T-SQL

> Lenguaje que, a diferencia de SQL, es compilado. Se crea en para una base de datos y es reutilizable.

- *Transact SQL* en Microsoft SQL Server
- *PL-SQL* en otros motores
- Permite la creación de
  - **Vistas**: equivalente a tabla virtual que representa los datos de una o más tablas de forma alternativa
  - **Funciones**: función definida por el usuario. No puede modificar datos
  - **Procedures**: colección de instrucciones T-SQL que pueden tomar y devolver parámetros proporcionados por el usuario. Pueden modificar datos y no están asociados a una tabla
  - **Triggers**: procedimiento especial que se almacena automáticamente, sin parámetros
- Tanto los procedures como los triggers solo retornan True o False según si fueron ejecutados correctamente
- Las vistas y los triggers son exclusivos a T-SQL
- Cuenta con un tipo especial de variable llamados **cursores**, que permiten ampliar el procesamiento de resultados

¿Cuando usamos un cursor?

Usamos un cursor cuando queremos realizar un tratamiento diferenciado para distintas filas del SELECT.

¿Devolver valores de un procedure?

Propiamente, no se puede retornar nada. Cuando un enunciado nos pide editar una tabla y "devolver" un valor, debemos usar un procedure y poblar un parámetro pasado por referencia con ese valor a "retornar". Esto se logra usando el keyword `OUTPUT`.

Herramientas accesorias del lenguaje

Declaración de variables

`DECLARE @local_variable data_type` permite crear una variable de tipo `data_type`. Se les asignan valores mediante instrucciones `SET` o `SELECT`. En Microsoft SQL Server es obligatorio que la variable comience con "@".

Ejemplo: `DECLARE @find varchar(30)`

Impresión

Para imprimir un string o el contenido de una variable en pantalla se usa una sentencia con la siguiente sintaxis:
`PRINT msg_str | @local_variable | string_expr`

Controles de flujo

Bloque de código({} en otros lenguajes)

`BEGIN ... END` encierra un conjunto de instrucciones T-SQL de forma tal que se ejecuten como si fueran una única instrucción.

IF ... ELSE

```
IF <boolean_expression> -- También existe CASE (símil switch)
    <sql_statement> | <statement_block>
[ ELSE
    <sql_statement> | <statement_blocl>]
```

RETURN

Sale incondicionalmente de una consulta o procedimiento

```
RETURN [expression]
```

WAITFOR

Especifica un tiempo, intervalo de tiempo o suceso que desencadena la ejecución de un bloque de instrucciones, procedimiento almacenado o transacción

```
WAITFOR {DELAY 'time' | TIME 'time'}`
```

WHILE

```
WHILE <boolean_expression>
    {<sql_statement> | <statement_block>}
[BREAK] -- sale del while
    {<sql_statement> | <statement_block>}
[CONTINUE] -- salta hacia la siguiente iteración
```

Manejo de errores

Funciones y variables

Función	Descripción
`ERROR_NUMBER()`	Devuelve el número de error
`ERROR_SEVERITY()`	Devuelve la severidad del error
`ERROR_STATE()`	Devuelve el estado del error
`ERROR_PROCEDURE()`	Devuelve el nombre del procedimiento almacenado que ha provocado el error
`ERROR_LINE()`	Devuelve el número de línea en el que se ha producido el error
`ERROR_MESSAGE()`	Devuelve el mensaje de error

También existe la variable `@@ERROR` que almacena el número de error producido por la última sentencia de T-SQL ejecutada, o `0` si no se produjo ninguno.

Niveles de severidad

Si bien se puede extraer información precisa desde los niveles de severidad del error, se agrupan de la siguiente manera:

- [0-10]: *warnings* que no detienen la ejecución
- [11-16]: errores que pueden ser resueltos por el usuario
- [17-19]: errores del motor
- [20-24]: errores irrecuperables

`#todo` seguir