

FINAL DRILLING M5 SR YUTI URZUA LEBUY

```
CREATE TABLE actor (  
  actor_id SERIAL PRIMARY KEY,  
  first_name VARCHAR(45) NOT NULL,  
  last_name VARCHAR(45) NOT NULL,  
  last_update TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP  
);  
  
CREATE TABLE film (  
  film_id SERIAL PRIMARY KEY,  
  title VARCHAR(255) NOT NULL,  
  description TEXT,  
  release_year INT,  
  language_id INT NOT NULL,  
  rental_duration INT NOT NULL,  
  rental_rate DECIMAL(4, 2) NOT NULL,  
  length INT,  
  replacement_cost DECIMAL(5, 2) NOT NULL,  
  rating VARCHAR(10),  
  last_update TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP,  
  special_features TEXT,  
  fulltext TSVECTOR  
);  
  
CREATE TABLE language (  
  language_id SERIAL PRIMARY KEY,  
  name VARCHAR(20) NOT NULL,  
  last_update TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP  
);  
  
CREATE TABLE category (  
  category_id SERIAL PRIMARY KEY,  
  name VARCHAR(25) NOT NULL,  
  last_update TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP  
);  
  
CREATE TABLE film_actor (  
  actor_id INT NOT NULL,  
  film_id INT NOT NULL,  
  last_update TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP,  
  PRIMARY KEY (actor_id, film_id),  
  FOREIGN KEY (actor_id) REFERENCES actor (actor_id),  
  FOREIGN KEY (film_id) REFERENCES film (film_id)
```

```

);

CREATE TABLE film_category (
    film_id INT NOT NULL,
    category_id INT NOT NULL,
    last_update TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP,
    PRIMARY KEY (film_id, category_id),
    FOREIGN KEY (film_id) REFERENCES film (film_id),
    FOREIGN KEY (category_id) REFERENCES category (category_id)
);

CREATE TABLE inventory (
    inventory_id SERIAL PRIMARY KEY,
    film_id INT NOT NULL,
    store_id INT NOT NULL,
    last_update TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (film_id) REFERENCES film (film_id)
);

CREATE TABLE customer (
    customer_id SERIAL PRIMARY KEY,
    store_id INT NOT NULL,
    first_name VARCHAR(45) NOT NULL,
    last_name VARCHAR(45) NOT NULL,
    email VARCHAR(50),
    address_id INT NOT NULL,
    activebool BOOLEAN NOT NULL DEFAULT TRUE,
    create_date DATE NOT NULL,
    last_update TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP,
    active INT
);

CREATE TABLE rental (
    rental_id SERIAL PRIMARY KEY,
    rental_date TIMESTAMP NOT NULL,
    inventory_id INT NOT NULL,
    customer_id INT NOT NULL,
    return_date TIMESTAMP,
    staff_id INT NOT NULL,
    last_update TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (inventory_id) REFERENCES inventory (inventory_id),
    FOREIGN KEY (customer_id) REFERENCES customer (customer_id),
    FOREIGN KEY (staff_id) REFERENCES staff (staff_id)
);

```

```
CREATE TABLE payment (
    payment_id SERIAL PRIMARY KEY,
    customer_id INT NOT NULL,
    staff_id INT NOT NULL,
    rental_id INT NOT NULL,
    amount DECIMAL(5, 2) NOT NULL,
    payment_date TIMESTAMP NOT NULL,
    FOREIGN KEY (customer_id) REFERENCES customer (customer_id),
    FOREIGN KEY (staff_id) REFERENCES staff (staff_id),
    FOREIGN KEY (rental_id) REFERENCES rental (rental_id)
);
```

```
CREATE TABLE staff (
    staff_id SERIAL PRIMARY KEY,
    first_name VARCHAR(45) NOT NULL,
    last_name VARCHAR(45) NOT NULL,
    address_id INT NOT NULL,
    email VARCHAR(50),
    store_id INT NOT NULL,
    active BOOLEAN NOT NULL DEFAULT TRUE,
    username VARCHAR(16) NOT NULL,
    password VARCHAR(40),
    last_update TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP,
    picture BYTEA
);
```

```
CREATE TABLE address (
    address_id SERIAL PRIMARY KEY,
    address VARCHAR(50) NOT NULL,
    address2 VARCHAR(50),
    district VARCHAR(20) NOT NULL,
    city_id INT NOT NULL,
    postal_code VARCHAR(10),
    phone VARCHAR(20) NOT NULL,
    last_update TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (city_id) REFERENCES city (city_id)
);
```

```
CREATE TABLE city (
    city_id SERIAL PRIMARY KEY,
    city VARCHAR(50) NOT NULL,
    country_id INT NOT NULL,
    last_update TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (country_id) REFERENCES country (country_id)
);
```

```

CREATE TABLE country (
    country_id SERIAL PRIMARY KEY,
    country VARCHAR(50) NOT NULL,
    last_update TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP
);

CREATE TABLE store (
    store_id SERIAL PRIMARY KEY,
    manager_staff_id INT NOT NULL,
    address_id INT NOT NULL,
    last_update TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (manager_staff_id) REFERENCES staff (staff_id),
    FOREIGN KEY (address_id) REFERENCES address (address_id)
);

--Construye las siguientes consultas:
--• Aquellas usadas para insertar, modificar y eliminar un Customer, Staff -
-y Actor.

--insertar Customer

INSERT INTO customer (store_id, first_name, last_name, email, address_id,
activebool, create_date, last_update, active)
VALUES (1, 'John', 'Doe', 'johndoe@example.com', 1, TRUE, CURRENT_DATE,
CURRENT_TIMESTAMP, 1);

--modificar Customer

UPDATE customer
SET email = 'john.doe.new@example.com', last_update = CURRENT_TIMESTAMP
WHERE customer_id = 1;

--eliminar Customer

DELETE FROM customer
WHERE customer_id = 1;

--insertar staff

INSERT INTO staff (first_name, last_name, address_id, email, store_id,
active, username, password, last_update)

```

```
VALUES ('Jane', 'Smith', 1, 'janesmith@example.com', 1, TRUE, 'janes',
'password123', CURRENT_TIMESTAMP);

-- modificar staff

UPDATE staff
SET email = 'jane.smith.new@example.com', last_update = CURRENT_TIMESTAMP
WHERE staff_id = 1;

--eliminar staff

DELETE FROM staff
WHERE staff_id = 1;

--insertar actor

INSERT INTO actor (first_name, last_name, last_update)
VALUES ('Robert', 'Downey', CURRENT_TIMESTAMP);

--modificar actor

UPDATE actor
SET last_name = 'Downey Jr.', last_update = CURRENT_TIMESTAMP
WHERE actor_id = 1;

--eliminar actor

DELETE FROM actor
WHERE actor_id = 1;

--• Listar todas las “rental” con los datos del “customer” dado un año y --
mes.

SELECT
    r.rental_id,
    r.rental_date,
    c.first_name,
    c.last_name,
    c.email
```

```
FROM
    rental r
JOIN
    customer c ON r.customer_id = c.customer_id
WHERE
    EXTRACT(YEAR FROM r.rental_date) = 2023
    AND EXTRACT(MONTH FROM r.rental_date) = 11;
```

--• Listar Número, Fecha (payment_date) y Total (amount) de todas las ---
"payment".

```
SELECT
    p.payment_id AS Numero,
    p.payment_date AS Fecha,
    p.amount AS Total
FROM
    payment p;
```

--• Listar todas las "film" del año 2006 que contengan un (rental_rate) --
mayor a 4.0.

```
SELECT
    f.title AS Pelicula,
    f.rental_rate AS TasaDeAlquiler
FROM
    film f
WHERE
    f.release_year = 2006
    AND f.rental_rate > 4.0;
```