E**X**CELERATOR

MANUAL DE USUARIO

Versión 2022.01



*Tabla de contenido:*

[INTRODUCCION 3](#_Toc85174842)

[REQUISITOS 3](#_Toc85174843)

[PROGRAMACION 3](#_Toc85174844)

[ESTRUCTURA DE TABLAS 4](#_Toc85174845)

[DEFINICIÓN DE PLANTILLAS DE EXCELERATOR 4](#_Toc85174846)

[DATOS DE CABECERA 4](#_Toc85174847)

[VARIABLES 5](#_Toc85174848)

[DATOS DE LINEAS 6](#_Toc85174849)

[FORMULAS DE EXCEL 8](#_Toc85174850)

[TRUCOS DE EXCEL 8](#_Toc85174851)

[IMAGEN VARIABLE EN FUNCION DEL VALOR DE UNA CELDA 8](#_Toc85174852)

# INTRODUCCION

Excelerator es una utilidad para generar documentos en Excel partiendo de una plantilla y rellenándola con datos de nuestra base de datos.

PLANTILLA XLSX —> EJECUCIÓN DE EXCELERATOR —> RESULTADO XLSX

La plantilla es un archivo Excel normal: puede tener textos, formatos, fórmulas, gráficos …

Excelerator lo completará con los datos adecuados obteniendo el archivo final.

* Excelerator lo pueden **USAR** los usuarios normales sin conocimientos de programación.
* Pero para **DEFINIR plantillas** es necesario tener conocimientos informáticos, especialmente de instrucciones SELECT de SQL y conocimientos medios/altos de EXCEL.

# REQUISITOS

Excelerator es una aplicación Windows .NET que requiere Ms Framework 4.6.1

Aunque puede ejecutarse de forma independiente, normalmente funcionará como “utilidad” de otra aplicación de la familia STR.

# PROGRAMACION

Excelerator no requiere programación en aplicaciones que hagan uso de StrMaestroDX.

Sin embargo, desde aplicaciones que no usen StrMaestroDX puede declararse un objeto Excelerator mediante el constructor de la clase \_\_Excelerator:

Public Sub New(ByVal conexion As StrMaestroDX.\_\_Conexion)

Por ejemplo, para ejecutar una plantilla de eXcelerator con una determinada condición SQL:

Dim exc As New \_\_Excelerator(conexion)

Return exc.Ejecutar(plantillaExcelerator, condicion)

# ESTRUCTURA DE TABLAS

Hemos visto que Excelerator se basa en una plantilla Excel, pero toda la definición de como se debe montar la nueva Excel se guarda en las siguientes tablas:

|  |  |  |
| --- | --- | --- |
| Tabla | Alias | Descripción |
| UT\_ExceleratorCab | XCC | Cabecera de la plantilla. Contiene la ruta de la plantilla Excel y del archivo a generar. También contiene el maestro que generará la Excel. |
| UT\_ExceleratorLin | XCL | Líneas. Contiene la instrucción SQL de un conjunto de datos a colocar en la Excel |
| UT\_ExceleratorVar | XCV | Variables que Excelerator solicitará al usuario para ser usadas en las instrucciones SQL de las líneas |

# DEFINICIÓN DE PLANTILLAS DE EXCELERATOR

Es muy fácil definir plantillas de Excelerator. Tan sólo hay que seguir los siguientes pasos:

## DATOS DE CABECERA

* Código: Código único de la plantilla
* Nombre: Descripción de la plantilla
* Archivo plantilla excel: Ruta de la plantilla sobre la que escribimos
* Archivo a generar: Ruta del archivo a generar
* Maestro: Indica desde que pantalla podremos usar esta plantilla



La ruta de la plantilla y del destino puede estar referenciada a la ruta de la aplicación que ejecuta Excelerator con cualquiera de las variables: {{RutaApp}} {{App}} {{Ruta}} {{Usuario}} {{User}}

Por ejemplo:

**RutaPlantilla = {{App}}\Excelerator\Plantilla1.xlsx**

La ruta del destino puede tenera variables de fecha, de usuario y de equipo:

{{Equipo}} {{NombreEquipo}} {{ComputerName}} {{User}} {{Usuario}} {{Now::formato}}

Por ejemplo:

**RutaDestino = {{App}}\Doc\PLZ\_{{Usuario}}\_{{{Equipo}}\_{{Now::yyyyMMddHHmmss}}.xlsx**

## VARIABLES

Cuando ejecutamos una plantilla de Excelerator, normalmente lo llamamos desde un maestro que fuerza una condición. Por ejemplo, el excelerator de las ventas a un cliente, lanzado desde un maestro de clientes podría enviar el siguiente filtro: **Cli.IdCliente=35**

Dicho filtro se pasará a todas las instrucciones SQL de las lineas, por ejemplo:

* Origen: **SELECT Importe FROM VENTAS JOIN CLIENTES Cli ON ….... WHERE @@C@@**
* Final: **SELECT Importe FROM VENTAS JOIN CLIENTES Cli ON ….... WHERE Cli.IdCliente=35**

Esto quiere decir que normalmente no necesitaremos variables para definir nuestros excelerators.

Sin embargo, a veces queremos informes por ejemplo entre dos fechas, por lo que podemos preguntar DESDE, HASTA, o cualquier cosa que se nos ocurra.

**Las variables son valores que se le preguntan al usuario antes de lanzar el proceso.**

Las variables se definen por un identificador numérico, y pueden ser de los tipos:

* T: Texto
* N: Número
* F: Fecha

Cuando las reemplazamos en nuestras intrucciones SQL usamos:

* @@Vx@@ para obtener el valor literal de la variable x.
* @@VSx@@ para el valor SQL de la variable x. Este es el que usaremos habitualmente.

Por ejemplo si tenemos una instrucción SQL con una variable 3 de tipo fecha:

* Origen: SELECT ….... FROM …... WHERE @@C@@ AND V.Fecha=@@VS3@@
* Final: SELECT ….... FROM …... WHERE **Cli.IdCliente=35** AND V.Fecha='20211003’

## DATOS DE LINEAS

Una plantilla puede tener varias lineas. Por ejemplo, un Excelerator que genere facturas tendrá las siguientes líneas:

* Cabecera de factura
* Lineas de factura
* Resumen de ivas de factura
* Totales

Es decir, cada línea de Excelerator representa un conjunto de datos.

Las líneas pueden a su vez ser:

* Normales: Se lanzan siguiendo el campo ORDEN
* Subsecciones: Son llamadas como DETALLE de cada registro de otra línea.

Por ejemplo, siguiendo con el ejemplo de las facturas, si cada línea de factura tuviera lotes, podríamos tener los LOTES de cada LINEA DE FACTURA con una subsección qwue introdujera una condición SQL adicional y se lanzara tras cada línea de factura:

* Cabecera de factura: Normal, orden=1
* Lineas de factura: Normal, orden=2
* Lotes de cada línea: Subsección, orden=1
* Resumen de ivas de factura: Normal, orden=3
* Totales: Normal, orden=4

Es decir, definida dicha plantilla de Excelerator, lo que hará el programa es:

Lanzar todas las líneas NORMALES ordenadas por orden

Si una línea tiene definida una SUBSECCION, la lanzamos por cada registro de dicha línea. Si en el ejemplo lanzamos una factura con 20 líneas de factura, la subsección de lotes se lanzará 20 veces, una vez tras colocar cada línea de factura en la excel.

Este proceso es recursivo: Cada LOTE podría tener una subsección adicional.

Para cada línea indicaremos:

* **Nº línea**: Identificador único de la línea para la plantilla
* **Orden**: Orden en que se lanzará esta línea
* **Instrucción SQL**: Conjunto de datos. Normalmente contendrá la variable **@@C@@** para pasar el filtro adecuado:  **SELECT ..…FROM....WHERE @@C@@ ORDER BY....**
* **Multicolumna**: Se rellenan en la Excel varias columnas
* **NumeroColumnas**: Se rellenan sólo las N primeras columnas. Si se deja a cero se rellenan todas las que tenga la instrucción SQL.
* **Multilinea**: Si queremos pasar varios registros a la Excel
* **Hoja destino**: Hoja dentro de la excel. Si se deja en blanco es la última usada.
* **Fila destino**: Nº de fila destino en la Excel. Si se deja a cero, será la última línea usada.
* Columna destino: Letra de la columna destino.
* **InsertarFilas**: Si se marca, las filas se insertan en la Excel. Sino se sobreescriben. Es importante saber que en caso de que esté marcada la opción MULTILINEA, la primera línea no se insertará. Este suele ser el comportamiento deseado: Si hay una línea no se inserta. Además, las filas insertadas tendrán el formato de celda de las filas anteriores.
* **SaltoFilas**: nº de filas que saltamos en cada registro.
* **SaltoFilasIni**: Nº de filas que saltamos antes del primer registro
* **SaltoFilasFin**: Nº de filas que saltamos tras el último registro
* **CeldaEstilo**: Celda con el estilo que queremos asignar. El estilo incluye: **Fuente, formato, alineación, bordes y color de relleno**
* **CeldaFuente**: Celda que tiene la fuente que queremos dar a las celdas que escribimos
* **CeldaColor**: Color de fondo y de texto que queremos dar a las celdas que escribimos

Cuando asignamos estilos, fuentes o colores, podemos distinguir varios formatos en una misma fila si los separamos por ;.

Por ejemplo, si tenemos una fila con 4 celdas y la primera tiene el estilo de la celda A1 y el resto la de A2 podemos asignar:

CeldaEstilo = ‘A1;A2;=;=’

El signo = coloca el mismo estilo que la celda anterior de la fila.

## FORMULAS DE EXCEL

Podemos usar fórmulas de Excel de dos maneras:

* Fórmulas que están en la hoja Excel que hace de plantilla
* Fórmulas embebidas en las instrucciones SQL

Por ejemplo, podemos poner una línea que en la celda destino C1 pinte esta SQL:

SELECT ‘=A1+B1’

Pero a veces, el número de fila será variable, por lo que podemos usar:

SELECT ‘=A{{Fila}}+B{{Fila}}’ que si estamos en la fila 10 quedará: Celda C10: =A10+B10

Por ello tenemos unas variables especiales:

* {{A}} {{Columna}} {{C}} Columna actual
* {{1}} {{Fila}} {{F}} Fila actual
* {{A1}} Celda actual
* {{Hoja!A1}} Celda actual indicando la hoja
* {{Fila-1}} Fila anterior
* {{Columna-1}} Columna anterior

# TRUCOS DE EXCEL

## TABLAS

Si va a usar información sobre la cual necesite realizar búsquedas, y además puede tener un rango variable, utilice siempre tablas, de tal forma que con una sola palabra (el nombre de la tabla) hacemos referencia a todo el rango y optimizaremos el funcionamiento general de Excel.

## IMAGEN VARIABLE EN FUNCION DEL VALOR DE UNA CELDA

[Cambiar una imagen a partir del valor de una celda en Excel](https://www.youtube.com/watch?v=KgBbKQ7VJgI)

