|  |
| --- |
|  |
| use sakila; |
|  |  |
|  | select \* from actor; |
|  |  |
|  | -- 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; |
|  | Créer un champ text et mettre dedans **$F{last\_name}**+" "+**$F{first\_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 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 \* from country; |
|  |  |
|  | 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. |
|  | alter table actor |
|  | add column middle\_name varchar(20) |
|  | after first\_name; |
|  |  |
|  | select \* from actor; |
|  |  |
|  | -- 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; |
|  |  |
|  | select \* from actor; |
|  |  |
|  | -- 3c. Now delete the middle\_name column. |
|  | alter table actor |
|  | drop column middle\_name; |
|  |  |
|  | select \* from actor; |
|  |  |
|  |  |
|  | -- 4a. List the last names of actors, as well as how many actors have that last name. |
|  | select last\_name, count(last\_name) |
|  | 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(last\_name) as 'Count of Last Name' |
|  | from actor group by last\_name having count(last\_name) > 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'; |
|  | -- check |
|  | select first\_name, last\_name |
|  | from actor |
|  | where first\_name = 'HARPO' 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.) |
|  |  |
|  | -- first find the unique id |
|  |  |
|  | select first\_name, last\_name, actor\_id |
|  | from actor |
|  | where first\_name = 'HARPO' and last\_name = 'WILLIAMS'; |
|  |  |
|  | update actor set first\_name = |
|  | CASE |
|  | when (actor\_id = 172 and first\_name = 'HARPO') then 'GROUCHO' |
|  | when (actor\_id = 172 and first\_name <> 'HARPO') then 'MUCHO GROUCHO' |
|  | ELSE |
|  | first\_name |
|  | END; |
|  |  |
|  | -- check |
|  | select first\_name, last\_name, actor\_id |
|  | from actor |
|  | where actor\_id= 172; |
|  |  |
|  | -- 5a. You cannot locate the schema of the address table. Which query would you use to re-create it? Hint: https://dev.mysql.com/doc/refman/5.7/en/show-create-table.html |
|  |  |
|  | 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, coalesce(a.address, 'No Address Found in Table') as 'address' |
|  | from |
|  | staff as s |
|  | 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.staff\_id, s.first\_name, s.last\_name, coalesce(concat('$', format(sum(p.amount), 2)), '$0') as amount |
|  | from |
|  | staff as s |
|  | join |
|  | payment as p |
|  | on |
|  | s.staff\_id = p.staff\_id |
|  | where |
|  | p.payment\_date >= '2005-08-01 00:00:00' |
|  | and |
|  | p.payment\_date < '2005-08-02 00:00:00' |
|  | 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 'Number of Actors' |
|  | from |
|  | film as f |
|  | inner join |
|  | film\_actor as fa |
|  | on f.film\_id = fa.film\_id |
|  | group by f.title; |
|  |  |
|  |  |
|  | -- 6d. How many copies of the film Hunchback Impossible exist in the inventory system? |
|  |  |
|  | select f.title, count(i.film\_id) as 'Number of Copies' |
|  | from |
|  | film as f |
|  | inner join |
|  | inventory as i |
|  | on |
|  | f.film\_id = i.film\_id |
|  | group by f.title |
|  | having f.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: Total amount paid |
|  |  |
|  | select c.first\_name, c.last\_name, sum(coalesce(p.amount, 0)) |
|  | from customer as c |
|  | join |
|  | payment as p |
|  | on c.customer\_id = p.customer\_id |
|  | group by c.first\_name, c.last\_name |
|  | order by c.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 f.title |
|  | from film as f |
|  | where |
|  | f.language\_id in |
|  | (select l.language\_id from language l where name = 'English') |
|  | and |
|  | f.title rlike '^[K,Q]'; |
|  |  |
|  |  |
|  | -- 7b. Use subqueries to display all actors who appear in the film Alone Trip. |
|  |  |
|  | select a.first\_name, a.last\_name |
|  | from actor as a |
|  | where |
|  | a.actor\_id in |
|  | (select fa.actor\_id from film\_actor fa |
|  | where fa.film\_id = |
|  | (select f.film\_id from film f 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 cust.first\_name |
|  | ,cust.last\_name |
|  | ,coalesce(cust.email, 'No Email Available') as 'Email' |
|  | from |
|  | customer cust |
|  | inner join |
|  | address as a |
|  | on cust.address\_id = a.address\_id |
|  | inner join |
|  | city |
|  | on a.city\_id = city.city\_id |
|  | inner join |
|  | country |
|  | on city.country\_id = country.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 famiy films. |
|  |  |
|  | select f.title |
|  | from film f |
|  | inner join |
|  | film\_category as fc |
|  | on |
|  | f.film\_id = fc.film\_id |
|  | inner join |
|  | category as c |
|  | on |
|  | fc.category\_id = c.category\_id |
|  | where |
|  | c.name = 'Family'; |
|  |  |
|  |  |
|  | -- 7e. Display the most frequently rented movies in descending order. |
|  |  |
|  | select f.title, count(r.rental\_id) as 'Number of Rentals' |
|  | from |
|  | film as f |
|  | inner join |
|  | inventory as i |
|  | on f.film\_id = i.film\_id |
|  | inner join |
|  | rental as r |
|  | on r.inventory\_id = i.inventory\_id |
|  | group by f.title |
|  | order by count(r.rental\_id) desc; |
|  |  |
|  | -- 7f. Write a query to display how much business, in dollars, each store brought in. |
|  |  |
|  | select s.store\_id |
|  | ,coalesce(concat('$', format(sum(p.amount), 2)), '$0') as amount |
|  | from |
|  | store as s |
|  | inner join |
|  | staff |
|  | on s.store\_id = staff.store\_id |
|  | inner join |
|  | payment as p |
|  | on staff.staff\_id = p.staff\_id |
|  | group by s.store\_id; |
|  |  |
|  |  |
|  | -- 7g. Write a query to display for each store its store ID, city, and country. |
|  |  |
|  | select s.store\_id |
|  | ,city.city |
|  | ,cntry.country |
|  | from |
|  | store as s |
|  | inner join |
|  | address as a |
|  | on s.address\_id = a.address\_id |
|  | inner join |
|  | city |
|  | on |
|  | city.city\_id = a.city\_id |
|  | inner join |
|  | country cntry |
|  | on |
|  | city.country\_id = cntry.country\_id; |
|  |  |
|  | -- 7h. List the top five genres in gross revenue in descending order. (Hint: you may need to use the following tables: category, film\_category, inventory, payment, and rental.) |
|  |  |
|  | select catg.name |
|  | ,coalesce(concat('$', format(sum(p.amount), 2)), '$0') as 'Gross Revenue' |
|  | from |
|  | category catg |
|  | inner join |
|  | film\_category fc |
|  | on catg.category\_id = fc.category\_id |
|  | inner join |
|  | inventory i |
|  | on fc.film\_id = i.film\_id |
|  | inner join |
|  | rental r |
|  | on i.inventory\_id = r.inventory\_id |
|  | inner join |
|  | payment p |
|  | on r.rental\_id = p.rental\_id |
|  | group by catg.name |
|  | order by coalesce(concat('$', format(sum(p.amount), 2)), '$0') desc |
|  | limit 5; |
|  |  |
|  | -- 8a. In your new role as an executive, you would like to have an easy way of viewing the Top five genres by gross revenue. Use the solution from the problem above to create a view. If you haven't solved 7h, you can substitute another query to create a view. |
|  |  |
|  | create view top\_five\_genres as |
|  | select catg.name |
|  | ,coalesce(concat('$', format(sum(p.amount), 2)), '$0') as 'Gross Revenue' |
|  | from |
|  | category catg |
|  | inner join |
|  | film\_category fc |
|  | on catg.category\_id = fc.category\_id |
|  | inner join |
|  | inventory i |
|  | on fc.film\_id = i.film\_id |
|  | inner join |
|  | rental r |
|  | on i.inventory\_id = r.inventory\_id |
|  | inner join |
|  | payment p |
|  | on r.rental\_id = p.rental\_id |
|  | group by catg.name |
|  | order by coalesce(concat('$', format(sum(p.amount), 2)), '$0') desc |
|  | limit 5; |
|  |  |
|  | -- check |
|  | select \* From top\_five\_genres; |
|  |  |
|  | -- 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; |
|  |  |