Exercise 3.6

May 30, 2022

STEP 1 – CHECK FOR AND CLEAN DIRTY DATA

DUPLICATE – Had we found any duplicates, we would be wise to remove them, likely by turning to a virtual table, known as a VIEW.

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,
        store_id,
        first_name,
        email,
        COUNT(*)
FROM customer
GROUP BY customer_id,
        store_id,
        first_name,
        email
HAVING COUNT(*) >1;
```

NON-UNIFORM – This will allow us to see what has been entered. For the examples below, if we know that there are only two store_ids, then if there was something entered that wasn't 1 or 2, we'd know it was a mistake. Or a rental_rate that was entered in writing instead of numerical.

Film

```
SELECT DISTINCT rental_rate
FROM film
GROUP BY rental_rate
```

Customer

```
SELECT DISTINCT store_id
FROM customer
GROUP BY store_id
```

MISSING VALUES – Once we find whether there are missing values, as can be seen below, we can then either impute values (if there are only a few values missing) or omit a column (if there are lots of values missing).

Film

```
SELECT
COUNT(title) AS count_title,
COUNT(rental_duration) AS count_rental_duration,
COUNT(rental_rate) AS count_rental_rate,
COUNT(*) AS count_rows
FROM film;
```

Customer

```
SELECT
COUNT(customer_id) AS count_customer_id,
COUNT(first_name) AS count_first_name,
COUNT(email) AS count_email,
COUNT(*) AS count_rows
FROM customer
```

STEP 2 - SUMMARIZE YOUR DATA

Film

```
SELECT 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_movie_length_minutes,
    MAX(length) AS max_movie_length_minutes,
    AVG(length) AS avg_movie_length_minutes,
    MIN(replacement_cost) AS min_replacement_cost,
    MAX(replacement_cost) AS max_replacement_cost,
    AVG(replacement_cost) AS avg_replacement_cost
FROM film;
```

	min_rental_duration smallint	max_rental_duration smallint	avg_rental_duration numeric	min_rental_rate numeric	max_rental_rate numeric	avg_rental_rate numeric
1	3	7	4.985	0.99	4.99	2.98

min_movie_length_minutes smallint	max_movie_length_minutes smallint	avg_movie_length_minutes numeric	min_replacement_cost numeric	max_replacement_cost numeric	avg_replacement_cost numeric
46	185	115.272	9.99	29.99	19.98

```
SELECT mode() WITHIN GROUP (ORDER BY title) AS title_value,
mode() WITHIN GROUP (ORDER BY release_year) AS release_year_value,
mode() WITHIN GROUP (ORDER BY rating) AS rating_value,
mode() WITHIN GROUP (ORDER BY special_features) AS special_features_value
FROM film;
```

	title_value character varying	release_year_value integer	rating_value mpaa_rating	special_features_value text[]
1	Academy Dinosaur	2006	PG-13	{Trailers,Commentaries,"Behind the Scenes"}

Customer

```
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
FROM customer;
```

min_customer_id integer	max_customer_id integer	avg_customer_id numeric	min_store_id smallint	max_store_id smallint	avg_store_id numeric	min_address_id smallint	max_address_id smallint	avg_ad numeri
1	599	300	1	2	1.4557595993322203	5	605	304.

```
SELECT mode() WITHIN GROUP (ORDER BY first_name) AS first_name_value,
mode() WITHIN GROUP (ORDER BY last_name) AS last_name_value,
mode() WITHIN GROUP (ORDER BY email) AS email_value
FROM customer;
```

	first_name_value character varying	last_name_value character varying	email_value character varying
1	Jamie	Abney	aaron.selby@sakilacustomer.org

STEP 3 – REFLECT ON YOUR WORK

Based on my experience, I still find Excel easier, but only because I have used it all my life. I think once I know what the formulae are, and become more familiar with how to write it, SQL will be easier. Also, the ability to summarize large quantities of data, makes SQL very attractive, I see a lot of value to this program. Having multiple tables and being able to summarize them in one spot will be very handy.