

### 3.6 Summarizing & Cleaning Data in SQL

Q1. Check for and clean dirty data:

#### 1.1 Duplicate Data

##### Duplicate Data Check from Film Table

The screenshot shows a SQL IDE interface. The top toolbar includes icons for file operations, query execution, and help. Below the toolbar, there are tabs for 'Query' and 'Query History'. The 'Query' tab is active, displaying the following SQL query:

```
1 SELECT film_id, title, description, release_year, language_id, rental_duration, rental_rate, length,  
2 replacement_cost, rating, last_update, special_features,  
3 COUNT (*)  
4 FROM film  
5 GROUP BY film_id, title, description, release_year, language_id, rental_duration, rental_rate, length,  
6 replacement_cost, rating, last_update, special_features  
7 HAVING COUNT (*) >1;  
8
```

To the right of the query editor is a 'Scratch Pad' tab. Below the query editor, there are tabs for 'Data Output', 'Messages', and 'Notifications'. The 'Data Output' tab is active, showing a table with the following columns and data types:

film_id	title	description	release_year	language_id	rental_duration	rental_rate	length	replacement_cost	rating
[PK] integer	character varying (255)	text	integer	smallint	smallint	numeric (4,2)	smallint	numeric (5,2)	mpaa_rating

##### Duplicate Data Check from Customer table

The screenshot shows a SQL IDE interface. The top toolbar includes icons for file operations, query execution, and help. Below the toolbar, there are tabs for 'Query' and 'Query History'. The 'Query' tab is active, displaying the following SQL query:

```
1 SELECT customer_id, store_id, first_name, last_name, email, address_id, activebool,active,  
2 COUNT(*)  
3 FROM Customer  
4 GROUP BY customer_id, store_id, first_name, last_name, email, address_id, activebool,active  
5 HAVING COUNT(*)>1;  
6
```

To the right of the query editor is a 'Scratch Pad' tab. Below the query editor, there are tabs for 'Data Output', 'Messages', and 'Notifications'. The 'Data Output' tab is active, showing a table with the following columns and data types:

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

After running the Query, the result is not showing any Duplicate Value. There are two ways of removing duplicate value

- Creating a View Table where we can select the Uniques record.
- Deleting the Duplicate records from the Table.

## 1.2 Non Uniform Data

Non Uniform Data check of Film table

Query		Query History	
1	SELECT DISTINCT	rating	
2	FROM	Film;	
3			

  

Data Output		Messages		Notifications	
	rating				
	mpaa_rating				
1	PG-13				
2	NC-17				
3	R				
4	G				
5	PG				

Non Uniform Data check of Customer table

Query		Query History	
1	SELECT DISTINCT	activebool	
2	FROM	Customer;	
3			

  

Data Output		Messages		Notifications	
	activebool				
	boolean				
1	true				

In Non Uniform Data, we can Update the Value. Once we figured out uniformity in data, we can update the value with the help of UPDATE query together with SET and WHERE query.

## 1.3 Missing Data

Another problem is the missing Data in Database. In this case ,either we ignore or don't use the column that having maximum missing value or we can replace the missing value with Impute Value with average or Mode.

Q2: Summarize your data:  
Description Summary of Film Table:

```
SELECT
MIN (film_id) AS min_film_id,
MAX (film_id) AS max_film_id,
AVG (film_id) AS avg_film_id,
MIN (release_year) AS min_release_year,
MAX (release_year) AS max_release_year,
AVG (release_year) AS avg_release_year,
MIN (language_id) AS min_language_id,
MAX (language_id) AS max_language_id,
AVG (language_id) AS avg_language_id,
MIN (rental_duration) AS min_rental_duration,
MAX (rental_duration) AS max_rental_duration,
AVG (rental_duration) AS avg_rental_duration,
MIN (rental_rate) AS min_rental_rate,
MAX (rental_rate) AS max_rental_rate,
AVG (rental_rate) AS avg_rental_rate,
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 title) AS mode_title,
MODE () WITHIN GROUP (ORDER BY description) AS mode_description,
MODE () WITHIN GROUP (ORDER BY rating) AS mode_rating,
MODE () WITHIN GROUP (ORDER BY special_features) AS mode_special_features,
MODE () WITHIN GROUP (ORDER BY fulltext) AS mode_fulltext
FROM film;
```

(2)

informa

eSQL C

ggers

is

ata Wre

as

ons

(1)

regates

ations

nains

: Config

: Diction

: Parser:

: Templa

sign Tab

ctions

erialize

Query

Query History

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

SELECT MIN (film\_id) AS min\_film\_id,

MAX (film\_id) AS max\_film\_id,

AVG (film\_id) AS avg\_film\_id,

MIN (release\_year) AS min\_release\_year,

MAX (release\_year) AS max\_release\_year,

AVG (release\_year) AS avg\_release\_year,

MIN (language\_id) AS min\_language\_id,

MAX (language\_id) AS max\_language\_id,

AVG (language\_id) AS avg\_language\_id,

MIN (rental\_duration) AS min\_rental\_duration,

MAX (rental\_duration) AS max\_rental\_duration,

AVG (rental\_duration) AS avg\_rental\_duration,

MIN (rental\_rate) AS min\_rental\_rate,

MAX (rental\_rate) AS max\_rental\_rate,

AVG (rental\_rate) AS avg\_rental\_rate,

MIN (length) AS min\_length,

MAX (length) AS max\_length,

AVG (length) AS avg\_length,

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

Data Output

Messages

Notifications

min\_film\_id

integer

max\_film\_id

integer

avg\_film\_id

numeric

min\_release\_year

integer

max\_releas

integer

1

1

1000

500.5000000000000000

2006

## Description Summary of 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 (active) AS min_active,  
MAX (active) AS max_active,  
AVG (active) AS avg_active,  
MODE () WITHIN GROUP (ORDER BY first_name) AS mode_first_name,  
MODE () WITHIN GROUP (ORDER BY last_name) AS mode_last_name,  
MODE () WITHIN GROUP (ORDER BY email) AS mode_email,  
MODE () WITHIN GROUP (ORDER BY active) AS mode_active,  
MODE () WITHIN GROUP (ORDER BY activebool) AS mode_activebool  
FROM customer;
```

The screenshot shows a database query editor interface. The top toolbar contains various icons for file operations, query execution, and settings. The main area displays the SQL query from the previous block. Below the query, the 'Data Output' tab is active, showing the results of the query. The results are presented in a table with columns: min\_film\_id, max\_film\_id, avg\_film\_id, min\_release\_year, max\_release\_year, avg\_release\_year, min\_language\_id, and mode\_activebool. The first row of data shows values: 1, 1, 1000, 2006, 2006, 2006.0000000000000000, 1, and 1.

	min_film_id	max_film_id	avg_film_id	min_release_year	max_release_year	avg_release_year	min_language_id	mode_activebool
1	1	1000	500.5000000000000000	2006	2006	2006.0000000000000000	1	1

### Q3 Reflect on the Work

I have worked on both the tool Excel and SQL. I will say, both worked perfectly for data profiling and their effectiveness depend upon the Amount of Data. Like ,For small Data we will consider Excel is best Option and For big data, I recommend SQL. But I also found, that SQL is faster than Excel and give us quick Answers.