#CrecimientoProfesional
#AprendeConLosPioneros

Online | DMC Perú

DMC Perú | CAPACITACIÓN PROFESIONAL

SESIÓN VIII

Docente: Victor Gutierrez
Data Architect

Agenda

Manejo Transact SQL Avanzado

- Mas formas de filtrado (in, any and some, all, exists).
- Manejo de operadores de conjunto (union, intersect, except).
- Modificación de tipo de variable (CAST, CONVERT,FORMAT, PARSE).
- Otras funciones (COALESCE, ISNULL and NULLIF).
- Sentencias de control (If-then-end if, If-then-else-end if,while loop, etiquetas).

CAPACITACIÓN PROFESIONAL

Mas formas de filtrado

https://es.wikipedia.org/wiki/SQL

- Los operadores ANY y ALL se usan con una cláusula WHERE o HAVING.
- El operador **ANY** devuelve verdadero si alguno de los valores de subconsulta cumple la condición.
- El operador **ALL** devuelve verdadero si todos los valores de subconsulta cumplen la condición.
- El operador **IN** le permite especificar múltiples valores en una cláusula WHERE.
- El operador IN es una abreviatura para múltiples condiciones OR.

```
SELECT column_name(s)
FROM table_name
WHERE column_name operator ALL / ANY
(SELECT column_name
        FROM table_name
        WHERE condition);
```

```
SELECT column_name(s)
FROM table_name
WHERE column_name IN (value1, value2);
IN (SELECT STATEMENT);
```

```
SELECT column_name(s)
FROM table_name
WHERE EXISTS
(SELECT column_name
    FROM table_name
    WHERE condition);
```

CAPACITACIÓN PROFESIONAL

Mas formas de filtrado

```
select *
from [dbo].[Employees]
where EmployeeID = any (select EmployeeID
                        from Orders
                        group by EmployeeID
                        having count(1) > 100)
```

ANY

Results | Messages

EmployeeID	LastName	FirstName	Title	TitleOfCourtesy
1	Davolio	Nancy	Sales Representative	Ms.
3	Leverling	Janet	Sales Representative	Ms.
4	Peacock	Margaret	Sales Representative	Mrs.
8	Callahan	Laura	Inside Sales Coordinator	Ms.

```
select EmployeeID, COUNT(1)
from Orders
group by EmployeeID
order by 2 desc
```

121 % | Results | Messages

	EmployeeID	(No column name)
1	4	156
2	3	127
3	1	123
4	8	104
5	2	96

Mas formas de filtrado

```
select *
from [dbo].[Employees]
where EmployeeID = all (select EmployeeID
                        from Orders
                        group by EmployeeID
                        having count(1) > 100)
```

Results | Messages

EmployeeID	LastName	FirstName	Title	TitleOfCourtesy	BirthDate	HireDat

ALL

```
select EmployeeID, COUNT(1)
from Orders
group by EmployeeID
order by 2 desc
```

Results | Messages

	EmployeeID	(No column name)
1	4	156
2	3	127
3	1	123
4	8	104
5	2	96

CAPACITACIÓN PROFESIONAL

Mas formas de filtrado

```
select *
from [dbo].[Region] r
left join [dbo].[Territories] t on t.RegionID = r.RegionID
left join [dbo].[EmployeeTerritories] et on et.TerritoryID = t.TerritoryID
left join [dbo].[Employees] e on e.EmployeeID = et.EmployeeID
where r.RegionID in (1,2)
```

21 %

Results | Messages

	RegionID	RegionDescription	TerritoryID	TerritoryDescription	RegionID	EmployeeID	TerritoryID	Employee	LastName
16	1	Eastern	20852	Rockville	1	4	20852	4	Peacock
17	1	Eastern	27403	Greensboro	1	4	27403	4	Peacock
18	1	Eastern	27511	Cary	1	4	27511	4	Peacock
19	1	Eastern	40222	Louisville	1	2	40222	2	Fuller
20	2	Western	60179	Hoffman Estates	2	7	60179	7	King
21	2	Western	60601	Chicago	2	7	60601	7	King
22	2	Western	80202	Denver	2	7	80202	7	King

IN

```
select * from region
```

21 %

Results | Messages

	RegionID	RegionDescription
1	1	Eastern
2	2	Western
3	3	Northern
4	4	Southern

CAPACITACIÓN PROFESIONAL

Mas formas de filtrado

```
select *
from [dbo].[Customers]
where exists
(
    select CustomerID, count(1) NroPedidos
    from [Orders]
    where [Customers].CustomerID = [Orders].CustomerID
    group by CustomerID
    having count(1) > 20)
```

Results | Messages

CustomerID	CompanyName	ContactName	ContactTitle	Address
ERNSH	Ernst Handel	Roland Mendel	Sales Manager	Kirchgasse 6
QUICK	QUICK-Stop	Horst Kloss	Accounting Manager	Taucherstraße 10
SAVEA	Save-a-lot Markets	Jose Pavarotti	Sales Representative	187 Suffolk Ln.

EXISTS

```
select CustomerID, count(1) NroPedidos
from [Orders]
group by CustomerID
```

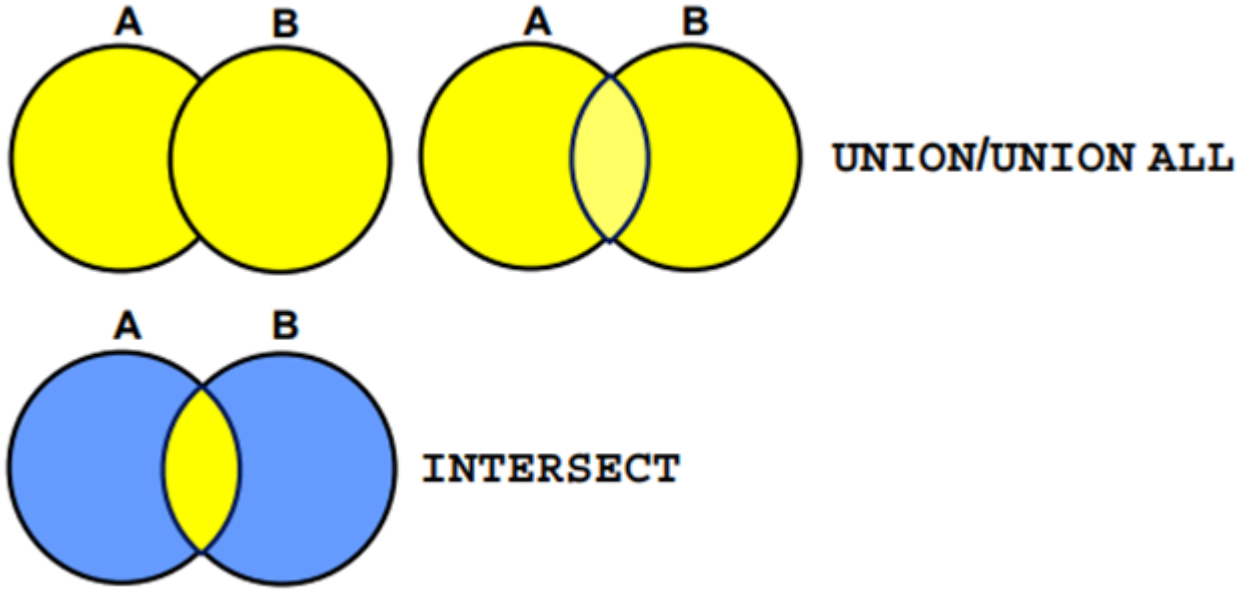
Results | Messages

	CustomerID	NroPedidos
1	ALFKI	6
2	ANATR	4
3	ANTON	7
4	AROUT	13

Manejo de operadores de conjunto

El operador UNION se utiliza para **combinar** el conjunto de resultados de dos o más instrucciones SELECT.

- Cada instrucción SELECT dentro de UNION debe tener el mismo número de columnas.
- Las columnas también deben tener tipos de datos similares.
- Las columnas en cada instrucción SELECT también deben estar en el mismo orden



Manejo de operadores de conjunto

Operador	Retorno
Union	Filas de ambas consultas después de eliminar duplicaciones
Union All	Filas de ambas consultas, incluidas todas las duplicaciones
Intersect	Filas comunes a ambas consultas.

```
select employee_id, job_id from [dbo].[employees]
union
select employee_id, job_id from [dbo].[job_history]
```

Results | Messages

employee_id	job_id
100	AD_PRES
101	AC_ACCOUNT
101	AC_MGR

UNION

```
select employee_id, job_id from [dbo].[employees]
union all
select employee_id, job_id from [dbo].[job_history]
```

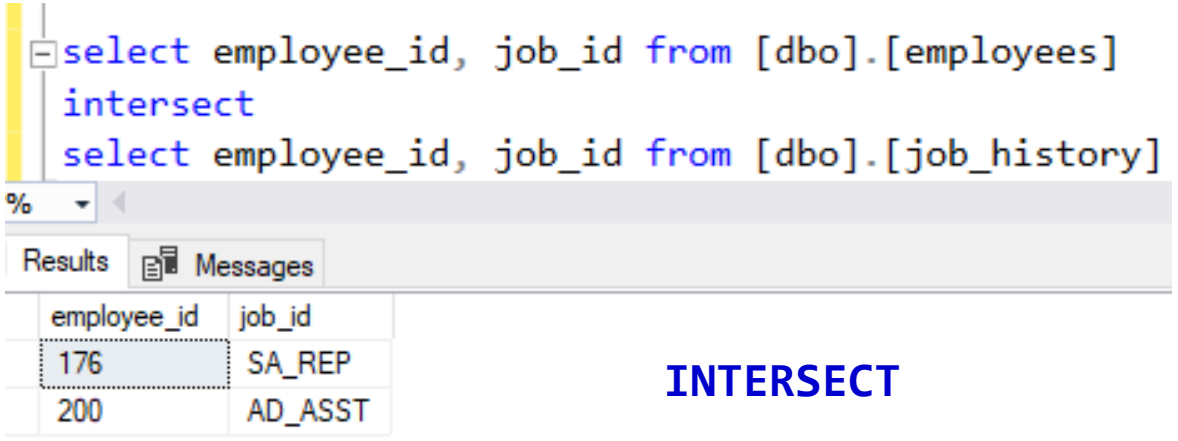
Results | Messages

employee_id	job_id
100	AD_PRES
101	AD_VP
102	AD_VP
103	IT_PROG

UNION ALL

Manejo de operadores de conjunto

Operador	Retorno
Union	Filas de ambas consultas después de eliminar duplicaciones
Union All	Filas de ambas consultas, incluidas todas las duplicaciones
Intersect	Filas comunes a ambas consultas.



INTERSECT

CAPACITACIÓN PROFESIONAL

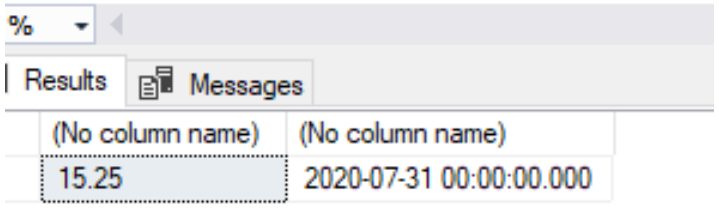
Modificación de tipo de variable

```
CAST(expression AS datatype(length))
CONVERT(data_type(length), expression, style)
```

- CAST es un estándar ANSI-SQL. CONVERT es función de SQL Server
- CAST está disponible en las bases de datos
- Existe una ligera diferencia CONVERT se ejecuta ligeramente mejor que la función CAST

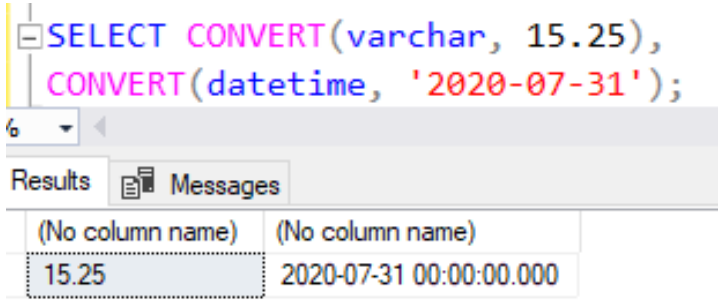
La función **CAST()** convierte un valor (de cualquier tipo) en un tipo de datos especificado.

```
SELECT CAST(15.25 AS varchar),
CAST('2020-07-31' AS datetime);
```



La función **CONVERT()** convierte un valor (de cualquier tipo) en un tipo de datos especificado.

```
SELECT CONVERT(varchar, 15.25),
CONVERT(datetime, '2020-07-31');
```



CAPACITACIÓN PROFESIONAL

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

CAPACITACIÓN PROFESIONAL

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

CAPACITACIÓN PROFESIONAL

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

Sentencias de control

IF THEN ELSE

El Bloque IF permite evaluar una condición que retorna **TRUE** para ejecutar una acción y **ELSE** para ejecutar en caso no cumpla la condición

```
IF
(SELECT COUNT(*) FROM [dbo].[countries]) = 0
PRINT 'No hay filas en la tabla Paises'
ELSE PRINT 'Hay filas en la tabla Paises' ;
```

```
1 %
Messages
Hay filas en la tabla Paises

Completion time: 2020-07-31T20:14:29.5967740-05:00
```

```
-- Uses AdventureWorks

DECLARE @maxWeight float, @productKey integer
SET @maxWeight = 100.00
SET @productKey = 424
IF @maxWeight <= (SELECT Weight from DimProduct WHERE ProductKey=@productKey)
    (SELECT @productKey, EnglishDescription, Weight, 'This product is too heav
ELSE
    (SELECT @productKey, EnglishDescription, Weight, 'This product is availabl
```

Sentencias de control

WHILE loop

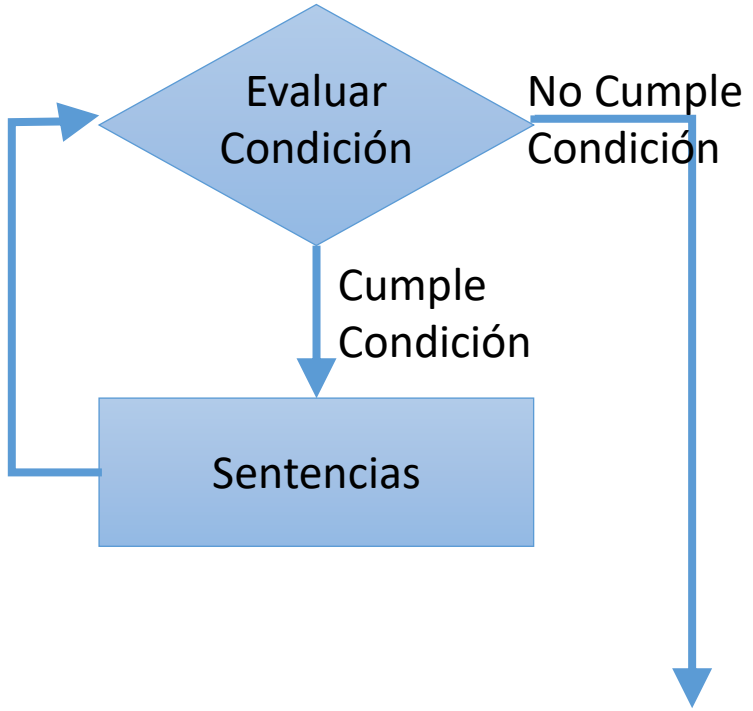
Establece una condición para la ejecución repetida de una instrucción SQL o bloque de instrucciones. Las declaraciones se ejecutan repetidamente siempre que la condición especificada sea verdadera.

```
DECLARE @Fila INT
SET @Fila=1
WHILE ( @Fila <= 10)
BEGIN
    PRINT '# Fila ' + CONVERT(VARCHAR,@Fila)
    SET @Fila  = @Fila  + 1
END
```

Messages

```
# Fila 1
# Fila 2
# Fila 3
# Fila 4
# Fila 5
# Fila 6
# Fila 7
# Fila 8
# Fila 9
# Fila 10
```

```
WHILE condition
BEGIN
    {...statements...}
END
```



Sentencias de control

Etiquetas

Altera el flujo de ejecución a una etiqueta. La instrucción o las instrucciones de Transact-SQL que siguen a GOTO se omiten y el procesamiento continúa en la etiqueta.

```
DECLARE @Fila int;
SET @Fila = 1;
WHILE @Fila < 10
BEGIN
    SELECT @Fila
    SET @Fila = @Fila + 1
    IF @Fila = 2 GOTO Salto_Uno -- Ejecuta el salto
    IF @Fila = 3 GOTO Salto_Dos -- No llega a este punto
END
Salto_Uno:
    SELECT '1- Salio del Bucle'
    GOTO Salto_Tres; -- Evita que se ejecute el Salto_Dos
Salto_Dos:
    SELECT '2- Salto Dos'
Salto_Tres:
    SELECT 'Salto Tres -- Fin';
```

Resultado

(No column name)
1

(No column name)
1- Salio del Bucle

(No column name)
Salto Tres -- Fin

Sentencias de control

Permiten el control de flujo de operaciones.

If Else

Nos Permite Ejecutar Instrucciones Condicionales.

While

Nos permite repetir la ejecución de un conjunto de instrucciones, mientras la condición sea verdadera

Case

La Sentencia Case Compara Un valor con una lista de valores y ejecuta una o más sentencias que corresponde. Y En Caso De No Cumplirse Devolverá Un Valor Por Defecto.

CAPACITACIÓN PROFESIONAL

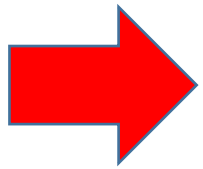
Sentencias de control

Permiten el control de flujo de operaciones.

```
IF <Expresion_Logica>
 <Instruccion>
ELSE
 <Instruccion>
```

If Else

```
IF (select count(1) from [Order Details] where
Quantity > 120) > 0
begin
print 'Pedidos con cantidades > 120'
select orderid, productid, quantity
from [Order Details]
where Quantity > 120
end
ELSE
print 'No se encontraron pedidos con cantidades
mayores a 120'
```



orderid	productid	quantity
10764	39	130
11072	64	130

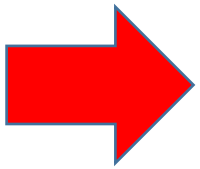
Sentencias de control

Permiten el control de flujo de operaciones.

```
While <Expresion_Logica>
Begin
      <Instrucción Repetición>
End
```

While

```
Declare @FecIni Date, @FecFin Date
Declare @NroFilas Int
Set @FecIni = '1996-07-01'
Set @FecFin = '1996-07-07'
While @FecIni <= @FecFin
Begin
    select @NroFilas = count(1)
    from Orders where OrderDate = @FecIni
    PRINT convert(varchar(10), @FecIni)
    + ' Nro Pedidos ‘
    + convert(varchar(10),@NroFilas)
    set @FecIni = DateAdd(DAY,1,@FecIni)
End
```



```
1996-07-01 Nro Pedidos 0
1996-07-02 Nro Pedidos 0
1996-07-03 Nro Pedidos 0
1996-07-04 Nro Pedidos 1
1996-07-05 Nro Pedidos 1
1996-07-06 Nro Pedidos 0
1996-07-07 Nro Pedidos 0
```

Sentencias de control

Permiten el control de flujo de operaciones.

Case
When <Condicion>
Then <Acciones x Verdad>
Else <Ninguna anterior>

Case

```
select a.*,
    case when a.NroDias < 3 then 'En Tiempo'
    when a.NroDias < 6 then 'En Observacion'
    else 'En Revision'
    end FlgIndicador
from (
    select o.OrderID, o.OrderDate, o.ShippedDate,
    DATEDIFF (day, o.OrderDate, o.ShippedDate) NroDias
    from Orders o) a
```

	NroDias	FlgIndicador
0:00:00.000	12	En Revision
0:00:00.000	5	En Observacion
0:00:00.000	4	En Observacion
0:00:00.000	7	En Revision
0:00:00.000	2	En Tiempo
0:00:00.000	6	En Revision

Referencias

Convert y Cast
https://docs.microsoft.com/en-us/sql/t-sql/functions/cast-and-convert-transact-sql?view=sql-server-ver15

Format
https://docs.microsoft.com/en-us/sql/t-sql/functions/format-transact-sql?view=sql-server-ver15

Intersect
https://docs.microsoft.com/en-us/sql/t-sql/language-elements/set-operators-except-and-intersect-transact-sql?view=sql-server-ver15

Union
https://docs.microsoft.com/en-us/sql/t-sql/language-elements/set-operators-union-transact-sql?view=sql-server-ver15

If Then Else
https://docs.microsoft.com/en-us/sql/t-sql/language-elements/else-if-else-transact-sql?view=sql-server-ver15

While
https://docs.microsoft.com/en-us/sql/t-sql/language-elements/while-transact-sql?view=sql-server-ver15

Agenda

Procedimientos

Objetivo: Manejo de procedimientos para almacenar cadenas de código.

- Concepto y tipos de procedimientos,
- Creación, ejecución y eliminación de procedimientos.

Store Procedures

- Los procedimientos almacenados en SQL son un conjunto de instrucciones de tipo Transact-SQL o un método de acceso a los datos
- Dichas instrucciones se almacenan de forma física con un nombre dentro de la base de datos.

Como unidad de programación en otros lenguajes tienen capacidades:

- Aceptar parámetros de entrada y devolver varios valores en forma de parámetros de salida
- Contienen instrucciones de programación que realicen operaciones en la base de datos.
- Pueden contener llamadas a otros procedimientos.
- Devolver un valor de estado a un programa que realiza una llamada para indicar si la operación se ha realizado correctamente o se han producido errores, y el motivo de estos.

Store Procedures

Tipos de procedimientos almacenados

Tipos Procedimientos	Descripción
Definidos por el usuario	Definido por el usuario, se almacenan en una o mas base de datos
Temporales	Definidos por el usuario, solo que residen en la BD TempDB. Hay dos tipos de procedimientos temporales: locales y globales. Se diferencian entre sí por los nombres, la visibilidad y la disponibilidad. Los procedimientos temporales locales tienen como primer carácter de sus nombres un solo signo de número (#); solo son visibles en la conexión actual del usuario y se eliminan cuando se cierra la conexión. Los procedimientos temporales globales presentan dos signos de número (##) antes del nombre; son visibles para cualquier usuario después de su creación y se eliminan al final de la última sesión en la que se usa el procedimiento.
Sistema	Los procedimientos del sistema se incluyen con SQL Server. Están almacenados físicamente en la base de datos interna y oculta *Resource* y se muestran de forma lógica en el esquema **sys**

Store Procedures

Creación de un Store Procedure

- Se requiere contar con los permisos para crear estos objetos
- Se puede especificar la información relevante a la construcción del Store
- Se puede especificar los parámetros de entrada o salida
- Se debe especificar una operación o validación en el cuerpo del store

```
-- ============================================
-- Author:      <Author,,Name>
-- Create date: <Create Date,,>
-- Description: <Description,,>
-- ============================================
CREATE PROCEDURE <Procedure_Name, sysname, ProcedureName>
    -- Add the parameters for the stored procedure here
    <@Param1, sysname, @p1> <Datatype_For_Param1, , int> = <Default_Value_For_Param1, , 0>,
    <@Param2, sysname, @p2> <Datatype_For_Param2, , int> = <Default_Value_For_Param2, , 0>
AS
BEGIN
    -- SET NOCOUNT ON added to prevent extra result sets from
    -- interfering with SELECT statements.
    SET NOCOUNT ON;

    -- Insert statements for procedure here
    SELECT <@Param1, sysname, @p1>, <@Param2, sysname, @p2>
END
GO
```

Store Procedures

Eliminación de un Store Procedure

- Eliminar un procedimiento puede hacer que los objetos y scripts dependientes produzcan un error
- Se requiere contar con el permiso ALTER en el esquema al que pertenece el procedimiento o el permiso CONTROL en el procedimiento.

```
DROP PROCEDURE <stored procedure name>;
GO
```

Store Procedures

Ejecución de un Store Procedure

- Hay dos formas diferentes de ejecutar un procedimiento almacenado. El primer método y más común es que una aplicación o un usuario llame al procedimiento. El segundo método consiste en establecer el procedimiento para que se ejecute automáticamente cuando se inicie una instancia de SQL Server

```
USE AdventureWorks2012;
GO
EXEC dbo.uspGetEmployeeManagers @BusinessEntityID = 50;
```

DIIC Perú | CAPACITACIÓN **PROFESIONAL**

Online | DIIC Perú