

EXCERCE 3-6: Summarizing & Cleaning Data in SQL

1/Check for and clean dirty data for film table

Duplicate value

Query

```
1 SELECT title,  
2 release_year,  
3 language_id,  
4 rental_duration,  
5 COUNT(*)  
6 FROM film  
7 GROUP BY title,  
8 release_year,  
9 language_id,  
10 rental_duration  
11 HAVING COUNT(*) > 1;  
12
```

Data Output Messages Notifications

title	release_year	language_id	rental_duration	count
character varying (255)	integer	smallint	smallint	bigint

There is no duplicate data

>>>I ll use SELECT DISTINCT to avoid counting the same value multiple times or I create the sql view, and I use DELETE

Non-uniform value

Query

```
1 SELECT DISTINCT rating  
2 FROM film;
```

Data Output Messages Notifications

rating
mpaa_rating
G
PG-13
PG
R
NC-17

Showing rows: 1 to 5 Page No: 1

Query

```
1 SELECT DISTINCT rental_duration  
2 FROM film;
```

Data Output Messages Notifications

rental_duration
smallint
4
6
7
3
5

Showing rows: 1 to 5 Page No: 1

Query

```
1 SELECT DISTINCT rental_rate  
2 FROM film;
```

Data Output Messages Notifications

rental_rate
numeric (4,2)
2.99
4.99
0.99

Showing rows: 1 to 3 Page No: 1

Query

```
1 SELECT DISTINCT release_year  
2 FROM film;
```

Data Output Messages Notifications

release_year
integer
2006

Showing rows: 1 to 1 Page No: 1

Query

```
1 SELECT title, COUNT(*)  
2 FROM film  
3 GROUP BY title  
4 HAVING COUNT(*) > 1;
```

Data Output Messages Notifications

title	count
character varying (255)	bigint

There is not non-uniform value

>>>If there is non-uniform value, I will use UPDATE with sql view :UPDATE film

SET rental_rate = ****

WHERE film_id = ***

Missing data

```

Query  Query History
1  SELECT *
2  FROM film
3  WHERE film_id IS NULL
4  OR title IS NULL
5  OR description IS NULL
6  OR release_year IS NULL OR language_id IS NULL
7  OR rental_duration IS NULL
8  OR rental_rate IS NULL
9  OR length IS NULL
10
11 OR replacement_cost IS NULL
12 OR rating IS NULL
13 OR special_features IS NULL
14 OR last_update IS NULL;
15
Data Output  Messages  Notifications
film_id  title  description  release_year  lang
[PK] integer  character varying (255)  text  integer  smal

```

There is not missing value, if there is, I will use update with sql view

2/Summarizing data for film table (MIN; MAX; AVG; MODE)

```

Query  Query History
1
2  SELECT MIN(rental_duration) AS min_duration,
3         MAX(rental_duration) AS max_duration,
4         AVG(rental_duration) AS avg_duration
5  FROM film;
6
Data Output  Messages  Notifications
min_duration  max_duration  avg_duration
smallint  smallint  numeric
1  3  7  4.9850000000000000

```

Column	MIN	MAX	AVG
rental duration	3	7	4.985
rental rate	0.99	4.99	2.98
length	46	185	115.272
replacement cost	9.99	29.99	19.984
film id	1	1000	500.5

```

Query  Query History
1  SELECT COUNT(*)
2  FROM film
Data Output  Messages  Notifications
count
bigint
1  1000

```

```

Query  Query History
1  SELECT MODE()
2  WITHIN GROUP (ORDER BY rating) AS mode_rating
3  FROM film;
Data Output  Messages  Notifications
mode_rating
mpaa_rating
1  PG-13

```

```

Query  Query History
1  SELECT MODE()
2  WITHIN GROUP (ORDER BY title) AS mode_title
3  FROM film;
Data Output  Messages  Notifications
mode_title
character varying
1  Academy Dinosaur

```

Column	Mode
title	Academy Dinosaur
rating	PG-13
special_features	Trailers

Check for and clean dirty data for Customer table

Duplicate value

The screenshot shows the SQL Developer interface with a query window. The query is as follows:

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

Below the query window, the 'Data Output' tab is active, showing the column definitions for the query results:

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

There is no duplicate data

>>>I ll use SELECT DISTINCT to avoid counting the same value multiple times or I create the sql view, and I use DELETE

Missing value

The screenshot shows the SQL Developer interface with a query window. The query is as follows:

```
1 SELECT *  
2 FROM customer  
3 WHERE customer_id IS NULL  
4 OR store_id IS NULL  
5 OR first_name IS NULL  
6 OR last_name IS NULL  
7 OR email IS NULL  
8 OR address_id IS NULL  
9 OR active IS NULL  
10 OR create_date IS NULL  
11 OR last_update IS NULL;  
12  
13  
14  
15
```

Below the query window, the 'Data Output' tab is active, showing the column definitions for the query results:

customer_id	store_id	first_name	last_name
[PK] integer	smallint	character varying (45)	character varyin

There is no missing value, if there is, I will use update with sql view

Non-uniform data

The screenshot shows the SQL Developer interface with a query window. The queries are as follows:

```
1 SELECT DISTINCT first_name  
2 FROM customer;  
3  
4 SELECT DISTINCT last_name  
5 FROM customer;  
6  
7 SELECT DISTINCT email  
8 FROM customer;  
9  
10 SELECT DISTINCT active  
11 FROM customer;  
12  
13 SELECT DISTINCT store_id  
14 FROM customer;  
15
```

Below the query window, the 'Data Output' tab is active, showing the column definitions for the query results:

store_id
smallint
1
2

There is no non-uniform value

>>>If there is non-uniform value, I will use UPDATE with SQL View

Summarizing data for Customer table (MIN; MAX; AVG; MODE)

Query

Query History

1

2

3

4

SELECT MIN(store_id) AS min_store,

MAX(store_id) AS max_store,

AVG(store_id) AS avg_store

FROM customer;

Data Output

























Messages

Notifications

<

Colomn	MIN	MAX	AVG
customer id	1	599	300
store id	1	2	1.45576
adress id	5	605	304.7245

Query	Query History
1	SELECT MODE() WITHIN GROUP (ORDER BY first_name)
2	FROM customer;

Data Output	Messages	Notifications
                       		
	mode character varying	
1	Jamie	

Column	Mode
first_name	Jamie
last_name	Abney
email	aaron.selby@sakila.customer.org

Reflect on your work

I feel that with SQL, the work is simpler and faster; you can insert three functions at the same time for example. You just need to master the SQL commands and syntax, which is easier with Excel, where with a few clicks you get what you want.