Exercises

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

SELECT UPPER(CONCAT(first_name,' ',last_name)) AS `Actor Name` from actor ORDER BY `Actor Name`;

```
SELECT UPPER(CONCAT(first_name,' ',last_name)) AS `Actor Name` from actor
ORDER BY `Actor Name`;
```

Actor Name
ADAM GRANT
ADAM HOPPER
AL GARLAND
ALAN DREYFUSS
ALBERT JOHANSSON
ALBERT NOLTE
ALEC WAYNE
ANGELA HUDSON
ANGELA WITHERSPOON
ANGELINA ASTAIRE

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

SELECT * from actor where last_name like '%GEN%';

```
SELECT * from actor where last_name like '%GEN%';
```

actor_id	first_name	last_name	last_update
14	VIVIEN	BERGEN	2006-02-15 04:34:33
41	JODIE	DEGENERES	2006-02-15 04:34:33
107	GINA	DEGENERES	2006-02-15 04:34:33
166	NICK	DEGENERES	2006-02-15 04:34:33
NULL	NULL	NULL	NULL

3. **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');

```
SELECT country_id,country from country where country IN('Afghanistan','Bangladesh','China');
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

country_id	country
1	Afghanistan
12	Bangladesh
23	China
NULL	NULL

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

SELECT last_name,count(*) count from actor

GROUP BY last_name

ORDER BY count DESC;

```
SELECT last_name,count(*) count from actor
GROUP BY last_name
ORDER BY count DESC;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

last_name	count
KILMER	5
NOLTE	4
TEMPLE	4
AKROYD	3
ALLEN	3
BERRY	3
DAVIS	3
DEGENERES	3
GARLAND	3
GUINESS	3

5.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(*) count from actor

GROUP BY last_name

HAVING count > 1

ORDER BY count DESC;

```
SELECT last_name,count(*) count from actor
GROUP BY last_name
HAVING count > 1
ORDER BY count DESC;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

last_name	count
KILMER	5
NOLTE	4
TEMPLE	4
AKROYD	3
ALLEN	3
HOPKINS	3
DAVIS	3
BERRY	3
HARRIS	3
GARLAND	3

6. **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";

Limit to 1000 rows

```
SELECT * FROM sakila.actor where first_name like '%HARPO%';

UPDATE actor SET first_name ="HARPO" WHERE first_name ="GROUCHO" AND last_name="WILLIAMS";
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

actor_id	first_name	last_name	last_update
172	HARPO	WILLIAMS	2024-07-20 11:43:46
NULL	NULL	NULL	NULL

7. **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 JOIN address USING(address_id);

```
2
3 •  SELECT first_name, last_name, address from staff JOIN address USING(address_id);
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

first_name	last_name	address
Mike	Hillyer	23 Workhaven Lane
Jon	Stephens	1411 Lillydale Drive

8. **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.film_id, f.title,COUNT(fa.actor_id) actor_count FROM film f INNER JOIN film_actor fa USING(film_id)

GROUP BY f.film_id

ORDER BY actor_count DESC

actor | actor | film_actor | film

Limit to 1000 rows

```
1 •  SELECT * FROM sakila.film;
2 •  SELECT f.film_id, f.title,COUNT(fa.actor_id) actor_count FROM film f INNER JOIN film_actor fa USING(film_id)
3    GROUP BY f.film_id
4    ORDER BY actor_count DESC
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

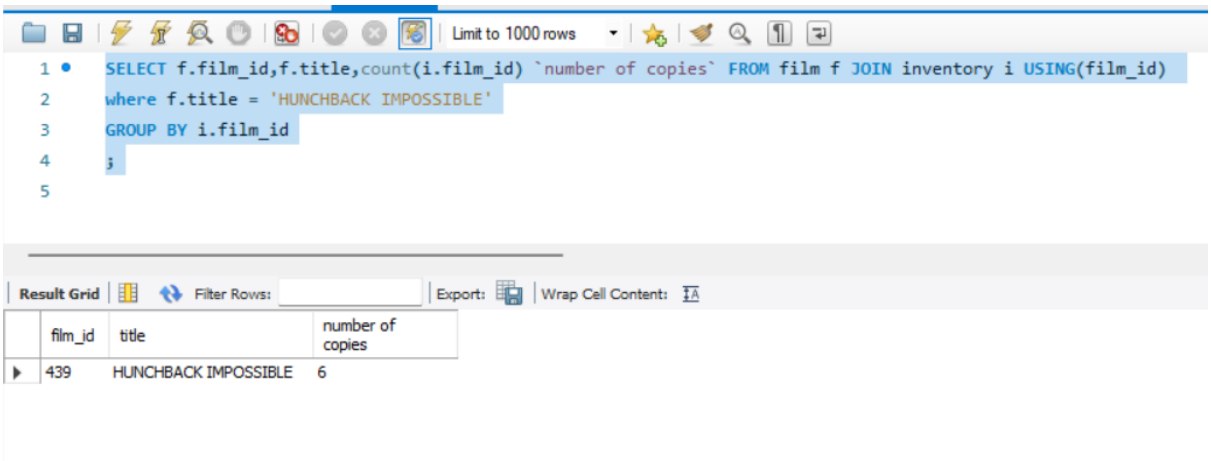
film_id	title	actor_count
508	LAMBS CINCINATTI	15
87	BOONDOCK BALLROOM	13
146	CHITTY LOCK	13
188	CRAZY HOME	13
249	DRACULA CRYSTAL	13
606	MUMMY CREATURES	13
714	RANDOM GO	13
34	ARABIA DOGMA	12
414	HELLFIGHTERS SIERRA	12
517	LESSON CLEOPATRA	12
529	LONELY ELEPHANT	12
802	SKY MIRACLE	12
892	TITANIC BOONDOCK	12
312	FIDDLER LOST	11
342	FUGITIVE MAGUIRE	11

Result Grid | Form Editor | Field Types | Query Stats

9. **How many copies of the film Hunchback Impossible exist in the inventory system?**

SELECT f.film_id,f.title,count(i.film_id) `number of copies` FROM film f JOIN inventory i USING(film_id) where f.title = 'HUNCHBACK IMPOSSIBLE'

GROUP BY i.film_id;



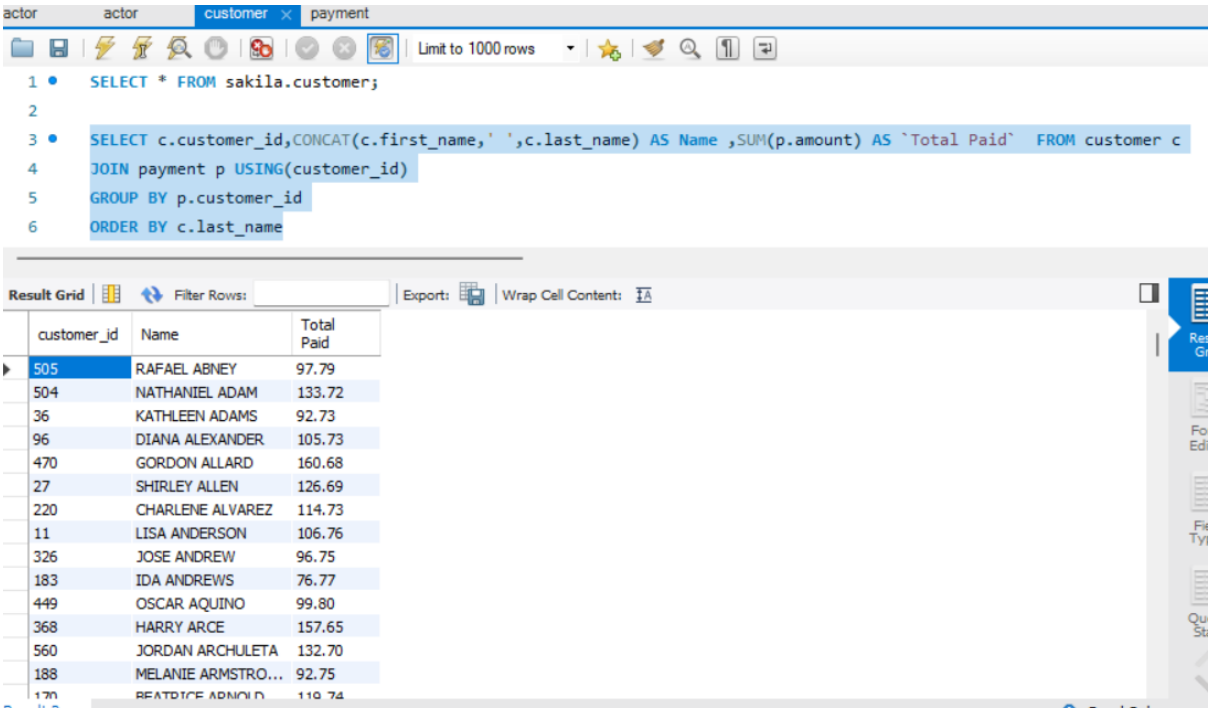
10. **Using the tables payment and customer and the JOIN command, list the total paid by each customer. List the customers alphabetically by last name**

SELECT c.customer_id,CONCAT(c.first_name,' ',c.last_name) AS Name ,SUM(p.amount) AS `Total Paid` FROM customer c

JOIN payment p USING(customer_id)

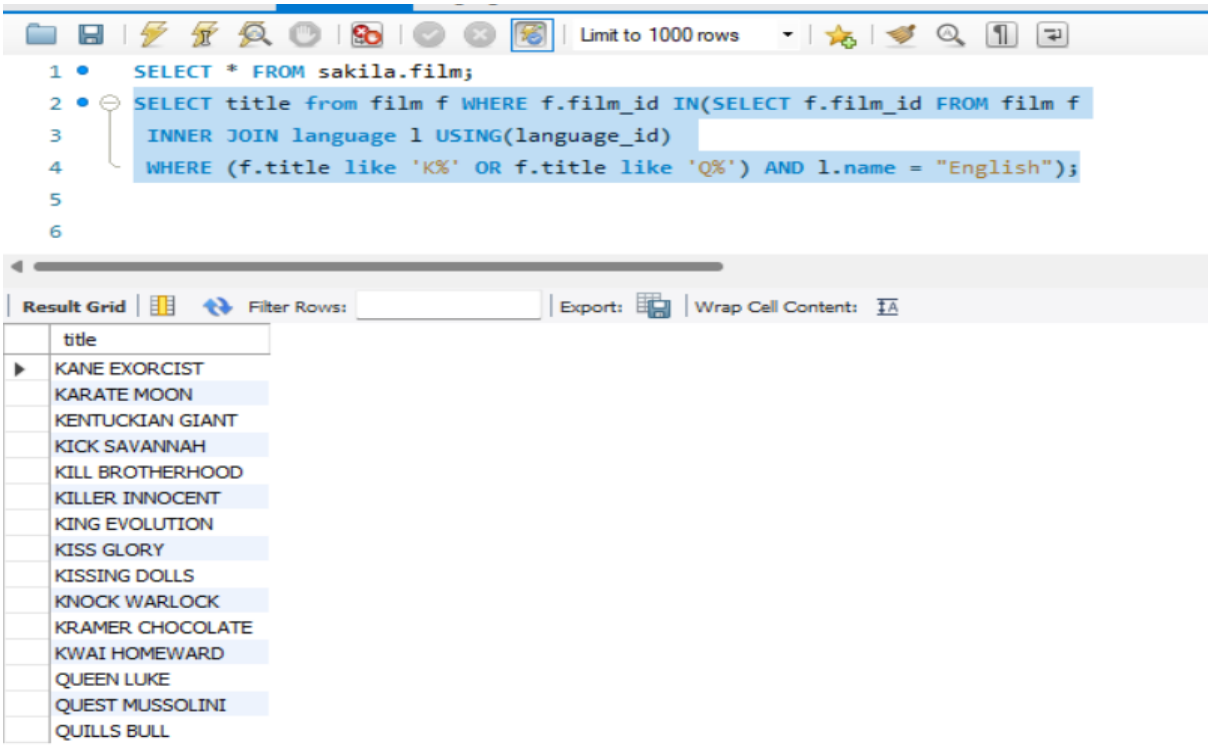
GROUP BY p.customer_id

ORDER BY c.last_name



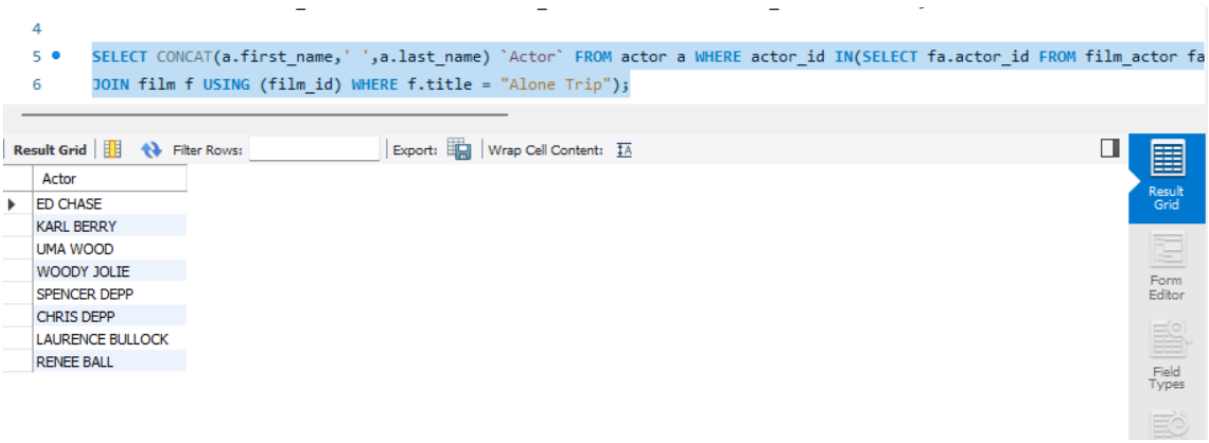
11 The music of Queen and Kris Kristofferson have seen an unlikely resurgence. As an unintended consequence, films starting with the letters `K` and `Q` have also soared in popularity. Use subqueries to display the titles of movies starting with the letters `K` and `Q` whose language is English.

SELECT title from film f WHERE f.film_id IN(SELECT f.film_id FROM film f INNER JOIN language l USING(language_id) WHERE (f.title like 'K%' OR f.title like 'Q%') AND l.name = "English");



12. Use subqueries to display all actors who appear in the film `Alone Trip`.

SELECT CONCAT(a.first_name,' ',a.last_name) `Actor` FROM actor a WHERE actor_id IN(SELECT fa.actor_id FROM film_actor fa JOIN film f USING (film_id) WHERE f.title = "Alone Trip");



13. **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.**

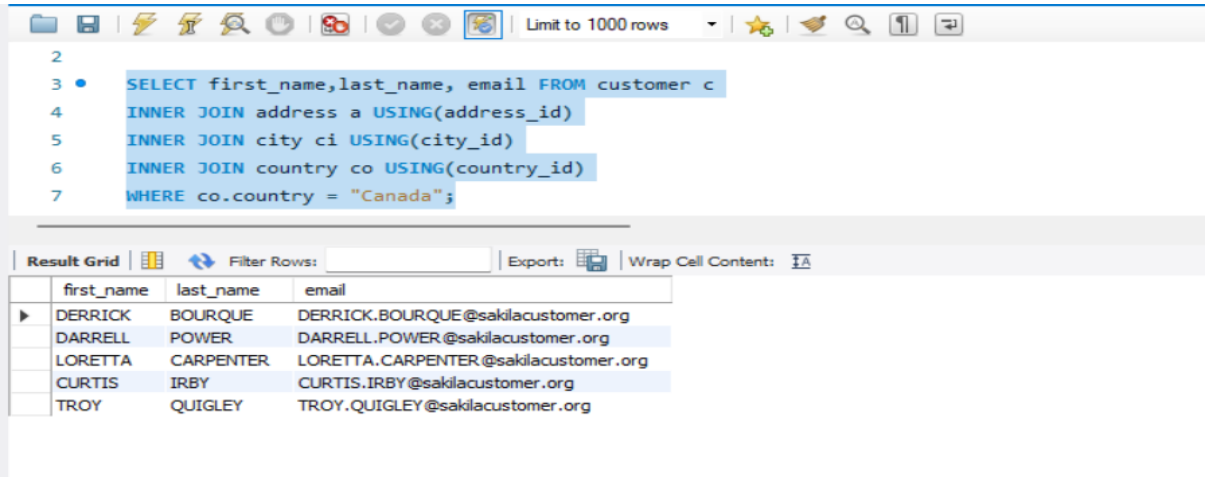
SELECT first_name,last_name, email FROM customer c

INNER JOIN address a USING(address_id)

INNER JOIN city ci USING(city_id)

INNER JOIN country co USING(country_id)

WHERE co.country = "Canada";



first_name	last_name	email
DERRICK	BOURQUE	DERRICK.BOURQUE@sakilacustomer.org
DARRELL	POWER	DARRELL.POWER@sakilacustomer.org
LORETTA	CARPENTER	LORETTA.CARPENTER@sakilacustomer.org
CURTIS	IRBY	CURTIS.IRBY@sakilacustomer.org
TROY	QUIGLEY	TROY.QUIGLEY@sakilacustomer.org

14. **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.**

We can write query in two ways

SELECT title from film f

INNER JOIN film_category fc USING(film_id)

INNER JOIN category c USING(category_id)

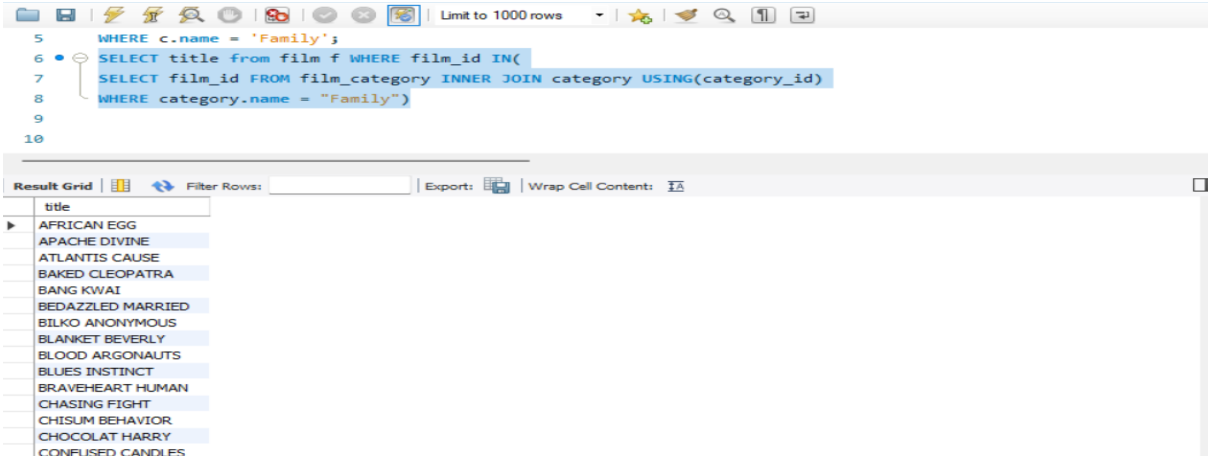
WHERE c.name = 'Family';

OR

SELECT title from film f WHERE film_id IN(

SELECT film_id FROM film_category INNER JOIN category USING(category_id)

WHERE category.name = "Family")



title
AFRICAN EGG
APACHE DIVINE
ATLANTIS CAUSE
BAKED CLEOPATRA
BANG KWAI
BEDAZZLED MARRIED
BILKO ANONYMOUS
BLANKET BEVERLY
BLOOD ARGONAUTS
BLUES INSTINCT
BRAVEHEART HUMAN
CHASING FIGHT
CHISUM BEHAVIOR
CHOCOLAT HARRY
CONFUSED CANDLES

15. Create a Stored procedure to get the count of films in the input category (IN category_name, OUT count)

```
CREATE PROCEDURE GetFilmCountByCategory(
  IN category_name VARCHAR(20),
  OUT film_count INT
)
BEGIN
   SELECT COUNT(*) INTO film_count
   FROM film f WHERE film_id IN(
          SELECT film_id FROM film_category INNER JOIN category USING(category_id)
          WHERE category.name = category_name);
END //
DELIMITER ;
SET @film_count = 0;
CALL GetFilmCountByCategory('Family', @film_count);
SELECT @film_count;
```

```
DELIMITER //
CREATE PROCEDURE GetFilmCountByCategory(
  IN category_name VARCHAR(20),
  OUT film_count INT
)
BEGIN
   SELECT COUNT(*) INTO film_count
   FROM film f WHERE film_id IN(
   SELECT film_id FROM film_category INNER JOIN category USING(category_id)
   WHERE category.name = category_name);
END //
DELIMITER ;
SET @film_count = 0;
CALL GetFilmCountByCategory('Family', @film_count);
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

@film_count
69

16. **Display the most frequently rented movies in descending order.**

```
SELECT f.title, COUNT(r.rental_id) rental_count FROM film f
INNER JOIN inventory i USING(film_id)
INNER JOIN rental r USING(inventory_id)
GROUP BY f.film_id
ORDER BY rental_count DESC
```

Limit to 1000 rows

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

title	rental_count
BUCKET BROTHERHOOD	34
ROCKETEER MOTHER	33
FORWARD TEMPLE	32
GRIT CLOCKWORK	32
JUGGLER HARDLY	32
RIDGEMONT SUBMARINE	32
SCALAWAG DUCK	32
APACHE DIVINE	31
GOODFELLAS SALUTE	31
HOBBIT ALIEN	31
NETWORK PEAK	31
ROBBERS JOON	31
RUSH GOODFELLAS	31
TIMBERLAND SKY	31
WIFE TURN	31
ZORRO ARK	31

17. **Write a query to display for each store its store ID, city, and country.**

```
SELECT s.store_id , c.city , co.country FROM store s
INNER JOIN address a USING(address_id)
INNER JOIN city c USING(city_id)
INNER JOIN country co USING(country_id);
```

Limit to 1000 rows

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

store_id	city	country
1	Lethbridge	Canada
2	Woodridge	Australia

18. List the genres and its gross revenue.

SELECT c.name genres ,SUM(p.amount) gross_revenue FROM film f

INNER JOIN inventory i USING(film_id)

INNER JOIN rental r USING(inventory_id)

INNER JOIN payment p USING(rental_id)

INNER JOIN film_category fc ON f.film_id = fc.film_id

INNER JOIN category c ON fc.category_id = c.category_id

GROUP BY c.name

ORDER BY gross_revenue DESC;

```
SELECT c.name genres ,SUM(p.amount) gross_revenue FROM film f
INNER JOIN inventory i USING(film_id)
INNER JOIN rental r USING(inventory_id)
INNER JOIN payment p USING(rental_id)
INNER JOIN film_category fc ON f.film_id = fc.film_id
INNER JOIN category c ON fc.category_id = c.category_id
GROUP BY c.name
ORDER BY gross_revenue DESC;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

genres	gross_revenue
Sports	5314.21
Sci-Fi	4756.98
Animation	4656.30
Drama	4587.39
Comedy	4383.58
Action	4375.85
New	4351.62
Games	4281.33
Foreign	4270.67
Family	4226.07
Documen...	4217.52
Horror	3722.54

19. Create a View for the above query(18)

CREATE VIEW genere_revenue_view AS

SELECT c.name genres ,SUM(p.amount) gross_revenue FROM film f

INNER JOIN inventory i USING(film_id)

INNER JOIN rental r USING(inventory_id)

INNER JOIN payment p USING(rental_id)

INNER JOIN film_category fc ON f.film_id = fc.film_id

INNER JOIN category c ON fc.category_id = c.category_id

GROUP BY c.name

ORDER BY gross_revenue DESC;

SELECT * FROM genere_revenue_view;

```
CREATE VIEW genere_revenue_view AS
SELECT c.name genres ,SUM(p.amount) gross_revenue FROM film f
INNER JOIN inventory i USING(film_id)
INNER JOIN rental r USING(inventory_id)
INNER JOIN payment p USING(rental_id)
INNER JOIN film_category fc ON f.film_id = fc.film_id
INNER JOIN category c ON fc.category_id = c.category_id
GROUP BY c.name
ORDER BY gross_revenue DESC;

SELECT * FROM genere_revenue_view;
```

genres	gross_revenue
Sports	5314.21
Sci-Fi	4756.98
Animation	4656.30
Drama	4587.39
Comedy	4383.58
Action	4375.85
New	4351.62
Games	4281.33
Foreign	4270.67

20. Select top 5 genres in gross revenue view.

SELECT * FROM genere_revenue_view LIMIT 5;

```
CREATE VIEW genere_revenue_view AS
SELECT c.name genres ,SUM(p.amount) gross_revenue FROM film f
INNER JOIN inventory i USING(film_id)
INNER JOIN rental r USING(inventory_id)
INNER JOIN payment p USING(rental_id)
INNER JOIN film_category fc ON f.film_id = fc.film_id
INNER JOIN category c ON fc.category_id = c.category_id
GROUP BY c.name
ORDER BY gross_revenue DESC;

SELECT * FROM genere_revenue_view LIMIT 5;
```

genres	gross_revenue
Sports	5314.21
Sci-Fi	4756.98
Animation	4656.30
Drama	4587.39
Comedy	4383.58