Taller de SQL

202601 - ECON 1306

Contenido

- Ordenar datos.

- Funciones de agregación simples:
  - Count
  - Sum
  - Max/Min
  - Avg/Std/Var

- Funciones de agregación más complejas:
  - Group by
  - Having

Manipulación de datos

- Recuperar información específica.
- Personalización y filtrado.
- Resumir datos.
- Ayudan a la toma de decisiones.

Toma de decisiones

Ordenar datos

Ordenar datos

```
SELECT *
FROM cliente
ORDER BY pais ASC
```

cliente				
nombre	**apellido**	**fecha_registro**	**pais**	**ciudad**
Melissa	Robles	2023-07-13	Colombia	Bogotá
Natalia	Perez	2023-07-13	Colombia	Medellín
Manuela	Rojas	2023-07-07	Perú	Lima
Pedro	Sanchez	2023-07-10	Canadá	Montreal
Pablo	Salgado	2023-04-01	Perú	Lima
Rodrigo	Angel	2023-04-10	Colombia	Medellín
Angela	Niño	2023-04-22	Canadá	Vancouver
Salomón	Carmona	2023-02-10	Colombia	Manizales
Verónica	Nuñez	2023-03-22	Perú	Lima
Salomón	Carmona	2023-02-10	Colombia	Bogota

cliente				
nombre	**apellido**	**fecha_registro**	**pais**	**ciudad**
Pedro	Sanchez	2023-07-10	Canadá	Montreal
Angela	Niño	2023-04-22	Canadá	Vancouver
Melissa	Robles	2023-07-13	Colombia	Bogotá
Natalia	Perez	2023-07-13	Colombia	Medellín
Rodrigo	Angel	2023-04-10	Colombia	Medellín
Salomón	Carmona	2023-02-10	Colombia	Manizales
Salomón	Carmona	2023-02-10	Colombia	Bogota
Manuela	Rojas	2023-07-07	Perú	Lima
Pablo	Salgado	2023-04-01	Perú	Lima
Verónica	Nuñez	2023-03-22	Perú	Lima

*** Nota**: Recuerden que la forma de comparar textos es en orden alfabético.

Canadá < Colombia

Ordenar datos

¡El órden importa!

```
SELECT *
FROM cliente
ORDER BY pais ASC, ciudad DESC
```

cliente				
nombre	apellido	fecha_registro	pais	ciudad
Angela	Niño	2023-04-22	Canadá	Vancouver
Pedro	Sanchez	2023-07-10	Canadá	Montreal
Natalia	Perez	2023-07-13	Colombia	Medellín
Rodrigo	Angel	2023-04-10	Colombia	Medellín
Salomón	Carmona	2023-02-10	Colombia	Manizales
Melissa	Robles	2023-07-13	Colombia	Bogotá
Salomón	Carmona	2023-02-10	Colombia	Bogota
Manuela	Rojas	2023-07-07	Perú	Lima
Pablo	Salgado	2023-04-01	Perú	Lima
Verónica	Nuñez	2023-03-22	Perú	Lima

```
SELECT *
FROM cliente
ORDER BY ciudad DESC, pais ASC
```

cliente				
nombre	apellido	fecha_registro	pais	ciudad
Angela	Niño	2023-04-22	Canadá	Vancouver
Pedro	Sanchez	2023-07-10	Canadá	Montreal
Natalia	Perez	2023-07-13	Colombia	Medellín
Rodrigo	Angel	2023-04-10	Colombia	Medellín
Salomón	Carmona	2023-02-10	Colombia	Manizales
Manuela	Rojas	2023-07-07	Perú	Lima
Pablo	Salgado	2023-04-01	Perú	Lima
Verónica	Nuñez	2023-03-22	Perú	Lima
Melissa	Robles	2023-07-13	Colombia	Bogotá
Salomón	Carmona	2023-02-10	Colombia	Bogota

Funciones simples de agregación

Funciones de agregación
COUNT

- La función COUNT() calcula el número de filas que retorna un query.
- Retorna una tabla con un único registro

```
SELECT COUNT(*)
FROM cliente
```

¿Cuántos clientes registrados hay en la base de la tienda?

count(*)
10

```
SELECT COUNT(*) AS colombianos
FROM cliente
WHERE pais = 'Colombia'
```

¿Cuántos clientes colombianos hay en la base de la tienda?

colombianos
5

* El keyword **AS** permite renombrar las columnas.

Funciones de agregación
COUNT y DISTINCT

Ahora, quiero contar el número de países distintos de los clientes de la tienda.

```
SELECT DISTINCT(pais)
FROM cliente
```

¿En cuáles países hay clientes de la tienda?

pais
Colombia
Perú
Canadá

```
SELECT COUNT(DISTINCT(pais)) AS paises
FROM cliente
```

¿En cuántos países hay clientes de la tienda?

paises
3

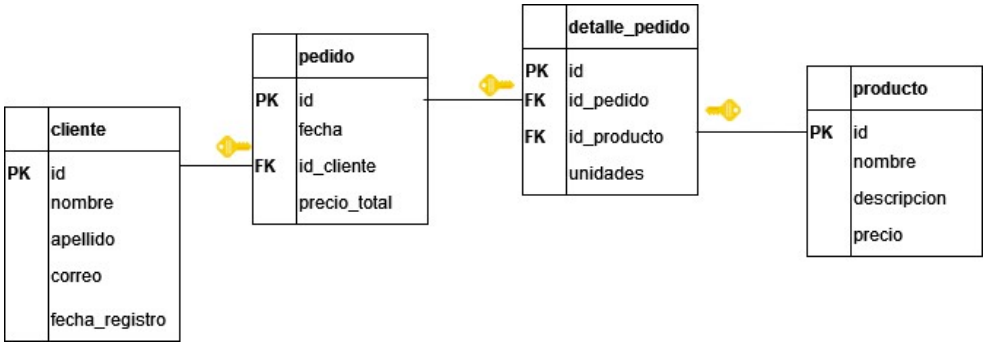
Funciones de agregación
AVG, SUM

```
SELECT AVG(precio) AS avg_precio
FROM producto
```

¿Cuál es el precio promedio de los productos de la tienda?

```
SELECT SUM(precio_total) AS venta
FROM pedido
WHERE fecha = '2023-08-08'
```

¿Cuánto se vendió en la tienda el día 8 de agosto de 2023?



Funciones de agregación

¿Qué pasa con los valores nulos?

A
NULL
1
2
NULL
5

¡Las funciones COUNT y AVG **no** tienen en cuenta los valores nulos!

AVG(A) = (1+2+5)/3 = 8/3 = 2.66

COUNT(**DISTINCT**(A)) = 3

¿Qué hacemos si queremos considerar esos valores dentro de nuestro cálculo?

Promedio ⇨ **SELECT SUM**(A) **/ COUNT**(*)

Para el count distinct necesitamos subqueries

Funciones de agregación

Operaciones entre variables

A	B
NULL	1
1	12
2	5
NULL	NULL
5	NULL

SELECT A*B
FROM table

A*B
NULL
12
10
NULL
NULL

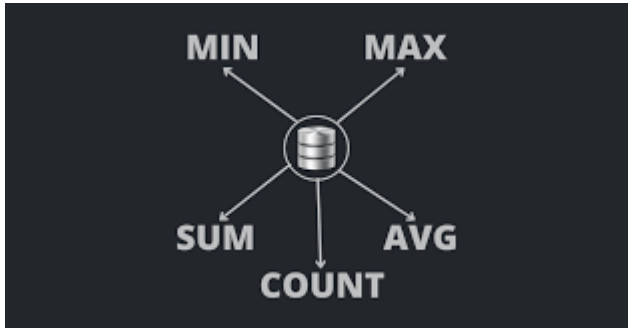
SELECT A+B
FROM table

A+B
NULL
13
7
NULL
NULL

Funciones de agregación

Otras...

- **MIN**: Mínimo de una expresión
- **MAX**: Máximo de una expresión
- **STD**: Desviación estándar
- **VARIANCE**: Varianza
- **LIMIT:** Limita el número de registros que se retornan

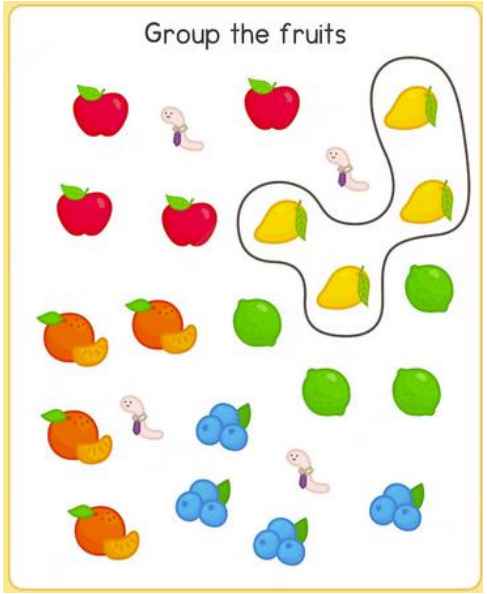


Funciones de agrupación

Agrupación

Necesitamos sacar información agregada de la base:

- Número de clientes por país
- Número de clientes por departamento
- Promedio de ventas por cada producto
- Desviación estándar del promedio acumulado por carrera de los estudiantes



Solución 1 (Número de clientes por país):

- Hacer una consulta por cada país

```
SELECT COUNT(*)
FROM clientes
WHERE pais  = ‘Colombia'
```

Solución 2:

- ¡Agrupar los datos!

Agrupación Ventas totales por día

pedido			
id	id_cliente	fecha	precio_total
1	1	2023-07-13	115
2	2	2023-07-13	120
3	3	2023-07-07	220
4	1	2023-07-10	10
5	1	2023-04-01	20
6	3	2023-04-10	300
7	1	2023-07-10	233
8	4	2023-04-01	12

Separo por fechas

pedido			
id	id_cliente	fecha	precio_total
5	1	2023-04-01	20
8	4	2023-04-01	12
pedido			
id	id_cliente	fecha	precio_total
6	3	2023-04-10	300
pedido			
id	id_cliente	fecha	precio_total
3	3	2023-07-07	220
pedido			
id	id_cliente	fecha	precio_total
4	1	2023-07-10	10
7	1	2023-07-10	233
pedido			
id	id_cliente	fecha	precio_total
1	1	2023-07-13	115
2	2	2023-07-13	120

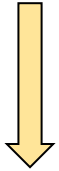
Cálculo el total por día

fecha	venta
2023-04-01	32
2023-04-10	300
2023-07-07	220
2023-07-10	243
2023-07-13	235

Agrupación Ventas totales por día

pedido			
id	id_cliente	fecha	precio_total
1	1	2023-07-13	115
2	2	2023-07-13	120
3	3	2023-07-07	220
4	1	2023-07-10	10
5	1	2023-04-01	20
6	3	2023-04-10	300
7	1	2023-07-10	233
8	4	2023-04-01	12

```
SELECT fecha, SUM(precio_total)
FROM pedido
GROUP BY fecha
```



fecha	venta
2023-04-01	32
2023-04-10	300
2023-07-07	220
2023-07-10	243
2023-07-13	235

Agrupación

SELECT columnas → Columnas que quiero mostrar
FROM tabla
WHERE condición
GROUP BY columnas_agrupacion

pedido			
id	**id_cliente**	**fecha**	**precio_total**
1	1	2023-07-13	115
2	2	2023-07-13	120
3	3	2023-07-07	220
4	1	2023-07-10	10
5	1	2023-04-01	20
6	3	2023-04-10	300
7	1	2023-07-10	233
8	4	2023-04-01	12

SELECT fecha, id_cliente, SUM(precio_total)
FROM pedido
GROUP BY fecha

¿Resultado?

ERROR: Las únicas columnas que puedo pedir son:
- Aquellas sobre las que se está agrupando
- Columnas con funciones de agregación

Agrupación Agrupación de múltiples columnas

pedido			
id	id_cliente	fecha	precio_total
1	1	2023-07-13	115
2	2	2023-07-13	120
3	3	2023-07-07	220
4	1	2023-07-10	10
5	1	2023-04-01	20
6	3	2023-04-10	300
7	1	2023-07-10	233
8	4	2023-04-01	12

id_cliente	fecha	precio_total
1	2023-04-01	20
4	2023-04-01	12
3	2023-04-10	300
3	2023-07-07	220
1	2023-07-10	243
1	2023-07-13	115
2	2023-07-13	120

```
SELECT id_cliente, fecha, SUM(precio_total)
FROM pedido
GROUP BY id_cliente, fecha
```

Agrupaciones Having

Agrupación HAVING

HAVING es otra función de filtro al igual que WHERE.

La diferencia principal es que HAVING puede filtrar sobre las columnas con agrupación:

```
SELECT fecha, SUM(precio_total)
FROM pedido
GROUP BY fecha
HAVING SUM(precio_total) > 10
```

Se consideran únicamente los días que vendieron más de 10$.

¿Diferentes o iguales?

```
SELECT fecha, SUM(precio_total)
FROM pedido
WHERE SUM(precio_total) > 10
GROUP BY fecha
```

ERROR: **WHERE** es un filtro sobre la base original. En la base original no existe una columna SUM(precio_total)

Agrupación

Ejercicio: ¿Qué resultado da el query sobre la siguiente tabla de ventas?

id	producto	categoria	vendedor	cantidad	precio_unitario
1	Producto A	Electrónicos	Vendedor 1	5	200
2	Producto B	Ropa	Vendedor 2	3	50
3	Producto A	Electrónicos	Vendedor 1	2	200
4	Producto B	Ropa	Vendedor 1	4	50
5	Producto A	Electrónicos	Vendedor 2	3	200
6	Producto C	Juguetes	Vendedor 2	6	30
7	Producto A	Electrónicos	Vendedor 1	5	300
8	Producto B	Ropa	Vendedor 2	1	10
9	Producto A	Electrónicos	Vendedor 1	2	20
10	Producto B	Ropa	Vendedor 1	5	5
11	Producto A	Electrónicos	Vendedor 2	1	100
12	Producto C	Juguetes	Vendedor 2	6	35

```
SELECT
        categoria,
        SUM(cantidad) AS total_cantidad_vendida,
        SUM(cantidad * precio_unitario) AS ingreso_total
FROM ventas
WHERE categoria IN ('Electrónicos', 'Ropa')
GROUP BY categoria
ORDER BY ingreso_total ASC;
```

Agrupación HAVING

Paso 1:
Definir variables de la tabla resultado

categoria	total_cantidad_vendida	ingreso_total

Paso 2:
Encontrar los valores de la agrupación

categoria	total_cantidad_vendida	ingreso_total
Electrónicos		
Ropa		

```
SELECT
        categoria,
        SUM(cantidad) AS total_cantidad_vendida,
        SUM(cantidad * precio_unitario) AS ingreso_total
FROM ventas
WHERE categoria IN ('Electrónicos', 'Ropa')
GROUP BY categoria
ORDER BY ingreso_total ASC;
```

Observación: No se puso la categoría 'Juguetes' por el filtro de WHERE.

Agrupación HAVING

Paso 3:
Completar los valores restantes

categoria	total_cantidad_vendida	ingreso_total
Electrónicos	18	3640
Ropa	13	385

Paso 4:
Ordenar

categoria	total_cantidad_vendida	ingreso_total
Ropa	13	385
Electrónicos	18	3640

```
SELECT
        categoria,
        SUM(cantidad) AS total_cantidad_vendida,
        SUM(cantidad * precio_unitario) AS ingreso_total
FROM ventas
WHERE categoria IN ('Electrónicos', 'Ropa')
GROUP BY categoria
ORDER BY ingreso_total ASC;
```

Agrupación HAVING

Ejercicio: ¿Qué resultado da el query sobre la siguiente tabla de ventas?

id	producto	categoria	vendedor	cantidad	precio_unitario
1	Producto A	Electrónicos	Vendedor 1	5	200
2	Producto B	Ropa	Vendedor 2	3	50
3	Producto A	Electrónicos	Vendedor 1	2	200
4	Producto B	Ropa	Vendedor 1	4	50
5	Producto A	Electrónicos	Vendedor 2	3	200
6	Producto C	Juguetes	Vendedor 2	6	30
7	Producto A	Electrónicos	Vendedor 1	5	300
8	Producto B	Ropa	Vendedor 2	1	10
9	Producto A	Electrónicos	Vendedor 1	2	20
10	Producto B	Ropa	Vendedor 1	5	5
11	Producto A	Electrónicos	Vendedor 2	1	100
12	Producto C	Juguetes	Vendedor 2	6	35

```
SELECT
        producto,
        categoria,
        vendedor,
        SUM(cantidad) AS total_cantidad_vendida,
        SUM(cantidad * precio_unitario) AS ingreso_total
FROM ventas
WHERE categoria IN ('Electrónicos', 'Ropa')
GROUP BY producto, categoria, vendedor
HAVING total_cantidad_vendida >= 5 AND ingreso_total >= 200
ORDER BY ingreso_total DESC;
```

Agrupación HAVING

Paso 1:
Definir variables de la tabla resultado

producto	categoría	vendedor	total_cantidad_vendida	ingreso_total

Paso 2:
Encontrar los valores de la agrupación

producto	categoría	vendedor	total_cantidad_vendida	ingreso_total
Producto A	Electrónicos	Vendedor 1		
Producto A	Electrónicos	Vendedor 2		
Producto B	Ropa	Vendedor 1		
Producto B	Ropa	Vendedor 2		

```
SELECT
        producto,
        categoria,
        vendedor,
        SUM(cantidad) AS total_cantidad_vendida,
        SUM(cantidad * precio_unitario) AS ingreso_total
FROM ventas
WHERE categoria IN ('Electrónicos', 'Ropa')
GROUP BY producto, categoria, vendedor
HAVING total_cantidad_vendida >= 5
        AND ingreso_total >= 200
ORDER BY ingreso_total DESC;
```

Observación: No se puso el producto C porque no es de la categoría ‘Electrónicos’ ni ‘Ropa’

Agrupación HAVING

```
SELECT
        producto,
        categoria,
        vendedor,
        SUM(cantidad) AS total_cantidad_vendida,
        SUM(cantidad * precio_unitario) AS ingreso_total
FROM ventas
WHERE categoria IN ('Electrónicos', 'Ropa')
GROUP BY producto, categoria, vendedor
HAVING total_cantidad_vendida >= 5
      AND ingreso_total >= 200
ORDER BY ingreso_total DESC;
```

Paso 3:
Completar los valores restantes

producto	categoría	vendedor	total_cantidad_vendida	ingreso_total
Producto A	Electrónicos	Vendedor 1	14	2940
Producto A	Electrónicos	Vendedor 2	4	700
Producto B	Ropa	Vendedor 1	9	225
Producto B	Ropa	Vendedor 2	4	160

Paso 4:
Filtrar el HAVING

producto	categoría	vendedor	total_cantidad_vendida	ingreso_total
Producto A	Electrónicos	Vendedor 1	14	2940
Producto B	Ropa	Vendedor 1	9	225

Paso 5:
Ordenar

Ordenado por ingreso total descendente