IFCD 0210. DESARROLLO DE APLICACIONES CON TECNOLOGÍAS WEB.

MF 0492_3. SERVIDOR

UF 1845_3. ACCESO A DATOS EN APLICACIONES WEB DEL ENTONO SERVIDOR.

…….

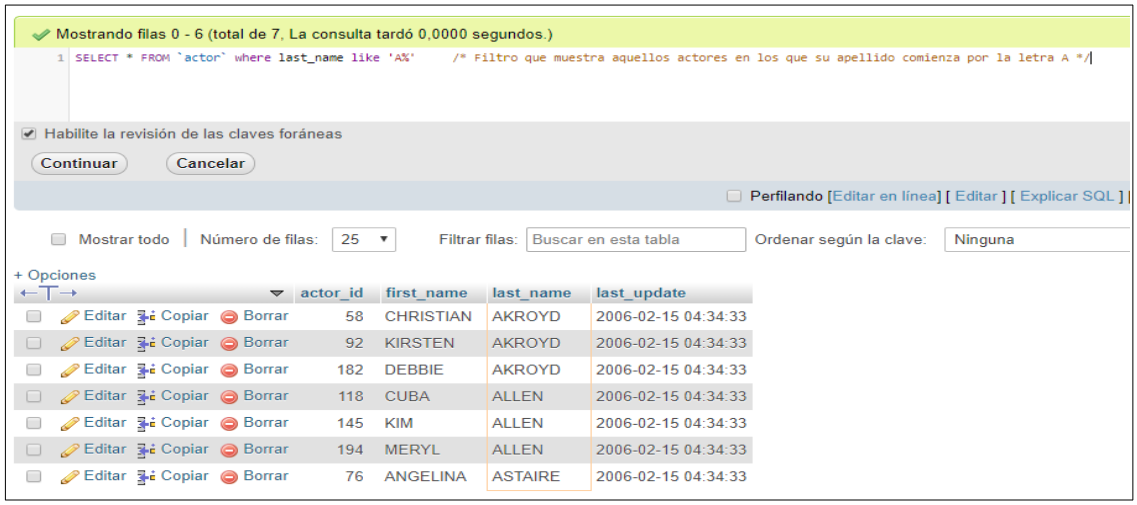
- Examen Práctico -

Angel David Morales Expósito

ACTIVIDAD 2.
Realizar cinco consultas SQL de libre elección.

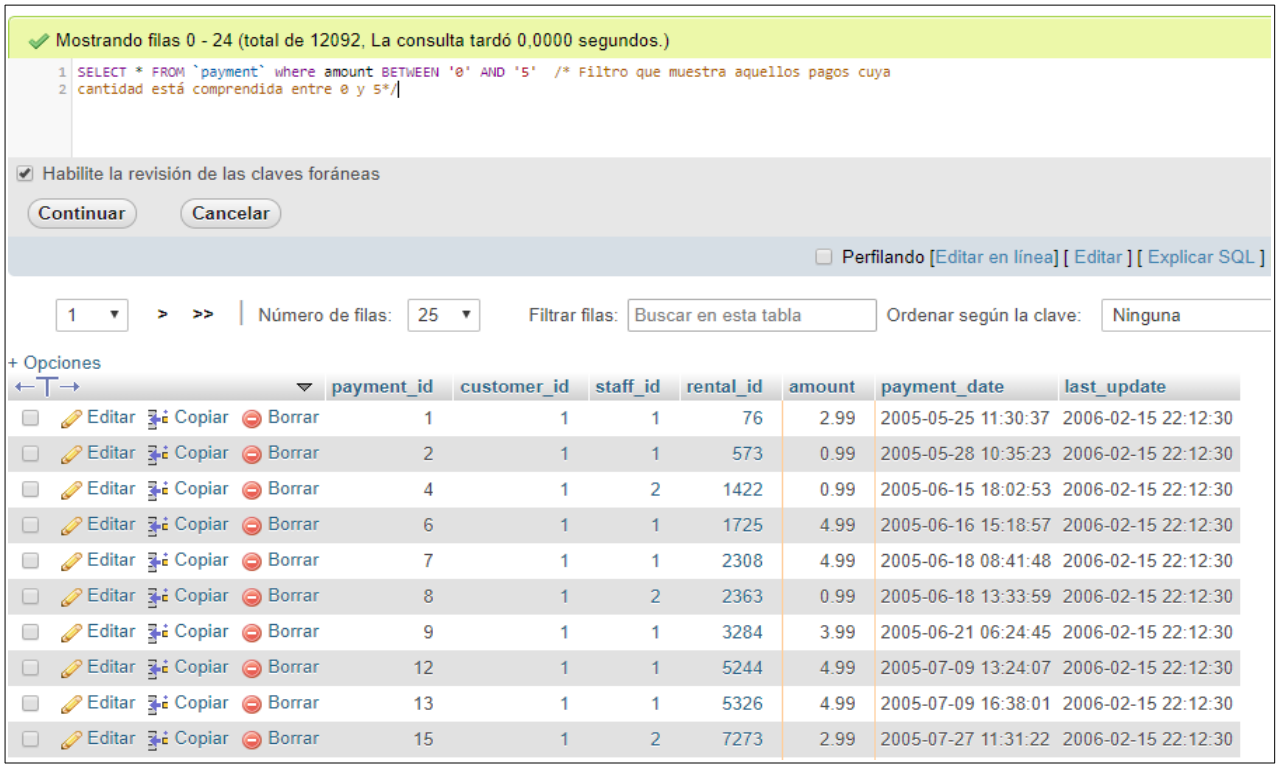
Consulta que muestra aquellos actores en los que su apellido comienza por la letra A.

SELECT * FROM ‘actor’ where last_name like ‘A%’



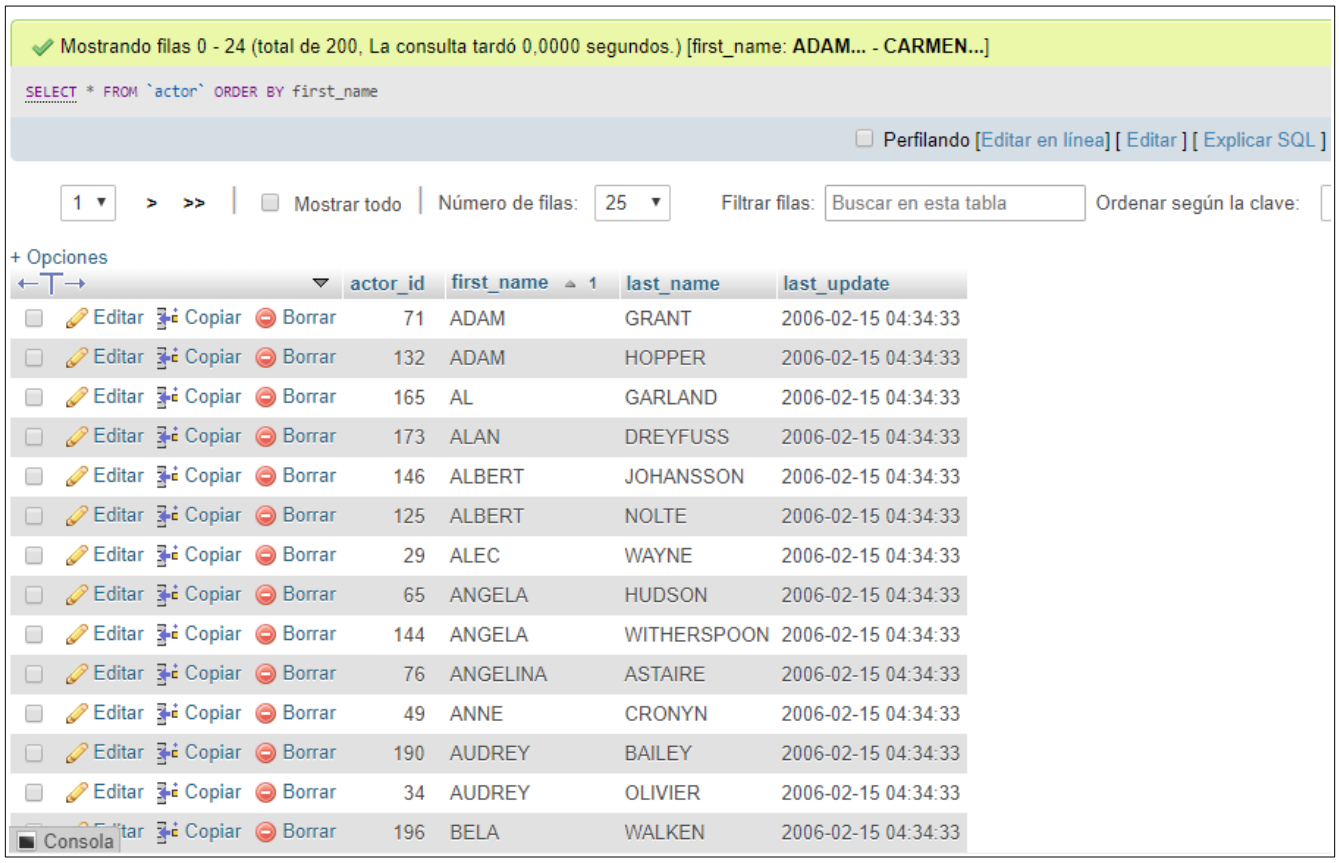
Consulta que muestra aquellos pagos cuya cantidad está comprendida entre 0 y 5.

SELECT * FROM ‘payment’ where amount BETWEEN ‘0’ AND ‘5’



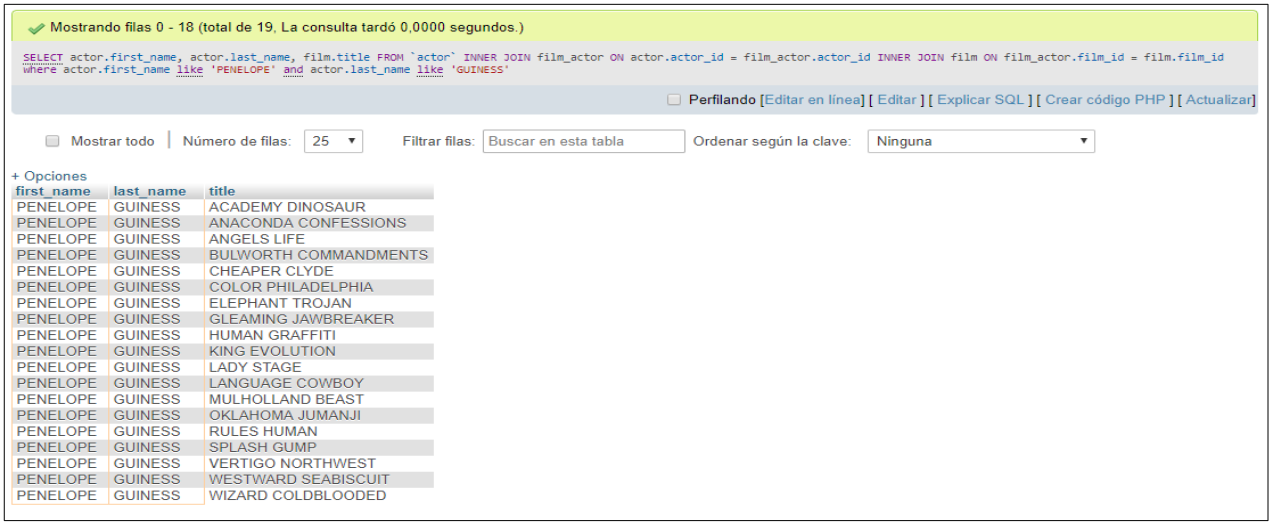
Ordena todos los actores por orden alfabético según su nombre.

SELECT * FROM ‘actor’ ORDER BY first_name



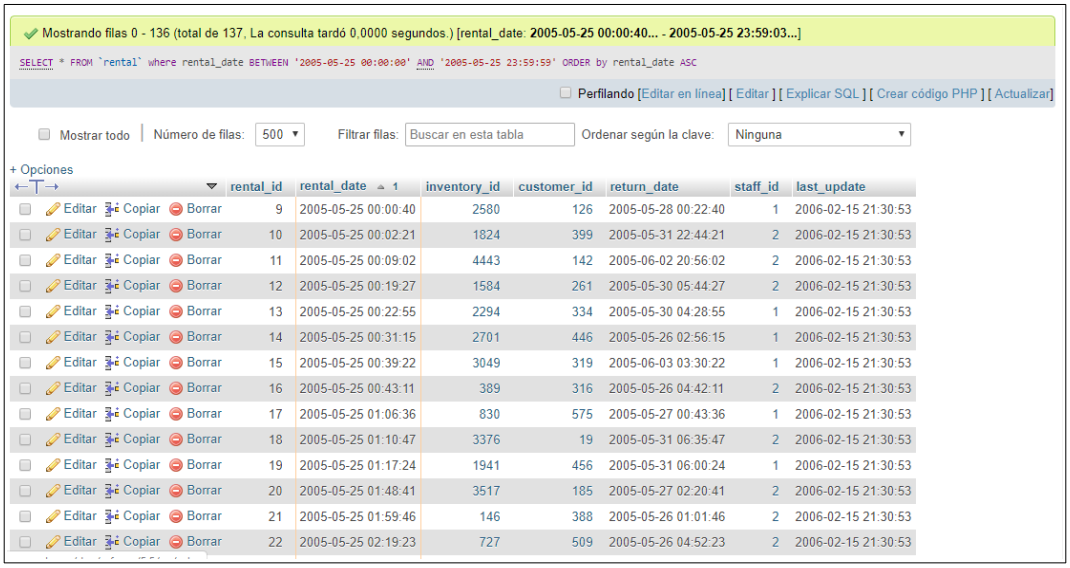
Consulta en la que se muestran todas las películas realizadas por la actriz PENELOPE GUINESS.

SELECT actor.first_name, actor.last_name, film.title FROM `actor`
INNER JOIN film_actor
ON actor.actor_id = film_actor.actor_id
INNER JOIN film
ON film_actor.film_id = film.film_id
where actor.first_name like 'PENELOPE' and actor.last_name like 'GUINESS'



Consulta en la que se muestran todas las rentas realizadas el día 2005-05-25, ordenadas por orden cronológico ascendente.

SELECT * FROM `rental` where rental_date BETWEEN '2005-05-25 00:00:00' AND '2005-05-25 23:59:59' ORDER by rental_date ASC

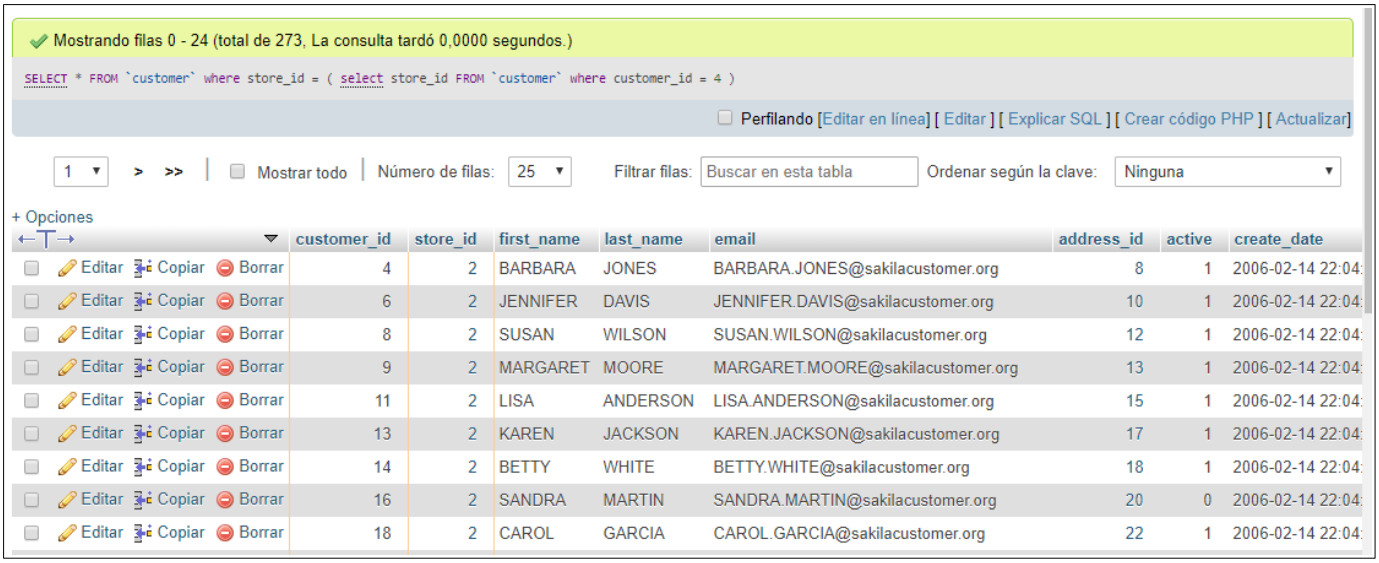


ACTIVIDAD 3.

***Realizar una subconsulta de libre eleccción en la que se genere al menos una vista* SQL**

*** Se muestran todos los registros que tengan un store_id igual que el que tiene el customer_id numero 4.**

```
SELECT * FROM `customer`
where store_id =
(
   select store_id FROM `customer`
   where customer_id = 4
   )
```



ACTIVIDAD 4

Implementar al menos dos triggers

*** Trigger que guarda la información de una dirección en otra tabla cuando se actualice la información de dicha dirección.**

```
CREATE TRIGGER `update_address` BEFORE UPDATE ON `address`
 FOR EACH ROW BEGIN
         IF (OLD.address_id != new.address_id ) OR (OLD.address != new.address ) OR (OLD.district !
= new.district ) OR (OLD.city_id != new.city_id ) OR (OLD.postal_code != new.postal_code ) OR
(OLD.phone != new.phone )
   THEN

      INSERT INTO old_address (
         old_address_id,
         address,
         district,
         city_id,
         postal_code,
         phone,
         change_date)

      VALUES (
         OLD.address_id,
         OLD.address,
         OLD.district,
         OLD.city_id,
         OLD.postal_code,
         OLD.phone,
         now()
         );

END IF;
END
```

*** Trigger que cuando se borra un registro de rental, comprueba que la película se ha entregado a tiempo, y de no ser así, se incluyen los datos del cliente en otra tabla llamada deuda.**

```
CREATE TRIGGER `add_deuda` BEFORE DELETE ON `rental`
 FOR EACH ROW BEGIN
          IF ( old.return_date < NOW() )
          THEN
       INSERT INTO deuda (
          rental_id,
          return_date,
          customer_id,
          first_name,
          last_name)
        VALUES (
           old.rental_id,
           old.return_date,
           old.customer_id,
           (SELECT first_name FROM `customer`
           where customer_id = old.customer_id),
           (SELECT last_name FROM `customer`
           where customer_id = old.customer_id)
           );
END IF;
END
```

ACTIVIDAD 5
IMPLEMENTAR AL MENOS UNA FUNCIÓN

***Función en la que se introduce el id de un usuario y devuelve la cantidad de dinero que ha gastado en pagos.**

```
DELIMITER $$
CREATE DEFINER=`root`@`localhost` FUNCTION `customer_payment_amount`(`usuario`
SMALLINT) RETURNS decimal(10,0)
   NO SQL
BEGIN
declare total decimal;

SET total = (SELECT SUM(amount)
FROM payment
WHERE customer_id = usuario);

return total;
END$$
DELIMITER ;
```

ACTIVIDAD 6
IMPLEMENTAR AL MENOS UN PROCEDIMIENTO

***Procedimiento que muestra todas las películas realizadas por la actriz Penélope Guinness (al igual que la consulta número 4).**

```
DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE `pe_pe`()
    NO SQL
SELECT actor.first_name, actor.last_name, film.title FROM `actor`
INNER JOIN film_actor
ON actor.actor_id = film_actor.actor_id
INNER JOIN film
ON film_actor.film_id = film.film_id
where actor.first_name like 'PENELOPE' and actor.last_name like 'GUINESS'$$
DELIMITER ;
```