

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).

a. FILM TABLE DUPLICATE CHECK

Query Editor

Query History

```
4      release_year,  
5      language_id,  
6      rental_duration,  
7      rental_rate,  
8      length,  
9      replacement_cost,  
10     rating,  
11     COUNT(*)  
12 FROM film  
13 GROUP BY film_id,  
14          title,  
15          description,  
16          release_year,  
17          language_id,  
18          rental_duration,  
19          rental_rate,  
20          length,  
21          replacement_cost,  
22          rating  
23 HAVING COUNT(*) > 1;
```

Data Output

Messages

Explain

Notifications

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

b. CUSTOMER VALUE DUPLICATE CHECK

Query Editor Query History

```
1  SELECT customer_id,
2         store_id,
3         first_name,
4         last_name,
5         email,
6         address_id
7         COUNT(*)
8  FROM customer
9  GROUP BY customer_id,
10         store_id,
11         first_name,
12         last_name,
13         email,
14         address_id
15  HAVING COUNT(*) > 1;
```

Data Output Messages Explain Notifications

```
ERROR:  syntax error at or near "("
LINE 7:  COUNT(*)
           ^
SQL state: 42601
Character: 89
```

- C.
- d. I do not see any returned duplicate values for either table. For this you can create a virtual table view where certain records can uniquely be selected. You would then delete the spotted duplicate records from the table or the view. You can also use GROUP BY or DISTINCT to uniquely select records
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.
- a. SUMMARY OF COLUMNS IN FILM TABLE

Query Editor

Query History

```

1  SELECT MIN(rental_duration) AS min_rent_period,
2         MAX(rental_duration) AS max_rent_period,
3         AVG(rental_duration) AS avg_rent_period,
4         MIN(rental_rate) AS min_rate_rate,
5         MAX(rental_rate) AS max_rent_rate,
6         AVG(rental_rate) AS avg_rent_rate,
7         COUNT(*) AS count_rows
8  FROM film;
```

Data Output

Messages

Explain

Notifications

	min_rent_period smallint	max_rent_period smallint	avg_rent_period numeric	min_rate_rate numeric	max_rent_rate numeric	av nu
1	3	7	4.9850000000000000	0.99	4.99	


b.

c. NON NUMERIC SUMMARY COLUMN

Query Editor Query History

```
1 SELECT mode() WITHIN GROUP (ORDER BY title)
2     AS modal_title,
3     mode() WITHIN GROUP (ORDER BY description)
4     AS modal_description,
5     mode() WITHIN GROUP (ORDER BY rating)
6     AS modal_rating,
7     COUNT(*) AS count_rows
8 FROM film;
```

Data Output Messages Explain Notifications

	modal_title character varying 	modal_description text
1	Academy Dinosaur	A Action-Packed Character Study of a Astronaut And a Explorer who must Reach

d.

e. SUMMARY FOR NUMERIC COLUMNS IN CUSTOMER TABLE

Query Editor Query History

```
1  SELECT MIN(customer_id) AS min_customer_id,  
2         MAX(customer_id) AS max_customer_id,  
3         AVG(customer_id) AS avg_customer_id,  
4         MIN(store_id) AS min_store_id,  
5         MAX(store_id) AS max_store_id,  
6         AVG(store_id) AS avg_store_id,  
7         MIN(address_id) AS min_address_id  
8         MAX(address_id) AS max_address_id  
9         AVG(address_id) AS avg_address_id  
10        COUNT(*)  
11 FROM customer;
```

f.

g. CUSTOMER TABLE SUMMARY FOR NON NUMERIC COLUMNS

Query Editor

Query History

```

1 SELECT mode() WITHIN GROUP (ORDER BY first_name)
2     AS modal_first_name,
3     mode() WITHIN GROUP (ORDER BY last_name)
4     AS modal_last_name,
5     mode() WITHIN GROUP (ORDER BY email)
6     AS modal_email,
7     COUNT(*) AS count_rows
8 FROM customer;

```

Data Output

Messages

Explain

Notifications

	modal_first_name character varying	modal_last_name character varying	modal_email character varying	count_rows bigint
1	Jamie	Abney	aaron.selby@sakilacustomer.org	599

h.

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.
 - a. For me personally overall Excel for small datasets is a lot more efficient for data profiling especially with the pivot table function which allows you to look at data more efficiently and you can make charts from that pivot table for specific types of data from

that pivot table. Renaming outputs for columns would take a ton more time in excel especially if there are multiple data sets that are huge like with rockbuster we have data for films and customer identification. With the right query that you input into SQL you get results in a split of a second compared to excel which would take a pretty long time to do. SQL is the superior program when it comes to speed with a bigger data set.