Title Page

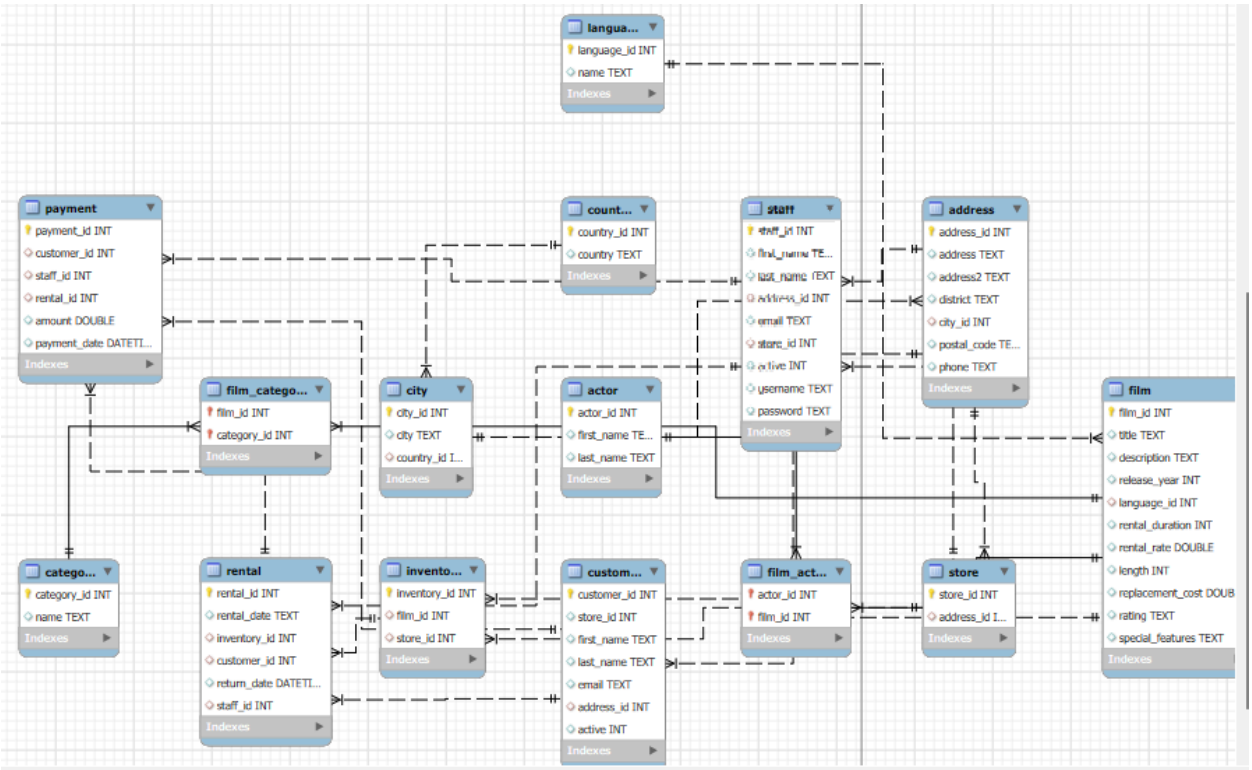
DB Assignment 4

Oscar Nama

11/1/25

SQL Section

ERD Diagram



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

```sql
SELECT AVG(film.length), category.name
FROM film
JOIN film_category ON film.film_id = film_category.film_id
JOIN category ON film_category.category_id = category.category_id
GROUP BY category.name
ORDER BY category.name ASC;
```

Result	Explanation

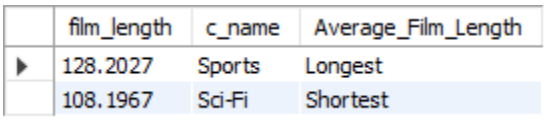
AVG(film.length)	name
111.6094	Action
111.0152	Animation
109.8000	Children
111.6667	Classics
115.8276	Comedy
108.7500	Documentary
120.8387	Drama
114.7826	Family
121.6986	Foreign
127.8361	Games
112.4821	Horror
113.6471	Music
111.1270	New
108.1967	Sci-Fi
128.2027	Sports
113.3158	Travel

I used JOIN statements to group films by category and find their average length

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

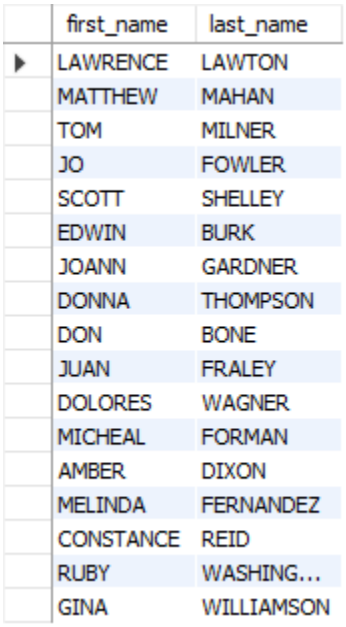
```sql
WITH avg_category_length AS ( -- CTE for the average length of each category
    SELECT AVG(film.length) AS film_length, category.name AS c_name
    FROM film
    JOIN film_category ON film.film_id = film_category.film_id
    JOIN category ON film_category.category_id = category.category_id
    GROUP BY category.name
)
SELECT film_length, c_name, 'Longest' AS Average_Film_Length -- 1st query finds
longest
FROM (
    SELECT film_length, c_name
    FROM avg_category_length
    ORDER BY film_length DESC
    LIMIT 1
) AS longest

UNION -- joins longest and shortest for easier presentation

SELECT film_length, c_name, 'Shortest' AS Average_Film_Length -- 2nd query
finds shortest
FROM (
    SELECT film_length, c_name
    FROM avg_category_length
```

```
    ORDER BY film_length ASC
    LIMIT 1
) AS shortest;
```

Result	Explanation
	Here, I used a CTE to find the average length of each category. Then, my separate queries found the category with the highest average length and the one with the lowest. I used UNION to merge the two tables for easier presentation.

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

```SQL
SELECT DISTINCT customer.first_name, customer.last_name
FROM customer
JOIN rental ON customer.customer_id = rental.customer_id
JOIN inventory ON rental.inventory_id = inventory.inventory_id
JOIN film ON inventory.film_id = film.film_id
JOIN film_category ON film.film_id = film_category.film_id
JOIN category ON film_category.category_id = category.category_id
WHERE category.name = "Action" -- filtering for action
      AND (customer.first_name, customer.last_name) NOT IN ( -- subquery
filters out comedy and classics
             SELECT DISTINCT customer.first_name, customer.last_name
             FROM customer
             JOIN rental ON customer.customer_id = rental.customer_id
             JOIN inventory ON rental.inventory_id = inventory.inventory_id
             JOIN film ON inventory.film_id = film.film_id
             JOIN film_category ON film.film_id = film_category.film_id
             JOIN category ON film_category.category_id = category.category_id
             WHERE category.name IN ("Comedy", "Classics") -- tuple
    );
```

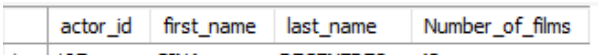
Result	Explanation
	I used a subquery here to find customers who had rented action, but not comedy or classics (using a tuple to group them)

first_name	last_name
LAWRENCE	LAWTON
MATTHEW	MAHAN
TOM	MILNER
JO	FOWLER
SCOTT	SHELLEY
EDWIN	BURK
JOANN	GARDNER
DONNA	THOMPSON
DON	BONE
JUAN	FRALEY
DOLORES	WAGNER
MICHEAL	FORMAN
AMBER	DIXON
MELINDA	FERNANDEZ
CONSTANCE	REID
RUBY	WASHING...
GINA	WILLIAMSON

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

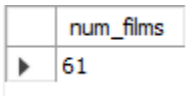
```SQL
SELECT actor.actor_id, actor.first_name, actor.last_name, COUNT(film.film_id)
AS Number_of_films
FROM actor
JOIN film_actor ON actor.actor_id = film_actor.actor_id
JOIN film ON film_actor.film_id = film.film_id
JOIN `language` ON film.language_id = `language`.language_id -- backticks
around language so that mySQL doesn't think its a keyword
```

```
WHERE `language`.name = "English"
GROUP BY actor.actor_id
ORDER BY Number_of_films DESC
LIMIT 1;
```

Result	Explanation
	Here I performed the necessary JOINs to find the single actor with the most English films, filtering for the language name and grouping by actor ID, while limiting to one result.

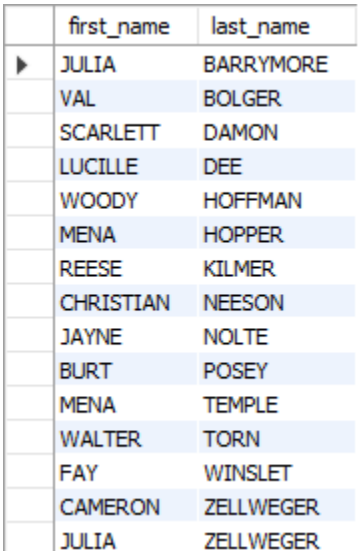
Query #5: How many distinct movies were rented for exactly 10 days from the store where Mike works?

```
SQL
SELECT COUNT(DISTINCT film.film_id) AS num_films
FROM rental
JOIN inventory ON rental.inventory_id = inventory.inventory_id
JOIN film ON inventory.film_id = film.film_id
JOIN store ON inventory.store_id = store.store_id
JOIN staff ON store.store_id = staff.store_id
WHERE staff.first_name = "Mike"
      AND DATEDIFF(rental.return_date, rental.rental_date) = 10; -- DATEDIFF
function finds difference easily
```

Result	Explanation
	Here, I got the count of distinct film IDs that were sold by Mike's store and used the DATEDIFF function to get the amount of time they were rented for, filtering for 10 days.

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

```sql
WITH film_cast_number AS ( -- CTE contains the movie with the biggest cast
       SELECT film_actor.film_id AS movie, COUNT(film_actor.actor_id) AS
num_actors
    FROM film_actor
    GROUP BY movie
    ORDER BY num_actors DESC
    LIMIT 1
)
SELECT DISTINCT actor.first_name, actor.last_name
FROM actor
JOIN film_actor ON actor.actor_id = film_actor.actor_id
JOIN film_cast_number ON film_actor.film_id = film_cast_number.movie
ORDER BY actor.last_name, actor.first_name;
```

Result	Explanation
	I used a CTE to find the movie with the biggest cast, then joined the CTE to my query by the film ID to get only actors who were in that film and list them out.

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