CONCURSO FUNCIONES

Índice

1. AES_ENCRYPT()

La función AES_ENCRYPT en SQL se utiliza para cifrar datos usando el algoritmo AES. Este cifrado convierte información
sensible en un formato que no puede ser leída sin la clave de cifrado correspondiente. Es útil para proteger datos como
contraseñas o información personal.
EJEMPLO:
SELECT AES_ENCRYPT('Pepe', 'clave_secreta') AS nombre_secreto
FROM usuario
En el ejemplo justo arriba:
'Pepe' es el dato que queremos cifrar.
'clave_secreta' es la cadena que se utiliza para cifrar 'Pepe' , y esta cadena también es necesaria para descifrar
'Pepe' utilizando otra función.
'nombre_secreto' es el alias para llamar al resultado como queramos, en mi caso, quería llamarlo 'nombre_secreto'.

2. RTRIM() Y LTRIM()

La función RTRIM elimina los espacios en blanco al final de una cadena de texto.
La función LTRIM hace justo lo contrario que la función RTRIM. LTRIM elimina los espacios en blanco al inicio de una cadena
de texto.
EJEMPLO CON RTRIM :
SELECT RTRIM(' hola mundo ') AS resultado;
En el ejemplo justo arriba el resultado sería el siguiente:
' hola mundo'
EJEMPLO CON LTRIM:
SELECT LTRIM(' hola mundo ') AS resultado;
En el ejemplo justo arriba el resultado sería:
'hola mundo '

3. AES_DECRYPT()

Esta función en sql se utiliza para desencriptar datos que fueron previamente cifrados con el algoritmo AES. Esta función toma
el dato cifrado y lo convierte nuevamente a su formato original, usando la misma clave de cifrado que se utilizó al encriptarlo.
EJEMPLO:
SELECT AES_DECRYPT('Pepe', 'clave_secreta') AS nombre_secreto
FROM usuario;
En el ejemplo justo arriba:
'Pepe' es el dato que ya está cifrado en la tabla usuario y queremos descifrar.
'clave_secreta' es la clave que se utilizó para cifrar el dato.

‘nombre_secreto’ es el resultado de la función, que contiene el correo electrónico en su formato original

4. CAST()

La función CAST en SQL se utiliza para convertir un valor de un tipo de dato a otro.
EJEMPLO:
SELECT CAST(‘123’ AS INT) AS numero;

En el ejemplo justo arriba:
‘123’ es un valor de tipo String.
AS INT indica que queremos convertir ese valor a un tipo entero.
numero será el resultado, que en este caso es el valor numérico 123 .

5. INSTR()

La función INSTR en SQL se utiliza para buscar la posición de una subcadena dentro de una cadena más grande.
Devuelve la posición en la que comienza la primera aparición de la subcadena, y si no se encuentra, devuelve 0 .
EJEMPLO:
SELECT INSTR(‘correo@deEjemplo.com’, ‘@’) AS posicion;
En el ejemplo justo arriba:
‘correo@deEjemplo.com’ es la cadena donde se busca la subcadena ‘@’.
INSTR devolverá la posición en la que aparece la subcadena ‘@’ en la cadena.
El resultado sería 7 , porque el símbolo ‘@’ está en la octava posición de la cadena.

6. PI()

Esta función devuelve el número PI.
EJEMPLO:
SELECT PI();
En el ejemplo justo arriba:
En Heidi, redondea el número PI y solo devuelve 3,141593

7. FLOOR(1 + RAND() * 100)

SELECT FLOOR(1 + RAND() * 100) AS numero_Aleatorio;
Esta consulta genera un número entero aleatorio entre 1 y 100.

8. DATEDIFF()

SELECT DATEDIFF('2025-03-10', '2025-03-01') AS dias_diferencia;
Esta función calcula la diferencia entre dos fechas,

9. BIT_COUNT()

SELECT BIT_COUNT(13) AS bits_en_uno;
Esta función cuenta el número de bits en 1 en la representación binaria de un número.

10. PERIOD_DIFF()

SELECT PERIOD_DIFF(202503, 202401) AS meses_diferencia;
Esta función calcula la diferencia en meses entre dos fechas en formato AAAAMM(Año y Mes).

11. SLEEP()

SELECT SLEEP(5);
Esta función pausa la ejecución de la consulta durante el número de segundos que se indique.

12. DATE_SUB()

SELECT DATE_SUB('2025-02-10', INTERVAL 3 DAY) AS nueva_fecha;
Esta función resta un intervalo de tiempo de una fecha dada.

13. ARRAY_AGG()

SELECT ARRAY_AGG(nombre) FROM empleados;
Se utiliza para agrupar valores de una columna en un array

14. SHA1()

SELECT SHA1('miContraseñaSecreta') AS hash_valor;
Genera un valor hash de tipo SHA-1 (Secure Hash Algorithm 1) para una cadena de texto dada

15. TIME()

SELECT TIME('2025-02-16 12:34:56') AS solo_hora;
Se utiliza para extraer la parte de la hora de una fecha o convertir un valor a solo hora, minutos y segundos.

16. BIT_LENGTH()

SELECT BIT_LENGTH('Hola') AS longitud_en_bits;
Devuelve la longitud en bits de una cadena de texto, es decir, cuántos bits se necesitan para almacenar esa cadena en su formato binario

17. UUID()

SELECT UUID() AS identificador_unico;
Genera un Identificador Único UniversaL

18. REPEAT()

Repite un string n veces
SELECT REPEAT('SQL', 3);
Devuelve "SQLSQLSQL".

19. CONV()

Convierte un número de una base a otra (decimal, binario, hexadecimal, etc.)
SELECT CONV(255, 10, 16);
Convierte 255 de decimal a hexadecimal (FF).

20. DAYOFYEAR()

Devuelve el número de día en el año (1-366).
SELECT DAYOFYEAR('2025-02-05');
Devuelve 36 porque es el día 36 del año.

21. MAKEDATE()

Crea una fecha a partir de un año y un número de día en el año.
SELECT MAKEDATE(2025, 36);
Devuelve '2025-02-05'.

22. ST_DISTANCE()

Calcula la distancia entre dos puntos geoespaciales.
SELECT ST_Distance(

ST_GeomFromText('POINT(0 0)'),
ST_GeomFromText('POINT(3 4)')

);
Devuelve 5, usando la fórmula de distancia euclidiana.

23. LAST_DAY()

Devuelve el último día del mes para la fecha dada.
SELECT LAST_DAY('2025-02-05');
Devuelve '2025-02-28'.

24. ST_AREA()

Calcula el área de una figura geométrica.
SELECT ST_Area(ST_GeomFromText('POLYGON((0 0, 4 0, 0 4, 4 4,))'));
Devuelve 16 (área del cuadrado 4x4).

25. ELT()

Devuelve el valor de una lista según el índice dado.
SELECT ELT(2, 'Rojo', 'Verde', 'Azul');
Devuelve 'Verde' (porque es el segundo elemento).

26. ROUND()

Esta función se utiliza para redondear un número a un número específico de decimales.
Por ejemplo:
SELECT ROUND(4.68778, 2);
Resultado: → “4.69”

27. ASCII()

Esta función devuelve el valor de la letra en la tabla ASCII del primer carácter de un string.
Por ejemplo:
SELECT ASCII('C');
Resultado: → “67”

28. SPACE()

La función genera un string con un número determinado de espacios.
Por ejemplo:
SELECT SPACE(8);
Resultado: → “ “ (String de separación de 8 espacios)

29. CONCAT_WS()

Esta función concatena strings utilizando cualquier símbolo introducido como separador.
Por ejemplo:
SELECT CONCAT_WS('', 'Trabajo', 'SQL', '!');**
Resultado: → 'Trabajo**SQL**!'

30. FIELD()

Esta función devuelve la posición de una cadena específica en una lista de cadenas.
Por ejemplo:
"Obtiene la posición de "cabra" en la lista "a, cabra, c, d"
SELECT FIELD('cabra', 'a', 'cabra', 'c', 'd');
Resultado: → 2
El primer parámetro de la función es el elemento a buscar y los elementos siguientes corresponden al array donde hay que
buscar el elemento para mostrar su índice.

31. CONVERT()

La función se utiliza para convertir datos a otros tipos de datos.
Por ejemplo:
SELECT CONVERT('165', UNSIGNED);
Resultado: → 165
Esto convierte el string/texto "165" al número 165.

32. Nitle() Over()

SELECT id, nombre, salario, NITLE(3) OVER(ORDER BY salario) AS grupo
FROM empleados;
Explicación: La función NTILE(n) divide el conjunto de resultados en n grupos casi iguales. Los datos se ordenan según la cláusula ORDER BY dentro de la ventana OVER(), y cada fila se asigna a un grupo.
Ejemplo:
Imagina que tenemos una tabla empleados con los siguientes datos:
id nombre salario
1 Ana 3000
2 Luis 2500
3 María 4000
4 Juan 5000
5 Sofía 3500

6 Pedro 2800
Si ejecutamos:
2 SELECT id, nombre, salario,
NTILE(3) OVER(ORDER BY salario) AS grupo
FROM empleados;
Los empleados se ordenan por salario, y se dividen en 3 grupos aproximadamente iguales:
id nombre salario grupo
2 Luis 2500 1
6 Pedro 2800 1
1 Ana 3000 2
5 Sofía 3500 2
3 María 4000 3
4 Juan 5000 3

�� Grupo 1: Salarios más bajos
�� Grupo 2: Salarios intermedios
�� Grupo 3: Salarios más altos
El resultado dependerá del número total de filas y de cuántos grupos (NTILE(n)) especificamos.

33. Ceil

SELECT ceil (5.4), ceil (5.5), ceil (5.6); → Devuelve 6 en todos los casos
La función CEIL() devuelve el valor entero más pequeño que sea mayor o igual a un número.
Nota: Esta función es igual a la función CEILING() .

34. Mod

SELECT mod (5,2); → 1
La función MOD() devuelve el resto de un número dividido por otro número. Se puede usar para mostrar los pares e impares por ejemplo.

35. Extract

SELECT extract (day from date '2017-1-1'), extract(month from date '2017-1-1'),
extract (year from date '2017-1-1');
La función EXTRACT() extrae una parte de una fecha determinada.

36. Sign

SELECT sign(-5), sign(0) ,sign(5); → -1, 0, +1
La función SIGN() devuelve el signo de un número.
Esta función devolverá uno de los siguientes:

Si número > 0, devuelve 1
Si número = 0, devuelve 0
Si el número < 0, devuelve -1

37. Quantity IF

SELECT OrderID, Quantity IIF (Quantity > 10, 'More', 'Less')
FROM OrderDetails;
La función IIF() devuelve un valor si una condición es VERDADERA, u otro valor si una condición es FALSA.

38. CHARINDEX

SELECT CHARINDEX ('OM' , 'Customer') AS MatchPosition; → 4
La función CHARINDEX() busca una subcadena en una cadena y devuelve la posición.
Si no se encuentra la subcadena, esta función devuelve 0.

Nota: Esta función realiza una búsqueda sin distinguir entre mayúsculas y minúsculas.

39. RPAD

SELECT RPAD("Pepe", 20, "ABC"); → PepeABCABCABCABCABCA
La función RPAD() rellena una cadena con otra cadena hasta una longitud determinada.
Nota: observe también la función LPAD() .

40. DATE_TRUNC

SELECT DATE_TRUNC ('MONTH', date '2017-04-02'); → '2017 - 04 - 01'
La función DATE_TRUNC trunca un valor de fecha, hora o indicación de fecha y hora en la unidad de tiempo especificada. Si tienes una fecha, pero quieres tener sólo el primer día del mes de esa fecha, truncamos el valor de la fecha a un parámetro "mes".

41. NULLIF

SELECT NULLIF(25, 25);
La función NULLIF() compara dos expresiones y devuelve NULL si son iguales. De lo contrario, se devuelve la primera expresión.

42. LAST_INSERT_ID

SELECT LAST_INSERT_ID();

LAST_INSERT_ID() (sin argumentos) devuelve el primer valor generado automáticamente que se insertó correctamente en una columna AUTO_INCREMENT como resultado de la última instrucción INSERT ejecutada. El valor de LAST_INSERT_ID() permanece sin cambios si no se insertan filas correctamente.
Si se le da un argumento a LAST_INSERT_ID(), se devolverá el valor de la expresión y la siguiente llamada a LAST_INSERT_ID() devolverá el mismo valor.

```
CREATE TABLE t (
id INTEGER UNSIGNED AUTO_INCREMENT PRIMARY KEY,
f VARCHAR(1))
ENGINE = InnoDB;
INSERT INTO t(f) VALUES(&amp;#39;a&amp;#39;);
SELECT LAST_INSERT_ID();
1
INSERT INTO t(f) VALUES(&amp;#39;b&amp;#39;);
INSERT INTO t(f) VALUES(&amp;#39;c&amp;#39;);
SELECT LAST_INSERT_ID();
3
```