Title: DB Assignment 4

Name: Jerome Grant

Date: November 9, 2025

SQL Section

1. What is the average length of films in each category? List the results in alphabetic order of categories.

```
109  -- Query 1 : Finding average length of films in each category and listing result in alphabetical order of categories
110  select category.name, round(avg(length),2) as Average_Length
111  from category inner join film_category using(category_id) inner join film using (film_id)
112  group by category.name
113  order by category.name;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

name	Average_Length
Action	111.61
Animation	111.02
Children	109.80
Classics	111.67
Comedy	115.83
Documentary	108.75
Drama	120.84
Family	114.78
Foreign	121.70
Games	127.84
Horror	112.48
Music	113.65
New	111.13
Sci-Fi	108.20
Sports	128.20
Travel	113.32

This query aims to find the average length of films in each category by selecting the category name and using the avg function to find the average length of films, rounded to two decimal places for my Average_Length column for my output. Then I joined three tables (category, film_category and film) using inner join, then grouping by the category name, and ordering the result in ascending order.

2. Which categories have the longest and shortest average film lengths?

```
116     -- Query 2: Finding which categories have the longest and shortest film lengths
117     -- Finding category with longest average film
118 •   (Select category.name, round(avg(length),2) as Average_Length
119     from category inner join film_category using(category_id) inner join film using(film_id)
120     group by category.name
121     having avg(length) >=all(
122         select avg(length)
123         from category inner join film_category using(category_id) inner join film using(film_id)
124         group by category.name)
125     )
126
127     union
128
129     -- Finding category with shortest average film
130     (Select category.name, round(avg(length),2) as Average_Length
131     from category inner join film_category using(category_id) inner join film using(film_id)
132     group by category.name
133     order by Average_Length
134     limit 1);
135
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

name	Average_Length
Sports	128.20
Sci-Fi	108.20

This query aims to find which categories have the longest and shortest film lengths by using a union set operation to join two queries into a single result set. It has the category name and average length as output. The first query looks to find the category with the maximum average film length by using the group by clause and having average length greater than all that is in the subquery to produce the first row in the result table. The second query looks to find the category with the minimum average film length, by ordering by Average_Length column in ascending order and returning only the first row of data that would contain the shortest average length film. The resulting table shows the category name as well as the longest and shortest average film length.

3. Which customers have rented action but not comedy or classic movies?

```
137    -- Query 3: Finding which customers rented action but not comedy or classic movies
138 •  Select distinct customer.customer_id, customer.first_name, customer.last_name
139    from category inner join film_category using(category_id) inner join film using(film_id) inner join inventory using(film_id)
140    inner join rental using (inventory_id) inner join customer using(customer_id)
141    where category.name = 'Action' and customer.active=1 and customer.customer_id not in(
142    Select distinct customer.customer_id
143    from category inner join film_category using(category_id) inner join film using(film_id) inner join inventory using(film_id)
144    inner join rental using (inventory_id) inner join customer using(customer_id)
145    where category.name in('Comedy','Classics'));
146
147
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

customer_id	first_name	last_name
17	DONNA	THOMPSON
90	RUBY	WASHINGTON
139	AMBER	DIXON
164	JOANN	GARDNER
171	DOLORES	WAGNER
213	GINA	WILLIAMSON
223	MELINDA	FERNANDEZ
232	CONSTANCE	REID
250	JO	FOWLER
323	MATTHEW	MAHAN
330	SCOTT	SHELLEY
350	JUAN	FRALEY
361	LAWRENCE	LAWTON
432	EDWIN	BURK
433	DON	BONE
445	MICHEAL	FORMAN
452	TOM	MILNER

This query finds which customers rented action films but not comedy or classic films. It does this by joining 5 tables(category, film_category, film, inventory, rental and customer) where when joined, the category name associated to a film when joined would have to be action, and the customer would have to be considered active. Furthermore, the customer's id should not be in a subquery where the category name is in the set of comedy and classics. The result was a table with 17 rows of data, showing the customer id, as well as their first and last name. The select distinct was used to remove duplicate results of a customer id showing enabling just one result for a customer's name that rented action movies but not comedy or classic ones.

4. Which actor has appeared in the most English-language movies?

```
148    -- Query 4: Finding which actor has appeared in the most English-language movies
149    Select distinct actor.actor_id, actor.first_name, actor.last_name, count(film_id) as Films
150    from actor inner join film_actor using(actor_id) inner join film using(film_id) inner join language using(language_id)
151    where language.name='English'
152    group by actor.actor_id
153    having count(film_id) >=all (
154        select count(film_id)
155        from actor inner join film_actor using(actor_id) inner join film using(film_id) inner join language using(language_id)
156        where language.name='English'
157        group by actor.actor_id
158        );
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

actor_id	first_name	last_name	Films
107	GINA	DEGENERES	42

This query aims to find which actor appeared in the most English-language movies. Select distinct was used to remove duplicates of an actor's id appearing, and the actor's first name, last name, and count of the number of films were selected to be shown in the resulting table. Four tables were joined (actor, film_actor, film, and language). The where clause specified that the language name had to be English, and it was grouped by the actor_id. A subquery was then used to find and return the actor_id with the highest film_id count.

5. How many distinct movies were rented for exactly 10 days from the store where Mike works?

```
160    -- Query 5: How many distinct movies were rented for exactly 10 days from the store where Mike works
161    Select count(distinct film_id) as Number_of_Movies
162    from film inner join inventory using(film_id) inner join store using(store_id) inner join staff using(store_id) inner join rental using(staff_id)
163    where staff.first_name ='Mike' and datediff(rental.return_date,rental_date)=10;
164
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

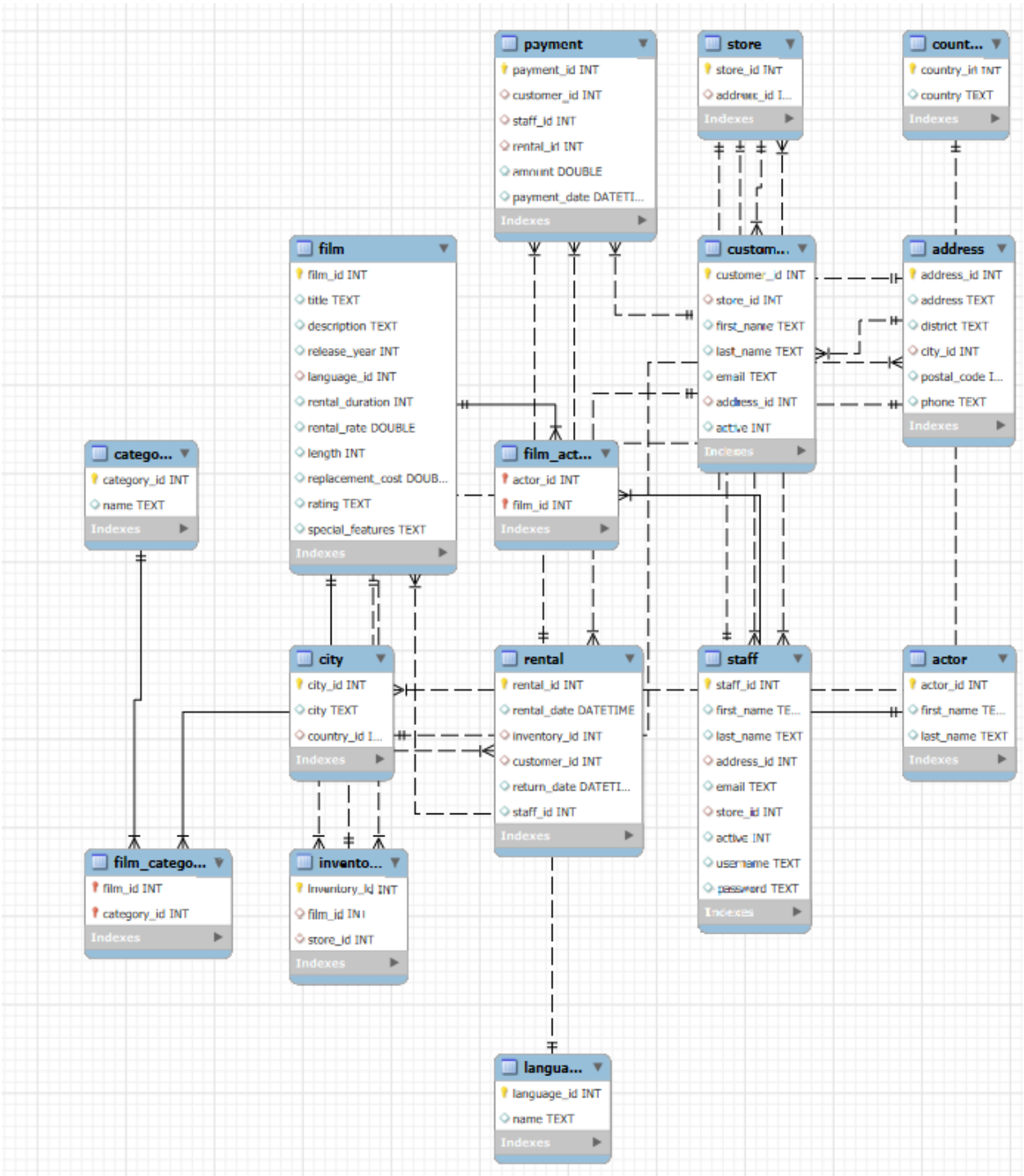
Number_of_Movies
759

This query aims to find the amount of movies that were rented for exactly 10 days from the store a staff named Mike works. This query does so by using the count function to count the distinct film ids. Five tables were joined (film, inventory, store, staff and rental) where the staff's first name has to be Mike and the datediff function, which calculates the difference between the return date and the rental date, returns a value equal to 10.

6. Alphabetically list actors who appeared in the movie with the largest cast of actors.

--N/A due to errors

ERD Diagram


payment
payment_id INT
customer_id INT
staff_id INT
rental_id INT
amount DOUBLE
payment_date DATETI...
Indexes
store
store_id INT
address_id I...
Indexes
count...
country_id INT
country TEXT
Indexes
film
film_id INT
title TEXT
description TEXT
release_year INT
language_id INT
rental_duration INT
rental_rate DOUBLE
length INT
replacement_cost DOUB...
rating TEXT
special_features TEXT
Indexes
custom...
customer_id INT
store_id INT
first_name TEXT
last_name TEXT
email TEXT
address_id INT
active INT
Indexes
address
address_id INT
address TEXT
district TEXT
city_id INT
postal_code I...
phone TEXT
Indexes
catego...
category_id INT
name TEXT
Indexes
film_act...
actor_id INT
film_id INT
Indexes
city
city_id INT
city TEXT
country_id I...
Indexes
rental
rental_id INT
rental_date DATETIME
inventory_id INT
customer_id INT
return_date DATETI...
staff_id INT
Indexes
staff
staff_id INT
first_name TE...
last_name TEXT
address_id INT
email TEXT
store_id INT
active INT
username TEXT
password TEXT
Indexes
actor
actor_id INT
first_name TE...
last_name TEXT
Indexes
film_catego...
film_id INT
category_id INT
Indexes
invento...
inventory_id INT
film_id INT
store_id INT
Indexes
langua...
language_id INT
name TEXT
Indexes