# Funciones SQL Server

## Funciones de Cadena

|  |  |
| --- | --- |
| **Función** | **Descripción** |
| **ASCII(** *expression* **)** | Devuelve el valor de código ASCII del carácter situado más a la izquierda de una expresión de cadena.  **Argumentos**  *expression*: cualquier expresión válida de un tipo **String** ASCII.  **Valor devuelto**  Valor de tipo **Int32**.  **Ejemplo**  ASCII('A') |
| **CHAR(** *expression* **)** | Convierte un código **Int32** en una cadena ASCII.  **Argumentos**  *expression*: valor de tipo **Int32**.  **Valor devuelto**  Valor de tipo **String** ASCII.  **Ejemplo**  char(97) |
| **CHARINDEX(***expression1, expression2* [,*start\_location*]**)** | Devuelve la posición inicial de la expresión especificada en una cadena de caracteres.  **Argumentos**  *expression1*: expresión que contiene la secuencia de caracteres que se va a buscar. La expresión puede ser de un tipo String (ASCII o Unicode) o Binary.  *expression2*: expresión, que normalmente es una columna, en la que se encuentra la secuencia especificada. La expresión puede ser de un tipo String (ASCII o Unicode) o Binary.  *start\_location*: (opcional) valor de tipo Int64 (no se devuelve en SQL Server 2000) o Int32 que representa la posición de caracteres en la que debe comenzar la búsqueda de expression1 en expression2. Si no se especifica start\_location, es un número negativo o es igual a cero, la búsqueda comienza al principio de expression2.  **Valor devuelto**  Valor de tipo **Int32**.  **Ejemplo**  CHARINDEX('h', 'habcdefgh', 2) |
| **DIFFERENCE(***expression, expression* **)** | Compara los valores de **SOUNDEX** de dos cadenas y evalúa la similitud entre ambas.  **Argumentos**  Tipo **String** Unicode o ASCII. *expression* puede ser una constante, una variable o una columna.  **Valor devuelto**  Devuelve un valor de tipo **Int32** que representa la diferencia entre los valores de SOUNDEX de dos expresiones de caracteres. El intervalo está comprendido entre 0 y 4. El valor 0 indica una similitud escasa o inexistente, y el valor 4 indica una elevada similitud o que los valores son iguales.  **Ejemplo**  // The following example returns a DIFFERENCE value of 4,  //the least possible difference or the best match.  DIFFERENCE('Green','Greene'); |
| **LEFT(** *expression, count* **)** | Devuelve la parte izquierda de una cadena de caracteres con el número de caracteres especificado.  **Argumentos**  *expression*: tipo String Unicode o ASCII. Use la función CAST para convertir character\_expression explícitamente.  *count*: valor de tipo **Int64** (no se devuelve en SQL Server 2000) o **Int32** que especifica cuántos caracteres de character\_expression se devolverán.  **Valor devuelto**  Valor **String** Unicode o ASCII.  **Ejemplo**  LEFT('SQL Server', 4) |
| **LEN(** *expression* **)** | Devuelve el número de caracteres de la expresión de cadena especificada, excluidos los espacios en blanco finales.  **Argumentos**  *expression*: expresión de un tipo **String** (ASCII o Unicode) o un tipo **Binary**.  **Valor devuelto**  Valor de tipo **Int32**.  **Ejemplo**  LEN('abcd') |
| **LOWER(** *expression* **)** | Devuelve una expresión de **String** después de convertir a minúsculas los datos de caracteres en mayúsculas.  **Argumentos**  *expression*: cualquier expresión válida del tipo **String**.  **Valor devuelto**  Valor de tipo **String**.  **Ejemplo**  LOWER('AbB') |
| **LTRIM(** *expression* **)** | Devuelve una expresión **String** tras quitar los espacios iniciales en blanco.  **Argumentos**  *expression*: cualquier expresión válida del tipo **String**.  **Valor devuelto**  Un objeto **String**.  **Ejemplo**  LTRIM(' d') |
| **NCHAR(** *expression* **)** | Devuelve el valor de tipo **String** Unicode correspondiente al código entero dado, tal como se define en el estándar Unicode.  **Argumentos**  *expression*: valor de tipo **Int32**.  **Valor devuelto**  Valor de tipo **String** Unicode.  **Ejemplo**  NCHAR(65) |
| **PATINDEX(***'%pattern%'*,*expression***)** | Devuelve la posición inicial de la primera aparición de un patrón en una expresión **String** especificada.  **Argumentos**  *'%pattern%'*: valor de tipo **String** ASCII o Unicode. Se pueden utilizar caracteres comodín; no obstante, el carácter % debe ir delante y detrás del patrón (excepto cuando se busque el primer o último carácter).  *expression*: cadena de tipo **String** ASCII o Unicode en la que buscar el patrón especificado.  **Valor devuelto**  Valor de tipo **Int32**.  **Ejemplo**  PATINDEX('abc', 'ab') |
| **QUOTENAME(***'char\_string'*[, '*quote\_char'*]**)** | Devuelve un valor de tipo **String** Unicode con los delimitadores agregados para convertirla en un identificador delimitado válido de SQL Server 2005.  **Argumentos**  *char\_string*: valor de tipo **String** Unicode.  *quote\_char*: cadena de un solo carácter que se utiliza como delimitador. Puede ser una comilla simple ('), un corchete izquierdo o derecho ([ ]) o una comilla doble ("). Si no se especifica *quote\_char*, se utilizarán corchetes.  **Valor devuelto**  Valor de tipo **String** Unicode.  **Ejemplo**  QUOTENAME('abc[]def') |
| **REPLACE(** *expression1*,*expression2, expression3***)** | Repite una expresión de carácter un número especificado de veces.  **Argumentos**  *expression1*: expresión de cadena que se va a buscar. expression1 puede ser de un tipo String Unicode o ASCII.  *expression2*: subcadena que se va a buscar. expression2 puede ser de un tipo String Unicode o ASCII.  *expression3*: cadena de reemplazo. expression3 puede ser de un tipo String Unicode o ASCII.  **Ejemplo**  REPLACE('aabbcc', 'bc', 'zz') |
| **REPLICATE(***char\_expression*, int\_*expression***)** | Repite una expresión de carácter un número especificado de veces.  **Argumentos**  *char\_expression*: tipo **String** Unicode o ASCII.  *int\_expression*: **Int64** (no se admite en SQL Server 2000) o **Int32**.  **Valor devuelto**  Tipo **String** Unicode o ASCII.  **Ejemplo**  REPLICATE('aa',2) |
| **REVERSE(** *expression* **)** | Devuelve un valor de tipo String Unicode o ASCII con las posiciones de los caracteres invertidas con respecto a la cadena de entrada.  **Argumentos**  *expression*: tipo **String** Unicode o ASCII.  **Valor devuelto**  Tipo **String** Unicode o ASCII.  **Ejemplo**  REVERSE('abcd') |
| **RIGHT(***char\_expression*, *count***)** | Devuelve la parte derecha de una cadena de caracteres con el número de caracteres especificado.  **Argumentos**  *char\_expression*: tipo String Unicode o ASCII. Use la función CAST para convertir character\_expression explícitamente.  *count*: valor de tipo **Int64** (no se devuelve en SQL Server 2000) o **Int32** que especifica cuántos caracteres de character\_expression se devolverán.  **Valor devuelto**  Un tipo **String** ASCII.  **Ejemplo**  RIGHT('SQL Server', 6) |
| **RTRIM(** *expression* **)** | Devuelve un valor de tipo String Unicode o ASCII después de quitar los espacios finales.  **Argumentos**  *expression*: tipo **String** Unicode o ASCII.  **Valor devuelto**  Tipo **String** Unicode o ASCII.  **Ejemplo**  RTRIM(' d e ') |
| **SOUNDEX(** *expression***)** | Devuelve un código de cuatro caracteres (SOUNDEX) para evaluar la similitud de dos cadenas.**Argumentos**  *expression*: tipo String Unicode o ASCII.  **Valor devuelto**  Valor de tipo **String** ASCII. Un código de cuatro caracteres (SOUNDEX) es una cadena que evalúa la semejanza de dos cadenas.  **Ejemplo**  Select SOUNDEX('Smith'), SOUNDEX('Smythe') FROM {1}  **Valores devueltos**  ----- -----  S530 S530 |
| **SPACE(** *int\_expression***)** | Devuelve un valor de tipo **String** ASCII de espacios repetidos.  **Argumentos**  *int\_expression*: valor de tipo **Int64** (no se devuelve en SQL Server 2000) o **Int32** que indica el número de espacios.  **Valor devuelto**  Valor de tipo **String** ASCII.  **Ejemplo**  SPACE(2) |
| **STR(** *float\_expression*[,*length*[,*decimal*]]**)** | Devuelve un valor **String** ASCII convertido a partir de datos numéricos.  **Argumentos**  *float \_expression*: expresión de un tipo de datos (**Double**) numérico aproximado con un separador decimal.  *length*: (opcional) valor de tipo **Int32** que representa la longitud total. Incluye el separador decimal, el signo, los dígitos y los espacios. El valor predeterminado es 10.  *decimal*: (opcional) valor de tipo **Int32** que representa el número de lugares a la derecha del separador decimal. decimal debe ser menor o igual que 16. Si decimal es mayor que 16, el resultado se trunca a dieciséis lugares a la derecha del separador decimal.  **Valor devuelto**  Valor de tipo **String** ASCII.  **Ejemplo**  STR(212.0) |
| **STUFF(** *str\_expression*,*start, length*,*str\_expression\_to\_insert***)** | Elimina una cantidad especificada de caracteres e inserta otro juego de caracteres a partir del punto inicial especificado de una expresión de cadena.  **Argumentos**  *str\_expression*: tipo **String** Unicode o ASCII.  *start:*valor **Int64** (no se devuelve en SQL Server 2000) o **Int32** que especifica la ubicación al principio de la eliminación e inserción.  *length*: valor **Int64** (no se devuelve en SQL Server 2000) o **Int32** que especifica el número de caracteres que se van a eliminar.  *str\_expression\_to\_insert*: valor **String** Unicode o ASCII.  **Valor devuelto**  Valor **String** Unicode o ASCII.  **Ejemplo**  STUFF('abcd', 2, 2, 'zz') |
| **SUBSTRING(***str\_expression*, *start, length***)** | Devuelve parte de una expresión **String**.  **Argumentos**  *str\_expression*: expresión de un tipo **String** (ASCII o Unicode) o un tipo **Binary**.  *start*: valor de tipo **Int64** (no se devuelve en SQL Server 2000) o **Int32** que especifica dónde comienza la subcadena. 1 se refiere al primer carácter de la cadena.  *length*: valor **Int64** (no se devuelve en SQL Server 2000) o **Int32** que especifica cuántos caracteres de la expresión se devolverán.  **Valor devuelto**  Valor de tipo **String** (ASCII o Unicode) o **Binary**.  **Ejemplo**  SUBSTRING('abcd', 2, 2) |
| **UNICODE(** *expression***)** | Devuelve el valor entero, según la definición del estándar Unicode, para el primer carácter de la expresión de entrada.  **Argumentos**  *expression*: valor **String** Unicode.  **Valor devuelto**  Valor de tipo **Int32**.  **Ejemplo**  UNICODE('a') |
| **UPPER(** *expression* **)** | Devuelve una expresión **String** después de convertir a mayúsculas los datos de caracteres que están en minúsculas.  **Argumentos**  *expression*: expresión de un tipo String ASCII o Unicode.  **Valor devuelto**  Tipo **String** ASCII o Unicode.  **Ejemplo**  UPPER('AbB') |

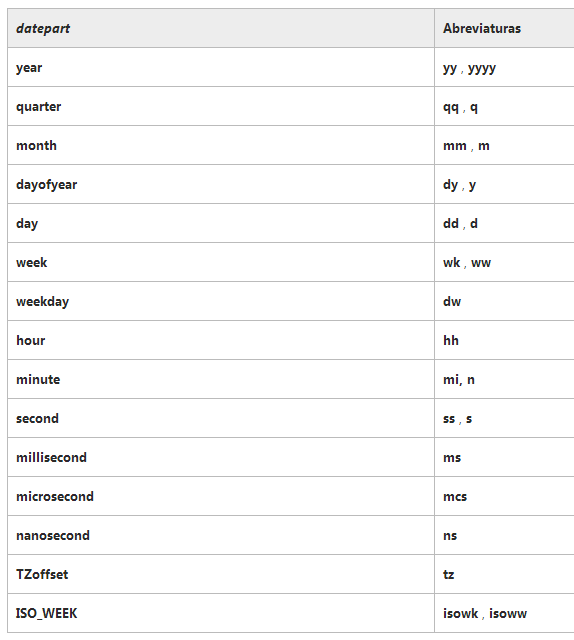
## Funciones Matemáticas

|  |  |
| --- | --- |
| **Función** | **Descripción** |
| **ABS(** *expression* **)** | Lleva a cabo la función que devuelve el valor absoluto.  **Argumentos**  *expression*: valor de tipo **Int32**,**Int64**, **Double** o **Decimal**.  **Valor devuelto**  Valor absoluto de la expresión especificada.  **Ejemplo**  ABS(-2) |
| **ACOS(** *expression* **)** | Devuelve el valor del arcocoseno de la expresión especificada.  **Argumentos**  *expression*: valor de tipo **Double**.  **Valor devuelto**  Valor de tipo **Double**.  **Ejemplo**  ACOS(.9) |
| **ASIN(** *expression* **)** | Devuelve el valor del arcoseno de la expresión especificada.  **Argumentos**  *expression*: valor de tipo **Double**.  **Valor devuelto**  Objeto **Double**.  **Ejemplo**  ASIN(.9) |
| **ATAN(** *expression***)** | Devuelve el valor del arcotangente de la expresión numérica especificada.  **Argumentos**  *expression*: valor de tipo **Double**.  **Valor devuelto**  Valor de tipo **Double**.  **Ejemplo**  ATAN(9) |
| **ATN2(** *expression*,*expression***)** | Devuelve el ángulo, en radianes, cuya tangente se encuentra entre las dos expresiones numéricas especificadas.  **Argumentos**  *expression*: valor de tipo **Double**.  **Valor devuelto**  Valor de tipo **Double**.  **Ejemplo**  ATN2(9, 8) |
| **CEILING(***expression* **)** | Convierte la expresión especificada al número entero más pequeño mayor o igual que él.  **Argumentos**  *expression*: valor de tipo **Int32**,**Int64**, **Double** o **Decimal**.  **Valor devuelto**  Valor de tipo**Int32**,**Int64**, **Double** o **Decimal**.  **Ejemplo**  SQL  SELECT VALUE product FROM AdventureWorksEntities.Products  AS product WHERE product.ListPrice =  CEILING(product.ListPrice) |
| **COS(** *expression* **)** | Calcula el coseno trigonométrico del ángulo especificado, en radianes.  **Argumentos**  *expression*: valor de tipo **Double**.  **Valor devuelto**  Valor de tipo **Double**.  **Ejemplo**  COS(45) |
| **COT(** *expression* **)** | Calcula la cotangente trigonométrica del ángulo especificado, en radianes.  **Argumentos**  *expression*: valor de tipo **Double**.  **Valor devuelto**  Valor de tipo **Double**.  **Ejemplo**  COT(60) |
| **DEGREES(** *radians***)** | Devuelve el ángulo correspondiente en grados.  **Argumentos**  *expression*: valor de tipo **Int32**,**Int64**, **Double** o **Decimal**.  **Valor devuelto**  Valor de tipo**Int32**,**Int64**, **Double** o **Decimal**.  **Ejemplo**  DEGREES(3.1) |
| **EXP(** *expression* **)** | Calcula el valor exponencial de la expresión numérica especificada.  **Argumentos**  *expression*: valor de tipo **Double**.  **Valor devuelto**  Valor de tipo **Double**.  **Ejemplo**  EXP(1) |
| **FLOOR(** *expression***)** | Convierte la expresión especificada al número entero más grande que sea menor o igual que ella.  **Argumentos**  *expression*: valor de tipo **Double**.  **Valor devuelto**  Valor de tipo **Double**.  **Ejemplo**  SQL  SELECT VALUE product FROM AdventureWorksEntities.Products  AS product WHERE product.ListPrice =  FLOOR(product.ListPrice) |
| **LOG(** *expression* **)** | Calcula el logaritmo natural de la expresión **float** especificada.  **Argumentos**  *expression*: valor de tipo **Double**.  **Valor devuelto**  Valor de tipo **Double**.  **Ejemplo**  LOG(100) |
| **LOG10(** *expression***)** | Devuelve el logaritmo en base 10 de la expresión **Double** especificada.  **Argumentos**  *expression*: valor de tipo **Double**.  **Valor devuelto**  Objeto **Double**.  **Ejemplo**  LOG10(100) |
| **PI()** | Devuelve el valor constante de Pi como un **Double**.  **Valor devuelto**  Objeto **Double**.  **Ejemplo**  PI() |
| **POWER(***numeric\_expression, power\_expression* **)** | Calcula el valor de la expresión especificada elevada a la potencia indicada.  **Argumentos**  *numeric\_expression*: valor de tipo**Int32**,**Int64**, **Double** o **Decimal**.  *power\_expression*: valor de tipo **Double** que representa la potencia a la que se va a elevar *numeric\_expression*.  **Valor devuelto**  Valor de la *numeric\_expression* especificada a la *power\_expression* especificada.  **Ejemplo**  POWER(2,7) |
| **RADIANS(***expression* **)** | Convierte grados en radianes.  **Argumentos**  *expression*: valor de tipo **Int32**,**Int64**, **Double** o **Decimal**.  **Valor devuelto**  Valor de tipo**Int32**,**Int64**  **Double** o  **Decimal** .  **Ejemplo**  RADIANS(360.0) |
| **RAND(**[inicialización]**)** | Devuelve un valor aleatorio de 0 a 1.  **Argumentos**  Retruns el valor de inicialización como **Int32**. Si la inicialización no se especifica, el motor de base de datos de SQL Server asigna uno de forma aleatoria. Para un valor de inicialización especificado, el resultado devuelto es siempre el mismo.  **Valor devuelto**  Valor **Double** aleatorio de 0 a 1.  **Ejemplo**  RAND() |
| **ROUND(***numeric\_expression, length* [ ,*function* ]**)** | Devuelve una expresión numérica, redondeada a la longitud o precisión especificadas.  **Argumentos**  *numeric\_expression*: valor de tipo**Int32**,**Int64**, **Double** o **Decimal**.  *length*: Valor de tipo **Int32** que representa la precisión a la que se va a redondear *numeric\_expression*. Si *length* es un número positivo,*numeric\_expression* se redondea al número de posiciones decimales que especifica *length*. Si *length* es un número negativo,*numeric\_expression* se redondea a la izquierda del separador decimal, según se especifica en *length*.  *function*: (opcional) Valor **Int32** que representa el tipo de operación que se va a realizar. Si la función se omite o tiene el valor 0 (predeterminado), *numeric\_expression*se redondea. Si se especifica un valor distinto de 0, *numeric\_expression*se trunca.  **Valor devuelto**  Valor de la *numeric\_expression* especificada a la *power\_expression* especificada.  **Ejemplo**  ROUND(748.58, -3) |
| **SIGN(** *expression* **)** | Devuelve el signo positivo (+1), cero (0) o negativo (-1) de la expresión especificada.  **Argumentos**  *expression*: valor de tipo **Int32**,**Int64**, **Double** o **Decimal**  **Valor devuelto**  Valor de tipo**Int32**,**Int64**, **Double** o **Decimal**.  **Ejemplo**  SIGN(-10) |
| **SIN(** *expression* **)** | Calcula el seno trigonométrico de un ángulo especificado, en radianes, y devuelve una expresión de tipo **Double**.  **Argumentos**  *expression*: valor de tipo **Double**.  **Valor devuelto**  Valor de tipo **Double**.  **Ejemplo**  SIN(20) |
| **SQRT(** *expression* **)** | Devuelve la raíz cuadrada de la expresión especificada.  **Argumentos**  *expression*: valor de tipo **Double**.  **Valor devuelto**  Valor de tipo **Double**.  **Ejemplo**  SQRT(3600) |
| **SQUARE(***expression* **)** | Devuelve el cuadrado de la expresión especificada.  **Argumentos**  *expression*: valor de tipo **Double**.  **Valor devuelto**  Valor de tipo **Double**.  **Ejemplo**  SQUARE(25) |
| **TAN(** *expression* **)** | Calcula la tangente de una expresión especificada.  **Argumentos**  *expression*: **Double**  **Valor devuelto**  **Double**  **Ejemplo**  TAN(45.0) |

## Funciones de fecha y hora

|  |  |
| --- | --- |
| **Función** | **Descripción** |
| **DATEADD(** *datepart*,*number*, *date***)** | Devuelve un valor **DateTime** nuevo que resulta de sumar un intervalo a la fecha especificada.  **Argumentos**  *datepart*: valor de tipo **String** que representa qué parte de la fecha se devuelve como el valor nuevo.  *number*: valor de tipo **Int32**, **Int64**, **Decimal** o **Double** que se usa para incrementar el valor de *datepart*.  *date:*expresión que devuelve un valor de tipo **DateTime**, **DateTimeOffset** o **Time** con precisión = [0-7], o una cadena de caracteres en un formato de fecha.  **Valor devuelto**  Valor de tipo **DateTime**, **DateTimeOffset** o **Time** nuevo con precisión = [0-7].  **Ejemplo**  DATEADD('day', 22, cast('6/9/2006' as DateTime)) |
| **DATEDIFF(** *datepart*,*startdate*, *enddate***)** | Devuelve el número de límites de fecha y hora entre dos fechas especificadas.  **Argumentos**  *datepart*: **String** que representa la parte de la fecha para calcular la diferencia.  *startdate*: la fecha de comienzo para el cálculo es una expresión que devuelve un valor de tipo **DateTime**, **DateTimeOffset** o**Time** con precisión = [0-7], o una cadena de caracteres en formato de fecha.  *enddate:*: la fecha de fin para el cálculo es una expresión que devuelve un valor de tipo **DateTime**, **DateTimeOffset** o **Time** con precisión = [0-7], o una cadena de caracteres en formato de fecha.  **Valor devuelto**  Valor de tipo **Int32**.  **Ejemplo**  DATEDIFF('day', cast('6/9/2006' as DateTime),  cast('6/20/2006' as DateTime)) |
| **DATENAME(** *datepart*,*date***)** | Devuelve una cadena de caracteres que representa el datepart especificado de la fecha especificada.  **Argumentos**  *datepart*: valor de tipo **String** que representa qué parte de la fecha se devuelve como el valor nuevo.  *date*: expresión que devuelve un valor de tipo **DateTime,DateTimeOffset** o **Time** con precisión = [0-7], o una cadena de caracteres en un formato de fecha.  **Valor devuelto**  La cadena de caracteres que representa el datepart especificado de la fecha especificada.  **Ejemplo**  DATENAME('year', cast('6/9/2006' as DateTime)) |
| **DATEPART(** *datepart*,*date***)** | Devuelve un número entero que representa el datepart especificado de la fecha dada.  **Argumentos**  *datepart*: valor de tipo **String** que representa qué parte de la fecha se devuelve como el valor nuevo.  *date*: expresión que devuelve un valor de tipo **DateTime,DateTimeOffset,** o **Time** con precisión = [0-7], o una cadena de caracteres en un formato de fecha.  **Valor devuelto**  El datepart especificado de la fecha especificada como un valor de tipo **Int32**.  **Ejemplo**  DATEPART('year', cast('6/9/2006' as DateTime)) |
| **DAY(** *date* **)** | Devuelve el día de la fecha especificada como un número entero.  **Argumentos**  *date*:expresión de tipo **DateTime** o **DateTimeOffset** con precisión = 0-7.  **Valor devuelto**  Día de la fecha especificada como un valor de tipo **Int32**.  **Ejemplo**  DAY(cast('6/9/2006' as DateTime)) |
| **GETDATE()** | Genera la fecha y hora actuales en el formato interno de SQL Server para los valores datetime.  **Valor devuelto**  La fecha y hora actuales del sistema como **DateTime** con una precisión de 3.  **Ejemplo**  GETDATE() |
| **GETUTCDATE()** | Devuelve el valor datetime en formato de hora universal coordinada (UTC) o del meridiano de Greenwich.  **Valor devuelto**  El valor **DateTime** con una precisión de 3 en formato UTC.  **Ejemplo**  GETUTCDATE() |
| **MONTH(** *date* **)** | Devuelve el mes de la fecha especificada como un número entero.  **Argumentos**  *date*:expresión de tipo **DateTime** o **DateTimeOffset** con precisión = 0-7.  **Valor devuelto**  Mes de la fecha especificada como un **Int32**.  **Ejemplo**  MONTH(cast('6/9/2006' as DateTime)) |
| **YEAR(** *date* **)** | Devuelve el año de la fecha especificada como un número entero.  **Argumentos**  *date*:expresión de tipo **DateTime** o **DateTimeOffset** con precisión = 0-7.  **Valor devuelto**  Año de la fecha especificada como un valor de tipo **Int32**.  **Ejemplo**  YEAR(cast('6/9/2006' as DateTime)) |
| **SYSDATETIME()** | Devuelve un valor **DateTime** con una precisión de 7.  **Valor devuelto**  Valor **DateTime** con una precisión de 7.  **Ejemplo**  SYSDATETIME() |
| **SYSUTCDATE()** | Devuelve el valor datetime en formato de hora universal coordinada (UTC) o del meridiano de Greenwich.  **Valor devuelto**  Valor **DateTime** con una precisión = 7 en formato UTC.  **Ejemplo**  SYSUTCDATE() |
| **SYSDATETIMEOFFSET()** | Devuelve un valor **DateTimeOffset** con una precisión de 7.  **Valor devuelto**  Valor **DateTimeOffset** con una precisión de 7 en formato UTC.  **Ejemplo**  SYSDATETIMEOFFSET() |

Los valores de datepart pueden ser: year (año), quarter (cuarto), month (mes), day (dia), week (semana), hour (hora), minute (minuto), second (segundo), millisecond (milisegundo), dw (día de la semana).



## Estilos de fecha y hora para convert

CONVERT ( data\_type [ ( length ) ] , expression [ , style ] )

Para campos FechaHora:

CONVERT ( varchar , campoFechaHora , style ] )

Cuando *expression* es un tipo de datos de fecha u hora,*style* puede ser uno de los valores que se muestran en la tabla siguiente. Otros valores se procesan como 0. SQL Server admite el formato de fecha en estilo árabe mediante el algoritmo kuwaití.

|  |  |  |  |
| --- | --- | --- | --- |
| **Sin el siglo (aa) (1)** | **Con el siglo (aaaa)** | **Estándar** | **Entrada/salida (3)** |
| - | **0** o **100**(1,2) | Predeterminado | mes dd aaaa hh:mia.m. (o p.m.) |
| **1** | **101** | EE. UU. | 1 = mm/dd/aa  101 = mm/dd/aaaa |
| **2** | **102** | ANSI | 2 = aa.mm.dd  102 = aaaa.mm.dd |
| **3** | **103** | Británico/Francés | 3 = dd/mm/aa  103 = dd/mm/aaaa |
| **4** | **104** | Alemán | 4 = dd.mm.aa  104 = dd.mm.aaaa |
| **5** | **105** | Italiano | 5 = dd-mm-aa  105 = dd-mm-aaaa |
| **6** | **106** (1) | - | 6 = dd mes aa  106 = dd mes aaaa |
| **7** | **107** (1) | - | 7 = Mes dd, aa  107 = Mes dd, aaaa |
| **8** | **108** | - | hh:mi:ss |
| - | **9** ó **109**(1,2) | Valor predeterminado + milisegundos | mes dd aaaa hh:mi:ss:mmma.m. (o p.m.) |
| **10** | **110** | EE. UU. | 10 = mm-dd-aa  110 = mm-dd-aaaa |
| **11** | **111** | JAPÓN | 11 = aa/mm/dd  111 = aaaa/mm/dd |
| **12** | **112** | ISO | 12 = aammdd  112 = aaaammdd |
| - | **13** o **113**(1,2) | Europeo predeterminado + milisegundos | dd mes aaaa hh:mi:ss:mmm(24h) |
| **14** | **114** | - | hh:mi:ss:mmm(24h) |
| - | **20** ó **120**(2) | ODBC canónico | aaaa-mm-dd hh:mi:ss(24h) |
| - | **21** ó **121**(2) | ODBC canónico (con milisegundos) | aaaa-mm-dd hh:mi:ss.mmm(24h) |
| - | **126**(4) | ISO8601 | aaaa-mm-ddThh:mi:ss.mmm (sin espacios)   |  | | --- | | **Nota Nota** | | Cuando el valor para milisegundos (mmm) es 0, no se muestra el valor de milisegundos. Por ejemplo, el valor '2012-11-07T18:26:20.000' se muestra como '2012-11-07T18:26:20'. | |
| - | **127**(6, 7) | ISO8601 con zona horaria Z. | aaaa-mm-ddThh:mi:ss.mmmZ (sin espacios)   |  | | --- | | **Nota Nota** | | Cuando el valor para milisegundos (mmm) es 0, no se muestra el valor de milisegundos. Por ejemplo, el valor '2012-11-07T18:26:20.000' se muestra como '2012-11-07T18:26:20'. | |
| - | **130** (1,2) | Hijri (5) | dd mes aaaa hh:mi:ss:mmma.m. |
| - | **131** (2) | Hijri (5) | dd/mm/aa hh:mi:ss:mmma.m. |

1 Estos valores de estilo devuelven resultados no deterministas. Incluye todos los estilos (aa) (sin el siglo) y un subconjunto de estilos (aaaa) (con el siglo).

2 Los valores predeterminados (*style***0** o **100**, **9** o **109**, **13** o **113**, **20** o **120** y **21** o **121**) siempre devuelven el siglo (aaaa).

3 Entrada cuando se convierte en **datetime**; salida cuando se convierte en datos de caracteres.

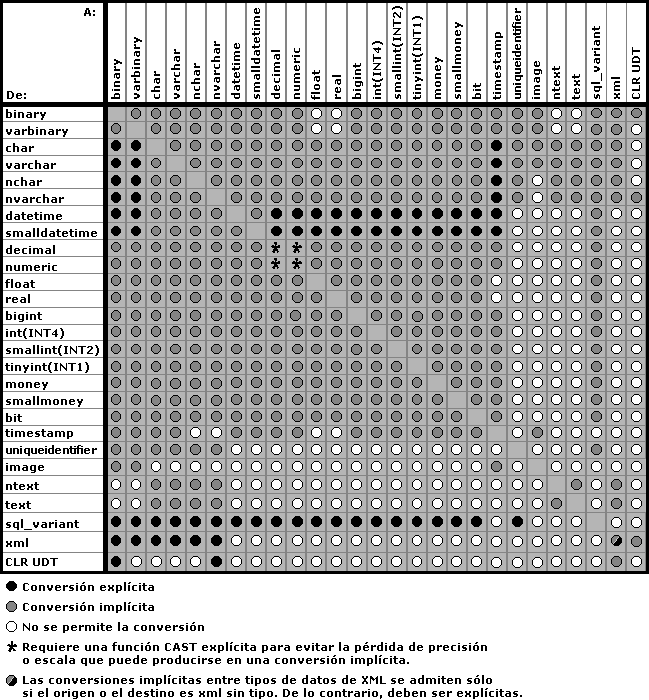
## CAST y CONVERT

Sintaxis para CAST:

CAST ( expression AS data\_type [ ( length ) ] )

Sintaxis para CONVERT:

CONVERT ( data\_type [ ( length ) ] , expression [ , style ] )



Las conversiones implícitas son aquellas conversiones que tienen lugar sin especificar las funciones CAST o CONVERT. El SQL Server las intenta realizar cada vez que los datos no casan.

Las conversiones explícitas son las realizadas a través de las funciones cast y convert.

Ejemplos:

CAST(ListPrice AS int)

SELECT 'The list price is ' + CAST(ListPrice AS varchar(12)) AS ListPrice

FROM Production.Product

SELECT CAST(10.6496 AS int)

SELECT CAST(10.3496847 AS money)