Funciones disponibles en Microsoft Jet SQL (Access)

A continuación se detallan, clasificadas en categorías, la casi totalidad de las funciones disponibles en Microsoft Jet SQL (el dialecto de SQL usado en Access) con utilidad dentro de sentencias SQL. Ténganse en cuenta las siguientes notas:

- Para cada función se ha indicado qué parámetros toma. Si alguno de ellos es opcional, se ha indicado encerrándolo entre paréntesis. Aquellas funciones que pueden tomar un número variable e indeterminado de parámetros lo indica con puntos suspensivos (…).
- Cuando junto al nombre de la función, al comienzo de su descripción, aparece un segundo nombre separado del primero mediante una flecha (por ejemplo, **CCur(expresión) → CMoneda**), se indica con esto que el diseñador de consultas de Access mostrará la función con el segundo nombre (en este ejemplo, CMoneda), a pesar de que en el SQL la única forma válida es la primera.

Funciones de conversión (librería VBA.Conversion)

CBool(*expresión*)

Convierte cualquier *expresión* numérica o de cadena válida a un valor lógico (booleano) de tipo BIT.
Ejemplos:

- CBool(3) → TRUE
- CBool('false') → FALSE

CByte(*expresión*)

Convierte cualquier *expresión* numérica o de cadena válida a un valor de tipo BYTE, es decir, un entero de 0 a 255.
Ejemplos:

- CByte(23.2) → 23
- CByte('50') → 50

CCur(*expresión*) → CMoneda

Convierte cualquier *expresión* numérica o de cadena válida a un valor de tipo CURRENCY, es decir, un decimal con cuatro decimales de precisión comprendido entre -922.337.203.685.477,5808 y 922.337.203.685.477,5807.
Ejemplos:

- CCur(2+7000) → 7002
- CCur('3' & '5') → 35

CDate(*expresión*) → CFecha

Convierte cualquier *expresión* numérica o de cadena válida a un valor de tipo DATETIME, es decir, una fecha comprendida entre 1/1/100 00:00:00 y 31/12/9999 23:59:59. Si la *expresión* es numérica, se entiende como el número de días transcurridos desde el 30/12/1899 (siendo además la parte decimal la hora del día correspondiente). Si la *expresión* es una cadena, se intentará convertir a fecha usando el formato local definido para las fechas y horas en el sistema; de no lograrse, se intentará con otros formatos de fecha (americano, japonés…) y, en último extremo, se recurrirá a convertir la cadena a número y aplicar el anterior criterio.
Ejemplos:

- CDate(33.5) → #01/31/1900 12:00:00#
- CDate('2/3/1') → #03/02/2001 00:00:00#

CDbl(*expresión*) → CDoble

Convierte cualquier *expresión* numérica o de cadena válida a un valor de tipo DOUBLE, es decir, un número en coma flotante de doble precisión: -1,79769313486232E308 a -4,94065645841247E-324 para valores negativos; 4,94065645841247E-324 a 1,79769313486232E308 para valores positivos. Si la *expresión* es una cadena, se usará como separador decimal el definido en el sistema.
Ejemplos:

- CDbl(50 + 0.4) → 50.4
- CDbl('-2.3') → -23
- CDbl('5,7') → 5.7

CInt(*expresión*) → CEntero

Convierte cualquier *expresión* numérica o de cadena válida a un valor de tipo SHORT, es decir, un número entero entre -32.768 y 32.767; las fracciones se redondean.
Ejemplos:

- CInt('052') → 52
- CInt(22.8) → 23

CLng(*expresión*) → CLargo

Convierte cualquier *expresión* numérica o de cadena válida a un valor de tipo LONG, es decir, un número entero entre -2.147.483.648 a 2.147.483.647; las fracciones se redondean.
Ejemplos:

- CLng('052') → 52
- CLng(22.8) → 23

CSng(*expresión*) → CSimple

Convierte cualquier *expresión* numérica o de cadena válida a un valor de tipo SINGLE, es decir, un número en coma flotante de simple precisión: -3,402823E38 a -1,401298E-45 para valores negativos; 1,401298E-45 a 3,402823E38 para valores positivos. Si la *expresión* es una cadena, se usará como separador decimal el definido en el sistema.
Ejemplos:

- CSng(50 + 0.4) → 50.4
- CSng('-2.3') → -23
- CSng('5,7') → 5.7

CStr(*expresión*) → CCadena

Convierte cualquier *expresión* válida a un valor de tipo TEXT, es decir, una cadena. Para realizar la conversión se tendrán en cuenta los valores de localización del sistema.
Ejemplos:

- CStr(2.3) → '2,3'
- CStr(#5/12/01 21:20#) → '12/05/01 21:20:00'
- CStr(TRUE) → '-1'

Fix(*número*) → SinDec

Trunca el resultado de una *expresión* numérica para obtener la parte entera de la misma.
Ejemplos:

- Fix(2.3) → 2
- Fix(2.8) → 2
- Fix(-3.6) → -3

Int(*número*) → Ent

Trunca el resultado de una *expresión* numérica para obtener la parte entera de la misma. La diferencia con **Fix** es que **Int** devuelve, para *expresiones* negativas, el primer entero inferior menor o igual.
Ejemplos:

- Int(2.3) → 2
- Int(2.8) → 2
- Int(-3.6) → -4

Hex$(*número*)

Devuelve una cadena con la representación hexadecimal (esto es, en base 16) de un *número*.
Ejemplos:

- Hex(59)→ '3B'

Oct$(*número*)

Devuelve una cadena con la representación octal (esto es, en base 8) de un *número*.
Ejemplos:

- Oct(59) ➔ '73'

Str$(*número*) → Cad

Devuelve una cadena de texto que representa un *número* dado con dos particularidades: siempre se deja a la izquierda espacio para el signo, y siempre se usa el punto (.) como separador decimal.

- Str(9) ➔ ' 9'
- Str(-5.2) ➔ '-5.2'

Val(*cadena*)

Convierte una *cadena* en un valor numérico. A diferencia de las funciones **CByte**, **CInt**, **CLng**, **CSng** y **CDbl**, **Val** presenta las siguientes particularidades:

- No exige que la *cadena* represente un número válido: se recuperarán tantos dígitos como se puedan, eliminando espacios y tabuladores. Al encontrar algún carácter no válido, se detendrá la conversión. Si la *cadena* sólo contiene caracteres no válidos, se devolverá un 0.
- Se reconocen los prefijos &H (para hexadecimal) y &O (para octal).
- Sólo se reconoce el punto (.) como separador decimal.

Ejemplos:

- Val('error') ➔ 0
- Val('5.2') ➔ 5.2
- Val('&hff') ➔ 255
- Val('5 0,2 y 2') ➔ 50

Funciones de fecha y hora (librería VBA.DateTime)

Now() → Ahora

Devuelve la fecha (y hoja) actual del sistema.
Ejemplo:

- Now()

DateAdd(*intervalo*, *número*, *fecha*) → AgregFecha

Incrementa una *fecha* dada en un *intervalo* de tiempo cuya duración viene dada por *número*, que puede ser positivo (para incrementar) o negativo (para decrementar). Los valores válidos que pude tomar *intervalo* son (adviértase que son cadenas):

intervalo	'yyyy'	'q'	'm'	'y'	'd'
significado	Año	Trimestre	Mes	Día del año	Día
intervalo	'w'	'ww'	'h'	'n'	's'
significado	Día de la semana	Semana	Hora	Minuto	Segundo

A efecto de sumar (**DateAdd**) a una fecha, 'yyyy', 'y', 'd' y 'w' servirán para sumar días. Una propiedad importante de esta función es que nunca devuelve una fecha no válida.
Ejemplos:

- DateAdd('d', 10, #3/1/01#) ➔ #3/11/01#
- DateAdd('d', 31, #1/31/2001#) ➔ #3/3/2001#
- DateAdd('m', -1, #2/28/2001#) ➔ #1/31/2001#

DateDiff(*intervalo*, *fecha1*, *fecha2*[, *primerdíadelasemana*[, *primerasemanadelaño*]]) → DifFecha

Calcula la diferencia entre dos *fechas*, permitiendo elegir el *intervalo* de tiempo en el que se expresa dicha diferencia. Los valores que puede tomar *intervalo* están especificados en la función **DateAdd**. El parámetro opcional *primerdíadelasemana* indica qué día debe tomarse como primero de la semana (si queremos hacer referencia a los *intervalos* 'w' y 'ww') y puede tomar los siguientes valores:

primerdíadelasemana	0	1	2	3
significado	El que tome el sistema	Domingo	Lunes	Martes
primerdíadelasemana	4	5	6	7
significado	Miércoles	Jueves	Viernes	Sábado

Si se omite, el valor predeterminado es 1 (Domingo). El parámetro opcional *primerasemanadelaño* indica cómo debe tomarse la primera semana del año (si queremos hacer referencia al intervalo 'ww') y puede tomar los siguientes valores:

primerasemanadelaño	significado
0	La que tome el sistema
1	Aquella en la que se encuentra el 1 de Enero
2	Aquella que tenga al menos cuatro días en el nuevo año
3	Aquella que esté completamente incluida en el nuevo año

Si se omite, el valor predeterminado es 1 (aquella en la que se encuentra el 1 de Enero).
Ejemplo:

- DateDiff('d', #1/1/01#, #12/24/01#) ➔ 357

DatePart(*intervalo*, *fecha*[, *primerdíadelasemana*[, *primerasemanadelaño*]]) → ParcFecha

Devuelve una parte concreta de la *fecha*, indicada por *intervalo*. Este parámetro se encuentra descrito en **DateAdd**, y los parámetros opcionales *primerdíadelasemana* y *primerasemanadelaño* se explican en la definición de **DateDiff**, teniendo el mismo significado.
Ejemplos:

- DatePart('q', #5/1/01#) ➔ 2
- DatePart('w', #5/4/1974#, 2) ➔ 6

DateSerial(*año*, *mes*, *día*) → SerieFecha

Devuelve una fecha compuesta por el *año*, *mes* y *día* especificados. La hora será siempre 00:00:00.
Ejemplo:

- DateSeria(1981, 2, 14) ➔ #02/14/1981 00:00:00#

TimeSerial(*hora*, *minuto*, *segundo*) → SerieHora

Devuelve una hora compuesta por la *hora*, *minuto* y *segundo* indicados. La fecha será siempre 12/30/1899.
Ejemplo:

- TimeSerial(20, 50, 12) ➔ #20:50:12#

DateValue(*cadena*) → ValorFecha

Devuelve una fecha que se encuentra representada en una cadena. Además de reconocer el formato de fecha que se haya especificado en el sistema, también es capaz de reconocer literales (como nombre de meses). Si la cadena contiene información sobre la hora, ésta se ignora; la hora del resultado será siempre 00:00:00.
Ejemplo:

- DateValue('31 de diciembre de 2001') ➔ #12/31/2001#

TimeValue(*cadena*) → ValorHora

Devuelve una hora que se encuentra representada en una cadena. Además de reconocer el formato de hora que se haya especificado en el sistema, también es capaz de reconocer literales (como AM/PM). Si la cadena contiene información sobre la fecha, ésta se ignora; la fecha del resultado será siempre 31/12/1899.
Ejemplo:

- TimeValue('2:35 p.m.') ➔ #14:35:00#

Year(*fecha*) → Año

Devuelve un número (entre 100 y 9999) que corresponde al año de la *fecha* especificada.
Ejemplo:

- Year(#2/1/80 14:31:20#) ➔ 1980

Month(*fecha*) → Mes

Devuelve un número (entre 1 y 12) que corresponde al mes de la *fecha* especificada.
Ejemplo:

- Month(#2/1/80 14:31:20#) ➔ 2

Day(*fecha*) → Día

Devuelve un número (entre 1 y 31) que corresponde al día de la *fecha* especificada.
Ejemplo:

- Day(#2/1/80 14:31:20#) ➔ 1

Hour(*fecha*) → Hora

Devuelve un número (entre 0 y 23) que corresponde a la hora de la *fecha* especificada.
Ejemplo:

- Hour(#2/1/80 14:31:20#) ➔ 14

Minute(*fecha*) → Minuto

Devuelve un número (entre 0 y 59) que corresponde al minuto de la *fecha* especificada.
Ejemplo:

- Minute(#2/1/80 14:31:20#) ➔ 31

Second(*fecha*) → Segundo

Devuelve un número (entre 0 y 59) que corresponde al segundo de la *fecha* especificada.
Ejemplo:

- Second(#2/1/80 14:31:20#) ➔ 20

WeekDay(*fecha*[, *primerdíadelasemana*]) → DíaSemana

Devuelve el día de la semana de una *fecha* dada equivale a **DatePart('w', *fecha*[, *primerdíadelasemana*])**.
Ejemplo:

- WeekDay(#5/4/1974#, 2) ➔ 6

Funciones de información (librería VBA.Information)

IsDate(*expresión*) → EsFecha

Devuelve TRUE si el resultado de *expresión* puede convertirse a una fecha válida (en el caso de las cadenas, sólo si respetan el formato especificado en el sistema).
Ejemplos:

- IsDate('Antonio') ➔ FALSE
- IsDate(Now()) ➔ TRUE
- IsDate('30 de enero de 1984') ➔ FALSE
- IsDate('30/01/1984') ➔ TRUE

IsNull(*expresión*) → EsNulo

Devuelve TRUE sólo si el resultado de *expresión* toma el valor NULL (es decir, es nulo).
Ejemplo:

- IsNull(NULL) ➔ TRUE

IsNumeric(*expresión*) → EsNum

Devuelve TRUE si el resultado de *expresión* puede convertirse a un número válido.
Ejemplos:

- IsNumeric('Antonio') → FALSE
- IsNumeric(54+2) → TRUE
- IsNumeric('30 ptas') → FALSE

Funciones de decisión (librería VBA.Interaction)

Choose(*índice*, *opción1*[, *opción2*[, ...]]) → Elegir

Devuelve, de entre todas las *opciones* que se especifiquen, aquella que aparezca en la posición dada por *índice*. Ha de asegurarse que *índice* siempre tome un valor entre 1 y el número de opciones especificadas.
Ejemplo:

- Choose(2, 'sí', 'no', 'a veces') → 'no'

IIf(*expresión*, *valorcierto*, *valorfalso*) → SiInm

Evalúa una *expresión* lógica y devuelve el resultado de *valorcierto* o de *valorfalso* si se obtuvo TRUE o FALSE, respectivamente.
Ejemplo:

- IIf(IsNumeric('54'), 'número', 'error') → 'número'

Switch(*expresión1*, *valor1*[, *expresión2*, *valor2*[, ...]]) → Conmutador

Examina una serie parejas de *expresiones* lógicas y *valores*, y devuelve el primer *valor* cuya correspondiente *expresión* sea cierta. Hay que asegurar que alguna *expresión* siempre se verifica (por ejemplo, añadiendo una última pareja en la que la *expresión* sea TRUE).
Ejemplo:

- Switch(IsNumeric('x'), 1, IsNumeric('zz'), 2, TRUE, 3) → 3

Funciones matemáticas (librería VBA.Math)

Abs(*número*)

Devuelve el valor absoluto de un *número*.
Ejemplos:

- Abs(5) → 5
- Abs(-2) → 2

Atn(*número*) → ArcTg

Devuelve el arcotangente de un *número*.
Ejemplo:

- Atn(1) → 0.785398163397448

Cos(*número*)

Devuelve el coseno de un *número*.
Ejemplo:

- Cos(1) → 0.54030230586814

Exp(*número*)

Devuelve la exponencial de un *número*; es decir, e elevado a dicho *número*.
Ejemplo:

- Exp(1) → 2.71828182845905

Log(*número*) → Ln

Devuelve el logaritmo natural o neperiano de un *número*.
Ejemplo:

- Exp(1) ➔ 0

Sgn(*número*) → Signo

Devuelve el signo de un *número*; es decir, 1 para los valores positivos, -1 para los negativos, y 0 en otro caso.
Ejemplo:

- Sgn(100) ➔ 1

Sin(*número*) → Sen

Devuelve el seno de un *número*.
Ejemplo:

- Cos(1) ➔ 0.841470984807897

Sqr(*número*) → Raíz

Devuelve la raíz cuadrada de un número.
Ejemplo:

- Sqr(2) ➔ 1.4142135623731

Tan(*número*) → Tg

Devuelve la tangente de un *número*.
Ejemplo:

- Tan(1) ➔ 1.5574077246549

Funciones de cadena (librería VBA.Strings)

Asc(*cadena*) → CódigoCar

Devuelve el código interno que representa en el sistema al primer carácter de la *cadena*.
Ejemplos:

- Asc('A') ➔ 65
- Asc('ahora') ➔ 97

Chr$(*código*) → Car

Devuelve una cadena conteniendo al carácter cuyo *código* (véase la función **Asc**).
Ejemplo:

- Chr$(66) ➔ 'B'

Format$(*expresión*[, *formato*[, *primerdíadelasemana*[, *primerasemanadelaño*]]]) → Formato

Devuelve una cadena en la que representa el resultado de una *expresión* dada con un *formato* determinado (si éste se omite, se usará el formato por defecto del sistema para el tipo de dato de *expresión*). El parámetro *formato* es una cadena dentro de la cual, de aparecer las siguientes cadenas especiales, se sustituirá por el resultado que se indique:

opción de *formato*	significado
@	Tomar un carácter de la cadena *expresión*, o un espacio si no lo hay en esta posición
&	Tomar un carácter de la cadena *expresión*, o nada si no lo hay en esta posición
<	Se tomarán los caracteres de la cadena *expresión* en minúscula
>	Se tomarán los caracteres de la cadena *expresión* en mayúscula
!	Llenar los marcadores @ y & de derecha a izquierda en lugar de de

	izquierda a derecha (que es lo normal)
:	Mostrar el separador de hora que se haya especificado en el sistema
/	Mostrar el separador de fecha que se haya especificado en el sistema
c	Mostrar la fecha como ddddd y la hora como ttttt, en este orden
d	Mostrar el día como un número sin cero a la izquierda (0-31)
dd	Mostrar el día como un número con un cero a la izquierda (00-31)
ddd	Mostrar el día como abreviatura (Dom-Sáb)
dddd	Mostrar el día como nombre completo (Domingo-Sábado)
ddddd	Mostrar la fecha completa (día, mes y año) en formato corto
dddddd	Mostrar la fecha completa (día, mes y año) en formato largo
w	Mostrar el día de la semana como número (1-7)
ww	Mostrar la semana del año como número (1-54)
m	Mostrar el mes como un número sin cero a la izquierda (1-12)
mm	Mostrar el mes como un número con cero a la izquierda (01-12)
mmm	Mostrar el mes como abreviatura (Ene-Dic)
mmmm	Mostrar el mes como nombre completo (Enero-Diciembre)
q	Mostrar el trimestre del año como un número (1-4)
y	Mostrar el día del año como un número (1-366)
yy	Mostrar el año con dos dígitos (00-99)
yyyy	Mostrar el año completo (100-9999)
h	Mostrar la hora sin cero a la izquierda (0-23)
hh	Mostrar la hora con un cero a la izquierda (00-23)
n	Mostrar el minuto sin cero a la izquierda (0-59)
nn	Mostrar el minuto con un cero a la izquierda (0θ 59)
s	Mostrar el segundo sin cero a la izquierda (0-59)
ss	Mostrar el segundo con un cero a la izquierda (00-59)
ttttt	Mostrar la hora completa (hora, minutos y segundo)
AM/PM	Mostrar 'AM' o 'PM' según corresponda, y la hora de 0 a 12
am/pm	Mostrar 'am' o 'pm' según corresponda, y la hora de 0 a 12
A/P	Mostrar 'A' o 'P' según corresponda, y la hora de 0 a 12
a/p	Mostrar 'a' o 'p' según corresponda, y la hora de 0 a 12
AMPM	Mostrar las marcas AM y PM según indique el sistema, y la hora de 0 a 12
0	Mostrar un dígito del número, o un 0 si no lo hay
#	Mostrar un dígito del número, o nada si no lo hay
.	Mostrar la posición decimal, según se defina en el sistema
%	Mostrar el signo de porcentaje (y antes de formatear, multiplicar *expresión* por 100)
,	Mostrar el separador de millares, según se defina en el sistema
E- E+ e- e+	Mostrar la marca de formato de número científico, si corresponde
\	Mostrar el siguiente carácter, aunque sea especial

Puede especificarse, si la *expresión* es una cadena, una segunda sección separada de la primera mediante un punto y coma (;), de forma que el formato de esta segunda sección se aplicará cuando *expresión* sea NULL o una cadena de longitud 0. Si la *expresión* es numérica, pueden especificarse formatos con hasta cuatro secciones (separadas entre sí por puntos y comas), de forma que: cuando tenemos dos secciones la primera de aplica a valores positivos y ceros, y la segunda a negativos; cuando tenemos tres secciones la primera se aplica a valores positivos, la segunda a negativos y la tercera a los ceros; y cuando tenemos cuatro secciones se aplican las tres primeras de la misma forma que en el caso de tres, dejando la cuarta para las *expresiones* NULL.
Los parámetros opcionales *primerdíadelasemana* y *primerasemanadelaño* se explican en la definición de **DateDiff**, teniendo el mismo significado.
Ejemplos:

- Format$(#3/1/01#, 'dd de mmmm de yyyy') ➔ '01 de Marzo de 2001'
- Format$(-3.2;'000.00;(##0.00)') ➔ '3,20'
- Format$(4, '0000') ➔ '0004'
- Format$('hola', 'i@@@@@@!') ➔ 'ihola '
- Format$('hola', 'i@@@>@@@\!') ➔ 'i HOLA!'

InStr([*comienzo*,]*cadena*, *subcadena*[, *método*]) → EnCad

Localiza la posición en la que una *subcadena* aparece dentro de una *cadena*. Opcionalmente, puede indicarse antes de ambas la posición de *comienzo* de búsqueda dentro de la *cadena*; de omitir, se empezará a buscar por el primer carácter de ésta. La función devolverá la posición de la *subcadena* dentro de la *cadena* si la encuentra, o 0 si no la encuentra. Opcionalmente puede especificarse el *método* de comparación que puede ser: 0 (valor predeterminado) para realizar una comparación binaria, o bien 1 para no distinguir entre mayúsculas y minúsculas a la hora de buscar.
Ejemplos:

- InStr('hola', 'l') → 3
- InStr(2, 'Andalucía', 'A') → 0
- InStr(2, 'Andalucía', 'A', 1) → 4

LCase$(*cadena*) → Minús

Convierte una *cadena* a minúscula.
Ejemplo:

- LCase$('Hola') → 'hola'

Left$(*cadena*, *longitud*) → Izq

Extrae una subcadena de *longitud* dada de una *cadena*, tomando los caracteres por la izquierda de la misma.
Ejemplo:

- Left$('hola', 2) → 'ho'

Len(*cadena*) → Longitud

Devuelve el número de caracteres de una *cadena* dada.
Ejemplo:

- Len('hola') → 4

LTrim$(*cadena*) → RecortarIzq

Elimina los espacios que aparezcan a la izquierda de una *cadena* dada.
Ejemplo:

- LTrim$(' hola ') → 'hola '

Mid$(*cadena*, *comienzo*[, *longitud*]) → Medio

Extrae una subcadena de una *cadena* dada, comenzando a cortar en la posición de *comienzo* y terminando cuando se extraen los caracteres indicados en *longitud* (si se especificó) o se alcanza el final de la *cadena*.
Ejemplos:

- Mid$('hola', 2, 2) → 'ol'
- Mid$('hola', 2) → 'ola'

Right$(*cadena*, *longitud*) → Der

Extrae una subcadena de *longitud* dada de una *cadena*, tomando los caracteres por la derecha de la misma.
Ejemplo:

- Right$('hola', 2) → 'la'

RTrim$(*cadena*) → RecortarDer

Elimina los espacios que aparezcan a la derecha de una *cadena* dada.
Ejemplo:

- RTrim$(' hola ') → ' hola'

Space$(*longitud*) → Espacio

Devuelve una cadena de *longitud* dada formada únicamente por espacios.
Ejemplo:

- Space$(10) → ' '

StrComp(*cadena1*, *cadena2*[, *método*]) → CompCadena

Compara dos *cadenas* devolviendo –1 si *cadena1* < *cadena2*, 1 si *cadena1* > *cadena2* y 0 si *cadena1* = *cadena2*. Opcionalmente puede especificarse el *método* de comparación, de la misma forma que en **InStr**.
Ejemplos:

- StrComp('a', 'z') ➔ -1

StrConv(*cadena*, *método*) → ConvCadena

Realiza una conversión de la *cadena* especificada según el *método* indicado, que puede ser:

- 1: La cadena se pasa a mayúscula (equivale a **UCase$**)
- 2: La cadena se pasa a minúscula (equivale a **LCase$**)
- 3: La cadena se prepara como un nombre propio, poniendo en mayúscula la primera letra de dada palabra.

Ejemplo:

- StrConv('ciUDAD real', 3) ➔ 'Ciudad Real'

String$(*longitud*, *carácter*) → Cadena

Devuelve una cadena de *longitud* dada formada únicamente por el *carácter* especificado.
Ejemplo:

- Strings$(10, 'a') ➔ 'aaaaaaaaaa'

Trim$(*cadena*) → Recortar

Elimina los espacios que aparezcan a derecha e izquierda de una *cadena* dada. Equivale a **LTrim$(RTrim$(*cadena*))**.
Ejemplo:

- Trim$(' hola ') ➔ 'hola'

UCase$(*cadena*) → Mayús

Convierte una *cadena* a mayúscula.
Ejemplo:

- LCase$('Hola') ➔ 'HOLA'

Funciones de datos (librería Access.Application)

NOTA IMPORTANTE: Las funciones de esta librería sólo están disponibles si se usan desde el propio Access. Al usar consultas que las contengan desde otra aplicación que conecte con la base de datos Access, fallarán.

DAvg(*expresión*, *dominio*[, *criterio*]) → DProm

Calcula la media aritmética de los valores devueltos por *expresión* para cada registro procedente del *dominio*, que sólo puede ser una tabla o el nombre de una consulta previamente almacenada en la base de datos.
Opcionalmente puede especificarse un *criterio* de filtrado a chequear antes de realizar el cálculo de la función.
Equivale a la función agregada SQL **AVG**.
Ejemplos:

- DAvg('salario', 'empleado')
- DAvg('salario*1000', 'empleado', 'fchingr > #1/1/90#')

DCount(*expresión*, *dominio*[, *criterio*]) → DCont

Realiza la cuenta del número de valores devueltos por *expresión* para cada registro procedente del *dominio*, que sólo puede ser una tabla o el nombre de una consulta previamente almacenada en la base de datos.
Opcionalmente puede especificarse un *criterio* de filtrado a chequear antes de realizar el cálculo de la función.
Equivale a la función agregada SQL **COUNT**.
Ejemplo:

- DCount('salario', 'empleado')

DMax(*expresión*, *dominio*[, *criterio*]) → DMáx
DMin(*expresión*, *dominio*[, *criterio*]) → DMín

Toman el máximo y el mínimo, respectivamente, de los valores devueltos por *expresión* para cada registro procedente del *dominio*, que sólo puede ser una tabla o el nombre de una consulta previamente almacenada en la base de datos. Opcionalmente puede especificarse un *criterio* de filtrado a chequear antes de realizar el cálculo de la función. Equivalen a las funciones agregadas SQL **MAX** y **MIN**, respectivamente.
Ejemplos:

- DMax('salario', 'empleado')
- DMin('salario', 'empleado', 'nomclab = "INGENIERO"'

DStDev(*expresión*, *dominio*[, *criterio*]) → DDesvEst
DStDevP(*expresión*, *dominio*[, *criterio*]) → DDesvEstP
DVar(*expresión*, *dominio*[, *criterio*])
DVarP(*expresión*, *dominio*[, *criterio*])

Calculan la desviación estándar (sobre la muestra y la población) y la varianza (sobre la muestra y la población), respectivamente, de los valores devueltos por *expresión* para cada registro procedente del *dominio*, que sólo puede ser una tabla o el nombre de una consulta previamente almacenada en la base de datos. Opcionalmente puede especificarse un *criterio* de filtrado a chequear antes de realizar el cálculo de la función. Equivalen a las funciones agregadas SQL **STDEV**, **STDEVP**, **VAR** y **VARP**, respectivamente.

DSum(*expresión*, *dominio*[, *criterio*]) → DSuma

Calcula la suma de los valores numéricos devueltos por *expresión* para cada registro procedente del *dominio*, que sólo puede ser una tabla o el nombre de una consulta previamente almacenada en la base de datos. Opcionalmente puede especificarse un *criterio* de filtrado a chequear antes de realizar el cálculo de la función. Equivale a la función agregada SQL **SUM**.
Ejemplos:

- DMax('salario', 'empleado')
- DMin('salario', 'empleado', 'nomclab = "INGENIERO"'

DFirst(*expresión*, *dominio*[, *criterio*]) → DPrim
DLast(*expresión*, *dominio*[, *criterio*]) → DÚltimo

Toman el primer y último valor, respectivamente, de la lista de valores devueltos por *expresión* para cada registro procedente del *dominio*, que sólo puede ser una tabla o el nombre de una consulta previamente almacenada en la base de datos. Opcionalmente puede especificarse un *criterio* de filtrado a chequear antes de realizar el cálculo de la función.
Ejemplos:

- DFirst('numempl', 'empleado')

DLookUp(*expresión*, *dominio*[, *criterio*]) → DBúsq

Toma el valor de *expresión* para el único registro procedente del *dominio*, que sólo puede ser una tabla o el nombre de una consulta previamente almacenada en la base de datos. Opcionalmente puede especificarse un *criterio* de filtrado a chequear antes de realizar el cálculo de la función.
Ejemplos:

- DLookUp('nomempl', 'empleado', 'numempl = 1015')

Eval(*cadena*)

Evalúa una expresión almacenada en una *cadena* y devuelve el resultado.
Ejemplo:

- Eval('2+3*5') → 17

Nz(*expresión*[, *valorsinull*])

Devuelve el valor de *expresión* salvo si éste es NULL, en cuyo caso se devuelve *valorsinull*, si se especificó, o el valor vacío (Empty), que se puede convertir automáticamente a 0 o a la cadena vacía cuando se opera con él.
Ejemplo:

- Nz(NULL, 'nulo') → 'nulo'