```
/* Author: Matthew Heino
      Assignment: Module 6 Assignment #2

    Note:  Use the Sakila database to write SQL queries to accomplish
    the tasks described below. Execute the queries on the Sakila database.
    Submit your queries as a PDF document.
*/

/* 1) Add a new customer to the customer table with the following details:

            First Name: "John"
            Last Name: "Doe"
            Email: "john.doe@example.com"
            Store ID: 1
            Active: 1
*/

-- To remove the foreign key constraints from the database error message
SET FOREIGN_KEY_CHECKS=0;

-- Note: needed to add the address_id equal to zero to get rid of the following
error
-- Error Code: 1048. Column 'address_id' cannot be null
INSERT INTO customer (first_name, last_name, email,address_id, store_id, active)
VALUES('John', 'Doe', 'john.doe@example.com', 0, 1, 1);

/* 2)  Insert a new record into the film table with these details:

            Title: "Code Masters"
        Description: "A movie about the world of coding."
        Release Year: 2024
        Language ID: 1
        Rental Duration: 7 days
        Rental Rate: 2.99
        Replacement Cost: 19.99
*/
 INSERT INTO film (title, film.description, release_year, language_id,
rental_duration, rental_rate, replacement_cost)
 VALUES('Code Masters', 'A movie about the world of coding.', 2024, 1, 7, 2.99,
19.99);

/* 3) Add a record in the actor table with:

            First Name: "Emily"
            Last Name: "Blunt"
*/
 INSERT INTO actor (first_name, last_name)
 VALUES ('Emily','Blunt');

/*
      4) Retrieve the first name, last name, and email of all active customers.
    Note: Assuming that 1 means active
*/
 SELECT first_name, last_name, email
 FROM customer
 WHERE customer.active = 1;

/*
      5) List the titles of films rented by the customer with ID 3
    Note: Using a subquery
*/

SELECT title
FROM film
WHERE film_id IN(
                  SELECT film_id
                  FROM inventory
                  WHERE inventory_id IN (
                              SELECT inventory_id
                              FROM rental
                              WHERE customer_id = 3));

/*
      6. Find all films with "Adventure" in their description.
    Display the title, description, and release year.
*/
 SELECT title, film.description, release_year
 FROM film
 WHERE film.description LIKE '%Adventure%';

/*
      7) For all films with a rental rate less than 2.00, increase
      the rental rate by 0.50.
*/

 UPDATE film
 SET rental_rate =
      CASE WHEN rental_rate < 2.00
    THEN rental_rate + 0.50
    ELSE rental_rate
 END;

/*
      8) Delete the film titled "Code Masters" that you created earlier
    in the assignment.
*/
DELETE FROM film WHERE film.title='Code Masters';

/*
      9) Remove all records from the rental table where the rental date
    is more than 5 years old.
*/
DELETE FROM rental
WHERE rental_date < NOW() - INTERVAL 5 YEAR;

/*
      10) Retrieve the title and description of all films released in the
    year 2006.

*/

SELECT title, film.description
FROM film
WHERE release_year = 2006;

/*
      11) Display the first_name, last_name, and email of all customers
    who have no rentals in the rental table.
```

```
*/
SELECT customer_id, first_name, last_name, email
FROM customer
WHERE customer_id NOT IN (
            SELECT DISTINCT(customer_id)
            FROM rental);

/*
      12) Find the total number of films for each category. Display the
    category name and the film count
*/

SELECT category.name AS "Cateogory Name", COUNT(*) AS "Film Count"
FROM film
INNER JOIN film_category
ON film_category.film_id = film.film_id
INNER JOIN category
ON category.category_id = film_category.category_id
GROUP BY category.name;


/*
      13) Calculate the total revenue generated by each store. Display
    the store id and the total revenue.
*/
SELECT store_id, SUM(amount) AS "Total Revenue"
FROM payment
INNER JOIN staff
ON staff.staff_id = payment.staff_id
GROUP BY staff.store_id;
```