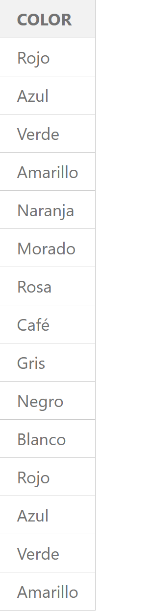
*Agrupando valores con GROUP BY*

La cláusula GROUP BY es una poderosa herramienta en SQL que se utiliza para agrupar filas con valores idénticos en una o varias columnas específicas, permitiendo realizar operaciones de agregación en cada grupo.

En este primer ejercicio aprenderemos a utilizar GROUP BY para obtener todos los elementos distintos de una tabla, lo mismo que previamente hicimos con distinct.

Tenemos la siguiente tabla colores:



Podemos seleccionar los elementos únicos utilizando GROUP BY de la siguiente forma:

SELECT color as color\_unico FROM colores GROUP BY color

Como resultado obtendremos:

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

Ejercicio

Dada la siguiente **tabla de usuarios**



Crea una consulta que nos muestre cada correo una única vez. La columa mostrada debe llamarse correo\_unico

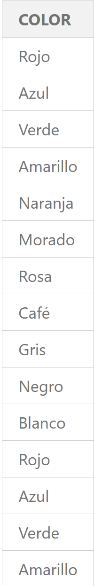
select correo as correo\_unico from usuarios group by correo

se selecciona la columna y luego le colocamos un alias para mostrar el resultado en otra columna, seleccionamos la tabla y con el group by seleccionamos la columna que queremos seleccionar para eliminar los elementos duplicados.

*Agrupar y contar*

*GROUP BY* es comúnmente utilizada junto con funciones de agregación como COUNT, MAX, MIN, SUM y AVG para obtener información resumida de un conjunto de datos.

En este ejercicio aprenderemos a agrupar y contar.

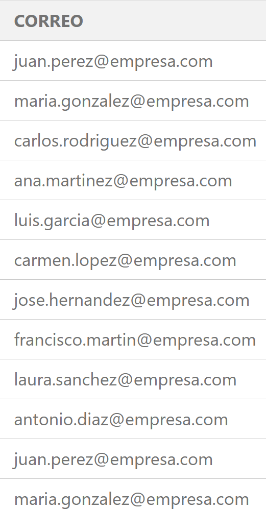
Tenemos la siguiente tabla colores:

Tabla

Descripción generada automáticamenteQueremos saber cuantas veces aparece cada color. Esto lo podemos lograr combinando GROUP BY y la función de agregación COUNT

SELECT color, COUNT(color) as Repeticiones FROM colores GROUP BY color =>

Ejercicio

Dada la siguiente **tabla de usuarios**

Crea una consulta que nos muestre cada correo una única vez junto a la cantidad de repeticiones. Las columnas deben llamarse correo y repeticiones.

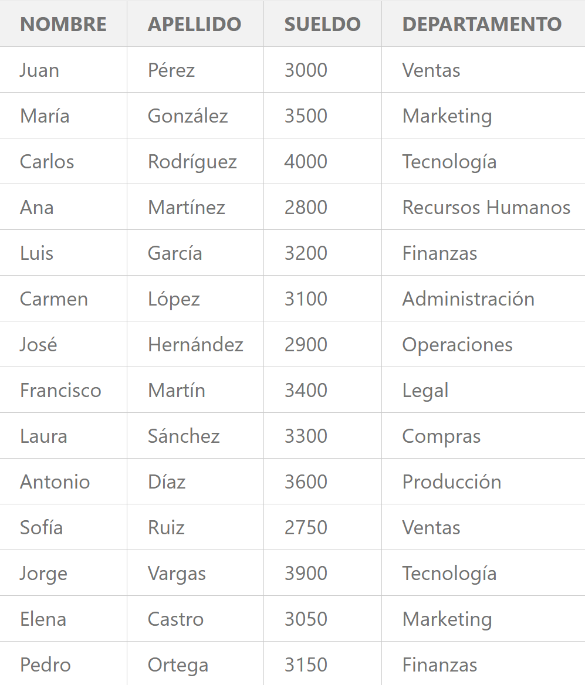
select correo, count(correo) as repeticiones from usuarios group by correo

se selecciona la columna, luego queremos que se cuente desde la misma columna y le damos un alias para que se muestre en una nueva columna y sea mas legible el resultado seleccionamos luego la tabla y por ultimo le decimos que agrupe desde la columna que se selecciono.

*Ejercitando agrupar y contar*

Ejercicio

Dada la siguiente **tabla empleados**



Se pide contar cuantas personas trabajan en cada departamento. Las columnas resultantes deben llamarse departamento y cantidad\_empleados

select departamento, count(departamento) as cantidad\_empleados from empleados group by departamento;

*Agrupar y sumar*

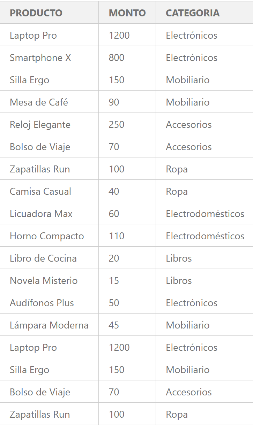
En este ejercicio agruparemos y sumaremos. La lógica de la consulta es la misma previamente mencionada, solo cambia la funcion de agrupacion a utilizar. Por ejemplo, tenemos la tabla pedidos con los siguientes datos:



Si queremos calcular cuanto ha gastado cada cliente, podemos realizar la siguiente consulta

SELECT Cliente, SUM(Monto) AS Monto\_Total FROM pedidos GROUP BY Cliente;

Ejercicio

Utilizando la siguiente **tabla ventas** de una empresa, crea una consulta que muestre cuanto se vendió en total por cada cateogría. Las columnas de la consulta deben llamarse categoria y monto\_total

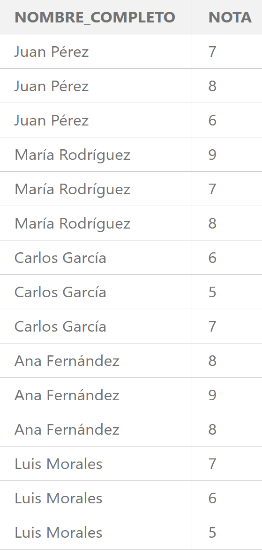
SELECT categoria, SUM(Monto) AS Monto\_Total FROM ventas GROUP BY categoria;

*Agrupar y promediar*

Previamente aprendimos que AVG nos permite calcular el promedio de los elementos de una columna en una tabla. En este ejercicio lo utilizaremos para calcular promedios por grupo.

SELECT grupo, AVG(columna) FROM tabla GROUP by grupo

Ejercicio

Dada la siguiente tabla de estudiantes

Encuentra el promedio de notas de cada estudiante. Las columnas deben tener el nombre de Nombre\_Completo y Promedio\_Notas respectivamente.

Este ejercicio tiene un supuesto importante: que no hay dos estudiantes con el mismo nombre y apellido. Discutiremos este tipo de supuestos más adelante cuando revisemos el concepto de integridad.

SELECT nombre\_completo, AVG(nota) as Promedio\_Notas FROM estudiantes GROUP by Nombre\_completo

Se selecciona la columna que queremos agrupar, luego se pide calcular el promedio y se hacer con la consulta AVG aplicandola al a columna nota y a esta columna se le coloca un alias para mostrarla como una nueva columna para que sea mas legible, luego se selecciona la tabla y por ultimo se agrupan los resultados de la columna seleccionada.

*Máximo por grupo*

En este ejercicio combinaremos la función de agregación MAX() con group by para poder obtener el monto mas alto de cada grupo. La sintaxis de la consulta será igual a las vistas previamente, es decir:

SELECT grupo, MAX(columna) FROM tabla GROUP by grupo

Ejercicio

Dada la siguiente **tabla de ventas**:

Crea una consulta para calcular el monto mas alto por cada categoría. La tabla resultante debe tener dos columnas: categoria y monto\_mas\_alto.

SELECT categoria, MAX(monto) as monto\_mas\_alto FROM ventas GROUP by categoria

Se selecciona la columna que queremos agrupar, luego con MAX se selecciona la columna en donde se hará el calculo y se le asigna un alias para mostrar en otra columna el resultado, luego se selecciona la tabla y por ultimo le decimos con group by que queremos agrupar el resultado.

Mínimo por grupo

En este ejercicio combinaremos la función MIN() con GROUP BY para poder obtener el monto mas bajo de cada grupo.La sintaxis de la consulta será igual a las vistas previamente, es decir:

SELECT grupo, MIN(columna) FROM tabla GROUP by grupo

Ejercicio

Dada la tabla ventas:

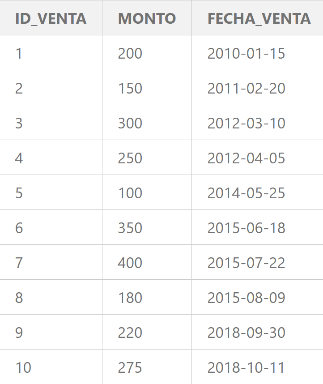
Crea una consulta para calcular el monto más bajo por cada categoría. La tabla resultante debe tener dos columnas: categoria y monto\_mas\_bajo.

select categoria, MIN(monto) as monto\_mas\_bajo from ventas group by categoria;

*Funciones de agregación y fechas*

A la hora de construir informes, frecuentemente necesitaremos entregar información agrupada en un periodo de tiempo. Para lograr esto utilizaremos una combinación de GROUP BY con la función strftime.

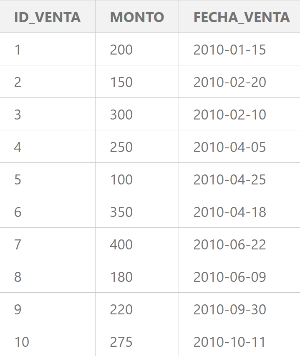
Tenemos la tabla "ventas" con la siguiente información:



Se nos solicita determinar el monto total de ventas por año. Para resolverlo tenemos que agrupar por fecha y sumar los montos de la siguiente forma:

SELECT SUM(monto), strftime("%Y", fecha\_venta) AS año FROM ventas GROUP BY strftime("%Y", fecha\_venta)

Ejercicio

Utilizando esta nueva tabla de ventas.

Calcula el total de ventas por mes. El nombre de las columnas resultantes será "suma\_ventas" y "mes" respectivamente.

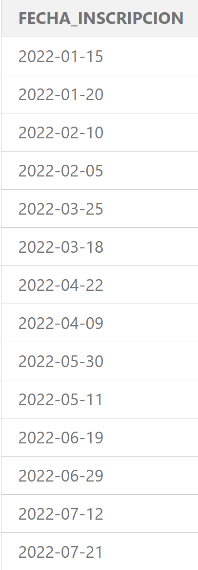
Pista: utiliza la función strftime con %m.

select SUM(monto) as suma\_ventas, strftime('%m',Fecha\_venta)as mes from ventas group by strftime('%m', Fecha\_venta)

primero se hace selección de la columna monto y se suma todo, luego se le agrega un alias para mostrar el resultado en esa columna nueva, luego se escoje lo que se quiere suma en este caso la suma por mes entonces seleccionamos con la sentencia strftime los meses y la columna desde donde queremos seleccionar y agregamos un alias para mostrar la nueva columna resultante y por ultimo se agrupa todo por los meses con group by y la columna donde se encuentra.

*Ejercitando funciones de agregación con fechas*

Ejercicio

Se tiene una tabla llamada inscripciones con distintas fechas de inscripciones de un usuario a un sitio web.

Cuenta cuántos usuarios se registraron cada mes. Las columnas resultantes deben llamarse "mes" y "cantidad\_usuarios".

Tip: Utiliza la función strftime con %m.

select strftime('%m',Fecha\_inscripcion) as mes, count(Fecha\_inscripcion) as cantidad\_usuarios from inscripciones group by strftime('%m',Fecha\_inscripcion)

primero seteamos que queremos seleccionar el mes desde la columna fecha\_inscripcion y le asignamos un alia, luego se cuenta sobre la columna y se le asigna un alias y se selecciona la tabla desde donde se hace la consulta y por ultimo los resultados se agrupan.

*Agrupando sin indicar el nombre de las columnas*

Cuando se trata de agrupar datos en una consulta SQL, existe una forma de evitar la redundancia de la cláusula SELECT. Por ejemplo, considera la siguiente consulta:

SELECT strftime("%Y", fecha\_venta) AS año, SUM(monto) FROM ventas GROUP BY strftime("%Y", fecha\_venta)

Puedes simplificarla de la siguiente manera:

SELECT strftime("%Y", fecha\_venta) AS año, SUM(monto) FROM ventas GROUP BY 1

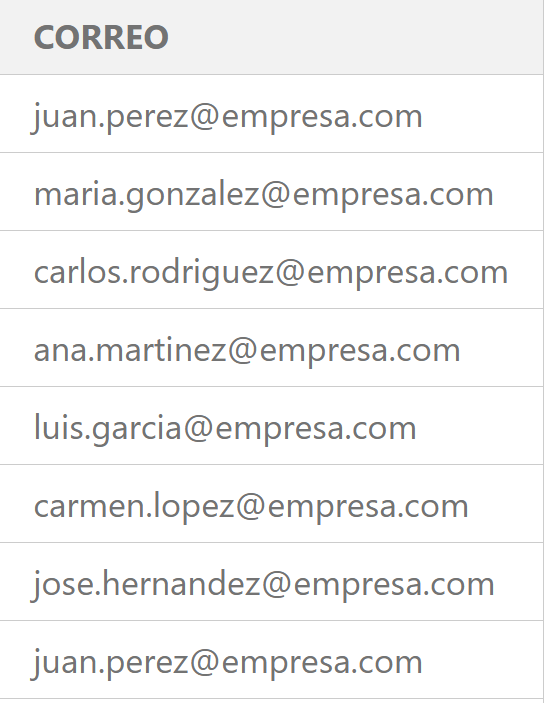
Esta notación se interpreta como "agrupa por el primer criterio". También es posible aplicar esta sintaxis en la cláusula ORDER BY:

SELECT strftime("%Y", fecha\_venta) AS año, SUM(monto) FROM ventas GROUP BY 1 ORDER BY 1

De esta manera, puedes lograr la misma agrupación y ordenamiento sin repetir la expresión de la cláusula SELECT.

Ejercicio

Dada la siguiente tabla de usuarios



Crea una consulta que nos muestre cada correo una única vez acompañado del número de veces que se repite. Las columnas deben llevar los nombres "correo" y "repeticiones", respectivamente, y deben estar ordenadas alfabéticamente.

select correo, count(correo) as repeticiones from usuarios group by 1 order by 1

*Agrupando por múltiples columnas*

En SQL es posible agrupar por múltiples columnas utilizando la siguiente sintaxis:

SELECT columna1, columna2, funcion\_agrupado(columna3) FROM tabla GROUP BY columna1, columna2

Y como aprendimos en el ejercicio anterior, también podemos escribir la consulta de la siguiente manera:

SELECT columna1, columna2, funcion\_agrupado(columna3) FROM tabla GROUP BY 1, 2

Ejercicio

Tenemos la siguiente **tabla estudiantes**

Calcula el promedio de cada estudiante en cada materia. Las columnas deben llamarse correo, materia y promedio\_notas

select correo, materia, avg(nota) as promedio\_notas from estudiantes group by correo, materia o “group by 1, 2”

se selecciona las columnas correo y materia, luego se calcula el promedio de nota y se le asigna un alias para mostrar el resultado en una nueva columna y luego se selecciona la tabla que corresponde las columnas y por ultimo se agrupa para mostrar de forma ordenada a cada estudiante con sus promedios en cada materia.