# *Final Sakila Analysis*

```
USE sakila;
```

1a. You need a list of all the actors who have Display the first and last names of all actors from the table actor.

```
SELECT first_name, last_name
FROM actor;
```

1b. Display the first and last name of each actor in a single column in upper case letters. Name the column Actor Name.

```
SELECT UPPER(CONCAT(first_name, ' ', last_name)) AS 'Actor Name'
FROM actor;
```

2a. You need to find the ID number, first name, and last name of an actor, of whom you know only the first name, "Joe." What is one query would you use to obtain this information?

```
SELECT actor_id, first_name, last_name
FROM actor
WHERE first_name = 'JOE';
```

2b. Find all actors whose last name contain the letters GEN:

```
SELECT actor_id, first_name, last_name
FROM actor
WHERE last_name LIKE '%GEN%';
```

2c. Find all actors whose last names contain the letters LI. This time, order the rows by last name and first name, in that order:

```
SELECT actor_id, first_name, last_name
FROM actor
WHERE last_name LIKE '%LI%'
ORDER BY last_name, first_name;
```

2d. Using IN, display the country_id and country columns of the following countries: Afghanistan, Bangladesh, and China:

```
SELECT country_id, country
FROM country
WHERE country IN ('Afghanistan', 'Bangladesh', 'China');
```

3a. Add a middle_name column to the table actor. Position it between first_name and last_name. Hint: you will need to specify the data type.

```
SELECT * FROM actor;

ALTER TABLE actor
ADD COLUMN middle_name VARCHAR(50) AFTER first_name;
```

3b. You realize that some of these actors have tremendously long last names. Change the data type of the middle_name column to blobs.

```
ALTER TABLE actor
MODIFY COLUMN middle_name BLOB;
```

3c. Now delete the middle_name column.

```
ALTER TABLE actor
DROP COLUMN middle_name;
```

#4a. List the last names of actors, as well as how many actors have that last name.

```
SELECT last_name, COUNT(*) AS 'Count'
FROM actor
GROUP BY last_name;
```

#4b. List last names of actors and the number of actors who have that last name, but only for names that are shared by at least two actors

```
SELECT last_name, COUNT(*) AS 'Count'
FROM actor
GROUP BY last_name
HAVING Count >= 2;
```

#4c. Oh, no! The actor HARPO WILLIAMS was accidentally entered in the actor table as GROUCHO WILLIAMS, the name of Harpo's second cousin's husband's yoga teacher. Write a query to fix the record.

```
SELECT * FROM actor;
UPDATE actor
SET first_name = 'HARPO'
WHERE first_name = 'GROUCHO' AND last_name = 'WILLIAMS';
```

#4d. Perhaps we were too hasty in changing GROUCHO to HARPO. It turns out that GROUCHO was the correct name after all! In a single query, if the first name of the actor is currently HARPO, change it to GROUCHO. Otherwise, change the first name to MUCHO GROUCHO, as that is exactly what the actor will be with the grievous error. BE CAREFUL NOT TO CHANGE THE FIRST NAME OF EVERY ACTOR TO MUCHO GROUCHO, HOWEVER! (Hint: update the record using a unique identifier.)

```
UPDATE actor
SET first_name =
CASE
WHEN first_name = 'HARPO'
THEN 'GROUCHO'
ELSE 'MUCHO GROUCHO'
END
```

```
WHERE actor_id = 172;
```

#5a. You cannot locate the schema of the address table. Which query would you use to re-create it?

```
DESCRIBE sakila.address;
```

#6a. Use JOIN to display the first and last names, as well as the address, of each staff member. Use the tables staff and address:

```
SELECT first_name, last_name, address
FROM staff s
INNER JOIN address a
ON s.address_id = a.address_id;
```

#6b. Use JOIN to display the total amount rung up by each staff member in August of 2005. Use tables staff and payment.

```
SELECT first_name, last_name, SUM(amount) AS 'Total'
FROM staff s
INNER JOIN payment p
ON s.staff_id = p.staff_id
GROUP BY s.first_name, s.last_name;
```

#6c. List each film and the number of actors who are listed for that film. Use tables film_actor and film. Use inner join.

```
SELECT title, COUNT(actor_id) AS 'Total'
FROM film f
INNER JOIN film_actor a
ON f.film_id = a.film_id
GROUP BY f.title;
```

#6d. How many copies of the film Hunchback Impossible exist in the inventory system?

```
SELECT title, COUNT(inventory_id) AS 'Total'
FROM film f
INNER JOIN inventory i
ON f.film_id = i.film_id
WHERE title = "Hunchback Impossible";
```

#6e. Using the tables payment and customer and the JOIN command, list the total paid by each customer. List the customers alphabetically by last name:

```
SELECT first_name, last_name, SUM(amount) AS 'Total Paid'
FROM payment p
INNER JOIN customer c
ON p.customer_id = c.customer_id
GROUP BY p.customer_id
ORDER BY last_name ASC;
```

#7a. The music of Queen and Kris Kristofferson have seen an unlikely resurgence. As an unintended consequence, films starting with the letters K and Q have also soared in popularity. Use subqueries to display the titles of movies starting with the letters K and Q whose language is English.

```
SELECT title
```

```
FROM film
WHERE language_id IN
   (SELECT language_id
   FROM language
   WHERE name = "English")
 AND (title LIKE 'K%') OR (title LIKE 'Q%');

#7b. Use subqueries to display all actors who appear in the film Alone Trip.
SELECT first_name, last_name
FROM actor
WHERE actor_id IN
(SELECT actor_id
FROM film_actor
          WHERE film_id IN
          (SELECT film_id
   FROM film
                    WHERE title = "Alone Trip"));

#7c. You want to run an email marketing campaign in Canada, for which you will need the names and
email addresses of all Canadian customers. Use joins to retrieve this information.
SELECT last_name, first_name, email
FROM customer
INNER JOIN customer_list
ON customer.customer_id = customer_list.ID
WHERE customer_list.country = 'Canada';

#7d. Sales have been lagging among young families, and you wish to target all family movies for a
promotion. Identify all movies categorized as family films.
SELECT title, name
FROM category, film, film_category
WHERE category.category_id = film_category.category_id
AND film_category.film_id = film.film_id
AND name = 'Family';

#7e. Display the most frequently rented movies in descending order.
SELECT title AS 'movie_title', COUNT(rental_date) AS 'rent_count'
FROM film, rental, inventory
WHERE film.film_id = inventory.film_id
AND inventory.inventory_id = rental.inventory_id
GROUP BY title
ORDER BY COUNT(rental_date) DESC;

#7f. Write a query to display how much business, in dollars, each store brought in.
SELECT store.store_id AS 'store', SUM(amount) AS 'total_revenue'
FROM store, staff, payment
WHERE store.store_id = staff.store_id
AND staff.staff_id = payment.staff_id
GROUP BY store.store_id
```

```
ORDER BY SUM(amount) DESC;

#7g. Write a query to display for each store its store ID, city, and country.
SELECT store_id, city, country
FROM store, address, city, country
WHERE store.address_id = address.address_id
AND address.city_id = city.city_id
AND city.country_id = country.country_id;

#7h. List the top five genres in gross revenue in descending order. (Hint: you may need to use the
following tables: category, film_category, inventory, payment, and rental.)
SELECT name, SUM(amount) AS 'gross_revenue'
FROM category, film_category, inventory, rental, payment
WHERE category.category_id = film_category.category_id
AND film_category.film_id = inventory.film_id
AND inventory.inventory_id = rental.inventory_id
AND rental.rental_id = payment.rental_id
GROUP BY name
ORDER BY gross_revenue DESC
LIMIT 5;

#8a. In your new role as an executive, you would like to have an easy way of viewing the Top five genres
by gross revenue. Use the solution from the problem above to create a view. If you haven't solved 7h,
you can substitute another query to create a view.
#DROP VIEW IF EXISTS top_five_genres;
CREATE VIEW top_five_genres AS

    SELECT name, SUM(amount) AS 'gross_revenue'
    FROM category, film_category, inventory, rental, payment
    WHERE category.category_id = film_category.category_id
    AND film_category.film_id = inventory.film_id
    AND inventory.inventory_id = rental.inventory_id
    AND rental.rental_id = payment.rental_id
    GROUP BY name
    ORDER BY gross_revenue DESC
    LIMIT 5;

#8b. How would you display the view that you created in 8a?
SELECT * FROM top_five_genres;

#8c. You find that you no longer need the view top_five_genres. Write a query to delete it.
DROP VIEW top_five_genres;
```