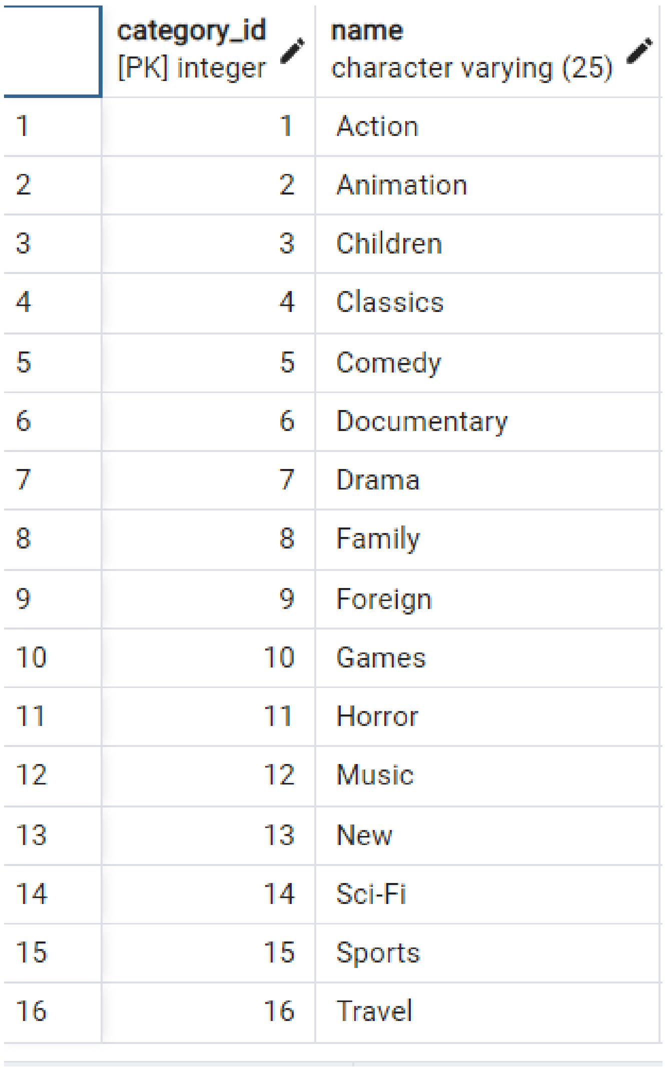
3.3: SQL for Data Analysts

Step 1.

- *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.*

Here several variants are possible:
1). SELECT *, name
    FROM category
2). SELECT name
    FROM category
3). I've done this one:
SELECT *
FROM category



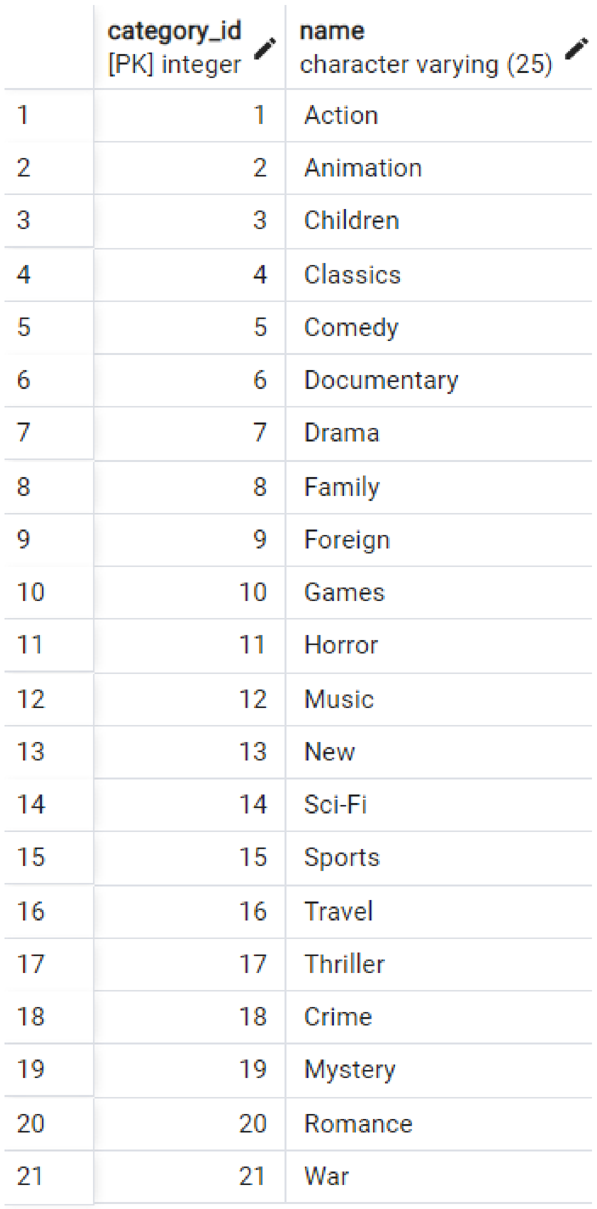
	category_id [PK] integer	name character varying (25)
1	1	Action
2	2	Animation
3	3	Children
4	4	Classics
5	5	Comedy
6	6	Documentary
7	7	Drama
8	8	Family
9	9	Foreign
10	10	Games
11	11	Horror
12	12	Music
13	13	New
14	14	Sci-Fi
15	15	Sports
16	16	Travel

Step 2.

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 (name)
VALUES ('Thriller'),
('Crime'),
('Mystery'),
('Romance'),
('War')
```



	category_id [PK] integer	name character varying (25)
1	1	Action
2	2	Animation
3	3	Children
4	4	Classics
5	5	Comedy
6	6	Documentary
7	7	Drama
8	8	Family
9	9	Foreign
10	10	Games
11	11	Horror
12	12	Music
13	13	New
14	14	Sci-Fi
15	15	Sports
16	16	Travel
17	17	Thriller
18	18	Crime
19	19	Mystery
20	20	Romance
21	21	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?*

  NOT NULL DEFAULT – this constraint ensures:

  - there are no numbers missing, value must be integer in a sequential order in column ‘category_id’;

  - value for column ‘name’ must be text, no numbers missing;

  - value for column ‘last_update’ must have no numbers missing, timestamp with current time zone;

  PRIMARY KEY - this constraint makes all values in the column ‘category_id’ into a primary key category_pkey.

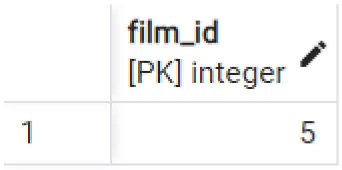
```
  CREATE TABLE category
(
  category_id integer NOT NULL DEFAULT nextval('category_category_id_seq'::regclass),
  name text COLLATE pg_catalog."default" NOT NULL,
  last_update timestamp with time zone NOT NULL DEFAULT now(),
  CONSTRAINT category_pkey PRIMARY KEY (category_id)
);
```

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.*

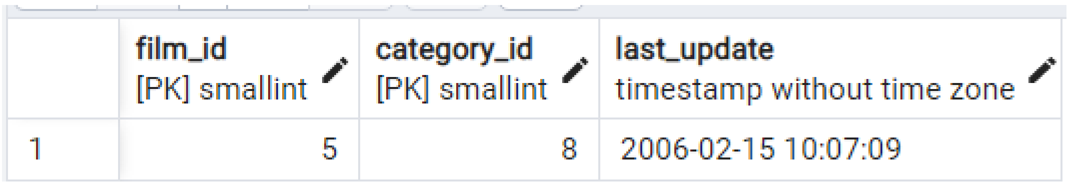
SELECT film_id
FROM film
WHERE title ='African Egg'



	film_id [PK] integer
1	5

SELECT *
FROM film_category

WHERE film_id=5

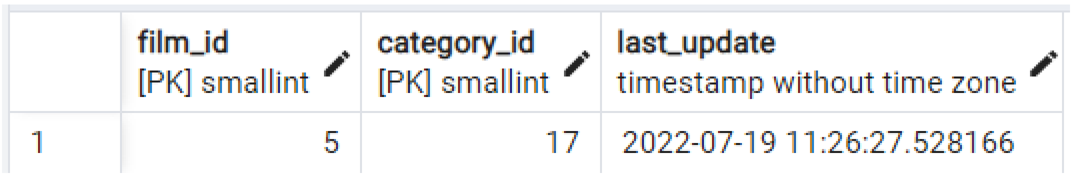


	film_id [PK] smallint	category_id [PK] smallint	last_update timestamp without time zone
1	5	8	2006-02-15 10:07:09

- *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=17
WHERE film_id=5

*The result for this command was not shown immediately ("Query returned successfully in 230 msec."), I've used SELECT command to see this result:



	film_id [PK] smallint	category_id [PK] smallint	last_update timestamp without time zone
1	5	17	2022-07-19 11:26:27.528166

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.

Step 3, where we Selected and Updated the information can be done easier in Excel, to my mind. We worked only with one value – African Egg – in the film_category table. In Excel it'd be easy to find this one genre and update the information about it, rather than writing the whole query, especially with several steps. I think, it's easier to find the rows or to insert new values with SQL, than with Excel where one must manually manipulate the rows, then manually update them.

Using SQL means using logic and knowledge of ERD. It’s beneficial for manipulation of large amount of data.

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.

The incorrect variant (typos are marked in yellow):

```
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)
}
```

Correct variant:

CREATE TABLE employees

(

employee_id VARCHAR(30) NOT NULL,

name VARCHAR(50),

contact_number VARCHAR(30),

designation_id INT,

last_update TIMESTAMP NOT NULL DEFAULT now(),

CONSTRAINT employees_pkey PRIMARY KEY (employee_id))