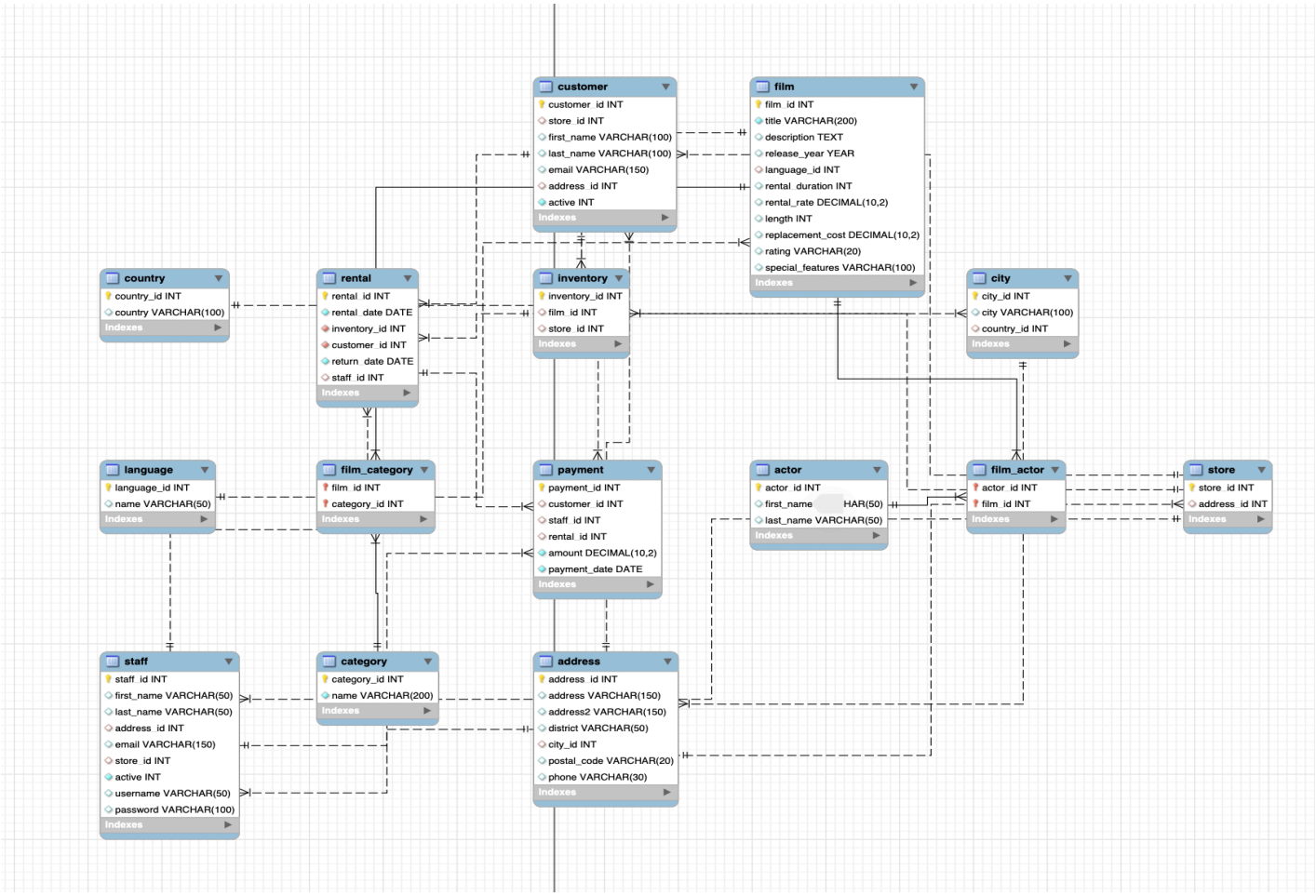
DB Assignment 4

Sarah Groark

October 22, 2024

ERD



Queries

1)

```sql
-- 1. what is the average length of films in each category?
-- list results in alphabetic order of categories
-- joins category on film_category (category_id) and film (film_id) to retrieve film categories + their lengths

select category.name as category_name, round(avg(film.length),2) as average_length
from category
inner join film_category
    on (category.category_id = film_category.category_id)
inner join film
    on (film_category.film_id = film.film_id)
group by category.name
order by category.name asc;
```

This query joins the category table with both the film_category and film tables. Using an inner join, the query joins these tables based on the film lengths of each category. The query groups by the category names and orders them alphabetically.

Output:

category_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

2)

```sql
-- 2. which categories have the longest and shortest average film lengths?
-- uses CTEs to filter through and choose min and max film lengths

WITH AverageFilmLengths AS (

    select c.name as category_name, round(avg(film.length),2) as average_length
    from category c
    inner join film_category
        on (c.category_id = film_category.category_id)
    inner join film
        on (film_category.film_id = film.film_id)
    group by c.name
    order by c.name asc

),

LongestCategory AS (

    select av.category_name as category_name, av.average_length
    from AverageFilmLengths av
    where average_length = (
        select max(average_length)
        from AverageFilmLengths av2
        )

),

ShortestCategory AS (

    select av.category_name, av.average_length
    from AverageFilmLengths av
    where average_length = (
        select min(average_length)
        from AverageFilmLengths av3
        )
)

select l.category_name as longest_category, l.average_length as longest_avg, s.category_name as shortest_category, s.average_length as shortest_avg
from LongestCategory l, ShortestCategory s;
```

This problem utilizes CTEs to output the longest and shortest film durations by category. AverageFilmLengths calculates the average film length per category by joining the category table with the film and film_category tables. The LongestCategory CTE filters the output from AverageFilmLengths to find the longest average length and its category. ShortestCategory, similarly, filters the AverageFilmLengths output to return the shortest average length of a movie and its category. The final selection joins these two results to display the names and average lengths of both categories in one row.

Output:

longest_category	longest_avg	shortest_category	shortest_avg
Sports	128.20	Sci-Fi	108.20

3)

```sql
-- 3. Which customers have rented action but not comedy or classic movies?
select distinct cu.customer_id, cu.first_name, cu.last_name
from customer cu
inner join rental r on (cu.customer_id = r.customer_id)
inner join inventory i on (r.inventory_id = i.inventory_id)
inner join film f on (i.film_id = f.film_id)
inner join film_category fc on (f.film_id = fc.film_id)
inner join category ca on (fc.category_id = ca.category_id)
where ca.name = 'Action'
AND cu.customer_id NOT IN (
    select distinct c2.customer_id
    from customer c2
    JOIN rental r2 ON c2.customer_id = r2.customer_id
    JOIN inventory i2 ON r2.inventory_id = i2.inventory_id
    JOIN film f2 ON i2.film_id = f2.film_id
    JOIN film_category fc2 ON f2.film_id = fc2.film_id
    JOIN category cat2 ON fc2.category_id = cat2.category_id
    WHERE cat2.name IN ('Comedy', 'Classics')

);
```

This query identifies the customers that rented action movies but no comedy or classics movies. This does so with the main selection joining 5 tables to retrieve the customers and the movies they have rented (along with their film category). This output is filtered for movie categories falling under 'Action'. The subquery then excludes the customers that have rented comedy or classics movies by joining the same tables and using the filter for the two excluded movie categories.

Output:

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

4)

```sql
-- 4. Which actor has appeared in the most English-language movies?
select distinct a.actor_id, a.first_name, a.last_name, count(fa.film_id) as movie_count
from actor a
inner join film_actor fa on (a.actor_id = fa.actor_id)
inner join film f on (fa.film_id = f.film_id)
inner join language l on (f.language_id = l.language_id)
where l.name = 'English'
group by a.actor_id, a.first_name, a.last_name
having count(fa.film_id) >= all(
    select count(fa2.film_id) as movie_count
    from film_actor fa2
    inner join film f2 on (fa2.film_id = f2.film_id)
    inner join language l2 on (f2.language_id = l2.language_id)
    where l2.name = 'English'
    group by fa2.actor_id
);
```

This query finds the actor who has been featured in the most English-speaking movies. The main selection finds the actors and the amount of movies they've been featured in where the language for those movies is English. This is possible due to the joining of the actor, film_actor, film, and language tables. The subquery then identifies the actor(s) with the highest of these counts.

Output:

actor_id	first_name	last_name	movie_count
107	GINA	DEGENERES	42

5)

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

select count(distinct f.film_id) as distinct_movie_count
from rental r
inner join inventory i on (r.inventory_id = i.inventory_id)
inner join film f on (i.film_id = f.film_id)
inner join store s on (i.store_id = s.store_id)
inner join staff st on (s.store_id = st.store_id)
where DATEDIFF(r.return_date, r.rental_date) = 10
    and st.first_name = 'Mike';
```

This query counts the number of movies rented for exactly 10 days from the specific store where Mike works at. This query joins the rental table with the inventory, film, store, and staff tables. This retrieves the rental records and identifies the distinct amount of movies in the store by filtering the staff first names for 'Mike' where the rental duration is 10 days (calculated using DATEDIFF()).

Output:

distinct_movie_count
61

6)

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

select a.first_name, a.last_name
from actor a
inner join film_actor fa on (a.actor_id = fa.actor_id)
where fa.film_id = (
    select fa2.film_id
    from film_actor fa2
    group by fa2.film_id
    order by count(fa2.actor_id) desc
    limit 1
)
order by a.first_name, a.last_name;
```

This query utilizes the main selection to list the actors that have participated in a movie. Then, the subquery finds the movie with the largest cast by selecting the top record (in descending order) of the films with the highest count of actors in that film. The final line of the query orders the actors name alphabetically by first name.

Output:

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