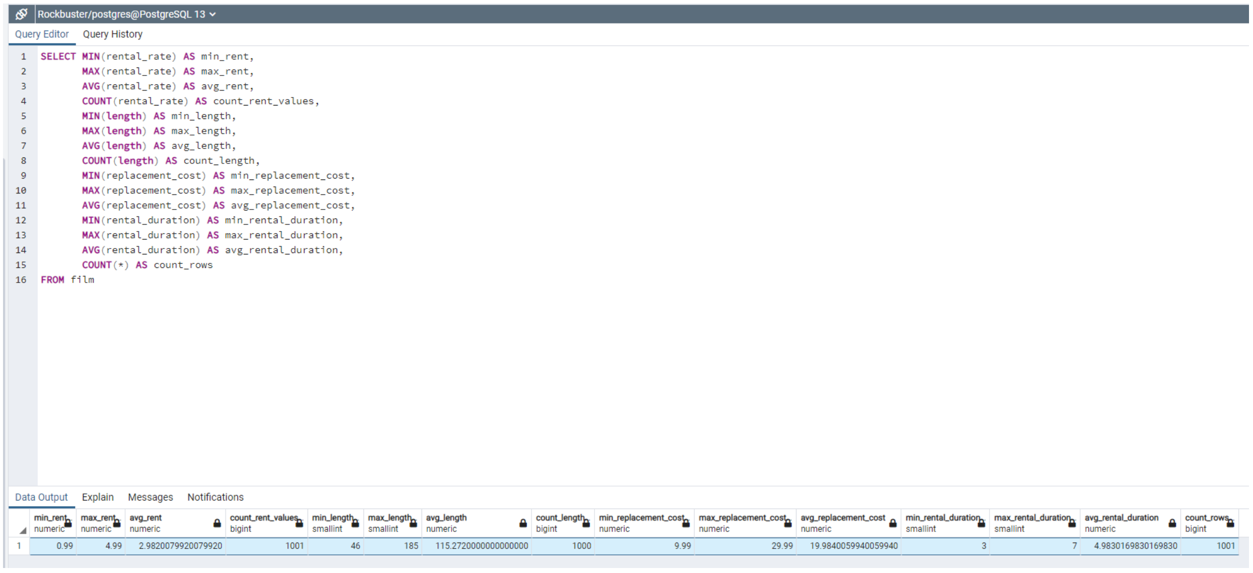


```
SELECT  customer_id,
        store_id,
        first_name,
        last_name,
        email,
        address_id,
        activebool,
        create_date,
        last_update,
        active,
COUNT(*)
FROM    customer
GROUP BY
        customer_id,
        store_id,
        first_name,
        last_name,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

No duplicates found in either table. If any data was found to be a duplicate, I would first verify that the data was indeed a duplicate, then make the necessary changes to it in order to either make it correct or flag it for deletion.

2) Film Table

Numerical


Rockbuster/postgres@PostgreSQL 13

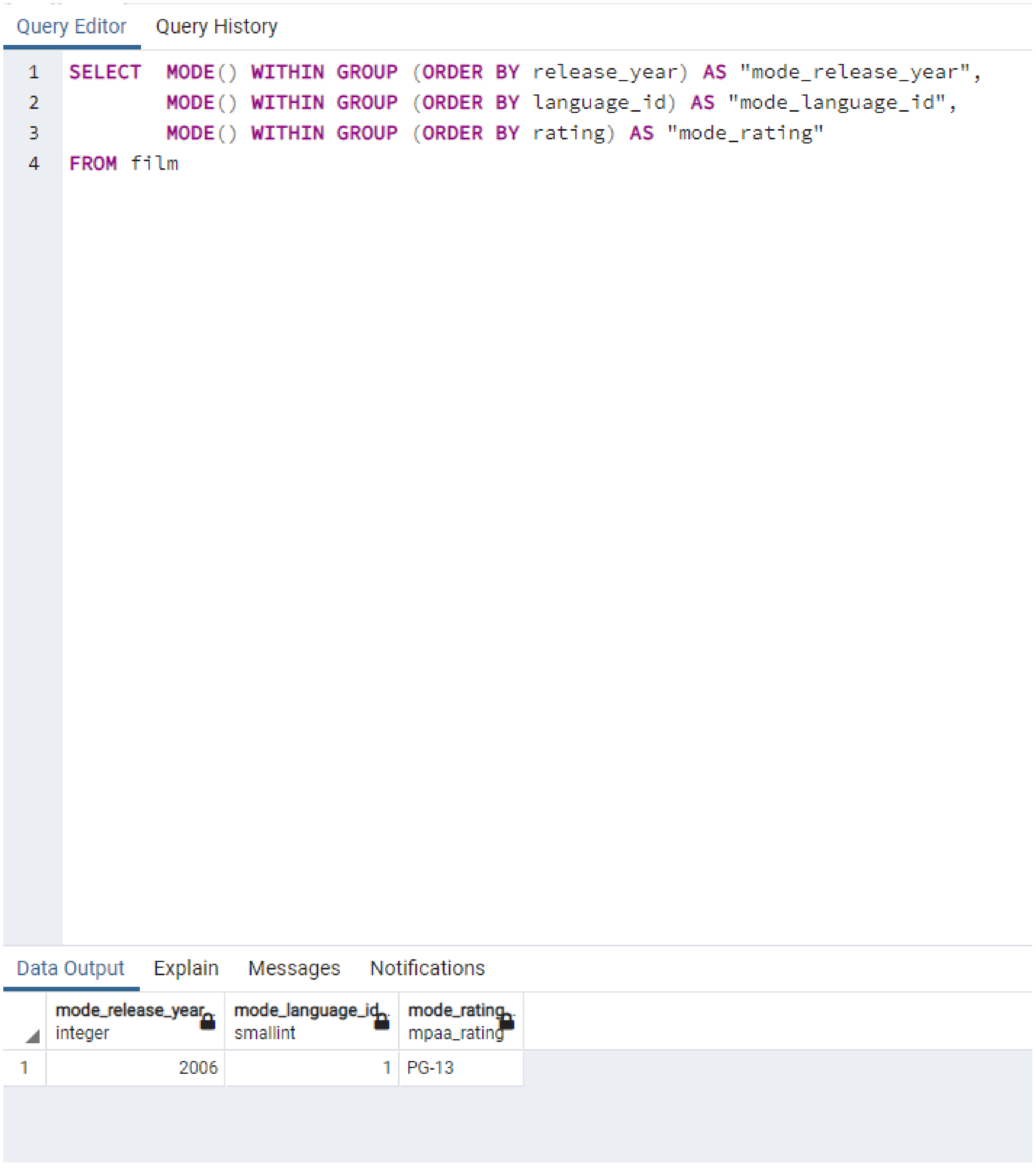
Query Editor | Query History

```sql
SELECT MIN(rental_rate) AS min_rent,
       MAX(rental_rate) AS max_rent,
       AVG(rental_rate) AS avg_rent,
       COUNT(rental_rate) AS count_rent_values,
       MIN(length) AS min_length,
       MAX(length) AS max_length,
       AVG(length) AS avg_length,
       COUNT(length) AS count_length,
       MIN(replacement_cost) AS min_replacement_cost,
       MAX(replacement_cost) AS max_replacement_cost,
       AVG(replacement_cost) AS avg_replacement_cost,
       MIN(rental_duration) AS min_rental_duration,
       MAX(rental_duration) AS max_rental_duration,
       AVG(rental_duration) AS avg_rental_duration,
       COUNT(*) AS count_rows
FROM film
```

Data Output | Explain | Messages | Notifications

	min_rent numeric	max_rent numeric	avg_rent numeric	count_rent_values bigint	min_length smallint	max_length smallint	avg_length numeric	count_length bigint	min_replacement_cost numeric	max_replacement_cost numeric	avg_replacement_cost numeric	min_rental_duration smallint	max_rental_duration smallint	avg_rental_duration numeric	count_rows bigint
1	0.99	4.99	2.9820079920079920	1001	46	185	115.2720000000000000	1000	9.99	29.99	19.9840059940059940	3	7	4.9830169830169830	1001

Non-Numerical

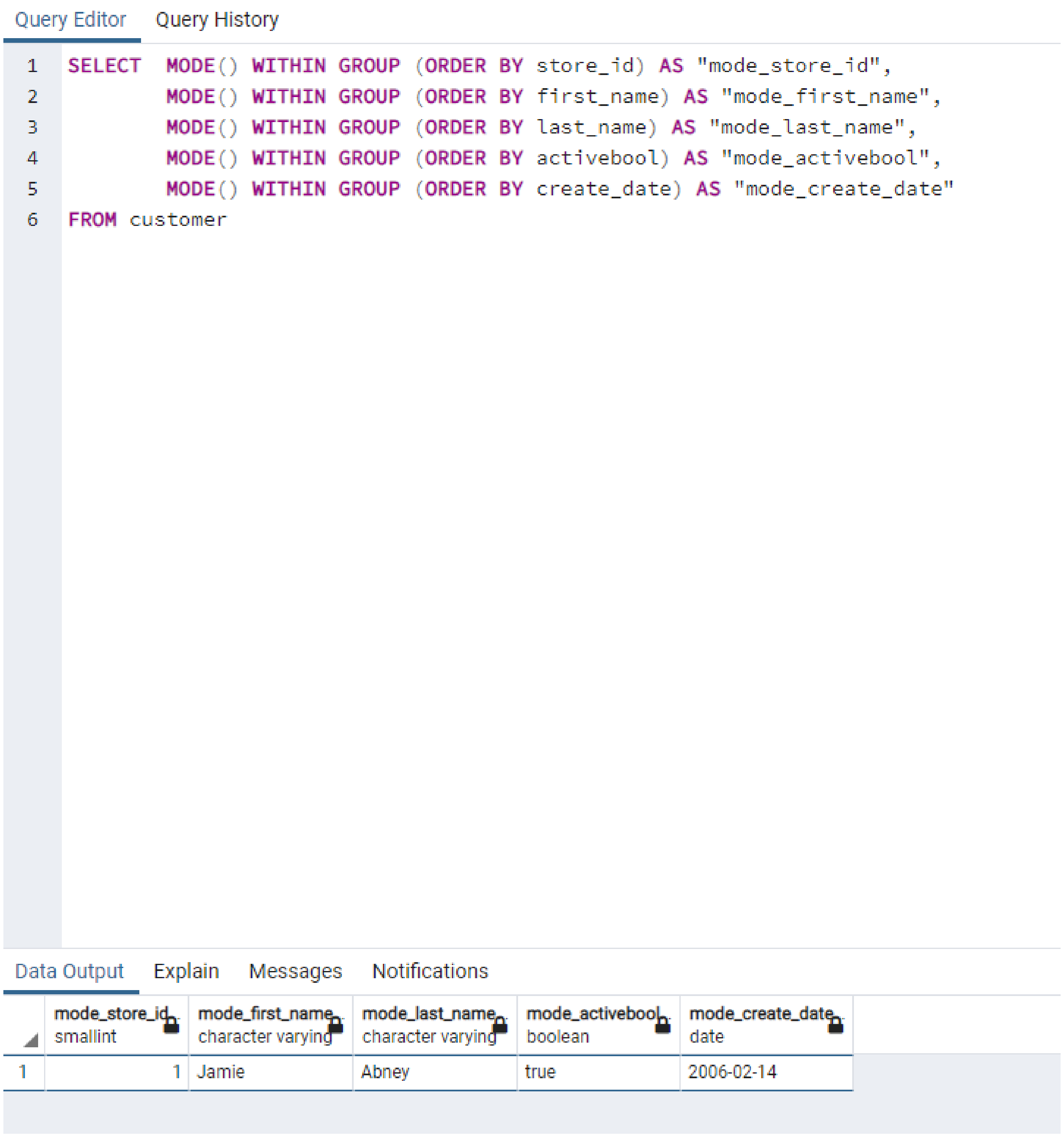


```sql
SELECT  MODE() WITHIN GROUP (ORDER BY release_year) AS "mode_release_year",
        MODE() WITHIN GROUP (ORDER BY language_id) AS "mode_language_id",
        MODE() WITHIN GROUP (ORDER BY rating) AS "mode_rating"
FROM film
```

	mode_release_year integer	mode_language_id smallint	mode_rating mpaa_rating
1	2006	1	PG-13

Customer Table

Non-Numerical



```
SELECT  MODE() WITHIN GROUP (ORDER BY store_id) AS "mode_store_id",
        MODE() WITHIN GROUP (ORDER BY first_name) AS "mode_first_name",
        MODE() WITHIN GROUP (ORDER BY last_name) AS "mode_last_name",
        MODE() WITHIN GROUP (ORDER BY activebool) AS "mode_activebool",
        MODE() WITHIN GROUP (ORDER BY create_date) AS "mode_create_date"
FROM customer
```

	mode_store_id smallint	mode_first_name character varying	mode_last_name character varying	mode_activebool boolean	mode_create_date date
1	1	Jamie	Abney	true	2006-02-14

Numerical data not necessary in customer table.

3) I'm still partial to Excel because of a) familiarity and b) I find it easier to use filter and find the results I want or to edit and manipulate the data I need vs trying to write code to do all of that. However, I know that eventually we're going to be seeing more and more tables and the next section deals with combining and joining tables, so it can eventually grow to be too large for excel to handle. That being said, so far I'm still preferring Excel.