Arquitectura y Administración de bases de datos con SQL 2021

Héctor Manuel Garduño Castañeda

Diciembre, 2021

scidata

Contenido

Se llaman funciones agregadas porque funcionan junto con el SELECT
Como se verá, la sintaxis es la misma:
SELECT COUNT()
SELECT SUM()
SELECT AVG()
SELECT MIN()
SELECT MAX()

El comando **COUNT** sirve para contar el número de registros de una consulta.

SELECT COUNT(⟨⟨nombre de la columna⟩⟩**)**, **COUNT (**⟨⟨nombre de otra columna⟩⟩**)**,...
FROM ⟨⟨nombre de la tabla⟩⟩
(**WHERE** condiciones extra)**;**

+ El COUNT se suele usar con el AS como se muestra en la siguiente diapositiva

Práctica

Contar todos los registros de la tabla *sales*; seleccionar al cliente CG-12520 y responder: ¿cuántas productos ordenó? y ¿en cuántas órdenes los pidió?

¿cuántas ordenes realizó?

SELECT COUNT(∗) **FROM** sales**;**

+ Cuenta todos los registros que tiene la tabla sales

SELECT COUNT(order_line) **AS** ”Número de productos que se ordenaron”,
COUNT(**DISTINCT** order_id) **AS** ”Número de órdenes distintas”
FROM sales **WHERE** customer_id = ’CG-12520’**;**

COUNT(order_line) cuenta el número de renglones de order_line y lo guarda con un alias
COUNT (DISTINCT order_id) cuenta el número de ordenes distintas y ese número lo guarda con un alias
y todo esto lo va hacer
WHERE customer_id = 'CG-12520'

El comando **SUM** sirve para realizar sumas de datos numéricos.

SELECT SUM(⟨⟨expresión⟩⟩**)**
FROM ⟨⟨nombre de la tabla⟩⟩
(**WHERE** condiciones extra)**;**

nombre de alguna columna

+ Podemos pensar que SELECT es para mostrar, entonces la indicación sería como:
muestrame la suma de "columna x"

Práctica

Calcular el total de gananacias por ventas que se tiene; calcular el total de productos Bretford CR4500 Series Slim Rectangular Table que se han vendido.

SELECT SUM(profit) **AS** "Ganancia Total" **FROM** sales**;**

SELECT product_id **FROM** product **WHERE** product_name = 'Bretford CR4500 Series Slim Rectangular Table'**;**
SELECT SUM(quantity) **AS** "Cantidad Total"
FROM sales **WHERE** product_id = 'FUR-TA-10000577'**;**

+ Se saca el product_id de la tabla product y luego esa información se usa para hacer la query de la tabla sales
(más adelante vamos a ver cómo anidar esto, o cómo hacerlo en menos líneas)

El comando **AVG** sirve para realizar promedios de datos numéricos.

SELECT AVG(⟨⟨expresión⟩⟩**)**
FROM ⟨⟨nombre de la tabla⟩⟩
(**WHERE** condiciones extra)**;**

+ Funciona igual que SUM pero este devuelve el promedio

Práctica

Calcular el promedio de edades de los clientes y mostrarlo como "Promedio de edad"; si por cada venta se paga una comisión del 10% al vendedor, ¿Cuánto se tendrá que pagar por venta en promedio? Mostrar el resultado como "Comisión promedio por venta".

SELECT AVG(age) **AS** "Prmedio de edad" **FROM** customer**;**

SELECT AVG(sales*0.10) **AS** "Comisión promedio por venta"
FROM sales**;**

Las funciones **MIN** y **MAX** sirven para hallar máximos y mínimos de columnas.

SELECT MIN(⟨⟨expresión⟩⟩**)**
FROM ⟨⟨nombre de la tabla⟩⟩
(**WHERE** condiciones extra)**;**

SELECT MAX(⟨⟨expresión⟩⟩**)**
FROM ⟨⟨nombre de la tabla⟩⟩
(**WHERE** condiciones extra)**;**

Práctica

Mostrar la menor entrada de dinero por ventas realizadas en junio de 2015 como "Mínimo entrada en junio"; Mostrar la máxima entrada de dinero por ventas realizadas en junio de 2015 como "Máxima entrada en junio". ¿En qué fecha del 2015 se tuvo la mayor entrada de dinero?

SELECT MIN(sales) **AS** "Mínimo de ventas en junio"
FROM sales **WHERE** order_date **BETWEEN** '2015-06-01' **AND** '2015-06-30';

BETWEEN usado con fechas

SELECT MAX(sales) **AS** "Máximo de ventas en junio"
FROM sales **WHERE** order_date **BETWEEN** '2015-06-01' **AND** '2015-06-30';

SELECT MAX(sales) **AS** "Máximo" **FROM** sales **WHERE** order_date **BETWEEN** '2015-01-01' **AND** '2015-12-31';

scidata

+ En el primer ejemplo NO pudimos usar LIKE puesto que la sentencia LIKE es usada para valores de tipo caracter, entonces podemos convertir la fecha en tipo caracter y luego usar LIKE