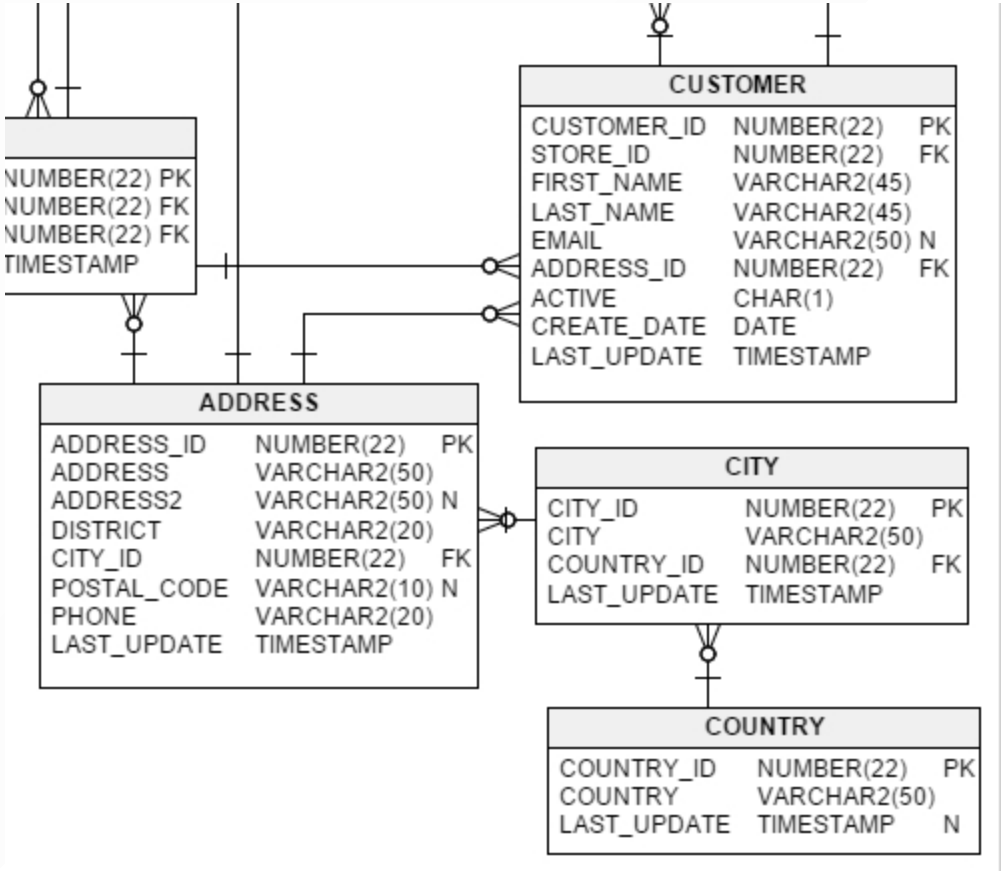
Lab4 Querying Multiple Tables

Lab4 Agenda

- Querying Multiple Tables
- Εργαστηριακές Ασκήσεις
- Εξαμηνιαία Εργασία

Querying Multiple Tables



```
MariaDB [sakila]> desc customer;
+-------------+------------------+------+-----+---------------------+-------------------------------+
| Field       | Type             | Null | Key | Default             | Extra                         |
+-------------+------------------+------+-----+---------------------+-------------------------------+
| customer_id | int(10) unsigned | NO   | PRI | NULL                | auto_increment                |
| store_id    | int(10) unsigned | NO   | MUL | NULL                |                               |
| first_name  | varchar(45)      | NO   |     | NULL                |                               |
| last_name   | varchar(45)      | NO   | MUL | NULL                |                               |
| email       | varchar(50)      | YES  |     | NULL                |                               |
| address_id  | int(10) unsigned | NO   | MUL | NULL                |                               |
| active      | tinyint(1)       | NO   |     | 1                   |                               |
| create_date | datetime         | NO   |     | NULL                |                               |
| last_update | timestamp        | NO   |     | current_timestamp() | on update current_timestamp() |
+-------------+------------------+------+-----+---------------------+-------------------------------+
MariaDB [sakila]> desc address;
+-------------+------------------+------+-----+---------------------+-------------------------------+
| Field       | Type             | Null | Key | Default             | Extra                         |
+-------------+------------------+------+-----+---------------------+-------------------------------+
| address_id  | int(10) unsigned | NO   | PRI | NULL                | auto_increment                |
| address     | varchar(50)      | NO   |     | NULL                |                               |
| address2    | varchar(50)      | YES  |     | NULL                |                               |
| district    | varchar(20)      | NO   |     | NULL                |                               |
| city_id     | int(10) unsigned | NO   | MUL | NULL                |                               |
| postal_code | varchar(10)      | YES  |     | NULL                |                               |
| phone       | varchar(20)      | NO   |     | NULL                |                               |
| last_update | timestamp        | NO   |     | current_timestamp() | on update current_timestamp() |
+-------------+------------------+------+-----+---------------------+-------------------------------+
```

foreign key constraint

- verify that the values in one table exist in another table
- it is not necessary to have a foreign key constraint in place in order to join two tables

simple joins

- cross join - Cartesian Product

```
SELECT c.first_name, c.last_name, a.address
FROM customer c JOIN address a;
```

```
  ...
| WADE         | DELVALLE      | 1325 Fukuyama Street                |
| AUSTIN       | CINTRON       | 1325 Fukuyama Street                |
+--------------+---------------+-------------------------------------+
361197 rows in set (0.071 sec)
```

599 customers x 603 addresses = 361.197

INNER JOIN

- describe how the two tables are related
- inner join by default, if not specified
- customer without address missing from result

- ANSI SQL92 Syntax

```
SELECT c.first_name, c.last_name, a.address
FROM customer c JOIN address a
ON c.address_id = a.address_id;
```

```
| ENRIQUE      | FORSYTHE      | 1101 Bucuresti Boulevard                |
| FREDDIE      | DUGGAN        | 1103 Quilmes Boulevard                  |
| WADE         | DELVALLE      | 1331 Usak Boulevard                     |
| AUSTIN       | CINTRON       | 1325 Fukuyama Street                    |
+--------------+---------------+-----------------------------------------+
599 rows in set (0.001 sec)
```

Prior syntax ❌

```
SELECT c.first_name, c.last_name, a.address
FROM customer c, address a
WHERE c.address_id = a.address_id;
```

```
MariaDB [sakila]> SELECT c.first_name, c.last_name, a.address
    -> FROM customer c, address a
    -> WHERE c.address_id = a.address_id;
+-------------+--------------+-----------------------------------+
| first_name  | last_name    | address                           |
+-------------+--------------+-----------------------------------+
| MARY        | SMITH        | 1913 Hanoi Way                    |
| PATRICIA    | JOHNSON      | 1121 Loja Avenue                  |
| LINDA       | WILLIAMS     | 692 Joliet Street                 |
```

OLD vs ANSI SQL92 Syntax

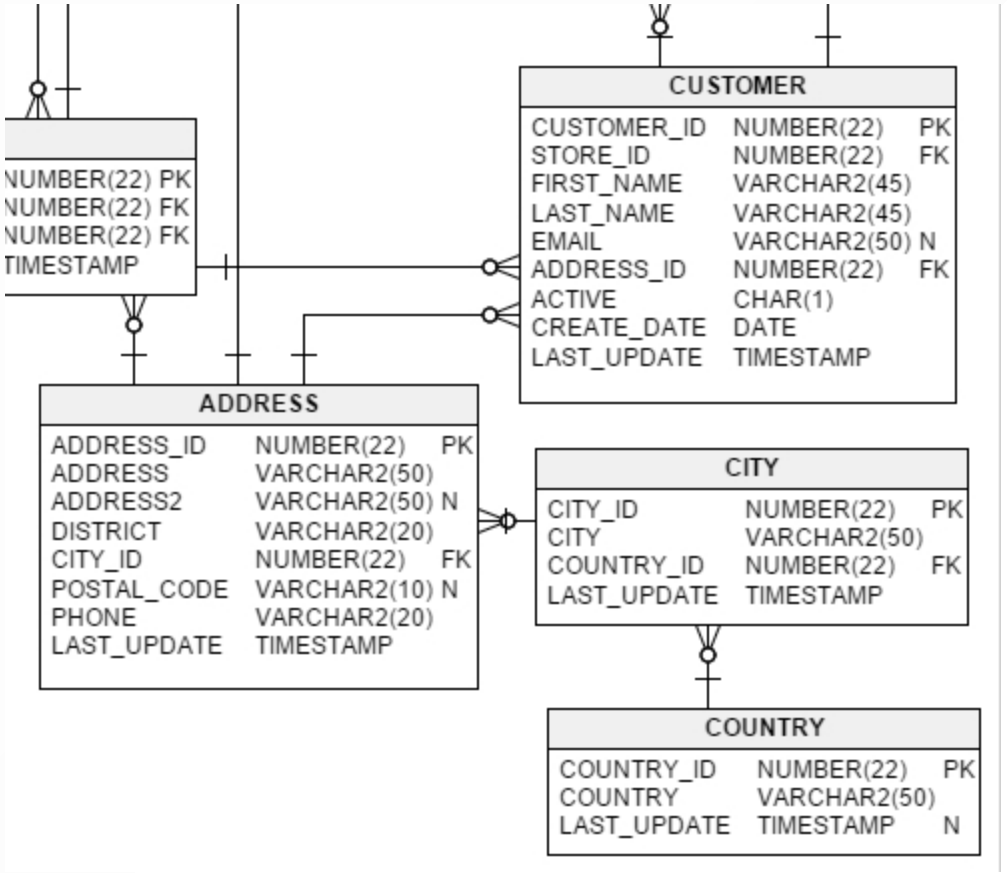
😕

```sql
SELECT c.first_name, c.last_name, a.address
FROM customer c, address a
WHERE c.address_id = a.address_id
AND a.postal_code = 52137;
```

😉

```sql
SELECT c.first_name, c.last_name, a.address
FROM customer c INNER JOIN address a
ON c.address_id = a.address_id
WHERE a.postal_code = 52137;
```

3 or more tables

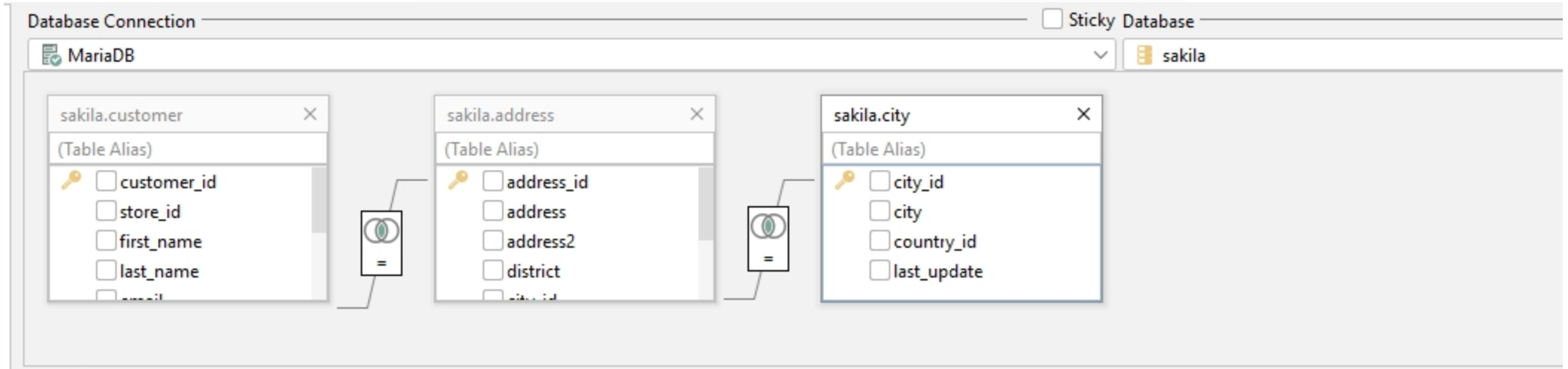


- show each customer's city

```
SELECT c.first_name, c.last_name, ct.city
FROM customer c
INNER JOIN address a
ON c.address_id = a.address_id
INNER JOIN city ct
ON a.city_id = ct.city_id;
```

```
+-------------+--------------+---------------------------+
| first_name  | last_name    | city                      |
+-------------+--------------+---------------------------+
| MARY        | SMITH        | Sasebo                    |
| PATRICIA    | JOHNSON      | San Bernardino            |
                           ...
```

3 or more tables




```sql
SELECT
    *
FROM
    sakila.customer
INNER JOIN
    sakila.address
ON
    (
        sakila.customer.address_id = sakila.address.address_id)
INNER JOIN
    sakila.city
ON
    (
        sakila.address.city_id = sakila.city.city_id) ;

```


changing the order

```sql
SELECT c.first_name, c.last_name, ct.city
FROM customer c
INNER JOIN address a
ON c.address_id = a.address_id
INNER JOIN city ct
ON a.city_id = ct.city_id;

SELECT c.first_name, c.last_name, ct.city
FROM city ct
INNER JOIN address a
ON a.city_id = ct.city_id
INNER JOIN customer c
ON c.address_id = a.address_id;

SELECT c.first_name, c.last_name, ct.city
FROM address a
INNER JOIN city ct
ON a.city_id = ct.city_id
INNER JOIN customer c
ON c.address_id = a.address_id;
```

Using Subqueries as Tables

```
SELECT c.first_name, c.last_name, addr.address, addr.city
 FROM customer c
 INNER JOIN
  (SELECT a.address_id, a.address, ct.city
    FROM address a
    INNER JOIN city ct
    ON a.city_id = ct.city_id
    WHERE ct.city  = 'Dallas'
  ) addr
 ON c.address_id = addr.address_id;
```

```
+------------+-----------+-------------------+--------+
| first_name | last_name | address           | city   |
+------------+-----------+-------------------+--------+
| BRYAN      | HARDISON  | 530 Lausanne Lane | Dallas |
+------------+-----------+-------------------+--------+
1 row in set (0.02 sec)
```

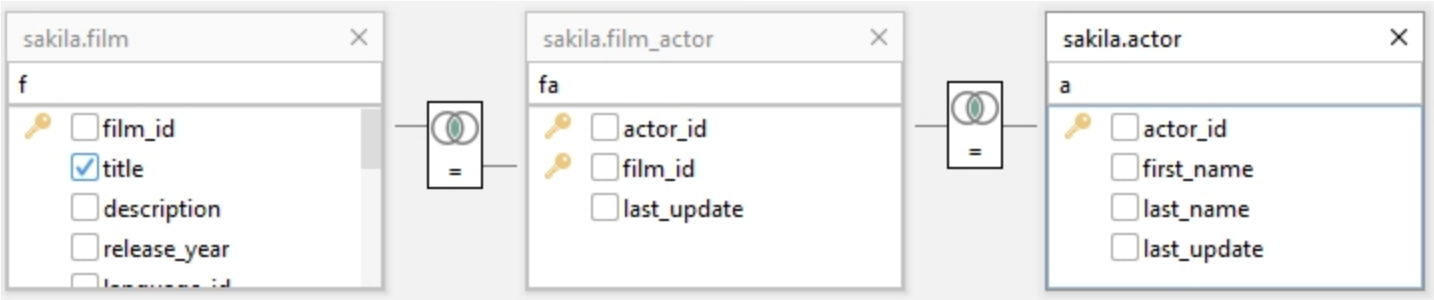
Using the Same Table Twice

- actors are related to the films in which they appeared via the film_actor table
- find all of the films in which a specific actor appeared

```
SELECT f.title
  FROM film f
    INNER JOIN film_actor fa
    ON f.film_id = fa.film_id
    INNER JOIN actor a
    ON fa.actor_id = a.actor_id
  WHERE (a.first_name = 'ED' AND a.last_name = 'CHASE')
```

```
+--------------------+
| title              |
+--------------------+
| ALONE TRIP         |
| ARMY FLINTSTONES   |
| ARTIST COLDBLOODED |
| BOONDOCK BALLROOM  |
| CADDYSHACK JEDI    |
      ...
| HUNTER ALTER       |
| IMAGE PRINCESS     |
| JEEPERS WEDDING    |
| LUCK OPUS          |
| NECKLACE OUTBREAK  |
| PLATOON INSTINCT   |
| SPICE SORORITY     |
| WEDDING APOLLO     |
| WEEKEND PERSONAL   |
| WHALE BIKINI       |
| YOUNG LANGUAGE     |
+--------------------+
22 rows in set (0.000 sec)
```

find all of the films in which a specific actor appeared



retrieve only those films in which 2 actors appeared

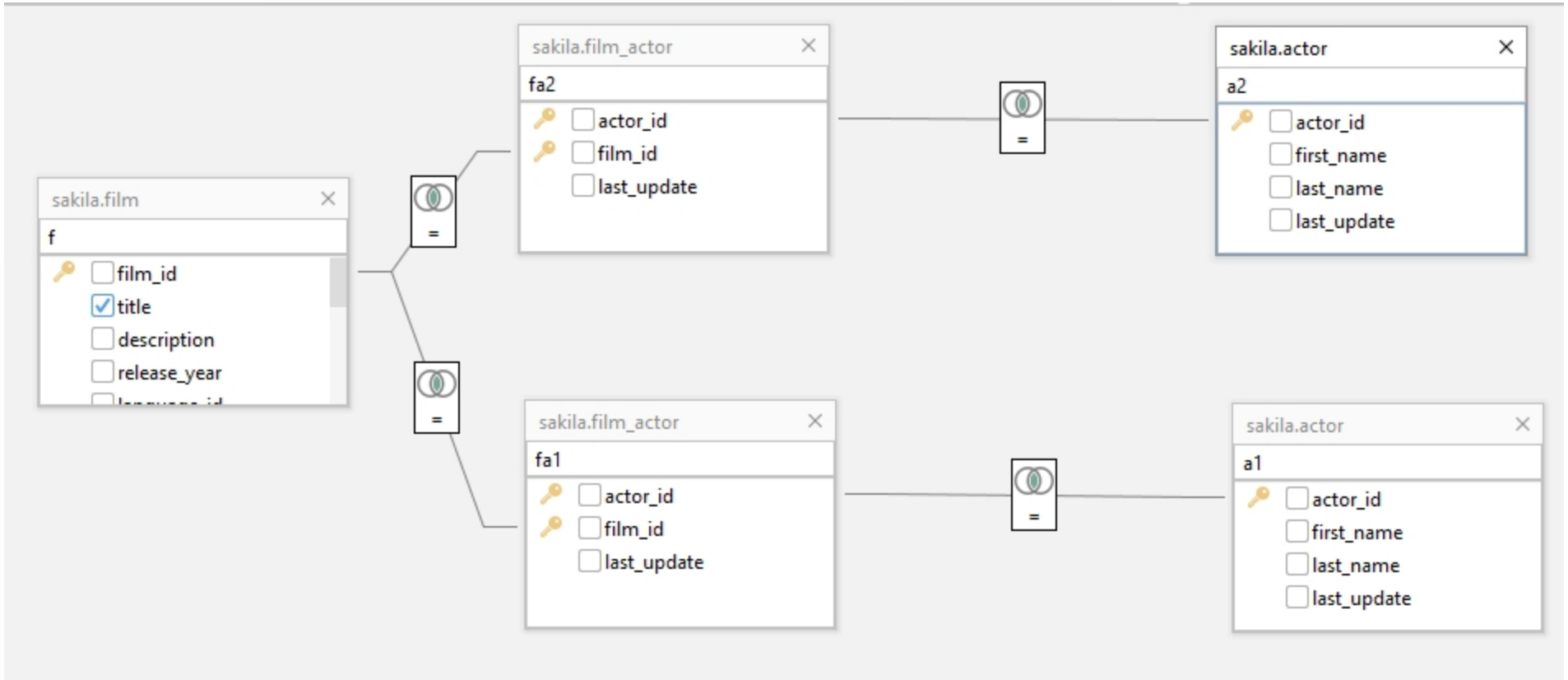
- e.g. ED CHASE and KARL BERRY

```
SELECT f.title
FROM film f
INNER JOIN film_actor fa1
  ON f.film_id = fa1.film_id
INNER JOIN actor a1
  ON fa1.actor_id = a1.actor_id
INNER JOIN film_actor fa2
  ON f.film_id = fa2.film_id
INNER JOIN actor a2
  ON fa2.actor_id = a2.actor_id
WHERE (a1.first_name = 'ED' AND a1.last_name = 'CHASE')
AND (a2.first_name = 'KARL' AND a2.last_name = 'BERRY');
```

retrieve only those films in which 2 actors appeared

```
+-------------+
| title       |
+-------------+
| ALONE TRIP |
+-------------+
1 row in set (0.001 sec)
```

retrieve only those films in which 2 actors appeared



Self-Joins

- join a table to itself
  - retrieve all customers whose last name matches the first name of another customer

```sql
SELECT
        c1.customer_id as customer_1_id,
        c1.first_name as customer_1_first_name,
        c1.last_name as customer_1_last_name,
        c2.customer_id as customer_2_id,
        c2.first_name as customer_2_first_name,
        c2.last_name
FROM customer c1
INNER JOIN customer c2
ON c1.last_name = c2.first_name
ORDER BY c1.last_name;
```

```
+---------------+-----------------------+----------------------+---------------+-----------------------+--------------+
| customer_1_id | customer_1_first_name | customer_1_last_name | customer_2_id | customer_2_first_name | last_name    |
+---------------+-----------------------+----------------------+---------------+-----------------------+--------------+
|           504 | NATHANIEL             | ADAM                 |           367 | ADAM                  | GOOCH        |
|            96 | DIANA                 | ALEXANDER            |           439 | ALEXANDER             | FENNELL      |
|            27 | SHIRLEY               | ALLEN                |           412 | ALLEN                 | BUTTERFIELD  |
                                          ...
|           267 | MARGIE                | WADE                 |           598 | WADE                  | DELVALLE     |
|           106 | CONNIE                | WALLACE              |           562 | WALLACE               | SLONE        |
|           138 | HAZEL                 | WARREN               |           462 | WARREN                | SHERROD      |
+---------------+-----------------------+----------------------+---------------+-----------------------+--------------+
51 rows in set (0.060 sec)
```

Εργαστηριακές Ασκήσεις 💻

1. List customers (first_name , last_name, address) living in Mexicali
2. List every film (title) in which an actor with the first name 'JOHN' appeared
3. How many distinct actors last names are there?
4. List all addresses that are in the same city.
5. Is 'Academy Dinosaur' available for rent from Store 1?
6. List all customers whose last name matches the first name of another customer.
7. List the total paid by each customer. (List the customers alphabetically by last name)
8. You want to run an email marketing campaign in Greece, for which you will need the names and email addresses of all Greek customers.

Εξαμηνιαία Εργασία 💻

- ~~Database Schema Design~~
  1. Start thinking about the entities you need
     - Identify entities, attributes and relationships from the problem description
     - identify cardinality ratios of the relationships found
  2. Design an E/R diagram for your database
     - Look for any issues that are apparent in the E/R diagram

Εξαμηνιαία Εργασία 💻

- Materialize Schema: DDL statements
  1. Create your tables
     - create a table for each entity
     - a table (representing an entity) should have:
       - a column for each attribute, with appropriate data type
       - a primary key and possibly some candidate keys

Wrap Up

1. [x] `Querying Multiple Tables`
2. [x] `Εργαστηριακές Ασκήσεις`
3. [x] `Εξαμηνιαία Εργασία`

Wrap Up

😱 **Απορίες** https://discord.gg/g3fFxWVPfD

Εργαστηριακές Ασκήσεις / Απαντήσεις

1. List customers (first_name, last_name, address) living in Mexicali

```
SELECT c.first_name, c.last_name, a.address, ct.city  FROM customer c
INNER JOIN address a  ON c.address_id = a.address_id
INNER JOIN city ct  ON a.city_id = ct.city_id
WHERE ct.city = 'Mexicali';
```

```
+------------+-----------+------------------+----------+
| first_name | last_name | address          | city     |
+------------+-----------+------------------+----------+
| DOUGLAS    | GRAF      | 785 Vaduz Street | Mexicali |
+------------+-----------+------------------+----------+
```

Εργαστηριακές Ασκήσεις / Απαντήσεις

2. List every film (title) in which an actor with the first name 'JOHN' appeared

```
SELECT f.title FROM film f
INNER JOIN film_actor fa  ON f.film_id = fa.film_id
INNER JOIN actor a   ON fa.actor_id = a.actor_id
WHERE a.first_name = 'JOHN';
```

```
+--------------------------+
| title                    |
+--------------------------+
| ALLEY EVOLUTION          |
| BEVERLY OUTLAW           |
      ...
| SONG HEDWIG              |
+--------------------------+
29 rows in set (0.002 sec)
```

Εργαστηριακές Ασκήσεις / Απαντήσεις

3. How many distinct actors last names are there?

```
select count(distinct last_name) from actor;
```

```
>select count(distinct last_name) as lastName from actor;
+----------+
| lastName |
+----------+
|      121 |
+----------+
1 row in set (0.00 sec)
```

Εργαστηριακές Ασκήσεις / Απαντήσεις

4. List all addresses that are in the same city.

```
SELECT a1.address addr1, a2.address addr2, a1.city_id
FROM address a1
INNER JOIN address a2
WHERE a1.city_id = a2.city_id
AND a1.address_id <> a2.address_id;
```

Εργαστηριακές Ασκήσεις / Απαντήσεις

5. Is 'Academy Dinosaur' available for rent from Store 1?

```
SELECT f.film_id, f.title,  s.store_id, i.inventory_id FROM  store s
INNER JOIN inventory i ON s.store_id = i.store_id
INNER JOIN film f ON i.film_id = f.film_id
WHERE f.title = 'Academy Dinosaur' AND s.store_id = 1 ;
```

```
+---------+-----------------+----------+--------------+
| film_id | title           | store_id | inventory_id |
+---------+-----------------+----------+--------------+
|       1 | ACADEMY DINOSAUR |        1 |            1 |
|       1 | ACADEMY DINOSAUR |        1 |            2 |
|       1 | ACADEMY DINOSAUR |        1 |            3 |
|       1 | ACADEMY DINOSAUR |        1 |            4 |
+---------+-----------------+----------+--------------+
4 rows in set (0.001 sec)
```

Εργαστηριακές Ασκήσεις / Απαντήσεις

6. List all customers whose last name matches the first name of another customer

```
SELECT
  a.customer_id, a.first_name, a.last_name,
  b.customer_id, b.first_name, b.last_name
FROM customer a
INNER JOIN customer b
ON a.last_name = b.first_name;
```

Εργαστηριακές Ασκήσεις / Απαντήσεις

7. List the total paid by each customer. (List the customers alphabetically by last name).

```
select concat(c.first_name,' ',c.last_name) as 'Customer Name', sum(p.amount) as 'Total Paid'
from payment as p
join customer as c
on p.customer_id = c.customer_id
group by p.customer_id;
```

Εργαστηριακές Ασκήσεις / Απαντήσεις

8. You want to run an email marketing campaign in Greece, for which you will need the names and email addresses of all Greek customers.

```
SELECT c.first_name, c.last_name, c.email FROM customer c
INNER JOIN address a ON (c.address_id = a.address_id)
INNER JOIN city ci ON (a.city_id = ci.city_id)
INNER JOIN country ctr ON (ci.country_id = ctr.country_id)
WHERE ctr.country = 'Greece';
```

```
LINDA   WILLIAMS        LINDA.WILLIAMS@sakilacustomer.org
ENRIQUE FORSYTHE        ENRIQUE.FORSYTHE@sakilacustomer.org
4 rows in set (0.001 sec)
```