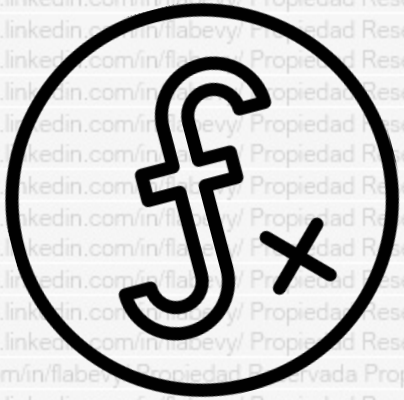
Microsoft®
SQL Server®

Prof. Flavio Bevilacqua
Mail: flabevy88@gmail.com
Linkedin: www.linkedin.com/in/flabevy

FUNCIONES

FUNCIONES INTEGRADAS

- Las funciones integradas en SQL Server son una serie de rutinas almacenadas que reciben una serie de ***parámetros*** con los cuales realizan operaciones concretas para retornar un resultado específico.
- En SQL server se pueden utilizar una serie de ***Funciones Integradas*** que viene incorporadas al motor de BBDD y por otro lado tenemos las ***Funciones Escalares*** las cuales pueden ser construidas por el usuario para re-utilizar scripts y poder ser utilizadas en cualquier momento.



FUNCIONES INTEGRADAS

- Las funciones integradas en SQL Server son una serie de rutinas almacenadas que reciben una serie de ***parámetros*** con los cuales realizan operaciones concretas para retornar un resultado específico.
- En SQL server se pueden utilizar una serie de ***Funciones Integradas*** que viene incorporadas al motor de BBDD y por otro lado tenemos las ***Funciones Escalares*** las cuales pueden ser construidas para poder ser utilizadas en cualquier momento de nuestro código.
- Ejemplos de Funciones Integradas:
  - **HOUR().** Retorna la Hora de un Valor Datetime.
  - **SUM()**. Suma de distintos valores.
  - **MID()**. Extrae valores de un campo tipo texto.
  - **AVG()**. Proporciona la media de una serie de valores.

FUNCIONES INTEGRADAS

- Las funciones integradas en SQL Server, con excepción de las Funciones de Agregación, nos permiten generar en nuestros resultados Columnas Calculadas.
- Las Columnas Calculadas se generan a través de Funciones Integradas como así también, mediante operaciones aritméticas indicadas en el SELECT.
- Las mismas operan a modo horizontal, generando un resultado por cada registro de la tabla, por lo que retornar un conjunto de valores a los efectos del resultado.
- Importante: Las Funciones Integradas no tienen incidencia en la tabla original, siempre y cuando vayan acompañada en una sentencia SELECT.

FUNCIONES DE TEXTO

CONCAT()

- La Función CONCAT es una función que opera sobre cadenas de texto STRING.
- Al igual que sucede con todas las Funciones Integradas, en este caso deberá ir acompañada de parámetros() los cuales podrán tener como argumento dos o mas cadenas juntas, donde haremos referencia los valores a anexar.
- ***CONCAT(argumento1, argumento2, argumento3)***
  - *CONCAT('SQL', ' is', ' fun!');*
  - *CONCAT('SQL', ' ', 'is', ' ', 'fun!');*

LTRIM() <> RTRIM()

- La Función LTRIM() es una palabra reservada, la cual deberá ir acompañada de parámetros() los cuales podrán tener como argumento una cadena de texto tipo String.
- La Función LTRIM() permite remover espacios vacíos ubicadas a la izquierda en un campo de texto. A su vez, existe su variante para el lado derecho RTRIM(), la cual opera del mismo modo pero removiendo espacios vacíos en el margen derecho de la cadena de texto.
- Sintaxis:
  - SELECT LTRIM(' SQL Tutorial') AS LeftTrimmedString;

REPLACE

- La Función LTRIM() es una palabra reservada, la cual deberá ir acompañada de parámetros() sobre los cuales se pasaron 3 argumentos, en todos sus casos tipo VARCHAR.
- REPLACE realiza comparaciones basadas en la intercalación de la entrada. Para realizar una comparación en una intercalación específica.
  - Argumento 1: Expresion de cadena que se buscara, debe hacer referencia a un campo tipo texto.
  - Argumento 2: La cadena de caracteres a remplazar.
  - Argumento 3: La cadena nueva que ingresara en remplazo del argumento 2.
- El siguiente ejemplo remplaza la cadena “cde” por “xxx”, indicado en el segundo y tercer argumento:
  - SELECT REPLACE ('abcdefghicde','cde','xxx’);

SUBSTRING (STR)

- La Función STR() Devuelve parte de una expresión de carácter.
- La Función STR requiere de parámetros, lo cual a través de ellos haremos referencia a los argumentos para que la función se ejecute correctamente.
- SUBSTRING (expresión, start, length)
  - ***Expresion:*** Es una expresion de carácter o texto.
  - ***Start:*** Es el segundo argumento y pasaremos una expresion tipo INT haciendo referencia donde comienzan los caracteres que pretendemos retornar.
  - ***Lenght:*** Es el tercer argumento que pasaremos en los parámetros, siendo este una expresion tipo INT. Este caso se indicara la cantidad de caracteres a retornar, teniendo en cuenta que la "cadena" comenzara en la posición indicada en el argumento nro.2

FUNCIONES DE CONVERSION

CAST()

- Estas funciones convierten una expresión de un tipo de datos a otro.
- La palabra reservada CAST debe acompañarse de parámetros como toda funciones escalar, requiriendo en este caso de dos tipos de argumentos obligatorios.
- ***CAST*** (expression AS datatype)
  - Expression: Cualquier expresión valida que sea necesaria de convertir.
  - Datatype: El tipo de dato al cual pretendemos convertir la expresión que estamos pasando en el primer argumento.
  - La separación entre el primer y el segundo argumento se establece a partir de la palabra ***AS***.

CONVERT()

- Estas funciones convierten una expresión de un tipo de datos a otro.
- La palabra reservada CONVERT debe acompañarse de parámetros como toda funciones escalar, requiriendo en este caso de dos tipos de argumentos obligatorios.
- ***CONVERT*** (datatype, expression)
  - Datatype: El tipo de dato al cual pretendemos convertir la expresión que estamos pasando en el primer argumento.
  - Expression: Cualquier expresión valida que sea necesaria de convertir.
  - CONVERT (varchar, 25.65);
  - CONVERT (datetime, '2017-08-25');

FUNCIONES TIEMPO

DATEDIFF()

- La Funcion ***DATEDIFF()*** es una función de inteligencia de tiempo que opera sobre valores tipo Date or Datetime.
- La Funcion ***DATEDIFF()*** deberá acompañarse de parámetros, donde indicaremos los tres argumentos necesarios para poder operar.
- Dado a que un valor tipo Date puede tener varios intervalos de tiempo expresado, el intervalo será nuestro primer argumento.
- SELECT ***DATEDIFF*** (intervalo, start_date, end_date)
  - ***Intervalo:*** Debemos hacer referencia a la dimensión (Month, Day, Hour, Year, Quarter)
  - ***Start_Date <> End_Date:*** Debemos pasarle campos tipo Date o Datetime en ambos argumentos.

DATEADD()

- La Funcion ***DATEADD()*** es una función de inteligencia de tiempo que opera sobre valores tipo Date or Datetime.
- La Funcion ***DATEADD()*** agrega un intervalo de hora/fecha a una fecha y luego devuelve la fecha
- SELECT ***DATEADD*** (intervalo, numero, campo_date)
  - ***Intervalo:*** Hace referencia a la dimensión de hora/fecha sobre el que se va a operar. (Month, Year, Hour, Day).
  - ***Numero*:** Se pasa un valor tipo INT indicando el intervalo de tiempo a mostrar. Puede ser positivo (para obtener fechas en el futuro) o negativo (para obtener fechas en el pasado)
  - ***Campo_Date:*** Campo o Valor tipo Date or Datetime.

GETDATE()

- La Función ***GETDATE***() devuelve la fecha y hora actual del sistema de la base de datos, en formato 'YYYY-MM-DD hh:mm:ss.mmm’.
- SELECT ***GETDATE();***

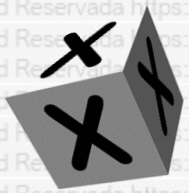
OTRAS FUNCIONES:

- ***Month:*** Sera posible pasar un argumento que haga referencia a un valor tipo Date o Datetime y obtendremos como resultado el numero del Mes.
- ***Year:*** Sera posible pasar un argumento que haga referencia a un valor tipo Date o Datetime y obtendremos como resultado el numero del Año.
- ***Hour:*** *S*era posible pasar un argumento que haga referencia a un valor tipo Date o Datetime y obtendremos como resultado el numero del Hora.

FUNCIONES DE AGREGACION

FUNCION DE AGREGACION

- Las Funciones de Agregación operan a modo vertical, sobre un campo especifico resumiendo un conjunto de valores a un único valor.
- En consecuencia la expresión que pasaremos como parámetros será el nombre de un columna.



FUNCION AGREGACION

- SUM
- COUNT
- MAX
- MIN
- AVERAGE
- COUNTROWS
- PRODUCT
- SUMX
- AVERAGEX

COMO LAS TABLAS DINAMICAS

- Las tablas dinámicas utilizan ***Funciones de Agregacion*** cuyos cálculos se adaptan al contexto.
- Con excepción del COUNT, las funciones de agregación trabajan sobre campos numéricos.
- En caso que nosotros deseemos "aperturar" este único valor en base a un contexto, utilizaremos un campo dimensional en una sentencia ***Group By.***

Provincia	Total Exportaciones
salta	$166.507.487.176.589.000
formosa	$160.812.754.161.228.000
Santa Fe	$132.633.474.361.183.000
La Rioja	$129.013.680.407.484.000
pba	$127.750.145.996.932.000
caba	$122.678.740.361.902.000
misiones	$109.359.181.556.313.000
cordoba	$105.565.001.421.621.000
La Pampa	$101.552.411.139.637.000
chaco	$100.585.720.740.934.000
mendoza	$96.164.136.573.805.600
San Juan	$96.141.591.123.344.400
neuquen	$94.659.523.512.661.900
San Luis	$84.464.730.515.251.600
santiago	$82.022.162.572.091.800
santa	$74.026.129.357.079.700
Entre Rios	$73.525.643.228.447.100
tucuman	$68.050.994.982.524.100

GROUP BY

GROUP BY

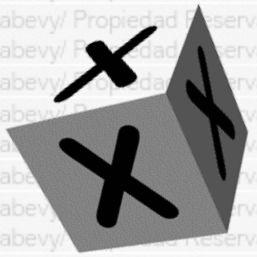
- La cláusula ***GROUP BY*** te permite organizar las filas de una consulta en grupos. Los grupos están determinados por las columnas que se especifican en la cláusula GROUP BY , siendo este el campo dimensional.
- En consecuencia la expresión indicada en el ***GROUP BY*** establece el contexto de nuestra tabla resultante.
- Sobre el comando ***GROUP BY*** haremos referencia a aquellos campos indicados en el SELECT que ***NO*** están acompañados por una ***Función de Agregación***.

```
SELECT COUNT(CustomerID), Country
FROM Customers
GROUP BY Country;
```

Se agrupan la "cantidad" de clientes por País

HAVING

HAVING



- La cláusula ***HAVING*** a menudo se utiliza en compañía de la clausula ***GROUP BY*** para filtrar nuestros resultados.
- Sintácticamente, siempre procede al comando GROUP BY y siempre hace referencia a la Columna Calculada sobre la cual deseamos aplicar un ***FILTRO***.
- Nótese que su funcionalidad es similar al ***WHERE*** pero su diferencia radica en que el Having opera sobre Columnas Calculadas siendo estas parte del resultado pero NO de la tabla original.

```
SELECT Websites.name, SUM(access_log.count) AS nums FROM Websites
INNER JOIN access_log
ON Websites.id=access_log.site_id
WHERE Websites.alexa < 200
GROUP BY Websites.name
HAVING SUM(access_log.count) > 200;
```

FIN