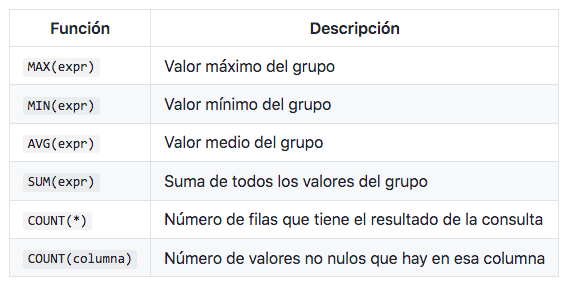
**UD6. SQL II. Realización de consultas**

1. **Consultas de resumen**

En SQL se pueden generar consultas más complejas que resuman cierta información, extrayendo información calculada de varios conjuntos de registros. Para poder generar información resumida hay que hacer uso de las funciones de columna. Estas funciones de columna convierten un conjunto de registros en una información simple cuyo resultado es un cálculo. A continuación, se expone una lista de las funciones de columna disponibles en SQL:



### Diferencia entre COUNT(\*) y COUNT(columna)

* COUNT(\*): Calcula el número de filas que tiene el resultado de la consulta.
* COUNT(columna): Cuenta el número de valores no nulos que hay en esa columna.

**Importante:** Tenga en cuenta la diferencia que existe entre las funciones COUNT(\*) y COUNT(columna), ya que devolverán resultados diferentes cuando haya valores nulos en la columna que estamos usando en la función.

**Ejemplos:**

Supongamos que tenemos los siguientes valores en la tabla alumno:

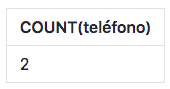


**La consulta:**

SELECT COUNT (teléfono)

FROM alumno;

devolverá:



**Mientras que la consulta:**

SELECT COUNT (\*)

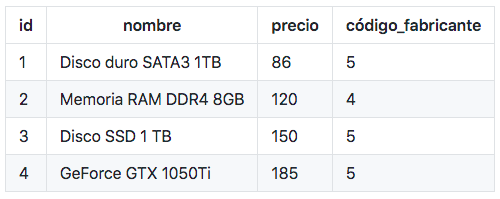
FROM alumno;

devolverá:



### Contar valores distintos COUNT(DISTINCT columna)

Supongamos que tenemos los siguientes valores en la tabla producto:



Y nos piden calcular el número de valores distintos de código de fabricante que aparecen en la tabla producto.

SELECT COUNT (DISTINCT código\_fabricante)

FROM producto;

Esta consulta devolverá:



A modo de ejemplo, se muestran algunas consultas resúmenes en la BBDD “tienda de informática”:

**# Consulta 1**

Devuelve el precio del producto más caro

SELECT MAX(precio)

FROM PRODUCTO;

**# Consulta 2**

Devuelve el precio del producto más barato.

SELECT MIN(precio)

FROM PRODUCTO;

**#Consulta 3**

¿Cuántos productos tiene el fabricante cuyo codigo\_fabricante es igual a 6?

SELECT COUNT(\*)

FROM PRODUCTO

WHERE codigo\_fabricante=6;

**#Consulta 4**

¿Cuál es el precio medio de los productos cuyo codigo\_fabricante es igual a 1?

SELECT AVG(precio)

FROM PRODUCTO

WHERE codigo\_fabricante=1;

1. **Subconsultas**

Las subconsultas se utilizan para realizar filtrados con los datos de otra consulta. Estos filtros pueden ser aplicados tanto en la cláusula WHERE para filtrar registros como en la cláusula HAVING para filtrar grupos. Las agrupaciones las estudiaremos en el siguiente bloque de teoria SQL III.

Los operadores que podemos usar en las subconsultas son los siguientes:

* Operadores básicos de comparación (>,>=, <, <=, !=, <>, =).
* Predicados IN y NOT IN.
* Predicados ALL y ANY.
* Predicados EXISTS y NOT EXISTS.

**Operadores básicos de comparación**

Los operadores básicos de comparación (>,>=, <, <=, !=, <>, =) se pueden usar cuando queremos comparar una expresión con el valor que devuelve una subconsulta.

Los operadores básicos de comparación los vamos a utilizar para realizar comparaciones con **subconsultas que devuelven un único valor, es decir, una columna y una fila**.

Por ejemplo, suponga que queremos conocer el nombre del producto que tiene el mayor precio. En este caso podríamos realizar una primera consulta para buscar cuál es el valor del precio máximo y otra segunda consulta para buscar el nombre del producto cuyo precio coincide con el valor del precio máximo. La consulta sería la siguiente:

**SELECT** nombre

**FROM** producto

**WHERE** precio =

(**SELECT** MAX(precio)

**FROM** producto);

**Predicados IN y NOT IN**

IN y NOT IN nos permiten comprobar si un valor está o no incluido en un conjunto de valores, que puede ser el conjunto de valores que devuelve una subconsulta.

IN y NOT IN los vamos a utilizar para realizar comparaciones con **subconsultas que pueden devolver varios valores, es decir, una columna y varias filas**.

Por ejemplo, una consulta que devuelva los nombres de los productos cuyo nombre de fabricante sea “Asus”:

**SELECT** nombre

**FROM** producto

**WHERE** codigo\_fabricante IN

(**SELECT** codigo

**FROM** fabricante

**WHERE** nombre LIKE ‘Asus’);

**Predicados ALL y ANY**

Los test cuantificados sirven para calcular la relación entre una expresión y todos los registros de la subconsulta (ALL) o algunos de los registros de la subconsulta (ANY).

De esta manera se podría saber cuál es el nombre(s) del producto(s) cuyo precio es mayor que el precio de todos productos cuyo codigo\_fabricante es 2.

**SELECT** nombre

**FROM** producto

**WHERE** precio > ALL

(**SELECT** precio

**FROM** producto

**WHERE** codigo\_fabricante=2);

Un ejemplo del oprador ANY se podría dar si queremos saber cuál es el nombre(s) del producto(s) cuyo codigo\_fabricante=2 y cuyo precio es mayor que el precio de alguno de los productos cuyo codigo\_fabricante es 3.

**SELECT** nombre

**FROM** producto

**WHERE** codigo\_fabricante=2 AND precio > ANY

(**SELECT** precio

**FROM** producto

**WHERE** codigo\_fabricante=3);

**Predicados EXISTS y NOT EXISTS:** *Queda pendiente para cuando estudiemos las consultas multitabla.*