1. GETDATE() 2
2. CASE THEN ELSE 2
3. FORMAT(GETDATE(), 'd') 2
4. TOP(5) 2
5. TOP(50) PERCENT 2
6. LEN() 3
7. YEAR(GETDATE()) 3
8. YEAR(expresión que date) 3
9. AVG() 3
10. COUNT(extel) 3
11. COUNT(DISTINCT extel) 3
12. COUNT(*) 4
13. SELECT NOMEM, SALAR + CASE WHEN NUMHI > 2 THEN (30(NUMHI-2)) ELSE 0 END AS SALAR* 4
14. UNION 4
15. Producto Cartesiano (FROM tabla, tabla) 4
16. SUBSTRING(expresión, inicio, fin) 4
17. SUM(expresión) 5
18. MAX(expresión) 5
19. MIN(expresión) 5
20. IN 5
21. <> 5
22. JOIN 6
23. SELF JOIN 6
24. LEFT JOIN 6
25. RIGHT JOIN 6
26. FULL JOIN 6
27. BETWEEN 7
28. LIKE 7
29. EXISTS (SUBSELECT) OR NOT EXISTS(SUBSELECT) 7
30. UPPER() 7
31. LOWER() 7
32. ROUND() 8
33. GROUP BY y HAVING 8
34. DATEDIFF(YEAR, fecha_nacimiento, GETDATE()) 8
35. INSERT INTO (tabla) VALUES 8
36. UPDATE (tabla) SET 8
37. DELETE FROM 8
38. BEGIN TRANSACTION; 9
39.CONVERT(tipo de dato,EXPRESION) 9

1. GETDATE()

Devuelve la fecha y hora actuales del sistema.

```
SELECT GETDATE();
-- Ejemplo: 2025-02-07 15:00:00.000
```

2. CASE THEN ELSE

Permite realizar condiciones dentro de una consulta SQL. Es similar a un `IF` en programación.

```
SELECT
    CASE
        WHEN NUMHI > 2 THEN 'Mayor que 2'
        ELSE 'Menor o igual a 2'
    END AS Estado
FROM Empleados;
```

3. FORMAT(GETDATE(), 'd')

Devuelve la fecha actual en un formato específico. `'d'` es el formato corto de la fecha.

```
SELECT FORMAT(GETDATE(), 'd');
-- Ejemplo: 2/7/2025
SELECT FORMAT(Edad,'dd-MMMM-yyyy') AS 'Edad'
FROM dbo.actor
```

4. TOP(5)

Limita el número de registros devueltos.

```
SELECT TOP(5) * FROM Empleados;
```

5. TOP(50) PERCENT

Devuelve un porcentaje de registros. En este caso, el 50% de las filas.

```
SELECT TOP(50) PERCENT * FROM Empleados;
```

6. LEN()

Devuelve la longitud de una cadena de texto.

```
SELECT LEN('Hola Mundo');
-- Ejemplo: 10
```

7. YEAR(GETDATE())

Extrae el año de una fecha.

```
SELECT YEAR(GETDATE());
-- Ejemplo: 2025
```

8. YEAR(expresión que date)

Extrae el año de una columna de tipo `DATE` o `DATETIME`.

```
SELECT YEAR(FechaNacimiento) FROM Empleados;
```

9. AVG()

Devuelve el promedio de una expresión numérica.

```
SELECT AVG(Salario) FROM Empleados;
```

10. COUNT(extel)

Cuenta el número de filas donde la columna no es `NULL`.

```
SELECT COUNT(Salario) FROM Empleados;
```

11. COUNT(DISTINCT extel)

Cuenta el número de valores únicos de una columna.

```
SELECT COUNT(DISTINCT Departamento) FROM Empleados;
```

12. COUNT(*)

Cuenta el número total de filas de una tabla.

```
SELECT COUNT(*) FROM Empleados;
```

13. *SELECT NOMEM, SALAR + CASE WHEN NUMHI > 2 THEN (30(NUMHI-2)) ELSE 0 END AS SALAR**

Realiza un cálculo condicional sobre el salario.

```
SELECT NOMEM, SALAR + CASE WHEN NUMHI > 2 THEN (30*(NUMHI-2)) ELSE 0
END AS SALAR
FROM Empleados;
```

14. UNION

Combina los resultados de dos consultas, eliminando duplicados. Solo se puede usar `ORDER BY` al final.

```
SELECT NOMEM, Salario FROM Empleados
UNION
SELECT NOMEM, Salario FROM Contratistas
ORDER BY Salario DESC;
```

15. Producto Cartesiano (FROM tabla, tabla)

Realiza un cruce entre todas las filas de las tablas involucradas. Puede ser peligroso si no se pone un `JOIN` adecuado.

```
SELECT * FROM Empleados, Departamentos;
```

16. SUBSTRING(expresión, inicio, fin)

Extrae una subcadena de una cadena de texto.

```
SELECT SUBSTRING(NOMEM, 1, 3) FROM Empleados;
-- Extrae los primeros 3 caracteres del nombre
```

17. SUM(expresión)

Suma todos los valores de una columna.

```
SELECT SUM(Salario) FROM Empleados;
```

18. MAX(expresión)

Devuelve el valor máximo de una columna.

```
SELECT MAX(Salario) FROM Empleados;
```

19. MIN(expresión)

Devuelve el valor mínimo de una columna.

```
SELECT MIN(Salario) FROM Empleados;
```

20. IN

Es equivalente a usar `= ANY` para comparar si un valor está dentro de un conjunto de valores.

```
SELECT * FROM Empleados WHERE Departamento IN ('Ventas',
'Marketing');
```

21. <>

Significa "distinto de" o "no igual a".

```
SELECT * FROM Empleados WHERE Departamento <> 'Ventas';
```

22. JOIN

Une dos tablas basándose en una condición.

```
SELECT E.NOMEM, D.NombreDepto
FROM Empleados E
JOIN Departamentos D ON E.DepartamentoID = D.ID;
```

23. SELF JOIN

Un tipo de `JOIN` donde una tabla se une consigo misma.

```
SELECT A.NOMEM AS Empleado, B.NOMEM AS Supervisor
FROM Empleados A
JOIN Empleados B ON A.SupervisorID = B.ID;
```

24. LEFT JOIN

Devuelve todas las filas de la primera tabla y las filas coincidentes de la segunda tabla.

```
SELECT E.NOMEM, D.NombreDepto
FROM Empleados E
LEFT JOIN Departamentos D ON E.DepartamentoID = D.ID;
```

25. RIGHT JOIN

Devuelve todas las filas de la segunda tabla y las filas coincidentes de la primera tabla.

```
SELECT E.NOMEM, D.NombreDepto
FROM Empleados E
RIGHT JOIN Departamentos D ON E.DepartamentoID = D.ID;
```

26. FULL JOIN

Devuelve todas las filas cuando hay una coincidencia en cualquiera de las dos tablas.

```
SELECT E.NOMEM, D.NombreDepto
FROM Empleados E
FULL JOIN Departamentos D ON E.DepartamentoID = D.ID;
```

27. BETWEEN

Compara si un valor está dentro de un rango.

```
SELECT * FROM Empleados WHERE Salario BETWEEN 2000 AND 5000;
```

28. LIKE

Compara un patrón dentro de una cadena, usando `%` para comodines.

```
SELECT * FROM Empleados WHERE NOMEM LIKE 'A%';
-- Nombres que comienzan con "A"
```

29. EXISTS (SUBSELECT) OR NOT EXISTS(SUBSELECT)

Verifica si existe al menos una fila que cumpla una condición dentro de una subconsulta.

```
SELECT * FROM Empleados E
WHERE EXISTS (SELECT 1 FROM Departamentos D WHERE D.ID =
E.DepartamentoID);
```

30. UPPER()

Convierte una cadena de texto a mayúsculas.

```
SELECT UPPER(NOMEM) FROM Empleados;
```

31. LOWER()

Convierte una cadena de texto a minúsculas.

```
SELECT LOWER(NOMEM) FROM Empleados;
```

32. ROUND()

Redondea un número a un número específico de decimales.

```
SELECT ROUND(Salario, 2) FROM Empleados;
```

33. GROUP BY y HAVING

Agrupa filas y permite realizar funciones agregadas, mientras que `HAVING` filtra esos grupos.

```
SELECT Departamento, AVG(Salario)
FROM Empleados
GROUP BY Departamento
HAVING AVG(Salario) > 3000;
```

34. DATEDIFF(YEAR, fecha_nacimiento, GETDATE())

Devuelve la diferencia entre dos fechas, en este caso, en años.

```
SELECT DATEDIFF(YEAR, FechaNacimiento, GETDATE()) FROM Empleados;
```

35. INSERT INTO (tabla) VALUES

Inserta una nueva fila en una tabla.

```
INSERT INTO Empleados (NOMEM, Salario) VALUES ('Juan Pérez', 2500);
```

36. UPDATE (tabla) SET

Actualiza los valores de una o más columnas en las filas de una tabla.

```
UPDATE Empleados SET Salario = 3000 WHERE NOMEM = 'Juan Pérez';
```

37. DELETE FROM

Elimina filas de una tabla.

```
DELETE FROM Empleados WHERE NOMEM = 'Juan Pérez';
```

38. BEGIN TRANSACTION;

ROLLBACK TRANSACTION;

39.CONVERT(tipo de dato,EXPRESION)

ej.ROUND(AVG(CONVERT(FLOAT,DATEDIFF(YEAR, Edad, GETDATE()))),2)

DATO Para Sacar el máximo o el mínimo de una columna puedes utilizar un ORDER BY con un TOP(1);