# SSIS For Loop Container – Detailed Tutorial

## 1. Introduction

The For Loop Container in SSIS allows you to execute one or more tasks repeatedly until a specific condition is met. It works similarly to a 'for' loop in programming languages like C#, Java, or Python.

## 2. When to Use

Use a For Loop Container when:  
- You need to run the same task n number of times.  
- You need to iterate through numeric counters.  
- You are exporting/importing data for multiple IDs or ranges.  
- You are processing files with a fixed numeric sequence.

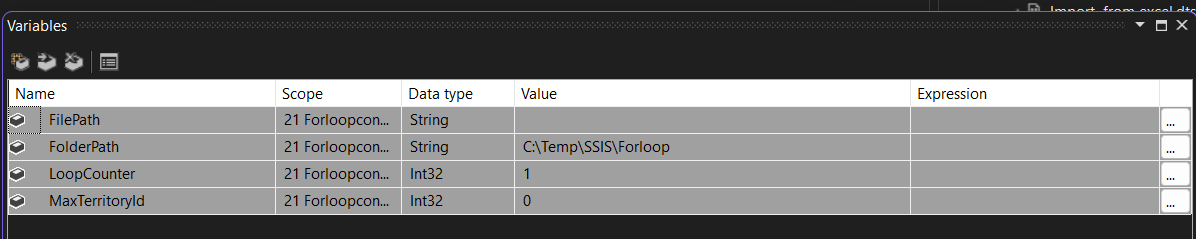
## 3. Real-World Use Case from Transcription

Scenario:  
We have the Sales.Customer table in AdventureWorks2014 database containing ~19,000 records with a column TerritoryID (values 1–10). We want to:  
- Loop through each TerritoryID (1 to MaxTerritoryID).  
- Export data for each TerritoryID into a separate CSV file (e.g., Customer\_1.csv, Customer\_2.csv, ...).

## 4. Step-by-Step Implementation in SSIS

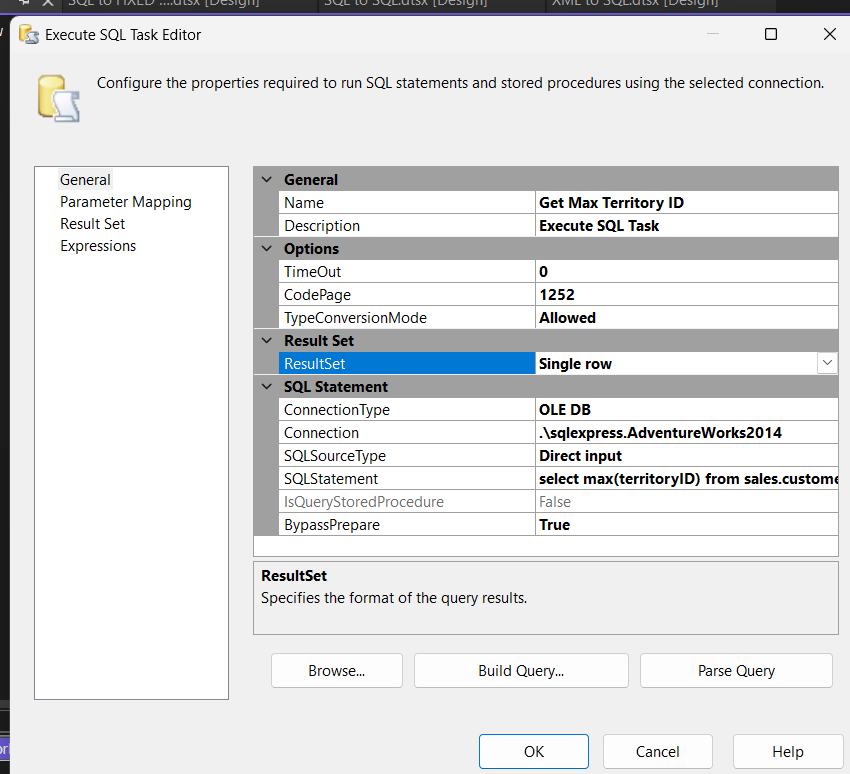
### Step 1 – Create SSIS Variables

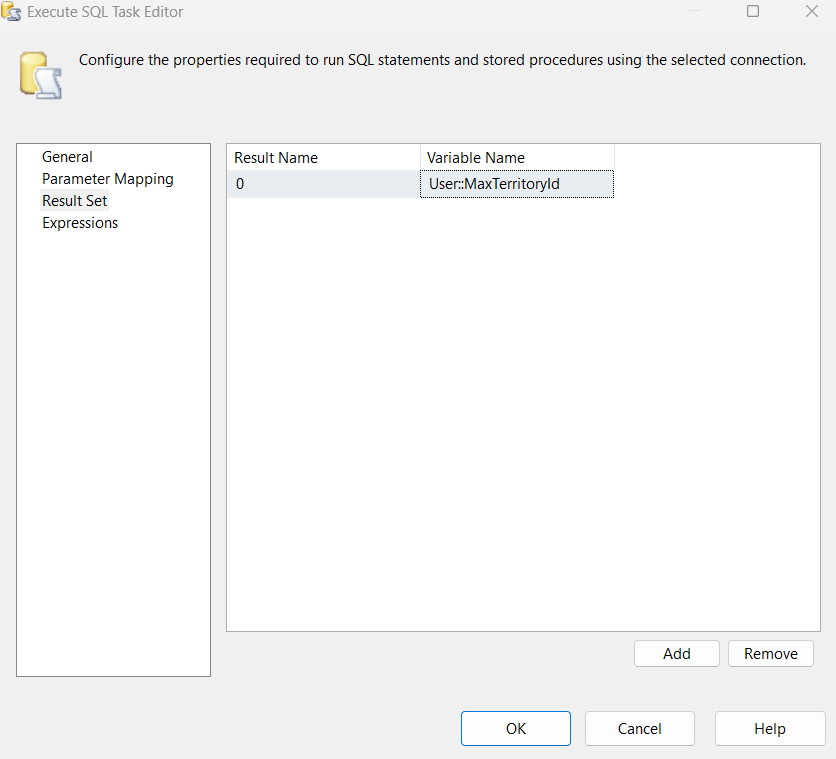
Go to SSIS → Variables and create:  
FolderPath (String) – C:\files  
FilePath (String) – Blank (will be dynamically assigned)  
LoopCounter (Int32) – 1 (starting value)  
MaxTerritoryId (Int32) – 0 (set dynamically)



### Step 2 – Get Max Territory ID

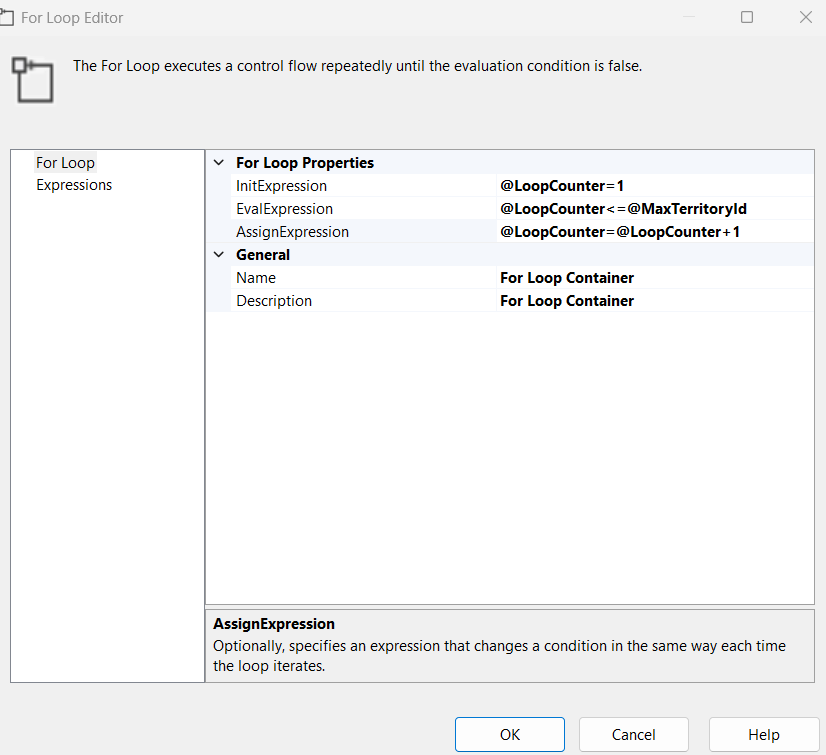
1. Drag an Execute SQL Task to Control Flow.  
2. Name it: Get Max Territory ID.  
3. Configure connection to AdventureWorks2014 DB.  
4. SQL Statement:  
 SELECT MAX(TerritoryID) AS MaxID FROM Sales.Customer  
5. Set Result Set to Single Row and map to User::MaxTerritoryId.





### Step 3 – Add For Loop Container

1. Drag a For Loop Container into Control Flow.  
2. Connect Get Max Territory ID → For Loop Container.  
3. Configure:  
 InitExpression: @LoopCounter = 1  
 EvalExpression: @LoopCounter <= @MaxTerritoryId  
 AssignExpression: @LoopCounter = @LoopCounter + 1



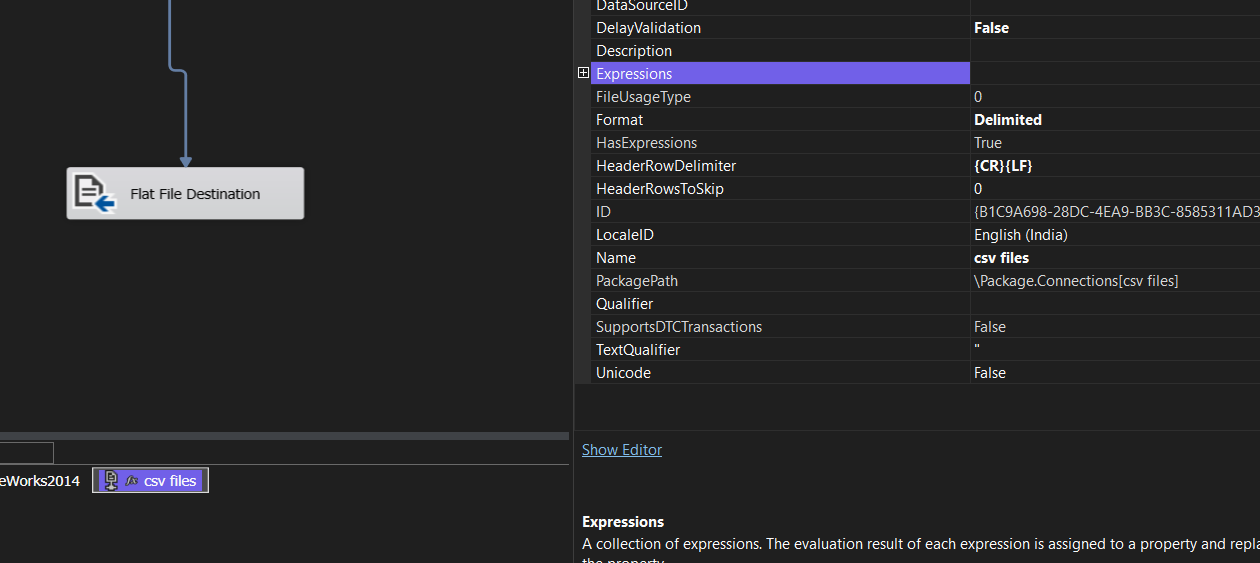
### Step 4 – Add Data Flow Task Inside Loop

Inside the For Loop, drag a Data Flow Task and name it 'Export Territory Data'.

### Step 5 – Configure Data Flow Task

Source – OLE DB Source:  
1. Connect to AdventureWorks2014.  
2. SQL Command:  
 SELECT \* FROM Sales.Customer WHERE TerritoryID = ?  
3. Map parameter to User::LoopCounter.  
  
Destination – Flat File Destination:  
1. Create Flat File Connection Manager.  
2. Initial file name: C:\files\Customer\_1.csv.  
3. Enable column names in first row and text qualifier as double quotes.  
4. Map columns.

### Step 6 – Make File Name Dynamic



In Flat File Connection Manager → Expressions → ConnectionString:  
@[User::FolderPath] + "\\Customer\_" + (DT\_WSTR,12)@[User::LoopCounter] + ".csv"

@[User::FolderPath] + "\\Customer\_" + (DT\_WSTR,12)@[User::LoopCounter] + ".csv"

### Step 7 – Execute Package

1. First task gets MaxTerritoryId.  
2. Loop runs from 1 to MaxTerritoryId.  
3. Each iteration reads and exports data to unique CSV file.

## 5. Expected Output

After execution, files generated in C:\files:  
Customer\_1.csv  
Customer\_2.csv  
...  
Customer\_10.csv  
Each file contains only customers from its respective TerritoryID.

## 6. Key Points to Remember

- For Loop Container works on numeric iteration.  
- Always parameterize queries.  
- Use variables + expressions for dynamic file paths.  
- Ensure variables are scoped correctly.

## 7. Example Variation

Instead of exporting Territory data:  
- Generate 5 monthly sales reports for Jan–May.  
- LoopCounter: 1–5  
- File name: SalesReport\_Month1.csv ... SalesReport\_Month5.csv  
- SQL query filters by month instead of TerritoryID.