3.3: SQL for Data Analysts

Directions

Create a new text document and call it “Answers 3.3.” You'll submit your queries, outputs, and written answers in this document at the end of the task.

Step 1:

Your first task is to find out what film genres already exist in the category table:

- Open pgAdmin 4, click the Rockbuster database, and open the Query Tool.
- Write a SELECT command to find out what film genres exist in the category table.
    - SELECT *
    - FROM category;
- Copy-paste the output into your answers document or write the answers out—it’s up to you. Make sure to include the category ID for each genre.

1 "Action"
2 "Animation"
3 "Children"
4 "Classics"
5 "Comedy"
6 "Documentary"
7 "Drama"
8 "Family"
9 "Foreign"
10 "Games"
11 "Horror"
12 "Music"
13 "New"
14 "Sci-Fi"
15 "Sports"

16 "Travel"

Step 2:

You're ready to add some new genres! Write an INSERT statement to add the following genres to the category table: Thriller, Crime, Mystery, Romance, and War:

- Copy-paste your INSERT commands into your answers document.

```
Insert INTO category(category_id,name)
VALUES
(17, 'Thriller'),
(18, 'Crime'),
(19, 'Mystery'),
(20, 'Romance'),
(21, 'War')
```

Utilizing the following, the output showed the values had been added to the list above which ended with 16 "travel"

```
SELECT *
FROM category;
```

- The CREATE statement below shows the constraints on the category table. Write a short paragraph explaining the various constraints that have been applied to the columns. What do these constraints do exactly? Why are they important?

```
CREATE TABLE category
(
 category_id integer NOT NULL DEFAULT nextval('category_category_id_seq'::regclass),
 name text COLLATE pg_catalog."default" NOT NULL,
 last_update timestamp with time zone NOT NULL DEFAULT now(),
 CONSTRAINT category_pkey PRIMARY KEY (category_id)
);
```

This command statement does the following: creates a table, removes null values in the integer list, nextval(regclass) advances sequence and return new value (effectively just generating a new id for values after ones that were null values); collates the the pg_catalog."default" with non-null values;

creates a constraint using PRIMARY KEY which gives each record in a table (in category_id) a unique ID. In summary, this will remove null values from a table and assign the previous IDs associated with the null values as the next one in the sequence which effectively removes the old value from the table. This also generates unique IDs for each subject in a row.

Step 3:

The genre for the movie *African Egg* needs to be updated to thriller. Work through the steps below to make this change:

- Write the SELECT statement to find the film_id for the movie *African Egg*.

```
SELECT film_id

FROM film

WHERE title = 'African Egg'
```

Output:

5	African Eg	A Fast-Pac	2006	1	6	2.99	130	22.99	G	50:59.0

- Once you have the film_ID and category_ID, write an UPDATE command to change the category in the film_category table (not the category table). Copy-paste this command into your answers document.

```
UPDATE film_category

SET category_id = 6

WHERE film_id = 5
```

To check:

```
SELECT *

FROM film_category

WHERE film_id = 5
```

Step 4:

Since there aren't many movies in the mystery category, you and your manager decide to remove it from the category table. Write a DELETE command to do so and copy-paste it into your answers document.

```
DELETE

FROM category

WHERE category_id = 19
```

(I used 19 as it referred to the 19th category, mystery. Though there's likely a simpler way, I then deleted category_id 20 and 21, then used the insert function from step 2 and manually changed the categories for romance and war to 19 and 20. See below:

```
DELETE

FROM category

WHERE category_id = 20


DELETE

FROM category

WHERE category_id = 21


Insert INTO category(category_id,name)

VALUES

(19, 'Romance'),

(20, 'War')
```

Step 5:

Based on what you've learned so far, think about what it would be like to complete steps 1 to 4 with Excel instead of SQL. Are there any pros and cons to using SQL? Write a paragraph explaining your answer.

a. One of the neat things about SQL is the ability to generate categories you can later assign to things. Then those categories are just sitting and waiting to be added in. In our example, we added distinct genres to a list so we could edit genres given movies later. In Excel, we would have to add a column for genre and manually input the genre ourselves. We could utilize the IF CONTAINS procedure in Excel to help narrow our search by using the description box as an identifier. There are likely syntax that let us apply the genre to multiple titles utilizing phrases found in the descriptors. In so far, however, the greatest disadvantage is the need to UPDATE each title manually; again, new syntax and tricks pending. To me, though, just adding categories I can assign later seems like a really useful tool.
b. As for the DELETE function in step 4, that is likely objectively easier in Excel as you can just filter Mystery in the genre column and then delete the genre for those titles, leaving them blank OR leaving them with their other descriptors.

Step 6:

Save your "Answers 3.3" document as a PDF and upload it here for your tutor to review.