<u>Ejercicios SQL Bootcamp Data Engineer - EDVAI</u>

Consignas:

- A) Escribir las querys/consultas necesarias para llegar al resultado (print), usando windows functions.
- B) Las consultas deben ser subidas a un proyecto público de github y compartir el link al instructor.

Nota: el proyecto de github debe tener al menos dos commits (puede ser uno por el punto B y otro subir un archivo .sql con las consultas) y deberá ser compartido con el instructor.

AVG

 Obtener el promedio de precios por cada categoría de producto. La cláusula OVER(PARTITION BY CategoryID) específica que se debe calcular el promedio de precios por cada valor único de CategoryID en la tabla.

Print[.]

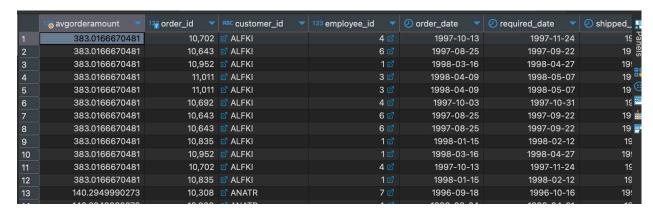
FIIIIL.			
RBC category_name T:	RBC product_name T:	123 unit_price \(\bar{\pi}	1% avgpricebycategory 🚺
Beverages	Guaraná Fantástica	4.5	37.9791666667
Beverages	Ipoh Coffee	46	37.9791666667
Beverages	Chartreuse verte	18	37.9791666667
Beverages	Côte de Blaye	263.5	37.9791666667
Beverages	Steeleye Stout	18	37.9791666667
Beverages	Sasquatch Ale	14	37.9791666667
Beverages	Lakkalikööri	18	37.9791666667
Beverages	Rhönbräu Klosterbier	7.75	37.9791666667
Beverages	Outback Lager	15	37.9791666667
Beverages	Chai	18	37.9791666667
Beverages	Laughing Lumberjack Lager	14	37.9791666667
Beverages	Chang	19	37.9791666667
Condiments	Gula Malacca	19.450000763	22.8541668256
Condiments	Original Frankfurter grüne Soße	13	22.8541668256
<u>.</u>			00.0544000050

Respuesta:

```
select c.category_name , p.product_name , p.unit_price ,
avg(p.unit_price ) over (partition by c.category_id) as avgpricebycategory
from products p inner join categories c
    on c.category_id = p.category_id
```

2. Obtener el promedio de venta de cada cliente:

Print:

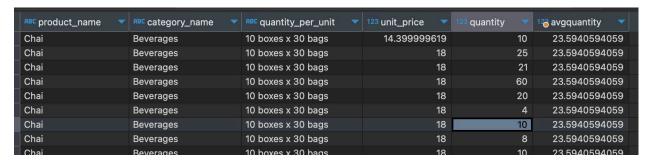


Respuesta:

select avg(od.unit_price * od.quantity) over (partition by customer_id) as avgorderamount, *
from orders o inner join order_details od
on o.order_id = od.order_id

3. Obtener el promedio de cantidad de productos vendidos por categoría (product__name, quantity_per_unit, unit_price, quantity, avgquantity) y ordenarlo por nombre de la categoría y nombre del producto

Print:



Respuesta:

```
select p.product_name , c.category_name , p.quantity_per_unit , p.unit_price , od.quantity,
avg(od.quantity ) over (partition by c.category_id ) as avgquantity
from products p inner join order_details od
on od.product_id = p.product_id
inner join categories c
on c.category_id = p.category_id
order by c.category_name, p.product_name
```

Consulta:

¿Qué es mejor hacer partition by por category_id porque es numérico ó category_name?

MIN

Selecciona el ID del cliente, la fecha de la orden y la fecha más antigua de la orden para cada cliente de la tabla 'Orders'.

Print:

customer_id	Ø order_date ▼	earliestorderdate ▼
M ALFKI	1998-01-15	1997-08-25
☑ ALFKI	1997-10-03	1997-08-25
☑ ALFKI	1998-04-09	1997-08-25
☑ ALFKI	1997-10-13	1997-08-25
☑ ALFKI	1997-08-25	1997-08-25
☑ ALFKI	1998-03-16	1997-08-25
☑ ANATR	1997-08-08	1996-09-18
☑ ANATR	1998-03-04	1996-09-18
☑ ANATR	1996-09- cus	stomer_id: varchar(5) -09-18
☑ ANATR	1997-11-28	1996-09-18
☑ ANTON	1997-09-22	1996-11-27
☑ ANTON	1997-05-13	1996-11-27
☑ ANTON	1998-01-28	1996-11-27
☑ ANTON	1997-09-25	1996-11-27
☑ ANTON	1997-04-15	1996-11-27
☑ ANTON	1997-06-19	1996-11-27
☑ ANTON	1996-11-27	1996-11-27

Respuesta:

select o.customer_id , o.order_date,
min(o.order_date) over (partition by o.customer_id) as earliestorderdate
from orders o inner join customers c
on c.customer_id = o.customer_id

MAX

 Seleccione el id de producto, el nombre de producto, el precio unitario, el id de categoría y el precio unitario máximo para cada categoría de la tabla Products.

Print:

12% product_id	RBC product_name	123 unit_price 🔻	123 category_id The state of	126 maxunitprice
24	Guaraná Fantástica	4.5	1 🗹	263.5
43	Ipoh Coffee	46	1 ₫	263.5
39	Chartreuse verte	18	1₫	263.5
38	Côte de Blaye	263.5	1₫	263.5
35	Steeleye Stout	18	1 ♂	263.5
34	Sasquatch Ale	14	1 ₫	263.5
76	Lakkalikööri	18	1 ♂	263.5
75	Rhönbräu Klosterbier	7.75	1 ₫	263.5
70	Outback Lager	15	1 ₫	263.5
<u> </u>	Chai	18	1 ₫	263.5
67	Laughing Lumberjack Lager	14	1 ♂	263.5
2	Chang	19	1 ♂	263.5
44	Gula Malacca	19.450000763	2 🗹	43.90000153
77	Original Frankfurter grüne Soße	13	2 🗹	43.90000153
	Monthy and Crombons Cours	40	2 =7	42,000,00152

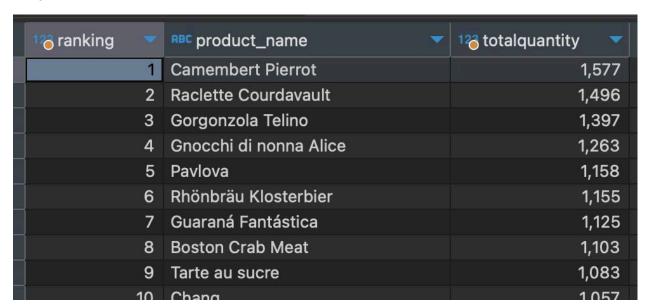
Respuesta

Otra forma con "inner join":

select p.product_id , p.product_name , p.unit_price , c.category_id ,
max(p.unit_price) over (partition by p.category_id) as maxunitprice
from products p inner join categories c
on c.category_id = p.category_id

Row_number

6. Obtener el ranking de los productos más vendidos Print:



Respuesta:

select row_number() over (order by sum(od.quantity) desc) as ranking, p.product_name,
sum(od.quantity) as totalquantity
from products p inner join order_details od
on p.product_id = od.product_id
group by p.product_id



7. Asignar numeros de fila para cada cliente, ordenados por customer_id Print:

Respuesta:

select row_number() over (order by c.customer_id asc) as rownumber,*
from customers c

8. Obtener el ranking de los empleados más jóvenes () ranking, nombre y apellido del empleado, fecha de nacimiento)

Print:

1% ranking	¥	employeename 🔻	Ø birth_date ▼
	1	Anne Dodsworth	1966-01-27
	2	Janet Leverling	1963-08-30
	3	Michael Suyama	1963-07-02
Ī	4	Robert King	1960-05-29
	5	Laura Callahan	1958-01-09
	6	Steven Buchanan	1955-03-04
	7	Andrew Fuller	1952-02-19
	8	Nancy Davolio	1948-12-08
	9	Margaret Peacock	1937-09-19

1er forma:

select row_number() over (order by e.birth_date **desc) as** *ranking*, **CONCAT(***e*.first_name , '',*e*.last_name) **AS** *employeename*, *e*.birth_date **from** employees *e*

2da forma:

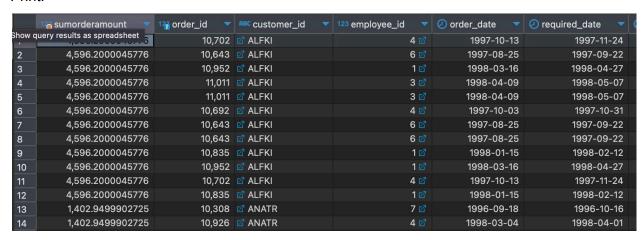
select

row_number() over(order by e.birth_date desc) as ranking,
e.first_name || ' ' || e.last_name as employeename, e.birth_date
from employees e

SUM

9. Obtener la suma de venta de cada cliente

Print:



Respuesta:

select sum(od.quantity * od.unit_price) over (partition by customer_id) as sumorderamount, *
from orders o inner join order_details od
on o.order_id = od.order_id

10. Obtener la suma total de ventas por categoría de producto

Print:

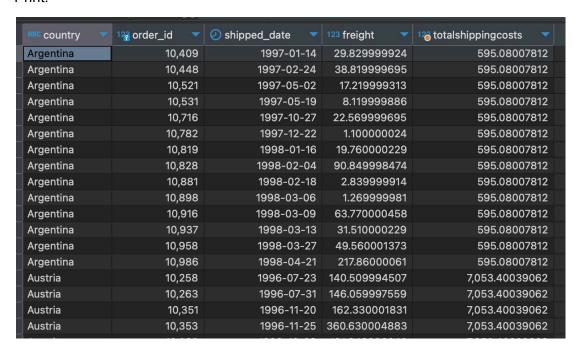
				i Î
Ctrl+click to open SQL conso	product_name	123 unit_price 🔻	123 quantity	1 ² 6 totalsales ▼
Beverages	Chai	14.399999619	10	286,526.9500956535
Beverages	Chai	18	25	286,526.9500956535
Beverages	Chai	18	21	286,526.9500956535
Beverages	Chai	18	60	286,526.9500956535
Beverages	Chai	18	20	286,526.9500956535
Beverages	Chai	18	4	286,526.9500956535
Beverages	Chai	18	10	286,526.9500956535
Beverages	Chai	18	8	286,526.9500956535
Beverages	Chai	18	10	286,526.9500956535
Beverages	Chai	18	40	286,526.9500956535
Beverages	Chai	18	6	286,526.9500956535
Reverages	Chai	18	3	286 526 9500956535

Respuesta:

```
select c.category_name , p.product_name , od.unit_price , od.quantity,
sum (od.quantity * od.unit_price ) over (partition by c.category_name ) as totalsales
from order_details od inner join products p
on od.product_id = p.product_id inner join categories c
on c.category_id = p.category_id
order by c.category_id , p.product_name
```

11. Calcular la suma total de gastos de envío por país de destino, luego ordenarlo por país y por orden de manera ascendente

Print:



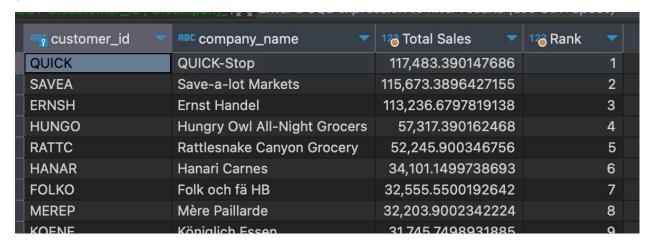
Respuesta:

select o.ship_country as country , o.order_id , o.shipped_date , o.freight ,
sum(o.freight) over (partition by o.ship_country) as totalshippingcosts
from orders o
order by o.ship country, o.order id asc

RANK

12. Ranking de ventas por cliente

Print:



Respuesta:

```
select o.customer_id , c.company_name , sum(od.unit_price * od.quantity ) as "Total Sales",
rank() over (order by sum(od.unit_price * od.quantity) desc )
from orders o inner join order_details od
on o.order_id = od.order_id inner join customers c
on c.customer_id = o.customer_id
group by o.customer_id , c.company_name
```

13. Ranking de empleados por fecha de contratación

Print:

129 employee_id	RBC first_name	RBC last_name	② hire_date ▼	126 Rank ▼
3	Janet	Leverling	1992-04-01	1
1	Nancy	Davolio	1992-05-01	2
2	Andrew	Fuller	1992-08-14	3
4	Margaret	Peacock	1993-05-03	4
5	Steven	Buchanan	1993-10-17	5
6	Michael	Suyama	1993-10-17	5
7	Robert	King	1994-01-02	7
8	Laura	Callahan	1994-03-05	8
9	Anne	Dodsworth	1994-11-15	9

Respuesta:

select e.employee_id , e.first_name , e.last_name , e.hire_date ,
rank() over (order by e.hire_date asc)
from employees e

14.Ranking de productos por precio unitario Print:

17 product_id	•	RBC product_name	•	123 unit_price 🔻	126 Rank	•
	38	Côte de Blaye		263.5		1
	29	Thüringer Rostbratwurst		123.790000916		2
Ī	9	Mishi Kobe Niku		97		3
	20	Sir Rodney's Marmalade		81		4
Ī	18	Carnarvon Tigers		62.5		5
	59	Raclette Courdavault		55		6
	51	Manjimup Dried Apples		53		7
	62	Tarte au sucre		49.299999237		8
	43	Ipoh Coffee		46		9
	28	Rössle Sauerkraut		45.599998474		10

Respuesta:

select p.product_id , p.product_name , p.unit_price ,
rank() over (order by p.unit_price desc)
from products p

LAG

15.Mostrar por cada producto de una orden, la cantidad vendida y la cantidad vendida del producto previo.

Print:

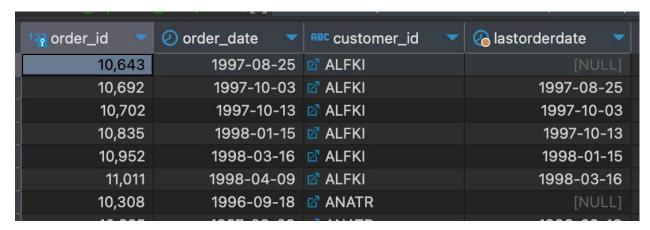
127 order_id 🔻	12 product_id ▼	123 quantity	1% prevquantity
10,248	11 🗹	12	[NULL]
10,248	42 🗹	10	12
10,248	72 🗹	5	10
10,249	14 🗹	9	5
10,249	51 🗹	40	9
10,250	41 🗹	10	40
10,250	51 🗹	35	10
10,250	65 🗹	15	35
10,251	22 🗹	6	15
10,251	57 ☑	15	6
10.051	GE =3	20	45

Respuesta:

select o.order_id , od.product_id , od.quantity ,
lag(od.quantity) over (order by o.order_id) as prevquantity
from orders o inner join order_details od
on o.order_id = od.order_id

16. Obtener un listado de ordenes mostrando el id de la orden, fecha de orden, id del cliente y última fecha de orden.

Print:

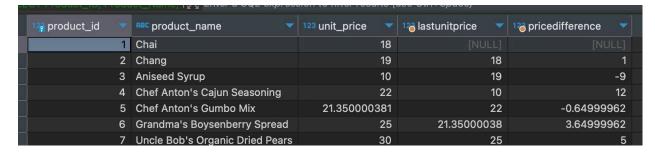


Respuesta:

select o.order_id, o.order_date, o.customer_id,
lag(o.order_date) over (partition by o.customer_id order by o.order_date) as lastorderdate
from orders o

17. Obtener un listado de productos que contengan: id de producto, nombre del producto, precio unitario, precio del producto anterior, diferencia entre el precio del producto y precio del producto anterior.

Print:



Respuesta:

select p.product_id, p.product_name , p.unit_price,
lag(p.unit_price) over (order by p.product_id) as lastunitprice,
p.unit_price - lag(p.unit_price) over (order by p.product_id) as pricedifference
from products p

LEAD

18. Obtener un listado que muestra el precio de un producto junto con el precio del producto siguiente:

Print:

Ctrlu	click to open SQL console	123 unit_price	1% nextprice
1	Chai	18	19
2	Chang	19	10
3	Aniseed Syrup	10	22
4	Chef Anton's Cajun Seasoning	22	21.35000038
5	Chef Anton's Gumbo Mix	21.350000381	25
6	Grandma's Boysenberry Spread	25	30
7	Uncle Bob's Organic Dried Pears	30	40
8	Northwoods Cranberry Sauce	40	97
9	Mishi Kobe Niku	97	31
10	Ikura	31	21
11	Queso Cabrales	21	38
12	Queso Manchego La Pastora	38	6
13	Konbu	6	23.25
14	Tofu	23.25	13
15	Genen Shouyu	13	17.45000076
16	Pavlova	17.450000763	39
17	Alice Mutton	39	62.5

Respuesta:

19. Obtener un listado que muestra el total de ventas por categoría de producto junto con el total de ventas de la categoría siguiente

Print:

RDC category_name	1% totalsales	1% nexttotalsales
Beverages	286,526.9500956535	113,694.7496814728
Condiments	113,694.7496814728	177,099.1006007195
Confections	177,099.1006007195	251,330.4997959137
Dairy Products	251,330.4997959137	100,726.7999253273
Grains/Cereals	100,726.7999253273	178,188.8009858131
Meat/Poultry	178,188.8009858131	105,268.6001739502
Produce	105,268.6001739502	141,623.0891823769
Seafood	141,623.0891823769	[NULL]

Respuesta:

```
select c.category_name ,
sum(od.unit_price * od.quantity) as totalsales,
lead (sum(od.unit_price * od.quantity)) over (order by c.category_name ) as nexttotalsales
--lead (c.category_name ) over (order by c.category_id ) as nexttotalsales
from order_details od inner join products p
on od.product_id = p.product_id inner join categories c
on c.category_id = p.category_id
group by c.category_name
```

Consulta

Para este caso agrupamos por category_name, y comparando con el print de pantalla esperado se ve OK. Pero en la vida real,¿está bien que sea el filtro por name ó es mejor por category_id?