Función integrada. Definición

Unha función, no contexto informático, defínese como un grupo de instrucións deseñadas para realizar unha tarefa específica, e que poden devolver un valor.

As funcións poden recibir valores, datos, a partires dos que se realizan as instrucións internas da función. Estes valores que se di que se lle pasan á función denomínanse *parámetros de entrada*.

A linguaxe T-SQL permite deseñar funcións, coñecidas como funcións definidas polo usuario, e que se verán nunha unidade didáctica posterior. As que aquí nos ocupan, as funcións integradas, son aquelas que xa veñen programadas no xestor, e que podemos dicir que nos aforran traballo, xa que non as temos que codificar nós.

Para usar unha función bastará con executalas nunha consulta, tendo en conta os parámetros que recibe e o tipo de dato que devolve. O feito de executar unha función coñecese como chamar unha función. Falaremos de usar ou chamar unha función, ou da execución ou chamada dunha función.

Para coñecer a clasificación das distintas funcións que fai o propio SQL Server, nada máis sinxelo que acceder ao SSMS, e no seu explorador de obxectos, nunha BD calquera escollemos o cartafol Programación, e baixamos dentro da árbore de cartafoles ata o cartafol de Funcións do sistema. Cada un dos cartafoles ten as funcións integradas de cada tipo cos parámetros que acepta e o valor que devolve.

Funcións de conversión de tipo

As funcións de conversión de tipos permiten converter unha expresión dun tipo de datos a outro de forma explícita, é dicir, indicando o tipo de dato no que imos converter.

As dúas que máis imos empregar nas nosas consultas son *cast* e *convert*.

Tanto a función *cast* como a función *convert* empréganse para a conversión de tipos de datos. A continuación explícase a súa sintaxe e exemplos do seu uso.

Función cast:

|  |  |
| --- | --- |
| Sintaxe función cast | |
| cast(expresión AS tipo\_dato) | |
| Parámetros de entrada | * expresión é calquera expresión válida como un valor constante, o nome dun campo, unha operación ou a chamada a unha función que devolve un valor. * tipo\_dato é o tipo de dato de destino. |
| Saída | * Devolve a expresión traducida ao tipo de dato de destino tipo\_dato. |

A función *cast* transforma a *expresión* indicada entre parénteses no *tipo\_dato* indicado despois do AS.

Utilizarémola principalmente nas operacións de concatenación de cadeas para transformar os datos que non son cadeas en tipo char ou varchar, para poder facer a concatenación, como podemos ver no exemplo 1.

* **Consulta de exemplo 1:** Amosar nunha soa columna de nome *datos\_empregados,* o número identificador e o nome completo de cada empregado co formato:

*número identificador - apelido1 apelido2, nome*

Función convert:

|  |  |
| --- | --- |
| Sintaxe función convert | |
| convert(tipo\_dato, expresión [,estilo]) | |
| Parámetros de entrada | * tipo\_dato é o tipo de dato de destino. * expresión é calquera expresión válida como un valor constante, o nome dun campo, unha operación ou a chamada a unha función que devolve un valor. * estilo é unha expresión de tipo enteiro que especifica como convert traducirá expresión. |
| Saída | * Devolve a expresión traducida ao tipo de dato de destino tipo\_dato. |

A función *convert* transforma a *expresión* no *tipo\_dato* indicado como primeiro parámetro xusto despois do parénteses.

O valor do terceiro parámetro *estilo* dependerá do tipo de dato que indiquemos. Utilizarase principalmente nos tipos de dato de datas. Permítenos escoller o formato co que representar unha data como dd/mm/aaaa, dd-mm-aaaa, dd mes aa. Verémolo no seguinte exemplo.

* **Consulta de exemplo 2:** Amosar nunha soa columna o nome de todas as actividades e o día que comezan co formato: nome\_actividade comeza o data\_inicio\_da\_actividade

Funcións de cadea de caracteres

As seguintes funcións realizan unha operación sobre unha cadea de caracteres e devolven un valor alfanumérico (de cadea) ou un valor numérico, segundo a tarefa da función.

Función ascii:

|  |  |
| --- | --- |
| Sintaxe función ascii | |
| ascii(expresión) | |
| Parámetros de entrada | * expresión é un tipo de dato char ou varchar. |
| Saída | * Devolve un número enteiro que se corresponde cun código ASCII. |

A función *ascii* devolve o valor do código ASCII do carácter que está máis á esquerda da expresión alfanumérica que se indica como parámetro entre parénteses.

Función char:

|  |  |
| --- | --- |
| Sintaxe función char | |
| char(expresión) | |
| Parámetros de entrada | * expresión é un enteiro entre 0 e 255. |
| Saída | * Devolve un carácter (char(1)), o carácter ASCII que se corresponde con expresión. * Devolve NULL no caso de que o número pasado estea fóra do intervalo [0-255]. |

A función *char* recibe como parámetro un número enteiro entre 0 e 255. Transforma o número que se lle pasa no carácter que lle corresponde a ese número no código ASCII. Devolverá NULL se lle pasamos á función un número que estea fóra do intervalo.

* **Consulta de exemplo 3:** Amosar o uso das funcións ascii e char para a letra A. A consulta non se fai sobre ningunha táboa, neste caso, empregamos a instrución SELECT só para amosar o resultado, como se fose un print.

Función left:

|  |  |
| --- | --- |
| Sintaxe función left | |
| left(expresión, N) | |
| Parámetros de entrada | * expresión é unha cadea de datos binarios ou alfanuméricos. * N é o número de caracteres que queremos extraer da parte esquerda da cadea. |
| Saída | * Devolve varchar se expresión é de tipo de datos non Unicode. * Devolve nvarchar cando expresión é de tipo Unicode. |

A función *left* devolve os N caracteres máis á esquerda da expresión pasada como primeiro parámetro.

* **Consulta de exemplo 4:** Obter a listaxe abreviada das catro primeiras letras dos nomes das actividades.

Función len:

|  |  |
| --- | --- |
| Sintaxe función len | |
| len(expresión) | |
| Parámetros de entrada | * expresión é unha cadea de datos binarios ou alfanuméricos. * expresión pode ser un valor constante, unha variable ou unha columna de datos binarios ou de caracteres. |
| Saída | * Devolve o número de caracteres da cadea pasada como parámetro, descontando os espazos en branco do final, se existen. * O tipo de datos devolto será int, exceptuando nos casos de que expresión sexa varchar(max), nvarchar(max) ou varbinary(max), que o tipo de dato devolto será bigint. |

Devolve o número de caracteres da expresión, sen contar os espazos en branco da dereita da cadea.

* **Consulta de exemplo 5:** Obter o nome e número de caracteres dos nomes dos empregados.

Función lower:

|  |  |
| --- | --- |
| Sintaxe función lower | |
| lower(expresión) | |
| Parámetros de entrada | * expresión é unha cadea de datos binarios ou alfanuméricos. |
| Saída | * Devolve a expresión de caracteres, convertida en minúsculas. O tipo da cadea devolta será varchar ou nvarchar. |

* **Consulta de exemplo 6:** Obter os nomes dos empregados tal como están gardados na BD, e nunha segunda columna os nomes dos empregados en minúsculas.
* **Consulta de exemplo 7:** Nomes completos dos socios en minúsculas. Primeiro pasando a minúsculas cada campo, e na segunda columna pasando a minúsculas despois de facer a concatenación dos campos.

Función ltrim:

|  |  |
| --- | --- |
| Sintaxe función ltrim | |
| ltrim(expresión) | |
| Parámetros de entrada | * expresión é unha cadea de datos binarios ou alfanuméricos. |
| Saída | * Devolve a expresión de caracteres , despois de quitarlle os espazos en branco iniciais. |

* **Consulta de exemplo 8:** Amosar unha cadea con espazos á esquerda na primeira columna, e a mesma cadea sen espazos á esquerda.

Función nchar:

|  |  |
| --- | --- |
| Sintaxe función nchar | |
| nchar(expresión) | |
| Parámetros de entrada | * expresión é un número enteiro positivo. |
| Saída | * Devolve o carácter Unicode correspondente ao código enteiro pasado. |

* **Consulta de exemplo 9:** Amosar o carácter unicode do código 248.

Función replace:

|  |  |
| --- | --- |
| Sintaxe función replace | |
| replace(cadea, subcadea\_buscada, cadea\_substitución) | |
| Parámetros de entrada | * cadea é unha cadea de datos binarios ou alfanuméricos. * subcadea\_buscada é o conxunto de caracteres que se vai buscar. Pode ser de datos binarios ou de caracteres, pero no pode ser unha cadea baleira (''). * cadea\_substitución é a cadea coa que se vai substituír a subcadea buscada. |
| Saída | * Devolve a cadea inicial modificada coa subcadea\_buscada cambiada pola cadea\_substitución, tantas veces como aparecese. |

* **Consulta de exemplo 10:** Obtersulta nome das actividades e nunha segunda columna o nome das actividades sen acento gráfico no a.
* **Consulta de exemplo 11:** Nesta consulta obtemos o nome das actividades e nunha segunda columna o nome das actividades sen acento gráfico en ningunha vogal.

Función right:

|  |  |
| --- | --- |
| Sintaxe función right | |
| right(expresión, N) | |
| Parámetros de entrada | * expresión é unha cadea de datos binarios ou alfanuméricos. * N é o número de caracteres que queremos extraer da parte dereita da cadea. |
| Saída | * Devolve varchar se expresión é de tipo de datos non Unicode. * Devolve nvarchar cando expresión é de tipo Unicode. |

A función *right* devolve os N caracteres máis á dereita da expresión pasada como primeiro parámetro.

* **Consulta de exemplo 12:** Obter a listaxe abreviada das catro últimas letras dos nomes das actividades.

Función rtrim:

|  |  |  |
| --- | --- | --- |
| **Sintaxe función rtrim** | |  |
| rtrim(expresión) | |  |
| **Parámetros de entrada** | expresión é unha cadea de datos binarios ou alfanuméricos. | |
| **Saída** | Devolve a expresión de caracteres, despois de quitarlle os espazos en branco iniciais. | |

* **Consulta de exemplo 13:** Amosar unha cadea con espazos á dereita na primeira columna, e a mesma cadea sen espazos á dereita. Para poder verificalo engadimos un punto ao final da cadea.

Función substring:

|  |  |
| --- | --- |
| Sintaxe función substring | |
| substring(expresión, pos\_ini, lonxitude) | |
| Parámetros de entrada | * expresión é unha cadea de datos binarios ou alfanuméricos. * pos\_ini é un enteiro que especifica a posición do primeiro carácter a devolver. Hai que ter en conta que o primeiro carácter da cadea ocupa a posición 1. * lonxitude é un enteiro que indica o número de caracteres a devolver. |
| Saída | * Devolve unha subcadea de expresión, empezando pola posición pos\_ini e de tantos caracteres como indique a lonxitude. |

* **Consulta de exemplo 14:** Obter os segundo, terceiro, cuarto e quinto caracteres do nome das cotas.

Función unicode:

|  |  |
| --- | --- |
| Sintaxe función unicode | |
| unicode(expresión) | |
| Parámetros de entrada | * expresión é un tipo de dato char ou varchar. |
| Saída | * Devolve un número enteiro que se corresponde cun código Unicode. |

* **Consulta de exemplo 15:** Amosar o código unicode do símbolo ø.

Función upper:

|  |  |
| --- | --- |
| Sintaxe función upper | |
| upper(expresión) | |
| Parámetros de entrada | * expresión é unha cadea de datos binarios ou alfanuméricos. |
| Saída | * Devolve a expresión de caracteres pasada á función, convertida en maiúsculas. O tipo da cadea devolta será varchar ou nvarchar. |

* **Consulta de exemplo 16:** Observacións das actividades tal como están gardadas e nunha segunda columna as mesmas observacións en maiúsculas.
  + 1. Funcións de tratamento de datas

Imos empezar lembrando os tipos de datos de data e hora:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Tipo de datos | Formato | Intervalo | Precisión | Axuste de zona horaria |
| time | hh:mm:ss[. nnnnnnn] | De 00:00:00.0000000 a 23:59:59.9999999 | 100 nanosegundos | Non |
| date | aaaa-mm-dd | |  |  | | --- | --- | |  |  |   De 0001-01-01 a 9999-12-31 | 1 día | Non |
| smalldatetime | aaaa-mm-dd hh:mm:ss | De 1900-01-01 a 2079-06-06 | 1 minuto | Non |
| datetime | aaaa-mm-dd hh:mm:ss[. nnn] | |  |  | | --- | --- | |  |  |   De 1753-01-01 a 9999-12-31 | 0,00333 segundos | Non |
| datetime2 | aaaa-mm-dd hh:mm:ss[. nnnnnnn] | De 0001-01-01 00:00:00.0000000 a 9999-12-31 23:59:59.9999999 | 100 nanosegundos | Non |
| datetimeoffset | aaaa-mm-dd hh:mm:ss[. nnnnnnn] [+|-]hh:mm | De 0001-01-01 00:00:00.0000000 a 9999-12-31 23:59:59.9999999 (en UTC) | 100 nanosegundos | Si |

Funcións que obteñen valores de data e hora do sistema

Hai que aclarar antes de nada, que os valores de data e hora, a súa exactitude, dependen da versión de Windows na que se executa a instancia de SQL Server.

Función sysdatetime:

|  |  |
| --- | --- |
| Sintaxe función sysdatetime | |
| sysdatetime() | |
| Parámetros de entrada | * Ningún. |
| Saída | * Devolve en valor datetime2(7) a data e hora do equipo no que se está a executar a instancia de SQL Server. Sen axuste de zona horaria. |

Función sysdatetimeoffset:

|  |  |
| --- | --- |
| Sintaxe función sysdatetimeoffset | |
| sysdatetimeoffset() | |
| Parámetros de entrada | * Ningún. |
| Saída | * Devolve en valor datetimeoffset(7) a data e hora do equipo no que se está a executar a instancia de SQL Server. Con axuste de zona horaria. |

Función sysutcdatetime:

|  |  |
| --- | --- |
| Sintaxe función sysutcdatetime | |
| sysutcdatetime() | |
| Parámetros de entrada | * Ningún. |
| Saída | * Devolve en valor datetime2(7) a data e hora do equipo no que se está a executar a instancia de SQL Server. A data e hora devólvense como hora universal coordinada (UTC). |

Función getdate:

|  |  |
| --- | --- |
| Sintaxe función getdate | |
| getdate() | |
| Parámetros de entrada | * Ningún. |
| Saída | * Devolve en valor datetime a data e hora do equipo no que se está a executar a instancia de SQL Server. Sen axuste de zona horaria. |

Función getutcdate:

|  |  |
| --- | --- |
| Sintaxe función getutcdate | |
| getutcdate() | |
| Parámetros de entrada | * Ningún. |
| Saída | * Devolve en valor datetime a data e hora do equipo no que se está a executar a instancia de SQL Server. A data e hora devólvense como hora universal coordinada (UTC). |

* **Consulta de exemplo 17:** Amosar a data do sistema empregando as distintas funcións.

Funcións que obteñen partes de data e hora

Función datename:

|  |  |
| --- | --- |
| Sintaxe función datename | |
| datename(parte\_da\_data\_hora, data\_hora) | |
| Parámetros de entrada | * parte\_da\_data\_hora é a parte da data que se quere obter. Os valores poden ser completos ou empregar as abreviaturas indicadas entre parénteses: * year (yy, yyyy) * quarter (qq, q) * month (mm, m) * dayofyear (dy, y) * day (dd, d) * week (wk, ww) * weekday (dw, w) * hour (hh) * minute (mi, n) * second (ss, s) * millisecond (ms) * microsecond (mcs) * nanosecond (ns) * TZoffset (tz) * ISO\_WEEK (ISOWK, ISOWW) * Data ou hora da que se vai extraer unha parte. |
| Saída | * Devolve unha cadea de caracteres que representa o parámetro parte\_da\_data\_hora, da data\_hora pasada á función. * O valor devolto depende do idioma definido con SET LANGUAGE. |

* **Consulta de exemplo 18:** Amosar uso da función *datename* para obter o nome de distintas partes dunha data e hora concreta: nanosegundos, segundos, minutos, hora, día, semana, mes e ano da data '20-01-2015 17:28:59.1945901'.

Función datepart:

|  |  |
| --- | --- |
| Sintaxe función datepart | |
| datepart(parte\_da\_data\_hora, data\_hora) | |
| Parámetros de entrada | * parte\_da\_data\_hora é a parte da data que se quere obter. Os valores son os mesmos que os da función datename anterior. * Data ou hora da que se vai extraer unha parte. |
| Saída | * Devolve un enteiro que representa o parámetro parte\_da\_data\_hora, da data\_hora pasada á función. * O valor devolto depende do idioma definido con SET LANGUAGE. |

* **Consulta de exemplo 19:** Amosar o uso da función *datepart* para obter o as distintas partes dunha data e hora concreta: nanosegundos, segundos, minutos, hora, día, semana, mes e ano da data '20-01-2015 17:28:59.1945901'.

Función day:

|  |  |
| --- | --- |
| Sintaxe función day | |
| day(data\_hora) | |
| Parámetros de entrada | * data\_hora que pode ser unha expresión, unha columna, unha variable ou unha constante de tipo data\_hora. |
| Saída | * Devolve un enteiro que representa o día (día do mes) da data\_hora especificada. * Devolve o mesmo que a funcións datepart(day, data\_hora). |

Función month:

|  |  |
| --- | --- |
| Sintaxe función month | |
| month(data\_hora) | |
| Parámetros de entrada | * data\_hora que pode ser unha expresión, unha columna, unha variable ou unha constante de tipo data\_hora. |
| Saída | * Devolve un enteiro que representa o número do mes da data\_hora especificada. * Devolve o mesmo que a funcións datepart(month, data\_hora). |

Función year:

|  |  |
| --- | --- |
| Sintaxe función year | |
| year(data\_hora) | |
| Parámetros de entrada | * data\_hora que pode ser unha expresión, unha columna, unha variable ou unha constante de tipo data\_hora. |
| Saída | * Devolve un enteiro que representa o ano da data\_hora especificada. * Devolve o mesmo que a funcións datepart(year, data\_hora). |

* **Consulta de exemplo 20:** Devolver o día, mes e ano da data '20-01-2015 17:28:59.1945901'.
* **Consulta de exemplo 21:** Obter a data de inicio das actividades nunha primeira columna, e nas seguintes amosarase o día, mes e ano desa mesma data de inicio.

Funcións que obteñen a data e hora dadas as partes

Función datefromparts:

|  |  |
| --- | --- |
| Sintaxe función datefromparts | |
| datefromparts(ano, mes, día) | |
| Parámetros de entrada | * Todos os parámetros de entrada son enteiros que especifican o que o nome indica. |
| Saída | * Devolve un valor date para o ano, mes e día especificados entre parénteses. |

Función datetimefromparts:

|  |  |
| --- | --- |
| Sintaxe función datetimefromparts | |
| datetimefromparts(ano, mes, día, hora, minuto, segundos, milisegundos) | |
| Parámetros de entrada | * Todos os parámetros de entrada son enteiros que especifican o que o nome indica. |
| Saída | * Devolve un valor datetime da data\_hora formado pola partes indicadas entre os parénteses. |

* **Consulta de exemplo 22:** Obter a data a partir do día 14, do mes 4 e do ano 1999 nunha primeira columna. Nunha segunda columna obterase a data formada polo día 14, do mes 4, do ano 1999, hora 21, minuto 15, 58 segundos e 2 milisegundos.

Funcións que obteñen diferenzas de data e hora

Función datediff:

|  |  |
| --- | --- |
| Sintaxe función datediff | |
| datediff(parte\_da\_data\_hora, data\_ini, data\_fin) | |
| Parámetros de entrada | * parte\_da\_data\_hora é a unidade de tempo na que queremos calcular a diferenza entre as datas pasadas á función, anos, días, meses... Os valores poden ser completos ou empregar as abreviaturas indicadas entre parénteses: * year (yy, yyyy) * quarter (qq, q) * month (mm, m) * dayofyear (dy, y) * day (dd, d) * week (wk, ww) * weekday (dw, w) * hour (hh) * minute (mi, n) * second (ss, s) * millisecond (ms) * microsecond (mcs) * nanosecond (ns) * data\_ini é a data de inicio do rango do que queremos calcular a diferenza en anos, días, meses, ou o que indiquemos en parte\_da\_data\_hora. * data\_fin é a data de fin di rango do que queremos calcular a diferenza en anos, días, meses, ou o que indiquemos en parte\_da\_data\_hora. |
| Saída | * Devolve a diferenza entre a data de inicio e a data de fin calculada en anos, días, meses, ou o que indiquemos en parte\_da\_data\_hora. |

* **Consulta de exemplo 23:** Obter as datas de inicio e fin, e duración en días, meses, semanas e horas de todas as actividades.

Funcións que modifican valores de data hora

Función dateadd:

|  |  |
| --- | --- |
| Sintaxe función dateadd | |
| dateadd(parte\_da\_data\_hora, N, data\_hora) | |
| Parámetros de entrada | * parte\_da\_data\_hora é a unidade de tempo que lle queremos engadir, ou quitar á data pasada á función, anos, días, meses... Os valores son os mesmos que os da función datediff anterior. * N, se é positivo será o que lle engadimos á data\_hora, e se é negativo será o que se lle quita á data\_hora. * data\_hora que queremos atrasar ou adiantar. |
| Saída | * Devolve a data\_hora retrasada en N, ou adiantada en -N anos, días, meses, ou o que indiquemos en parte\_da\_data\_hora. |

* **Consulta de exemplo 24:** Obter a listaxe coa data de inicio das actividades na primeira columna. Nas seguintes aparecerán: a data de inicio adiantada 1 ano, a data de inicio adiantada 3 meses, a data de inicio retrasada 4 días e a data de inicio retrasada 2 horas.

Funcións que validan valores de data e hora

Función isdate:

|  |  |
| --- | --- |
| Sintaxe función isdate | |
| isdate(expresión) | |
| Parámetros de entrada | * expresión é unha cadea de caracteres. |
| Saída | * Devolve 1 se expresión é un valor de tipo time, date ou datetime válido, se non o é devolve 0. * Tamén devolve 0 se a expresión é un valor datetime2. |

* **Consulta de exemplo 25:** Comprobar a validez de diferentes datas. As columnas co nome *data\_erronea* teñen un formato incorrecto. Poñer vos os datos.

Funcións matemáticas

Función ceiling:

|  |  |
| --- | --- |
| Sintaxe función ceiling | |
| ceiling(expresión) | |
| Parámetros de entrada | * expresión numérica. |
| Saída | * Devolve o enteiro máis pequeno maior ou igual que a expresión numérica especificada, é dicir, fai unha aproximación por exceso. |

Función floor:

|  |  |
| --- | --- |
| Sintaxe función floor | |
| floor(expresión) | |
| Parámetros de entrada | * expresión numérica. |
| Saída | * Devolve o enteiro máis grande que sexa menor ou igual que a expresión numérica especificada, é dicir, fai unha aproximación por defecto |

* **Consulta de exemplo 26:** Aproximar por exceso e por defecto o número 134.873.

Función power:

|  |  |
| --- | --- |
| Sintaxe función power | |
| power(expresión, y) | |
| Parámetros de entrada | * expresión é de tipo float. * y é a potencia á que se eleva a expresión. |
| Saída | * Devolve o valor de expresión elevado á potencia y, especificada. É dicir, obteremos expresióny |

* **Consulta de exemplo 27:** Realizar varias potencias poñendo vos os exemplos.

Función sign:

|  |  |
| --- | --- |
| Sintaxe función sign | |
| sign(expresión) | |
| Parámetros de entrada | * expresión numérica. |
| Saída | * Devolve +1 se o signo da expresión é positivo, 0 se expresión é 0 ou -1 se a expresión é negativa. |

* **Consulta de exemplo 28:** Verificar se as expresións 18, -5, 0 e NULL, son positivas ou negativas.

Función sqrt:

|  |  |
| --- | --- |
| Sintaxe función sqrt | |
| sqrt(expresión) | |
| Parámetros de entrada | * expresión é de tipo float. |
| Saída | * Devolve a raíz cadrada de expresión. |

Función square:

|  |  |
| --- | --- |
| Sintaxe función square | |
| square(expresión) | |
| Parámetros de entrada | * expresión é de tipo float. |
| Saída | * Devolve o cadrado de expresión. |

* **Consulta de exemplo 29:** Devolver a raíz cadrada do número 49, e o cadrado do número 7.

Funcións do sistema

Función isnull:

|  |  |
| --- | --- |
| Sintaxe función isnull | |
| isnull(expresión, valor\_de\_substitución) | |
| Parámetros de entrada | * expresión que se vai validar se é nula. Pode ser de calquera tipo. * valor\_de\_substitución é polo que se vai substituír expresión cando teña valor null. |
| Saída | * Devolve o valor que teña expresión, salvo se é nulo, caso no que devolverá o valor\_de\_substitución. |

* **Consulta de exemplo 30:** Obter o nome das actividades e as súas observacións. Nas actividades que non teñan observacións deberá aparecer a frase 'Sen observacións'.

Función isnumeric:

|  |  |
| --- | --- |
| Sintaxe función isnumeric | |
| isnumeric(expresión) | |
| Parámetros de entrada | * expresión que se vai evaluar se é numérica. |
| Saída | * Determina se unha expresión é un tipo numérico válido. Devolve 1 se expresión é de un tipo de dato numérico válido, e se non o era devolve 0. |

* **Consulta de exemplo 31:** Comprobar se o contido das columnas seguintes da táboa SOCIO son ou non numéricas.

Función @@language:

|  |  |
| --- | --- |
| Sintaxe función @@language | |
| @@language | |
| Saída | * Devolve o nome do idioma en uso. |

Función @@max\_connections:

|  |  |
| --- | --- |
| Sintaxe función @@max\_connections | |
| @@max\_connections | |
| Saída | * Devolve o número máximo de conexións de usuario simultáneas que se permitan nunha instancia de SQL Server. |

Función @@servername:

|  |  |
| --- | --- |
| Sintaxe función @@servername | |
| @@servername | |
| Saída | * Devolve o nome do servidor local no que se executa SQL Server. |

Función @@spid:

|  |  |
| --- | --- |
| Sintaxe función @@spid | |
| @@spid | |
| Saída | * Devolve o identificador (Id) de sesión do proceso de usuario actual. |

Función @@textsize:

|  |  |
| --- | --- |
| Sintaxe función @@textsize | |
| @@textsize | |
| Saída | * Devolve o valor actual da opción TEXTSIZE, que especifica o tamaño dos datos varchar(max), nvarchar(max), varbinary(max), text, ntext e image devoltos por unha instrución SELECT. |

Función @@version:

|  |  |
| --- | --- |
| Sintaxe función @@version | |
| @@version | |
| Saída | * Devolve información do sistema e a compilación para a instalación actual de SQL Server. |

* **Consulta de exemplo 32:** Amosar os valores que devolven as funcións do sistema que acabamos de expoñer: @@language, @@max\_connections, @@servername, @@spid, @@textsize e @@version.

Funcións diversas

Funcións lóxicas

Función case:

|  |  |
| --- | --- |
| Sintaxe función case | |
| **Sintaxe1:**  CASE expresión  WHEN valor THEN result1  [WHEN valor THEN result2  …  WHEN valor THEN resultN]  [ELSE outro\_resultado ]  END  **Sintaxe2:**  CASE  WHEN condición THEN result  [WHEN condición THEN result2  …  WHEN condición THEN resultN]  [ELSE outro\_resultado ]  END | |
| Parámetros de entrada | * expresión, é a expresión avaliada. |
| Saída | * Na primeira sintaxe avalía unha expresión e compáraa cunha lista de valores e devolve un dos resultados posibles. * Na segunda sintaxe avalía cada unha das condicións e devolve o resultado da condición verdadeira. |

* **Consulta de exemplo 33:** Repetimos a consulta 30 pero empregando CASE.

Función coalesce:

|  |  |
| --- | --- |
| Sintaxe función coalesce | |
| coalesce(expresión1 [,expresión2,...,expresiónN) | |
| Parámetros de entrada | * as expresións poden ser de calquera tipo |
| Saída | * Avalía os argumentos en orde e devolve o valor actual da primeira expresión que non é NULL. * Se todos os argumentos pasados á función son NULL, coalesce devolve NULL. * A función COALESCE é un método abreviado de CASE. Internamente o servidor reescribe o código de COALESCE como o CASE seguinte:   CASE  WHEN expresión1 is not null THEN expresión1  WHEN expresión2 is not null THEN expresión2  …  ELSE expresiónN  END |

* **Consulta de exemplo 34:** Nome, apelidos e teléfono de todos os socios. Deberá amosarse o teléfono1, se o ten, senón o teléfono2, e se non ten teléfono amosarase a frase 'Sen teléfono'. A vantaxe de coalesce con respecto a isnull, e que podemos poñer máis de 2 argumentos.

Función iif:

|  |  |
| --- | --- |
| Sintaxe función iif | |
| iif(expresión, valor\_true, valor\_false) | |
| Parámetros de entrada | * expresión booleana válida. * valor\_true é o valor que se vai devolver se a expresión que se avalía é verdadeira. * valor\_false é o valor que se vai devolver se a expresión que se avalía é falsa. |
| Saída | * Devolve un dos valores, dependendo de se a expresión se avalía como verdadeira (true) ou como falsa (false). |

* **Consulta de exemplo 35:** NIF e cargo dos empregados. Se é profesor/a aparecerá PROFESORADO como cargo, senón ADMINISTRATIVO.

Funcións de tipos de datos

Función datalength:

|  |  |
| --- | --- |
| Sintaxe función datalength | |
| datalength(expresión) | |
| Parámetros de entrada | * expresión é calquera tipo de dato. |
| Saída | * Devolve o número de bytes usados para representar calquera expresión. |

* **Consulta de exemplo 36:** Cantidade de bytes do número, nif e nome do socio número 1001.

Función @@identity:

|  |  |
| --- | --- |
| Sintaxe función @@identity | |
| @@identity | |
| Saída | * Devolve o último valor de identidade xerado para calquera táboa na sesión actual. |

* **Consulta de exemplo 37:** Para comprobar o funcionamento desta función crearemos unha táboa temporal #t1 cun campo identity. Despois de engadir dúas filas na táboa, comprobamos coa función o último valor de identidade xerado na BD.

Función ident\_current:

|  |  |
| --- | --- |
| Sintaxe función ident\_current | |
| ident\_current(táboa\_vista) | |
| Parámetros de entrada | * táboa é o nome da táboa da que queremos coñecer o seu último valor de identidade. |
| Saída | * Devolve o último valor de identidade xerado para a táboa ou vista especificada. |

* **Consulta de exemplo 38:** Creamos unha nova táboa temporal #t2 e engadímoslle unha fila e despois comprobamos o valor de identidade da táboa #t1, e tamén o último valor de identidade engadido na BD. É importante poñer entre comiñas simples o nóme da táboa que lle pasamos entre parénteses á función.

Funciones de metadatos

Función col\_name:

|  |  |
| --- | --- |
| Sintaxe función col\_name | |
| col\_name(táboa\_id, columna\_id) | |
| Parámetros de entrada | * táboa\_id é o número de identificación da táboa que contén á columna. * columna\_id é o número de identificación da columna, da posición que ocupa na táboa. |
| Saída | * Devolve o nome dunha columna a partires do número de identificación da táboa e do número de identificación de columna pasados á función. |

Función db\_id:

|  |  |
| --- | --- |
| Sintaxe función db\_id | |
| db\_id(nome\_bd) | |
| Parámetros de entrada | * nome\_bd é o nome da BD da que queremos coñecer o identificador. |
| Saída | * Devolve o número identificador da base de datos. |

Función db\_name:

|  |  |
| --- | --- |
| Sintaxe función db\_name | |
| db\_name(id\_bd) | |
| Parámetros de entrada | * id\_db é o número identificador da bd da que queremos coñecer o nome. |
| Saída | * Devolve o nome da BD que se corresponde co identificador pasado á función. |

Función object\_id:

|  |  |
| --- | --- |
| Sintaxe función object\_id | |
| object\_id(nome\_obxecto[, tipo\_obxecto]) | |
| Parámetros de entrada | * nome\_obxecto é o obxecto do que queremos coñecer o identificador. * tipo\_obxecto é un tipo de obxecto de BD. |
| Saída | * Devolve o número de identificación do obxecto de base de datos. |

Función object\_name:

|  |  |
| --- | --- |
| Sintaxe función object\_name | |
| object\_name(id\_obxecto[, database\_id]) | |
| Parámetros de entrada | * id\_obxecto é o identificador do obxecto do que queremos saber o nome. * database\_id é o identificador da bd onde se vai buscar o obxecto. |
| Saída | * Devolve o nome do obxecto do identificador , da bd especificada. |

Función parsename:

|  |  |
| --- | --- |
| Sintaxe función parsename | |
| parsename(nome\_obxecto, parte\_obxecto) | |
| Parámetros de entrada | * nome\_obxecto é o nome do obxecto do que se desexa recuperar a parte especificada. * parte\_obxecto é a parte do obxecto de bd que se vai devolver. Pode ter un dos valores seguintes: * 1 = Nome do obxecto * 2 = Nome do esquema * 3 = Nome da base de datos * 4 = Nome do servidor |
| Saída | * Devolve a parte especificada dun nome de obxecto. |

* **Consulta de exemplo 39:** Nesta consulta amosamos o uso das funcións de metadatos aquí explicadas.

Consultas Propostas

Realizar o código T-SQL adecuado para obter a información que se pide en cada unha das consultas propostas na BD EMPRESA. Aínda que non se indique, nos resultados todas as columnas terán un nome que identifique correctamente a información que amosan.

* Consultas propostas na BD EMPRESA.
* **Proposta 1.** Desexamos coñecer o código ASCII da vogal E. Na consulta deberás devolver nunha columna a vogal en maiúscula, e nunha segunda o código ASCII que lle corresponde.
* **Proposta 2.** Consulta que devolve o carácter que lle corresponde aos seguintes códigos ASCII: 70, 80, 90.
* **Proposta 3.** Queremos obter unha listaxe que en cada liña teña o seguinte texto: *O empregado con nome e apelidos X ten que acadar unha cota de vendas anual de Y*. Sendo X o nome e os apelidos do empregado, e Y a cota de vendas. É importante fixarse no segundo apelido. A listaxe terá por título *Empregados e cotas.*
* **Proposta 4.** Consulta que devolva as datas nas que se contrataron empregados. O formato das diferentes datas será dd-mm-aaaa e o nome da columna *Datas de contratación.*
* **Proposta 5.** Queremos obter un nome abreviado das sucursais. Ese nome comporase polos tres primeiros caracteres da cidade, os dous últimos da rexión e separado por un guión baixo, o número de caracteres do nome da cidade.
* **Proposta 6.** Queremos obter un nome abreviado dos produtos. Ese nome comporase polo segundo carácter do código do fabricante en minúscula, máis o terceiro, cuarto, quinto e sexto da descrición do produto. Nunha primeira columna o código aparecerá en minúsculas, e nunha segunda en maiúsculas.
* **Proposta 7.** Listaxe cos nomes dos empregados co formato *ape1 ape2, nome*. Se algún empregado non ten segundo apelido, por exemplo Susanne Smith, no resultado aparecerá *Smith, Sussane*, sen espazos antes da coma.
* **Proposta 8.** Queremos amosar os distintos títulos dos nosos empregados en castelán, e para iso deberemos substituír a palabra *VENDAS* por *VENTAS.*
* **Proposta 9.** Consulta que devolva a seguinte información de tempo en distintas columnas co nome adecuado cada unha:
* data e hora actuais sen axuste de zona horaria,
* data e hora actuais con axuste de zona horaria,
* mes actual en número,
* mes actual en número *(emprega unha función diferente á da anterior columna)*,
* ano actual,
* mes actual en nome,
* hora actual,
* nanosegundos actuais.
* **Proposta 10.** Listaxe que devolva o nome de todos os empregados (nome, ape1, ape2), a data de contrato, e nunha última columna a data de contrato adiantada un ano. O formato das dúas datas será dd/mm/aaaa (con barras).
* **Proposta 11.** Listaxe que devolva o número de cada pedido coa data de pedido. Nunha terceira columna deberá aparecer a mesma data de pedido pero retrasada dous meses. O formato das dúas datas será dd-mm-aaaa (con guións).
* **Proposta 12.** Listaxe que devolva o nome e apelidos (nome, ape1, ape2) de cada empregado, a data de contrato e o número de anos que hai que leva traballando na empresa cada un deles.
* **Proposta 13.** Consulta que devolva a descrición de cada produto co seu prezo nunha segunda columna, e ademais deberán amosarse en columnas diferentes:
* o prezo como un enteiro aproximado por defecto,
* o prezo como un enteiro aproximado por exceso,
* a raíz cadrada do prezo,
* o cadrado do prezo, e,
* o cubo do prezo.
* **Proposta 14.** Repite a consulta anterior pero agora só amosaremos a descrición, o prezo e a raíz cadrada, pero a raíz cadrada deberá amosarse con como moito 4 cifras na parte enteira e 3 na decimal.
* **Proposta 15.** Consulta que devolva a seguinte información do servidor no que está a nosa instancia de SQL Server: idioma, número máximo de conexións permitidas, nome do servidor e da instancia e versión do xestor.
* **Proposta 16.** Consulta que amose a descrición do produto e as súas existencias. Nunha terceira columna de nome estado\_existencias amosarase o seguinte:
* Se o número de existencias é superior a 20 aparecerá a palabra *Suficientes*.
* Se o número de existencias é menor ou igual a 20 aparecerá *Insuficientes*.

Esta consulta deberás resolvela de dous xeitos posibles, en dúas consultas diferentes, empregando dúas funcións lóxicas distintas.