**Homework #10 – SQL Vincent Wong**

**UCB Berkeley Data Bootcamp**

**July 2018**

1a. 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`**

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 first\_name, last\_name, actor\_id**

**FROM actor WHERE first\_name = "Joe";**

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

**SELECT first\_name, last\_name, actor\_id**

**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 first\_name, last\_name, actor\_id**

**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**

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.

**ALTER TABLE actor**

**ADD COLUMN middle\_name varchar(60) 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(last\_name) AS 'last\_name\_frequency' FROM actor**

**GROUP BY last\_name HAVING `last\_name\_frequency` >= 2;**

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(last\_name) AS 'last\_name\_frequency'**

**FROM actor GROUP BY last\_name Having `last\_name\_frequency` >= 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.

**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!

**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?

**SHOW CREATE TABLE 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 s.first\_name, s.last\_name, a.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 s.first\_name, s.last\_name, SUM(p.amount) FROM staff AS s**

**INNER JOIN payment AS p**

**ON p.staff\_id = s.staff\_id**

**WHERE MONTH(p.payment\_date) = 08 AND YEAR(p.payment\_date) = 2005**

**GROUP BY s.staff\_id;**

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 f.title, COUNT(fa.actor\_id) AS 'Actors' FROM film\_actor AS fa**

**INNER JOIN film as f**

**ON f.film\_id = fa.film\_id**

**GROUP BY f.title**

**ORDER BY Actors desc;**

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

**SELECT title, COUNT(inventory\_id) AS '# of copies' FROM film**

**INNER JOIN inventory USING (film\_id) WHERE title = 'Hunchback Impossible'**

**GROUP BY title;**

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 c.first\_name, c.last\_name, SUM(p.amount) AS 'Total Amount Paid' FROM payment AS p**

**JOIN customer AS c**

**ON p.customer\_id = c.customer\_id**

**GROUP BY c.customer\_id**

**ORDER BY c.last\_name;**

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 title LIKE 'K%' OR title LIKE 'Q%' AND language\_id IN**

**(**

**SELECT language\_id FROM language WHERE name = 'English'**

**);**

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 =**

**(**

**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 first\_name, last\_name, email, country**

**FROM customer cus**

**JOIN address a**

**ON (cus.address\_id = a.address\_id)**

**JOIN city cit**

**ON (a.city\_id = cit.city\_id)**

**JOIN country ctr**

**ON (cit.country\_id = ctr.country\_id)**

**WHERE ctr.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 famiy films.

**SELECT title, c.name**

**FROM film f**

**JOIN film\_category fc**

**ON (f.film\_id = fc.film\_id)**

**JOIN category c**

**ON (c.category\_id = fc.category\_id)**

**WHERE name = 'family';**

7e. Display the most frequently rented movies in descending order.

**SELECT title, COUNT(title) as 'Rentals'**

**FROM film**

**JOIN inventory**

**ON (film.film\_id = inventory.film\_id)**

**JOIN rental**

**ON (inventory.inventory\_id = rental.inventory\_id)**

**GROUP by title**

**ORDER BY rentals desc;**

7f. Write a query to display how much business, in dollars, each store brought in.

**SELECT s.store\_id, SUM(amount) AS Gross FROM payment p**

**JOIN rental r ON (p.rental\_id = r.rental\_id)**

**JOIN inventory I ON (i.inventory\_id = r.inventory\_id)**

**JOIN store s ON (s.store\_id = i.store\_id)**

**GROUP BY s.store\_id;**

7g. Write a query to display for each store its store ID, city, and country.

**SELECT store\_id, city, country FROM store s**

**JOIN address a ON (s.address\_id = a.address\_id)**

**JOIN city cit ON (cit.city\_id = a.city\_id)**

**JOIN country ctr ON(cit.country\_id = ctr.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 SUM(amount) AS 'Total Sales', c.name AS 'Genre' FROM payment p**

**JOIN rental r ON (p.rental\_id = r.rental\_id)**

**JOIN inventory i ON (r.inventory\_id = i.inventory\_id)**

**JOIN film\_category fc ON (i.film\_id = fc.film\_id)**

**JOIN category c ON (fc.category\_id = c.category\_id)**

**GROUP BY c.name**

**ORDER BY SUM(amount) DESC;**

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.

**CREATE VIEW top\_five\_genres AS**

**SELECT SUM(amount) AS 'Total Sales', c.name AS 'Genre' FROM payment p**

**JOIN rental r ON (p.rental\_id = r.rental\_id)**

**JOIN inventory I ON (r.inventory\_id = i.inventory\_id)**

**JOIN film\_category fc ON (i.film\_id = fc.film\_id)**

**JOIN category c ON (fc.category\_id = c.category\_id)**

**GROUP BY c.name**

**ORDER BY SUM(amount) DESC**

**LIMIT 10;**

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;**