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.**
- **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.**

SELECT category_id, name AS genre FROM Category

category_id	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.**

I had to do this one by one – don't know how to add more than one new genre at once

INSERT INTO category (name) VALUES ('war')

name
Action
Animation
Children
Classics
Comedy
Documentary
Drama
Family
Foreign
Games
Horror
Music
New
Sci-Fi
Sports
Travel
thriller
crime
mystery
romance
war

- **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)
);
```

NOT NULL means there can't be any missing data or values – so we will always have a complete record

DEFAULT means a default value will be added for all new records

PRIMARY Key means this columns is et as the primary key – it's the unique identifier for this table

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 title, film_id FROM film WHERE title = 'African Egg'

film_id= 5

- **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.**

SELECT category_id, film_id FROM film_category WHERE film_id = 5

Category_id = 8

SELECT category_id, name FROM category WHERE category_id = 8

Current category is 'family'

SELECT name, category_id FROM category WHERE name = 'thriller'

Thriller category_id = 17

UPDATE command to change the category in the film_category table (not the category table

UPDATE film_category SET category_id = 17 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 name = 'mystery'

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.

This would all be easily done in Excel, and in fact for these small changes, it might be a bit simpler.

If there were bigger changes, SQL would show its value. The advantage of doing so with SQL is that – with these tables connected, if I change the name of a category, say, that's then done throughout – I don't need to find and change that name everywhere it exists. And I also have a record of the queries I've made – which I can undo if needed.

Step 6:

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

Bonus Task

The SQL query below contains some typos. See if you can fix it based on what you've learned so far about SQL and data types; then try running it in pgAdmin 4. If the query works, copy it into your Answers 3.3 document.

If you get this you're a SQL champ!

CREATE TBL 3EMPLOYEES

{

employee_id VARINT(30) NOT EMPTY

name VARCHAR(50),

contact_number VARCHAR(30),

designation_id INT,

last_update TIMESTAMP NOT NULL DEF now()

CONSTRAIN employee_pkey PRIMARY KEY (employee_id)

```
CREATE TABLE employees (
   employee_id INT NOT NULL,
   name VARCHAR(50),
   contact_number VARCHAR(30),
   designation_id INT,
   last_update TIMESTAMP NOT NULL DEFAULT now(),
   CONSTRAINT employee_pkey PRIMARY KEY (employee_id)
);
```