Funciones Transact-SQL

1. Funciones de fecha

GETDATE

La función GETDATE devuelve la fecha actual en formato datetime.

```
GETDATE()
```

Ejemplo:
SELECT GETDATE() AS Ahora;

Una variante es la función GETUTCDATE()

GETUTCDATE

Devuelve la hora UTC (hora universal coordinada u hora del meridiano de Greenwich) actual. La hora UTC actual se deriva de la hora local actual y la configuración de zona horaria del sistema operativo del equipo en el que se ejecuta la instancia de Microsoft SQL Server.

DATEPART

Devuelve un entero que representa la parte de la fecha especificada expresada en la unidad indicada.

```
DATEPART (unidad,fecha)
```

unidad	Abreviaturas	Significado
year	yy, yyyy	año
quarter	qq, q	cuatrimestre
month	mm, m	mes
dayofyear	dy, y	nº día del año
day	dd, d	día
week	wk, ww	semana
weekday	dw, w	día de la semana
hour	hh	hora
minute	mi, n	minuto
second	ss, s	segundo
millisecond	ms	milisegundo

Week (wk, ww) indica el número de semana del año, refleja los cambios realizados en SET DATEFIRST (que indica el primer día de la semana). El 1 de enero de cualquier año define el número de inicio para week, por ejemplo: DATEPART(wk, '1 Ene 2009') = 1.

Weekday (dw) devuelve un número que corresponde al día de la semana, por ejemplo: Lunes = 1, Sábado = 6. El número generado con la unidad weekday depende del valor establecido por SET DATEFIRST.

Fecha

Es una expresión que devuelve un valor datetime o smalldatetime, o una cadena de caracteres con formato de fecha. El tipo de datos datetime sólo sirve para fechas a partir del 1 de enero de 1753.

Ejemplos
SELECT DATEPART(dw, '20/10/2008') devuelve 1 (es lunes).
SELECT 'dd',DATEPART(dd, '20/10/2008') devuelve 20
SELECT 'dy',DATEPART(dy, '20/10/2008') devuelve 294

DATEFROMPARTS

Devuelve la fecha correspondiente al año, mes y día especificados.

```
DATEFROMPARTS(año, mes, día)
```

Los tres parámetros son enteros.
Ejemplo SELECT DATEFROMPARTS(2016,03,19) devuelve la fecha 2016-03-19

DAY

Devuelve un entero que representa el día de la fecha especificada.

```
DAY(fecha)
```

Esta función es equivalente a DATEPART(**dd,** *fecha*).

MONTH

Devuelve un entero que representa el mes de la fecha especificada.

```
MONTH(fecha)
```

Esta función es equivalente a DATEPART(**mm,** *fecha*).

YEAR

Devuelve un entero que representa el año de la fecha especificada.

```
YEAR(fecha)
```

Esta función es equivalente a DATEPART(**yy,** *fecha*).

DATENAME

Devuelve una cadena de caracteres que representa el valor de la unidad especificada de una fecha especificada.

```
DATENAME(unidad, fecha)
```

Unidad puede tomar los mismos valores que en la función DATEPART
Cuando la unidad corresponde a mes o día de la semana, devuelve el valor en letras.
Ejemplos:
SELECT 'dd',DATENAME(dd, '20/10/2008') devuelve 20
SELECT 'mm',DATENAME(mm, '20/10/2008') devuelve Octubre
SELECT 'yy',DATENAME(yy, '20/10/2008') devuelve 2008
SELECT 'dw',DATENAME(dw, '20/10/2008') devuelve Lunes

DATEADD

Devuelve un valor datetime nuevo que resulta de sumar un intervalo de tiempo a una fecha especificada.

```
DATEADD(intervalo, número, fecha)
```

"Añade a la *fecha número intervalos*".
Valores válidos para intervalo:

intervalo	Abreviaturas	Significado
year	**yy**, **yyyy**	**año**
quarter	**qq**, **q**	**cuatrimestre**
month	**mm**, **m**	**mes**
dayofyear	**dy**, **y**	**nº día del año**
day	**dd**, **d**	**día**
week	**wk**, **ww**	**semana**
weekday	**dw**, **w**	**día de la semana**
hour	**hh**	**hora**
minute	**mi**, **n**	**minuto**
second	**ss**, **s**	**segundo**
millisecond	**ms**	**milisegundo**

Si *número* no es entero, se descarta la parte decimal.
Cuando *intervalo* es month, el número de días del mes afecta al resultado.
Por ejemplo,
SELECT DATEADD(month, 1, '08/30/2006')
SELECT DATEADD(month, 1, '08/31/2006')
Las dos instrucciones siguientes devuelven 2006-09-30 00:00:00.000.
Agosto tiene 31 días y septiembre tiene 30 días, al agregar un mes al final de agosto, se devuelve el último día de septiembre.

DATEDIFF

Devuelve el número de intervalos que hay entre las dos fechas especificadas.

```
DATEDIFF( intervalo , fechainicial , fechafinal )
```

Se resta fechainicial de fechafinal. Para expresar el tipo de intervalo se utilizan los mismos valores que en la función DATEADD.

El valor devuelto es de tipo entero por lo que si el resultado es un número mayor que el máximo entero, la función da error. Para los milisegundos, el número máximo es de 24 días, 20 horas, 31 minutos y 23.647 segundos. Para los segundos, el número máximo es 68 años.

Por ejemplo:

```
SELECT DATEDIFF(year, contrato, getdate()) AS [Años trabajados]
FROM empleados;
```

Nos devuelve la diferencia de años existentes entre la fecha de hoy (GETDATE()) y la fecha de contrato del empleado.

@@DATEFIRST

Devuelve el primer día de la semana establecido con SET DATEFIRST.

SET DATEFIRST

Establece el primer día de la semana en un número del 1 al 7.

```
SET DATEFIRST número
```

Se puede utilizar la función @@DATEFIRST para ver el valor actual de SET DATEFIRST.
La opción SET DATEFIRST se establece en tiempo de ejecución.
Número es un entero que indica el primer día de la semana. Puede ser uno de los siguientes valores.

número	Primer día de la semana
1	Lunes
2	Martes
3	Miércoles
4	Jueves
5	Viernes
6	Sábado
7	Domingo

1.1. FUNCIONES DE CADENA

ASCII

Devuelve el valor de código ASCII del carácter situado más a la izquierda de una expresión de caracteres.

```
ASCII(expresion_cadena)
```

expresion_cadena Es una expresión de tipo **char** o **varchar**.
La función devuelve un valor **int**.

Ejemplo:
ASCII('abc') devuelve 97, lo mismo que ASCII('a').

CHAR

Devuelve el carácter según el código ASCII correspondiente a un número entero.

```
CHAR(numero)
```

Numero es un número entero entre 0 y 255 de lo contrario la función devuelve NULL.

Se suele utilizar para insertar caracteres de control en cadenas alfanuméricas.

Carácter de control	Valor
Tabulación	**char(9)**
Avance de línea	**char(10)**
Retorno de carro	**char(13)**

Ejemplo:
ASCII('abc') devuelve a.

NCHAR

Devuelve el carácter Unicode correspondiente al código entero dado, tal como se define en el estándar Unicode.

```
NCHAR(numero)
```

Numero es un número entero entre 0 y 65535 de lo contrario la función devuelve NULL.

Ejemplo:
NCHAR(400) devuelve ε

UNICODE

Devuelve el valor entero, según la definición del estándar Unicode, del primer carácter de la expresión de entrada.

```
UNICODE(expresion_cadena)
```

expresion_cadena Es una expresión de tipo N**char** o N**varchar**.
La función devuelve un valor **int**.

Ejemplo:
UNICODE(N'Ɛ') devuelve ε

LEN

Devuelve el número de caracteres de la expresión de cadena especificada, excluidos los espacios en blanco finales.

```
LEN(Expresion_cadena)
```

Expresion_cadena es una expresión de cadena
El valor devuelto es de tipo bigint si *Expresion_cadena* tiene el tipo de datos varchar(max), nvarchar(max) o varbinary(max); en caso contrario, int.
Ejemplo:
LEN('abcdef ') devuelve 7

LTRIM

Devuelve una cadena tras quitar todos los espacios iniciales en blanco.

```
LTRIM(Expresion_cadena)
```

Expresion_cadena es una expresión de cadena, debe ser un tipo de datos implícitamente convertible a varchar excepto text, ntext e image.

El valor devuelto es de tipo varchar o nvarchar.
Ejemplo:
LTRIM(' ab cdef ') devuelve ab cdef .

RTRIM

Devuelve una cadena tras quitar todos los espacios finales en blanco.

```
RTRIM(Expresion_cadena)
```

Expresion_cadena es una expresión de cadena, debe ser un tipo de datos implícitamente convertible a varchar excepto **text, ntext e image**.

El valor devuelto es de tipo varchar o nvarchar.
Ejemplo:
RTRIM(' ab cdef ') devuelve ab cdef.

LEFT

Devuelve la parte izquierda de una cadena de caracteres con el número de caracteres especificado.

```
LEFT(Expresion_cadena, longitud)
```

Expresion_cadena es una expresión de cadena
Longitud: es el número de caracteres a devolver. Puede ser de tipo bigint, si es negativo la función devuelve un error.
El valor devuelto es de tipo varchar o nvarchar.
Ejemplo: LEFT('abcdef ', 3) devuelve abc

RIGHT

Devuelve la parte derecha de una cadena de caracteres con el número de caracteres especificado.

```
RIGHT(Expresion_cadena, longitud)
```

Expresion_cadena es una expresión de cadena
Longitud: es el número de caracteres a devolver. Puede ser de tipo bigint, si es negativo la función devuelve un error.

El valor devuelto es de tipo varchar o nvarchar.
Ejemplo:
RIGHT('abcdef ', 3) devuelve def

SUBSTRING

Devuelve parte de una expresión de caracteres, binaria, de texto o de imagen.

```
SUBSTRING ( expresion ,inicio , longitud )
```

expresion
Es una cadena de caracteres, una cadena binaria, texto, imagen, una columna o una expresión que incluye una columna. No se puede utilizar expresiones que incluyan funciones de agregado.
inicio
Es un entero que especifica dónde comienza la subcadena. *start* puede ser de tipo **bigint**.
longitud
Es un entero positivo (puede ser de tipo **bigint**) que especifica cuántos caracteres o bytes de la *expression* se van a devolver. Si *longitud* es negativo, se devuelve un error.

Según el tipo de datos de *Expresion* los datos devueltos son de tipo:

Expresion	Valor devuelto
char/varchar/text	**varchar**
nchar/nvarchar/ntext	**nvarchar**
binary/varbinary/image	**varbinary**

Los valores (*inicio* y *longitud*) que utilizan los tipos de datos **ntext**, **char** o **varchar** deben especificarse en número de caracteres. Los desplazamientos que utilizan los tipos de datos **text**, **image**, **binary** o **varbinary** deben especificarse en número de bytes.

Ejemplo:
SUBSTRING('abcdef',3,2) devuelve cd

LOWER

Devuelve una cadena después de convertir en minúsculas los caracteres en mayúsculas.

```
LOWER(Expresion_cadena)
```

Expresion_cadena es una expresión de cadena, debe ser un tipo de datos implícitamente convertible a varchar.

El valor devuelto es de tipo varchar o nvarchar.
Ejemplo:
LOWER('aBcdEf ', 3) devuelve abcdef

UPPER

Devuelve una cadena después de convertir en mayúsculas los datos de caracteres en minúsculas.

```
UPPER(Expresion_cadena)
```

Expresion_cadena es una expresión de cadena, debe ser un tipo de datos implícitamente convertible a varchar.

El valor devuelto es de tipo varchar o nvarchar.
Ejemplo:
UPPER('aBcdEf ', 3) devuelve ABCDEF

REPLACE

Reemplaza todas las apariciones de un valor de cadena especificado por otro valor de cadena.

```
REPLACE(cadena_original, cadena_a_quitar, cadena_a_poner)
```

Las cadenas pueden ser de tipo carácter o binario. Se pueden combinar con COLLATE.
Devuelve nvarchar si una de las cadenas es de tipo nvarchar, en caso contrario devuelve varchar.
Ejemplo:
REPLACE('Esto es una cuestión','es','xx') devuelve xxto xx una cuxxtión

STUFF

Elimina el número de caracteres especificado e inserta otro conjunto de caracteres en el punto de inicio indicado.

```
STUFF (expresión_origen, inicio , longitud , expresión_sustituta)
```

expresión_origen indica la cadena que se va a modificar.
inicio: la posición de inicio de la subcadena a eliminar, puede ser de tipo bigint.
longitud: la longitud de la subtira a eliminar, puede ser de tipo bigint.
Si *inicio* o *longitud* es negativo, se devuelve una cadena nula.
Si *inicio* es mayor que la longitud de *expresión_origen*, se devuelve una cadena nula.
Si *longitud* es mayor que la longitud de *expresión_origen*, se elimina hasta el último carácter de *expresión_origen*.
expresión_sustituta es la cadena a insertar en el lugar de la que se elimina.
Si la longitud de la cadena resultante es mayor que el máximo admitido, se genera un error.
Tanto *expresión_origen* como *expresión_sustituta* pueden ser tanto de caracteres como binarios.
Ejemplo:
STUFF('abcdef', 2, 3, 'ijklmn') devuelve aijklmnef

QUOTENAME

Devuelve una cadena Unicode con los delimitadores agregados para convertirla en un identificador delimitado válido de Microsoft SQL Server 2005.

```
QUOTENAME ( 'cadena' [ , 'caracter' ] )
```

Cadena es una cadena de caracteres.
Carácter es el carácter delimitador. Puede ser una comilla simple ('), un corchete izquierdo o derecho (**[ ]**) o una comilla doble ("). Si no se especifica, se utilizarán corchetes.
Ejemplo:
QUOTENAME('nombre de campo') devuelve [nombre de campo]
QUOTENAME('nombre de campo','"') devuelve "nombre de campo"

SPACE

Devuelve una cadena de espacios repetidos.

```
SPACE(número)
```

Número es un entero positivo que indica el número de espacios en blanco a devolver, si es negativo la función devuelve una cadena null.
Para incluir espacios en datos Unicode o para devolver más de 8.000 espacios en blanco, utilice REPLICATE en lugar de SPACE.
Ejemplo:
PRINT RTRIM('Ana ') + SPACE(1) + LTRIM(' García') devuelve Ana García.

STR

Devuelve una cadena de caracteres a partir de datos numéricos.

```
STR ( número [ , longitud [ , decimales ] ] )
```

Número es el número a convertir debe ser de tipo numérico.
Longitud es la longitud de la cadena devuelta incluye todos los caracteres, dígitos, punto decimal, signo y espacios. EL valor predeterminado es 10.
Decimales es el número de cifras a la derecha del separador decimal. El valor de *decimales* debe ser menor o igual que 16. Si es mayor que 16, el resultado se trunca en el decimosexto lugar a la derecha del separador decimal. El valor predeterminado es cero.
La función redondea automáticamente los valores.
Cuando la el valor excede de la longitud especificada la función devuelve **.

```
DECLARE @n numeric(5,3);
DECLARE @x float;
SELECT @n=3.67;
```

```
SELECT @x=3.45;
PRINT 'A'+STR(@n,8,2)+'B'
PRINT 'F'+STR(@x,8,2)+'G'
PRINT 'a'+STR(@n,8)+'b'
PRINT 'f'+STR(@x,8)+'g'
```
Da como salida:
```
A    3.67B
F    3.45G
a       4b
f       3g
```

REPLICATE

Repite una expresión de caracteres un número especificado de veces.

```
REPLICATE(expresion_cadena, longitud)
```

expresion_cadena Es una expresión de un tipo de datos que puede convertirse implícitamente a varchar. También puede ser una constante, una variable o una columna de datos binarios.

Longitud es un número entero positivo (puede ser de tipo bigint). Si es negativo, se devuelve una cadena NULL.

Ejemplo: REPLICATE('abc',5) devuelve abcabcabcabcabc

REVERSE

Devuelve una expresión de caracteres invertida.

```
REVERSE(expresion_cadena)
```

expresion_cadena Es una expresión de un tipo de datos que puede convertirse implícitamente a varchar. También puede ser una constante, una variable o una columna de datos binarios.

Ejemplo: REVERSE('abcdef') devuelve fedcba

CHARINDEX

Devuelve la posición inicial de la expresión especificada en una cadena de caracteres.

```
CHARINDEX( cadena_a_buscar , cadena_origen [ , inicio ] )
```

cadena_a_buscar y *cadena_origen* son de tipo de datos carácter.

Inicio indica pa posición a partir de la cual se empieza a buscar. Puede ser de tipo bigint. Si no se especifica o si es cero o negativo, la búsqueda empieza en el primer carácter de *cadena_origen*.

Ejemplo: CHARINDEX('fg', 'abcdefghyfg') devuelve 6

PATINDEX

Devuelve la posición inicial de la primera repetición de un patrón en la expresión especificada, o ceros si el patrón no se encuentra, en todos los tipos de datos de texto y caracteres.

```
PATINDEX ( '%patrón%' , expresion )
```

Patrón es una cadena literal. Se pueden incluir caracteres comodín, aunque el carácter % debe preceder y seguir a *patrón* (excepto cuando se busca el primer o el último carácter). Es una expresión de la categoría de tipo de datos de cadena de caracteres.

Expression es de la categoría de tipo de datos de cadena de caracteres. Se puede combinar con COLLATE.

La función devuelve **bigint** si *expresion* es de los tipos de datos **varchar** o **nvarchar,** en caso contrario, **int**.

Si expresión es una columna, la consulta devolverá todas las filas del origen de la consulta (después de ejecutar el WHERE) e indicará valores distintos de cero para las filas en las que se haya encontrado el patrón y cero para el resto.

Ejemplo:

```
PRINT PATINDEX('%cd%','aasecdgtcd') devuelve 5
PRINT PATINDEX('cd%','aasecdgtcd')   devuelve 0 (lo busca al principio).
PRINT PATINDEX('%cd','aasecdgtcd')   devuelve 9 (lo busca al final).
```

CONCAT

Facilita otra forma de concatenar cadenas.

```
CONCAT ( cadena1, cadena2 [,...] )
```

Ejemplo: CONCAT('uno ', 'dos ', 'tres ', 'fin') es equivalente a 'uno ' + 'dos ' + 'tres ' + 'fin'

2. Otras funciones

ROUND

Devuelve un valor numérico, redondeado a la longitud o precisión especificadas y del mismo tipo.

```
ROUND ( expresion_numerica , longitud [ ,funcion ] )
```

expresion_numerica

Es una expresión de la categoría de tipo de datos numérico exacto o numérico aproximado, excepto el tipo de datos **bit**.

longitud

Es la precisión con la que se redondea *expresion_numerica*. Debe ser una expresión de tipo **tinyint**, **smallint** o **int**. Si *longitud* es un número positivo, se redondea al número de posiciones decimales que especifica *longitud*. Si *longitud* es un número negativo, se redondea a la izquierda del separador decimal, según se especifica en *longitud*.

funcion

Es el tipo de operación que se realiza. *function* debe ser de tipo **tinyint**, **smallint** o **int**. Si *function* se omite o tiene el valor 0 (predeterminado), se redondea. Si se especifica un valor distinto de 0, se trunca.

Ejemplos:

ROUND(1265.45627, 2) devuelve 1265.46000
ROUND(1265.45627, 3) devuelve 1265.45600
ROUND(1265.45627, 2,1) devuelve 1265.45000 trunca
ROUND(1265.45627, -3) devuelve 1000.00000
ROUND(1265.45627, -2) devuelve 1300.00000

CAST y CONVERT

Convierten una expresión de un tipo de datos en otro de forma explícita. CAST y CONVERT proporcionan funciones similares.

```
CAST ( expresion AS tipo_dato [ (longitud) ])

CONVERT (tipo_dato [ (longitud) ] , expresion [ , estilo ] )
```

expresion Es cualquier expresión válida.

tipo_dato Es el tipo de datos de destino proporcionado por el sistema. Incluye **xml**, **bigint** y **sql_variant**. No se pueden utilizar tipos de datos de alias.

Longitud Es un parámetro opcional de los tipos de datos **nchar**, **nvarchar**, **char**, **varchar**, **binary** o **varbinary**. Para CONVERT, si no se ha especificado el parámetro *longitud*, el valor predeterminado es 30 caracteres.

estilo

Es el estilo del formato de fecha usado para convertir datos de tipo **datetime** o **smalldatetime** en datos de caracteres (con tipo de datos **nchar**, **nvarchar**, **char**, **varchar**, **nchar** o **nvarchar**), o para convertir datos de caracteres de formatos de fecha y hora conocidos en datos de tipo **datetime** o **smalldatetime**; o bien, el formato de cadena usado para convertir datos de tipo **float**, **real**, **money** o **smallmoney** en datos de caracteres (con tipo de datos **nchar**, **nvarchar**, **char**, **varchar**, **nchar** o **nvarchar**). Cuando e*stilo* es NULL, el resultado devuelto también es NULL.

En las conversiones money a carácter los estilos utilizados son:

Valor	Resultado
0 valor predeterminado	Sin separadores de millar cada tres dígitos a la izquierda del separador decimal y dos dígitos a la derecha del separador decimal; por ejemplo, 4235,98.
1	Separadores de millar cada tres dígitos a la izquierda del separador decimal y dos dígitos a la derecha del separador decimal; por ejemplo, 3.510,92.
2	Sin separadores de millar cada tres dígitos a la izquierda del separador decimal y cuatro dígitos a la derecha del separador decimal; por ejemplo, 4235,9819.

En las conversiones de tipo fecha a tipo alfanumérico, por ejemplo los estilos 3 y 103 permiten convertir la fecha con el estilo dd/mm/aa y dd/mm/aaaa respectivamente.

Ejemplo:

```
DECLARE @fecha AS datetime;
SELECT @fecha= GETDATE();
PRINT @fecha;                    devuelve Oct 27 2008  7:20PM
PRINT CONVERT (CHAR(8),@fecha,3)  devuelve 27/10/08
```

```
PRINT CONVERT (CHAR(10),@fecha,103)      devuelve 27/10/2008
```

Para más información consulta la página:
https://technet.microsoft.com/es-es/library/ms187928(v=sql.130).aspx
https://technet.microsoft.com/en-us/library/ms187928(v=sql.130).aspx CAST CONVERT Clause

FORMAT

Devuelve un valor con formato con el formato y la referencia cultural opcional especificados en SQL Server 2012. Se recomienda usar la función FORMAT para aplicar formato específico de la configuración regional de los valores de fecha/hora y de número, y usar CAST o CONVERT para las conversiones de tipos de datos generales.

```
FORMAT (valor, formato [, cultura])
```

valor Es el valor a formatear puede ser numérico o fecha.
formato Es el formato a aplicar.
cultura Indica la referencia cultural a aplicar.
Para más información consulta la página:
https://msdn.microsoft.com/es-es/library/hh213505(v=sql.110).aspx
https://msdn.microsoft.com/es-es/library/hh213505(v=sql.110).aspx

IIF

Evalúa una condición y devuelve el primer o segundo valor dependiendo de si el resultado de la condición es verdadero o no.

```
IIF ( condicion, valor_siverdadero, valor_else )
```

condicion es la consición que se evalúa.
valor_siverdadero es el valor devuelto si la condición se cumple.
valor_else es el valor devuelto si la condición da false o null.

Ejemplo: IIF(campo>=100, 'mayor o igual', 'menor o nulo')

CASE

Evalúa una lista de condiciones y devuelve una de las varias expresiones de resultado posibles.
CASE tiene dos formatos:

- La función CASE sencilla compara una expresión con un conjunto de expresiones sencillas para determinar el resultado.
- La función CASE buscada evalúa un conjunto de expresiones booleanas para determinar el resultado.

Ambos formatos son compatibles con un argumento ELSE opcional.

Función CASE simple:

```
CASE expresion_entrada
     WHEN valor THEN resultado
    [ ...n ]
     [
    ELSE resultado_sino
     ]
END
```

Expresion_entrada es la expresión que calcula el valor a evaluar.
Valor es cualquier expresión que devuelva un valor del mismo tipo que *expresion_entrada* (o por lo menos de conversión implícita).
Resultado es el valor devuelto por la función si *expresión_entrada* es igual a *valor*.
Resultado_sino Es el valor devuelto por la función si ninguna comparación se evalúa como TRUE. Si se omite este argumento y ninguna comparación se evalúa como TRUE, CASE devuelve NULL.
Resultado_sino y todos los *Resultado* deben tener el mismo tipo de datos o deben ser de conversión implícita.
Cómo funciona:

- Evalúa *expresion_entrada* y, en el orden especificado, la expresión *expresion_entrada* = *valor* para cada cláusula WHEN.
- Devuelve *resultado* de la primera *expresion_entrada* = *valor* que se evalúa como TRUE.
- Si *expresion_entrada* = *valor* no se evalúa como TRUE, SQL Server 2005 Database Engine (Motor de base de datos de SQL Server 2005) devuelve *resultado_sino* si se especifica una cláusula ELSE, o bien un valor NULL si no se especifica ninguna cláusula ELSE.

Ejemplo:

CASE estado_civil
WHEN 'c' THEN 'casado/a'
WHEN 's' THEN 'soltero/o'
WHEN 'v' THEN 'viudo/a'
ELSE 'otro'
END

Si queremos evaluar si el campo es nulo no podemos añadir un WHEN NULL THEN 'nulo' porque la condición estado_civil = NULL no es TRUE cuando el campo es nulo, en este caso habría que utilizar la segunda forma de la función CASE y utilizar el operador is null.

Función CASE buscada:

```
CASE
     WHEN condicion THEN resultado
    [ ...n ]
     [
    ELSE resultado_sino
     ]
END
```

Condicion es la condición que se evalúa y si el resultado es TRUE la función devuelve resultado.
Devuelve *resultado* de la primera *condicion* que se evalúa como TRUE. SI ninguna condición se evalúa como TRUE la función devuelve NULL.
Ejemplo:

```
CASE
        WHEN estado_civil IS NULL THEN 'nulo'
        WHEN estado_civil = 'c' THEN 'casado/a'
        WHEN estado_civil = 's' THEN 'soltero/o'
        WHEN estado_civil = 'v' THEN 'viudo/a'
        ELSE 'otro'
END
```

CHOOSE

Devuelve el elemento en el índice especificado de la lista de valores indicada.

```
CHOOSE ( indice, valor1, valor2 [,...] )
```

indice es un valor numérico que indica el índice del elemento a devolver de la lista de valores.
valor1, valor2 [,...] representa una lista de n valores.
Cuando el valor de *indice* tiene decimales, se asume su valor redondeado.
La función CHOOSE devuelve NULL cuando *indice* es nulo o está fuera del rango [1..n]

Ejemplo: CHOOSE(3, 'uno ', 'dos ', 'tres ', 'cuatro') devuelve 'tres'

ISNULL

Reemplaza el valor NULL con el valor de reemplazo especificado.

```
ISNULL ( expresion_entrada , valor )
```

expresion_entrada es la expresión que indica donde se va a comprobar la existencia de NULL.
Valor es el valor por el cual se reemplaza NULL.

Ejemplo: ISNULL(precio, 0)
Si en precio hay un valor no nulo la función devuelve ese valor, si precio contiene NULL entonces la función devuelve 0.

COALESCE

Devuelve la primera expresión distinta de NULL entre sus argumentos.

```
COALESCE ( expresion [ ,...n ] )
```

Si todos los argumentos son NULL, la función devuelve NULL.

Por ejemplo, tenemos dos campos *fijo* y *movil*, en uno tenemos el teléfono fijo del cliente y en el otro tenemos el teléfono móvil y queremos obtener el teléfono fijo si tiene y sino el móvil, lo pondríamos así:
COALESCE(fijo, movil)

Referencia:
https://msdn.microsoft.com/library/ms174318(SQL.130).aspx