3.6 Summarizing & Cleaning Data in SQL

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. **non-uniform**

```sql
-- film
SELECT rating
FROM film
GROUP BY rating;


-- customer
SELECT activebool
FROM customer
GROUP BY activebool;

SELECT active
FROM customer
GROUP BY active;

SELECT email
FROM customer
WHERE email NOT LIKE '%@%';
```

When there’s non-uniform data, we can as far as it’s possible try to correct the data manually to make them uniform.

   b. **duplicate data**

```sql
-- film
SELECT
    title,
    release_year,
    language_id,
    rental_duration,
    COUNT(*)
FROM film
GROUP BY
    title,
    release_year,
    language_id,
    rental_duration
HAVING COUNT(*) >1;
```

```
-- customer
SELECT
    customer_id,
    COUNT(*)
FROM customer
GROUP BY
    customer_id
HAVING COUNT(*) >1;
```

We can delete the duplicate values or we can hide them with the view option or even use the DISTINCT statement in the query.

c. missing values

```
-- film
SELECT
    COUNT(title) AS "COUNT of title",
    COUNT(rental_duration) AS "COUNT of rental_duration",
    COUNT(rental_rate) AS "COUNT of rental_rate",
    COUNT(length) AS "COUNT of length",
    COUNT(replacement_cost) AS "COUNT of replacement_cost",
    COUNT(*) AS "COUNT (rows)"
FROM film;


-- customer
SELECT
    COUNT(first_name) AS "COUNT of first_name",
    COUNT(last_name) AS "COUNT of last_name",
    COUNT(email) AS "COUNT of email",
    COUNT(address_id) AS "COUNT of address_id",
    COUNT(activebool) AS "COUNT of activebool",
    COUNT(create_date) AS "COUNT of create_date",
    COUNT(last_update) AS "COUNT of last_update",
    COUNT(active) AS "COUNT of active",
    COUNT(*) AS "COUNT (rows)"
FROM customer;
```

When data is missing, we could try to fill them with the mode or the average value, for example.

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.

```sql
-- customer
-- numeric
SELECT
    MIN(customer_id) AS "MIN of customer_id",
    MAX(customer_id) AS "MAX of customer_id",
    AVG(customer_id) AS "AVG of customer_id",
    MIN(store_id) AS "MIN of store_id",
    MAX(store_id) AS "MAX of store_id",
    AVG(store_id) AS "AVG of store_id",
    MIN(active) AS "MIN of active",
    MAX(active) AS "MAX of active",
    AVG(active) AS "AVG of active"
FROM customer;
```

MIN of customer_id	MAX of customer_id	AVG of customer_id
1	599	300

MIN of store_id	MAX of store_id	AVG of store_id
1	2	1.455759599

MIN of active	MAX of active	AVG of active
0	1	0.974958264

```sql
-- non-numeric
SELECT
    MODE() WITHIN GROUP (ORDER BY first_name)
       AS "MODE of first_name",
    MODE() WITHIN GROUP (ORDER BY last_name)
       AS "MODE of last_name",
    MODE() WITHIN GROUP (ORDER BY email)
       AS "MODE of email",
    MODE() WITHIN GROUP (ORDER BY activebool)
       AS "MODE of activebool"
FROM customer;
```

MODE of first_name
Jamie

MODE of last_name
Abney

MODE of email
aaron.selby@sakilacustomer.org

MODE of activebool
TRUE

```sql
-- film
-- numeric
SELECT
    MIN(film_id) AS "MIN of film_id",
    MAX(film_id) AS "MAX of film_id",
    AVG(film_id) AS "AVG of film_id",
    MIN(release_year) AS "MIN of release_year",
    MAX(release_year) AS "MAX of release_year",
    AVG(release_year) AS "AVG of release_year",
    MIN(language_id) AS "MIN of language_id",
    MAX(language_id) AS "MAX of language_id",
    AVG(language_id) AS "AVG of language_id",
    MIN(rental_duration) AS "MIN of rental_duration",
    MAX(rental_duration) AS "MAX of rental_duration",
    AVG(rental_duration) AS "AVG of rental_duration",
    MIN(rental_rate) AS "MIN of rental_rate",
    MAX(rental_rate) AS "MAX of rental_rate",
    AVG(rental_rate) AS "AVG of rental_rate",
    MIN(length) AS "MIN of length",
    MAX(length) AS "MAX of length",
    AVG(length) AS "AVG of length"
FROM film;
```

MIN of film_id	MAX of film_id	AVG of film_id
1	1000	500.5

MIN of release_year	MAX of release_year	AVG of release_year
2006	2006	2006

MIN of language_id	MAX of language_id	AVG of language_id
1	1	1

MIN of rental_duration	MAX of rental_duration	AVG of rental_duration
3	7	4.985

MIN of rental_rate	MAX of rental_rate	AVG of rental_rate
0.99	4.99	2.98

MIN of length	MAX of length	AVG of length
46	185	115.272

```sql
-- non-numeric
SELECT
    MODE() WITHIN GROUP (ORDER BY title)
       AS "MODE of title",
    MODE() WITHIN GROUP (ORDER BY description)
       AS "MODE of description",
    MODE() WITHIN GROUP (ORDER BY rating)
       AS "MODE of rating",
    MODE() WITHIN GROUP (ORDER BY special_features)
       AS "MODE of special_features",
    MODE() WITHIN GROUP (ORDER BY fulltext)
       AS "MODE of fulltext"
FROM film;
```

MODE of title
Academy Dinosaur

MODE of description
A Action-Packed Character Study of a Astronaut And a Explorer who must Reach a Monkey in A MySQL Conv

MODE of rating
PG-13

MODE of special_features
{Trailers,Commentaries,"Behind the Scenes"}

MODE of fulltext
'baloon':19 'confront':14 'documentari':5 'feminist':8,11,16 'mile':2 'must':13 'spi':1 'thrill':4

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.

   Excel feels more intuitive as you can see the data directly, but SQL feels more directly and trustworthy – plus you have a better overview of what you're doing. I prefer SQL – especially when there's more data.