# FUNDAMENTOS DE PROGRAMACION TRANSACT-SQL

- SQL es un lenguaje de consulta para los sistemas de bases de datos relacionales, pero que no posee la potencia de los lenguajes de programación.

- Sin embargo, SQL es la herramienta ideal para trabajar con bases de datos

- TRANSACT SQL es el lenguaje de programación que proporciona Microsoft SQL Server para extender el SQL estándar con otro tipo de instrucciones y elementos propios de los lenguajes de programación.

# CARACTERÍSTICAS GENERALES DEL LENGUAJE

- ✓ Definición de variables.
- ✓ Tipos de datos
- ✓ Estructuras de control de flujo.
- ✓ Gestión de excepciones.
- ✓ Funciones predefinidas.
- ✓ Elementos para la visualización, que permiten mostrar mensajes definidos por el usuario gracias a la cláusula PRINT.

# Estructuras de bloque de programa

- Cada programa está formado por grupos de órdenes SQL llamadas bloques. Cada bloque puede contener, a su vez, nuevos bloques.
- La estructura más sencilla es la siguiente:

```
BEGIN
  Sentencias
  ...
END;
/
```

# VARIABLES

- Una variable es una entidad a la que se asigna un valor. Este valor puede cambiar durante el proceso donde se utiliza la variable.
- Las variables locales se declaran, nombran y escriben mediante la palabra clave declare, y reciben un valor inicial mediante una instrucción select o set.
- A cada variable local se le debe asignar un tipo de dato definido por el usuario
- Los nombres de las variables locales deben empezar con el símbolo “@”. Este símbolo hace que SQL interprete el nombre como un nombre de variable y no un nombre de objeto de la base de datos.
- Por ejemplo: DECLARE @empleados INT. Con esto hemos definido la variable @empleados de tipo entero.

# Ejemplos de variables y sus declaraciones

```
--DECLARA UNA VARIABLE
DECLARE @VARIABLE <TIPO DE DATO>

-- ASIGNA VALOR A UNA VARIABLE
SET @VARIABLE= VALOR
```

**En el ejemplo siguiente, declaramos una variable y le asignamos un valor, la cual será utilizada en una clausula WHERE.**

```
DECLARE @PRECIO DECIMAL
SET @PRECIO = 50
SELECT * FROM COMPRA.PRODUCTOS
WHERE P.PRECIOUNIDAD >  @PRECIO
GO
```

# CONSTANTES

- Una constante es un valor específico o un símbolo que representa un valor de dato específico. El formato de las constantes depende del tipo de datos del valor que representan. Las más utilizadas son:
- Las constantes numéricas se escriben mediante una cadena de números, con la consideración de que el separador decimal es un punto, no una coma, y que si se trata de un valor monetario deberemos incluir la moneda al inicio de la constante. De forma predeterminada, los valores serán positivos. Para indicar lo contrario escribimos el signo - al principio.
- Las constantes de fecha y hora van entre comillas simples y con un formato de fecha y hora adecuado.
- Y las constantes en cadenas de caracteres van entre comillas simples. Por ejemplo: 'Juan García López'.
- Para indicar valores negativos y positivos añadimos el prefijo + o - según sea el valor positivo o negativo.

Estos son algunos ejemplos de constantes **datetime**:

```SQL
'December 5, 1985'
'5 December, 1985'
'851205'
'12/5/98'
```

Estos son algunos ejemplos de constantes **money**:

```SQL
$12
$542023.14
```

Estos son algunos ejemplos de constantes **decimal**:

```SQL
1894.1204
2.0
```

Estos son algunos ejemplos de constantes **integer**:

```SQL
1894
2
```

# Conjunto de Caracteres y Unidades Léxicas

- Identificadores

Los identificadores son los nombres de los objetos de la base de datos: servidores, bases de datos, tablas, vistas, columnas, índices, desencadenadores, procedimientos, restricciones, reglas, etcétera.

Reglas de formato de los identificadores:

- No puede ser una palabra reservada.
- El primer carácter del nombre tiene que ser una letra (según se define en Unicode Standard 2.0) o un carácter de subrayado (*_*).
- Los caracteres siguientes pueden ser letras o números, tal como se define en el Estándar Unicode 2.0, el carácter de subrayado (*_*), *o los caracteres @, $ y #.*

# Conjunto de Caracteres y Unidades Léxicas

## Expresiones y Comparaciones

Una expresión es una combinación de símbolos y operadores que el motor de base de datos de SQL Server evalúa para obtener un único valor. Una expresión simple puede ser una sola constante, variable, columna o función escalar. Los operadores se pueden usar para combinar dos o más expresiones simples y formar una expresión compleja.

Debemos también tener en cuenta que los operadores incluyen varias categorías comunes,  a continuación:

- ✓ *Comparativa*
- ✓ Lógica
- ✓ Aritmética
- ✓ Concatenación
- ✓ Asignación

# Conjunto de Caracteres y Unidades Léxicas

## Expresiones y Comparaciones

Una expresión es una combinación de símbolos y operadores que el motor de base de datos de SQL Server evalúa para obtener un único valor. Una expresión simple puede ser una sola constante, variable, columna o función escalar. Los operadores se pueden usar para combinar dos o más expresiones simples y formar una expresión compleja.

Debemos también tener en cuenta que los operadores incluyen varias categorías comunes,  a continuación

- ✓ *Comparativa*
- ✓ Lógica
- ✓ Aritmética
- ✓ Concatenación
- ✓ Asignación

# Conjunto de Caracteres y Unidades Léxicas

- Al igual que con otros ambientes matemáticos, los operadores están sujetos a normas que rigen la precedencia. La siguiente tabla describe el orden en que los operadores de T-SQL se evalúan, y estos son importantes tenerlos en cuenta a la hora de programar, utilizándolos:

| Orden de evaluación | Operador |
|---|---|
| 1 | ( ) Parentesis |
| 2 | *, /, % (Multiplicación, División, Modulo) |
| 3 | +, - (Añadir / Positivo / Concatenar, Substracción / Negativo) |
| 4 | =, <, >, <=, >=, <>, !=, !>, !< (Comparación) |
| 5 | NOT |
| 6 | AND |
| 7 | BETWEEN, IN, LIKE, OR |
| 8 | = (Asignación) |

# ESTRUCTURAS DE CONTROL

## Estructura Selectiva IF

La estructura IF evalúa una condición lógica y en función del resultado booleano (true o false) se realiza una u otra expression.

# Estructura Selectiva IF - Sintáxis

```
IF <Condición_Lógica>
<BEGIN>
        <Expresiones_CondiciónTrue>
<END>
ELSE
<BEGIN>
        < Expresiones_CondiciónFalse>
<END>
```

# PROCEDIMIENTOS ALMACENADOS

En Transact SQL los procedimientos almacenados pueden devolver valores (numérico entero, texto, fecha, etc.) o conjuntos de resultados. Los procedimientos almacenados al estar dentro de la base de datos pueden contener un conjunto complejo de instrucciones que al utilizarlos nos reducen la complejidad de las instrucciones.

# PROCEDIMIENTOS ALMACENADOS

- Para crear un procedimiento almacenado debemos emplear la sentencia CREATE PROCEDURE.
- Para ejecutar el procedimiento que ya está creado se usa la orden EXECUTE
- Para declarar una variable usamos la orden DECLARE
- Para asignar valores a una variable usamos la orden SET
- Para mostrar mensajes podemos usar PRINT

# EJEMPLO DE PROCEDIMIENTO

**/*Crear un procedimiento almacenado que muestre un saludo*/**

```
CREATE PROCEDURE spSaludo
AS
BEGIN
        PRINT 'Hola Mundo'
END
go
```

# EJEMPLO #2 DE PROCEDIMIENTO

**/*Crear un procedimiento almacenado para sumar dos números (usamos dos parámetros)*/**

```
CREATE PROCEDURE spSuma2
        @dato1 INT,
        @dato2 INT
AS
BEGIN
        DECLARE @suma INT
        SET @suma = (@dato1 + @dato2)
        PRINT 'La suma es: ' + CONVERT(VARCHAR(50), @suma)
END
```

# EJEMPLO #1 DE IF

```
DECLARE
        @Web varchar(100),
        @siglas varchar(5)

   SET @siglas = ‘google'

   IF  @siglas = ‘google'
      BEGIN
         PRINT 'www.google.com'
      END
   ELSE
     BEGIN
         PRINT ‘Otro buscador Web'
     END
```

# ESTRUCTURA CASE

La estructura condicional **CASE** permite evaluar una expresión y devolver un valor u otro.

La sintaxis general de case es:

```
CASE <expresion>
        WHEN <valor_expresion> THEN <valor_devuelto>
        WHEN <valor_expresion> THEN <valor_devuelto>
        ELSE <valor_devuelto> -- Valor por defecto
    END
```

# EJEMPLO DE CASE

```
DECLARE @Web varchar(100),
        @siglas varchar(3)

SET @siglas = ‘google'
   SET @Web = (CASE @siglas
               WHEN ‘google' THEN 'www.google.com'
               WHEN ‘yahoo' THEN 'www.yahoo.com’
ELSE 'www.google.com'
            END)
PRINT @Web
```

# ESTRUCTURAS DE ITERACCIÓN - BUCLE WHILE

- ✓ **El bucle WHILE se repite mientras expresión se evalué como verdadero.**

**Sintaxis:**

```
WHILE <expresion>
   BEGIN
     ...
   END

DECLARE @contador int
   SET @contador = 0
   WHILE (@contador < 100)
   BEGIN
     SET @contador = @contador + 1

   PRINT 'Iteraccion del bucle ‘ + convert(varchar, @contador)
   END
```