




# EJERCICIOS SQL

Máster Data Analytics para la Empresa - EDEM

**Carlos Donoso Cabero**

1. Proporciona una SQL que muestre los siguientes datos:



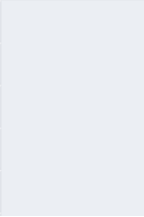
- - Nombre Actor
- - Apellido Actor

 dvdrental/postgres@server1

Query Editor   Query History

1   `select first_name, last_name from actor;`


Data Output   Explain   Messages   Notifications

	<b>first_name</b> character varying (45) 	<b>last_name</b> character varying (45) 	
1	Penelope	Guinness	
2	Nick	Wahlberg	
3	Ed	Chase	
4	Jennifer	Davis	
5	Johnny	Lollobrigida	

```
select first_name, last_name from actor;
```

2. Proporciona una SQL que muestre los siguientes datos:

- - Nombre Actor
- - Título de la Película

 dvdrental/postgres@server1

Query Editor   Query History

```
1 select
2     concat(actor.first_name, ' ', actor.last_name) as actor_name,
3     film.title
4 from actor
5     inner join film_actor on actor.actor_id = film_actor.actor_id
6     inner join film on film_actor.film_id = film.film_id;
```


Data Output   Explain   Messages   Notifications

	<b>actor_name</b> text	<b>title</b> character varying (255)
1	Penelope Guinness	Academy Dinosaur
2	Penelope Guinness	Anaconda Confessions
3	Penelope Guinness	Angels Life
4	Penelope Guinness	Bulworth Commandments
5	Penelope Guinness	Cheaper Clyde

```
select
    concat(actor.first_name, ' ', actor.last_name) as actor_name,
    film.title
from actor
    inner join film_actor on actor.actor_id = film_actor.actor_id
    inner join film on film_actor.film_id = film.film_id;
```

3. Proporciona una SQL que muestre los siguientes datos:




- - Nombre Actor
- - Número de películas
- - Ordenar de mayor a menor

 dvdrental/postgres@server1

[Query Editor](#) [Query History](#)

```
1 select
2     concat(actor.first_name, ' ', actor.last_name) as actor_name,
3     count(distinct film.film_id) as num_films
4 from actor
5     inner join film_actor on actor.actor_id = film_actor.actor_id
6     inner join film on film_actor.film_id = film.film_id
7 group by actor.actor_id
8 order by num_films desc;
```


[Data Output](#) [Explain](#) [Messages](#) [Notifications](#)

	 actor_name text	 num_films bigint	
1	Gina Degeneres	42	
2	Walter Torn	41	
3	Mary Keitel	40	
4	Matthew Carrey	39	
5	Sandra Kilmer	37	

```
select
    concat(actor.first_name, ' ', actor.last_name) as actor_name,
    count(distinct film.film_id) as num_films
from actor
    inner join film_actor on actor.actor_id = film_actor.actor_id
    inner join film on film_actor.film_id = film.film_id
group by actor.actor_id
order by num_films desc;
```

4. Proporciona una SQL que muestre los siguientes datos:

- - Película
- - Número de veces alquilada

 dvdrental/postgres@server1

Query Editor   Query History

```
1 select
2     film.title,
3     count(distinct rental.rental_id) as num_rentals
4 from film
5     inner join inventory on film.film_id = inventory.film_id
6     inner join rental on inventory.inventory_id = rental.inventory_id
7 group by film.film_id
8 order by num_rentals desc;
```


Data Output   Explain   Messages   Notifications

	title character varying (255)	num_rentals bigint
1	Bucket Brotherhood	34
2	Rocketeer Mother	33
3	Grit Clockwork	32
4	Scalawag Duck	32
5	Juggler Hardly	32

```
select
    film.title,
    count(distinct rental.rental_id) as num_rentals
from film
    inner join inventory on film.film_id = inventory.film_id
    inner join rental on inventory.inventory_id = rental.inventory_id
group by film.film_id
order by num_rentals desc;
```

5. Proporciona una SQL que muestre los siguientes datos:

- - Película
- - Dinero recaudado por película

dvdrental/postgres@server1

Query Editor

Query History

```
1 select
2     film.title,
3     sum(payment.amount) as amount
4 from film
5     inner join inventory on film.film_id = inventory.film_id
6     inner join rental on inventory.inventory_id = rental.inventory_id
7     inner join payment on rental.rental_id = payment.rental_id
8 group by film.film_id
9 order by amount desc;
```

Data Output

Explain

Messages

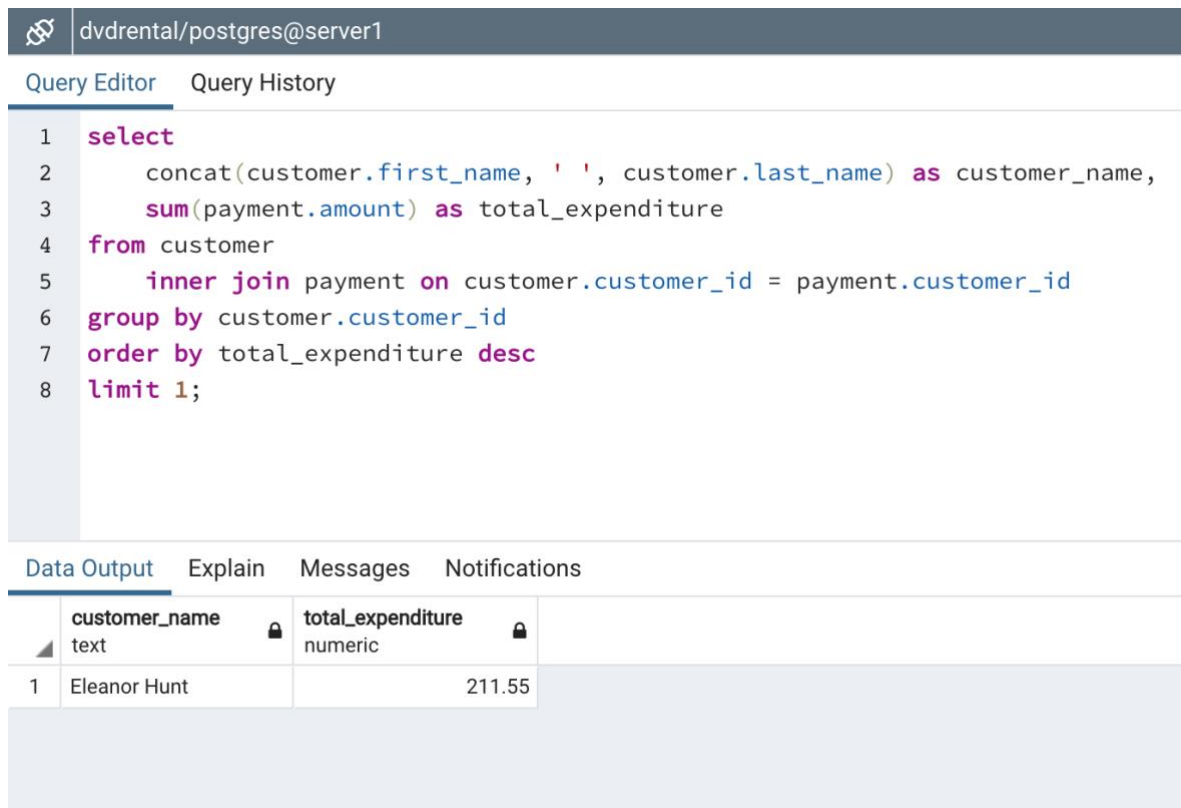
Notifications

	<div>title</div> <div>character varying (255)</div>	<div>amount</div> <div>numeric</div>	
1	Telegraph Voyage	215.75	
2	Zorro Ark	199.72	
3	Wife Turn	198.73	
4	Innocent Usual	191.74	
5	Hustler Party	190.78	

```
select
    film.title,
    sum(payment.amount) as amount
from film
    inner join inventory on film.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 film.film_id
order by amount desc;
```

6. Proporciona una SQL que muestre los siguientes datos:

- Nombre del mejor cliente (mayor gasto)



The screenshot shows a PostgreSQL Query Editor interface. At the top, the connection is labeled 'dvdrental/postgres@server1'. Below this, there are tabs for 'Query Editor' and 'Query History'. The 'Query Editor' tab is active, displaying a SQL query. Below the query editor, there are tabs for 'Data Output', 'Explain', 'Messages', and 'Notifications'. The 'Data Output' tab is active, showing the results of the query in a table format.

```
1 select
2     concat(customer.first_name, ' ', customer.last_name) as customer_name,
3     sum(payment.amount) as total_expenditure
4 from customer
5     inner join payment on customer.customer_id = payment.customer_id
6 group by customer.customer_id
7 order by total_expenditure desc
8 limit 1;
```

	customer_name text	total_expenditure numeric
1	Eleanor Hunt	211.55

```
select
    concat(customer.first_name, ' ', customer.last_name) as customer_name,
    sum(payment.amount) as total_expenditure
from customer
    inner join payment on customer.customer_id = payment.customer_id
group by customer.customer_id
order by total_expenditure desc
limit 1;
```

7. Proporciona una SQL que muestre los siguientes datos:

- Nombre del mejor cliente (mayor número alquileres)

dvrental/postgres@server1

Query Editor Query History

```
1 select
2     concat(customer.first_name, ' ', customer.last_name) as customer_name,
3     count(rental.rental_id) as total_rentals
4 from customer
5     inner join rental on customer.customer_id = rental.customer_id
6 group by customer.customer_id
7 order by total_rentals desc
8 limit 1;
9
```

Data Output Explain Messages Notifications

	customer_name text	total_rentals bigint	
1	Eleanor Hunt	46	

```
select
    concat(customer.first_name, ' ', customer.last_name) as customer_name,
    count(rental.rental_id) as total_rentals
from customer
    inner join rental on customer.customer_id = rental.customer_id
group by customer.customer_id
order by total_rentals desc
limit 1;
```