Consultas de datos

Contents

[Select from 1](#_Toc118390091)

[DISTINCT, WHERE, AS, ORDER BY, TOP, GROUP BY, HAVING 2](#_Toc118390092)

[Operadores logicos 4](#_Toc118390093)

[Operadores relacionales 7](#_Toc118390094)

[Funciones de agregación 10](#_Toc118390095)

# Select from

**La sentencia *select*, como lo indica su nombre, permite seleccionar información a extraer y gracias a esto visualizar el resultado.**

**La cláusula *from* complementa al *select*. Esta declara la tabla desde la cual se va a extraer la información.**

**Realizar Consulta de la Tabla Cliente.** Así vez la tabla “cliente” completa

select \*

from cliente

**Seleccionar ciertas columnas**. Selecciona las columnas “N\_Cliente” y “Sexo” de la tabla clientes

select N\_Cliente, Nombre, Sexo

FROM Cliente;

**Realizar Consulta de la Tabla Cliente a detalle**

SELECT N\_Cliente, Nombre, Sexo

FROM Cliente WHERE Nombre = 'Christian';

Este bloque de código realiza la consulta de algunas columnas de la Tabla Cliente, tomando en cuenta una condición.



DISTINCT, WHERE, AS, ORDER BY, TOP, GROUP BY, HAVING

**La sentencia *Distinct* permite eliminar registros repetidos al obtener el resultado.**

**¡Importante! Tener en cuenta que cada registro recuperado se compone de la *unión* de columnas. En muchas ocasiones podemos creer que son iguales, pero son diferentes**

Table

Description automatically generated

Text

Description automatically generatedTengo dos tablas con los siguientes datos ingresados:



SELECT Distinct Nombre, Sexo Select Distinct \*

FROM Cliente; From Cliente2

SELECT Distinct Nombre, Sexo

FROM Cliente2;

Graphical user interface, table

Description automatically generated with medium confidence

En este caso solo me trae un registro porque identifica que los demás son repetidos.

Te devuelve los datos diferentes para las columnas seleccionadas.

DEPENDE de las COLUMNAS ELEGIDAS

La sentencia ***where*** permite **agregar condiciones** para **filtrar** los resultados. Obtendremos únicamente los registros que cumplan con esas condiciones.

select N\_Cliente, Sexo, Nombre //devuelve los nombres con F

from Cliente2

Where Nombre like 'F%'

select N\_Cliente, Sexo, Nombre

from Cliente2

Where Nombre in ('Fernando')

select N\_Cliente, Sexo, Nombre

from Cliente2

Where N\_Cliente between 100 and 400

Lo que esta en verde son operadores lógicos

SELECT \* FROM Customers  
WHERE Country='Mexico';

SELECT \* FROM Customers  
WHERE CustomerID=1;

Chart

Description automatically generated with medium confidenceLa sentencia *Order By* permite **ordenar el resultado** de la consulta de forma **ascendente o descendente** con respecto a una o varias columnas.

SELECT

N\_Cliente,

ID\_Edad

FROM Cliente2

ORDER BY ID\_Edad asc

Te devuelve TODOS los datos de las columnas N\_cliente e ID\_edad de la tabla cliente2 ORDENADOS de forma ascendente segun el ID\_edad

Los ordenes ASC o DESC se usan para el abecedario también.

La sentencia select top indica la cantidad de registros que quiero que me devuelva

Table

Description automatically generated

SELECT TOP(3)

N\_Cliente,

ID\_Edad

FROM Cliente2

ORDER BY ID\_Edad asc

Idem que antes pero ahora me devuelve SOLAMENTE los tres primeros registros

SELECT TOP(3)\*from cliente

te devuelve los tres primeros registros de la tabla “cliente”

SELECT TOP 3 \* FROM Customers  
WHERE Country='Germany';

Te devuelve los tres primeros registros de la tabla customers donde se cumple al condicion

La sentencia *as* permite **renombrar** el resultado (de forma temporal) de cualquier objeto dentro de la base de datos: campos, tablas, vistas, entre otros.

Table

Description automatically generated

SELECT N\_cliente as id, nombre

FROM Cliente2

WHERE N\_cliente > 400;

Graphical user interface

Description automatically generated with medium confidence

SELECT N\_Cliente + ID\_Edad as Campo

From Cliente2

SELECT N\_Cliente - ID\_Edad as Campo

From Cliente2

SELECT N\_Cliente \* ID\_Edad as Campo

From Cliente2

# Operadores logicos

Table

Description automatically generated

Ejemplos con like

Table

Description automatically generatedSi busco que este cierta palabra entre un texto mas grande pongo lo que busco entre % %. Si

**LIKE**

busco un match exacto con el dato pongo la palabra sola

SELECT AsignaturasID,Nombre

FROM Asignaturas

WHERE Nombre like 'UX writing'

Graphical user interface, table, Excel

Description automatically generated

SELECT AsignaturasID,Nombre

FROM Asignaturas

WHERE Nombre like '%UX%'

Si le pedis de la siguiente manera, te devuelve todas las columnas de los registros que cumplen la condición anterior

select\*from Asignaturas

where Nombre like '%UX%'

Table

Description automatically generated

Otro ejemplo:

select\*from staff

where nombre like 'a%'

#me busca registros donde el nombre empieza con a

select\*from staff

where nombre like 'a%o'

#me busca registros donde el nombre empieza con a y termina con o

Ejemplos con **IN**:

The IN operator allows you to specify multiple values in a WHERE clause, is a shorthand for multiple OR conditions.

Se puede usar not in también.

select Nombre,Apellido,[Codigo Postal]

from Staff

where [Codigo Postal] in ('6414','6960')

select Nombre,Apellido,Documento,Asignatura

from Staff



where

Asignatura in ( SELECT AsignaturasID FROM Asignaturas where nombre like('%UX%'))



Tabla 2

Tabla 1

The BETWEEN operator selects values within a given range. The values can be numbers, text, or dates.

SELECT column\_name(s)  
FROM table\_name  
WHERE column\_name BETWEEN value1 AND value2;

O

SELECT \* FROM Products  
WHERE Price NOT BETWEEN 10 AND 20;

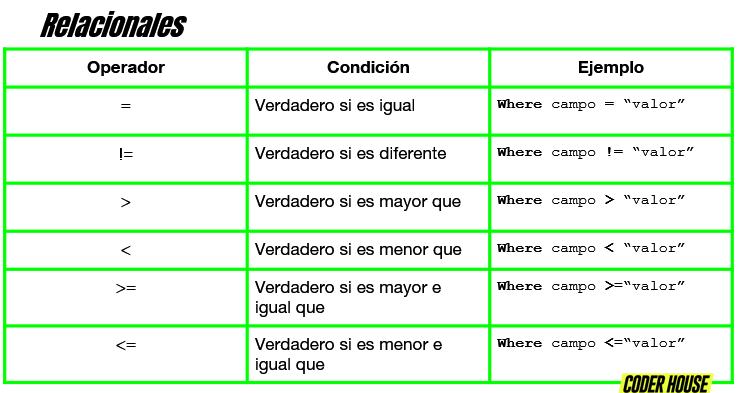
SELECT\*FROM Staff

select upper (concat (nombre,'-' ,apellido)) as Nombres\_apellidos, [Fecha Ingreso]

from Staff

where [Fecha Ingreso] between'2021-01-01' and '2021-12-31'

# Operadores relacionales



EJEMPLOS

Select Distinct

Nombre,

Sexo,

Telefono

From Cliente2

Select Distinct

Nombre,

Sexo,

Telefono

From Cliente2

Where N\_Cliente > 400;

Select Distinct

Nombre as Cliente,

Sexo,

Telefono

From Cliente2

Where N\_Cliente > 400;

# Funciones de agregación

La sentencia count() te devuelve la cantidad de registros de una columna que cumplen o no con cierta condición. Sin condición te devuelve la cantidad de registros nada mas

select count (nombrecolumna)

from nombretabla;

select count (ID\_Edad) as Edad

from Cliente2;

Otro ejemplo. Tengo listado de cursos/carreras de coder. “Asignaturasid” es el numeroID para cada uno de ellos (distingue también según si es turno mañana/tarde/noche).

select count (asignaturasid) as cant\_asignaturas

from asignaturas

where area='5'

select tipo, count (asignaturasid) as cant\_asignaturas



from Asignaturas

where area='5'

Msg 8120, Level 16, State 1, Line 17

Column 'Asignaturas.Tipo' is invalid in the select list because it is not contained in either an aggregate function or the GROUP BY clause.

¿Por qué pasa eso?

select asignaturasid,tipo

from asignaturas

where area='5'

Eso tiene multiples resultados, mientras que count te devuelve un resultado que no se corresponde con ninguna en particular, supongo que va por ahí el error…..

select tipo, count (asignaturasid) as cant\_asignaturas

from Asignaturas



where area='5'

group by tipo

Te devuelve el count ordenado por “tipo”. Como que las dos columnas que estoy mencionando arriba de alguna forma tienen que estar vinculadas para el resultado, no estoy pidiendo simplemente una vista de la columna entera sino que estoy enfoncandome en el count de las asignaturas.

The AVG() function returns the average value of a numeric column.

SELECT AVG(column\_name)  
FROM table\_name  
WHERE condition;

The SUM() function returns the total sum of a numeric column.

SELECT SUM(column\_name)  
FROM table\_name  
WHERE condition;

La sentencia *group by* sirve para agrupar datos según una dada columna (verificar cuando tenga sentido). Permite **agrupar por campos** el resultado de una agregación. *Group by* permite **agrupar por campos** el resultado de una agregación.

Casi siempre que usamos la funcion GROUP BY tiene que haber una FUNCION de AGREGACION. O tener sentido

select \* from Cliente2

SELECT Sexo,COUNT(N\_Cliente)as Cant\_Clientes

FROM Cliente2

GROUP BY Sexo

Table

Description automatically generated

La sentencia *order by* sirve para que te devuelva los resultados en un determinado orden. La sentencia *Order By* permite **ordenar el resultado** de la consulta de forma **ascendente o descendente** con respecto a una o varias columnas.

SELECT COUNT (N\_cliente)as Cantidad, nombre

from Cliente2

Group by nombre

Order by COUNT (N\_CLIENTE) DESC;

Table

Description automatically generated

Las **sentencias** tienen un **orden** para su correcto funcionamiento.

\*\* fijate que cuando uso funciones de agregación en el select, aquella columna que no este dentro de la funcion de agregación, tiene que ir en el group by. Sino tira ERROR.

La sentencia *Having*,como el *where*,permite establecer **condiciones para filtrar los resultados**. **Opera** únicamente sobre **campos** que fueron **generados a partir de una función**, como por ejemplo los registros creados a partir de un *Order by*. The HAVING command is used instead of WHERE with aggregate functions.

SELECT Sexo,COUNT(N\_Cliente)as Cant\_Clientes

FROM Cliente2

GROUP BY Sexo

HAVING COUNT(N\_Cliente)> 3

Idem que antes pero añade un “filtro” mas como condición, es decir, solo quiero ver aquellos valores con cantidad de cliente mayor a 3

Cuantos clientes con el mismo nombre tengo…

SELECT COUNT (N\_cliente)as Cantidad, nombre

FROM Cliente3

GROUP BY nombre

HAVING COUNT (N\_cliente) >= 2

ORDER BY COUNT (N\_CLIENTE) DESC;

solo quiere que devuelva aquellos registros que dieron una cantidad de clientes mayor o igual a 2 y que los ordene de manera descendiente según la cantidad

Otro ejemplo

SELECT COUNT(CustomerID), Country

FROM Customers

GROUP BY Country

HAVING COUNT(CustomerID) > 5;

Background pattern

Description automatically generated

SELECT COUNT(CustomerID), Country  
FROM Customers  
GROUP BY Country  
HAVING COUNT(CustomerID) > 5  
ORDER BY COUNT(CustomerID) DESC;