ANSWERS 3.6 - Summarizing & Cleaning Data in SQL

1a. Finding Duplicate values

Query Query History

```sql
--Shows duplicate values based on columns selected
SELECT customer_id,store_id,first_name,last_name,
address_id,
Count(*)
FROM customer
GROUP BY customer_id,store_id,first_name,last_name,
address_id
HAVING COUNT(*)>1
```

Data output Messages Notifications

customer_id [PK] integer	store_id smallint	first_name character varying (45)	last_name character varying (45)	address_id smallint	count bigint

No limit

Query Query History

```sql
--Shows duplicate values based on columns selected
SELECT film_id,title,release_year,language_id,
rental_duration,
Count(*)
FROM film
GROUP BY film_id, title,release_year,
language_id,rental_duration
HAVING COUNT(*)>1
```

Data output Messages Notifications

film_id [PK] integer	title character varying (255)	release_year integer	language_id smallint	rental_duration smallint	count bigint

There are no duplicates, but to remove a duplicate, one would either create a view with unique records which is also known as a virtual table or the other is to delete the duplicates. But I will rather create a virtual table by selecting unique records and if I have to delete any duplicates, I will better use a DSP model.

1b. Finding non-uniform values

Query Query History

```
--show non-uniform value in film table base on rating
SELECT DISTINCT rating FROM film
GROUP BY rating
```

Data output Messages Notifications

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

There are not any non-uniform values but to fix this if there are any, I will need to update any values that represent the real values in different formats to be in the exact format as the real value format e.g GEN, g, would have to be updated to be G.

1c. Finding Missing Values

Query Query History Execute/Refresh F5 Scratch Pad

```
--Finding missing values in film
SELECT
COUNT(film_id) AS count_of_film_id,
COUNT(title) AS count_of_title,
COUNT(rental_duration)AS count_of_rental_duration,
COUNT(rental_rate)AS count_of_rental_rate,
COUNT(replacement_cost)AS count_of_replacement_cost,
COUNT(*)AS All_row_count
FROM film
```

Data output Messages Notifications

	count_of_film_id bigint	count_of_title bigint	count_of_rental_duration bigint	count_of_rental_rate bigint	count_of_replacement_cost bigint	all_row_count bigint
1	1000	1000	1000	1000	1000	1000

Query Query History

```sql
--Finding missing values from customer table
SELECT
COUNT(customer_id)AS count_of_customer_id,
COUNT(first_name)AS count_of_first_name,
COUNT(last_name)AS count_of_last_name,
COUNT(*) AS count_of_row
FROM customer
```

Data output Messages Notifications

	count_of_customer_id bigint	count_of_first_name bigint	count_of_last_name bigint	count_of_row bigint
1	599	599	599	599

If there are a lot of missing values in a column, I would better ignore that column to minimize the impact on the analysis. If there are just a few missing values and I know what the values are I can fill it in or just leave it as it is.

2. **<u>Descriptive statistics for numerical value in the film table</u>**

Query Query History Scratch Pad

```sql
--Culculating descriptive statistics for numerical colunms
SELECT
MAX(rental_duration) AS Maximum_rental_duration,
MIN(rental_duration) AS Minimum_rental_duration,
AVG(rental_duration)AS Average_rental_duration,
MAX(rental_rate)AS Maximum_rental_rate,
MIN(rental_rate)AS Minimum_rental_rate,
AVG(rental_rate)AS Average_rental_rate,
MAX(length)As Maximum_length,
MIN(length)AS Minimum_lenghth,
AVG(length)AS Average_length,
MAX(replacement_cost)AS Maximum_replacement_cost,
MIN(replacement_cost)AS Minimum_replacement_cost,
AVG(replacement_cost)AS Average_replacement_cost
FROM film
```

Data output Messages Notifications

	maximum_rei smallint	minimum_ren smallint	average_rental_durat numeric	maximum_rental_rate numeric	minimum_rer numeric	average_renta numeric	maximum_leng smallint	minimum_len smallint	average_lengt numeric	maximum_rep numeric	minimum_rep numeric	average_repla numeric
1	7	3	4.98500000000000	4.99	0.99	2.980000000	185	46	115.2720000	29.99	9.99	19.98400000

2b. **Descriptive statistics for the numerical column in the customer table**

I didn't calculate the Average for these numerical values as we are with "id" so I sticked with Max and Minimum to know the highest and lowest value for the ids.

Query Query History Scratch Pad

```
SELECT
Max(customer_id)AS Maximum_customer_id,
MIN(customer_id)AS Minimum_customer_id,
MAX(store_id)AS Maximum_store_id,
MIN(store_id)AS Minimum_store_id,
MAX(address_id)AS Maximum_address_id,
MIN(address_id) AS Minimum_address_id,
MAX(active)AS Maximum_active,
MIN(active)AS Minimum_active
FROM customer
```

Data output Messages Notifications

	maximum_customer_id integer	minimum_customer_id integer	maximum_store_id smallint	minimum_store_id smallint	maximum_address_id smallint	minimum_address_id smallint	maximum_active integer	minimum_active integer
1	599	1	2	1	605	5	1	0

2c. **Descriptive statistics for non-numerical value in the film table**

Query Query History

```
--Calculating the mode value for non-numerical columns.
SELECT mode()WITHIN GROUP(ORDER BY title)AS modal_title,
mode()WITHIN GROUP(ORDER BY description)AS modal_description,
mode()WITHIN GROUP(ORDER BY rating)AS modal_rating
FROM film
```

Data output Messages Notifications

	modal_title character varying	modal_description text	modal_rating mpaa_rating
1	Academy Dinosaur	A Action-Packed C...	PG-13

2d. Descriptive statistics for non-numerical value in the customer table

Query Query History

```
--Calculating the mode value for non-numerical columns.
SELECT mode()WITHIN GROUP(ORDER BY first_name)AS modal_first_name,
mode()WITHIN GROUP(ORDER BY last_name)AS modal_last_name,
mode()WITHIN GROUP(ORDER BY activebool) AS modal_activebool
FROM customer
```

Data output Messages Notifications

	modal_first_name character varying	modal_last_name character varying	modal_activebool boolean
1	Jamie	Abney	true

3. Using a pivot table for profiling in excel is quite easy when it comes to profiling big data, I think SQL is much easier and faster as much as one knows how to write the syntax and command properly. With Excel, you must go through various stages, like creating the table first, then following some procedures if you want to aggregate the data but with SQL one just needs to know the SQL syntax, write, and click execute and the answer is right there. Overall, using SQL is much easy and faster when it comes to large data and excel is also great when the data isn't large.