

## Data Analytics Immersion

### 3.6: Summarizing & Cleaning Data in SQL

Mehreen Becker

Rockbuster's database engineers have loaded some new data into the database, and your manager has asked you to clean and profile it. Follow the instructions below to complete their request:

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)

The screenshot shows the pgAdmin 4 interface. The left sidebar is titled 'Object Explorer' and lists various database objects: Foreign Tables, Functions, Materialized Views, Operators, Procedures, Sequences, and Tables (16). The 'Tables (16)' section is expanded, showing tables like Actor, address, category, city, country, customer, film, film\_actor, film\_category, inventory, language, payment, rental, staff, store, Trigger Functions, Types, and Views. Below the Object Explorer is a 'Processes' tab. The main area is the 'Query Editor' with the following SQL code:

```
1 select title, release_year, language_id, rental_duration
2 from film
3 group by title, release_year, language_id, rental_duration
4 having count (*) >1
```

Below the query editor is a 'Data Output' pane showing the schema of the result:

title	release_year	language_id	rental_duration
character varying (255)	integer	smallint	smallint

At the bottom of the pgAdmin window, status bars show 'Total rows: 0' and 'Query complete 00:00:00.278'. A green message bar at the bottom right says 'Successfully run. Total query runtime: 278 msec. 0 rows affected.'

I would first identify duplicate records by analyzing key fields (SELECT column, COUNT (\*) FROM table GROUP BY column HAVING COUNT (\*) >1) to see where the same values appear more than once and then I would decide which records to keep and remove the extra duplicates to ensure only unique and accurate data remains.

pgAdmin 4 Object Tools Edit View Window Help pgAdmin 4 Sun 19 Oct 16:49

Object Explorer Processes Testing/postgres... Testing/postgres... postgres/postgres... Preferences Rockbuster/postgres@My Server\*

```

1 SELECT *
2   from film
3  where film_id is null
4  or title is null
5  or description is null
6  or release_year is null
7  or language_id is null
8  or rental_duration is null
9  or rental_rate is null
10 or length is null
11 or replacement_cost is null
12 or rating is null
13 or last_update is null
14 or special_features is null

```

Data Output Messages Notifications

film_id	title	description	release_year	language_id	rental_duration	rental_rate	length	replacement_cost	rating	mpaa_rating	last_update

Total rows: 0 Query complete 00:00:00.090 LF Ln 14, Col 28

pgAdmin 4 Object Tools Edit View Window Help pgAdmin 4 Sun 19 Oct 20:37

Object Explorer Processes Testing/postgres... Testing/postgres... postgres/postgres... Preferences Rockbuster/postgres@My Server\*

```

1 SELECT rental_duration FROM film GROUP BY rental_duration

```

Data Output Messages Notifications

rental_duration
1
2
3
4
5

Showing rows: 1 to 5 Page No: 1 of 1 ✓ Successfully run. Total query runtime: 265 msec. 5 rows affected. LF Ln 1, Col 1

Total rows: 5 Query complete 00:00:00.295

The screenshot shows the pgAdmin 4 interface with the following details:

- Object Explorer:** Shows a tree view of database objects including Foreign Tables, Functions, Materialized Views, Procedures, Sequences, and Tables (16). The Tables node is expanded, showing Actor, address, category, city, country, customer, film, film\_actor, film\_category, inventory, language, payment, rental, staff, store, and trigger functions.
- Query Editor:** Displays the SQL query: `SELECT release_year FROM film GROUP BY release_year`.
- Data Output:** Shows the results of the query in a table format. The table has one column, `release_year`, with the value 2006.
- Status Bar:** Shows "Total rows: 1" and "Query complete 00:00:00.078".

The screenshot shows the pgAdmin 4 interface with the following details:

- Object Explorer:** Shows a tree view of database objects including Foreign Tables, Functions, Materialized Views, Operators, Procedures, Sequences, and Tables (16). The Tables node is expanded, showing Actor, address, category, city, country, customer, film, film\_actor, film\_category, inventory, language, payment, rental, staff, store, and trigger functions.
- Query Editor:** Displays three consecutive SQL queries:
  - `SELECT release_year FROM film GROUP BY release_year`
  - `SELECT rental_rate FROM film GROUP BY rental_rate`
  - `SELECT rental_duration FROM film GROUP BY rental_duration`
- Data Output:** Shows the results of the second query in a table format. The table has one column, `rental_rate`, with values 2.99, 4.99, and 0.99.
- Status Bar:** Shows "Total rows: 3" and "Query complete 00:00:00.073". A green message bar at the bottom right says "Successfully run. Total query runtime: 73 msec. 3 rows affected."

## Customer Table

pgAdmin 4 Object Tools Edit View Window Help pgAdmin 4 Sun 19 Oct 20:33

Object Explorer Tables (16) Query History

```
1 SELECT
2   customer_id, store_id, first_name, last_name, email,
3   COUNT (*)
4   FROM customer GROUP BY customer_id, store_id, first_name, last_name, email
5   HAVING COUNT (*) >1
```

Data Output Messages Notifications

customer_id	store_id	first_name	last_name	email	count
-------------	----------	------------	-----------	-------	-------

Total rows: 0 Query complete 00:00:00.075 ✓ Successfully run. Total query runtime: 75 msec. 0 rows affected. LF Ln 2, Col 53

pgAdmin 4 Object Tools Edit View Window Help pgAdmin 4 Sun 19 Oct 20:28

Object Explorer Tables (16) Query History

```
1 SELECT *
2   FROM customer
3   WHERE customer_id IS NULL
4   OR store_id IS NULL
5   OR first_name IS NULL
6   OR last_name IS NULL
7   OR email IS NULL
8   OR address_id IS NULL
9   OR activebool IS NULL
10  OR create_date IS NULL
11  OR last_update IS NULL
12  OR active IS NULL
```

Data Output Messages Notifications

customer_id	store_id	first_name	last_name	email	address_id	activebool	create_date	last_update
-------------	----------	------------	-----------	-------	------------	------------	-------------	-------------

Total rows: 0 Query complete 00:00:00.278 ✓ Successfully run. Total query runtime: 278 msec. 0 rows affected. LF Ln 11, Col 3

pgAdmin 4 Object Tools Edit View Window Help pgAdmin 4 Sun 19 Oct 20:47

Object Explorer Tables (16) Query History

```
1 SELECT customer_id
2 FROM customer
3 GROUP BY customer_id
```

Data Output Messages Notifications

customer_id
184
87
477
273
550
394
51
272
70

Total rows: 599 Query complete 00:00:00.228 LF Ln 1, Col 19

pgAdmin 4 Object Tools Edit View Window Help pgAdmin 4 Sun 19 Oct 20:42

Object Explorer Tables (16) Query History

```
1 SELECT customer_id, store_id, first_name, last_name, activebool, last_update
2 FROM customer
3 GROUP BY customer_id
```

Data Output Messages Notifications

customer_id	store_id	first_name	last_name	activebool	last_update
184	1	Vivian	Ruiz	true	2013-05-26 14:49:45.738
87	1	Wanda	Patterson	true	2013-05-26 14:49:45.738
477	1	Dan	Paine	true	2013-05-26 14:49:45.738
273	2	Priscilla	Lowe	true	2013-05-26 14:49:45.738
550	2	Guy	Brownlee	true	2013-05-26 14:49:45.738
394	2	Chris	Brothers	true	2013-05-26 14:49:45.738
51	1	Alice	Stewart	true	2013-05-26 14:49:45.738
272	1	Kay	Caldwell	true	2013-05-26 14:49:45.738
70	2	Christina	Ramirez	true	2013-05-26 14:49:45.738

Total rows: 599 Query complete 00:00:00.093 ✓ Successfully run. Total query runtime: 93 msec. 599 rows affected. LF Ln 3, Col 21

- 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.

```

1 SELECT
2   --rental duration
3   min (rental_duration) as minimum_rental_duration,
4   max (rental_duration) as maximum_rental_duration,
5   avg (rental_duration) as average_rental_duration,
6   count (rental_duration) as count_rental_duration,
7   --rental rate
8   min (rental_rate) as minimum_rental_rate,
9   max (rental_rate) as maximum_rental_rate,
10  avg (rental_rate) as average_rental_rate,
11  count (rental_rate) as count_rental_rate,
12  --replacement cost
13  min (replacement_cost) as minimum_replacement_cost,
14  max (replacement_cost) as maximum_replacement_cost,
15  avg (replacement_cost) as average_replacement_cost,
16  count (replacement_cost) as count_replacement_cost,
17  --release year
18  min (release_year) as minimum_release_year,
19  max (release_year) as maximum_release_year,
20  avg (release_year) as average_release_year,
21  count (release_year) as count_release_year
22  from film;

```

Data Output Messages Notifications

minimum_rental_duration	maximum_rental_duration	average_rental_duration	count_rental_duration
3	7	4.985000000000000	1000

Total rows: 1    Query complete 00:00:00.066    ✓ Successfully run. Total query runtime: 66 msec. 1 row    LF    Ln 21, Col 43

Rental Duration

minimum_rental_duration	maximum_rental_duration	average_rental_duration	count_rental_duration
3	7	4.985000000000000	1000

Rental Rate

minimum_rental_rate	maximum_rental_rate	average_rental_rate	count_rental_rate
0.99	4.99	2.980000000000000	1000

Replacement Cost

minimum_replacement_cost	maximum_replacement_cost	average_replacement_cost	count_replacement_cost
9.99	29.99	19.984000000000000	1000

Release Year

minimum_release_year	maximum_release_year	average_release_year	count_release_year
2006	2006	2006.000000000000000	1000

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.

From my experience, as someone who has been working with SQL for about a week, I think Excel feels more straightforward for basic data profiling since it's easy to filter, sort and use quick functions on (like average etc) without having to write out the queries. However, SQL feels more powerful and is generally faster when it comes to handling

larger datasets. I'm sure that when I learn more and am more familiar with the commands, I will be able to profile data more efficiently.