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).

Duplicate Data from Film

Query | Query History

```
SELECT  film_ID, title, rental_rate,
                release_year,
                language_id,
                rental_duration
FROM film
GROUP BY film_ID, title, rental_rate,
                release_year,
                language_id,
                rental_duration
                HAVING COUNT(*) >1
```

Data Output | Messages | Notifications

film_id [PK] integer	title character varying (255)	rental_rate numeric (4,2)	release_year integer	language_id smallint	rental_duration smallint

Duplicate Data from Customer

Query | Query History | Scratch

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

Data Output | Messages | Notifications

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

Non-Uniform Data

We can use GROUP BY or DISTINCT to select unique records.

Query | Query History

```
1  SELECT rating
2  FROM film
3  GROUP BY rating
```

Data Output | Messages | Notifications

	rating mpaa_rating
1	PG
2	R
3	NC-17
4	PG-13
5	G

There were no duplicate or non-uniform values in the film or customer table. But if the results were different and you were given permission to alter the database you could create a virtual table "View" select the unique records and delete the duplicate records from there.

If there were non-uniform data we could fix this issue with UPDATE command combined with SET and WHERE, to replace the values that should be differently represented.

For incorrect data, a mix of the commands shown before can also help, by grouping and organizing the information in a more visible way for the analyst to review it.

For missing data, in some cases where a column has too much information missing, it might even be valid to remove it from queries or **impute values** is to "fill in" missing values with estimates. Below is the SQL syntax for imputing missing values with a column average:

UPDATE tablename

SET = AVG(col1)

WHERE col1 IS NULL

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 customer --descriptive statistics for customer table

```
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(create_date) AS min_create_date,
MAX(create_date) AS max_create_date,
MODE() WITHIN GROUP (ORDER BY create_date) AS create_date,
MIN(last_update) AS min_last_update,
MAX(last_update) AS max_last_update,
```

MODE() WITHIN GROUP (ORDER BY last_update) AS last_update,

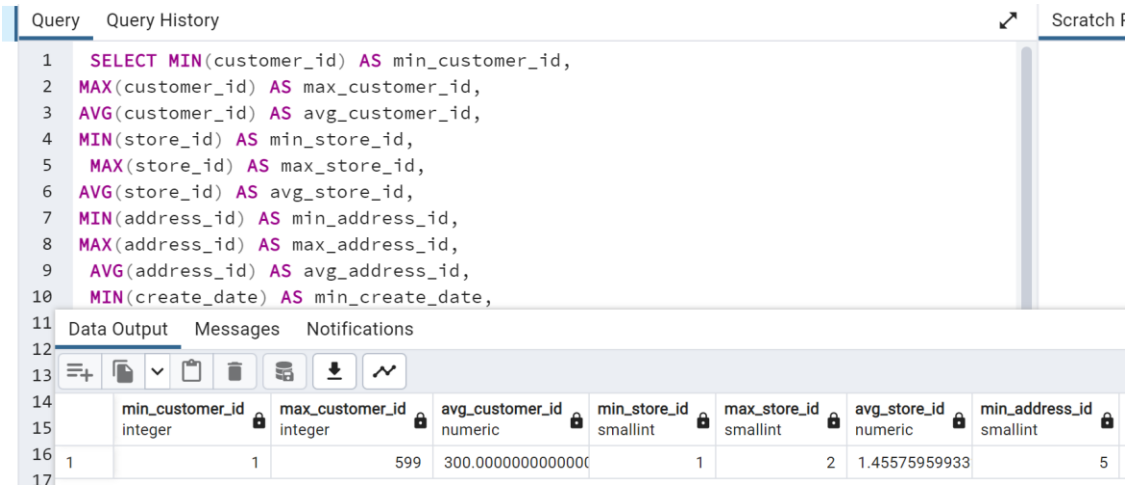
MODE() WITHIN GROUP (ORDER BY first_name) AS first_name,

MODE() WITHIN GROUP (ORDER BY last_name) AS last_name,

MODE() WITHIN GROUP (ORDER BY email) AS email,

MODE() WITHIN GROUP (ORDER BY create_date) AS create_date,

MODE() WITHIN GROUP (ORDER BY active) AS mode_active FROM customer;



Summary for film --descriptive statistics for film table

SELECT MIN(rental_rate) AS min_renatl_rate,

MAX(rental_rate) AS max_rental_rate,

AVG(rental_rate) AS avg_renatal_rate,

MIN(rental_duration) AS min_rental_duration,

MAX(rental_duration) AS max_rental_duration,

AVG(rental_duration) AS avg_rental_duration,

MIN(film_id) AS min_film, MAX(film_id) AS max_film,

AVG(film_id) AS avg_film, MIN(language_id) AS min_language,

MAX(language_id) AS max_language,

AVG(language_id) AS avg_language,

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,

MODE() WITHIN GROUP (ORDER BY rating) AS rating_value,

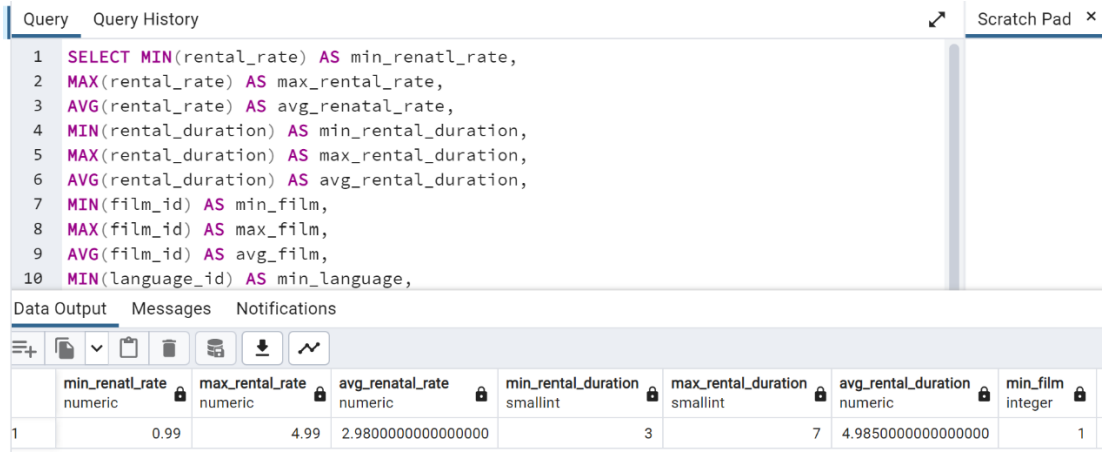
MODE() WITHIN GROUP (ORDER BY special_features) AS feature_value,

MODE() WITHIN GROUP (ORDER BY release_year) AS release_year,

MODE() WITHIN GROUP (ORDER BY title) AS title_value,

MODE() WITHIN GROUP (ORDER BY fulltext) AS fulltext

FROM film



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.

When it comes to a small amount of data, you can use excel. When it comes to a large amount of data SQL is much better suited for data profiling. With SQL once the query has been written it can be applied time and again without much effort.