

|  |
| --- |
| Administración de Bases de Datos |
|  |
| Edicion AGOSTO 2021  Instituto de Formación Técnica Superior Nro. 11  Docente: Lic. Norberto Orlando |

**UNIDAD 4: Clausulas que acompañan a las consultas en SQL**

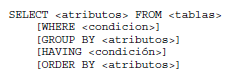
* Consulta basica
* Tipos de Operadores
* Ordenamiento
* Valor NULL
* Función DISTINCT
* Operador UNION
* Expresión CASE
* Agrupaciones, Funciones Agregadas y Resúmenes
* Recuperar Datos Agrupados
* Filtrar Datos Agrupados

Al finalizar la clase será capaz de

* Entender los conceptos básicos de Bases de Datos
* Dominar los distintos tipos de operadores
* Ordenar conjuntos de datos
* Manejar valores NULL
* Manipular información obteniendo un conjunto de resultado con diferentes valores.
* Obtener varios conjuntos de resultados en una misma consulta .
* Manipular valores como así también el ordenamiento de los resultados.
* Evaluar argumentos en orden con valores Null
* Generar resúmenes con funciones agregadas y agrupaciones

SELECT

La sentencia Select es una sentencia SQL, que pertenece al conjunto del Lenguaje de Manipulación de Datos y que sirve para recuperar registros de una o varias tablas, de una o varias bases de datos. Su sintaxis es la siguiente:



Donde las mayúsculas representan palabras reservadas y lo encerrado entre corchetes es opcional, puede ser omitido. Una vez vista la anterior forma de representación, vamos a detenernos en la sintaxis de la sentencia Select.

Se compone de tres partes:

* SELECT <atributos>: permite hacer una proyección de las tablas, es decir, seleccionar los campos que deseamos recuperar de la base de datos, separados por comas. Si se especifica el símbolo \*, se obtendrán todos los campos de la tabla.
* FROM <tablas>: permite especificar la tabla de la cual se desean obtener los datos.
* WHERE <condición>: permite establecer una condición de recuperación de las filas de la/s tabla/s. Sólo se obtendrán aquellas filas que verifiquen dicha condición, que será opcional. En el caso de que se omita esta parte, se recuperarán todas las filas.
* GROUP BY <atributos>: permite establecer una selección de campos cuando se utilizan funciones escalares o de conteo.
* HAVING <condición>: establece una condición para los atributos obtenidos como resultado de la aplicación de funciones escalares.
* ORDER BY <atributos>: permite obtener el resultado de la consulta ordenado por los atributos especificados.

Comentarios

Los comentarios se pueden insertar en una línea independiente. El servidor no evalúa el comentario.

Sintaxis

-- Aquí un Comentario

Comentario de bloque

Indica texto proporcionado por el usuario. El servidor no evalúa el texto situado entre /\* y \*/.

Sintaxis

/\* Aquí

un

Comentario

\*/

Alias y Literales

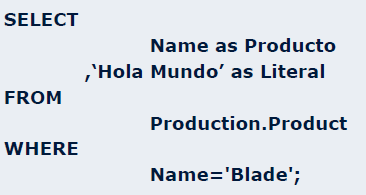
Hay dos tipos de alias que se utilizan con mayor frecuencia. Alias de columna y alias de tabla.

Los alias de columna existen para ayudar en la organización del resultado. El uso de un alias de columna haría el resultado mucho más legible.

El segundo tipo de alias es el alias de tabla. Esto se alcanza al colocar un alias directamente luego del nombre de tabla en la cláusula FROM.

Sintaxis

En el siguiente ejemplo renombra la columna Name por Producto.

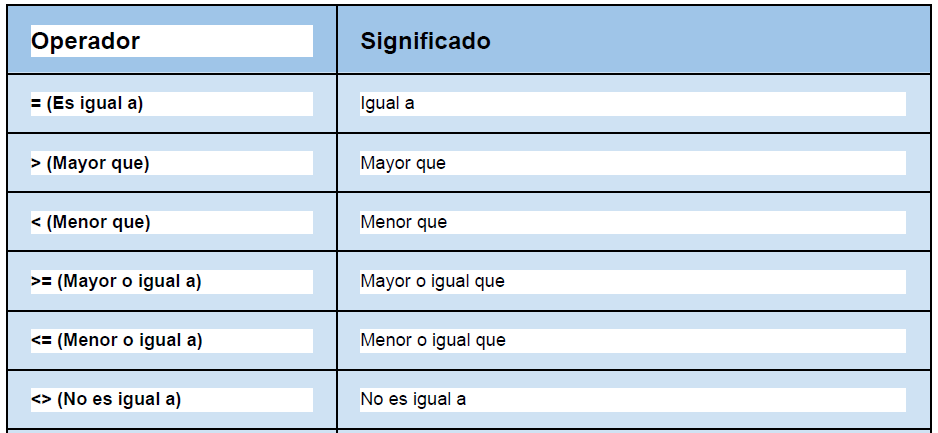


Uso de Operadores

Operadores de comparación

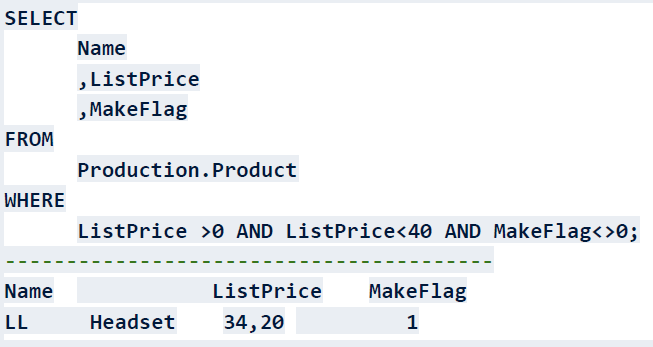
Los operadores de comparación comprueban si dos expresiones son iguales. Se pueden usar en todas las expresiones, excepto en las de los tipos de datos text, ntext, image.

En la siguiente tabla se presentan los operadores de comparación Transact-SQL.



Sintaxis

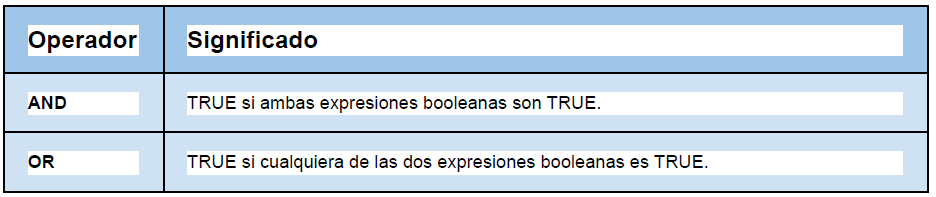
En el ejemplo siguiente hace uso de los operadores mayor menor y distinto.



Operadores lógicos

Los operadores lógicos comprueban la veracidad de alguna condición. Al igual que los operadores de comparación, devuelven el tipo de datos Boolean con el valor TRUE, FALSE.

Operador Significado



En el ejemplo siguiente hace uso del operador OR.

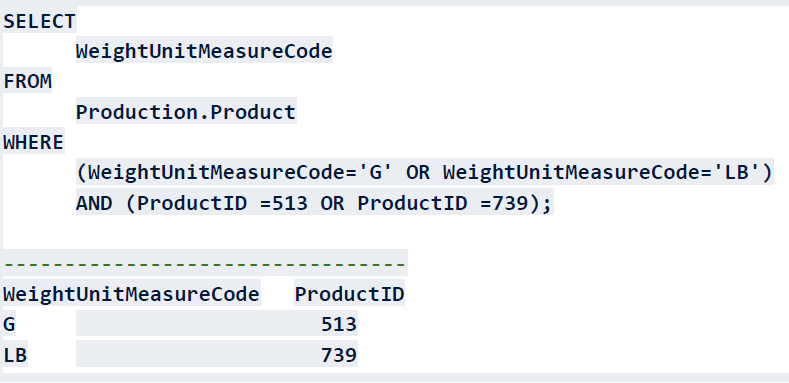
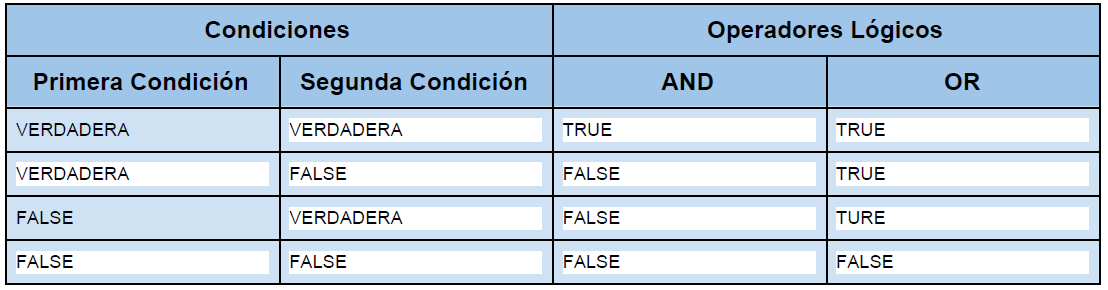
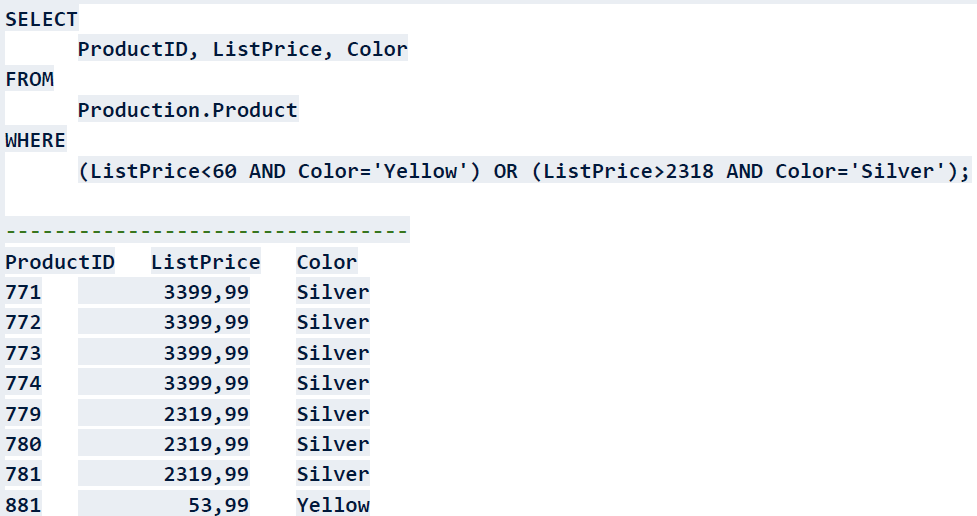


Tabla de verdad

Los operadores lógicos comprueban la veracidad de alguna condición. Al igual que los operadores de comparación, devuelven el tipo de datos Boolean con el valor TRUE, FALSE.



En el ejemplo siguiente hace uso de los operadores AND y OR combinados

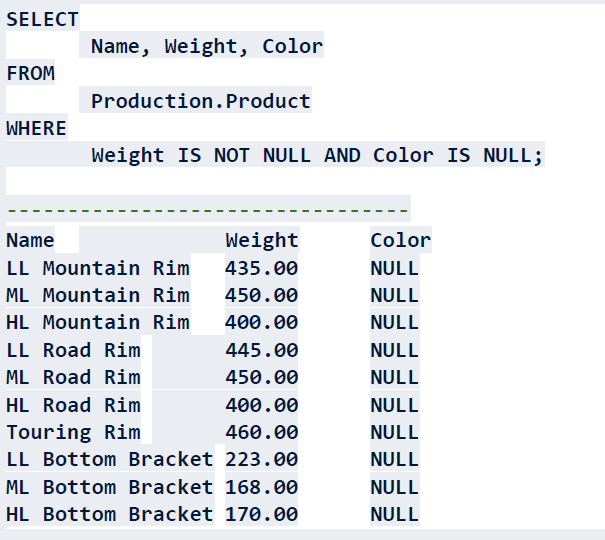


Controlar Valores [Not] Null

Los valores NULL se utilizan en bases de datos relacionales cuando el valor de una columna se desconoce o falta. Un NULL no es ni una cadena vacía (en los tipos de datos de caracteres o de fecha y hora) ni un valor cero (en los tipos de datos numéricos). La especificación ANSI SQL-92 afirma que un NULL debe ser igual en todos los tipos de datos; por lo tanto, todos los NULL se tratan de forma coherente.

Sintaxis

En el siguiente ejemplo se buscan las filas con los productos que tengan establecido el peso y no el color.



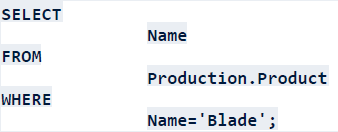
Criterios de Selección

WHERE

La cláusula WHERE se utiliza cuando se desea recuperar información específica de una tabla sin incluir otros datos irrelevantes. Asi SQL ofrece una característica llamada cláusula WHERE, que podemos utilizar para restringir los datos que se recuperan. La condición que usted proporciona en la cláusula WHERE filtra las filas recuperadas y le da solo las filas que usted esperaba ver. La cláusula WHERE se puede utilizar junto con SELECT, DELETE, UPDATE.

Sintaxis

En el ejemplo siguiente se selecciona una lista con los nombres de los productos que son espadas.



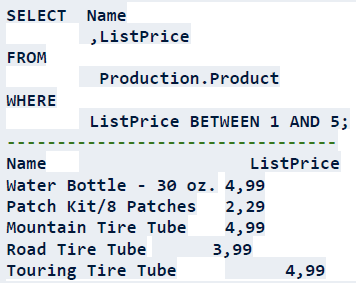
BETWEEN

Especifica un intervalo que se va a probar.

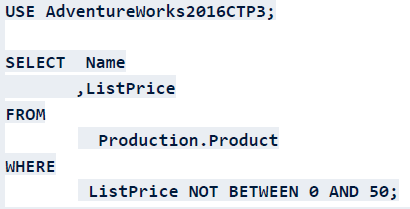
La cláusula BETWEEN especifica un rango de búsqueda. Para indicar que deseamos recuperar los registros según el intervalo de valores de un campo.

Sintaxis

En el ejemplo siguiente se selecciona una lista con los nombres y precio de lista de los productos cuyo precio se encuentre comprendido en el rango de cero a cincuenta pesos.



En el ejemplo siguiente se selecciona una lista con los nombres y precio de lista de los productos cuyo precio no se encuentre comprendido en el rango de cero a cincuenta pesos.

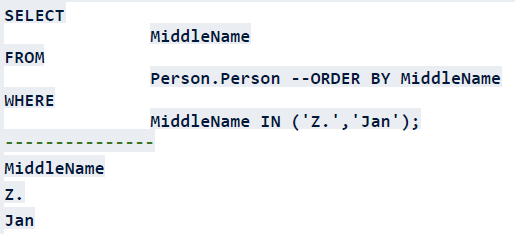


IN

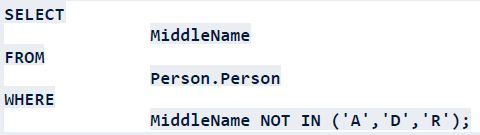
Determina si un valor especificado coincide con algún valor de una subconsulta o una lista. La cláusula IN especifica un rango de búsqueda. Este operador devuelve aquellos registros cuyo campo indicado coincide con alguno de los en una lista.

Sintaxis

En el ejemplo siguiente se selecciona una lista de personas cuyo segundo nombres comiencen con las iniciales [Z., Jan].



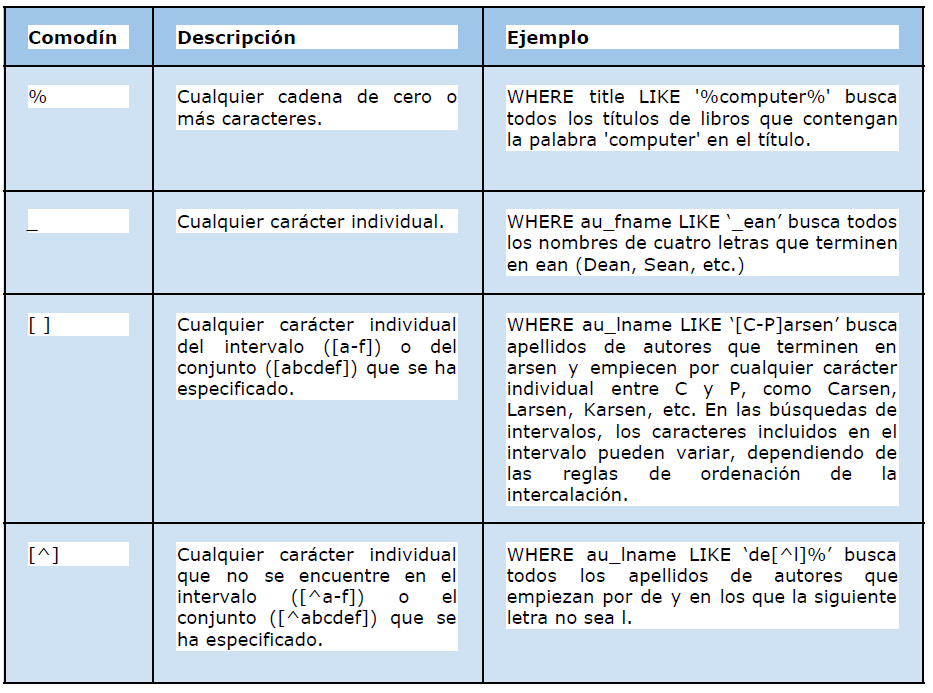
En el ejemplo siguiente se selecciona una lista de personas cuyo segundo nombres no comiencen con las iniciales [A, D, R].



LIKE

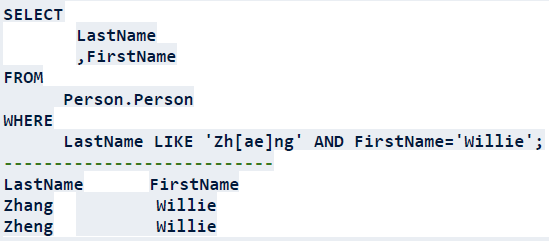
Determina si una cadena de caracteres específica coincide con un patrón especificado. Un patrón puede contener caracteres normales y caracteres comodín. Durante la operación de búsqueda de coincidencias de patrón, los caracteres normales deben coincidir exactamente con los caracteres especificados en la cadena de caracteres. Sin embargo, los caracteres comodín pueden coincidir con fragmentos arbitrarios de la cadena. El uso de caracteres comodín hace que el operador LIKE sea más flexible que los operadores de comparación de cadenas = y !=.

Si alguno de los argumentos no es del tipo de datos de cadena de caracteres, Motor de base de datos de SQL Server lo convierte al tipo de datos de cadena de caracteres, si es posible.

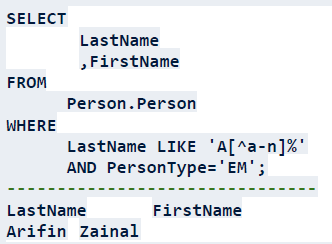


Sintaxis

En el siguiente ejemplo se buscan las filas con empleados en la tabla Person con el apellido de Zheng o Zhang.



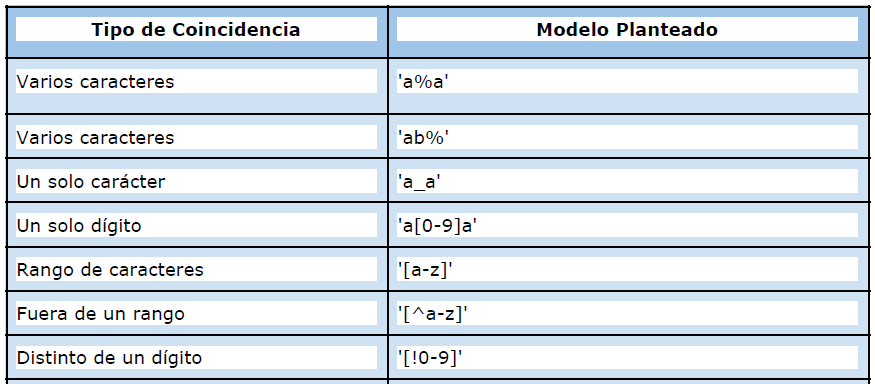
En el siguiente ejemplo se buscan las filas con empleados en la tabla Person con el apellido que comience con A y que no tenga desde la A a la N como segundo carácter y el resto cualquier carácter.



Utilizar caracteres comodín como literales

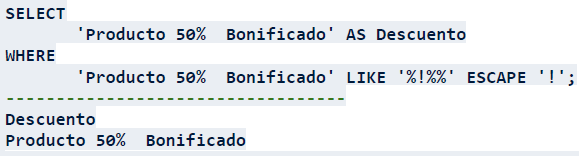
Los caracteres comodín que se utilizan en la operación de búsqueda de coincidencias de patrón se pueden utilizar como literales.

Para utilizar un carácter comodín como literal, inclúyase entre corchetes. La tabla siguiente muestra varios ejemplos del uso de la palabra clave LIKE y los caracteres comodín [ ].



Operación de búsqueda de coincidencias de patrón con la cláusula ESCAPE

Para buscar el signo de porcentaje como carácter en lugar de como carácter comodín, debe suministrarse la palabra clave ESCAPE y el carácter de escape. Supongamos que tenemos una columna denominada comment que contiene el texto 30%. Para buscar filas que contengan la cadena 30% en cualquier parte de la columna de comentarios, especifique una cláusula WHERE como esta:



Ordenamiento

ORDER BY

Ordena los datos devueltos por una consulta en SQL Server.

Use esta cláusula para lo siguiente:

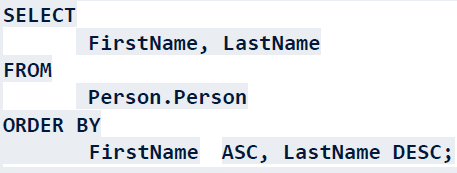
Ordenar el conjunto de resultados de una consulta por la lista de columnas especificada y, opcionalmente, limitar las filas devueltas a un intervalo especificado. El orden en que se devuelven las filas en un conjunto de resultados no se puede garantizar, a menos que se especifique una cláusula ORDER BY.

Características

* Especifica el orden de los resultados de la sentencia SELECT
* Ordena los resultados de una consulta.
* Se puede realizar el ordenamiento por una o más columnas
* Esta cláusula es inválida para las vistas
* Se puede determinar que el orden sea Ascendente o Descendente

Sintaxis

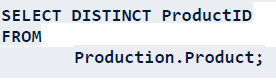
En el siguiente ejemplo se buscan las personas ordenadas por Nombre y luego por Apellido.



SELECT DISTINCT

Devuelve todos los estados posibles para la columna seleccionada del modelo. Los valores devueltos varían dependiendo de si la columna especificada contiene valores discretos, valores numéricos de datos discretos o valores numéricos continuos.

Sintaxis



Operador de conjunto UNION

Combina los resultados de dos o más consultas en un solo conjunto de resultados que incluye todas las filas que pertenecen a las consultas de la unión. La operación UNION es distinta de la utilización de combinaciones de columnas de dos tablas.

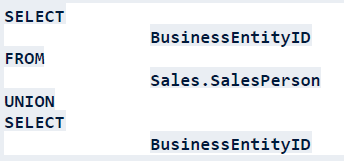
A continuación se muestran las reglas básicas para combinar los conjuntos de resultados de dos consultas con UNION:

* El número y el orden de las columnas debe ser el mismo en todas las consultas.
* Los tipos de datos deben ser compatibles.

UNION

Específica que se deben combinar varios conjuntos de resultados para ser devueltos como un solo conjunto de resultados.

Sintaxis

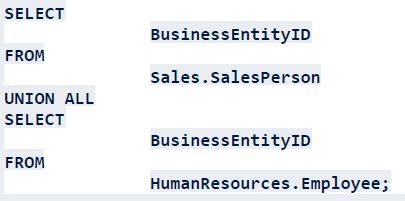




UNION ALL

Agrega todas las filas a los resultados. Incluye las filas duplicadas. Si no se especifica, las filas duplicadas se quitan.

Sintaxis



CASE

Evalúa una lista de condiciones y devuelve una de las varias expresiones de resultado posibles.

La expresión CASE tiene dos formatos:

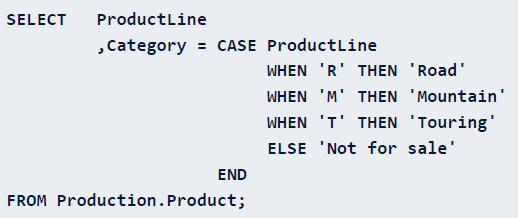
* La expresión CASE sencilla compara una expresión con un conjunto de expresiones sencillas para determinar el resultado.
* La expresión CASE buscada evalúa un conjunto de expresiones booleanas para determinar el resultado.
* Ambos formatos admiten un argumento ELSE opcional.

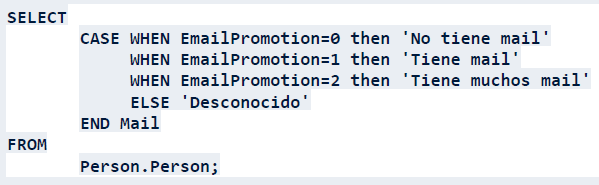
CASE se puede utilizar en cualquier instrucción o cláusula que permite una expresión

válida. Por ejemplo, puede utilizar CASE en instrucciones como SELECT, UPDATE, DELETE y SET, y en cláusulas como select\_list, IN, WHERE, ORDER BY y HAVING.

Sintaxis

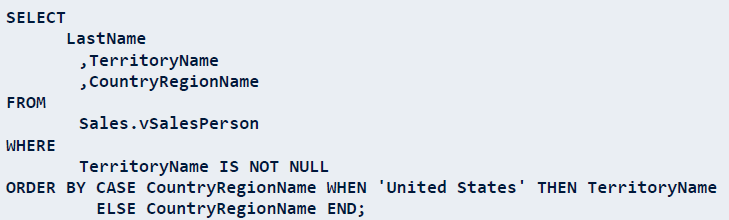
El siguiente ejemplo muestra el uso del case dentro del cuerpo del SELECT





Ordenamiento con CASE

El siguiente ejemplo muestra el uso del case dentro ORDER BY:



Funciones de Agregado

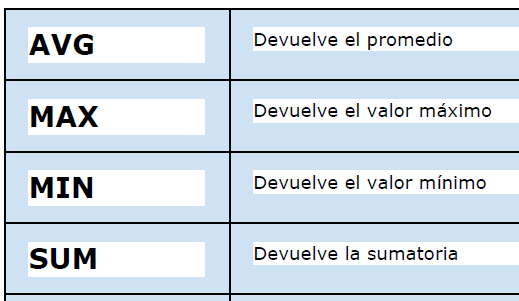
Funciones

Una función de agregado realiza un cálculo sobre un conjunto de valores y devuelve un solo valor.

Con la excepción de COUNT, las funciones de agregado ignoran los valores NULL.

Las funciones de agregado se suelen usar con la cláusula GROUP BY de la instrucción SELECT.

Transact-SQL proporciona las siguientes funciones de agregado:

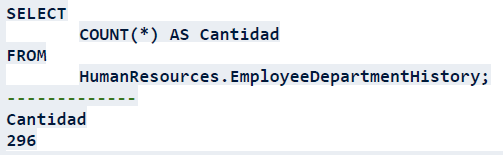


Función COUNT

Esta función devuelve el número de elementos encontrados en un grupo.

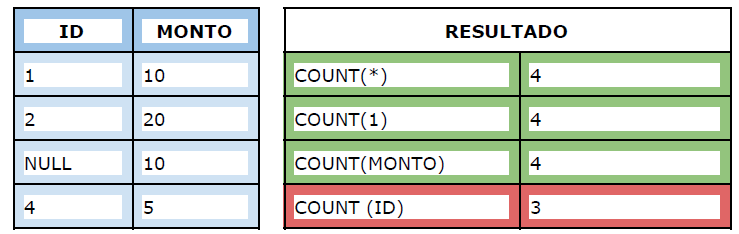
COUNT siempre devuelve un valor de tipo de datos int. COUNT\_BIG siempre devuelve un valor de tipo de datos bigint.

Sintaxis



Comportamiento con valores NULL de la función de agregado Count

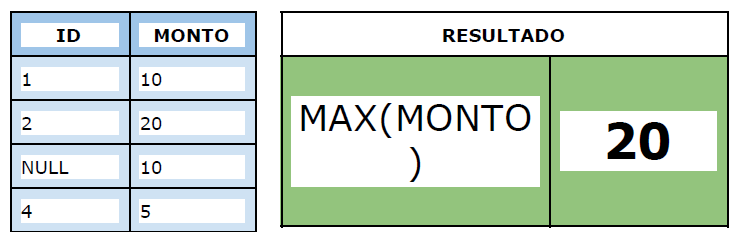
Dada la siguiente tabla veamos los posibles resultados:

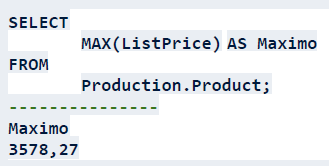


Importante La función de agregado Count ignora los valores NULL.

Función MAX

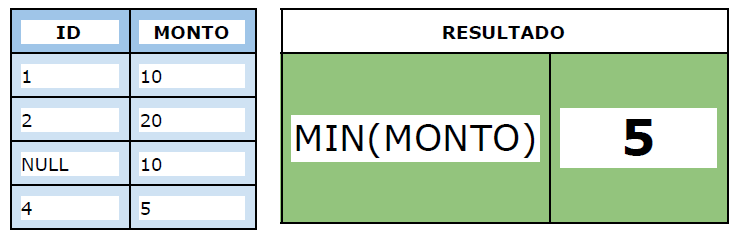
Devuelve el valor máximo de la expresión.





Función MIN

Devuelve el valor mínimo de la expresión

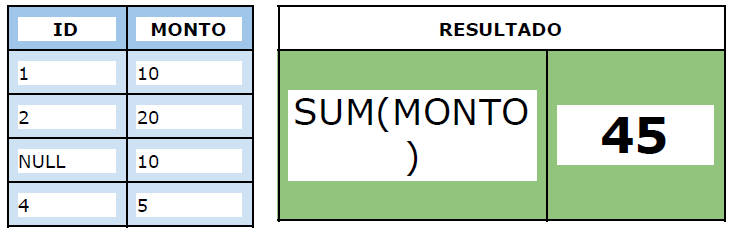


Sintaxis

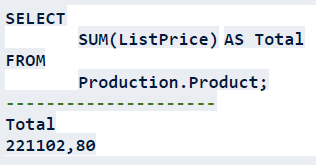


Función SUM

Devuelve la suma de todos los valores o solo de los valores DISTINCT de la expresión. SUM sólo puede utilizarse con columnas numéricas. Se omiten los valores NULL.

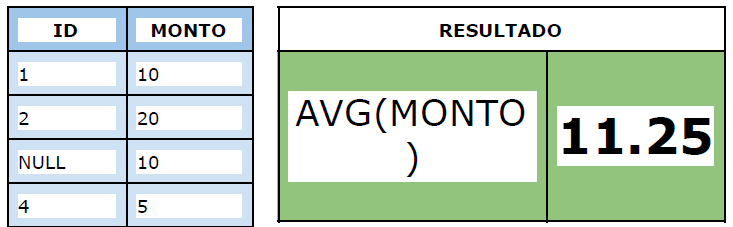


Sintaxis

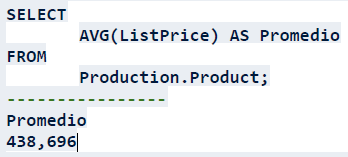


Función AVG

Devuelve el promedio de los valores de un grupo. Omite los valores NULL.



Sintaxis



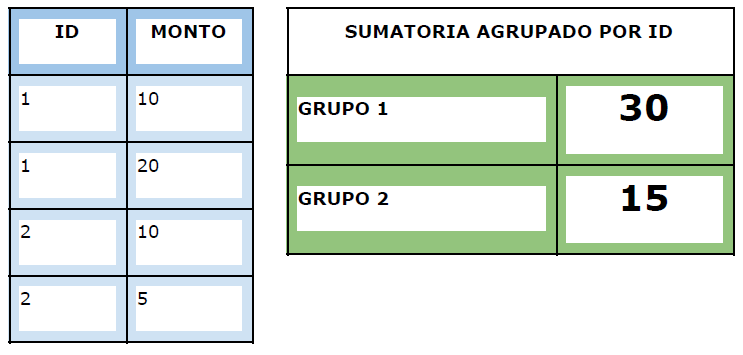
Agrupación y Funciones de Agregado

GROUP BY

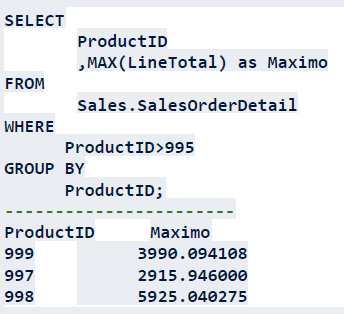
A grupa un conjunto de filas seleccionado en un conjunto de filas de resumen de acuerdo con los valores de una o más columnas o expresiones. Se devuelve una fila para cada grupo. Las funciones de agregado de la lista <select> de la cláusula SELECT proporcionan información de cada grupo en lugar de filas individuales.

Características

* Especifica los grupos en los que se deben colocar las filas de salida
* GROUP BY Establece la lista de columnas por las cuales se agrupará la información
* Esta cláusula organiza los resúmenes de datos agrupados.
* Si se incluyen funciones de agregado en la <lista de selección> de la cláusula SELECT,
* GROUP BY calcula un valor de resumen para cada grupo.
* Cuando se especifica GROUP BY, cada columna que esté en una expresión no agregada de la lista de selección se debe incluir en la lista de GROUP BY o la expresión GROUP BY debe coincidir exactamente con la expresión de la lista de selección.
* Si no se especifica la cláusula ORDER BY, los grupos devueltos con la cláusula GROUP BY no están en un orden determinado. Se recomienda utilizar siempre la cláusula ORDER BY para especificar un orden

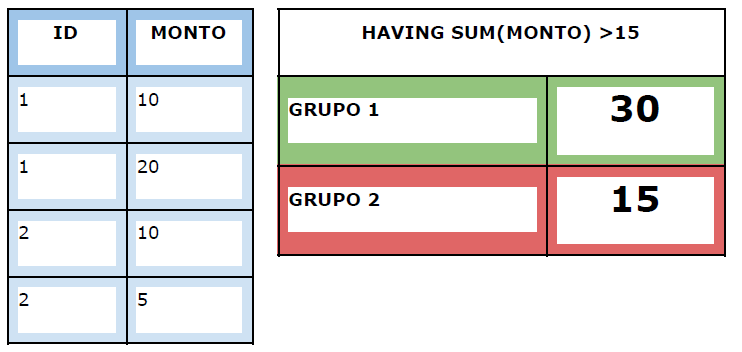


Sintaxis



HAVING

Especifica una condición de búsqueda para un grupo. HAVING solo se puede utilizar con la instrucción SELECT. Normalmente, HAVING se usa con una cláusula GROUP BY. Cuando no se usa GROUP BY, hay un solo grupo implícito agregado.



Sintaxis

