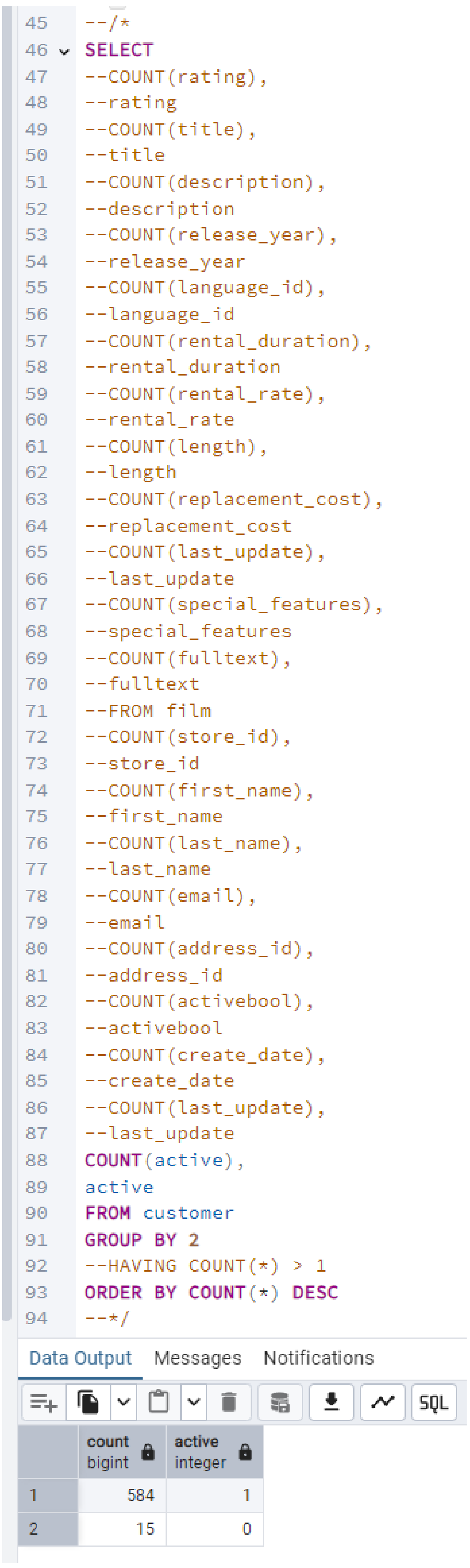
Answers 3.6

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).

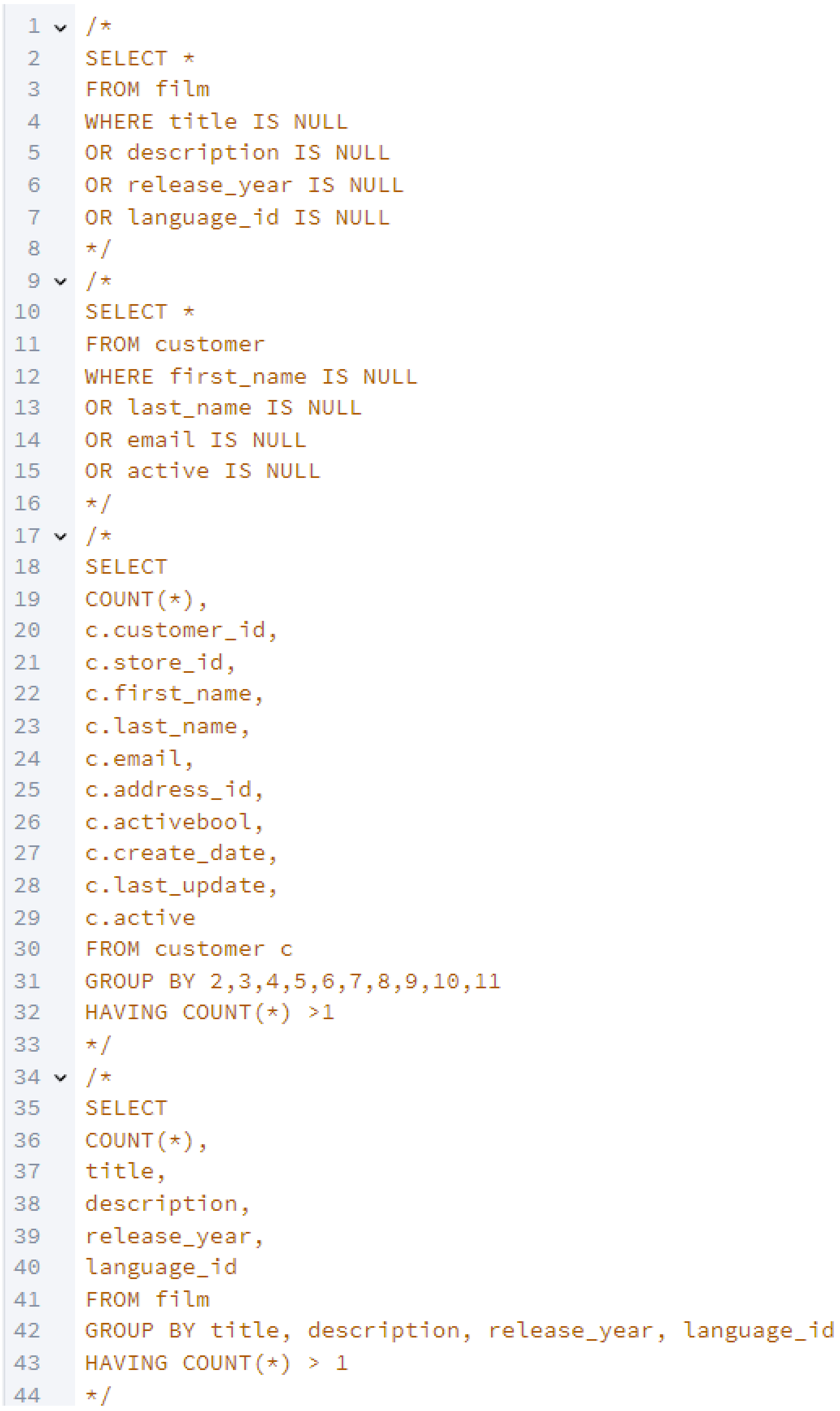
- Manual check one by one for non-uniform



```
--/*
SELECT
--COUNT(rating),
--rating
--COUNT(title),
--title
--COUNT(description),
--description
--COUNT(release_year),
--release_year
--COUNT(language_id),
--language_id
--COUNT(rental_duration),
--rental_duration
--COUNT(rental_rate),
--rental_rate
--COUNT(length),
--length
--COUNT(replacement_cost),
--replacement_cost
--COUNT(last_update),
--last_update
--COUNT(special_features),
--special_features
--COUNT(fulltext),
--fulltext
--FROM film
--COUNT(store_id),
--store_id
--COUNT(first_name),
--first_name
--COUNT(last_name),
--last_name
--COUNT(email),
--email
--COUNT(address_id),
--address_id
--COUNT(activebool),
--activebool
--COUNT(create_date),
--create_date
--COUNT(last_update),
--last_update
COUNT(active),
active
FROM customer
GROUP BY 2
--HAVING COUNT(*) > 1
ORDER BY COUNT(*) DESC
--*/
```

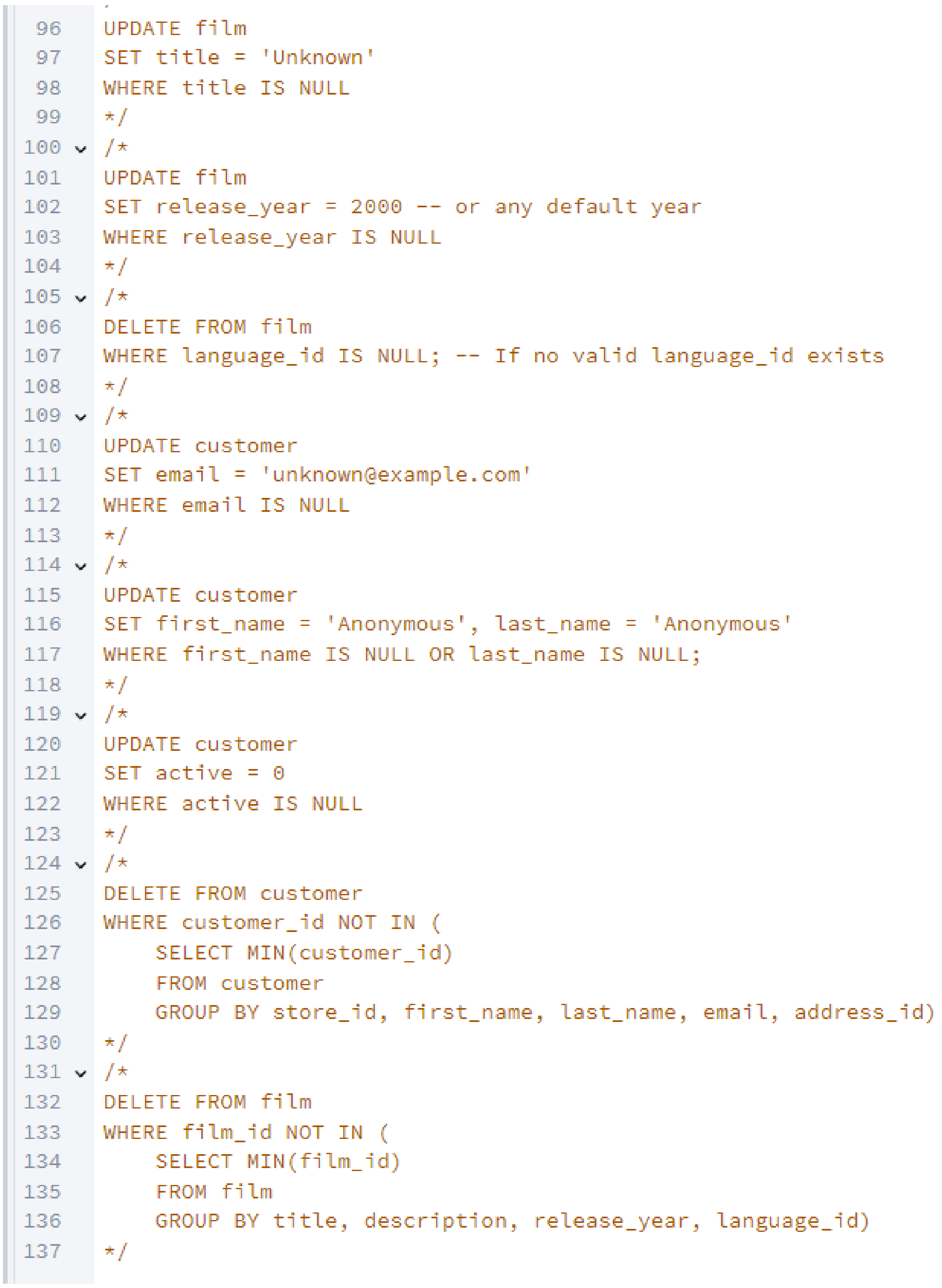
	count bigint	active integer
1	584	1
2	15	0

- duplicate data, or missing values



```
/*
SELECT *
FROM film
WHERE title IS NULL
OR description IS NULL
OR release_year IS NULL
OR language_id IS NULL
*/
/*
SELECT *
FROM customer
WHERE first_name IS NULL
OR last_name IS NULL
OR email IS NULL
OR active IS NULL
*/
/*
SELECT
COUNT(*),
c.customer_id,
c.store_id,
c.first_name,
c.last_name,
c.email,
c.address_id,
c.activebool,
c.create_date,
c.last_update,
c.active
FROM customer c
GROUP BY 2,3,4,5,6,7,8,9,10,11
HAVING COUNT(*) >1
*/
/*
SELECT
COUNT(*),
title,
description,
release_year,
language_id
FROM film
GROUP BY title, description, release_year, language_id
HAVING COUNT(*) > 1
*/
```

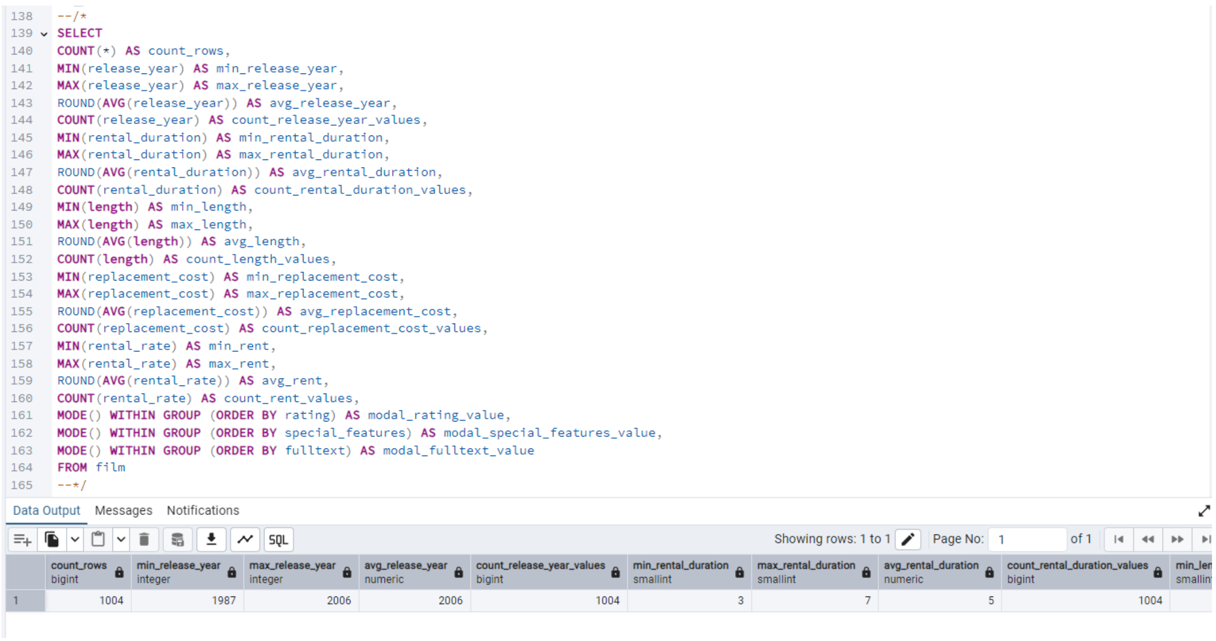
- Query for the potential fix:



```sql
UPDATE film
SET title = 'Unknown'
WHERE title IS NULL
*/
/*
UPDATE film
SET release_year = 2000 -- or any default year
WHERE release_year IS NULL
*/
/*
DELETE FROM film
WHERE language_id IS NULL; -- If no valid language_id exists
*/
/*
UPDATE customer
SET email = 'unknown@example.com'
WHERE email IS NULL
*/
/*
UPDATE customer
SET first_name = 'Anonymous', last_name = 'Anonymous'
WHERE first_name IS NULL OR last_name IS NULL;
*/
/*
UPDATE customer
SET active = 0
WHERE active IS NULL
*/
/*
DELETE FROM customer
WHERE customer_id NOT IN (
    SELECT MIN(customer_id)
    FROM customer
    GROUP BY store_id, first_name, last_name, email, address_id)
*/
/*
DELETE FROM film
WHERE film_id NOT IN (
    SELECT MIN(film_id)
    FROM film
    GROUP BY title, description, release_year, language_id)
*/
```

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.

- Film table



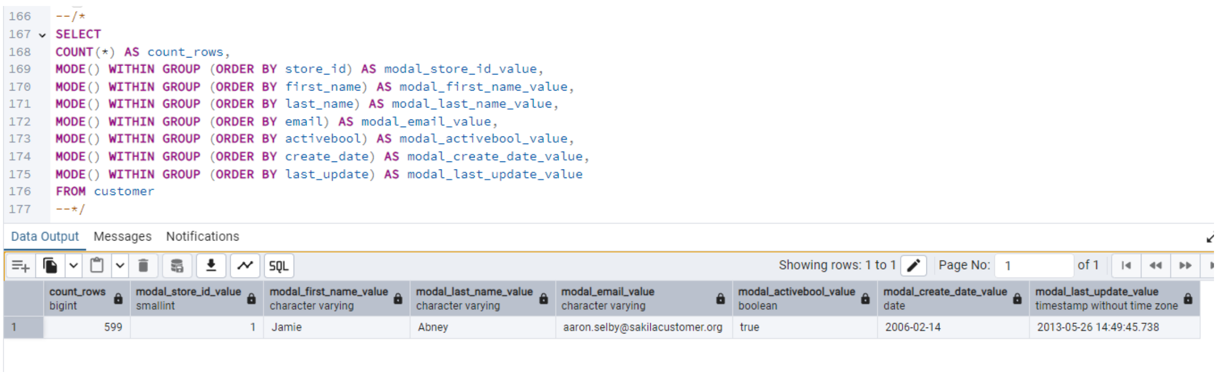
```
--/*
SELECT
COUNT(*) AS count_rows,
MIN(release_year) AS min_release_year,
MAX(release_year) AS max_release_year,
ROUND(AVG(release_year)) AS avg_release_year,
COUNT(release_year) AS count_release_year_values,
MIN(rental_duration) AS min_rental_duration,
MAX(rental_duration) AS max_rental_duration,
ROUND(AVG(rental_duration)) AS avg_rental_duration,
COUNT(rental_duration) AS count_rental_duration_values,
MIN(length) AS min_length,
MAX(length) AS max_length,
ROUND(AVG(length)) AS avg_length,
COUNT(length) AS count_length_values,
MIN(replacement_cost) AS min_replacement_cost,
MAX(replacement_cost) AS max_replacement_cost,
ROUND(AVG(replacement_cost)) AS avg_replacement_cost,
COUNT(replacement_cost) AS count_replacement_cost_values,
MIN(rental_rate) AS min_rent,
MAX(rental_rate) AS max_rent,
ROUND(AVG(rental_rate)) AS avg_rent,
COUNT(rental_rate) AS count_rent_values,
MODE() WITHIN GROUP (ORDER BY rating) AS modal_rating_value,
MODE() WITHIN GROUP (ORDER BY special_features) AS modal_special_features_value,
MODE() WITHIN GROUP (ORDER BY fulltext) AS modal_fulltext_value
FROM film
--*/
```

Data Output Messages Notifications

Showing rows: 1 to 1 Page No: 1 of 1

	count_rows bigint	min_release_year integer	max_release_year integer	avg_release_year numeric	count_release_year_values bigint	min_rental_duration smallint	max_rental_duration smallint	avg_rental_duration numeric	count_rental_duration_values bigint	min_len smallin
1	1004	1987	2006	2006	1004	3	7	5	1004	

- Customer table



```
--/*
SELECT
COUNT(*) AS count_rows,
MODE() WITHIN GROUP (ORDER BY store_id) AS modal_store_id_value,
MODE() WITHIN GROUP (ORDER BY first_name) AS modal_first_name_value,
MODE() WITHIN GROUP (ORDER BY last_name) AS modal_last_name_value,
MODE() WITHIN GROUP (ORDER BY email) AS modal_email_value,
MODE() WITHIN GROUP (ORDER BY activebool) AS modal_activebool_value,
MODE() WITHIN GROUP (ORDER BY create_date) AS modal_create_date_value,
MODE() WITHIN GROUP (ORDER BY last_update) AS modal_last_update_value
FROM customer
--*/
```

Data Output Messages Notifications

Showing rows: 1 to 1 Page No: 1 of 1

	count_rows bigint	modal_store_id_value smallint	modal_first_name_value character varying	modal_last_name_value character varying	modal_email_value character varying	modal_activebool_value boolean	modal_create_date_value date	modal_last_update_value timestamp without time zone
1	599	1	Jamie	Abney	aaron.selby@sakilacustomer.org	true	2006-02-14	2013-05-26 14:49:45.738

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.

- **Reflection: Excel vs. SQL for Data Profiling**

From my experience, both Excel and SQL are valuable tools for data profiling, but they serve different purposes depending on the dataset size and the complexity of the task. Excel is more intuitive and user-friendly, especially for smaller datasets or visualizing trends and patterns quickly through features like pivot tables and conditional formatting. However, it becomes cumbersome and slow for large datasets or complex operations.

SQL, on the other hand, is far more efficient for handling large-scale data and complex queries. It allows for precise and automated operations like filtering, grouping, and joining multiple tables with ease. The ability to write repeatable queries in SQL ensures consistency and reduces the chances of manual errors. While SQL requires a steeper learning curve compared to Excel, it excels in speed and scalability, making it the better choice for large datasets or data stored in relational databases.

In conclusion, for quick, small-scale profiling tasks, Excel might be more practical, but for robust, large-scale analysis, SQL is the more effective and reliable tool.