**Ejercicio 4 ‘estirado’ ((sql2))**

SELECT

CASE

WHEN genres LIKE '%Action%' THEN 'Action movies'

ELSE 'Rest of movies'

END AS genres\_cat,

COUNT(movie\_title) AS total\_movies,

ROUND(AVG(imdb\_score),3) AS AVG\_imdb\_score,

SUM(num\_voted\_users) AS total\_votes,

AVG(num\_voted\_users) AS avg\_votes,

SUM(movie\_facebook\_likes) AS total\_likes,

SUM(movie\_facebook\_likes)/COUNT(movie\_title) AS likes\_per\_film,

AVG(movie\_facebook\_likes) AS avg\_likes

FROM imdb\_movies

GROUP BY genres\_cat

Examen SQL-2

Narrativa chatgpt para encontrar una solución recomendada. by Pedro:

“hola chat, eres un profesor y experto en SQL”

“comentame las funciones y comandos basicos de SQL”

“Estoy en un entorno lectivo y de nivel basico de sql. Tengo un problema redactando una query, tengo un bbdd de peliculas, quiero mostrar el conteo de las películas relacionadas con los géneros (Action, Crime, Comedy, Drama, Romance),

indicando la película con mayor número de votos en cada caso (num\_voted\_users).

En la columna de genero, muchas peliculas comparten varios generos, en formato genero/genero/genero, y solo se hace conteo en el primero que sale, pero yo quiero que si una pelicula tiene accion/drama/comedia, por ejemplo, se sume a todas las categorias”

(4)

SELECT (CASE WHEN UPPER(genres) LIKE '%ACTION%' THEN 'Action' END) AS Genero, COUNT(movie\_title) AS num\_movies, movie\_title, MAX(num\_voted\_users) AS num\_voted

FROM imdb\_movies

WHERE UPPER(genres) LIKE '%ACTION%'

UNION

SELECT (CASE WHEN UPPER(genres) LIKE '%CRIME%' THEN 'Crime' END), COUNT(movie\_title) AS num\_movies, movie\_title, MAX(num\_voted\_users) AS num\_voted

FROM imdb\_movies

WHERE UPPER(genres) LIKE '%CRIME%'

UNION

SELECT (CASE WHEN UPPER(genres) LIKE '%COMEDY%' THEN 'Comedy' END), COUNT(movie\_title) AS num\_movies, movie\_title, MAX(num\_voted\_users) AS num\_voted

FROM imdb\_movies

WHERE UPPER(genres) LIKE '%COMEDY%'

UNION

SELECT (CASE WHEN UPPER(genres) LIKE '%DRAMA%' THEN 'Drama' END), COUNT(movie\_title) AS num\_movies, movie\_title, MAX(num\_voted\_users) AS num\_voted

FROM imdb\_movies

WHERE UPPER(genres) LIKE '%DRAMA%'

UNION

SELECT (CASE WHEN UPPER(genres) LIKE '%ROMANCE%' THEN 'Romance' END), COUNT(movie\_title) AS num\_movies, movie\_title, MAX(num\_voted\_users) AS num\_voted

FROM imdb\_movies

WHERE UPPER(genres) LIKE '%ROMANCE%'

Narrativa CHATGPT

(4)

SELECT

CASE

WHEN genres LIKE '%Action%' THEN 'Action\_movie'

WHEN genres LIKE '%Crime%' THEN 'Crime\_movie'

WHEN genres LIKE '%Comedy%' THEN 'Comedy\_movie'

WHEN genres LIKE '%Drama%' THEN 'Drama\_movie'

WHEN genres LIKE '%Romance%' THEN 'Romance\_movie' ELSE NULL END AS Movie\_category

FROM imdb\_movies

GROUP by Movie\_category

HAVING Movie\_category is NOT NULL ;

tengo este codigo, pero no me lo está filtrando bien, porque a la hora de contar los géneros de todas las películas no llega a contarlas todas bien. hay películas que tienen varios generos, por tanto, si una pelicula tiene dos categorías deberá sumar uno a cada categoria, hazlo con union

SELECT

'Action\_movie' AS Movie\_category,

COUNT(\*) AS Movie\_count,

movie\_title,

MAX(num\_voted\_users) AS num\_voted

FROM imdb\_movies

WHERE genres LIKE '%Action%'

UNION

SELECT

'Crime\_movie' AS Movie\_category,

COUNT(\*) AS Movie\_count,

movie\_title,

MAX(num\_voted\_users) AS num\_voted

FROM imdb\_movies

WHERE genres LIKE '%Crime%'

UNION

SELECT

'Comedy\_movie' AS Movie\_category,

COUNT(\*) AS Movie\_count,

movie\_title,

MAX(num\_voted\_users) AS num\_voted

FROM imdb\_movies

WHERE genres LIKE '%Comedy%'

UNION

SELECT

'Drama\_movie' AS Movie\_category,

COUNT(\*) AS Movie\_count,

movie\_title,

MAX(num\_voted\_users) AS num\_voted

FROM imdb\_movies

WHERE genres LIKE '%Drama%'

UNION

SELECT

'Romance\_movie' AS Movie\_category,

COUNT(\*) AS Movie\_count,

movie\_title,

MAX(num\_voted\_users) AS num\_voted

FROM imdb\_movies

WHERE genres LIKE '%Romance%'

Tira en mariaDB::

DROP TABLE IF EXISTS genre\_table;

CREATE TABLE genre\_table(genre VARCHAR(15));

INSERT INTO genre\_table VALUES ('Action');

INSERT INTO genre\_table VALUES('Crime');

INSERT INTO genre\_table VALUES ('Comedy');

INSERT INTO genre\_table VALUES ('Drama');

INSERT INTO genre\_table VALUES ('Romance');

SELECT

genre AS 'Genre',

(

SELECT

movie\_title

FROM imdb\_movies

WHERE genres LIKE CONCAT('%',genre,'%')

ORDER BY num\_voted\_users DESC

LIMIT 1

) AS 'Most voted movie',

(

SELECT

COUNT(\*)

FROM imdb\_movies

WHERE genres LIKE CONCAT('%',genre,'%')

) AS 'Number of movies'

FROM genre\_table

EJERCICIO 5 BY Álvaro L

SELECT

eye\_color,

count(DISTINCT eye\_color) as número\_de\_veces,

homeworld

from star\_wars\_characters\_buena

WHERE eye\_color != "unknown" and homeworld != "NA"

group by homeworld

order by número\_de\_veces DESC

EJERCICIO 7 BY Gracia

SELECT

personal\_status,

SUM(case when class ="good" then credit\_amount end) as otorg,

SUM(case when class = "bad" then credit\_amount end) as no\_otorg

from LoanData

group by 1

**Ejercicios de examen de SQL II corregido**

Anotaciones:

Corrección

Líneas para probar

Cosas que he puesto mal y hay que borrar

Ejercicio 1 - ¿Cuántas películas duran menos de 60 minutos?; ¿Cuántas entre 60 y 100? Y¿Cuántas más de 100?

/\*Select

case

WHEN duration < 60 THEN 'SHORT'

WHEN duration BETWEEN 60 AND 100 THEN 'MIDDLE'

WHEN duration > 100 THEN 'LONG'

END AS 'Duration\_clasification',

Count(movie\_title) as number\_of\_movies

FROM imdb\_movies

WHERE Duration\_clasification is not NULL

group by 1

ORDER by 2 DESC

(Having gasta más recursos de infraestructura, en este ejercicio es mejor tirar del where porque es más eficiente)

Ejercicio 2 - ¿Cuántas películas de acción hay que duren menos de 60 minutos? Haz un listado de las mismas.

Select

movie\_title,

genres

FROM imdb\_movies

where genres like '%Action%' and duration < 60

SELECT

count(movie\_title) as number\_action\_movies,

genres

FROM imdb\_movies

where genres like'%action%' and duration < 60

Ejercicio 3 - ¿Cuál sería el día de menos cotización en una tendencia alcista en el año 2018?; ¿Y la media ese mismo día? – MARIADB

SELECT

date,

EXTRACT(day FROM date) as day,

EXTRACT(month FROM date) as month,

EXTRACT(year FROM date) as year ,

(esto para explorar)

high,

ROUND((high+low)/2,3) AS AVG\_cot

From bitcoin\_daily\_rates\_formatdate

where EXTRACT(year FROM date) = 2018

order by 2 DESC

limit 1

Ejercicio 4 – Mostrar el conteo de las películas relacionadas con los géneros (Action, Crime, Comedy, Drama, Romance), indicando la película con mayor número de votos en cada caso (num\_voted\_users).

Select

case when MAX(num\_voted\_users) then movie\_title end as 'movie\_more\_voted',

Case

WHEN genres like '%action%' THEN 'Action'

WHEN genres like '%crime%' THEN 'Crime'

WHEN genres like '%comedy%' THEN 'Comedy'

WHEN genres like '%drama%' THEN 'Drama'

WHEN genres like '%romance%' THEN 'Romance'

end as 'genres\_2',

count(DISTINCT movie\_title) as num\_movies\_by\_genre,

MAX(num\_voted\_users) AS num\_voted\_genre

from imdb\_movies

WHERE genres\_2 is not NULL

group by genres\_2

order by 3 DESC

WARNING - El case when para cuando la primera condición es encontrada, entonces esos valores caen en esa condición y no bajan a la siguiente condición. Entonces si una película en este caso, pertenece a varios géneros, Case when no lo va a contar. Solo va a contar la 1ª vez

—-------------------- Alternativa de solución —--------------------------------------------------

SELECT

movie\_title,

case when genres like '%Action%' then 'action' end as genre\_2,

SUM(num\_voted\_users) AS num\_total\_votes,

count(DISTINCT movie\_title) as num\_movies\_by\_genre,

MAX(num\_voted\_users) AS num\_voted\_genre

FROM imdb\_movies

WHERE genres LIKE '%action%'

UNION

SELECT

movie\_title,

case when genres like '%Crime%' then 'crime' end as genre\_2,

SUM(num\_voted\_users)AS num\_total\_votes,

count(DISTINCT movie\_title) as num\_movies\_by\_genre,

MAX(num\_voted\_users) AS num\_voted\_genre

FROM imdb\_movies

WHERE genres LIKE '%crime%'

UNION

SELECT

movie\_title,

case when genres like '%Comedy%' then 'comedy' end as genre\_2,

SUM(num\_voted\_users)AS num\_total\_votes,

count(DISTINCT movie\_title) as num\_movies\_by\_genre,

MAX(num\_voted\_users) AS num\_voted\_genre

FROM imdb\_movies

WHERE genres LIKE '%comedy%'

union

SELECT

movie\_title,

case when genres like '%Drama%' then 'drama' end as genre\_2,

SUM(num\_voted\_users)AS num\_total\_votes,

count(DISTINCT movie\_title) as num\_movies\_by\_genre,

MAX(num\_voted\_users) AS num\_voted\_genre

FROM imdb\_movies

WHERE genres LIKE '%drama%'

union

SELECT

movie\_title,

case when genres like '%Romance%' then 'romance' end as genre\_2,

SUM(num\_voted\_users)AS num\_total\_votes,

count(DISTINCT movie\_title) as num\_movies\_by\_genre,

MAX(num\_voted\_users) AS num\_voted\_genre

FROM imdb\_movies

WHERE genres LIKE '%romance%'

order by 4 DESC

Ejercicio 5 - Mostrar el número de personajes que tienen el mismo color de ojos (eye\_color) y el planeta de origen (homeworld).No mostrar color de ojos desconocidos (unknown) ni planetas sin datos/nombre (NA).

SELECT

eye\_color,

homeworld,

count(DISTINCT name) as num\_people\_same\_eye\_color

from star\_wars\_characters\_buena

WHERE eye\_color!= ‘unknown’ AND homeword != is not 'NA'

group by 1,2

order by 3 DESC

Ejercicio 6 - Identificar y calcular el presupuesto de aquellas películas de James Bond que fueron dirigidas por John Glen y protagonizadas por Timothy Dalton.

Select

film,

budget,

box\_office,

actor,

director

from jamesbond\_new

where director like '%john glen%' and actor like '%Timothy Dalton%'

order by 2

Ejercicio 7 - ¿Cuál es el monto de los créditos otorgados y no otorgados según el Status personal?

SELECT

personal\_status,

class,

count (class)

SUM (credit\_amount) AS total\_amount

FROM LoanData

group by 1,2

order by 3 DESC

Otra opción:

Select personal\_status AS estado,

SUM (Case WHEN class =’good’ then credit\_amount END) as ‘importe aprobado’,

SUM (Case WHEN class =’bad’ then credit\_amount END) as ‘importe denegado’

FROM LoanData

Group by personal\_status

Ejercicio 8 - Obtén un listado de las películas de acción con actor protagonista con más de 10000 likes en Facebook y cuyas películas hayan sido valoradas con al menos un 8 en imdb. Todo ello con fechas anteriores a 2012.

Select

movie\_title,

actor\_1\_facebook\_likes,

imdb\_score

FROM imdb\_movies

where imdb\_score >= 8 and actor\_1\_facebook\_likes >10000 AND genres like '%action%' and title\_year < 2012

Ejercicio 9 - Artistas de los ‘80 y 2000, con su discográfica.

SELECT

artist,

label,

release\_year

FROM rolling\_top\_albums\_1

where

(release\_year between 1980 and 1989)

or (release\_year BETWEEN 2000 and 2009)

Order by release\_year

Ejercicio 10 - Queremos saber cuáles son las 20 películas y género al que pertenecen, con mayor presupuesto y beneficio

SELECT

movie\_title,

genres,

budget,

gross

from imdb\_movies

where budget is not NULL and gross is not NULL

order by 3 DESC, 4 DESC (ordenarlo por budget y gross a nivel de negocio no aporta nada, así que dejarlo solo ordenado por budget o solo por gross, aunque lo ponga en el enunciado)

LIMIT 20

Ejercicio 11 - Agregar las siguientes métricas para 10 primeras localizaciones:

• Profits

• Number of company

• Employees

SELECT

location,

sum (profits) AS total\_profits,

count(DISTINCT company) AS count\_companies,

sum (employees) AS total\_employees

FROM fortune

Group by location

ORDER by 1 DESC

(mejor ordenarlo por alguna métrica, pero estas métricas no nos suelen valer para mucho porque estos datos se van a pasar a PowerBI, Python, etc)

limit 10

Ejercicio 12 - Calcular el número de sesiones con Tablet (campo categoría de dispositivo – deviceCategory, registrados con tablet

SELECT

devicecategory**,**

SUM(sessions) as total\_sessions

FROM google\_analytics\_formatdate

WHERE devicecategory like '%tablet%'

GROUP by 2

Ejercicio 13 - ¿Cuál es el día del año 2019 que más usuarios visitaron la web desde Google por el dispositivo móvil? (acordarse de usar mariaDB)

SELECT

users,

date,

devicecategory,

source

From google\_analytics\_formatdate

WHERE Extract (year from date) = ‘2019’ and devicecategory like '%mobile%' and source like '%google%'

[ like '%2019%' (no es seguro, mejor usar EXTRACT) ]

order by 1 desc

limit 1

Ejercicio 14 - Clasificar las películas por su tipología (B/N y Color) y su mayor puntuación en la plataforma.

SELECT

movie\_title,

color,

imdb\_score

from imdb\_movies

Where color is ‘%color%’ or color is ‘%black and white%’

[GROUP by color]

order by imdb\_score DESC

LIMIT 1

Ejercicio 15 - ¿Qué banda noruega de Black Metal fichó por la discográfica “Deathlike Silence Productions” y en qué año?

select

artist, album, label, release\_year

from rolling\_top\_albums\_1

where label like '%Deathlike Silence%' and sub\_metal\_genre like '%black metal%'

**Subconsultas**

En SQL no se pueden hacer dos operaciones seguidas porque es secuencial.

Pero podemos necesitar una tabla que surge de transformar de otra tabla y para hacerlo todo en la misma query se hacen subconsultas. Para ello se genera una tabla temporal. La sintaxis empieza con

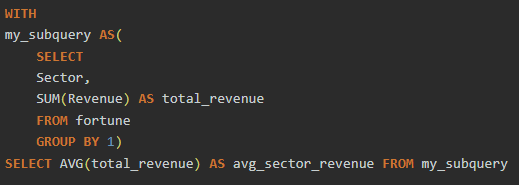
WITH

MYSUB AS ( Select… From),

Mysub2 as (Select…. From)

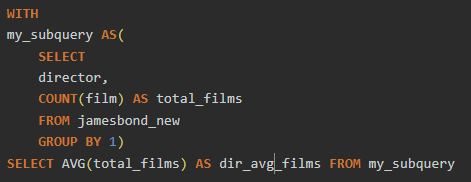
Select … From MYSUB

Separando cada tabla temporal por comas puedo poner las tablas temporales que quiera. Esto puedo utilizarlo para hacer una join o una union

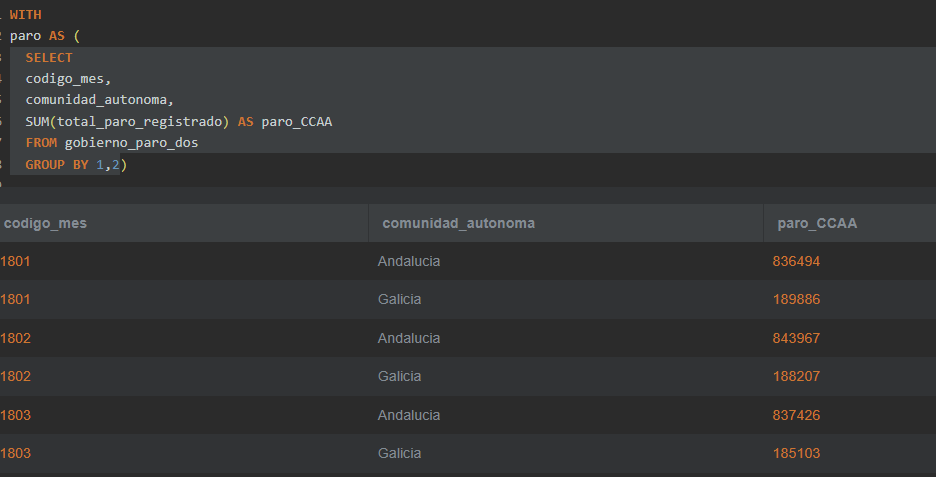
****

Los directores de películas de James Bond han trabajado en promedio en dos películas cada uno.

¿Cómo obtendrías esta información?

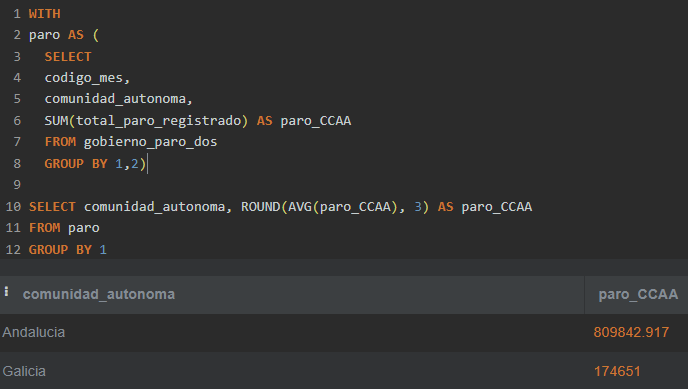
****

**Ejercicio 2.** La tabla gobierno\_paro reporta data mensual de paro para municipios de España en 2018. Calcular la media de paro de cada comunidad autónoma en 2018.



Lo señalado nos muestra la suma de empleo por código mes y por comunidad autónoma.

Ahora seleccionamos de nuestra tabla anterior CCAA, y calculamos el paro de paro\_CCAA de la tabla paro, agrupamos por código mes y por comunidad autónoma, y obtenemos esto:



**Teoría de conjuntos (presentación Cáceres SQL)**

1. Primero identificamos cual es la columna que relaciona ambas tablas (en este caso cliente ID que se llaman igual en las dos tablas, pero no tienen porqué llamarse igual).

Tipos de join

* Inner join: todas las observaciones que están en tabla 1 y a la vez están en la tabla 2. Ejemplo de presupuesto de boda. (t1: contactos - t2: presupuestos)
* Left Join: toda la tabla 1 y los que están en tabla 1 y 2. Salen las columnas que pongamos en el SELECT
* Right join: lo contrario. Todas las observaciones de tabla 2 y las comunes de tabla 1 (no se suele hacer ya que se suele resolver como una left join)
* Full join: todo el producto cartesiano. Todas las observaciones de tabla 1 y 2

1. Elegimos la tabla principal
2. Identificamos el nombre de los campos que se seleccionan

Cuando hagamos la estructura hacer nombre\_tabla.nombre\_columna. Hacemos la query

Lo que hay después del FROM se entiende como tabla 1, después del INNER JOIN será tabla 2.

SELECT

Tabla1.variable1,

Tabla1.variable2,

Tabla2.variable1,

Tabla2.variable3

FROM tabla1

INNER JOIN tabla2

Tabla2 ON tabla1.variable1=tabla2.variable1

Ejercicio :)

Extraer consolas no discontinuadas y su fecha de lanzamiento (first retail availability). Agregar ventas globales de videojuegos

publicados desde el año 2000. Ordenar según ventas totales de videojuegos (descendiente).

select

tc.console\_name,

tc.first\_retail\_availability,

ROUND(SUM(tg.NA\_sales + tg.EU\_sales + tg.JP\_sales + tg.Other\_sales)) as global\_game\_sales

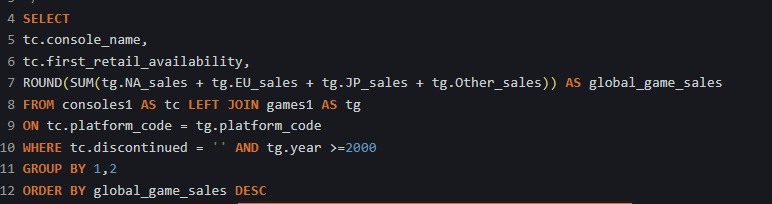
from consoles1 as tc left join games1 as tg

on tc.platform\_code = tg.platform\_code

where tc.discontinued = '' and tg.year >=2000

group by 1,2

order by global\_game\_sales desc



**EJERCICIO FINAL JOINS (carpeta comprimida)**

**(a)**

SELECT

VEN.FECHA AS FECHA,

CLI.NOMBRE AS NOMBRE\_CLIENTE,

CLI.APELLIDO AS APELLIDO\_CLIENTE,

CLI.DNI AS DNI\_CLIENTE,

CLI.CORREO AS CORREO\_CLIENTE,

SUM(VEN.TOTAL) AS TOTAL\_VENTAS

FROM VENTAS VEN INNER JOIN CLIENTES CLI

ON (VEN.ClienteID = CLI.ID)

GROUP BY VEN.Fecha, CLI.NOMBRE, CLI.APELLIDO, CLI.DNI, CLI.CORREO

ORDER BY VEN.Fecha

**(b.i) - mejor solución**

SELECT

VEN.ID AS FACTURA,

PR.CODIGO AS CODIGO\_PRODUCTO,

PR.NOMBRE AS NOMBRE\_PRODUCTO,

SUM(DT.CANTIDAD) AS CANTIDAD\_PRODUCTO

FROM VENTAS S PR

ON (DT.productoid=PR.ID)

GROUP BY 1,2,3

**(b.ii)**

SELECT

VEN.ID AS FACTURA,

VEN INNER JOIN VENTAS\_DETALLE DT

ON (VEN.ID=DT.VENTA\_ID)

INNER JOIN PRODUCTOPR.CODIGO AS CODIGO\_PRODUCTO,

PR.NOMBRE AS NOMBRE\_PRODUCTO,

DT.CANTIDAD AS CANTIDAD\_PRODUCTO

FROM VENTAS VEN INNER JOIN VENTAS\_DETALLE DT

ON (VEN.ID=DT.VENTA\_ID)

INNER JOIN PRODUCTOS PR

ON (DT.productoid=PR.ID)

biz case i

**EJERCICIO 1**

SELECT

p.category,

ROUND(SUM(o.revenue)) AS total\_revenue

FROM orders\_bc o INNER JOIN productos\_bc p

ON o.product\_id = p.product\_id

WHERE EXTRACT(YEAR FROM o.order\_date) = 2022

GROUP BY 1

ORDER BY total\_revenue DESC

EJERCICIOS BASICOS DE REPORTING



SELECT p.product\_id,

COALESCE(SUM(o.quantity),0 ) as quantity

FROM productos\_bc AS p

LEFT JOIN orders\_bc as o

ON p.product\_id = o.product\_id

WHERE p.category = 'home' AND quantity IS NULL

GROUP by 1

Ejercicios comportamiento del cliente

1.

select t1.category, round(avg(t2.order\_id/t3.customer\_id),2) as avg\_orders\_perCustomer

from productos\_bc t1 join orders\_bc t2 on t1.product\_id = t2.product\_id

Join customers\_bc t3 on t2.customer\_id = t3.customer\_id

where t2.order\_date>='2022-01-01'

Group by 1

order by 2 DESC;aaaa

1. **- comportamiento de cliente**

WITH mysub AS(

SELECT

o.customer\_id,

p.category,

COUNT(o.order\_id) AS total\_orders

FROM orders\_bc o INNER JOIN productos\_bc p

ON o.product\_id = p.product\_id

WHERE EXTRACT(YEAR FROM o.order\_date) = 2022

GROUP BY o.customer\_id, p.category

)

SELECT

category,

AVG(total\_orders) AS avg\_orders

FROM mysub

GROUP BY 1

ORDER BY 2 DESC