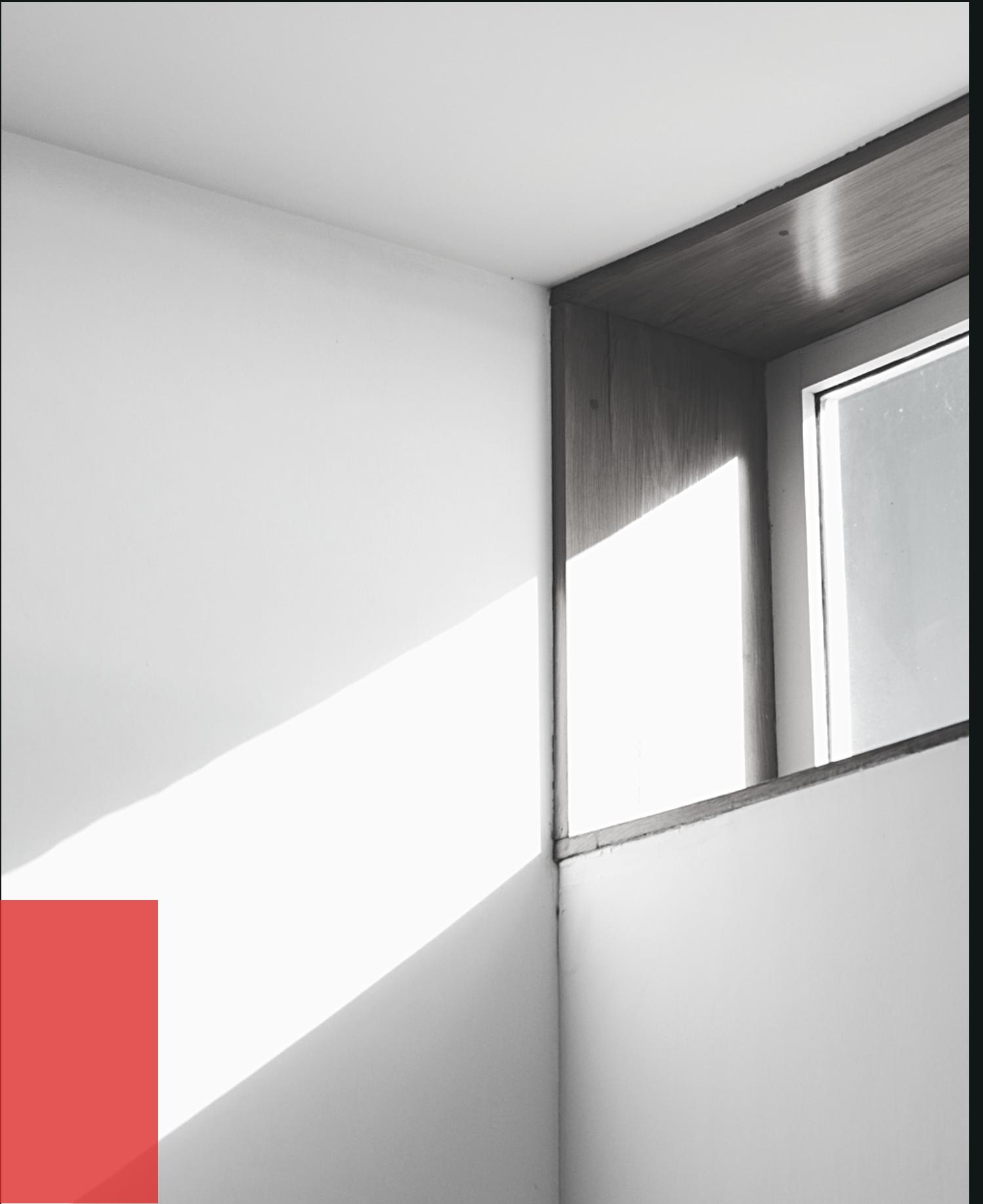


SQL advanced

Some leads for you to keep learning...



1. Tipos de funciones



1. Tipos de funciones

>>>

■ Función escalar

Opera una sola fila, devuelve un solo valor.

■ Función agregada

Obtenemos un total o un promedio para un grupo de elementos con determinada categoría.

■ Función de ventana

No solo obtengo el resultado de la función agregada sino además el detalle de cada fila.

a b c

ES		
ES		
ES		
NO	NO	NO
NO	NO	NO
NO	NO	NO
UK	UK	UK
UK	UK	UK
UK	UK	UK

Resultados
de la función
agregada

a

ES	
NO	
UK	

Funciones de agregado

Funciones de ventana

a b c

ES		
ES		
ES		
NO	NO	NO
NO	NO	NO
NO	NO	NO
UK	UK	UK
UK	UK	UK
UK	UK	UK

Resultados
de la función
ventana

a b c

ES		
ES		
ES		
NO	NO	NO
NO	NO	NO
NO	NO	NO
UK	UK	UK
UK	UK	UK
UK	UK	UK

Funciones escalares

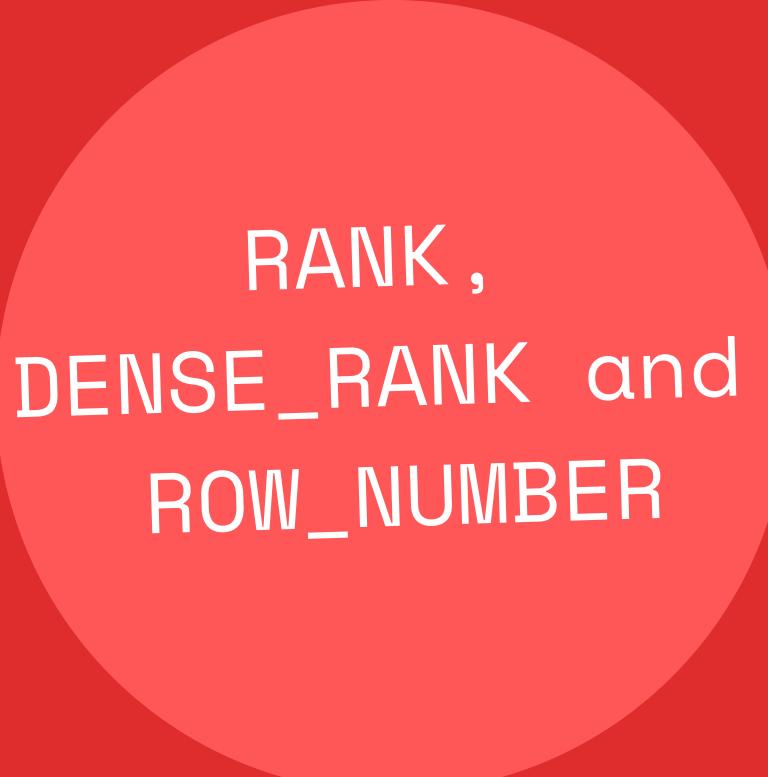
```
SELECT companyname SUBSTRING(companyname , 1 , 5) primeras5letras  
GETDATE() fechahoy  
FROM customers
```

Funciones de agregado

```
SELECT country, COUNT(*) FROM customers  
GROUP BY country
```

Funciones de ventana

```
SELECT c.Categoryname , p.ProductName , p.UnitPrice ,  
RANK() OVER(PARTITION BY c.Categoryname ORDER BY p.UnitPrice DESC)  
AS position  
FROM Products p  
INNER JOIN Categories c  
ON p.Category_ID = c.Category_ID
```



RANK ,
DENSE_RANK and
ROW_NUMBER

■ OVER

Define una ventana o conjunto de filas que debe utilizar la función de ventana, incluyendo cualquier orden salvo ORDER BY clause.

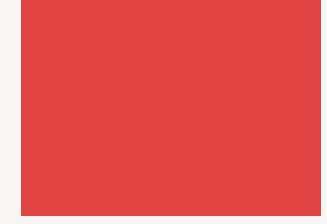
■ PARTITION BY

Con una cláusula de partición de ventana especificada, la cláusula OVER restringe el conjunto de filas a aquellos con los mismos valores en los elementos del subconjunto.

■ ROWS BETWEEN / AND CURRENT ROW

Por sí mismo, OVER no está restringido e incluye todas las filas.





2. Subqueries

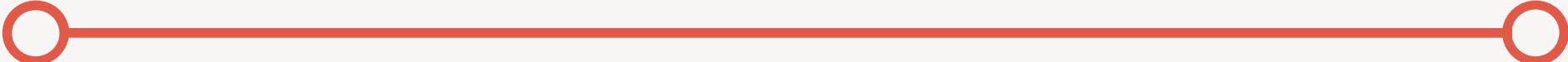




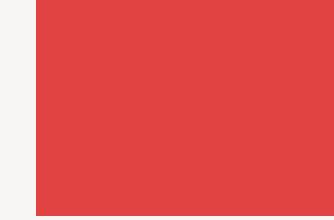
2. Subqueries

Una subconsulta es una consulta anidada en una instrucción **SELECT, INSERT, UPDATE o DELETE**, o bien en otra subconsulta.

```
SELECT *  
FROM invoices i, (SELECT strftime('%Y', InvoiceDate) as year  
FROM invoices) y  
  
WHERE y.year = '2009'
```



Esto nos permite crear una "tabla de forma temporal" y de cara solo a realizar otra u otras queries sobre ella.



3. String formatting



3. String formatting

■ LEFT, RIGHT

obtienes una substring desde el principio o desde el final de la string

■ TRIM

quita caracteres al principio y al final de una string

>>>

■ POSITION

devuelve el número de posición

■ SUBSTR

obtienes una substring dada una posición

■ CONCAT

concatena strings

■ UPPER, LOWER

pasa a mayúscula o minúscula

■ COALESCE

substituye valores nulos



4. Date-time manipulation



4. Date-time manipulation

>>>

■ **EXTRACT**

Extrae una parte de interés de una fecha.

```
SELECT EXTRACT(MONTH  
FROM "2017-06-15");
```

■ **DATEDIFF**

Devuelve un número que es la diferencia en días entre dos fechas.

```
SELECT DATEDIFF("2017-06-25",  
"2017-06-15");
```



5. Operadores de conjunto



5. UNION

UNION por defecto nos mostrará los valores distintos y por tanto, no considerará duplicados.

Para considerar duplicados utiliza UNION ALL.

INTERSECT

El operador INTERSECT mantiene las filas que son comunes a todas las consultas.

```
SELECT City FROM Customers  
UNION  
SELECT City FROM Suppliers  
ORDER BY City;
```



A

1
2
3

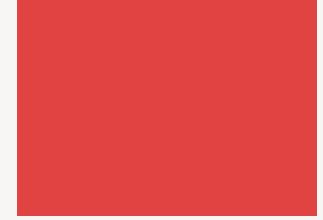
B

3
4
5

A UNION B



1
2
3
4
5



5. HAVING



5. HAVING

Nos permite aplicar condiciones a los datos agregados con GROUP BY.

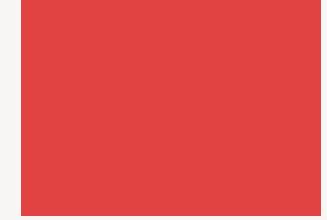
Es el WHERE de los datos agregados.



```
SELECT Employees.LastName, COUNT(Orders.OrderID) AS NumberOfOrders  
FROM Orders  
INNER JOIN Employees ON Orders.EmployeeID = Employees.EmployeeID  
WHERE LastName = 'Davolio' OR LastName = 'Fuller'  
GROUP BY LastName  
HAVING COUNT(Orders.OrderID) > 25;
```

¿Qué estamos pidiendo con esta Query? ¡Voluntarios!





7. Crear base de datos y más

SQL_advanced.ipynb

