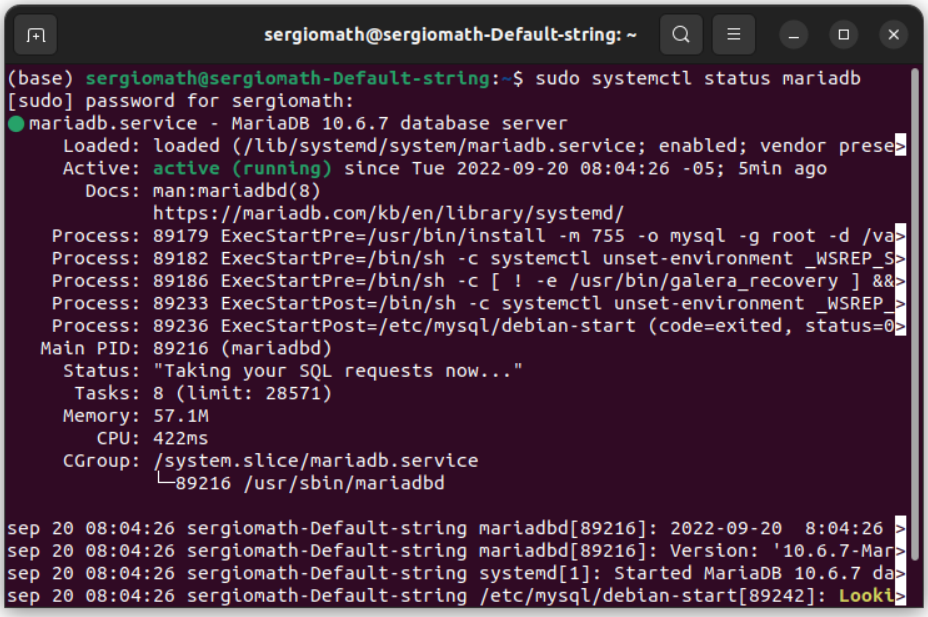
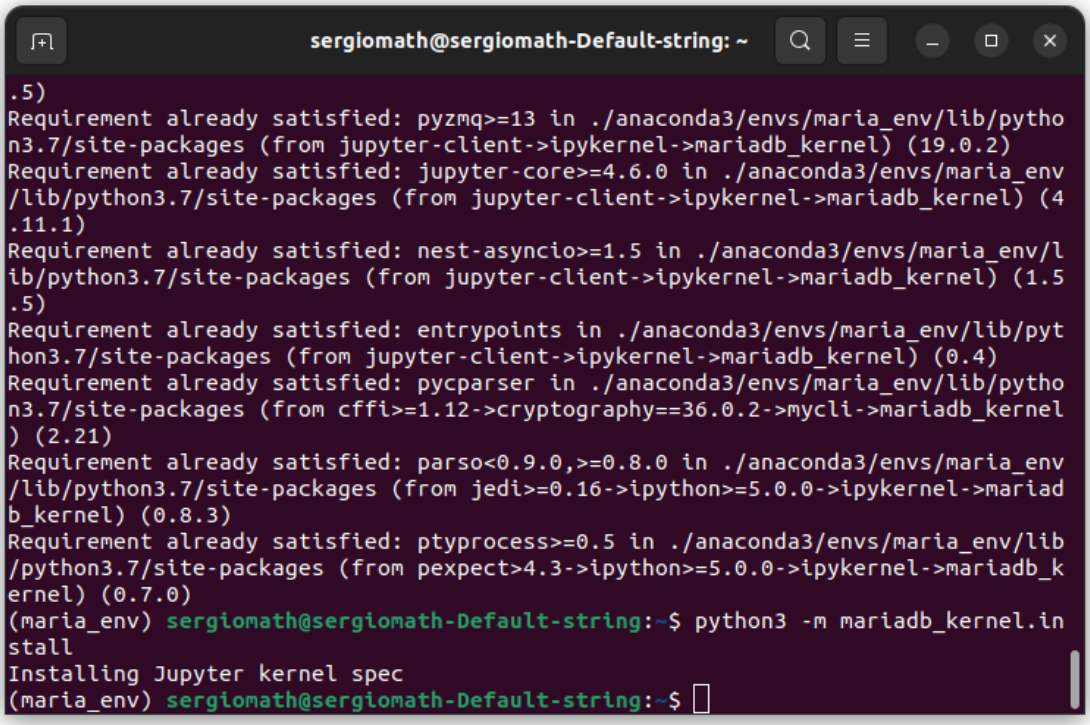
1. Instale localmente en su computadora SQL o MariaD



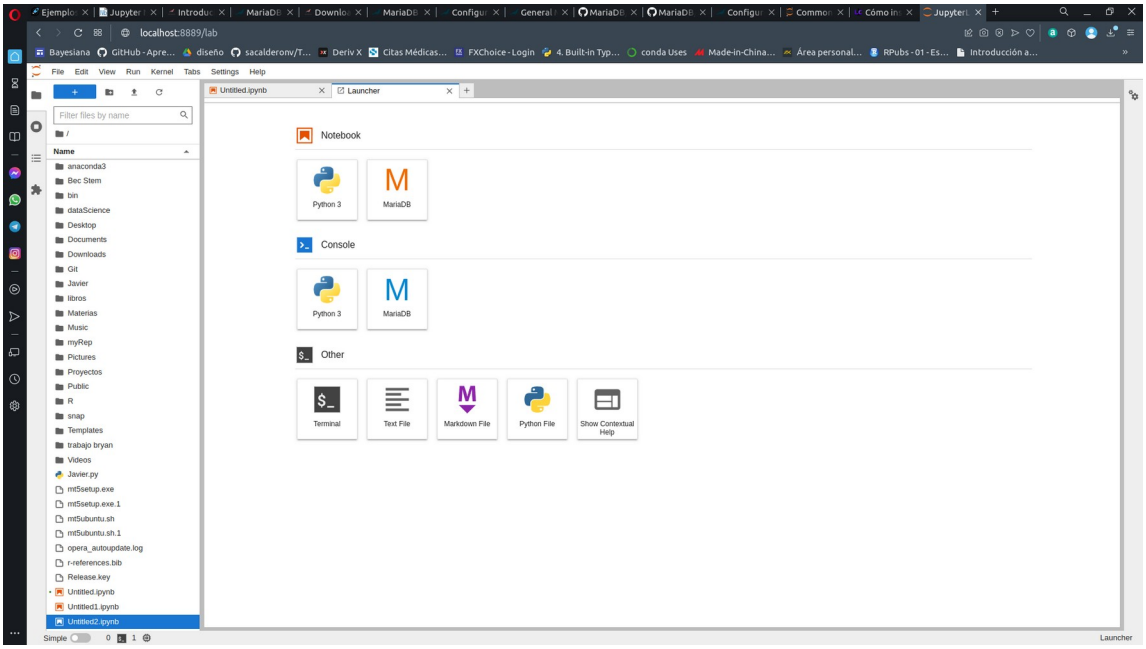
2. Instale la base de datos salika. Ver enlace abajo y siga las instrucciones del cuaderno de Jupyter lab

sakila instalada

3. Cree una ambiente de anaconda nuevo, instale Python y Jupyter lab. Agregue la extensión para MariaDB

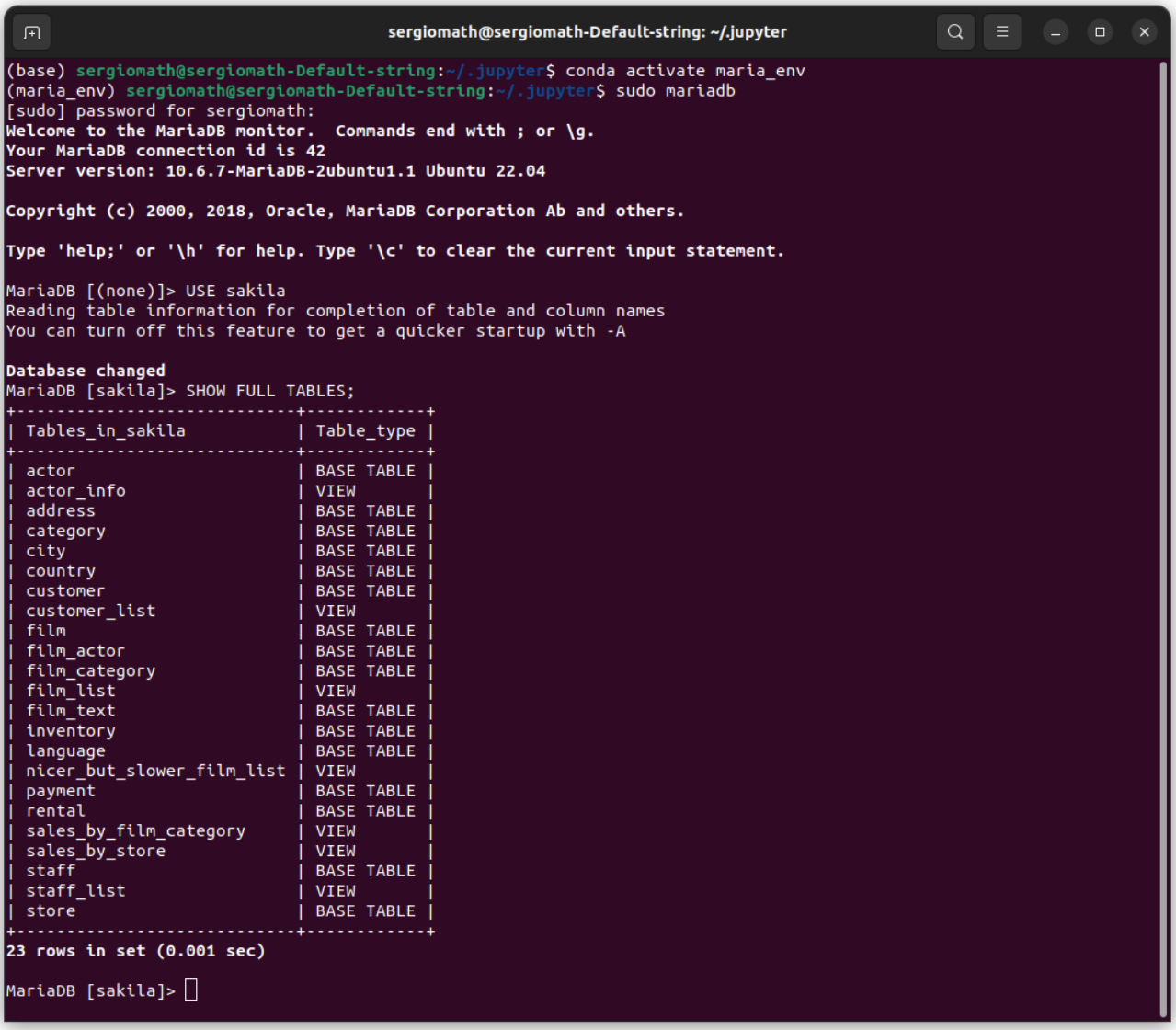


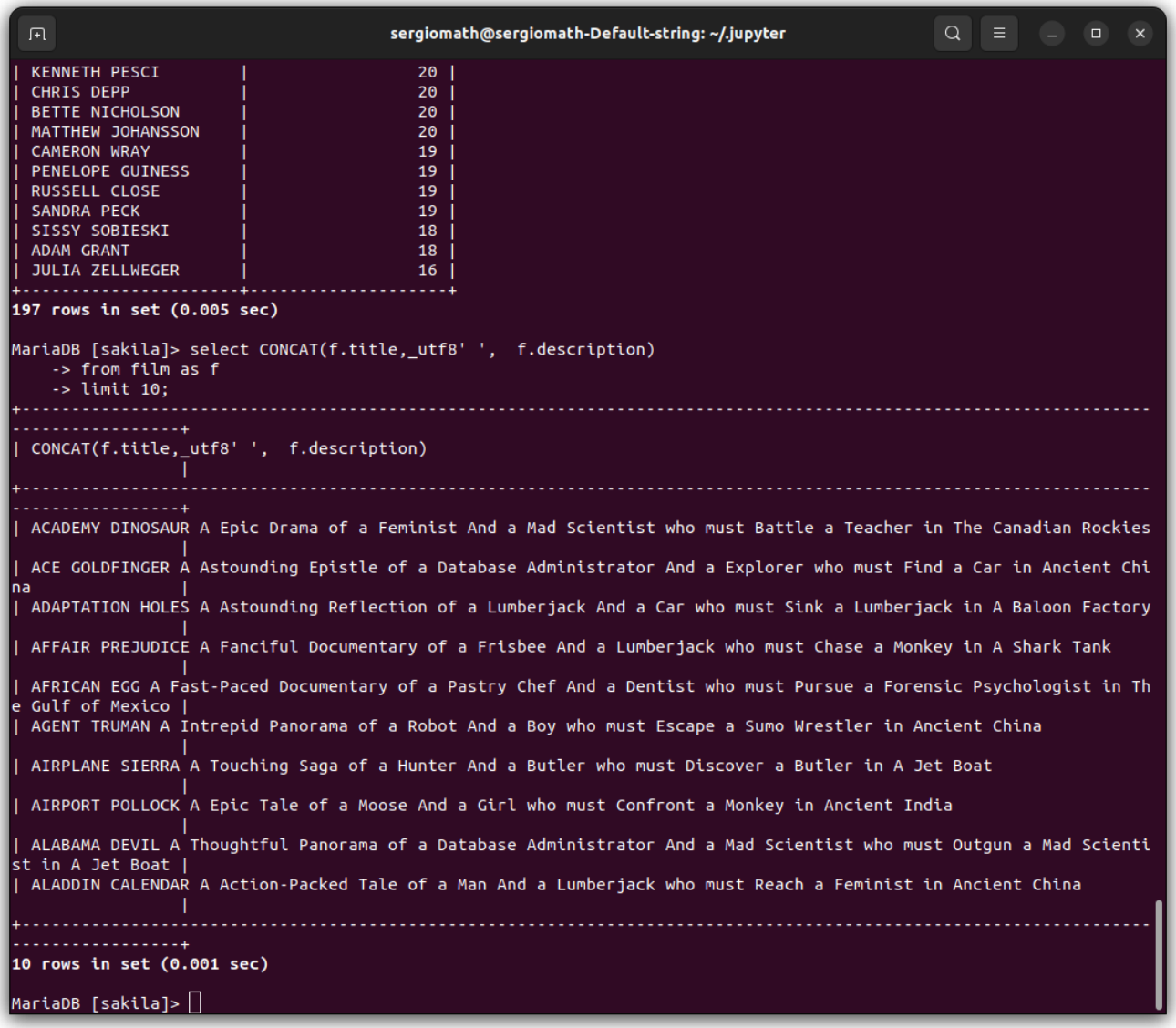
Ambiente creado

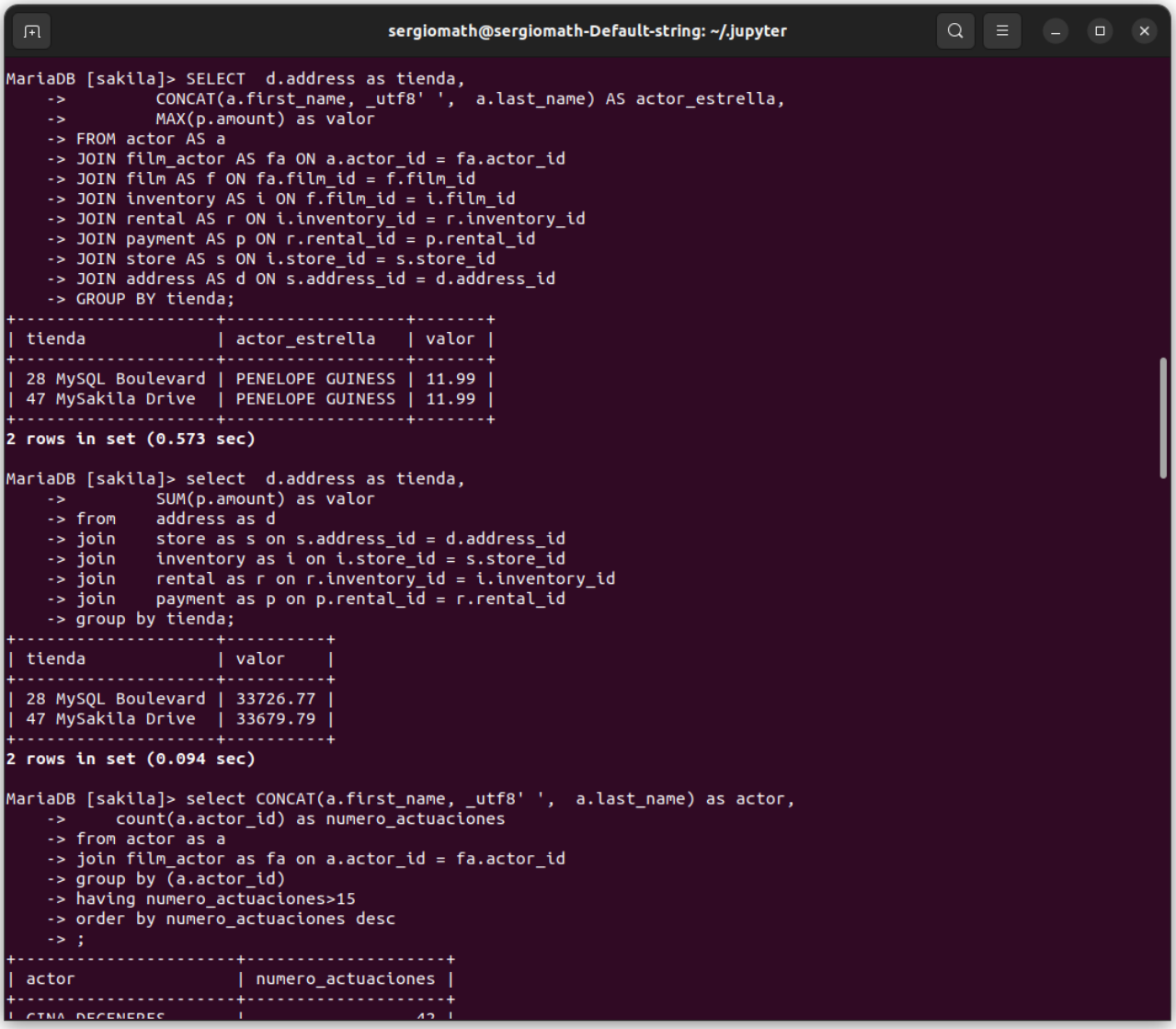


Extension mariaDB

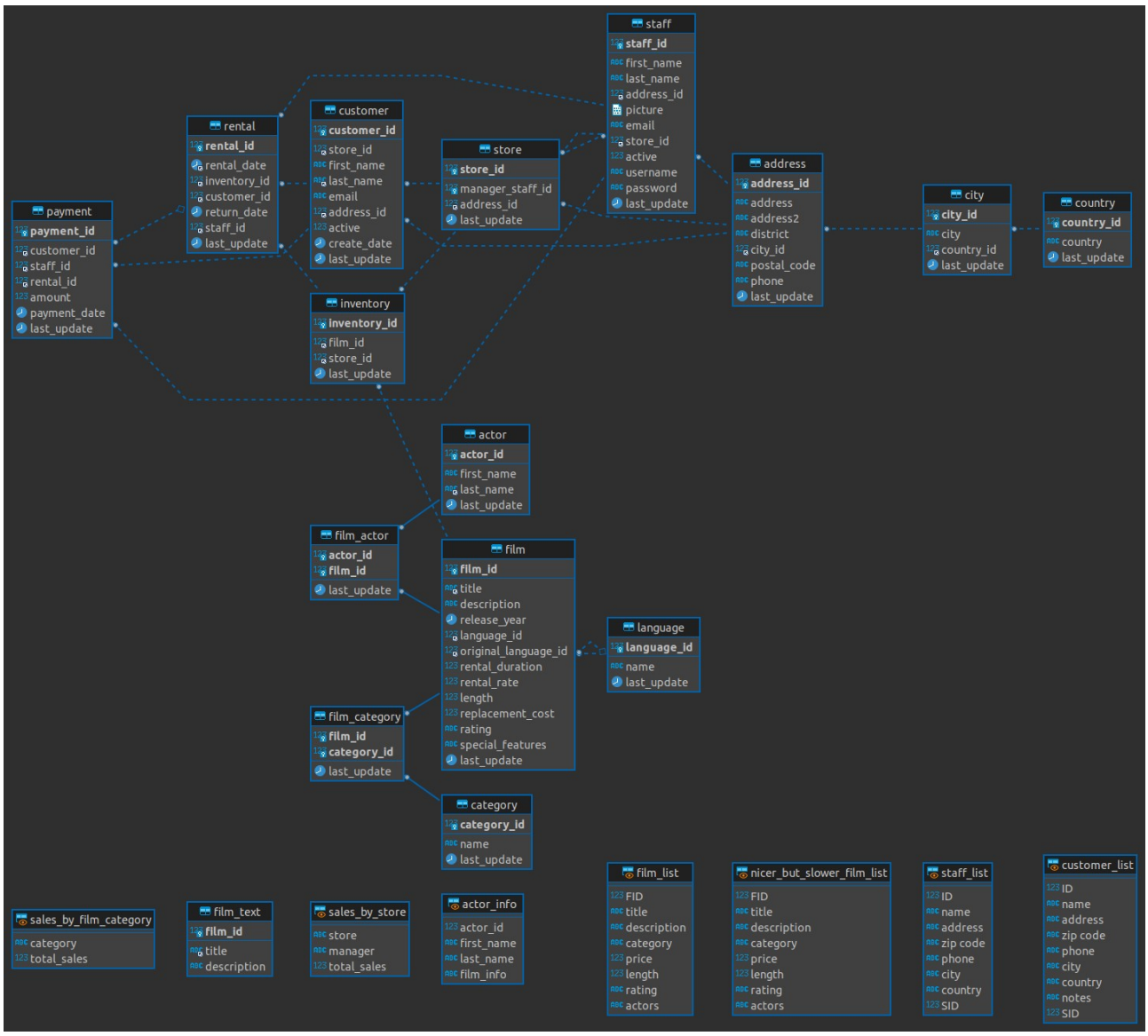
4. Ejecute la guía del enlace, consultado la base directamente desde la consola de MariaDB.



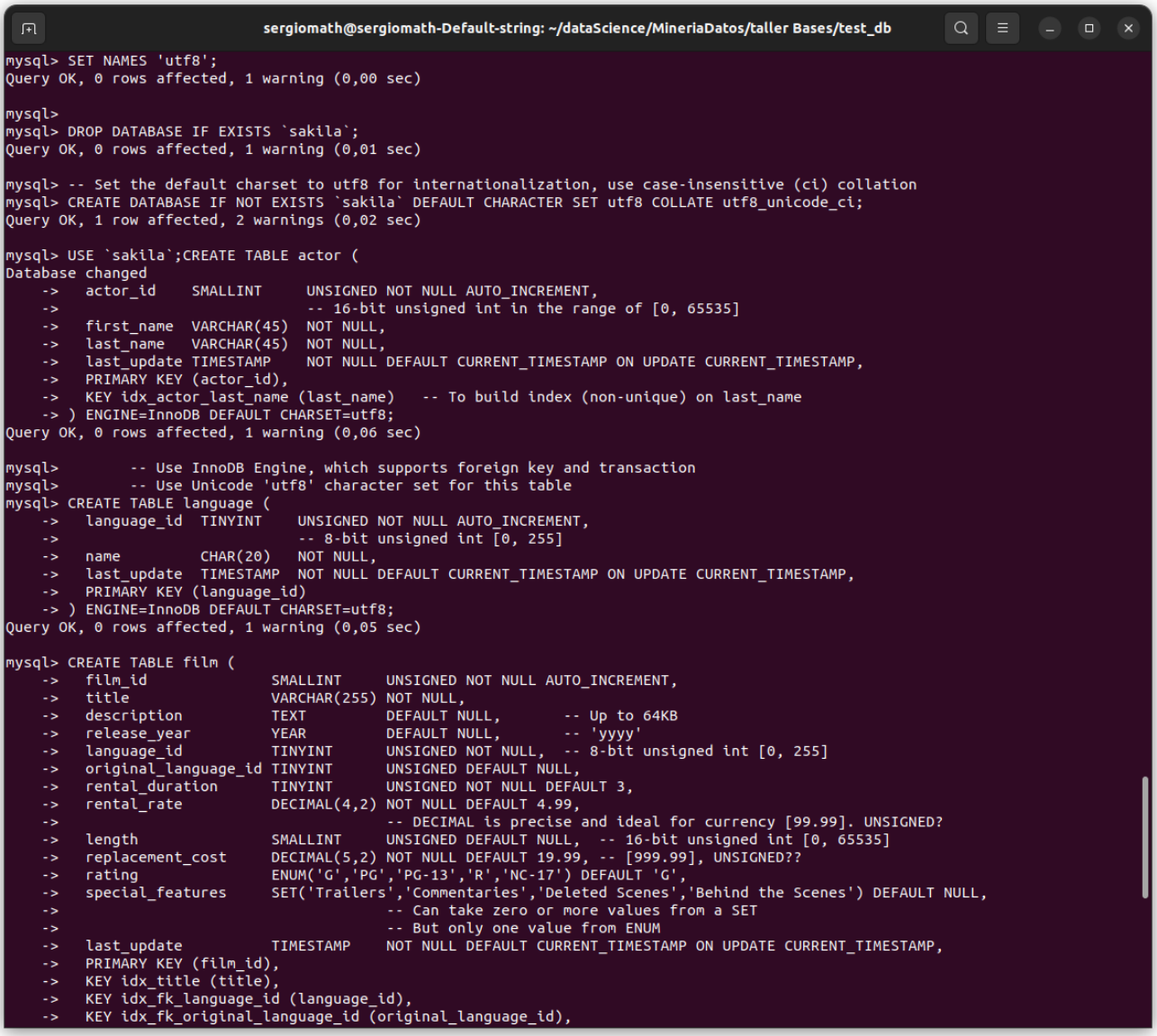

sergiomath@sergiomath-Default-string: ~/.jupyter
KENNETH PESCI	20
CHRIS DEPP	20
BETTE NICHOLSON	20
MATTHEW JOHANSSON	20
CAMERON WRAY	19
PENELOPE GUINESS	19
RUSSELL CLOSE	19
SANDRA PECK	19
SISSY SOBIESKI	18
ADAM GRANT	18
JULIA ZELLWEGER	16
197 rows in set (0.005 sec)
MariaDB [sakila]> select CONCAT(f.title,_utf8' ', f.description)
-> from film as f
-> limit 10;
| CONCAT(f.title,_utf8' ', f.description)
| ACADEMY DINOSAUR A Epic Drama of a Feminist And a Mad Scientist who must Battle a Teacher in The Canadian Rockies
| ACE GOLDFINGER A Astounding Epistle of a Database Administrator And a Explorer who must Find a Car in Ancient China
| ADAPTATION HOLES A Astounding Reflection of a Lumberjack And a Car who must Sink a Lumberjack in A Baloon Factory
| AFFAIR PREJUDICE A Fanciful Documentary of a Frisbee And a Lumberjack who must Chase a Monkey in A Shark Tank
| AFRICAN EGG A Fast-Paced Documentary of a Pastry Chef And a Dentist who must Pursue a Forensic Psychologist in The Gulf of Mexico
| AGENT TRUMAN A Intrepid Panorama of a Robot And a Boy who must Escape a Sumo Wrestler in Ancient China
| AIRPLANE SIERRA A Touching Saga of a Hunter And a Butler who must Discover a Butler in A Jet Boat
| AIRPORT POLLOCK A Epic Tale of a Moose And a Girl who must Confront a Monkey in Ancient India
| ALABAMA DEVIL A Thoughtful Panorama of a Database Administrator And a Mad Scientist who must Outgun a Mad Scientist in A Jet Boat
| ALADDIN CALENDAR A Action-Packed Tale of a Man And a Lumberjack who must Reach a Feminist in Ancient China
10 rows in set (0.001 sec)
MariaDB [sakila]>

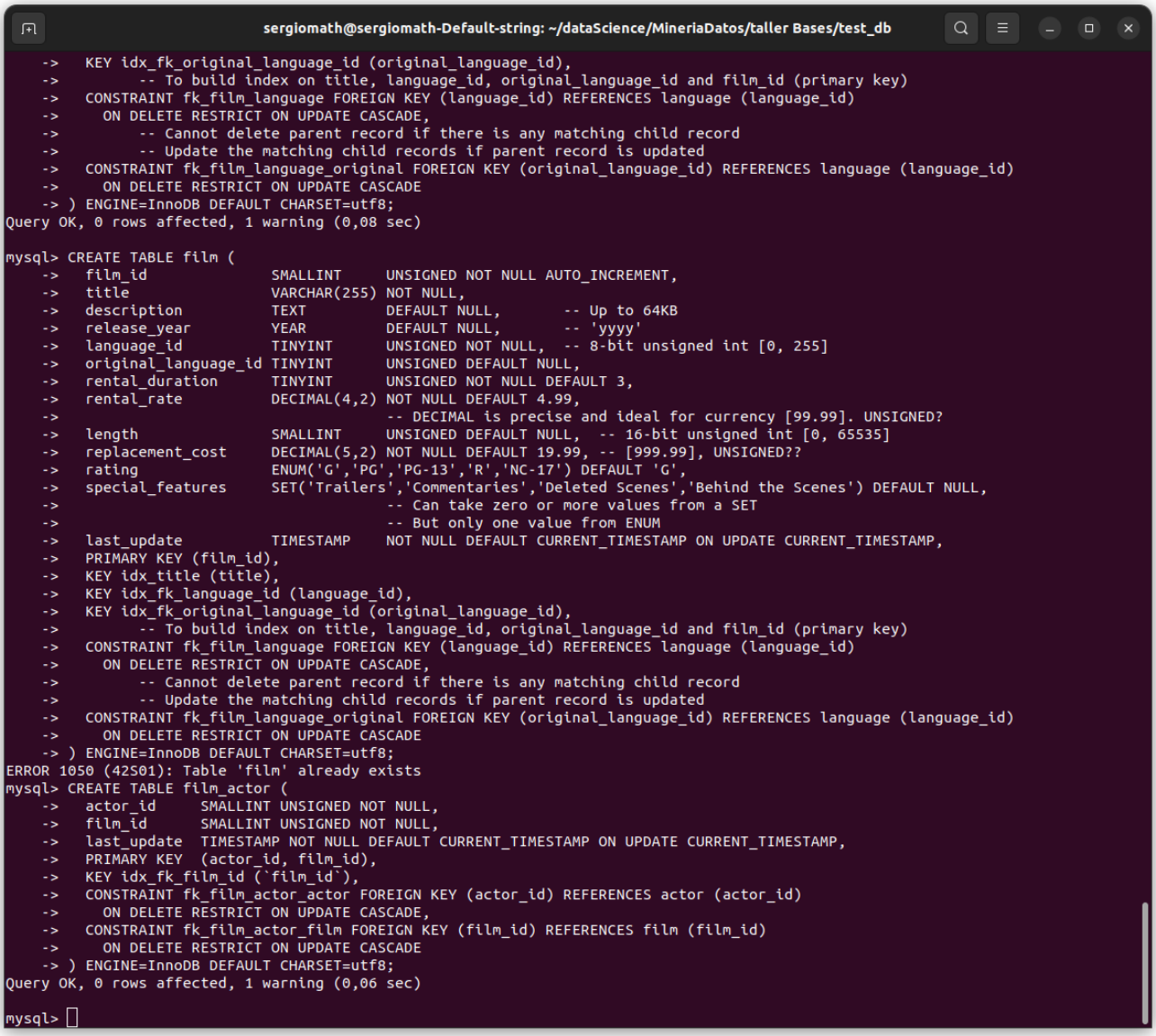

sergiomath@sergiomath-Default-string: ~/.jupyter
MariaDB [sakila]> SELECT d.address as tienda,
    ->   CONCAT(a.first_name, _utf8' ', a.last_name) AS actor_estrella,
    ->   MAX(p.amount) as valor
    -> FROM actor AS a
    -> JOIN film_actor AS fa ON a.actor_id = fa.actor_id
    -> JOIN film AS f ON fa.film_id = f.film_id
    -> JOIN inventory AS i ON f.film_id = i.film_id
    -> JOIN rental AS r ON i.inventory_id = r.inventory_id
    -> JOIN payment AS p ON r.rental_id = p.rental_id
    -> JOIN store AS s ON i.store_id = s.store_id
    -> JOIN address AS d ON s.address_id = d.address_id
    -> GROUP BY tienda;
+--------------------+-----------------+-------+
| tienda | actor_estrella | valor |
+--------------------+-----------------+-------+
| 28 MySQL Boulevard | PENELOPE GUINESS | 11.99 |
| 47 MySakila Drive | PENELOPE GUINESS | 11.99 |
+--------------------+-----------------+-------+
2 rows in set (0.573 sec)
MariaDB [sakila]> select d.address as tienda,
    ->   SUM(p.amount) as valor
    -> from   address as d
    -> join   store as s on s.address_id = d.address_id
    -> join   inventory as i on i.store_id = s.store_id
    -> join   rental as r on r.inventory_id = i.inventory_id
    -> join   payment as p on p.rental_id = r.rental_id
    -> group by tienda;
+--------------------+----------+
| tienda | valor |
+--------------------+----------+
| 28 MySQL Boulevard | 33726.77 |
| 47 MySakila Drive | 33679.79 |
+--------------------+----------+
2 rows in set (0.094 sec)
MariaDB [sakila]> select CONCAT(a.first_name, _utf8' ', a.last_name) as actor,
    ->   count(a.actor_id) as numero_actuaciones
    -> from actor as a
    -> join film_actor as fa on a.actor_id = fa.actor_id
    -> group by (a.actor_id)
    -> having numero_actuaciones>15
    -> order by numero_actuaciones desc
    -> ;
+---------------------+--------------------+
| actor | numero_actuaciones |
+---------------------+--------------------+

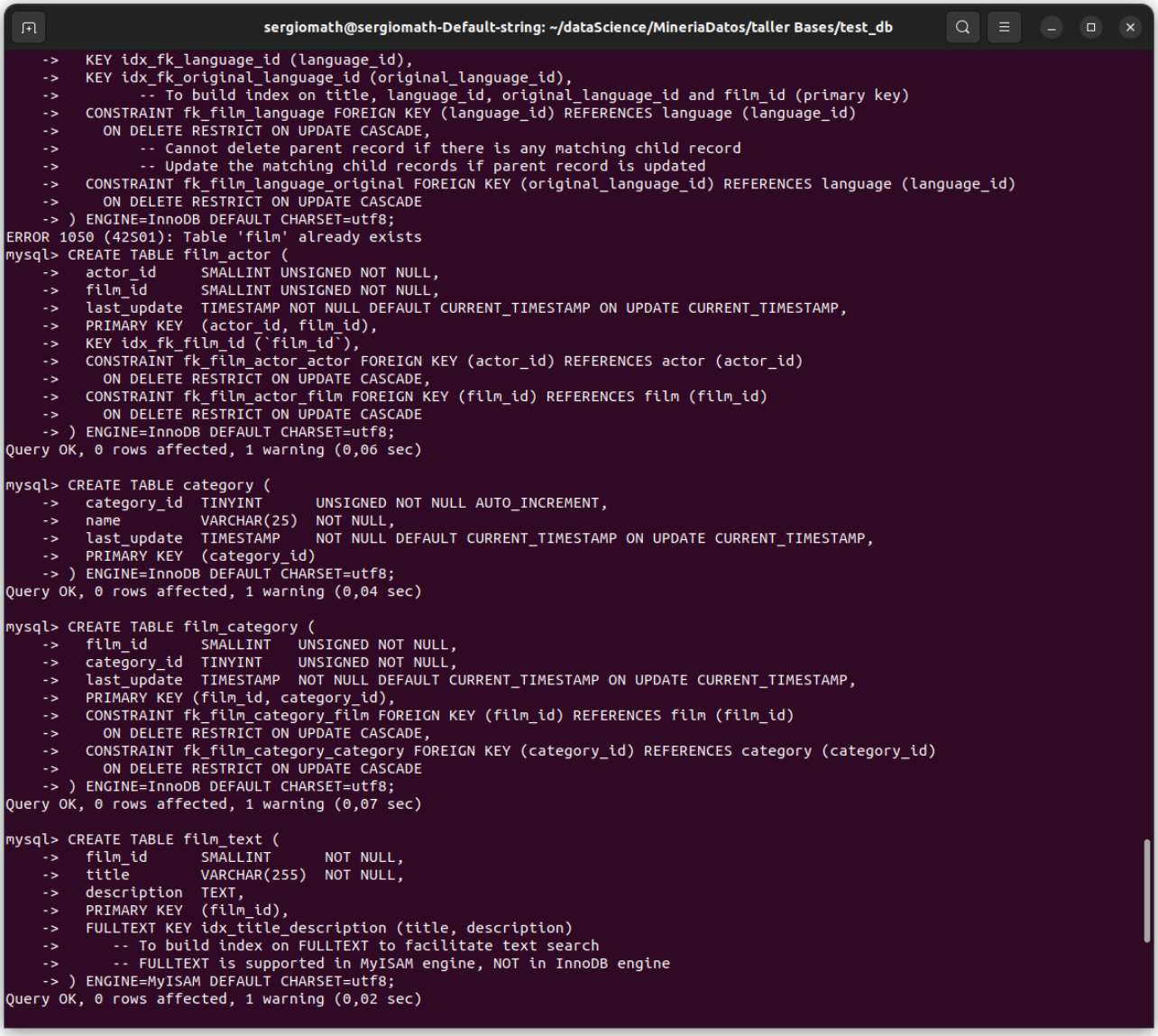
5. Obtenga el diagrama entidad-relación usando algún designe como Workbench o DBeaver.



6.Escriba las consultas propuestas en el tutorial deirectamente en la consola y tomo fotos de evidencia del trabajo.




sergiomath@sergiomath-Default-string: ~/dataScience/MineriaDatos/taller Bases/test_db
-> KEY idx_fk_original_language_id (original_language_id),
-> -- To build index on title, language_id, original_language_id and film_id (primary key)
-> CONSTRAINT fk_film_language FOREIGN KEY (language_id) REFERENCES language (language_id)
-> ON DELETE RESTRICT ON UPDATE CASCADE,
-> -- Cannot delete parent record if there is any matching child record
-> -- Update the matching child records if parent record is updated
-> CONSTRAINT fk_film_language_original FOREIGN KEY (original_language_id) REFERENCES language (language_id)
-> ON DELETE RESTRICT ON UPDATE CASCADE
->) ENGINE=InnoDB DEFAULT CHARSET=utf8;
Query OK, 0 rows affected, 1 warning (0,08 sec)
mysql> CREATE TABLE film (
-> film_id SMALLINT UNSIGNED NOT NULL AUTO_INCREMENT,
-> title VARCHAR(255) NOT NULL,
-> description TEXT DEFAULT NULL, -- Up to 64KB
-> release_year YEAR DEFAULT NULL, -- 'yyyy'
-> language_id TINYINT UNSIGNED NOT NULL, -- 8-bit unsigned int [0, 255]
-> original_language_id TINYINT UNSIGNED DEFAULT NULL,
-> rental_duration TINYINT UNSIGNED NOT NULL DEFAULT 3,
-> rental_rate DECIMAL(4,2) NOT NULL DEFAULT 4.99,
-> -- DECIMAL is precise and ideal for currency [99.99]. UNSIGNED?
-> length SMALLINT UNSIGNED DEFAULT NULL, -- 16-bit unsigned int [0, 65535]
-> replacement_cost DECIMAL(5,2) NOT NULL DEFAULT 19.99, -- [999.99], UNSIGNED??
-> rating ENUM('G','PG','PG-13','R','NC-17') DEFAULT 'G',
-> special_features SET('Trailers','Commentaries','Deleted Scenes','Behind the Scenes') DEFAULT NULL,
-> -- Can take zero or more values from a SET
-> -- But only one value from ENUM
-> last_update TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP,
-> PRIMARY KEY (film_id),
-> KEY idx_title (title),
-> KEY idx_fk_language_id (language_id),
-> KEY idx_fk_original_language_id (original_language_id),
-> -- To build index on title, language_id, original_language_id and film_id (primary key)
-> CONSTRAINT fk_film_language FOREIGN KEY (language_id) REFERENCES language (language_id)
-> ON DELETE RESTRICT ON UPDATE CASCADE,
-> -- Cannot delete parent record if there is any matching child record
-> -- Update the matching child records if parent record is updated
-> CONSTRAINT fk_film_language_original FOREIGN KEY (original_language_id) REFERENCES language (language_id)
-> ON DELETE RESTRICT ON UPDATE CASCADE
->) ENGINE=InnoDB DEFAULT CHARSET=utf8;
ERROR 1050 (42S01): Table 'film' already exists
mysql> CREATE TABLE film_actor (
-> actor_id SMALLINT UNSIGNED NOT NULL,
-> film_id SMALLINT UNSIGNED NOT NULL,
-> last_update TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP,
-> PRIMARY KEY (actor_id, film_id),
-> KEY idx_fk_film_id (`film_id`),
-> CONSTRAINT fk_film_actor_actor FOREIGN KEY (actor_id) REFERENCES actor (actor_id)
-> ON DELETE RESTRICT ON UPDATE CASCADE,
-> CONSTRAINT fk_film_actor_film FOREIGN KEY (film_id) REFERENCES film (film_id)
-> ON DELETE RESTRICT ON UPDATE CASCADE
->) ENGINE=InnoDB DEFAULT CHARSET=utf8;
Query OK, 0 rows affected, 1 warning (0,06 sec)
mysql>


sergiomath@sergiomath-Default-string: ~/dataScience/MineriaDatos/taller Bases/test_db
-> KEY idx_fk_language_id (language_id),
-> KEY idx_fk_original_language_id (original_language_id),
-> -- To build index on title, language_id, original_language_id and film_id (primary key)
-> CONSTRAINT fk_film_language FOREIGN KEY (language_id) REFERENCES language (language_id)
-> ON DELETE RESTRICT ON UPDATE CASCADE,
-> -- Cannot delete parent record if there is any matching child record
-> -- Update the matching child records if parent record is updated
-> CONSTRAINT fk_film_language_original FOREIGN KEY (original_language_id) REFERENCES language (language_id)
-> ON DELETE RESTRICT ON UPDATE CASCADE
->) ENGINE=InnoDB DEFAULT CHARSET=utf8;
ERROR 1050 (42S01): Table 'film' already exists
mysql> CREATE TABLE film_actor (
-> actor_id SMALLINT UNSIGNED NOT NULL,
-> film_id SMALLINT UNSIGNED NOT NULL,
-> last_update TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP,
-> PRIMARY KEY (actor_id, film_id),
-> KEY idx_fk_film_id (`film_id`),
-> CONSTRAINT fk_film_actor_actor FOREIGN KEY (actor_id) REFERENCES actor (actor_id)
-> ON DELETE RESTRICT ON UPDATE CASCADE,
-> CONSTRAINT fk_film_actor_film FOREIGN KEY (film_id) REFERENCES film (film_id)
-> ON DELETE RESTRICT ON UPDATE CASCADE
->) ENGINE=InnoDB DEFAULT CHARSET=utf8;
Query OK, 0 rows affected, 1 warning (0,06 sec)
mysql> CREATE TABLE category (
-> category_id TINYINT UNSIGNED NOT NULL AUTO_INCREMENT,
-> name VARCHAR(25) NOT NULL,
-> last_update TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP,
-> PRIMARY KEY (category_id)
->) ENGINE=InnoDB DEFAULT CHARSET=utf8;
Query OK, 0 rows affected, 1 warning (0,04 sec)
mysql> CREATE TABLE film_category (
-> film_id SMALLINT UNSIGNED NOT NULL,
-> category_id TINYINT UNSIGNED NOT NULL,
-> last_update TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP,
-> PRIMARY KEY (film_id, category_id),
-> CONSTRAINT fk_film_category_film FOREIGN KEY (film_id) REFERENCES film (film_id)
-> ON DELETE RESTRICT ON UPDATE CASCADE,
-> CONSTRAINT fk_film_category_category FOREIGN KEY (category_id) REFERENCES category (category_id)
-> ON DELETE RESTRICT ON UPDATE CASCADE
->) ENGINE=InnoDB DEFAULT CHARSET=utf8;
Query OK, 0 rows affected, 1 warning (0,07 sec)
mysql> CREATE TABLE film_text (
-> film_id SMALLINT NOT NULL,
-> title VARCHAR(255) NOT NULL,
-> description TEXT,
-> PRIMARY KEY (film_id),
-> FULLTEXT KEY idx_title_description (title, description)
-> -- To build index on FULLTEXT to facilitate text search
-> -- FULLTEXT is supported in MyISAM engine, NOT in InnoDB engine
->) ENGINE=MyISAM DEFAULT CHARSET=utf8;
Query OK, 0 rows affected, 1 warning (0,02 sec)