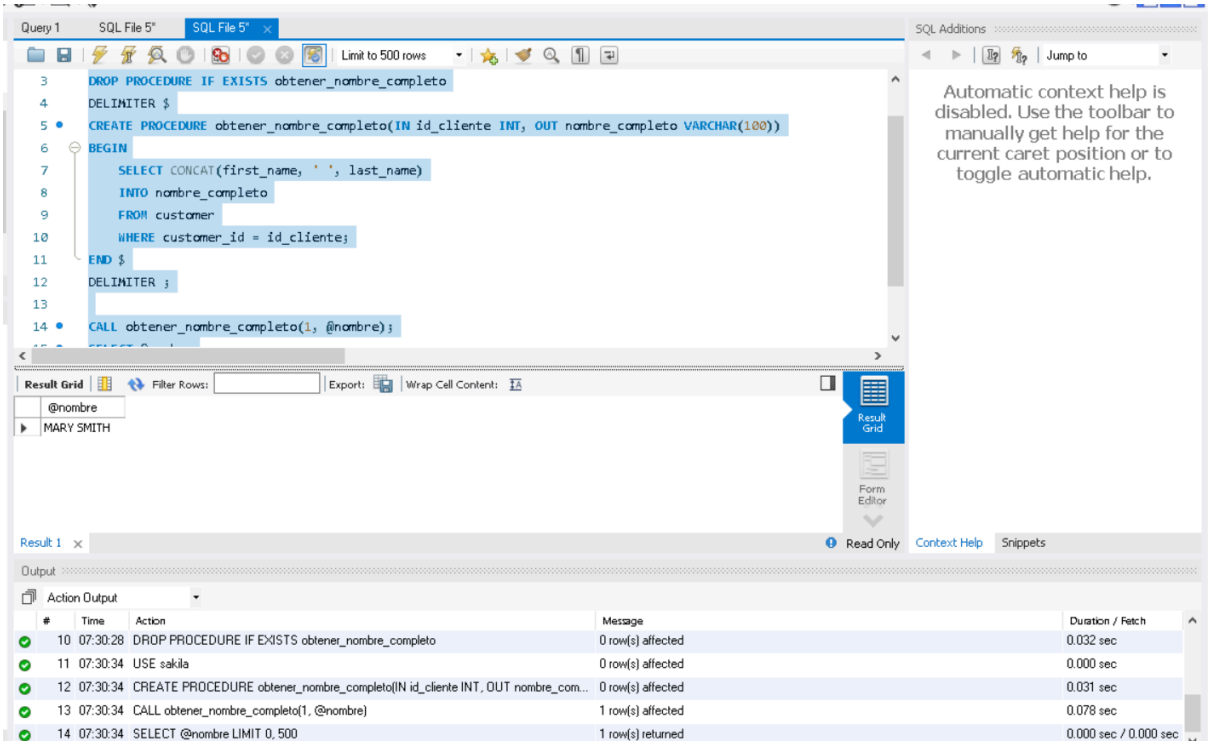
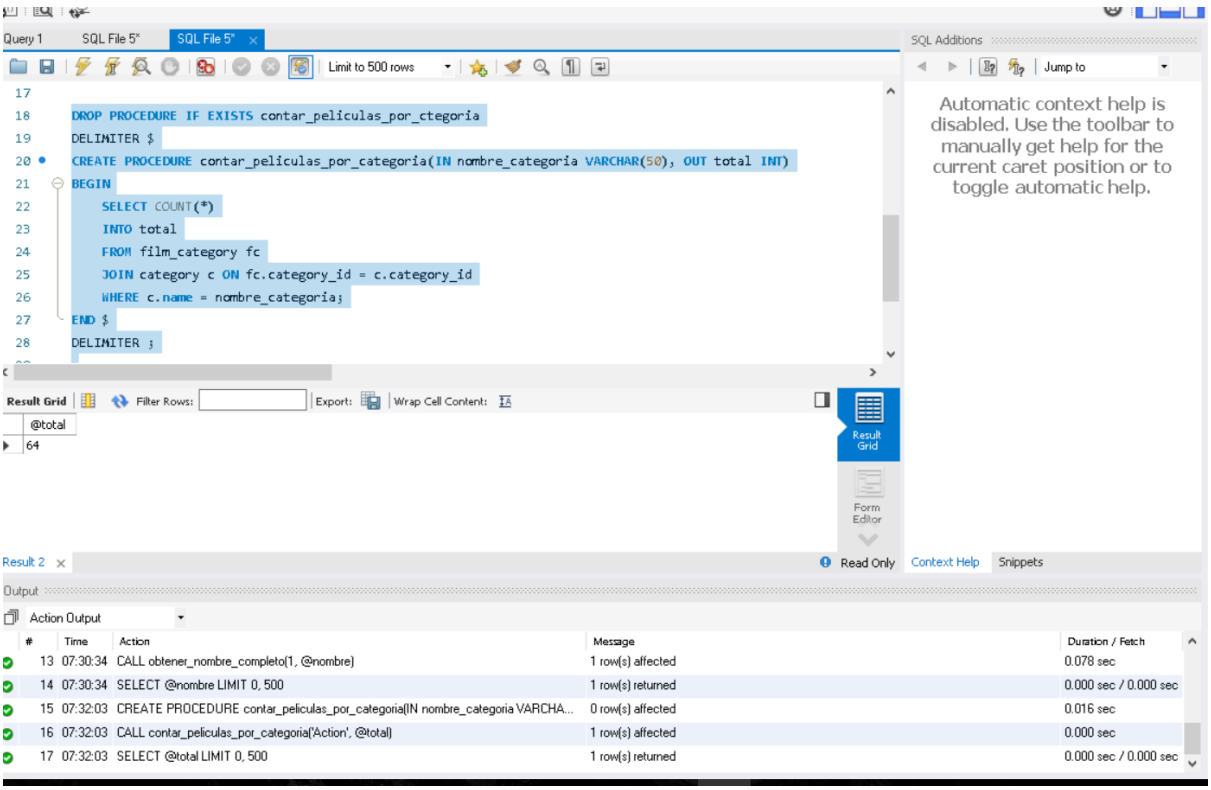
Actividad 28

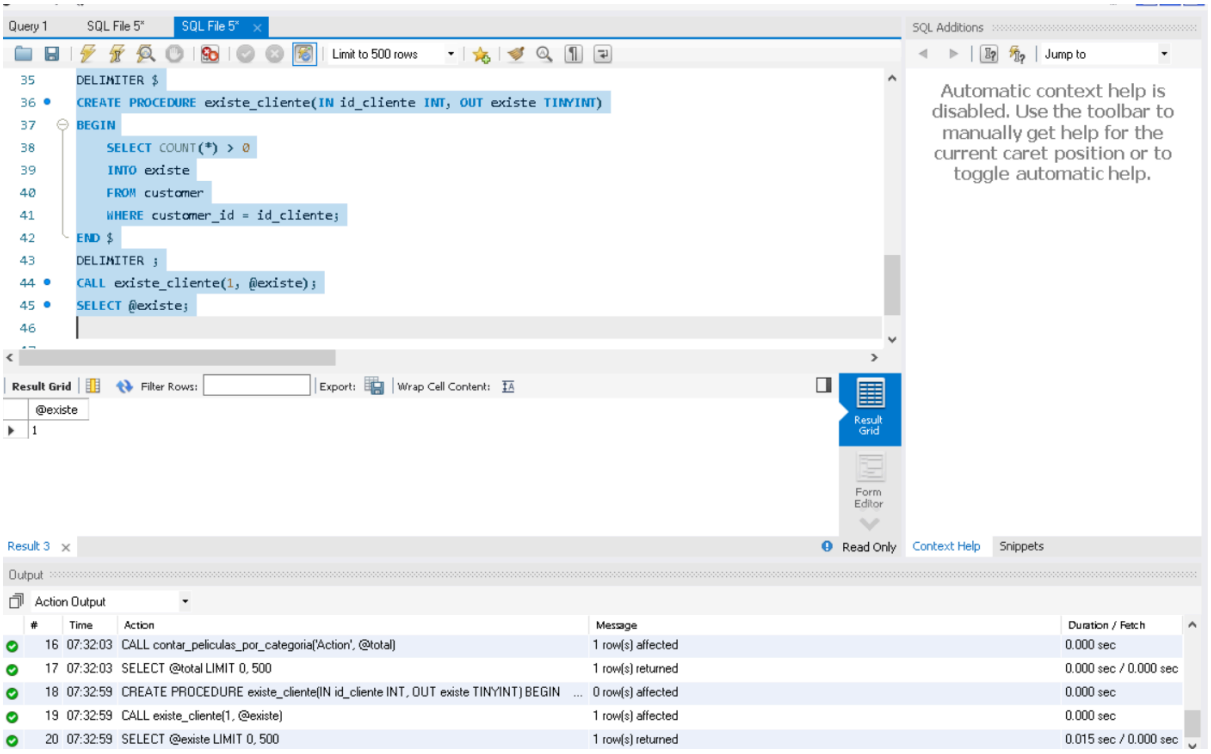
Crea un procedimiento que reciba el customer_id y devuelva el nombre completo (first_name + last_name) como salida.



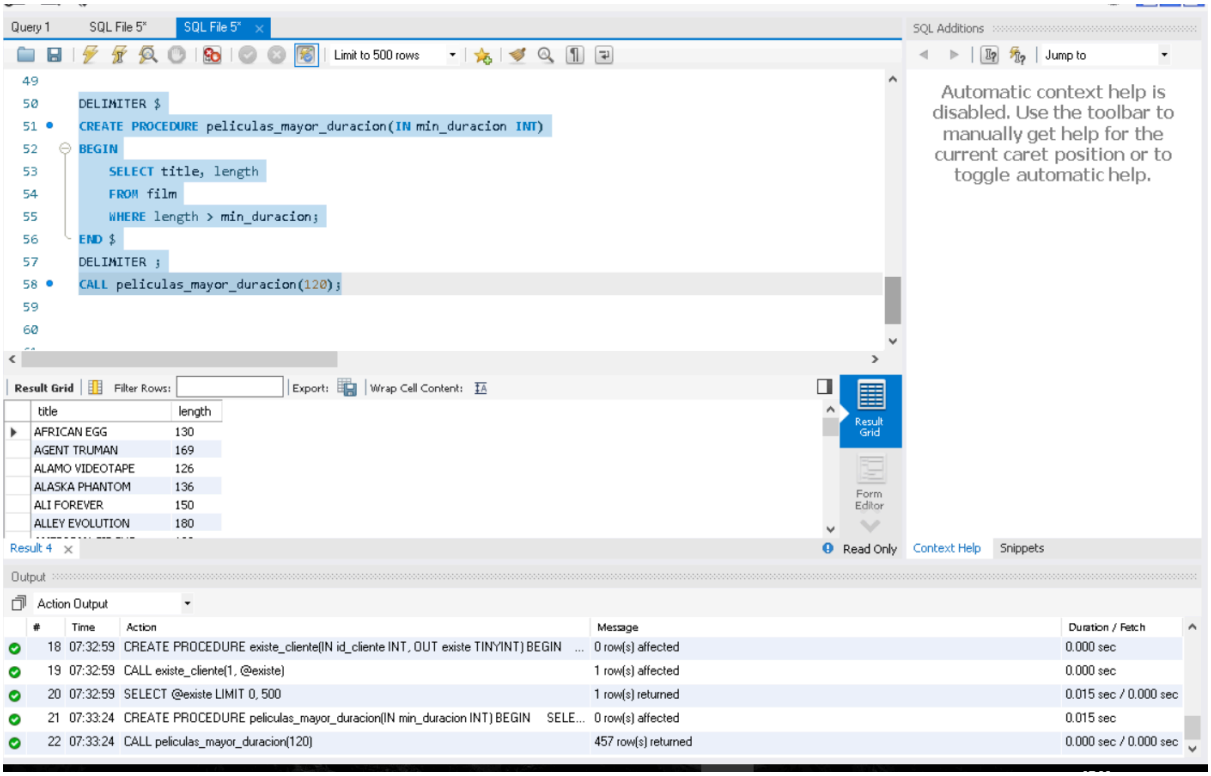
Crear un procedimiento que reciba el nombre de una categoría (por ejemplo, 'Action') y devuelva cuántas películas hay en esa categoría.



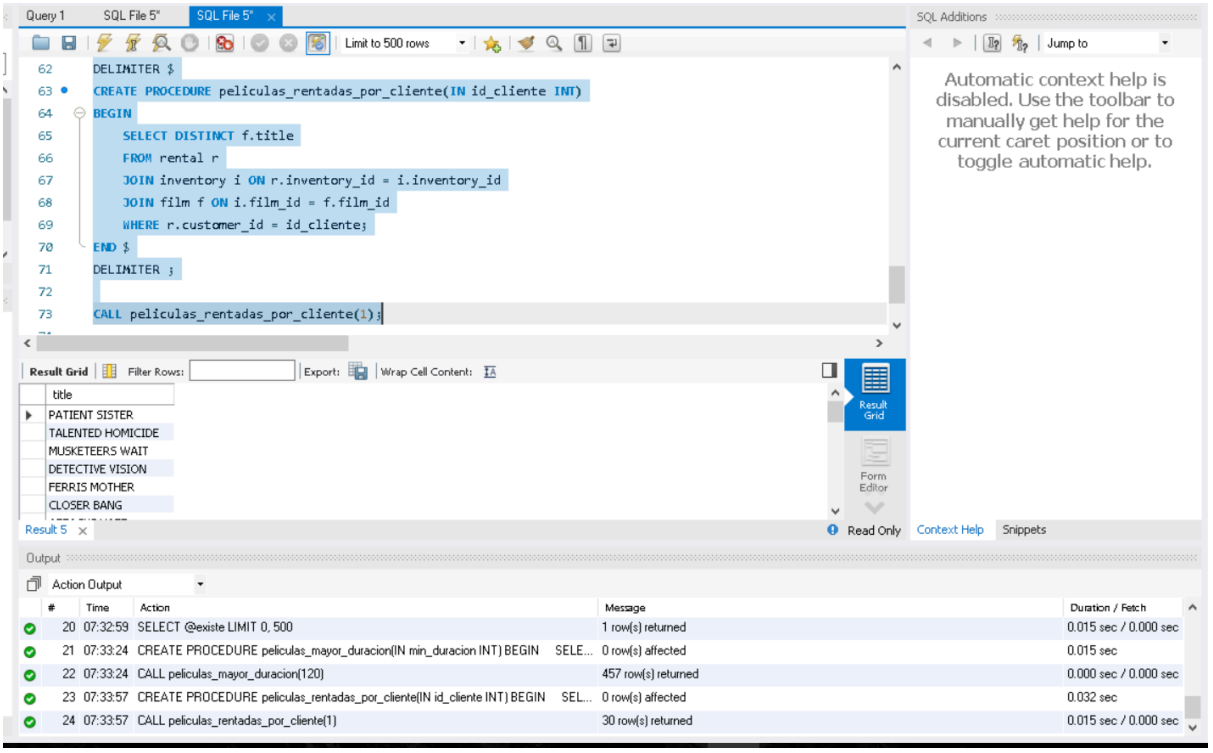
Crea un procedimiento que reciba un customer_id y devuelva 1 si existe, o 0 si no.



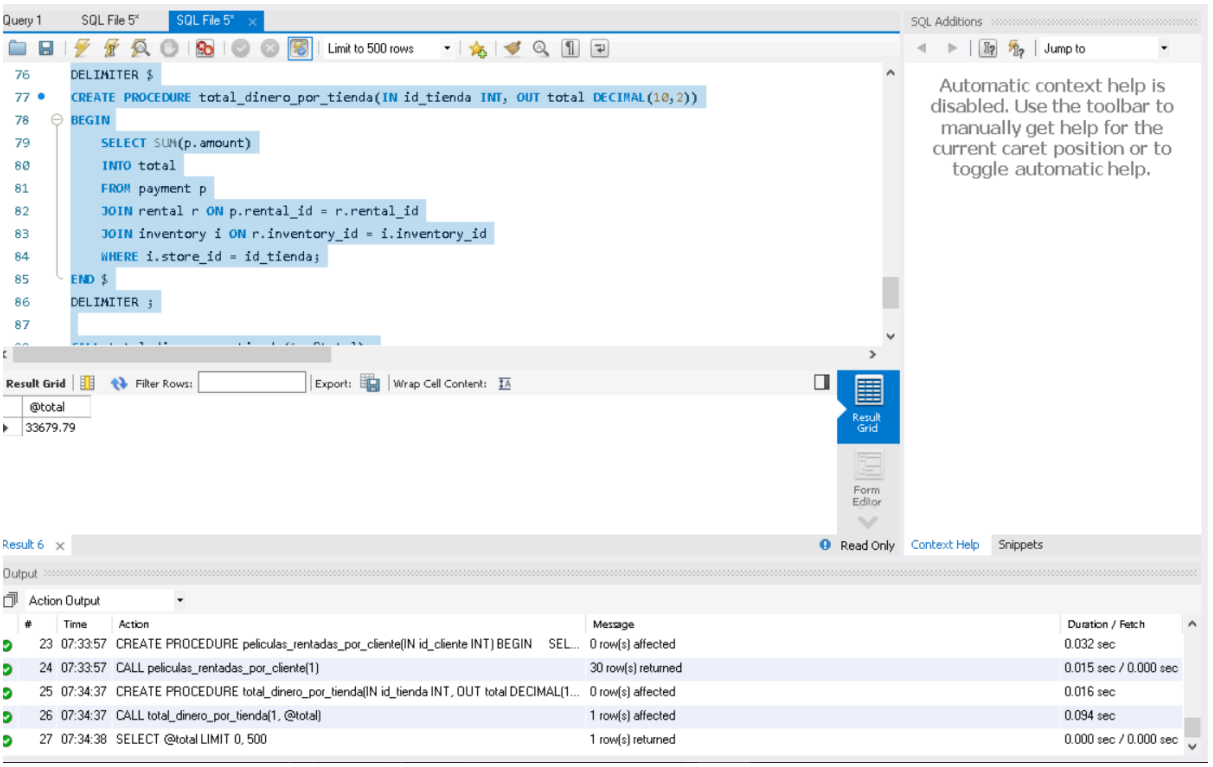
Crear un procedimiento que reciba un número (min_duracion) y muestre todas las películas que duran más que ese valor.



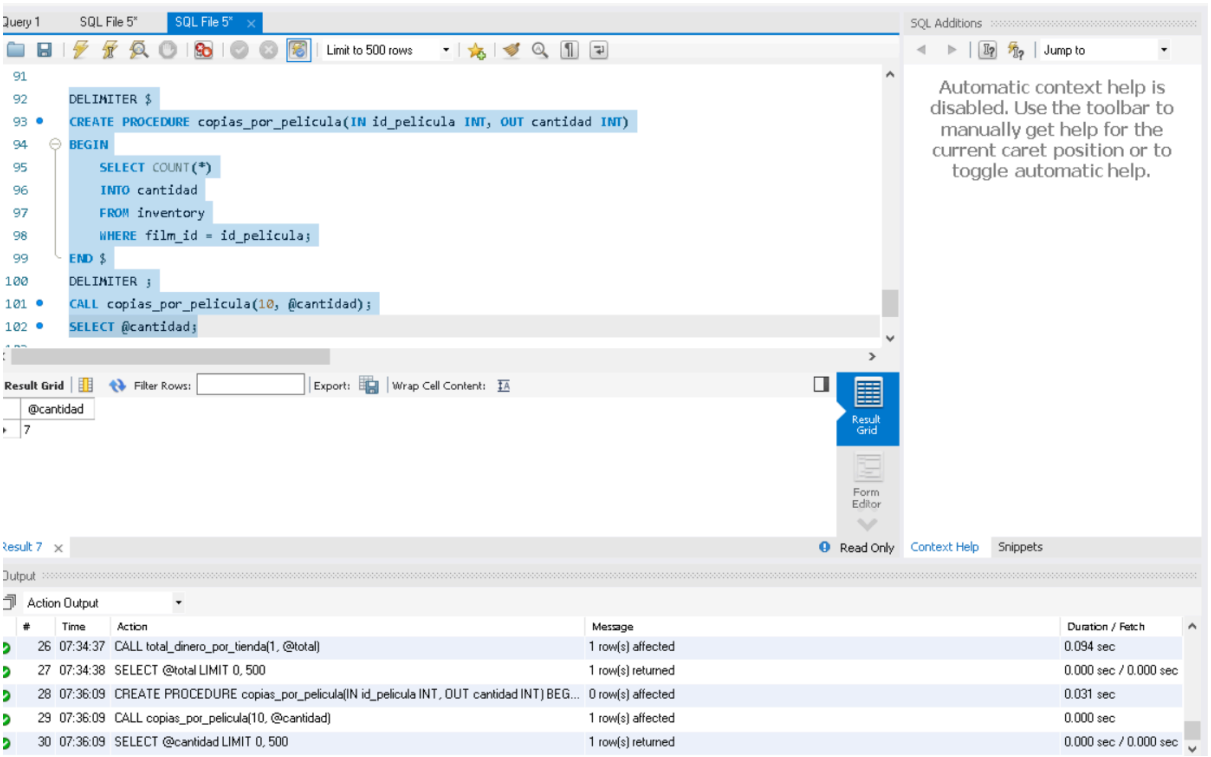
Crea un procedimiento que reciba el customer_id y devuelva una lista de títulos de películas que ha rentado.



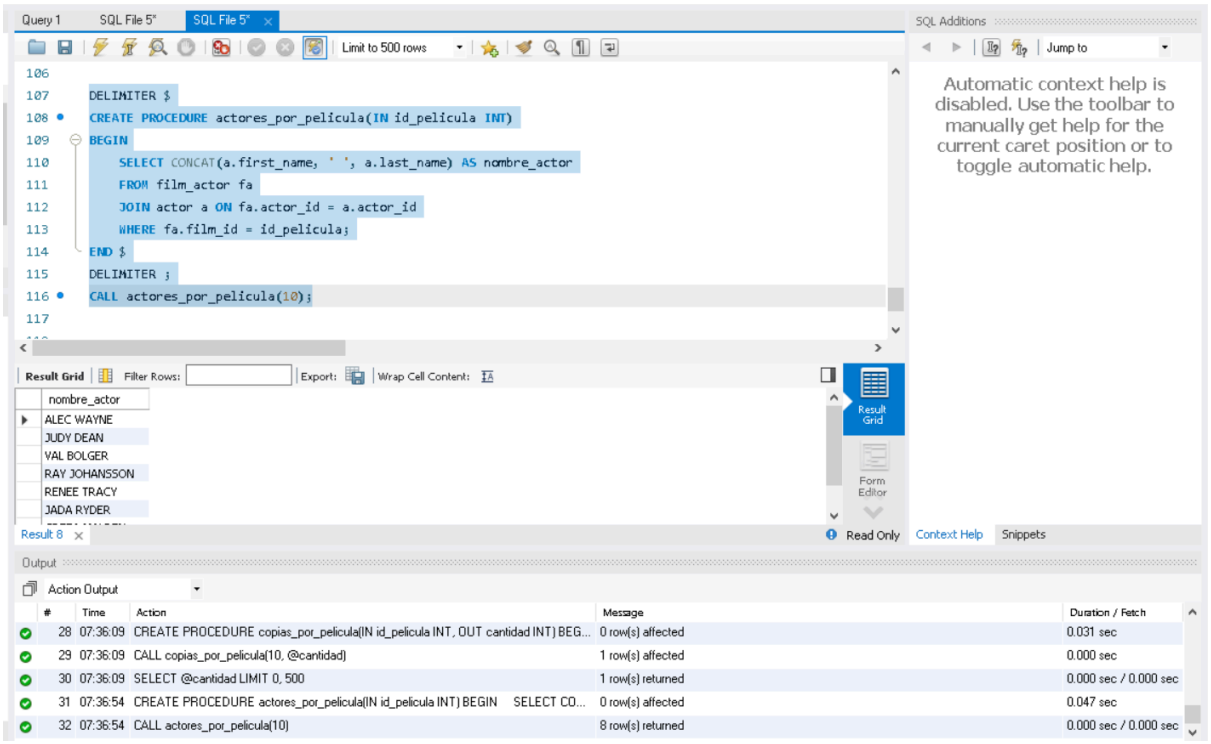
Crear un procedimiento que reciba el store_id y devuelva el total de dinero generado por esa tienda.



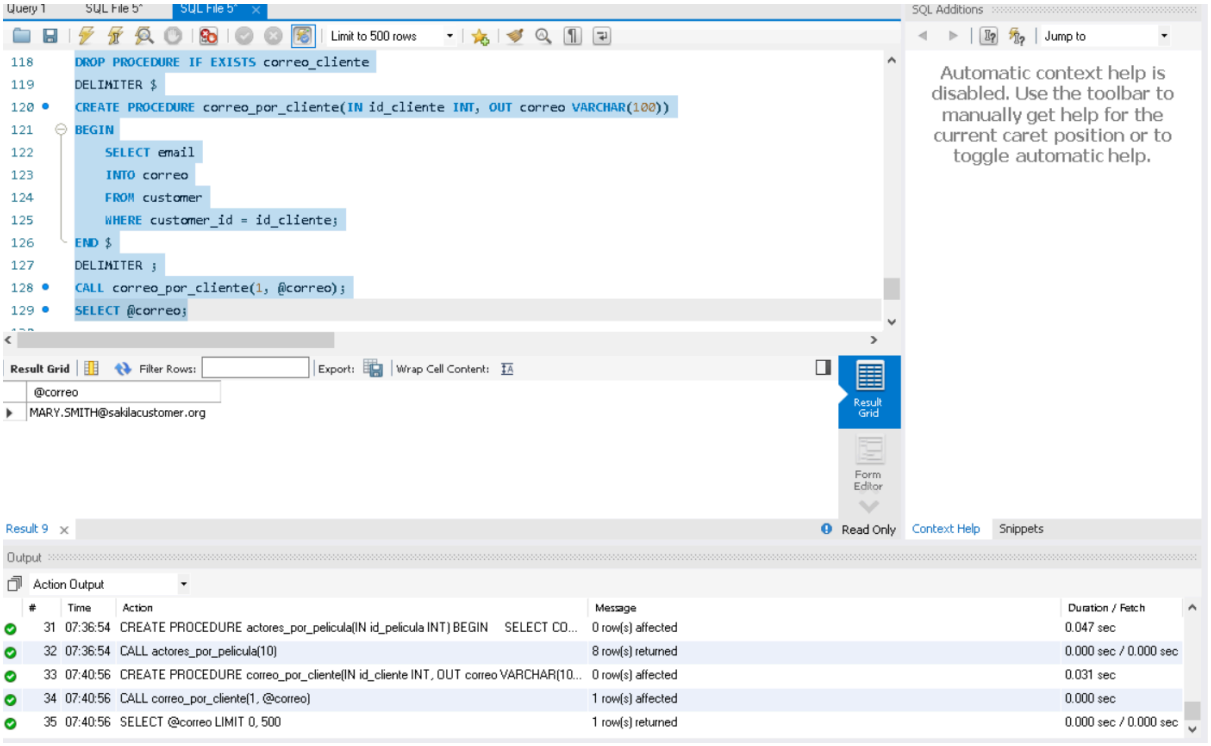
Crear un procedimiento que reciba el film_id y devuelva el número de copias disponibles en la tabla inventory.



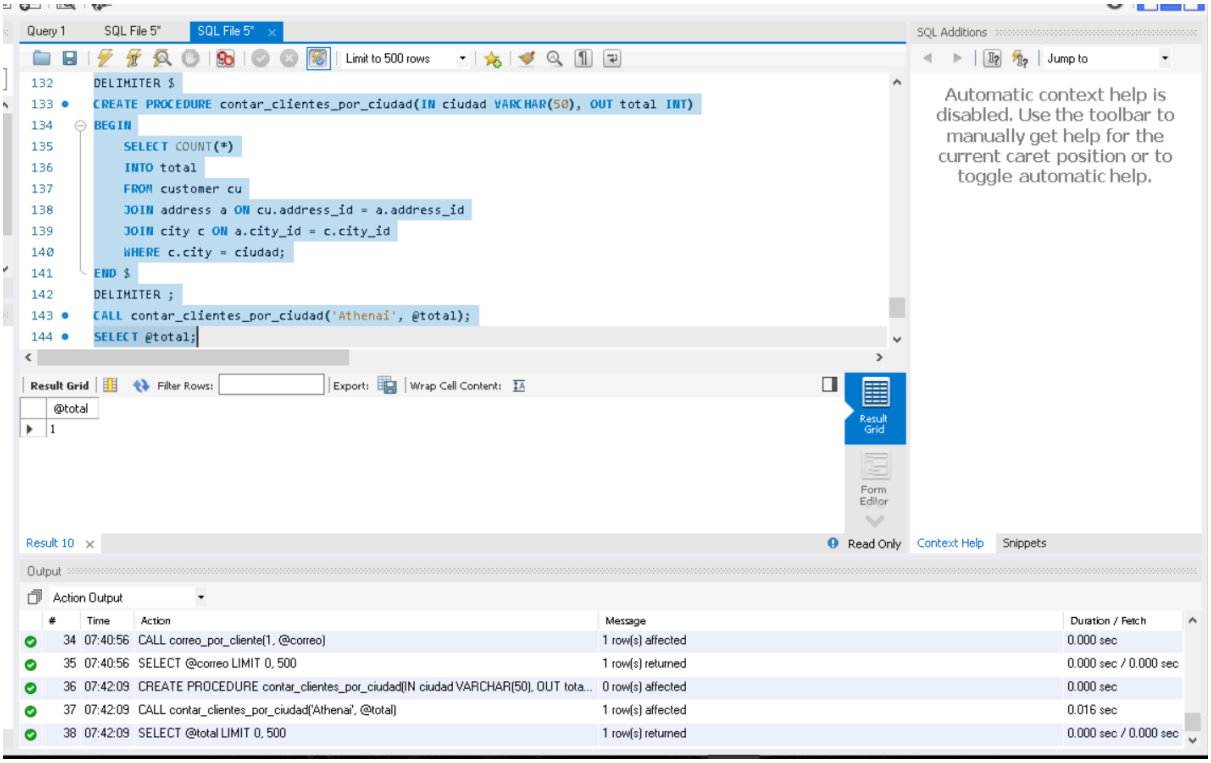
Crear un procedimiento que reciba el film_id y muestre una lista con los nombres de los actores que participan en esa película.



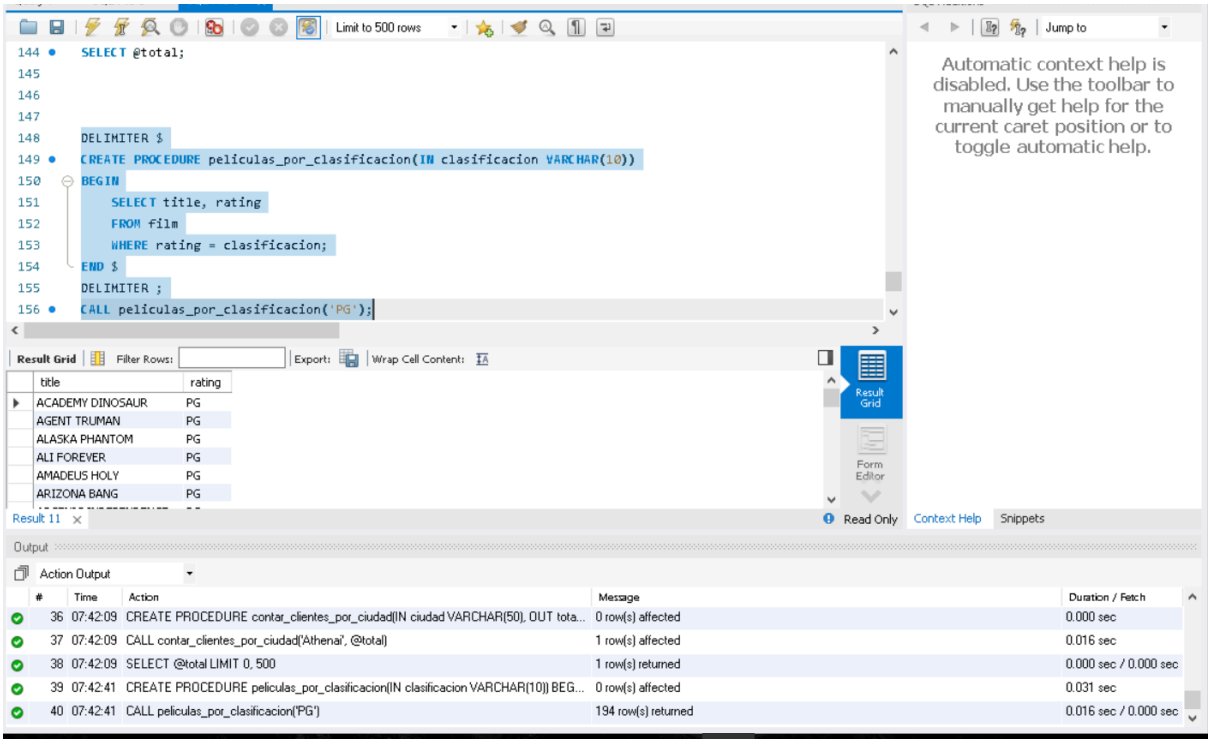
Recibir el customer_id y devolver el correo electrónico del cliente.



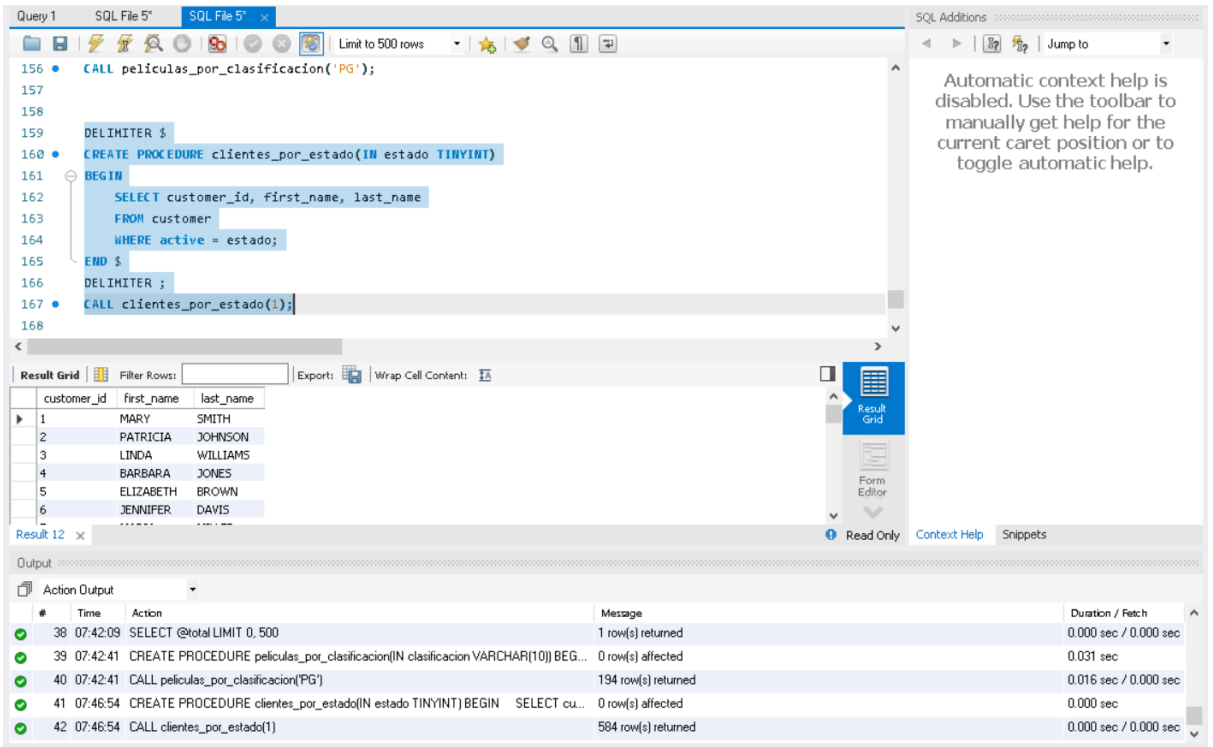
Recibir el nombre de una ciudad y devolver cuántos clientes viven ahí.



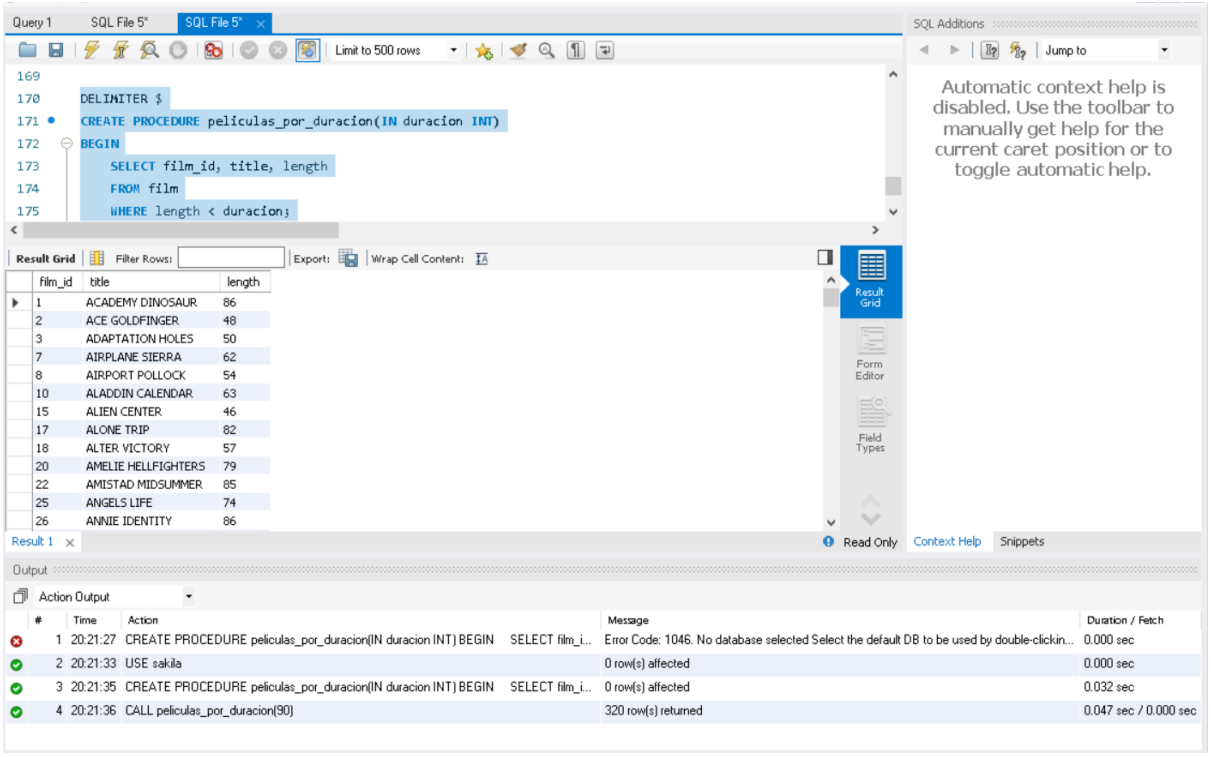
Recibir la clasificación (rating, como 'PG' o 'R') y mostrar las películas con esa clasificación.



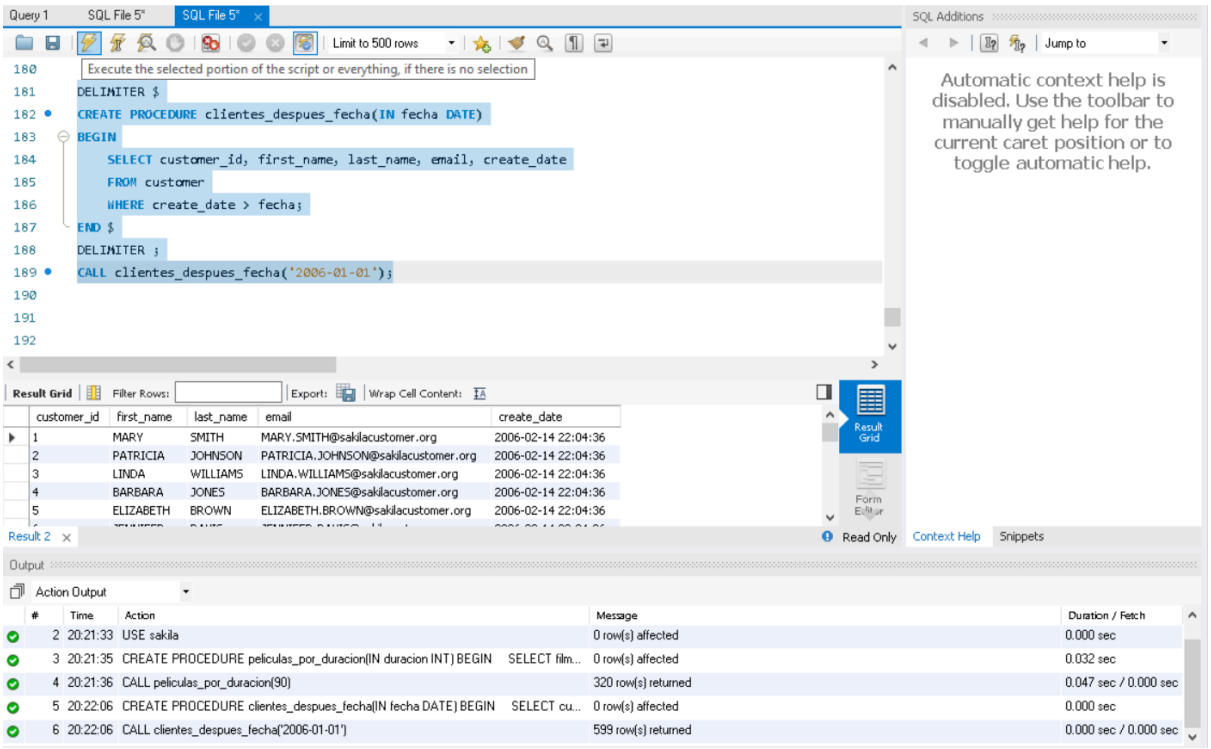
Recibir el estado (activo o inactivo) y mostrar la lista de clientes.



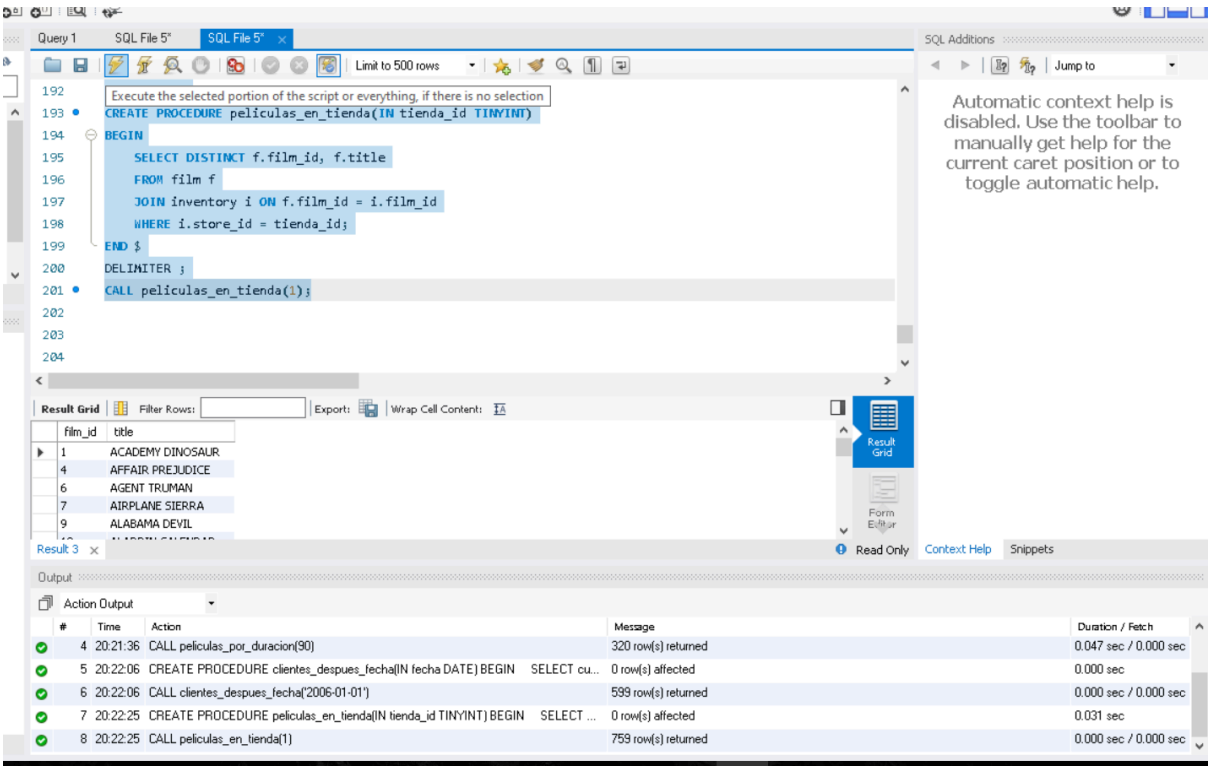
Recibir un número y mostrar las películas con duración menor a ese valor.



Recibir una fecha y listar los clientes registrados después de esa fecha.



Recibir un store_id y mostrar todas las películas disponibles en esa tienda.



TODO EL CODIGO:

```
USE sakila;


DROP PROCEDURE IF EXISTS obtener_nombre_completo

DELIMITER $

CREATE PROCEDURE obtener_nombre_completo(IN id_cliente INT, OUT
nombre_completo VARCHAR(100))

BEGIN

   SELECT CONCAT(first_name, ' ', last_name)

   INTO nombre_completo

   FROM customer

   WHERE customer_id = id_cliente;

END $

DELIMITER ;


CALL obtener_nombre_completo(1, @nombre);

SELECT @nombre;
```

```
DROP PROCEDURE IF EXISTS contar_peliculas_por_ctegoria

DELIMITER $

CREATE PROCEDURE contar_peliculas_por_categoria(IN nombre_categoria
VARCHAR(50), OUT total INT)

BEGIN

   SELECT COUNT(*)

   INTO total

   FROM film_category fc

   JOIN category c ON fc.category_id = c.category_id

   WHERE c.name = nombre_categoria;

END $

DELIMITER ;


CALL contar_peliculas_por_categoria('Action', @total);

SELECT @total;



DELETE PROCEDURE IF EXISTS existe_cliente

DELIMITER $

CREATE PROCEDURE existe_cliente(IN id_cliente INT, OUT existe TINYINT)

BEGIN

   SELECT COUNT(*) > 0

   INTO existe

   FROM customer

   WHERE customer_id = id_cliente;

END $

DELIMITER ;

CALL existe_cliente(1, @existe);

SELECT @existe;







DELIMITER $

CREATE PROCEDURE peliculas_mayor_duracion(IN min_duracion INT)
```

```
BEGIN
    SELECT title, length
    FROM film
    WHERE length > min_duracion;
END $
DELIMITER ;
CALL peliculas_mayor_duracion(120);



DELIMITER $
CREATE PROCEDURE peliculas_rentadas_por_cliente(IN id_cliente INT)
BEGIN
    SELECT DISTINCT f.title
    FROM rental r
    JOIN inventory i ON r.inventory_id = i.inventory_id
    JOIN film f ON i.film_id = f.film_id
    WHERE r.customer_id = id_cliente;
END $
DELIMITER ;

CALL peliculas_rentadas_por_cliente(1);



DELIMITER $
CREATE PROCEDURE total_dinero_por_tienda(IN id_tienda INT, OUT total
DECIMAL(10,2))
BEGIN
    SELECT SUM(p.amount)
    INTO total
    FROM payment p
    JOIN rental r ON p.rental_id = r.rental_id
    JOIN inventory i ON r.inventory_id = i.inventory_id
    WHERE i.store_id = id_tienda;
END $
```

```
DELIMITER ;

CALL total_dinero_por_tienda(1, @total);
SELECT @total;



DELIMITER $
CREATE PROCEDURE copias_por_pelicula(IN id_pelicula INT, OUT cantidad INT)
BEGIN
   SELECT COUNT(*)
   INTO cantidad
   FROM inventory
   WHERE film_id = id_pelicula;
END $
DELIMITER ;
CALL copias_por_pelicula(10, @cantidad);
SELECT @cantidad;




DELIMITER $
CREATE PROCEDURE actores_por_pelicula(IN id_pelicula INT)
BEGIN
   SELECT CONCAT(a.first_name, ' ', a.last_name) AS nombre_actor
   FROM film_actor fa
   JOIN actor a ON fa.actor_id = a.actor_id
   WHERE fa.film_id = id_pelicula;
END $
DELIMITER ;
CALL actores_por_pelicula(10);

DROP PROCEDURE IF EXISTS correo_cliente
DELIMITER $
```

```
CREATE PROCEDURE correo_por_cliente(IN id_cliente INT, OUT correo VARCHAR(100))
BEGIN
   SELECT email
   INTO correo
   FROM customer
   WHERE customer_id = id_cliente;
END $
DELIMITER ;
CALL correo_por_cliente(1, @correo);
SELECT @correo;



DELIMITER $
CREATE PROCEDURE contar_clientes_por_ciudad(IN ciudad VARCHAR(50), OUT total
INT)
BEGIN
   SELECT COUNT(*)
   INTO total
   FROM customer cu
   JOIN address a ON cu.address_id = a.address_id
   JOIN city c ON a.city_id = c.city_id
   WHERE c.city = ciudad;
END $
DELIMITER ;
CALL contar_clientes_por_ciudad('Athenai', @total);
SELECT @total;



DELIMITER $
CREATE PROCEDURE peliculas_por_clasificacion(IN clasificacion VARCHAR(10))
BEGIN
   SELECT title, rating
   FROM film
   WHERE rating = clasificacion;
```

```
END $
DELIMITER ;
CALL peliculas_por_clasificacion('PG');



DELIMITER $
CREATE PROCEDURE clientes_por_estado(IN estado TINYINT)
BEGIN
   SELECT customer_id, first_name, last_name
   FROM customer
   WHERE active = estado;
END $
DELIMITER ;
CALL clientes_por_estado(1);



DELIMITER $
CREATE PROCEDURE peliculas_por_duracion(IN duracion INT)
BEGIN
   SELECT film_id, title, length
   FROM film
   WHERE length < duracion;
END $
DELIMITER ;
CALL peliculas_por_duracion(90);



DELIMITER $
CREATE PROCEDURE clientes_despues_fecha(IN fecha DATE)
BEGIN
   SELECT customer_id, first_name, last_name, email, create_date
   FROM customer
   WHERE create_date > fecha;
END $
```

```
DELIMITER ;
CALL clientes_despues_fecha('2006-01-01');



DELIMITER $
CREATE PROCEDURE peliculas_en_tienda(IN tienda_id TINYINT)
BEGIN
   SELECT DISTINCT f.film_id, f.title
   FROM film f
   JOIN inventory i ON f.film_id = i.film_id
   WHERE i.store_id = tienda_id;
END $
DELIMITER ;
CALL peliculas_en_tienda(1);
```