Homework Assignment #1

Data Management for Data Science

<u>Student</u>

Vigèr Durand AZIMEDEM TSAFACK (1792126)

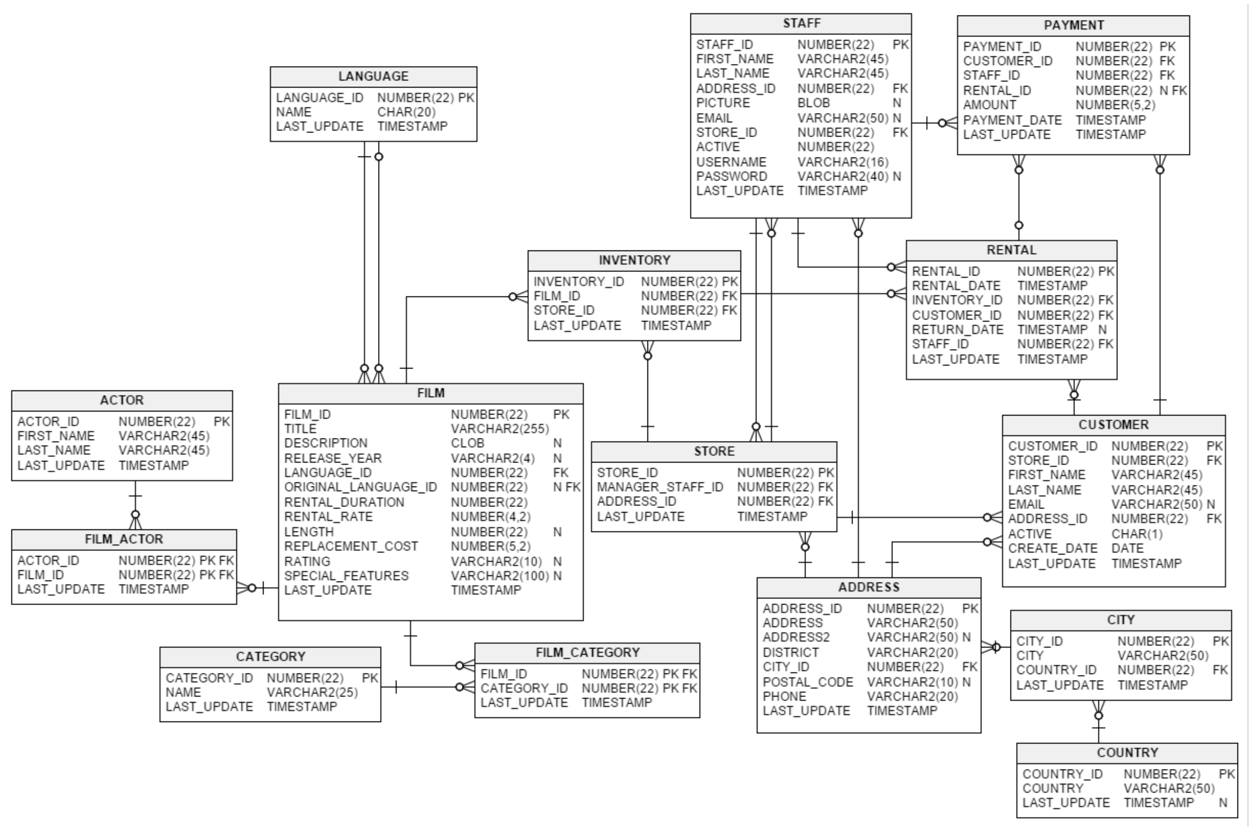
<u>Data</u>

We'll use the **Sakila** schema, which can be found on the following link.
http://dev.mysql.com/doc/index-other.html ("sakila database")

The MySQL's Sample Salika (DVD Rental) Database is a complex database with 16 tables. It also illustrates features such as Views, Stored Procedures and Triggers.

We will dig into a deep understanding of this database and then we will formulate (and solve) a set of 10 SQL queries on it. We are dealing with a nicely normalized schema modelling a DVD rental store, featuring things like films, actors, film-actor relationships, and a central inventory table that connects films, stores, and rentals.

Follows a graph showing relationships between all the tables of the database:



Queries

1. What are the names of all the languages in the database (sorted alphabetically)?

```sql
select l.name
from language l
order by l.name asc;
```

name
English
Italian
Japanese
Mandarin
French
German

2. Return the full names (first and last) of actors with “SON” in their last name, ordered by their first name.

```sql
select CONCAT(first_name, ' ', last_name) as full_name
from actor
-- where last_name REGEXP '.*SON.*'
where last_name like '%SON%'
order by first_name asc;
```

full_name
ALBERT JOHANSSON
ANGELA HUDSON
BETTE NICHOLSON
CHRISTIAN NEESON
JAYNE NEESON
MATTHEW JOHANSSON
MERYL GIBSON
RAY JOHANSSON
WILL WILSON

3. Find all the addresses where the district is not empty (i.e., contains some text), and return these districts sorted.

```sql
select address, district
from address
where district is not null and district not like ''
order by district;
```

address	district
669 Firozabad Loop	Abu Dhabi
535 Ahmadnagar Manor	Abu Dhabi
1078 Stara Zagora Drive	Aceh
842 Salzburg Lane	Adana
663 Baha Blanca Parkway	Adana
614 Pak Kret Street	Addis Abeba
751 Lima Loop	Aden
1157 Nyeri Loop	Adygea
387 Mwene-Ditu Drive	Ahal
775 ostka Drive	al-Daqahliya
1759 Niznekamsk Avenue	al-Manama
1152 Citrus Heights Manor	al-Qadarif
1987 Coacalco de Berriozbal Loop	al-Qalyubiya
765 Southampton Drive	al-Qalyubiya
289 Santo Andr Manor	al-Sharqiya

4. Return the first and last names of actors who played in a film involving a “Crocodile” and a “Shark”, along with the release year of the movie, sorted by the actors’ last names.

```sql
select a.first_name, a.last_name, f.release_year, f.title, f.description
from film_actor fa
    inner join film f on fa.film_id = f.film_id
    inner join actor a on fa.actor_id = a.actor_id
where CONCAT(f.title, ' ', f.description) like '%crocodile%'
and CONCAT(f.title, ' ', f.description) like '%shark%'
order by a.last_name asc;
```

first_name	last_name	release_year	title	description
KIRSTEN	AKROYD	2006	MADIGAN DORADO	A Astounding Character Study of a A Shark And a A Shark who must Discover a Crocodile in The Outback
KIM	ALLEN	2006	CLEOPATRA DEVIL	A Fanciful Documentary of a Crocodile And a Technical Writer who must Fight a A Shark in A Baloon
AUDREY	BAILEY	2006	WARLOCK WEREWOLF	A Astounding Yarn of a Pioneer And a Crocodile who must Defeat a A Shark in The Outback
JULIA	BARRYMORE	2006	SHOOTIST SUPERFLY	A Fast-Paced Story of a Crocodile And a A Shark who must Sink a Pioneer in Berlin
VIVIEN	BASINGER	2006	CONNECTICUT TRAMP	A Unbelieveable Drama of a Crocodile And a Mad Cow who must Reach a Dentist in A Shark Tank
VIVIEN	BERGEN	2006	EXCITEMENT EVE	A Brilliant Documentary of a Monkey And a Car who must Conquer a Crocodile in A Shark Tank

5. How many films involve a “Crocodile” and a “Shark”?

```sql
select count(*)
from film f
where CONCAT(f.title, ' ', f.description) like '%crocodile%'
and CONCAT(f.title, ' ', f.description) like '%shark%';
```

count(*)
10

6. What is the average running time of films by category?

```sql
 select fc.category_id, avg(length) as avg_running_time
 from film f, film_category fc
 where  f.film_id = fc.film_id
 group by fc.category_id;
```

category_id	avg_running_time
1	111.6094
2	111.0152
3	109.8000
4	111.6667
5	115.8276
6	108.7500
7	120.8387
8	114.7826
9	121.6986
10	127.8361
11	112.4821
12	113.6471
13	111.1270
14	108.1967
15	128.2027
16	113.3158

7. Which actor has appeared in the most films?

```sql
select a.actor_id, a.first_name, a.last_name, film_count
from
    (select actor_id, count(*) as film_count
    from film_actor fa
    group by fa.actor_id) as fa inner join actor a on fa.actor_id = a.actor_id
order by film_count desc limit 1;
```

actor_id	first_name	last_name	film_count
107	GINA	DEGENERES	42

8. When is each copy(inventory) of 'Academy Dinosaur' due?

```sql
select inventory_id, max(return_date)
from rental
where inventory_id in (
    select inventory_id
    from inventory
    where film_id = (
        select film_id
        from film
        where title = 'Academy Dinosaur'
        )
    )
    group by inventory_id;
```

inventory_id	max(return_date)
1	2005-08-30 22:26:43
2	2005-08-30 20:08:01
3	2005-08-25 18:58:37
4	2005-08-23 21:09:42
6	2005-08-01 04:08:11
7	2005-08-27 02:18:08
8	2005-08-22 22:01:16

9. Find all the film categories in which there are between 55 and 65 films. Return the names of these categories and the number of films per category, sorted by the number of films.

```sql
select category.name as category_name, film_count
from (select category_id, count(*) as film_count
        from film_category
      group by category_id) as film_category
      inner join category on category.category_id = film_category.category_id
where film_count > 55 and film_count < 65
order by film_count asc;
```

category_name	film_count
Horror	56
Travel	57
Classics	57
Comedy	58
Children	60
Games	61

Sci-Fi	61
Drama	62
New	63
Action	64

10. In how many film categories is the average difference between the film replacement cost and the rental rate larger than 17?

```sql
select count(*)
from (
    select name, avg(replacement_rental_diff) as avg_replacement_rental_diff
    from (
        select category.name, (replacement_cost - rental_rate) as replacement_rental_diff

        from film_category
            inner join film on film.film_id = film_category.film_id
            inner join category on category.category_id = film_category.category_id
    ) as replacement
        group by name
     ) as replacement_avg
     where avg_replacement_rental_diff > 17
```

count(*)
8

References

https://www.ntu.edu.sg/home/ehchua/programming/sql/SampleDatabases.html
https://www.jooq.org/sakila