	RAMA:	Informática	CICLO:	Desenvolvimento de Aplicacions Multiplataforma		
	MÓDULO	Bases de datos				CURSO: 1º
	PROTOCOLO:	Apuntes clases	AVAL:	2	DATA:	2021/2022
	UNIDAD COMPETENCIA					


a.

Tema 5

Funciones

Índice

1.	Introducción	1
2.	Funciones matemáticas.....	1
3.	Funciones de listas	2
4.	Funciones agregadas.....	2
5.	Funciones de cadenas de caracteres	3
6.	Funciones para el manejo de fechas.....	4
7.	Funciones de información	8
8.	Conversión de tipos	9
9.	Funciones de compresión, cifrado y hashing	9
10.	Funciones que trabajan con ips	10
11.	Otras funciones.....	10

	RAMA:	Informática	CICLO:	Desenvolvimento de Aplicacions Multiplataforma		
	MÓDULO	Bases de datos				CURSO: 1º
	PROTOCOLO:	Apuntes clases	AVAL:	2	DATA:	2021/2022
	UNIDAD COMPETENCIA					

1. Introducción

Este tema, en el que se verán parte de las funciones disponibles en *MariaDB/MySQL*, está basado en el manual del *MySQL* sobre funciones. Para consultar su contenido de forma íntegra se puede consultar el siguiente enlace: <https://dev.mysql.com/doc/refman/5.7/en/functions.html>

Una función se podrá aplicar sobre columnas, constantes o el resultado de aplicar una operación o función sobre los anteriores, es decir, se pueden anidar funciones.

2. Funciones matemáticas¹

Las funciones matemáticas realizan una operación sobre un número, o una cadena y retornan un número.

El valor *n* representa un número que puede ser una constante, una variable, una columna o el resultado de realizar una operación o aplicar un función sobre los anteriores.

Las principales funciones matemáticas son:


FUNCIÓN	DEVUELVE	EJEMPLO: <u>SELECT</u>
ABS(<i>n</i>)	Valor absoluto de <i>n</i> .	ABS(-10)→10, ABS(10)→10
CEIL(<i>n</i>)	El siguiente número entero superior o igual a <i>n</i> .	CEIL(11.2)→12, CEIL(-10.3)→-10
CONV(<i>n</i> , baseO, BaseD)	Convierte un número <i>n</i> de la base de origen baseO a la base de destino BaseD.	CONV('a', 16, 2) →1010 CONV('10', 10, 2) →1010
COS(<i>n</i>)	Coseno del valor <i>n</i> .	COS(0)→1
CRC32(<i>n</i>)	Valor del chequeo de redundancia cíclica de la cadena <i>n</i>	CRC32('BD') → 4062690033
FLOOR(<i>n</i>)	El primer número entero inferior o igual a <i>n</i> .	FLOOR(16) → 16, FLOOR(20.3) → 20
MOD(<i>n</i> , <i>m</i>)	Resto de dividir <i>n</i> entre <i>m</i>	MOD(10,4) → 2
POW(<i>n</i> , <i>exp</i>)	Eleva el número <i>x</i> al exponente <i>exp</i> .	POW(2,3) → 8
RAND()	Valor aleatorio entre 0 y 1. Para calcular un valor aleatorio <i>x</i> entre dos números <i>k</i> y <i>j</i> (<i>k</i> < <i>x</i> < <i>j</i>) se usa la formula: FLOOR(<i>k</i> + RAND() * (<i>j</i> - <i>k</i>))	RAND() → 0.013939858427215237
ROUND(<i>n</i>)	Valor de <i>n</i> redondeado a cero decimales. Si la parte decimal de <i>n</i> es menor que .5 devuelve el entero inferior, en caso contrario el superior.	ROUND(4.44) → 4 ROUND(4.5) → 5
ROUND(<i>n</i> , <i>m</i>)	Igual que el anterior pero redondeando a <i>m</i> decimales. Si el valor de <i>m</i> es negativo redondea la parte entera.	ROUND(4.47,1)→4.5,ROUND(4.47,2)→4.47 ROUND(11,-1)→10, ROUND(111,-2)→ 100
SIGN(<i>n</i>)	Signo de <i>n</i> . Puede tomar los valores -1, 0 o 1 si <i>n</i> es menor, igual o mayor que cero respectivamente.	SIGN(-12) → -1 , SIGN(2) → 1
SIN(<i>n</i>)	Seno del valor <i>n</i> .	SIN(0) → 11
SQRT(<i>n</i>)	Raíz cuadrada del número positivo <i>n</i> .	SQRT(4) → 2
TRUNCATE(<i>n</i> , <i>m</i>)	Trunca el número <i>n</i> para que tenga <i>m</i> decimales. Si <i>m</i> es negativo pone <i>m</i> números a la izquierda del punto a cero.	TRUNCATE(2.55, 1)→2.5, TRUNCATE(2.5,0)→2 TRUNCATE(22, -1)→20,TRUNCATE(22.1, -2)→0

Obtener el nombre, los minutos y el entero superior a los minutos de los jugadores.

```
SELECT nombre, minutos, CEIL(minutos), ROUND(minutos,1) FROM jugadores LIMIT 2;
```

nombre	minutos	ceil (minutos)	round(mi nutos, 0)
Kareem Abdul -Jabbar	34.3	35	34
Magi c Johnson	36.7	37	37

¹ <https://dev.mysql.com/doc/refman/5.7/en/mathematical-functions.html>
<https://mariadb.com/kb/en/library/numeric-functions/>

	RAMA:	Informática	CICLO:	Desenvolvimento de Aplicacions Multiplataforma		
	MÓDULO	Bases de datos				CURSO: 1º
	PROTOCOLO:	Apuntes clases	AVAL:	2	DATA:	2021/2022
	UNIDAD	COMPETENCIA				

3. Funciones de listas

Existen una serie de funciones que no trabajan con valores simples sino que toman como argumentos una lista de dos o más valores ya sean estos constantes, columnas o el resultado de aplicar una función o operación sobre los valores anteriores. El resultado es un único valor.

Tenemos las siguientes funciones de listas de valores:

FUNCIÓN	DEVUELVE	EJEMPLO: <u>SELECT</u>
COALESCE (val1, val2, ...)	Primer valor no nulo de la lista o Null si solo hay valores nulos	COALESCE(NULL, 3, 5) → 3 COALESCE(NULL, NULL) → NULL
GREATEST (val1, val2, ...)	El mayor valor de la lista. Si en la lista hay un valor nulo devuelve Null. ²	GREATEST(3, 5, 8, 12, 34, 132) → 132 GREATEST(3, '5', 8, 12, '34', 12) → 34 GREATEST('DOS', 'TRES', 'SEIS') → 'TRES' GREATEST('2', '3', '6') → 6
LEAST (val1, val2, ...)	Igual que greatest pero con el menor valor de la lista. ³	LEAST(3, 5, 8, 12, 34, 132) → 3 LEAST(3, 5, 8, NULL, 34, 132) → NULL

4. Funciones agregadas³

Las funciones agregadas trabajan con grupos de filas, si la clausula GROUP BY no está presente trabaja con todas las filas de la tabla y el resultado tendrá un único resultado, calculando valores como la suma, media, valor máximo, ... del grupo de filas. Los valores nulos no se tienen en cuenta y se ignoran.

Si con las funciones agregadas se usa la cláusula Distinct solo se tendrá en cuenta los valores distintos.

Tenemos las siguientes funciones agregadas:

FUNCIÓN	DEVUELVE
AVG([Distinct] expr)	Media aritmética de los valores expr, con valores no nulos, de las filas devueltas en la consulta.
COUNT ([Distinct] *)	Cuenta el número de filas devueltas por la consulta. Las columnas con nulos se ignoran.
COUNT ([Distinct] expr)	Cuenta el número de filas, con el valor de expr no nulo, devueltas por la consulta.
MAX([Distinct] expr)	Valor máximo, ya sea numérico o cadenas, de expr de los valores expr, con valores no nulos, de las filas devueltas por la consulta.
MIN([Distinct] expr)	Valor mínimo, ya sea numérico o cadenas, de los valores expr, con valores no nulos, de las filas devueltas por la consulta.
SUM([Distinct] expr)	Suma de los valores expr de las filas devueltas por la consulta.
GROUP_CONCAT(expr)	Cadena con la concatenación de los valores no nulos devueltos por la consulta.

Ejemplo de consultas son:

EJEMPLO

- Cálculo de número medio de partidos de los jugadores de la tabla jugadores:
`SELECT AVG(partidos) FROM jugadores;`
- Cálculo del número de filas de la tabla jugadores:
`SELECT COUNT(*) FROM jugadores;`
- Contar el número de valoraciones distintas:
`SELECT COUNT(DISTINCT valoracion) FROM jugadores;`

^{2 3} https://dev.mysql.com/doc/refman/5.7/en/comparison-operators.html#function_least criterios de comparación.

³ <https://mariadb.com/kb/en/library/aggregate-functions/>
<https://dev.mysql.com/doc/refman/5.7/en/group-by-functions.html>

COLEXIO VIVAS S.L.	RAMA:	Informática	CICLO:	Desenvolvimento de Aplicacions Multiplataforma		
	MÓDULO	Bases de datos				CURSO: 1º
	PROTOCOLO:	Apuntes clases	AVAL:	2	DATA:	2021/2022
	UNIDAD	COMPETENCIA				

- Número máximo de partidos jugados por un jugador:
`SELECT MAX(partidos) FROM jugadores;`
- Obtener el nombre mínimo (alfabéticamente) de la tabla jugadores:
`SELECT MIN(nombre) FROM jugadores;`
- Consigue la suma de todos los partidos de los jugadores:
`SELECT SUM(partidos) FROM jugadores;`
- Consigue una cadena con los nombres de los jugadores:
`SELECT GROUP_CONCAT(nombre) AS nombres FROM jugadores;`

5. Funciones de cadenas de caracteres⁴

Estas funciones permiten la manipulación de cadenas de caracteres. Las funciones más usuales son:

FUNCIÓN	DEVUELVE	EJEMPLO: <u>SELECT</u>
ASCII(cad)	Valor ASCII de la primera letra de la cadena cad. 0 si la cadena está vacía y Null si la cadena es nula.	ASCII(3) → 51 ASCII('a') → 97
BIN(N)	Cadena de caracteres con la representación binaria del número N. Si N es nulo el resultado el nulo.	BIN(22) → 10110 BIN(8) → 1000;
BINARY	Convierte una cadena a una cadena binaria permitiendo comparaciones byte a byte lo que permite diferenciar mayúsculas de minúsculas. Afecta a toda la operación.	'a' = 'A' → 1, binary 'a' = 'A' → 0 'a' = 'a' → 1, binary 'a' = 'a' → 0
CONCAT(cad1, [cad2, ...])	Cadena resultante de unir las cadenas pasadas como parámetros.	CONCAT('My',Null,'QL') → Null CONCAT('My','S','QL') → 'MySQL'
CONCAT_WS(cadseparador, cad1, [cad2, ...])	Cadena resultante de unir las cadenas pasadas como parámetros usando cadseparador como separador.	CONCAT_WS('#','My','S','QL') → My#S#QL
ELT(n, cad, cad1, cad2, ...)	Elemento indicado por el número n siendo N >=1. Null si el elemento no existe.	SELECT ELT(2,'A','B','C') → 'B' SELECT ELT(6,'A','B','C') → Null
FIELD(cad, cad1, cad2, ...)	Índice de la primera ocurrencia de cad en las lista en la lista cad1, cad2, ... o 0 si en la lista no existe cad.	FIELD('si', 'hola', 'si', 'Va') → 2 FIELD('so', 'hola', 'si', 'Va') → 0
CHAR_LENGTH(cad)	Número de caracteres de la cadena cad.	CHAR_LENGTH('cadena') → 6 CHAR_LENGTH('ñ€') → 2
LENGTH(cad)	Devuelve o tamaño de la cadena cad en bytes. Se tiene en cuenta los caracteres que ocupen varios bytes.	LENGTH('cadena') → 6 LENGTH('ñ€') → 5
LOCATE(substr, cad [, pos])	Posición de la primera ocurrencia de substr en cad. Si se especifica pos se comienza a buscar en la posición pos. Si no se encuentra substr retorna 0.	LOCATE('car', 'canica') → 0 LOCATE('ca', 'canica') → 1 LOCATE('ca', 'canica', 3) → 5
LOWER(cad)	Cadena cad en minúsculas.	LOWER('PARTIDO') → 'partido'
UPPER(cad)	Cadena cad en mayúsculas.	UPPER('dam') → 'DAM'
LTRIM(cad)	Elimina, en cad, los espacios en blanco por la izquierda.	LTRIM(' cad ') → 'cad '
RTRIM(cad)	Elimina, en cad, los espacios en blanco por la derecha.	RTRIM(' cad ') → ' cad'
TRIM ([{TRAILING LEADING BOTH } [conj] FROM] cad)	Cadena cad con todos los prefijos, sufijos o ambos eliminados.	TRIM(' cad ') → 'cad' TRIM(LEADING 'ca' FROM 'canica') → 'nica' TRIM(BOTH 'ca' FROM 'canica') → 'ni'
REPEAT(cad, N)	Cadena en donde aparece cad concatenado consigo mismo N veces.	REPEAT('la ',6) → 'la la la la la la'
REPLACE(cad,cad2,cad3)	Cadena resultante de substituir en cad todas las ocurrencias de cad2 con el valor de cad3.	REPLACE('canica', 'ca', '_a_') → '_a_ni_a_'

⁴ <https://mariadb.com/kb/en/string-functions/>

COLEXIO VIVAS S.L.	RAMA:	Informática	CICLO:	Desenvolvimento de Aplicacions Multiplataforma		
	MÓDULO	Bases de datos				CURSO: 1º
	PROTOCOLO:	Apuntes clases	AVAL:	2	DATA:	2021/2022
	UNIDAD COMPETENCIA					

REVERSE(cad)	Cadena cad invertida.	REVERSE('cadena') → 'anedac'
STRCMP(expr1,expr2)	-1, 0 o 1 si expr1 es menor, igual o mayor que expr2 respectivamente.	STRCMP('aa', 'aa') → 0 STRCMP('aa', 'ac') → -1 STRCMP('aa', 'a0') → 1
LEFT(cad, N)	Devuelve los primeros N caracteres por la izquierda.	LEFT('PARTIDO', 3) → 'PAR'
RIGHT(cad, N)	Devuelve los primeros N caracteres por la derecha.	RIGHT('PARTIDO', 3) → 'IDO'
SUBSTRING (cad, pos)	Subcadena de cad desde la posición pos hasta el final. El primer carácter de cad tiene índice 1.	SUBSTRING('PARTIDO', 3) → RtiDO SUBSTRING('PARTIDO', -3) → iDO
SUBSTRING (cad, pos, n)	Es idéntico al anterior pero crea la subcadena desde la posición pos con tamaño de n caracteres. Si n tiene un valor negativo la subcadena se crea desde el final.	SUBSTRING('PARTIDO', 3, 4) → RtiD SUBSTRING('PARTIDO', -3, 2) → iD
FORMAT(num, cantDecim)	Cadena de caracteres con el número num formateado con cantDecim posiciones decimales.	FORMAT(2.123456, 4) → '2.1235' FORMAT(332.26,0) → '332'

EJEMPLO

```
SELECT UPPER(nombre), LENGTH(nombre), SUBSTRING(NOMBRE, 2, 12) AS sub FROM jugadores LIMIT 3;
```

upper (nombre)	length (nombre)	sub
KAREEM ABDUL-JABBAR	19	areem Abdul-
MAGIC JOHNSON	13	agic Johnson
JERRY WEST	10	erry West

6. Funciones para el manejo de fechas⁵

MariaDB/MySQL tiene disponibles una serie de funciones para trabajar de forma específica con fechas y horas. MariaDB/MySQL realiza automáticamente conversiones de tipos convirtiendo estos tipos de datos en formato cadena a numérico y viceversa. Como separador de las distintas partes de una fecha u hora se puede utilizar cualquier delimitador (no solo -, / o :) siempre que sea consistente en todo el campo.

En MariaDB/MySQL las columnas de esta consulta son iguales y producen el mismo resultado: 2020.

```
SELECT YEAR(20200601), YEAR('20200601'), YEAR('2020/06/01'), YEAR('2020#06#01');
```


Todas estas funciones admiten o producen uno de los tipos de datos siguientes:

TIPO	FORMATO	BYTES	VALORES DISPONIBLES
YEAR	YYYY 2 ó 4 dígitos	1	Con cuatro dígitos un valor entre 1901 y 2155 Con dos dígitos los valores 70-99 son convertidos a 1970-1999 y los valores a 00-69 a 2000-2069.
DATE	YYYY-MM-DD	3	Un valor entre 1000-01-01 a 9999-12-31
TIME	HH:MM:SS	3*	Un valor entre -838:59:59.999999 a 838:59:59.999999
DATETIME	YYYY-MM-DD HH:MM:SS DATE + TIME	8*	Un valor entre 01-01 00:00:00.000000 y 9999-12-31 23:59:59.000000
TIMESTAMP		4*	Un valor entre 1970-01-01 00:00:01 y 2038-01-19 05:14:07

* Estos tipos de datos admiten el uso de microsegundos que permiten aumentar su precisión a costa de ocupar más espacio. Se puede consultar el espacio adicional ocupado en la tabla adyacente.

PRECISIÓN EN SEGUNDOS	BYTES OCUPADOS
0	0 bytes
1, 2	1 byte
3, 4	2 bytes
5, 6	3 bytes

⁵ <https://mariadb.com/kb/en/date-time-functions/>

	RAMA:	Informática	CICLO:	Desenvolvimento de Aplicacions Multiplataforma		
	MÓDULO	Bases de datos				CURSO: 1º
	PROTOCOLO:	Apuntes clases	AVAL:	2	DATA:	2021/2022
	UNIDAD	COMPETENCIA				

Y pueden usar alguna de las siguientes unidades

UNIDAD	VALOR
MICROSECOND	Microsegundos
SECOND	Segundos
MINUTE	Minutos
HOURL	Horas


UNIDAD	VALOR
DAY	Día
WEEK	Semana
MONTH	Mes
YEAR	Año
QUARTER	Trimestre del año

Con las siguientes combinaciones entre ellos:

UNIDAD	VALOR	EJEMPLO
SECOND_MICROSECOND	Segundos microsegundos	'10:020'
MINUTE_MICROSECOND	Minutos Segundos Microsegundos	'10:20:030'
MINUTE_SECOND	Minutos Segundos	'10:20'
HOURL_MICROSECOND	Horas Minutos Segundos Microsegundos	'10:20:30:040'
HOURL_SECOND	Horas Minutos Segundos	'10:20:30'
HOURL_MINUTE	Horas Minutos	'10:20'
DAY_MICROSECOND	Días Horas Minutos Segundos Microsegundos	'10:20:30:40:050'
DAY_SECOND	Días Horas Minutos Segundos	'10:20:30:40'
DAY_MINUTE	Días Horas Minutos	'10:20:30'
DAY_HOURL	Días Horas	'10:20'
YEAR_MONTH	Años Meses	'2010:10'


Las funciones más usuales son:

FUNCIÓN	DEVUELVE	EJEMPLO: <u>SELECT</u>
CURRENT_TIME()	Hora actual del sistema	CURRENT_TIME() → 23:45:46
CURRENT_DATE()	Fecha actual del sistema	CURRENT_DATE() → 2017-12-16
NOW() CURRENT_TIMESTAMP()	Fecha y hora actual (combinación de los dos anteriores)	NOW() → 2017-12-16 23:45:46 En los ejemplos siguientes NOW() representa esta fecha.
DATE(dateTime)	Parte de la fecha de una fecha especificada.	DATE(NOW()) → 2017-12-16
TIME(dateTime)	Parte de la hora de una fecha especificada.	TIME(NOW()) → 23:45:46
LAST_DAY(fecha)	Fecha del último día del mes indicado por fecha.	LAST_DAY(now()) → 2017-12-31
MICROSECOND (hora)	Microsegundos de la hora especificada. Con valores entre 0 y 999999.	MICROSECOND('11:23:120.123456') → 123456
SECOND(hora)	Segundos de la hora especificada. Con valores entre 0 y 59.	SECOND(NOW()) → 46
MINUTE(hora)	Minutos de la hora especificada. Con valores entre 0 y 59.	MINUTE('23:45:46') → 45
HOURL(hora)	Horas de la hora especificada. Con valores entre 0 y 23.	HOURL(NOW()) → 23
DAYNAME(fecha)	Nombre del día del parámetro fecha.	DAYNAME(NOW()) → 'Saturday'
DAYOFMONTH(fecha)	Día del mes de la fecha especificada. Con valores entre 1 y 31.	DAYOFMONTH(NOW()) → 16

	RAMA:	Informática	CICLO:	Desenvolvimento de Aplicacions Multiplataforma		
	MÓDULO	Bases de datos				CURSO: 1º
	PROTOCOLO:	Apuntes clases	AVAL:	2	DATA:	2021/2022
	UNIDAD	COMPETENCIA				

DAYOFYEAR(fecha)	Día del año del parámetro fecha como un valor entre 1 y 366.	DAYOFYEAR('2017-12-16') → 350
DAYOFWEEK(fecha)	Día de la semana del parámetro fecha como número entre el 1 y el 7. Donde el 1 representa el domingo y el 7 el sábado.	DAYOFWEEK(NOW()) → 7
WEEKDAY(fecha)	Día de la semana del parámetro fecha como número entre el 0 y el 6. Donde el 0 representa el lunes y el 6 el domingo.	WEEKDAY(NOW()) → 5
WEEK (fecha, [modo])	Número de la semana del año del parámetro fecha. Si modo está presente se debe especificar un modo válido ⁶ si se omite se usa el modo por defecto.	WEEK(NOW(),2) → 51
MONTH(fecha)	Mes del año del parámetro fecha como un valor entre 1 y 12.	MONTH(NOW()) → 12
MONTHNAME(fecha)	Nombre del mes del parámetro fecha.	MONTHNAME(NOW()) → 'December'
YEAR(fecha)	Año del parámetro fecha como un valor entre 1000 y 9999.	YEAR(NOW()) → 2017
QUARTER(fecha)	Trimestre del año del parámetro fecha como un valor entre 1 y 4.	QUARTER(NOW()) → 4
MAKEDATE(año, día del año)	Crea una fecha a partir del año y un número de día dentro del año	MAKEDATE(2017,31) → 2017-01-31 MAKEDATE(2017,36) → 2017-02-05
MAKETIME(hora, minuto, segundo)	Crea una hora a partir de sus argumentos: hora, minuto y segundo:	MAKETIME(13,57,33) → 13:57:33 MAKETIME(13,67,33) → Null
SEC_TO_TIME(segundos)	Dato tipo hora (hh:mm:ss) creado desde los segundos pasados como parámetro.	SEC_TO_TIME(3660) → 01:01:00
TIME_TO_SEC(hora)	Segundos contenidos en la hora pasada como parámetro.	TIME_TO_SEC ('01:01:05') → 3665
DATE_FORMAT(fecha, cadenaDeFormato)	Parámetro fecha con el formato especificado en cadenaDeFormato, pudiendo usarse las constantes de formato que se verán en la tabla siguiente a esta.	DATE_FORMAT('2017-12-16 23:45:46', 'Hoy es %W %d %M de %Y y son las %H:%i') → 'Hoy es Saturday 16 December de 2017 y son las 23:45'
TIME_FORMAT(hora, cadenaDeFormato)	Parámetro hora con el formato especificado en cadenaDeFormato, pudiendo usarse las constantes de formato que se verán en la tabla siguiente a esta.	Time_FORMAT('23:45:46', 'Son las %H:%i:%s') → 'Son las 23:45:46'
EXTRACT (tipo FROM fecha)	Obtiene un tipo de dato fecha de la fecha pasado como parámetro.	EXTRACT(YEAR FROM NOW()) EXTRACT(HOUR FROM NOW());
ADDTIME(expr1, expr2)	Fecha resultante de añadir a expr1 expr2	ADDTIME('2017-12-31 23:59:59.999999', '1 1:1:1.000002'); → '2018-01-02 01:01:01.000001'
DATE_ADD(fecha, INTERVAL expresión unidad) ADDDATE(fecha, INTERVAL expresión unidad)	Fecha resultante de añadir al parámetro fecha un periodo de tiempo (expresión puede ser negativa lo que restará el periodo de tiempo). Unidad es o una de las unidades vistas o una de las combinaciones disponibles.	DATE_ADD('2017-12-22', INTERVAL 14 DAY) → 2018-01-05 DATE_ADD('2017-12-22 13:21:00', INTERVAL '10:20:30' HOUR_SECOND) → 2017-12-22 23:41:30 DATE_ADD('2017-12-22', INTERVAL -1 WEEK) → 2017-12-15

⁶ <https://mariadb.com/kb/en/library/week/>


	RAMA:	Informática	CICLO:	Desenvolvimento de Aplicacions Multiplataforma		
	MÓDULO	Bases de datos				CURSO: 1º
	PROTOCOLO:	Apuntes clases	AVAL:	2	DATA:	2021/2022
	UNIDAD COMPETENCIA					

PERIOD_ADD(fecha, cantidad Meses)	Fecha a la que se le suma la cantidad de meses. Tanto la fecha pasada como parámetro como el resultado tienen el formato AAAAMM o AAMM.	PERIOD_ADD('201709', 6) → 201803 PERIOD_ADD('1709', '6') → 201803
DATEDIFF(expr1, expr2)	Número de días entre expr1 y expr2. No se tiene en cuenta la parte de la hora para el cálculo.	DATEDIFF('2017-12-20 23:59:59', '2017-12-10') → 10
TIMEDIFF(expr1, expr2)	Formato hora con expr1 menos expr2.	TIMEDIFF('2017:12:22 11:00:00', '2017:12:21 10:00:00.000001') → 24:59:59.999999 TIMEDIFF('12:00:00', '10:00:00.000001') → 01:59:59.999999
PERIOD_DIFF(fecha1, fecha2)	Número de meses entre fecha1 y fecha2 (las fechas con formato AAMM o AAAAMM).	PERIOD_DIFF('201709', '201704') → 5 PERIOD_DIFF('1709', '201704') → 5
UNIX_TIMESTAMP() UNIX_TIMESTAMP(fecha)	Entero que representa la marca de tiempo Unix de la hora actual, si se ejecuta sin parámetros, o de la fecha especificada. La marca de tiempo Unix son los segundos desde del 1 de enero de 1970.	UNIX_TIMESTAMP() → 1513559887 UNIX_TIMESTAMP('2017-12-20 10:21:10') → 1513761670
FROM_UNIXTIME(marcaDe TiempoUnix) FROM_UNIXTIME(marcaDe TiempoUnix, cadenaFormato)	Fecha producida de convertir la marcaDe TiempoUnix. Si se especifica la cadenaFormato esta es usada para realizar la conversión.	FROM_UNIXTIME(1) → 1970-01-01 01:00:01 FROM_UNIXTIME(3661) → 1970-01-01 02:01:01 FROM_UNIXTIME(1513761670, '%d %M %Y') → 20 December 2017

Especificadores de formato

ESP.	PROPÓSITO
%a	Nombre del día de la semana abreviado (Sun..Sat)
%b	Nombre del mes abreviado (Jan..Dec)
%c	Mes como número (0..12)
%D	Día del mes con sufijo inglés (0th, 1st, 2nd, ...)
%d	Día del mes como número de dos cifras (00..31)
%e	Día del mes como número (0..31)
%f	Microsegundos (000000..999999)
%H	Hora con dos cifras (00..23)
%h	Hora con dos cifras (01..12)
%I	Hora con dos cifras (01..12)
%i	Minutos (00..59)
%j	Día del año (001..366)
%k	Hora (0..23)
%l	Hora (1..12)
%M	Nombre del mes (January..December)
%m	Mes como número (00..12)

ESP.	PROPÓSITO
%p	AM o PM
%r	Hora en formato 12 horas (hh:mm:ss seguido de AM o PM)
%S	Segundos (00..59)
%s	Segundos (00..59)
%T	Hora en formato 24 horas (hh:mm:ss)
%U	Semana (00..53), donde domingo es el primer día de la semana. Week modo 0.
%u	Semana (00..53), donde lunes es el primer día de la semana. Week modo 1.
%V	Semana (01..53), donde domingo es el primer día de la semana. Week modo 2. Usado con %X
%v	Semana (01..53), donde lunes es el primer día de la semana; . Week modo 3. Usado con %X
%W	Nombre del día de la semana (Sunday..Saturday)
%w	Día de la semana (0=Sunday..6=Saturday)
%X	Año para la semana donde domingo es el primer día de la semana, numérico, cuatro dígitos; usado con %V
%x	Año para la semana, donde lunes es el primer día de la semana, numérico, cuatro dígitos; usado con %v
%Y	Año, numérico, cuatro dígitos
%y	Año, numérico (dos dígitos)

	RAMA:	Informática	CICLO:	Desenvolvimento de Aplicacions Multiplataforma		
	MÓDULO	Bases de datos				CURSO: 1º
	PROTOCOLO:	Apuntes clases	AVAL:	2	DATA:	2021/2022
	UNIDAD COMPETENCIA					

La variable global donde se establece el lenguaje utilizado para mostrar los días, nombres de los meses y las abreviaturas es:

```
SELECT @@lc_time_names;
+-----+
| @@lc_time_names |
+-----+
| en_US           |
+-----+
```

Mediante el comando siguiente posemos modificar el idioma y establecerlo a español de España:

```
SET @@lc_time_names = 'es_ES';


SELECT DAYNAME ('2000-12-25'), MONTHNAME ('2000-12-25');
+-----+-----+
| DAYNAME ('2000-12-25') | MONTHNAME ('2000-12-25') |
+-----+-----+
| lunes                  | diciembre                 |
+-----+-----+
```

7. Funciones de información⁷

Las funciones en esta sección ofrecen información al usuario.

FUNCIÓN	DEVUELVE	EJEMPLO: <u>SELECT</u>
BENCHMARK(num, expr)	Ejecuta la expresión expr num veces. Se usa para comprobar cuanto se tarda en ejecutar una sentencia. Siempre devuelve el valor 0.	BENCHMARK(1000000, concat('hello','goodbye'));
CHARSET(cadena)	Codificación usada en el parámetro cadena.	CHARSET('abc') → UTF8 CHARSET(CONVERT('abc' USING latin1)) → latin1
COLLATION(cadena)	Colación usada en el parámetro cadena.	COLLATION('abc') → utf8_general_ci COLLATION(_latin1 'abc') → latin1_swedish_ci
CONNECTION_ID()	Identificador único de conexión.	CONNECTION_ID() → 2
CURRENT_USER CURRENT_USER()	Nombre de usuario y nombre de equipo de conexión que determina los privilegios de acceso.	CURRENT_USER() → root@localhost
USER()	Nombre de usuario y nombre de equipo de conexión proporcionado por el cliente. Puede diferir de CURRENT_USER().	USER() → root@localhost
DATABASE() SCHEMA()	Nombre de base de datos actual o Null si no hay base de datos seleccionada. .	DATABASE () → tema5
FOUND_ROWS()	Número de filas devueltas por la última sentencia SELECT.	SELECT * FROM JUGADORES LIMIT 4; FOUND_ROWS() → 4
ROW_COUNT()	Número de filas insertadas, actualizadas o borradas por la consulta anterior.	ROW_COUNT() → 3
VERSION()	Versión del servidor.	VERSION() → 10.1.10-MariaDB

⁷ <https://mariadb.com/kb/en/information-functions/>

	RAMA:	Informática	CICLO:	Desenvolvimento de Aplicacions Multiplataforma		
	MÓDULO	Bases de datos				CURSO: 1º
	PROTOCOLO:	Apuntes clases	AVAL:	2	DATA:	2021/2022
	UNIDAD	COMPETENCIA				

8. Conversión de tipos⁸

En MariaDB/MySQL se disponen de dos funciones para conversión de tipos: CAST Y CONVERT. Ambos toman como parámetro de entrada un valor de un tipo y producen otro valor de un tipo diferente.



FUNCIÓN	DEVUELVE	EJEMPLO: <u>SELECT</u>
CAST(expr AS type)	Convierte el valor expr al tipo de dato type. Sintaxis estándar SQL.	CAST('abc' AS BINARY) → 'abc' CAST('1' AS UNSIGNED INTEGER) → 1 CAST(123 AS CHAR CHARACTER SET UTF8) → '123'
CONVERT(expr, type)	Convierte el valor expr al tipo de dato type. Es equivalente a CAST. Sintaxis ODBC.	CONVERT('ABC', BINARY) → 'abc' CONVERT('1', UNSIGNED INTEGER) → 1
CONVERT(expr USING transcoding_name)	Convierte cadenas de texto a diferentes conjuntos de caracteres. Sintaxis estándar SQL.	CONVERT('abc' USING utf8) → 'abc'

El parámetro type puede ser uno de los siguientes tipos:

BINARY	CHAR
DATE	DATETIME
TIME	DECIMAL
SIGNED [INTEGER]	UNSIGNED [INTEGER]


9. Funciones de compresión, cifrado y hashing⁹

Las siguientes funciones permiten comprimir y cifrar datos.

FUNCIÓN	DEVUELVE	EJEMPLO: <u>SELECT</u>
COMPRESS(cad)	Comprime la cadena cad como una cadena binaria.	LENGTH(REPEAT('a', 1000)) → 1000 LENGTH(COMPRESS(REPEAT('a', 1000))) → 21 LENGTH(COMPRESS('a')) → 13 LENGTH(COMPRESS(REPEAT('a', 16))) → 15
UNCOMPRESS(cad)	Descomprime la cadena comprimida cad	UNCOMPRESS(COMPRESS('string')) → 'string'
AES_ENCRYPT(cad, pass)	Cifra la cadena cad con la contraseña pass usando el algoritmo AES.	AES_ENCRYPT('cad','pass') → 
AES_DECRYPT(cad, pass)	Descifra la cadena cifrada cad usando la contraseña pass mediante el algoritmo AES.	
DES_ENCRYPT(cad, pass)	Cifra la cadena cad con la contraseña pass usando el algoritmo Triple DES.	DES_ENCRYPT('cad','pass') → 
DES_DECRYPT(cad, pass)	Descifra la cadena cifrada cad usando la contraseña pass mediante el algoritmo Triple DES.	
PASSWORD(pass)	Calcula la contraseña hash de la contraseña de texto pass. Ya no es considerado seguro.	PASSWORD('notagoodpwd') → *3A70EE9FC6594F88CE9 E959CD51C5A1C002DC937
MD5(cad)	Calcula el MD5 128-bits checksum para la cadena cad. Ya no es considerado seguro	MD5('cad') → b5fde512c76571c8afd6a6089eaf42a
SHA(cad) SHA1(cad)	Calcula el SHA-1 160-bit checksum para la cadena cad. Ya no es considerado seguro.	SHA('cad') → 462c18d5b89050fb1b7f8fca1e535af868009675
SHA2(cad,hash_len)	Calcula el SHA-2 checksum para la cadena cad. Hash_len indica la longitud en bits y puede ser: 224, 256, 384 o 512 que corresponden con SHA-224, SHA-256, SHA-384 y SHA-512.	SHA('cad', 512) → 2a50026d5676023e36b37c0.....

⁸ <https://dev.mysql.com/doc/refman/8.0/en/cast-functions.html>

⁹ <https://mariadb.com/kb/en/encryption-hashing-and-compression-functions/>

	RAMA:	Informática	CICLO:	Desenvolvimento de Aplicacions Multiplataforma		
	MÓDULO	Bases de datos				CURSO: 1º
	PROTOCOLO:	Apuntes clases	AVAL:	2	DATA:	2021/2022
	UNIDAD COMPETENCIA					

10. Funciones que trabajan con ips¹⁰

Conjunto de otras funciones que trabajan con direcciones Ip.

FUNCIÓN	DEVUELVE	EJEMPLO: <u>SELECT</u>
INET_ATON(expr)	Dada una dirección IP devuelve su representación numérica.	INET_ATON('192.168.1.1') → 3232235777
INET6_ATON(expr)	Dada una dirección IPv6 devuelve su representación numérica.	HEX(INET6_ATON('10.0.1.1')) → 0A000101
INET_NTOA(expr)	Dada una representación numérica de una dirección IP devuelve una cadena con la dirección.	INET_NTOA(3232235777) → '192.168.1.1'
INET6_NTOA(expr)	Dada una representación numérica de una dirección IPv6 devuelve una cadena con la dirección.	INET6_NTOA(UNHEX('0A000101')) → '10.0.1.1'
IS_IPV4(expr)	1 si expr es una dirección IP. 0 en otro caso.	IS_IPV4('1110.0.1.1') → 0 IS_IPV4('192.168.1.1') → 1
IS_IPV6(expr)	1 si expr es una dirección IPv6. 0 en otro caso.	IS_IPV6('48f3::d432:1431:ba23:846f') → 1

11. Otras funciones¹¹

Función que no tenía cabida en apartados anteriores.

FUNCIÓN	DEVUELVE	EJEMPLO: <u>SELECT</u>
SLEEP(número)	Pausa la consulta por número de segundos. Puede tener decimales.	SLEEP(5.5) SLEEP(55)

¹⁰ <https://mariadb.com/kb/en/miscellaneous-functions/>

¹¹ <https://mariadb.com/kb/en/sleep/>