SIMPLE AGGREGATIONS

Columns values can be aggregated by applying functions to the column in the select clause:

- COUNT
- SUM
- AVG
- MAX
- MIN

COUNT & COUNT WITH DISTINCT:

Count: The count function provides the number of rows returned by SELECT clause.
General form:

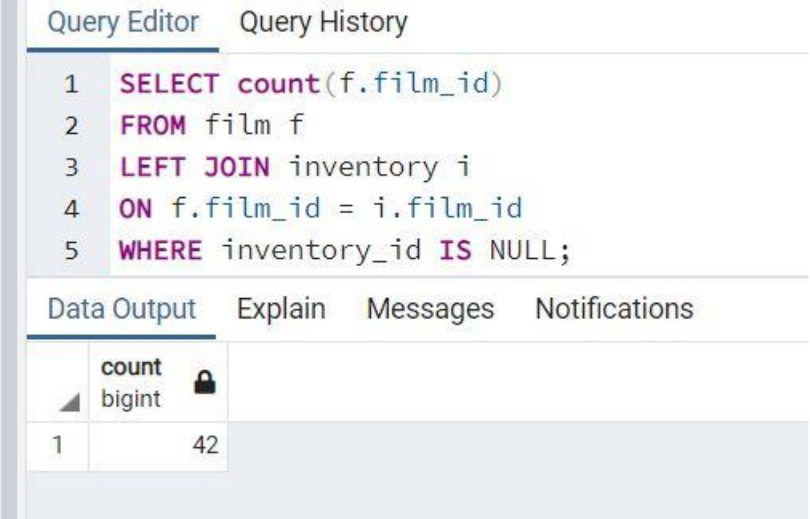
SELECT COUNT (*)   - counts the number of values.
SELECT COUNT (attribute)

- DISTINCT inside COUNT can be used to get a unique set of values.

General form:

SELECT COUNT (DISTINCT (attribute))

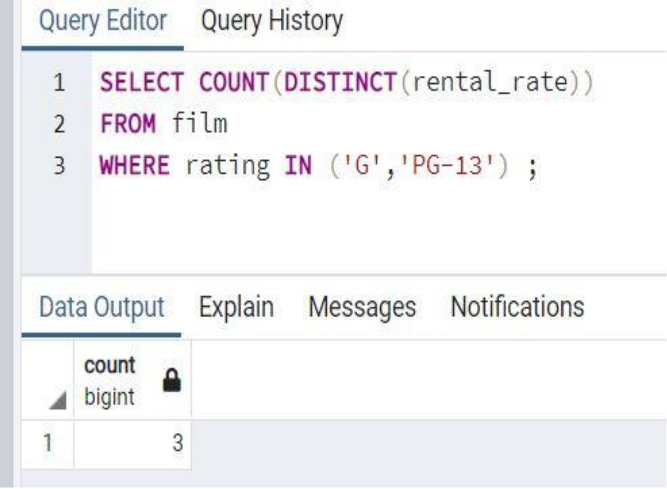
Example: How many films do not have any inventory? Hint: Left Join the film table to the inventory table.



```
SELECT count(f.film_id)
FROM film f
LEFT JOIN inventory i
ON f.film_id = i.film_id
WHERE inventory_id IS NULL;
```

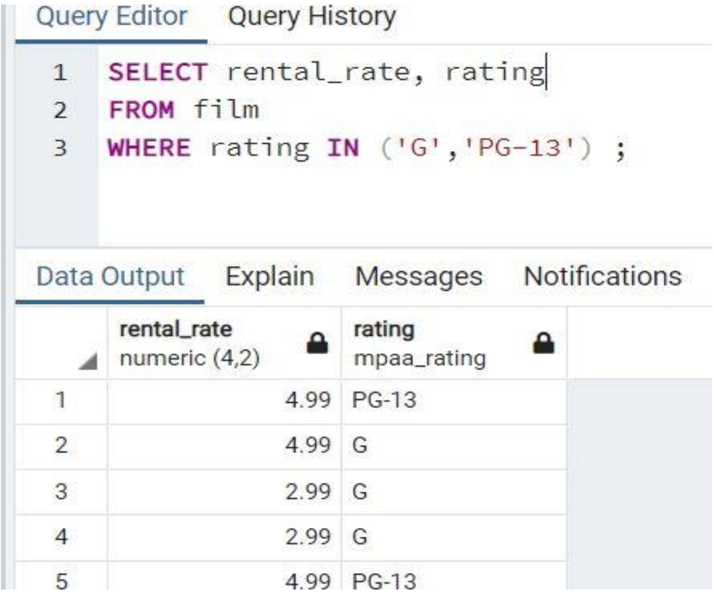
	count bigint
1	42

Example: Count the number of unique rental rates assigned to films with either ‘G’ or ‘PG-13’ rating.
Hint: Refer to the film table



```
SELECT COUNT(DISTINCT(rental_rate))
FROM film
WHERE rating IN ('G','PG-13') ;
```

	count bigint
1	3



```
SELECT rental_rate, rating
FROM film
WHERE rating IN ('G','PG-13') ;
```

	rental_rate numeric (4,2)	rating mpaa_rating
1	4.99	PG-13
2	4.99	G
3	2.99	G
4	2.99	G
5	4.99	PG-13

(see the above result)

SUM:

Like SUM function in Excel --- it sums up all the values in a column.

Unlike COUNT, SUM function is only for columns with numeric data.

If there is a null value, the SUM function treats it as zero.

AVG:

Like AVERAGE function in Excel, the AVG function returns the mean of the data.

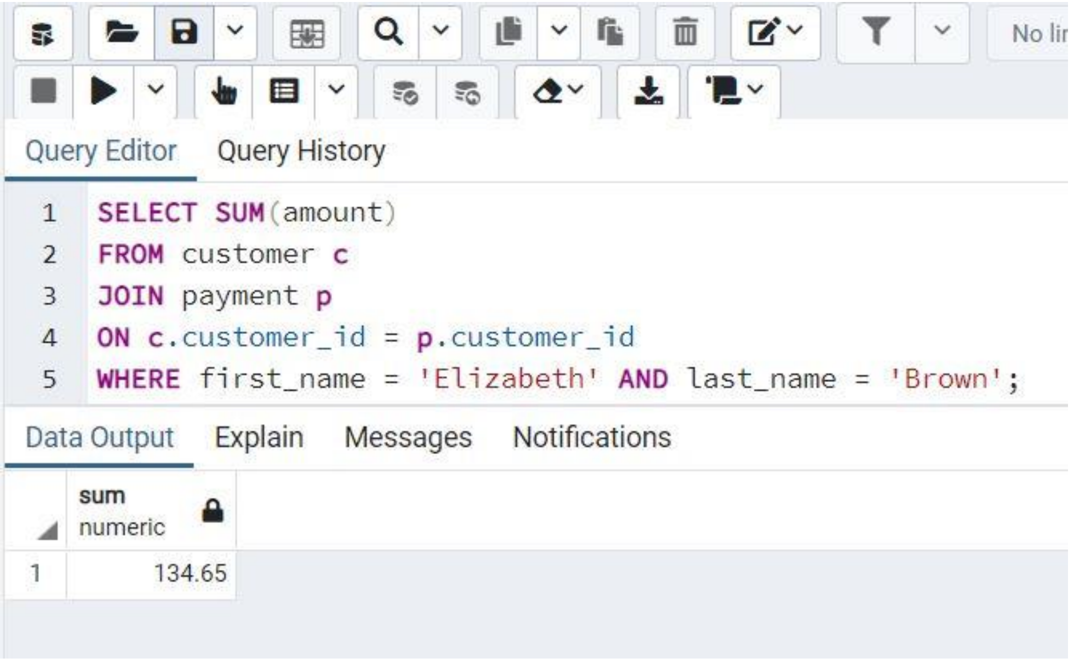
(Mean is obtained by dividing the sum of all the values in the column by the number of values in a column.

This aggregate function ignores the NULL values in both the numerator and the denominator.

Example_1:

Calculate the total amount paid by the customer named Elizabeth Brown.

Hint: Join the customer table to the payment table



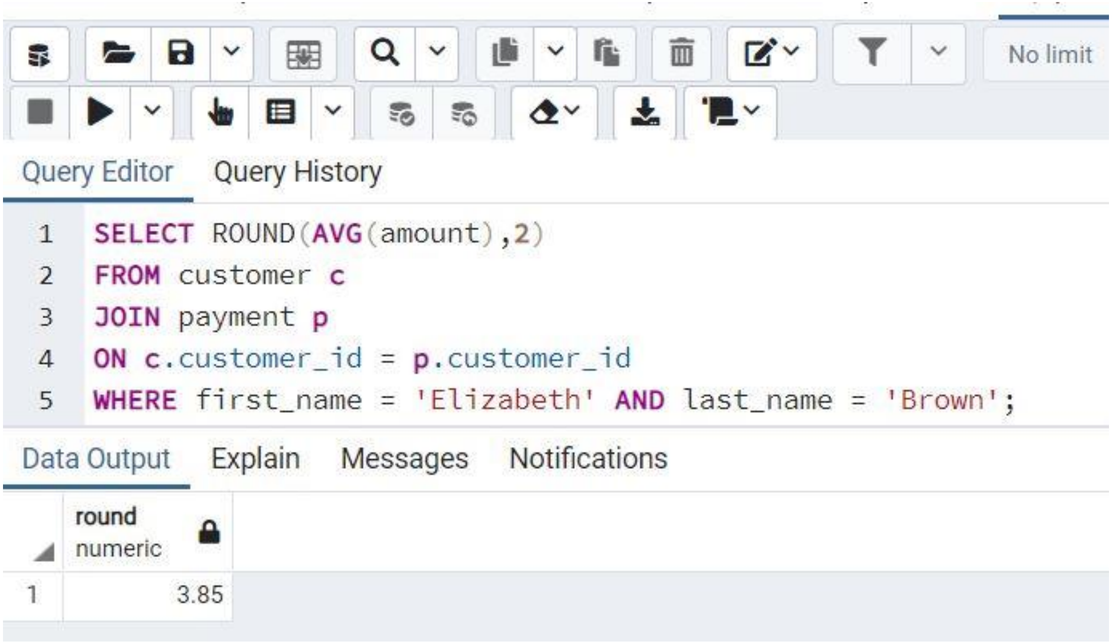
```sql
SELECT SUM(amount)
FROM customer c
JOIN payment p
ON c.customer_id = p.customer_id
WHERE first_name = 'Elizabeth' AND last_name = 'Brown';
```

```
   sum
   numeric
1  134.65
```

Example_2:

Calculate the total amount paid by the customer named Elizabeth Brown.

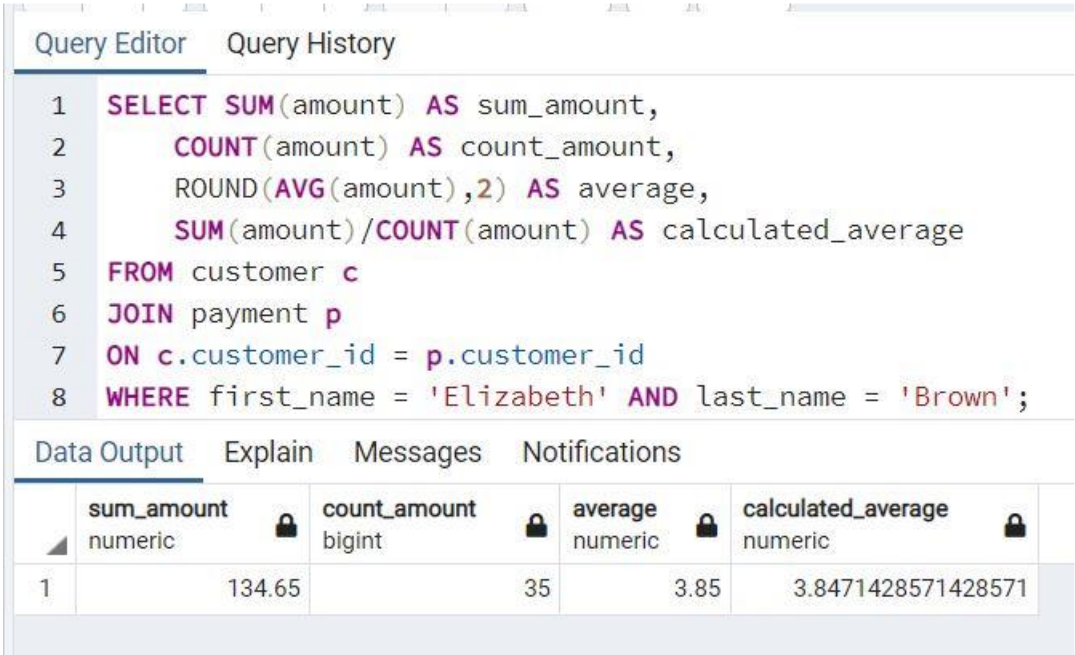
Hint: Join the customer table to the payment table



```sql
SELECT ROUND(AVG(amount),2)
FROM customer c
JOIN payment p
ON c.customer_id = p.customer_id
WHERE first_name = 'Elizabeth' AND last_name = 'Brown';
```

```
   round
   numeric
1  3.85
```

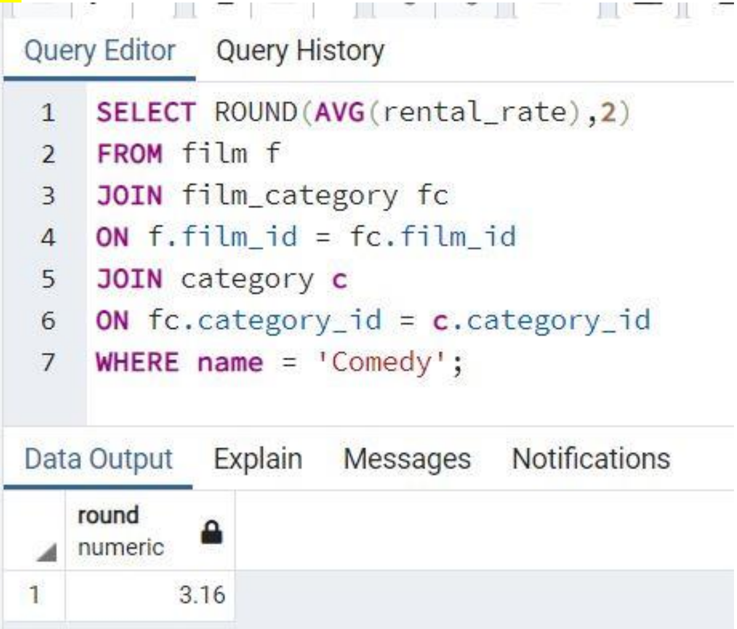
Results,



```sql
SELECT SUM(amount) AS sum_amount,
    COUNT(amount) AS count_amount,
    ROUND(AVG(amount),2) AS average,
    SUM(amount)/COUNT(amount) AS calculated_average
FROM customer c
JOIN payment p
ON c.customer_id = p.customer_id
WHERE first_name = 'Elizabeth' AND last_name = 'Brown';
```

	sum_amount numeric	count_amount bigint	average numeric	calculated_average numeric
1	134.65	35	3.85	3.8471428571428571

Q. Find the average rental rate (or price) charged for comedy films. Round it up to 2 decimal places.

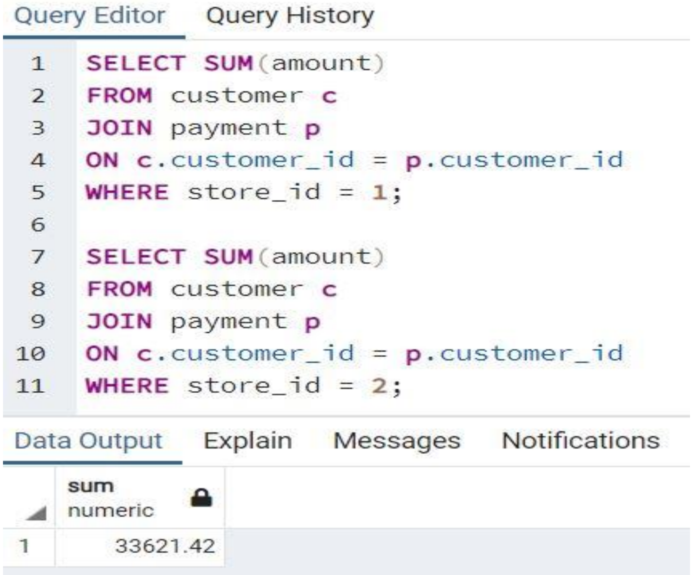


```sql
SELECT ROUND(AVG(rental_rate),2)
FROM film f
JOIN film_category fc
ON f.film_id = fc.film_id
JOIN category c
ON fc.category_id = c.category_id
WHERE name = 'Comedy';
```

	round numeric
1	3.16

Q. Find the total revenue for each store: (Note: Revenue is based on amount paid by the customers)

a) Store 1 b) Store 2



```sql
SELECT SUM(amount)
FROM customer c
JOIN payment p
ON c.customer_id = p.customer_id
WHERE store_id = 1;

SELECT SUM(amount)
FROM customer c
JOIN payment p
ON c.customer_id = p.customer_id
WHERE store_id = 2;
```

	sum numeric
1	33621.42

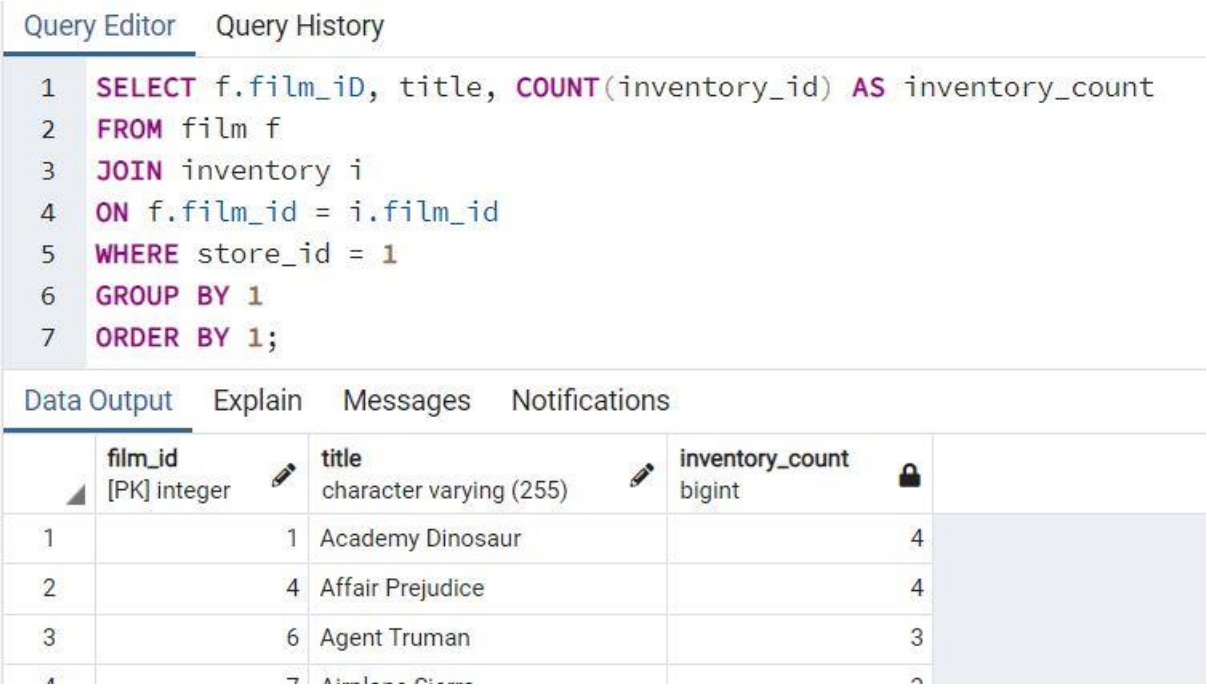
GROUPBY:

GROUPBY allows you to aggregate data within subsets of data or subgroup categories.
If any aggregation is used, then each element of the SELECT clause must be either:

1. Aggregated, or
2. An attribute on the GROUPBY list

Order of statements
SELECT > FROM > WHERE > **GROUPBY** > ORDERBY > LIMIT

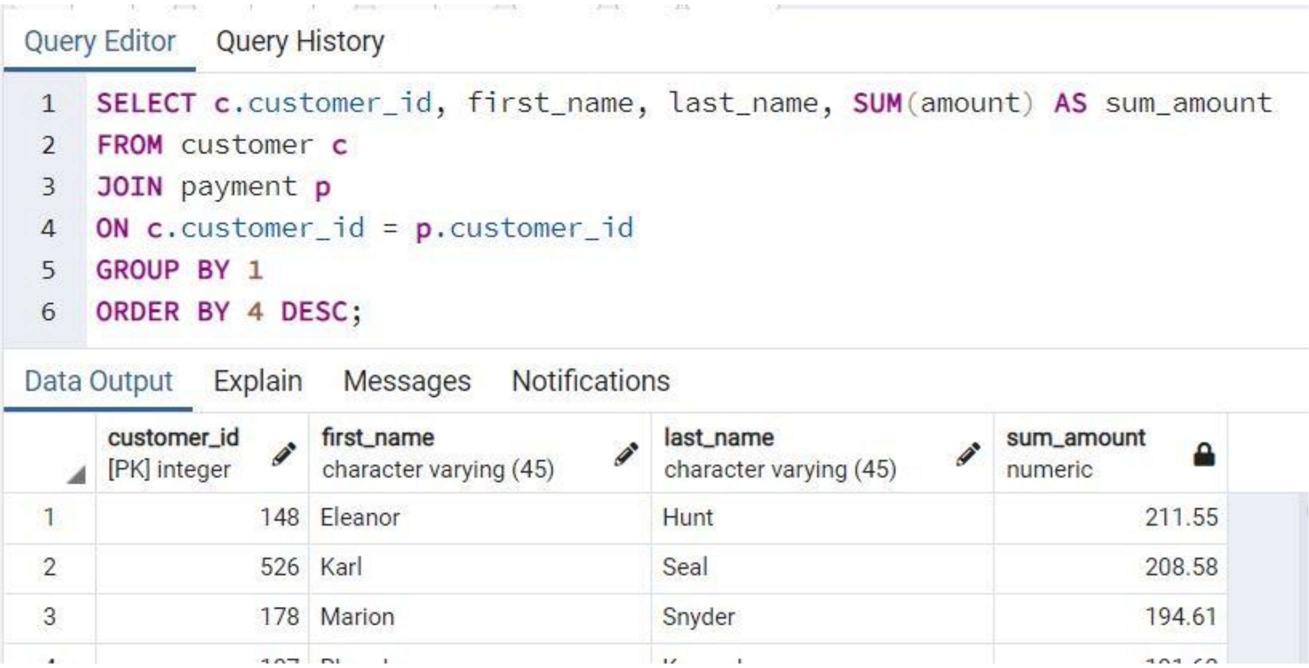
Example 1:
Show the number of inventory per film at store 1. Show the film ID, title and inventory count and sort the result by film id. Hint: Link the film table to the inventory table.



```sql
SELECT f.film_iD, title, COUNT(inventory_id) AS inventory_count
FROM film f
JOIN inventory i
ON f.film_id = i.film_id
WHERE store_id = 1
GROUP BY 1
ORDER BY 1;
```

	film_id [PK] integer	title character varying (255)	inventory_count bigint
1	1	Academy Dinosaur	4
2	4	Affair Prejudice	4
3	6	Agent Truman	3
4	7	Airplane Sierra	2

Example 2: Who are the company's power customer? (those who bring in the most revenue). Show their customer id, first name, last name and total amount paid to the company.
Hint: Join the customer table to the payment table. Note: Revenue is based on the amount paid by the customers.

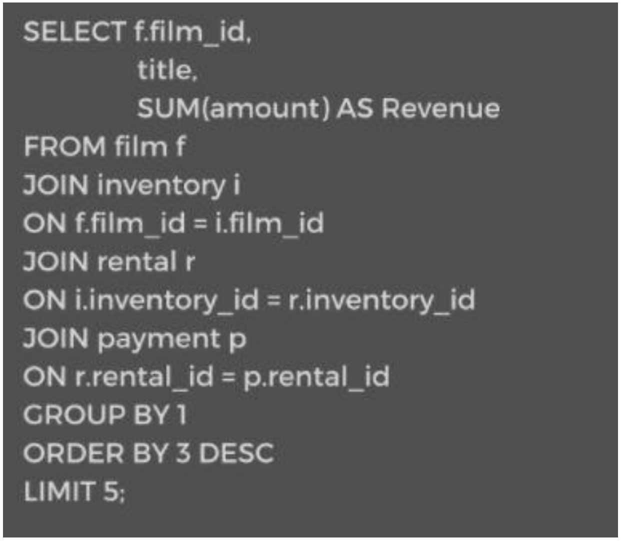


```sql
SELECT c.customer_id, first_name, last_name, SUM(amount) AS sum_amount
FROM customer c
JOIN payment p
ON c.customer_id = p.customer_id
GROUP BY 1
ORDER BY 4 DESC;
```

	customer_id [PK] integer	first_name character varying (45)	last_name character varying (45)	sum_amount numeric
1	148	Eleanor	Hunt	211.55
2	526	Karl	Seal	208.58
3	178	Marion	Snyder	194.61
4	137	Rhonda	Kennedy	191.62

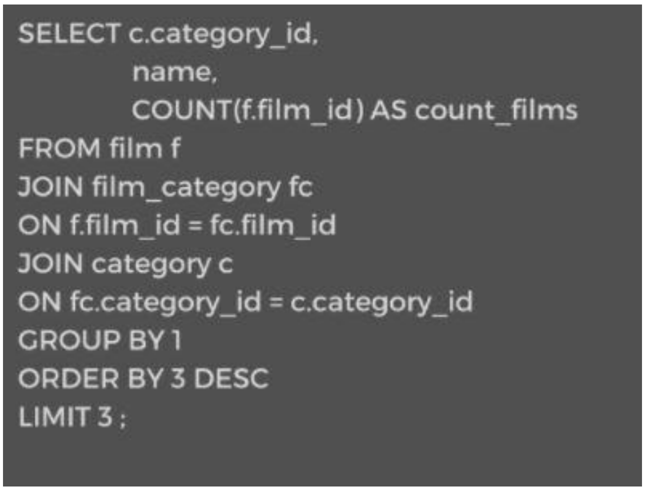
TRY:

1. Which are the top 5 revenue-generating films? Compute the company's revenue per film title. Show the film id, title and the total revenue per title and sort the results by revenue.



```
SELECT f.film_id,
            title,
            SUM(amount) AS Revenue
FROM film f
JOIN inventory i
ON f.film_id = i.film_id
JOIN rental r
ON i.inventory_id = r.inventory_id
JOIN payment p
ON r.rental_id = p.rental_id
GROUP BY 1
ORDER BY 3 DESC
LIMIT 5;
```

2. Which top 3 categories do we have the most films in? Show the category id, category name and the count of films.



```
SELECT c.category_id,
            name,
            COUNT(f.film_id) AS count_films
FROM film f
JOIN film_category fc
ON f.film_id = fc.film_id
JOIN category c
ON fc.category_id = c.category_id
GROUP BY 1
ORDER BY 3 DESC
LIMIT 3 ;
```

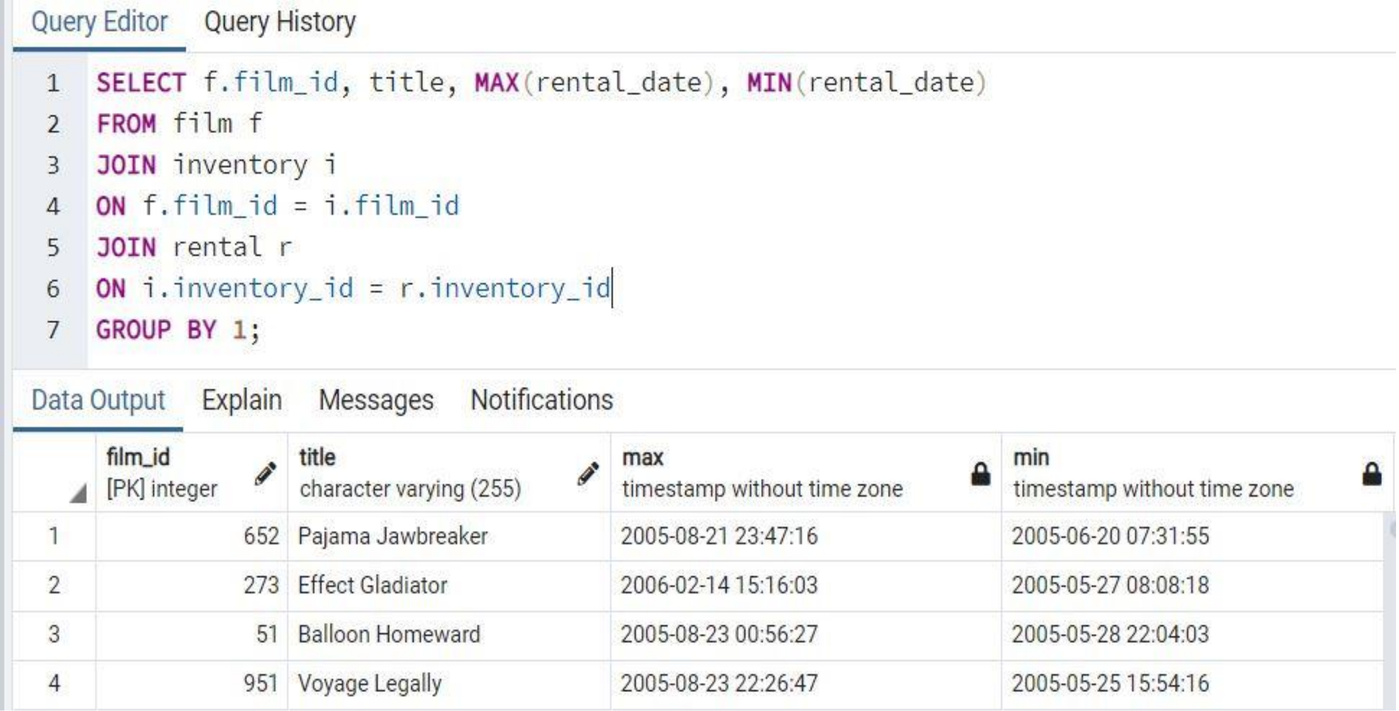
MIN & MAX

MIN will return the lowest number, earliest date or if it's a non-numerical value – the value closest alphabetically to A. **MAX** will return the highest number, latest date or if it's a non-numerical value – the value closest alphabetically to Z.

General Form:

SELECT MIN (attribute)
SELECT MAX (attribute)

Example: Show the last date each film was rented out and the first time it was rented out. (Make sure to include film ID and title.

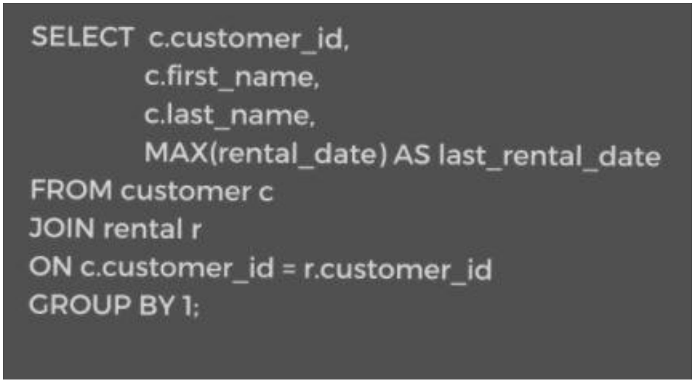


```sql
SELECT f.film_id, title, MAX(rental_date), MIN(rental_date)
FROM film f
JOIN inventory i
ON f.film_id = i.film_id
JOIN rental r
ON i.inventory_id = r.inventory_id
GROUP BY 1;
```

	film_id [PK] integer	title character varying (255)	max timestamp without time zone	min timestamp without time zone
1	652	Pajama Jawbreaker	2005-08-21 23:47:16	2005-06-20 07:31:55
2	273	Effect Gladiator	2006-02-14 15:16:03	2005-05-27 08:08:18
3	51	Balloon Homeward	2005-08-23 00:56:27	2005-05-28 22:04:03
4	951	Voyage Legally	2005-08-23 22:26:47	2005-05-25 15:54:16

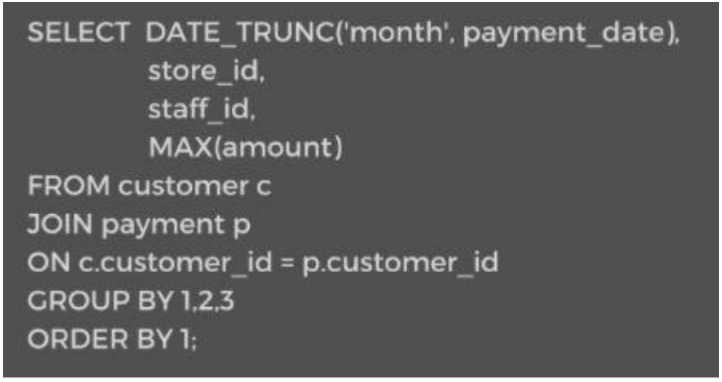
TRY IT:

1. Check out which customers have been inactive. Produce a table showing each customer's ID, first name, last name, and last rental date. Hint: Join the customer table to the rental table.



```sql
SELECT  c.customer_id,
        c.first_name,
        c.last_name,
        MAX(rental_date) AS last_rental_date
FROM customer c
JOIN rental r
ON c.customer_id = r.customer_id
GROUP BY 1;
```

2. Show the highest payment transaction processed per month by each stuff in each store. Hint: Join the customer table to the payment table. Remember to show the highest payment transaction per month, per store and per staff.



```sql
SELECT  DATE_TRUNC('month', payment_date),
        store_id,
        staff_id,
        MAX(amount)
FROM customer c
JOIN payment p
ON c.customer_id = p.customer_id
GROUP BY 1,2,3
ORDER BY 1;
```