Andrew Smith

Career Foundry Data Immersion

January 16, 2022

3.6: Summarizing & Cleaning Data in SQL (Answers 3.6)

1)

**a)**There were no duplicates found in either table but, if there were any, I would start the duplicate removal process by creating a virtual table or “view” with the same columns used in the commands below. Then I would create a duplicate deleting command with that keeps the lowest unique ID value, so the table only keeps one version of the duplicate.

**a)Duplicate command: Film table**

SELECT title,

release\_year,

language\_id,

rental\_duration,

COUNT(\*)

FROM film

GROUP BY title,

release\_year,

language\_id,

rental\_duration

HAVING COUNT(\*) >1;

**a)Duplicate command: Customer table**

SELECT customer\_id,

store\_id,

first\_name,

last\_name,

email,

COUNT(\*)

FROM customer

GROUP BY customer\_id,

store\_id,

first\_name,

last\_name,

email

HAVING COUNT(\*) >1;

**b)**Nonuniform/Incorrect data Commands

Every column appears to be uniform in both tables, but if anything was not uniform, then I would use the GROUP BY command for each category to find which piece of data is different from the rest. Then I would use an UPDATE function to change the nonuniform data to match the corresponding value.

**b)**Example GROUP BY command:

SELECT rating

FROM film

GROUP BY rating

**b)**Example UPDATE command that has nonuniform values:

UPDATE film

SET rating = 'G'

WHERE rating IN ('gen',

'g',

'General')

**c)**Missing Values Commands

There appears to be no missing data in either table, but if there was then I would have two options depending on the scenario. I could simply ignore a column that contained missing data when conducting a typical SELECT command and adding a note explaining why a column is left out. The other option would be to impute the null values with an estimate. I could do this by creating an UPDATE command where the average of a column is used to fill in the missing values.

**c)**Example SELECT command that is ignoring a column with missing values

SELECT film\_id,

title,

length... –rental\_duration ignored in select because it has a lot of missing values

FROM film

**c)**Example UPDATE command to fill in the missing values with the column averages

UPDATE film

SET = AVG(rental\_duration)

WHERE rental\_duration IS NULL

2)

**Numerical Descriptive Statistic Queries (Customer table)**

(MIN, MAX, AVG) Customer\_id, store\_id,address\_id,active

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,

min(active) AS min\_active,

max(active) AS max\_active,

avg(active) AS avg\_active

FROM customer;

Graphical user interface, text, application

Description automatically generated

EXTRA?

(MIN, MAX) Create\_date, last\_update

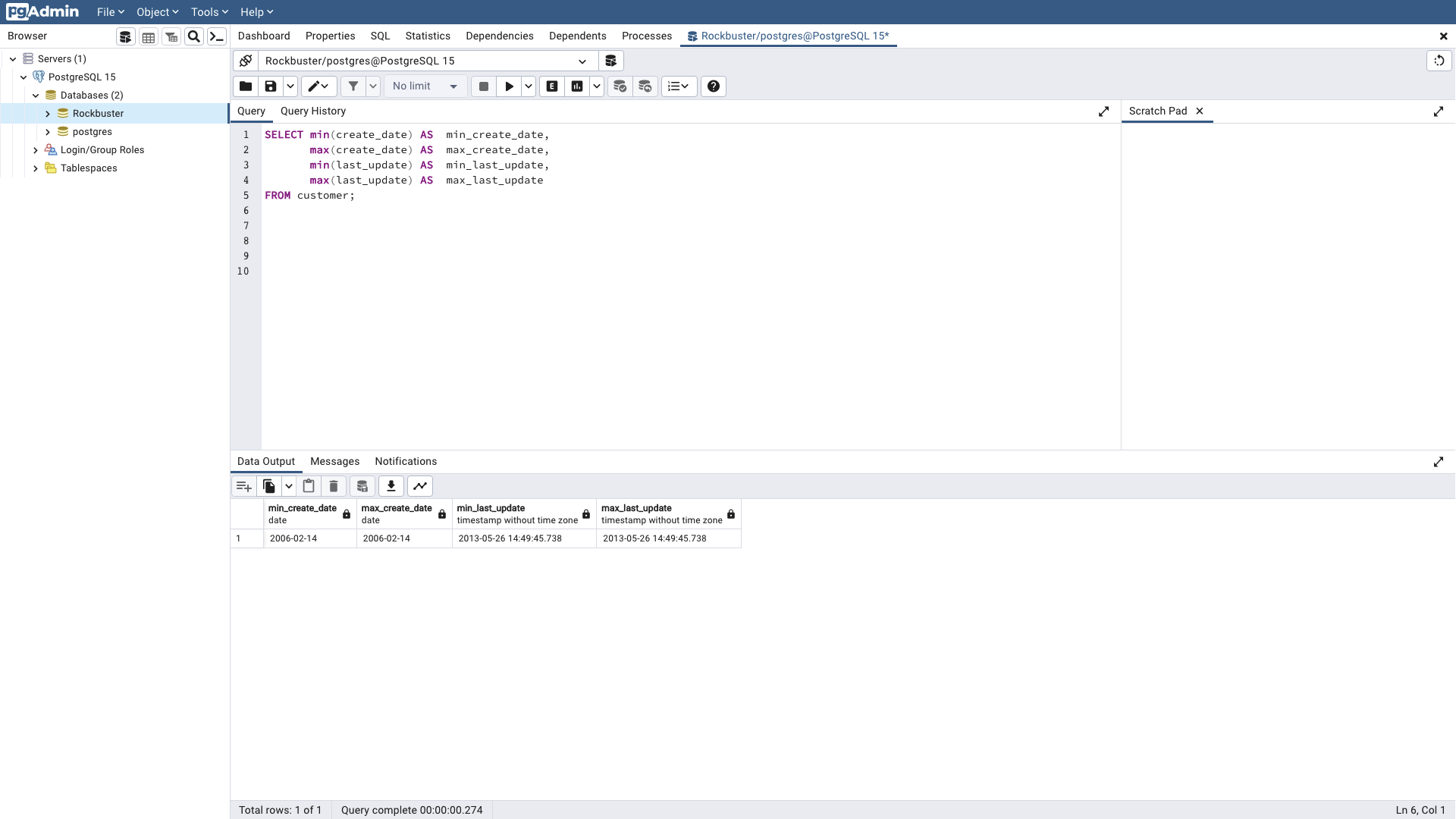
SELECT min(create\_date) AS min\_create\_date,

max(create\_date) AS max\_create\_date,

min(last\_update) AS min\_last\_update,

max(last\_update) AS max\_last\_update

FROM customer;



\*avg command did not work for the create\_date & last\_update columns

**Non-Numerical Descriptive Statistic Queries (Customer table)**

First\_name COUNT

SELECT first\_name,

COUNT(\*)

FROM customer

GROUP BY first\_name

ORDER BY COUNT DESC

Graphical user interface, application, table

Description automatically generated

First\_name MODE

SELECT mode() WITHIN GROUP (ORDER BY first\_name)

AS modal\_value

FROM customer;

Graphical user interface, text, application

Description automatically generated

Last\_name COUNT

SELECT last\_name,

COUNT(\*)

FROM customer

GROUP BY last\_name

ORDER BY COUNT DESC

Graphical user interface, application, table

Description automatically generated

Last\_name MODE

SELECT mode() WITHIN GROUP (ORDER BY last\_name)

AS modal\_value

FROM customer;

Graphical user interface, text, application

Description automatically generated

Email COUNT

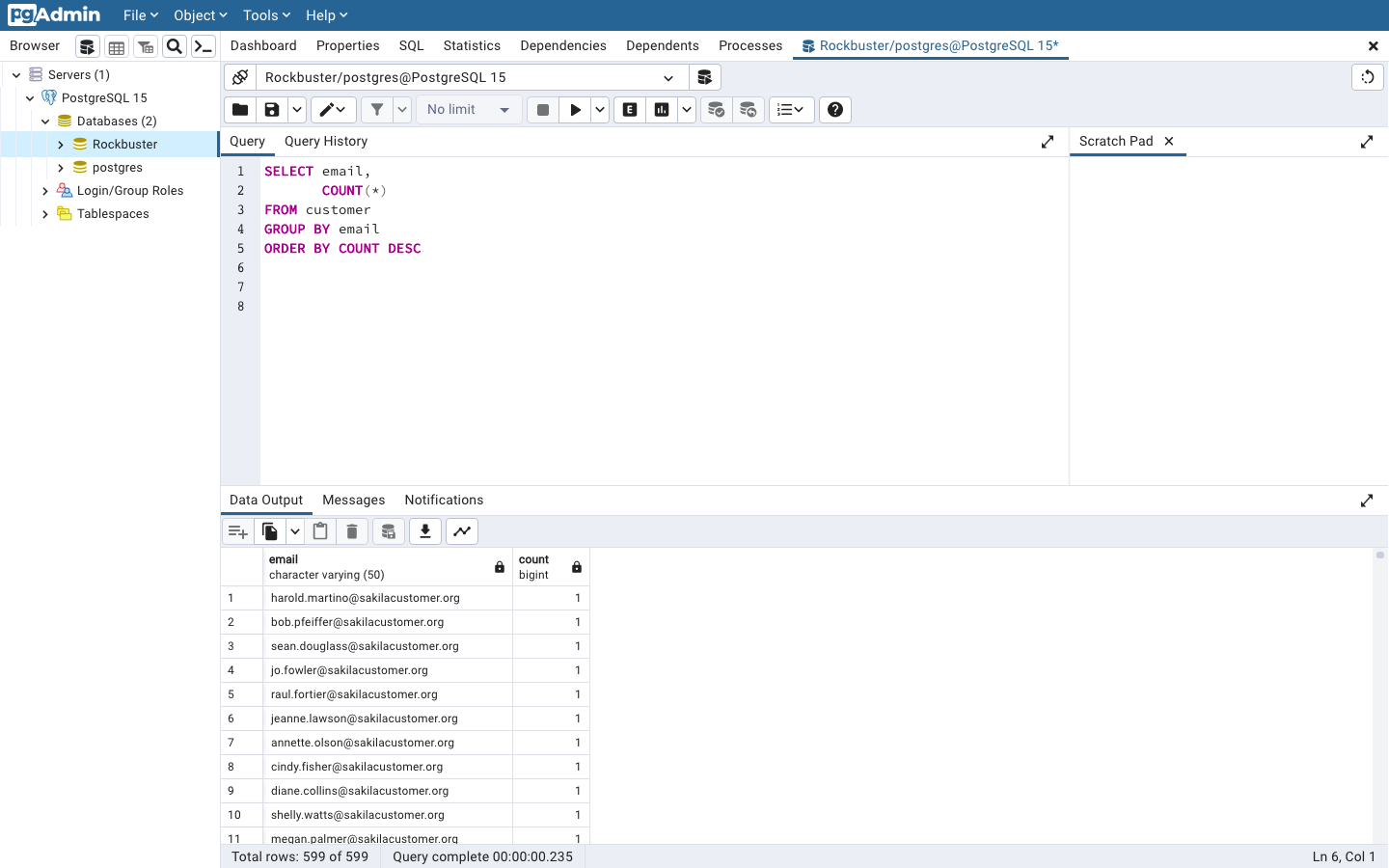
SELECT email,

COUNT(\*)

FROM customer

GROUP BY email

ORDER BY COUNT DESC



Email MODE

SELECT mode() WITHIN GROUP (ORDER BY email)

AS modal\_value

FROM customer;

Graphical user interface, text, application

Description automatically generated

Activebool COUNT

SELECT activebool,

COUNT(\*)

FROM customer

GROUP BY activebool

ORDER BY COUNT DESC

Graphical user interface, application

Description automatically generated

Activebool MODE

SELECT mode() WITHIN GROUP (ORDER BY activebool)

AS modal\_value

FROM customer;

Graphical user interface, text, application

Description automatically generated

EXTRA?

Create\_date COUNT

SELECT create\_date,

COUNT(\*)

FROM customer

GROUP BY create\_date

ORDER BY COUNT DESC

Graphical user interface, application

Description automatically generated

Create\_date MODE

SELECT mode() WITHIN GROUP (ORDER BY create\_date)

AS modal\_value

FROM customer;

Graphical user interface, text, application

Description automatically generated

Last\_update COUNT

SELECT last\_update,

COUNT(\*)

FROM customer

GROUP BY last\_update

ORDER BY COUNT DESC

Graphical user interface, text, application

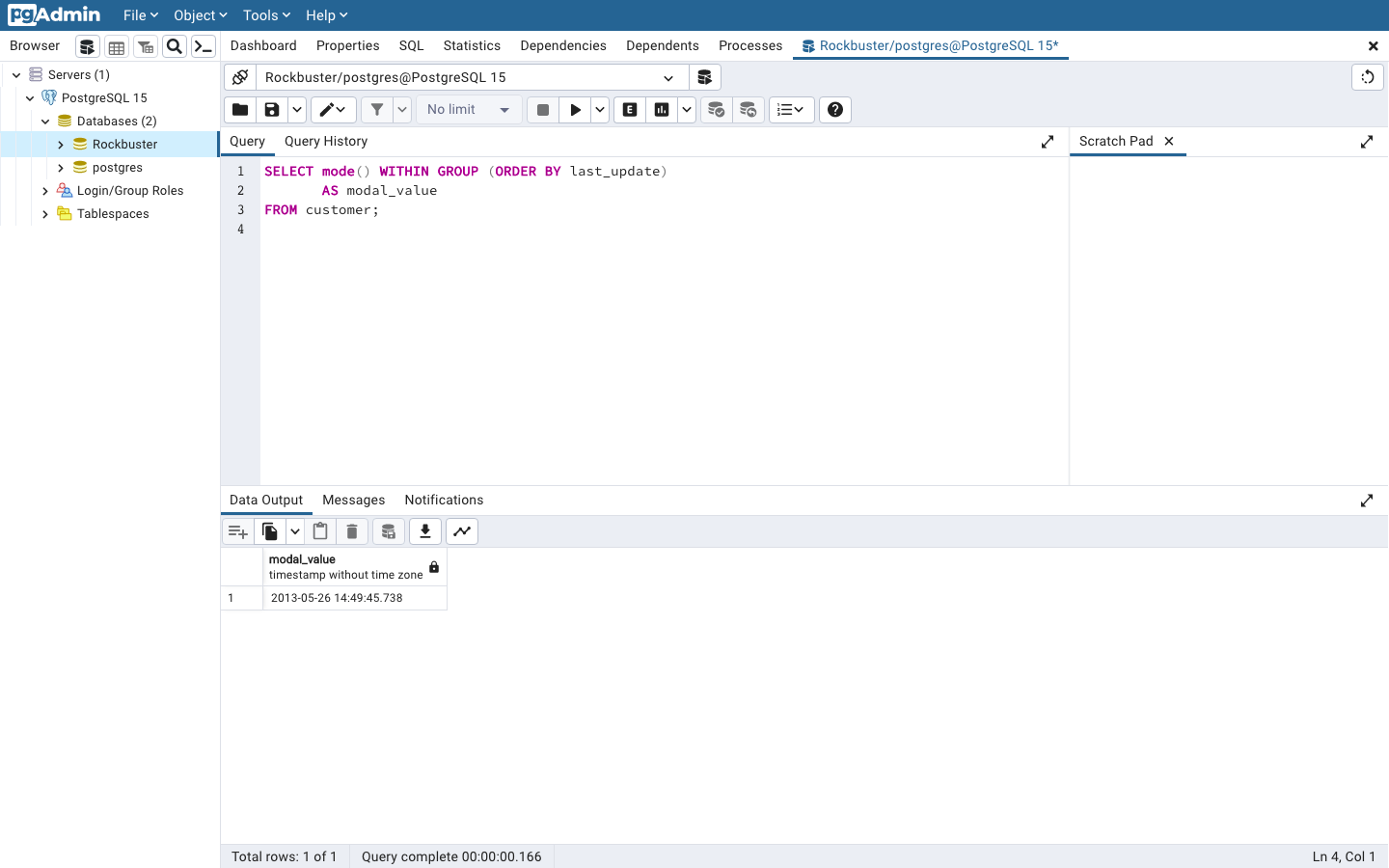
Description automatically generated

Last\_update MODE

SELECT mode() WITHIN GROUP (ORDER BY last\_update)

AS modal\_value

FROM customer;



**\*\*\*\*Numerical Descriptive Statistic Queries (Film table)**

(MIN, MAX, AVG) film\_id, release\_year, language\_id, rental\_duration, rental\_rate, length, replacement cost

SELECT MIN(film\_id) AS min\_film\_id,

MAX(film\_id) AS max\_film\_id,

AVG(film\_id) AS avg\_film\_id,

MIN(release\_year) AS min\_release\_year,

MAX(release\_year) AS max\_release\_year,

AVG(release\_year) AS avg\_release\_year,

MIN(language\_id) AS min\_language\_id,

MAX(language\_id) AS max\_language\_id,

AVG(language\_id) AS avg\_language\_id,

MIN(rental\_duration) AS min\_rental\_duration,

MAX(rental\_duration) AS max\_rental\_duration,

AVG(rental\_duration) AS avg\_rental\_duration,

MIN(rental\_rate) AS min\_rental\_rate,

MAX(rental\_rate) AS max\_rental\_rate,

AVG(rental\_rate) AS avg\_rental\_rate,

MIN(length) AS min\_length,

MAX(length) AS max\_length,

AVG(length) AS avg\_length,

MIN(replacement\_cost) AS min\_replacement\_cost,

MAX(replacement\_cost) AS max\_replacement\_cost,

AVG(replacement\_cost) AS avg\_replacement\_cost

FROM film;

Graphical user interface, text, application

Description automatically generated

Graphical user interface, text, application, Word

Description automatically generated

Graphical user interface, text, application

Description automatically generated

-There are three screenshots since I had to scroll through the query/query output to show all results.

EXTRA?

(MIN, MAX) last\_update

SELECT min(last\_update) AS min\_last\_update,

max(last\_update) AS max\_last\_update

FROM film;

Graphical user interface, text, application

Description automatically generated

\*avg command did not work for the last\_update column

**Non-Numerical Descriptive Statistic Queries (Film table)**

(COUNT & MODE) Title, rating, description, special\_features, fulltext

Title COUNT

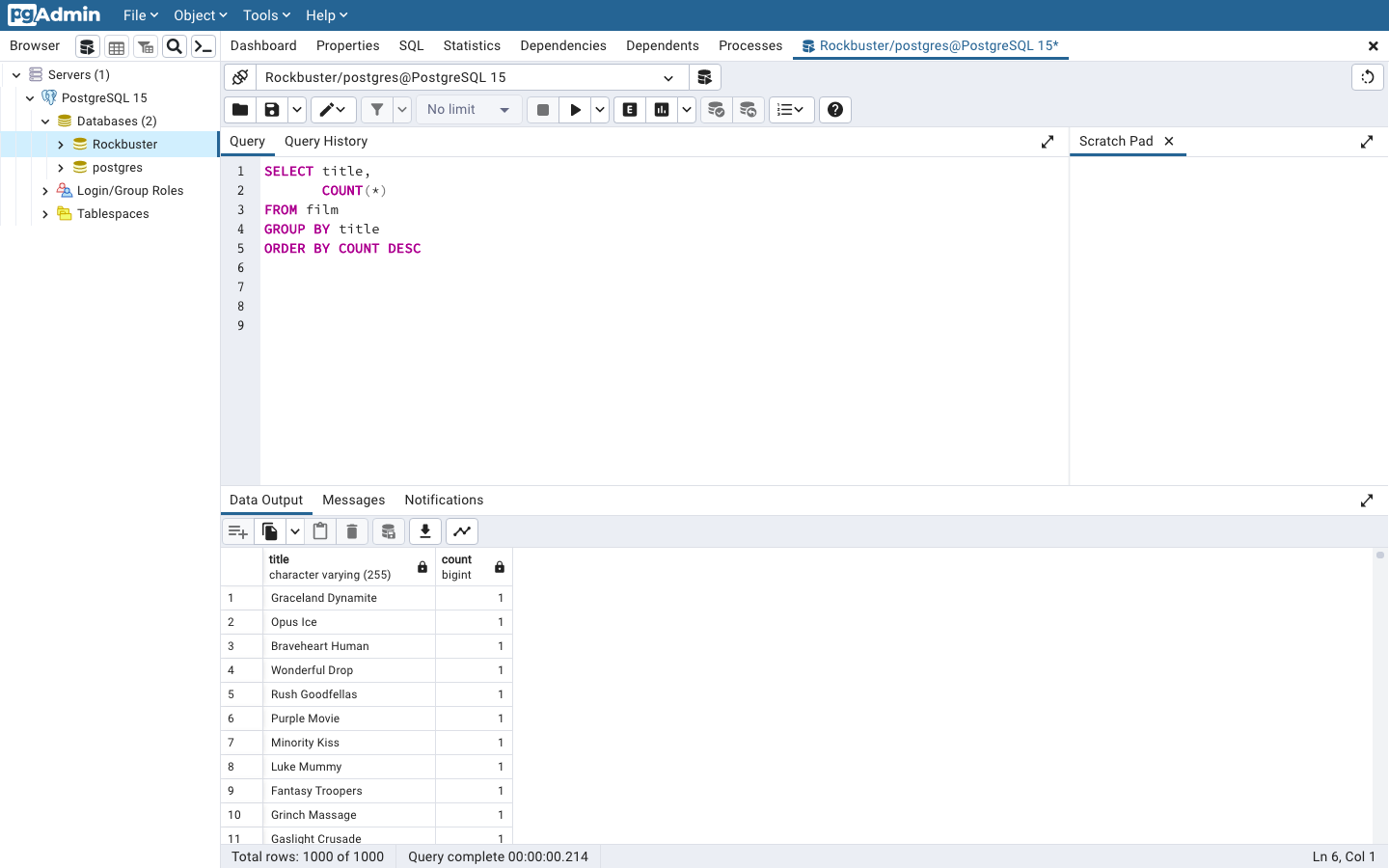
SELECT title,

COUNT(\*)

FROM film

GROUP BY title

ORDER BY COUNT DESC



Title MODE

SELECT mode() WITHIN GROUP (ORDER BY title)

AS modal\_value

FROM film;

Graphical user interface, text, application

Description automatically generated

Rating COUNT

SELECT rating,

COUNT(\*)

FROM film

GROUP BY rating

ORDER BY COUNT DESC

Graphical user interface

Description automatically generated

Rating MODE

SELECT mode() WITHIN GROUP (ORDER BY rating)

AS modal\_value

FROM film;

Graphical user interface, text, application

Description automatically generated

Description COUNT

SELECT description,

COUNT(\*)

FROM film

GROUP BY description

ORDER BY COUNT DESC

Graphical user interface, text, application

Description automatically generated

Description MODE

SELECT mode() WITHIN GROUP (ORDER BY description)

AS modal\_value

FROM film;

Graphical user interface, text, application

Description automatically generated

Special\_features COUNT

SELECT special\_features,

COUNT(\*)

FROM film

GROUP BY special\_features

ORDER BY COUNT DESC

Graphical user interface, text, application

Description automatically generated

Special\_features MODE

SELECT mode() WITHIN GROUP (ORDER BY special\_features)

AS modal\_value

FROM film;

Graphical user interface, text, application

Description automatically generated

Fulltext COUNT

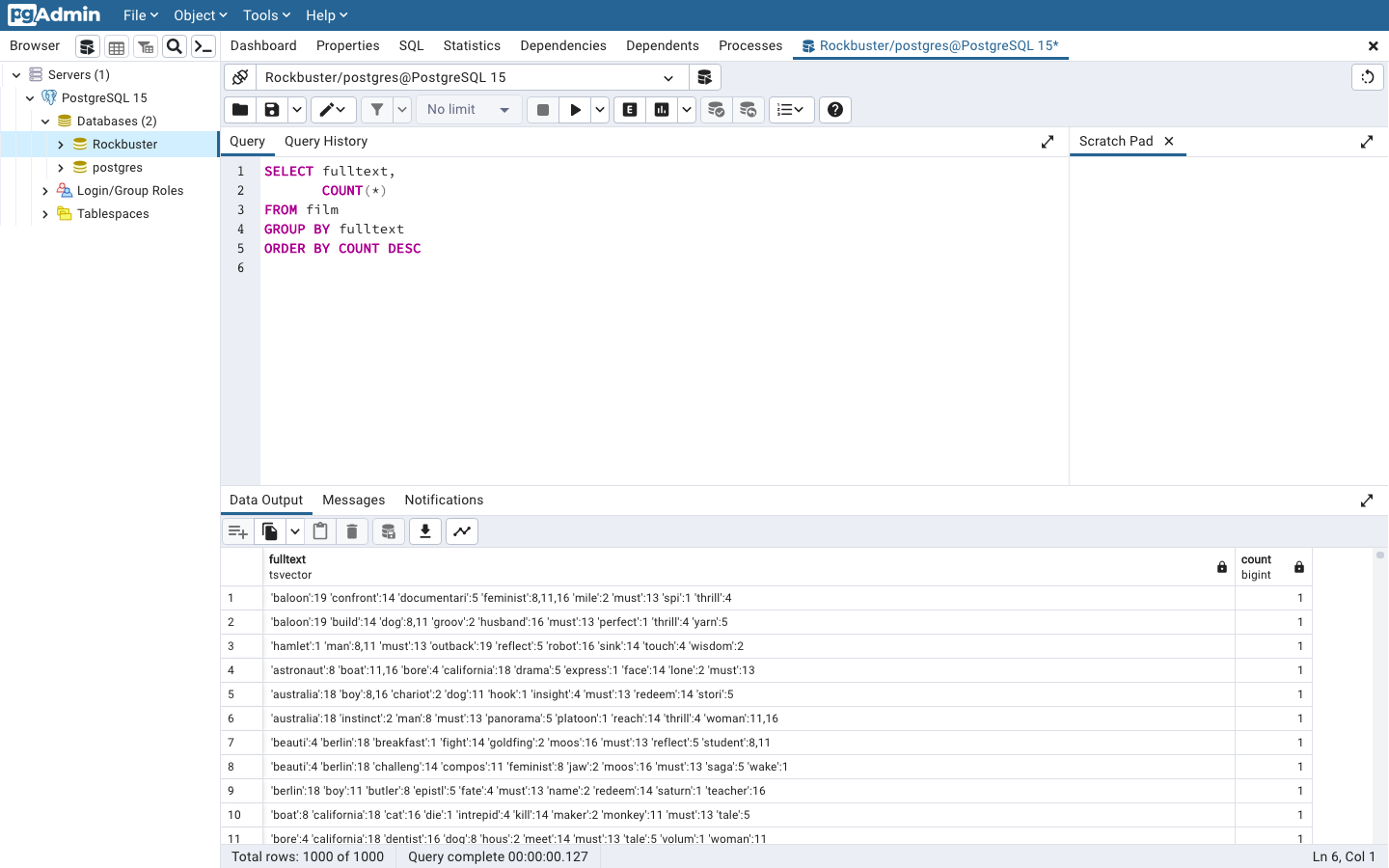
SELECT fulltext,

COUNT(\*)

FROM film

GROUP BY fulltext

ORDER BY COUNT DESC



Fulltext MODE

SELECT mode() WITHIN GROUP (ORDER BY fulltext)

AS modal\_value

FROM film;

Graphical user interface, text, application

Description automatically generated

EXTRA?

Last\_update COUNT

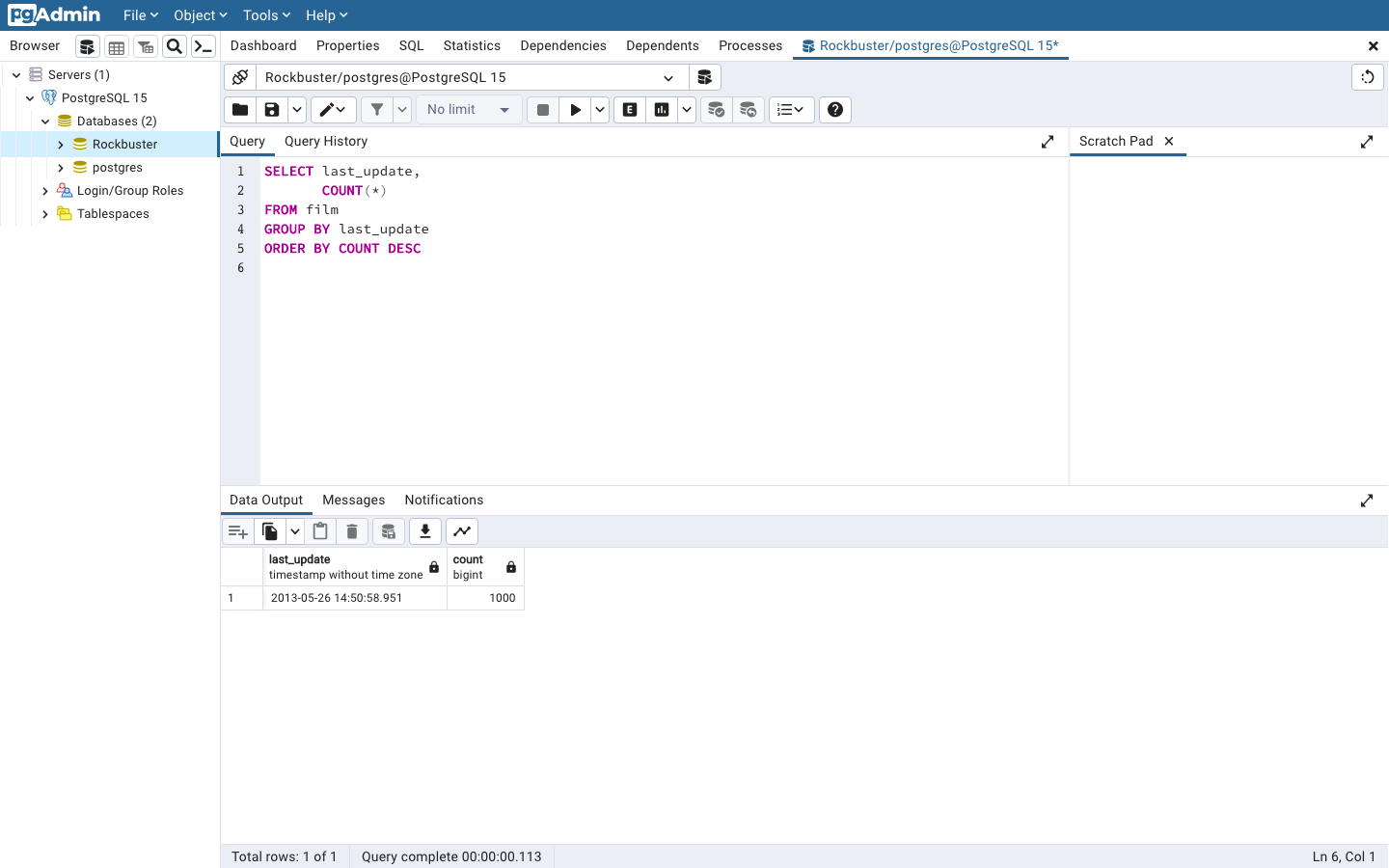
SELECT last\_update,

COUNT(\*)

FROM film

GROUP BY last\_update

ORDER BY COUNT DESC



Last\_update MODE

SELECT mode() WITHIN GROUP (ORDER BY last\_update)

AS modal\_value

FROM film;

Graphical user interface, text, application

Description automatically generated

3)

I think SQL is much more effective for data profiling compared to Excel for a few different reasons. I find it is easier and faster to type a command in SQL to calculate descriptive statistics rather than having to highlight columns/rows in Excel and then typing a formula. SQL is especially helpful for performing multiple descriptive statistics at once rather than inputting each formula at a time in Excel.