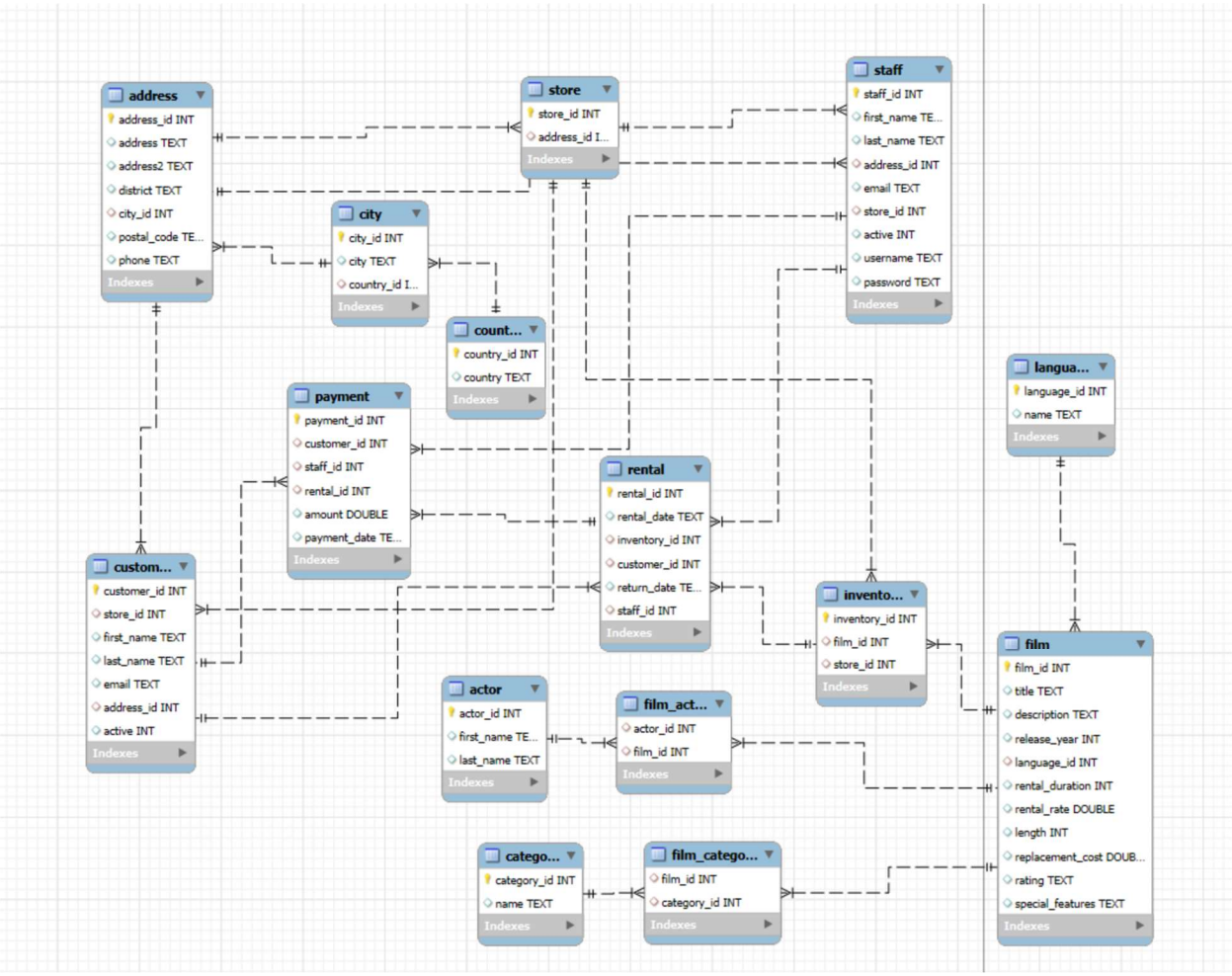
Title: DB Assignment 4

Name: Paul Anderson

Date: 02 November 2025

ERD Diagram:

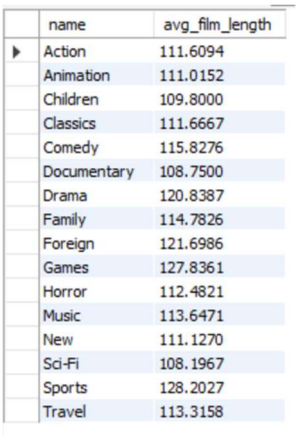


Query 1: What is the average length of films in each category? List the results in alphabetic order of categories

-- getting the category name and getting all the film length averages and grouping them by category

```
select category.name, avg(film.length) as avg_film_length
from category
join film_category on category.category_id = film_category.category_id
join film on film_category.film_id = film.film_id
group by category.name
order by category.name;
```

Output:



name	avg_film_length
Action	111.6094
Animation	111.0152
Children	109.8000
Classics	111.6667
Comedy	115.8276
Documentary	108.7500
Drama	120.8387
Family	114.7826
Foreign	121.6986
Games	127.8361
Horror	112.4821
Music	113.6471
New	111.1270
Sci-Fi	108.1967
Sports	128.2027
Travel	113.3158

explanation:

First, the query gets the category names, later to be grouped by, and the average length of films within each category. Next the category are order alphabetically.

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

```
-- initial search looking for the longest average film length
select category.name, avg(film.length) as avg_film_length
from category
join film_category on category.category_id = film_category.category_id
join film on film_category.film_id = film.film_id
group by category.name
```

```
having avg_film_length >= all ( -- seeing which film category has the greatest average
        select avg(film.length)
   from category
        join film_category on category.category_id = film_category.category_id
        join film on film_category.film_id = film.film_id
        group by category.name)

union

-- search looking for the shortest average film length
select category.name, avg(film.length) as avg_film_length
from category
join film_category on category.category_id = film_category.category_id
join film on film_category.film_id = film.film_id
group by category.name
having avg_film_length <= all ( -- seeing which film category has the greatest average
        select avg(film.length)
   from category
        join film_category on category.category_id = film_category.category_id
        join film on film_category.film_id = film.film_id
        group by category.name);
```

Output: ***NOTE: OTHER COMPUTERS OUTPUT THE CORRECT RESULT, MY OWN LAPTOP IS BLOCKING THE SECOND RESULT FROM BEING SHOW. PLEASE CONTACT THE TA FOR PROOF OF ISSUE.***

	name	avg_film_length
▶	Sports	128.2027

Explanation:

Two different queries. The first query gets category name and the average length of films in each category. After than it searches for the highest average via the having subquery. The second query does the exact same thing, but searches for the lowest average film length using the having command.

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

```
-- getting customers who rented out action movies

select customer.customer_id, customer.first_name, customer.last_name

from customer

join rental on customer.customer_id = rental.customer_id

join inventory on rental.rental_id = inventory.inventory_id

join film on inventory.film_id = film.film_id

join category on film.film_id = category.category_id

where category.name = 'Action'


-- remove the next query

except


-- getting customers who rented out comedy or classics

select customer.customer_id, customer.first_name, customer.last_name
```

```
from customer
join rental on customer.customer_id = rental.customer_id
join inventory on rental.rental_id = inventory.inventory_id
join film on inventory.film_id = film.film_id
join category on film.film_id = category.category_id
where category.name = 'Comedy' or 'Classics';
```

Output:

customer_id	first_name	last_name
130	CHARLOTTE	HUNTER
459	TOMMY	COLLAZO
408	MANUEL	MURRELL
333	ANDREW	PURDY
222	DELORES	HANSEN
549	NELSON	CHRISTENSON
269	CASSANDRA	WALTERS
239	MINNIE	ROMERO

Explanation:

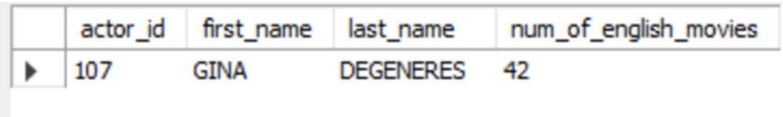
The first query gets all customers ids, first names, and last names, who rented out films under the category of action, this was done using the ‘where’ key word. Using the ‘except’ key word, the query then subtracts from the list those who had rented out either comedies or classics via the ‘where’ command.

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

```
-- gets a list of all the films that are english
with english_movies as (
select film.film_id, film.title
from film
```

```
join language on film.language_id = language.language_id

where language.name = 'English'

)


-- uses the previous cte to get the actors who have appeared in all english movies

select actor.actor_id, actor.first_name, actor.last_name, count(english_movies.title) as
num_of_english_movies

from english_movies

join film_actor on english_movies.film_id = film_actor.film_id

join actor on film_actor.actor_id = actor.actor_id

group by actor.actor_id, actor.first_name, actor.last_name

having num_of_english_movies >= all ( -- sorts for the actor who has been in the most
english movies

        select count(english_movies.title)

   from english_movies

        join film_actor on english_movies.film_id = film_actor.film_id

        join actor on film_actor.actor_id = actor.actor_id

        group by actor.actor_id, actor.first_name, actor.last_name

);
```

Output:



	actor_id	first_name	last_name	num_of_english_movies
▶	107	GINA	DEGENERES	42

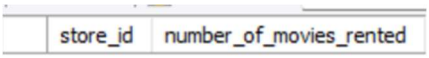
Explanation: The query starts by using a CTE called English_movies, which searches and returns a list of all movies that are English. This CTE is then used in the main search which

gets a count of all English movies an actor has been in, returning the actor with the most participation in different English movies via the ‘having’ key word.

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

```
-- CTE to get the store where mike worked at
with store_where_mike_works as (
        select store.store_id
  from store
  join staff on store.store_id = staff.store_id
  where staff.first_name = 'Mike'
)

-- main query getting the count of movies that were rented for 10 days.
select distinct store_where_mike_works.store_id, count(film.title) as
number_of_movies_rented
from store_where_mike_works
join inventory on store_where_mike_works.store_id = inventory.store_id
join film on inventory.film_id = film.film_id
where film.rental_duration = 10
group by store_where_mike_works.store_id;
```

output:



	store_id	number_of_movies_rented

Explanation: The query uses the CTE ‘store_where_mike_works’ to search and return the store where mike works, for use in the main search. The main search then gets count of film titles, grouped by the store where Mike works, where the rental duration is equal to 10. Note that the most any film can be rented out is 8, therefore it returns an empty list.

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

```
with largest_actors as (
        select film.film_id, count(actor.actor_id) as num_of_actors
  from actor
  join film_actor on actor.actor_id = film_actor.actor_id
  join film on film_actor.film_id = film.film_id
  group by film.film_id
  having num_of_actors >= all ( -- searching for the film with the greatest number of actors
                select count(actor.actor_id)
    from actor
                join film_actor on actor.actor_id = film_actor.actor_id
                join film on film_actor.film_id = film.film_id
                group by film.film_id
  )
)
```

-- Main query, just getting the actors first and last name that are in the result from the CTE search

```
select largest_actors.film_id, actor.first_name, actor.last_name
from largest_actors
join film_actor on largest_actors.film_id = film_actor.film_id
join actor on film_actor.actor_id = actor.actor_id
order by actor.last_name;
```

output:

film_id	first_name	last_name
508	JULIA	BARRYMORE
508	VAL	BOLGER
508	SCARLETT	DAMON
508	LUCILLE	DEE
508	WOODY	HOFFMAN
508	MENA	HOPPER
508	REESE	KILMER
508	CHRISTIAN	NEESON
508	JAYNE	NOLTE
508	BURT	POSEY
508	MENA	TEMPLE
508	WALTER	TORN
508	FAY	WINSLET
508	CAMERON	ZELLWEGER
508	JULIA	ZELLWEGER

Explanation: Using the CTE ‘largest_actor’, the query initially searches for the film with the most actors, grouped by film id. Then in the main search, it returns all the actors’ first names and last names. The search is finally ordered by last name alphabetically.