|  |
| --- |
| USE sakila; |
|  |  |
|  | -- Part 1 |
|  | -- 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' |
|  | FROM actor; |
|  |  |
|  | -- Part 2 |
|  | -- 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'); |
|  |  |
|  | -- Part 3 |
|  | -- 3a. You want to keep a description of each actor. |
|  | -- You don't think you will be performing queries on a description, so create a column in the table actor named description and use the data type BLOB. |
|  | ALTER TABLE actor |
|  | ADD COLUMN description BLOB AFTER last\_update; |
|  |  |
|  | -- 3b. Very quickly you realize that entering descriptions for each actor is too much effort. Delete the description column. |
|  | ALTER TABLE actor |
|  | DROP COLUMN description; |
|  |  |
|  | -- Part 4 |
|  | -- 4a. List the last names of actors, as well as how many actors have that last name. |
|  | SELECT last\_name, COUNT(\*) as 'Last Name 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 'Last Name Count' |
|  | FROM actor |
|  | GROUP BY last\_name |
|  | HAVING COUNT(\*) >=2; |
|  |  |
|  | -- 4c. The actor HARPO WILLIAMS was accidentally entered in the actor table as GROUCHO WILLIAMS. 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. |
|  | UPDATE actor |
|  | SET first\_name = 'GROUCHO' |
|  | WHERE first\_name='HARPO' AND last\_name='WILLIAMS'; |
|  |  |
|  | -- PART 5 |
|  | -- 5a. You cannot locate the schema of the address table. Which query would you use to re-create it? |
|  | SHOW CREATE TABLE address; |
|  |  |
|  | -- PART 6 |
|  | -- 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 INNER JOIN address |
|  | ON staff.address\_id = address.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 Amount' |
|  | FROM staff INNER JOIN payment |
|  | ON staff.staff\_id = payment.staff\_id AND payment\_date LIKE '2005-08%' |
|  | GROUP BY first\_name, 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 'Actor Count' |
|  | FROM film\_actor INNER JOIN film |
|  | ON film\_actor.film\_id = film.film\_id |
|  | GROUP BY title; |
|  |  |
|  | -- 6d. How many copies of the film Hunchback Impossible exist in the inventory system? |
|  | SELECT title, COUNT(title) as 'Copies Available' |
|  | FROM film INNER JOIN inventory |
|  | ON film.film\_id = inventory.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 by Each Customer' |
|  | FROM payment INNER JOIN customer |
|  | ON payment.customer\_id = customer.customer\_id |
|  | GROUP BY first\_name, last\_name |
|  | ORDER BY last\_name; |
|  |  |
|  |  |
|  | -- Part 7 |
|  | -- 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 title IN |
|  | ( |
|  | SELECT title |
|  | FROM film |
|  | WHERE language\_id IN |
|  | ( |
|  | SELECT language\_id |
|  | FROM language |
|  | WHERE name ='English' |
|  | )language |
|  | ); |
|  |  |
|  | -- 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 first\_name, last\_name, email |
|  | FROM customer |
|  | JOIN address |
|  | ON (customer.address\_id = address.address\_id) |
|  | JOIN city |
|  | ON (city.city\_id = address.city\_id) |
|  | JOIN country |
|  | ON (country.country\_id = city.country\_id) |
|  | WHERE country.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 |
|  | FROM film |
|  | WHERE film\_id IN |
|  | ( |
|  | SELECT film\_id |
|  | FROM film\_category |
|  | WHERE category\_id IN |
|  | ( |
|  | SELECT category\_id |
|  | FROM category |
|  | WHERE name='Family' |
|  | ) |
|  | ); |
|  |  |
|  | -- 7e. Display the most frequently rented movies in descending order. |
|  | SELECT title, COUNT(rental\_id) as 'Rental Count' |
|  | FROM rental |
|  | JOIN inventory |
|  | ON (rental.inventory\_id = inventory.inventory\_id) |
|  | JOIN film |
|  | ON (inventory.film\_id = film.film\_id) |
|  | GROUP BY film.title |
|  | ORDER BY COUNT(rental\_id) DESC; |
|  |  |
|  | -- 7f. Write a query to display how much business, in dollars, each store brought in. |
|  | SELECT store.store\_id, SUM(amount) |
|  | FROM store |
|  | INNER JOIN staff |
|  | ON store.store\_id = staff.store\_id |
|  | INNER JOIN payment |
|  | ON payment.staff\_id = staff.staff\_id |
|  | GROUP BY store.store\_id; |
|  |  |
|  | -- 7g. Write a query to display for each store its store ID, city, and country. |
|  | SELECT store\_id, city, country |
|  | FROM store |
|  | INNER JOIN address |
|  | ON store.address\_id = address.address\_id |
|  | INNER JOIN city |
|  | ON city.city\_id = address.city\_id |
|  | INNER JOIN country |
|  | ON country.country\_id = city.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) |
|  | FROM category |
|  | INNER JOIN film\_category |
|  | ON category.category\_id = film\_category.category\_id |
|  | INNER JOIN inventory |
|  | ON film\_category.film\_id = inventory.film\_id |
|  | INNER JOIN rental |
|  | ON inventory.inventory\_id = rental.inventory\_id |
|  | INNER JOIN payment |
|  | ON rental.rental\_id = payment.rental\_id |
|  | GROUP BY name |
|  | ORDER BY SUM(amount) DESC LIMIT 5; |
|  |  |
|  | -- Part 8 |
|  | -- 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. |
|  | CREATE VIEW top\_revenues\_by\_genre AS |
|  | SELECT name, SUM(amount) |
|  | FROM category |
|  | INNER JOIN film\_category |
|  | ON category.category\_id = film\_category.category\_id |
|  | INNER JOIN inventory |
|  | ON film\_category.film\_id = inventory.film\_id |
|  | INNER JOIN rental |
|  | ON inventory.inventory\_id = rental.inventory\_id |
|  | INNER JOIN payment |
|  | ON rental.rental\_id = payment.rental\_id |
|  | GROUP BY name |
|  | ORDER BY SUM(amount) DESC LIMIT 5; |
|  |  |
|  | -- 8b. How would you display the view that you created in 8a? |
|  | SELECT \* FROM top\_revenues\_by\_genre; |
|  |  |
|  | -- 8c. You find that you no longer need the view top\_five\_genres. Write a query to delete it. |
|  | DROP VIEW top\_revenues\_by\_genre; |