GESTIÓN DE DATOS

Actualizar y eliminar datos con Update y Delete.

ING. JAIME LLANOS BARDALES

Introducción

- **Quiz**

Poll Everywhere

Agenda

LENGUAJE TRANSACT SQL SENTENCIAS Y FUNCIONES

- Funciones de agregación.
- Consultas de manipulación de datos.
- Ejercicios prácticos.

Lenguaje Transact SQL / Funciones de Agrupación

Las funciones de grupo operan en conjuntos de filas para dar un resultado por grupo

- AVG
- COUNT
- MAX
- MIN
- SUM

	DEPARTMENT_ID	SALARY
1	10	4400
2	20	13000
3	20	6000
4	110	12000
5	110	8300
6	90	24000
7	90	17000
8	90	17000
9	60	9000
10	60	6000
...		
18	80	11000
19	80	8600
20	(null)	7000

Maximum salary in EMPLOYEES table

MAX(SALARY)
24000

Lenguaje Transact SQL / Funciones de Agrupación

Universidad Nacional de Cajamarca
"Norte de la Universidad Peruana"

Syntaxis:

```
SELECT      group_function(column), ...
FROM        table
[WHERE      condition]
[ORDER BY   column];
```

Function	Description
AVG(exp)	Average value of n, ignoring null values
COUNT({*|[DISTINCT|ALL]expr })	Number of rows, where expr evaluates to something other than null (count all selected rows using *, including duplicates and rows with nulls)
MAX(expr)	Maximum value of expr, ignoring null values
MIN(expr)	Minimum value of expr, ignoring null values
SUM(n)	Sum values of n, ignoring null values

Lenguaje Transact SQL / Funciones de Agrupación

Universidad Nacional de Cajamarca
"Norte de la Universidad Peruana"

Puede usar **AVG** y **SUM** para datos numéricos.

```
SELECT  AVG(salary), MAX(salary),
        MIN(salary), SUM(salary)
FROM    employees
WHERE   job_id LIKE '%REP%';
```

	AVG(SALARY)	MAX(SALARY)	MIN(SALARY)	SUM(SALARY)
1	8150	11000	6000	32600

Puede usar **MIN** y **MAX** para los tipos de datos numéricos, de caracteres y de fecha

```
SELECT  MIN(hire_date), MAX(hire_date)
FROM        employees;
```

	MIN(HIRE_DATE)	MAX(HIRE_DATE)
1	17-JUN-87	29-JAN-00

Lenguaje Transact SQL / Funciones de Agrupación

Universidad Nacional de Cajamarca
"Norte de la Universidad Peruana"

Usando la función COUNT

- COUNT (*)
- COUNT (expr)
- COUNT (DISTINCT expr)

COUNT (*) devuelve el número de filas en una tabla que satisfacen los criterios de la declaración SELECT, incluidas las filas duplicadas y las filas que contienen valores nulos en cualquiera de las columnas. Si se incluye una cláusula WHERE devuelve el número de filas que satisfacen la condición

COUNT (expr) devuelve el número de valores no nulos que están en la columna identificada por expr.

COUNT (DISTINCT expr) devuelve el número de valores únicos no nulos que están en la columna identificada por expr.

`COUNT(*)` returns the number of rows in a table:

1
```
SELECT COUNT(*)
FROM   employees
WHERE  department_id = 50;
```

	COUNT(*)
1	5

`COUNT(expr)` returns the number of rows with non-null values for *expr*:

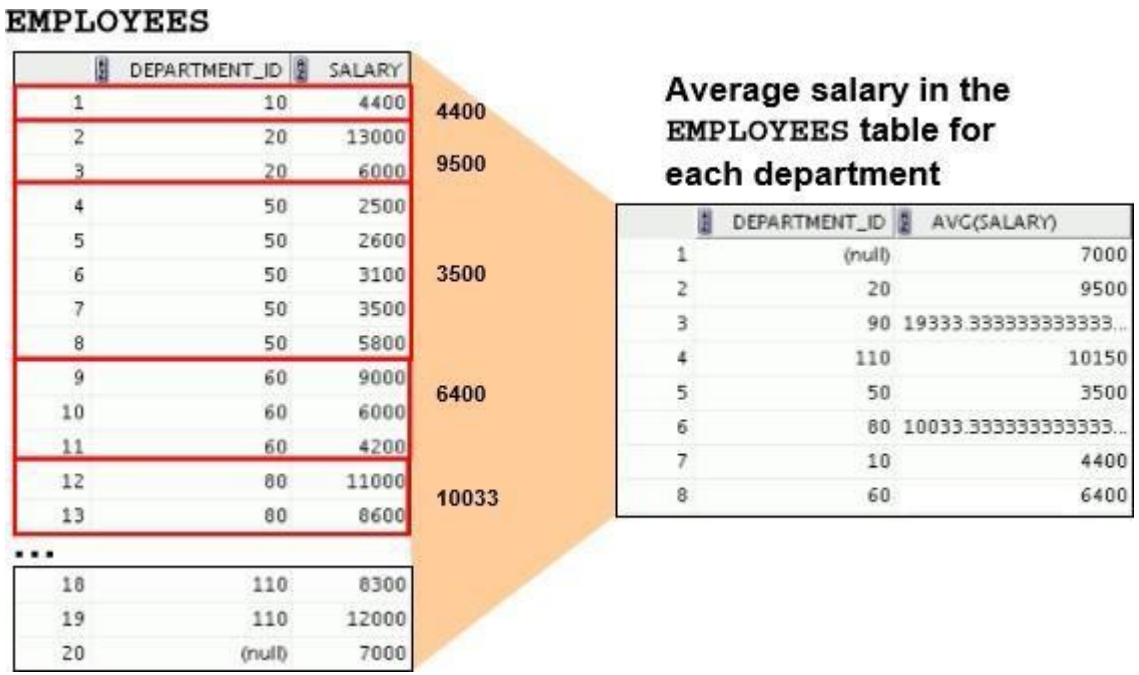
2
```
SELECT COUNT(commission_pct)
FROM   employees
WHERE  department_id = 80;
```

	COUNT(COMMISSION_PCT)
1	3

Lenguaje Transact SQL / Group by

Universidad Nacional de Cajamarca
"Norte de la Universidad Peruana"

- ✓ Puede dividir las filas de una tabla en grupos más pequeños utilizando la cláusula GROUP BY.
- ✓ Las columnas en el SELECT deben coincidir con la clausula
- ✓ La clausula WHERE excluye los registros antes de la agrupación.
- ✓ No se puede usar alias en el GROUP BY
- ✓ Solo se puede excluir las columnas del SELECT cuando aplica a todo el resultado
- ✓ Puede agrupar mas de una columna de diferentes tablas.




```
SELECT      column, group_function(column)
FROM        table
[WHERE      condition]
[GROUP BY group_by_expression]
[ORDER BY column];
```

Lenguaje Transact SQL / Having

Universidad Nacional de Cajamarca
"Norte de la Universidad Peruana"

Restringir los resultados usando la clausula Having

Cuando usa la cláusula **having**, el servidor restringe los grupos de la siguiente manera:

- Las filas están agrupadas
- Se aplica la función de grupo.
- Se muestran los grupos que coinciden con la cláusula **having**

```
SELECT     column, group_function
FROM       table
[WHERE     condition]
[GROUP BY  group_by_expression]
[HAVING    group_condition]
[ORDER BY  column];
```

```
SELECT     department_id, MAX(salary)
FROM       employees
GROUP BY   department_id
HAVING     MAX(salary)>10000 ;
```

	DEPARTMENT_ID	MAX(SALARY)
1	20	13000
2	90	24000
3	110	12000
4	80	11000

Lenguaje Transact SQL / Update

Universidad Nacional de Cajamarca
"Norte de la Universidad Peruana"

Actualizar registros a una tabla

Opción #1 Especificar las columnas en los valores y la tabla

```
UPDATE table_name
SET column1 = value1, column2 = value2, ...
WHERE condition;
```

Opción #2 Utilizando una Sub Consulta

```
UPDATE   tempDataView
SET      marks =
         (
           SELECT marks
           FROM tempData b
           WHERE tempDataView.Name = b.Name
         )
```

Lenguaje Transact SQL / Insert

Universidad Nacional de Cajamarca
"Norte de la Universidad Peruana"

Para insertar valores a una tabla usamos el comando INSERT, se requiere:

1. Nombre de la tabla a donde se insertan los valores
2. Identificar las columnas que reciben los valores
3. Asignar los valores para las columnas que se insertan

```
Sintaxis:
INSERT INTO table_name (column1, column2
, column3, ...)
VALUES (value1, value2, value3, ...);
```

```
INSERT INTO ventas (tienda, fecha, producto, cliente, importe)
VALUES
('Lima','01/01/2019','Papel Bond A4','Librería Lápiz y Papel',250);
```

1 2 3

Observe:

- El carácter que separa a las columnas y los valores es la coma ,
- Los campos de tipo texto inician y terminan con el apostrofe '
- Los valores de tipo numérico se escriben directamente.
- La sentencia termina con el carácter punto y coma ;

Lenguaje Transact SQL / Insert

Universidad Nacional de Cajamarca
"Norte de la Universidad Peruana"

Inserta registros de una tabla

Opción #1 Especificar las columnas en los valores y la tabla

```
INSERT INTO table_name (column1, column2, column3, ...)
VALUES (value1, value2, value3, ...);
```

Opción #2 Especificar las columnas en los valores

```
INSERT INTO table_name
VALUES (value1, value2, value3, ...);
```

Opción #3 Insertar desde una consulta

```
INSERT INTO table_name
SELECT column1, column2, ...
FROM table_name
WHERE condition;
```

Lenguaje Transact SQL / Insert

Universidad Nacional de Cajamarca
"Norte de la Universidad Peruana"

Insertar Registros

Ahora insertaremos los datos de ejemplo:

```
INSERT INTO ventas (tienda, fecha, producto, cliente, importe) VALUES ('Lima','01/01/2019','Papel Bond A4','Librería Lápiz y Papel',250);
INSERT INTO ventas (tienda, fecha, producto, cliente, importe) VALUES ('Lima','01/01/2019','Teclados','Librería  Centro',100);
INSERT INTO ventas (tienda, fecha, producto, cliente, importe) VALUES ('Lima','01/01/2019','Mouse','Librería Culqui',200);
INSERT INTO ventas (tienda, fecha, producto, cliente, importe) VALUES ('Arequipa','01/01/2019','Papel Bond A4','La casa de papel',850);
INSERT INTO ventas (tienda, fecha, producto, cliente, importe) VALUES ('Arequipa','01/01/2019','Teclados','Cabinas Arequipa',75);
INSERT INTO ventas (tienda, fecha, producto, cliente, importe) VALUES ('Arequipa','01/01/2019','Mouse','Centro Técnico Azul',25);
INSERT INTO ventas (tienda, fecha, producto, cliente, importe) VALUES ('Iquitos','01/01/2019','Papel Bond A4','Librería Los Amigos',1500);
INSERT INTO ventas (tienda, fecha, producto, cliente, importe) VALUES ('Iquitos','01/01/2019','Teclados','Cabinas de la Selva',50);
INSERT INTO ventas (tienda, fecha, producto, cliente, importe) VALUES ('Iquitos','01/01/2019','Mouse','Oficentro',150);
```

Lenguaje Transact SQL / Insert

Universidad Nacional de Cajamarca
"Norte de la Universidad Peruana"

Insertar Registros

Ahora insertaremos los datos de ejemplo:

```
INSERT INTO ventas (tienda, fecha, producto, cliente, importe) VALUES ('Lima','01/01/2019','Papel Bond A4','Librería Lápiz y Papel',250);
INSERT INTO ventas (tienda, fecha, producto, cliente, importe) VALUES ('Lima','01/01/2019','Teclados','Librería  Centro',100);
INSERT INTO ventas (tienda, fecha, producto, cliente, importe) VALUES ('Lima','01/01/2019','Mouse','Librería Culqui',200);
INSERT INTO ventas (tienda, fecha, producto, cliente, importe) VALUES ('Arequipa','01/01/2019','Papel Bond A4','La casa de papel',850);
INSERT INTO ventas (tienda, fecha, producto, cliente, importe) VALUES ('Arequipa','01/01/2019','Teclados','Cabinas Arequipa',75);
INSERT INTO ventas (tienda, fecha, producto, cliente, importe) VALUES ('Arequipa','01/01/2019','Mouse','Centro Técnico Azul',25);
INSERT INTO ventas (tienda, fecha, producto, cliente, importe) VALUES ('Iquitos','01/01/2019','Papel Bond A4','Librería Los Amigos',1500);
INSERT INTO ventas (tienda, fecha, producto, cliente, importe) VALUES ('Iquitos','01/01/2019','Teclados','Cabinas de la Selva',50);
INSERT INTO ventas (tienda, fecha, producto, cliente, importe) VALUES ('Iquitos','01/01/2019','Mouse','Oficentro',150);
```

Inserción de Registros masivo

Universidad Nacional de Cajamarca
"Norte de la Universidad Peruana"

Vamos a crear una tabla para poblarla con registros de manera masiva

```
/*  Creamos una tabla para insertar los registros */
create table dbo.resplanilla_jcd
(ID_DPTO NUMERIC DEFAULT 0,
NOM_DPTO VARCHAR(100) DEFAULT '.',
CANT NUMERIC DEFAULT 0,
 SUELDOS NUMERIC(15,3) DEFAULT 0,
 SUELDOMENOR NUMERIC(15,3) DEFAULT 0,
 SUELDOMAYOR NUMERIC(15,3) DEFAULT 0,
 PROMEDIOSUELDO NUMERIC(15,6) DEFAULT 0,
 FECACTUALIZACION DATETIME  DEFAULT '20010101');

/*  Insertamos registros a partir de un select */

insert into dbo.resplanilla_jcd
 select e.id_dpto,
 d.NOMBRE_DPTO,
 count(*) as cant,
 sum(e.sueldo) as sueldos,
 min(e.sueldo) as sueldomenor,
 max(e.sueldo) as sueldomayor,
 round(avg(e.sueldo),2) as promediosueldo,
 GETDATE()
 from dbo.cl_empleados e
 left join dbo.CL_DEPARTAMENTOS d on e.ID_DPTO = d.ID_DPTO
 where isnull(e.id_dpto,0) > 0
 group by e.id_dpto, d.NOMBRE_DPTO;
```

100 %

Results | Messages

	ID_DPTO	NOM_DPTO	CANT	SUELDOS	SUELDOMENOR	SUELDOMAYOR	PROMEDIOSUELDO	FECACTUALIZACION
1	110	ACCOUNTING	2	20300.000	8300.000	12000.000	10150.000000	2021-08-12 05:13:24.833
2	10	ADMINISTRATION	1	4400.000	4400.000	4400.000	4400.000000	2021-08-12 05:13:24.833
3	90	EXECUTIVE	3	58000.000	17000.000	24000.000	19333.330000	2021-08-12 05:13:24.833
4	100	FINANCE	6	51600.000	6900.000	12000.000	8600.000000	2021-08-12 05:13:24.833
5	40	HUMAN RESOURCES	1	6500.000	6500.000	6500.000	6500.000000	2021-08-12 05:13:24.833
6	60	IT	5	28800.000	4200.000	9000.000	5760.000000	2021-08-12 05:13:24.833
7	20	MARKETING	2	19000.000	6000.000	13000.000	9500.000000	2021-08-12 05:13:24.833
8	70	PUBLIC RELATIONS	1	10000.000	10000.000	10000.000	10000.000000	2021-08-12 05:13:24.833
9	30	PURCHASING	6	24900.000	2500.000	11000.000	4150.000000	2021-08-12 05:13:24.833
10	80	SALES	34	304500.000	6100.000	14000.000	8955.880000	2021-08-12 05:13:24.833
11	50	SHIPPING	45	156400.000	2100.000	8200.000	3475.560000	2021-08-12 05:13:24.833

Universidad Nacional de Cajamarca
"Norte de la Universidad Peruana"

REFERENCIAS

- Insert
- https://docs.microsoft.com/es-es/sql/t-sql/statements/insert-transact-sql?view=sql-server- ver15
- Update
- https://docs.microsoft.com/es-es/sql/t-sql/queries/update-transact-sql?view=sql-
- server-ver15
- Where
- https://docs.microsoft.com/es-es/sql/t-sql/queries/where-transact-sql?view=sql-server-ver15
- Funciones de Agregación
- https://docs.microsoft.com/es-es/sql/t-sql/functions/aggregate-functions-transact-sql?view=sql-server- ver15

Agenda

1. Funciones de tipo fecha (dateadd, datediff, etc)
2. Funciones de tipo texto (len, substring, replace etc)
3. Funciones de matemáticas (round, ceiling, floor)
4. Funciones de conversión (cast, convert)
5. Ejercicios prácticos

Lenguaje Transact SQL / Funciones fecha

Universidad Nacional de Cajamarca
"Norte de la Universidad Peruana"

DATEADD : Devuelve un valor datetime nuevo que se basa en la suma de un intervalo a la fecha especificada.

Sintaxis
DATEADD (*partedeFecha , numero, Fecha)*

partedeFecha	Abreviatura
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

Adicionar 15 días a la fecha del pedido.

```
SELECT DATEADD(day, 15, orderdate) AS Fecha15d
FROM Orders
```

Lenguaje Transact SQL / Funciones fecha

- **DATEDIFF** : Devuelve el número de días, meses o años que han transcurrido entre dos fechas especificadas.
- *Sintaxis*
- DATEDIFF (*parteFecha* , *Fechainicial* , *fechafinal*)

Mostrar los días transcurridos entre el pedido y la fecha actual

```
SELECT OrderID, CustomerID, OrderDate,
DATEDIFF(YEAR,orderdate,GETDATE()) AS "Años Transcurridos"
FROM Orders
```

OrderID	CustomerID	OrderDate	Años Transcurridos
10248	VINET	1996-07-04 00:00:00.000	24
10249	TOMSP	1996-07-05 00:00:00.000	24
10250	HANAR	1996-07-08 00:00:00.000	24
10251	VICTE	1996-07-08 00:00:00.000	24
10252	SUPRD	1996-07-09 00:00:00.000	24

- La función GetDate()
- y la hora Actual
- devuelve la Fecha

```
select getdate()
```

Results | Messages

(No column name)
2020-06-14 08:07:49.613

Lenguaje Transact SQL / Funciones fecha

Universidad Nacional de Cajamarca
"Norte de la Universidad Peruana"

DATENAME : Devuelve una cadena de caracteres que representa la parte de la fecha especificada de la fecha especificada.

Sintaxis
DATENAME (*parteFecha , fecha)*

Adicionar una columna que muestre el nombre del mes pedido,

```
SELECT *, DATENAME (MONTH, ORDERDATE) AS MesPedido
FROM Orders
```

Mostrar los meses y el numero de pedidos, ordenando los meses con mas números de pedidos.

```
SELECT DATENAME(MONTH, ORDERDATE), COUNT(1)
FROM Orders
GROUP BY  DATENAME(MONTH, ORDERDATE)
order by 2 desc
```

Lenguaje Transact SQL / Funciones fecha

Universidad Nacional de Cajamarca
"Norte de la Universidad Peruana"

DAY: Devuelve un entero que representa la parte del día de la fecha especificada.

Sintaxis: **DAY** (fecha)

MONTH : Devuelve un entero que representa el mes de una fecha especificada.

Sintaxis: **MONTH** (fecha)

YEAR : Devuelve un entero que representa la parte de año de la fecha especificada.

Sintaxis: **YEAR** (fecha)

Mostrar en números el, año, mes, día de la fecha de pedido de la tabla pedidos.

```
SELECT ORDERID, ORDERDATE, YEAR(ORDERDATE) 'Año', MONTH(ORDERDATE) 'Mes',
DAY(ORDERDATE) 'Dia'
FROM orders
```

Funciones de tratamiento de cadenas

Universidad Nacional de Cajamarca
"Norte de la Universidad Peruana"

Función	USO
LOWER (VALOR)	Transforma los valores de caracteres a minúsculas
UPPER (VALOR)	Transforma los valores de caracteres a mayúsculas
CONCAT (VALOR1,... VALOR n)	Concatena valores, es similar al uso de SIMBOLO +
SUBSTRING (VALOR, m [,n]), RIGHT(CADENA,NC) o LEFT(CADENA,NC)	Devuelve los n valores a partir del carácter m de una cadena
LEN (VALOR)	Muestra o cuenta la longitud de una cadena o campo
CHARINDEX (VALOR, 'CADENA')	Devuelve la posición de una subcadena dentro de una cadena
RTRIM(CADENA) Y LTRIM(CADENA)	Esta función Elimina todos los caracteres en blanco especificados ya sea desde el principio o el final de una cadena.
REPLACE (TEXTO, 'CADENA_DE_BUSQUEDA', 'CADENA DE REEMPLAZO')	Esta función reemplaza una secuencia de caracteres de una cadena con otro conjunto de caracteres.

Lenguaje Transact SQL / Funciones cadena

Universidad Nacional de Cajamarca
"Norte de la Universidad Peruana"

LEN : Devuelve el número de caracteres de la expresión de cadena dada, excluidos los espacios finales en blanco.

Sintaxis: LEN (expressionTexto)

SUBSTRING : Devuelve una parte de una expresión de caracteres

Sintaxis: SUBSTRING (*expressionTexto , Posicion inicial , numeroCaracteres)*

REPLACE : Reemplaza por una tercera expresión todas las apariciones de la segunda expresión de cadena proporcionada en la primera expresión de cadena.

REPLACE ('*expressionTexto1' , 'expressionTexto2' , 'expressionTexto3')*

REPLICATE Repite una expresión de caracteres un número especificado de veces.

REPLICATE (*expressionTexto , numeroVeces)*

Ejemplos

```
SELECT FirstName, LEN(FirstName), SUBSTRING(FirstName,1,3),REPLICATE(FirstName, 3),
REPLACE(FirstName,'en','UNI')
FROM Employees
```

Lenguaje Transact SQL / Funciones numéricas

Universidad Nacional de Cajamarca
"Norte de la Universidad Peruana"

ROUND: Devuelve una expresión numérica, redondeada a la longitud o precisión especificada.

Sintaxis: ROUND (*numero , longitud)*

CEILING : Devuelve un numero entero más pequeño mayor o igual que la expresión numérica dada.

Sintaxis: CEILING (*expressionNumerica)*

FLOOR : Devuelve el numero entero más grande menor o igual que la expresión numérica dada.

Sintaxis: FLOOR (*expresionNumerica)*

Veremos los diferentes tipos de redondeo

```
SELECT FLOOR(123.65), FLOOR(-123.45), FLOOR($123.45)   --- Redondeo superior
SELECT CEILING($123.45), CEILING($-123.45), CEILING($0.75) --- Redondeo inferior
SELECT ROUND(748.58, -2) ----- Redondeo con parámetros
SELECT ROUND(748.58, -1)
SELECT ROUND(748.58, +1)
SELECT ROUND(748.58, +2)
```

Convertir valores (CONVERT y CAST)

CONVERT y CAST se utilizan para convertir datos de su tipo original a otro tipo de datos. Su sintaxis general es la siguiente:

CONVERT ([tipo de dato] , [expresión que se modificará])

CAST ([expresión que se modificará] **AS** [tipo de dato])

- Convertir un dato **varchar** a entero (**int**):

CONVERT

```
SELECT CONVERT(INT,'01')
```

Results | Messages

	(No column name)
1	1

CAST

```
SELECT CAST('01' AS INT)
```

Results | Messages

	(No column name)
1	1

- **Convertir dato entero (int) a varchar**: En este tipo de conversiones debe tomarse en cuenta que el tipo de dato "varchar()" requiere que se parametrice la cantidad de caracteres (longitud) que tendrá el resultado de la conversión, razón por la cual, si un número entero se convierte a **varchar** con 2 caracteres de longitud (**varchar(2)**), el resultado devuelto será un número con 2 caracteres como máximo.

Ejemplos:

- Se convierte el número "11" a tipo de datos **varchar(2)** (2 caracteres):

CONVERT

```
SELECT CONVERT(varchar(2),11)
```

Results | Messages

	(No column name)
1	11

CAST

```
SELECT CAST(11 AS VARCHAR(2))
```

Results | Messages

	(No column name)
1	11

4. Funciones de conversión

Universidad Nacional de Cajamarca
"Norte de la Universidad Peruana"

Diferencias entre usar CAST y CONVERT:

Solo existen 2 diferencias fundamentales entre utilizar CAST o CONVERT para convertir tipos de datos:

- CAST es soportado por el estándar ANSI mientras CONVERT no lo es.
- CONVERT soporta tipo de datos de fecha (date, datetime...) mientras CAST no, es decir, con CAST no se pueden modificar tipos de datos de fecha.

A pesar de estas diferencias, ambas (CAST y CONVERT) tienen el mismo comportamiento a nivel de performance del script inclusive.

Es recomendable (según información encontrada en blogs de otros autores e inclusive entre la documentación de Microsoft) utilizar CAST debido a que es un estándar ANSI, lo que le da ventajas de compatibilidad en cuanto a los caracteres generados. De todas formas, al momento de convertir datos de fecha se utilizará CONVERT.

Funciones de Fila para conversión – Convert y Cast

Universidad Nacional de Cajamarca
"Norte de la Universidad Peruana"

A continuación se muestran formas de conversión de fechas:

La expresión "GETDATE()" devuelve la fecha actual del sistema:

```
SELECT GETDATE() -- Resultado -> '2016-02-09 13:07:19.543'

SELECT CONVERT(DATE, GETDATE()) -- Resultado -> '2016-02-09'

SELECT CONVERT(SMALLDATETIME, GETDATE()) -- Resultado -> '2016-02-09 13:04:00'


SELECT CONVERT(DATETIME, GETDATE()) -- Resultado -> '2016-02-09 13:03:42.637'
```

Universidad Nacional de Cajamarca
"Norte de la Universidad Peruana"

CAST Y CONVERT

Convierten una expresión de un tipo de datos en otro. **CAST y CONVERT proporcionan funciones similares.**

```
SELECT 'EL PRECIO ES ' + CAST([UnitPrice] AS VARCHAR(12)) , [UnitPrice]
FROM  [dbo].[Products]
WHERE [UnitPrice] > 15.00
```

(No column name)	UnitPrice
EL PRECIO ES 18.00	18,00
EL PRECIO ES 19.00	19,00
EL PRECIO ES 22.00	22,00

Convertir el campo OrderID a cadena para buscar los ID que terminan en 70

```
select *
from orders
where convert(varchar,OrderID) like '%70'
```

OrderID	CustomerID	EmployeeID	OrderDate
10270	WARTH	1	1996-08-01 0(
10370	CHOPS	6	1996-12-03 0(
10470	BONAP	4	1997-03-11 0(

Modificación de tipo de variable

FORMAT (value, format [, culture])

La función FORMAT () formatea un valor con el formato especificado (y una cultura opcional en SQL Server 2017).

Use la función FORMAT () para formatear valores de fecha / hora y valores numéricos. Para conversiones de tipo de datos generales, use CAST () o CONVERT ().

```
SELECT FORMAT(123456789, '##-##-#####'),
FORMAT(123456789, '###,###,###'),
FORMAT (123456789, 'C','en-us')
```

%

Results | Messages

(No column name)	(No column name)	(No column name)
12-34-56789	123,456,789	$123,456,789.00

Modificación de tipo de variable

PARSE (string_value AS data_type [USING culture])

Devuelve el resultado de una expresión, traducida al tipo de datos solicitado en SQL Server.

```
SELECT PARSE('Monday, 27 July 2020' AS datetime2 USING 'en-US') fecha,
PARSE('€345,98' AS money USING 'de-DE') monto
```

Results | Messages

fecha	monto
2020-07-27 00:00:00.0000000	345,98

```
SET LANGUAGE 'English';
SELECT PARSE('12/16/2010' AS datetime2) AS Result;
```

Results | Messages

Result
2010-12-16 00:00:00.0000000

Otras funciones

COALESCE(*val1, val2, ..., val_n*)
ISNULL(*expression, value*)
SELECT NULLIF(25, 25)

La función **COALESCE()** devuelve el primer valor no nulo en una lista.

```
SELECT COALESCE(NULL, 1, 2, 'SQLServer');
```

Results | Messages

(No column name)
1

La función **ISNULL()** devuelve un valor especificado si la expresión es NULL.

```
SELECT ISNULL(NULL,0)
```

Results | Messages

(No column name)
0

La función **NULLIF()** devuelve NULL si dos expresiones son iguales; de lo contrario, devuelve la primera expresión.

```
SELECT NULLIF(0,0)
```

Results | Messages

(No column name)
NULL

REFERENCIAS

Funciones de Cadena

https://docs.microsoft.com/es-es/sql/t-sql/functions/string-functions-transact-sql?view=sql-server-ver15

Funciones Numéricas

https://docs.microsoft.com/es-es/sql/odbc/reference/appendixes/numeric-functions?view=sql-server-ver15

Funciones de Fecha

https://docs.microsoft.com/es-es/sql/t-sql/functions/date-and-time-data-types-and-functions-transact-sql?view=sql-server-ver15

Cast y Convert

https://docs.microsoft.com/es-es/sql/t-sql/functions/cast-and-convert-transact-sql?view=sql-server-ver15

Herramientas de Trabajo

Universidad Nacional de Cajamarca
"Norte de la Universidad Peruana"

Online

SQL Online

SQL Server Management Studio

v. 19.1

Microsoft SQL Server

Fin de la sesión