# SSIS Sequence Container for Parallel Processing

## 1. Introduction

The Sequence Container in SSIS is used to group multiple tasks together and control their execution. While it is often used to run tasks sequentially, it can also be used to run tasks in parallel, which can significantly reduce execution time for certain workloads.

## 2. When to Use Sequence Container for Parallel Processing

You should use a sequence container to run tasks in parallel when:  
- You have independent tasks that don’t rely on each other’s outputs.  
- You want to optimize execution time by using available CPU and memory resources efficiently.  
- Your workload involves large data processing where multiple operations can run at the same time.

Real-World Examples:

1. Creating Indexes on Large Tables – Instead of creating indexes one by one, split them into multiple tasks and run them in parallel.

2. Loading Multiple Files of Different Formats – Each file type can have its own Data Flow Task running simultaneously.

3. Data Aggregations – Running separate aggregations or transformations at the same time to save processing time.

## 3. Scenario from the Example

We have:  
- A CSV file to load into a SQL Server table (TestData).  
- Need to create 6 indexes on that table after loading.  
- Instead of creating all indexes in one task or sequentially, we will:  
 - Create 3 Execute SQL Tasks, each handling 2 indexes.  
 - Run these tasks in parallel inside a Sequence Container.

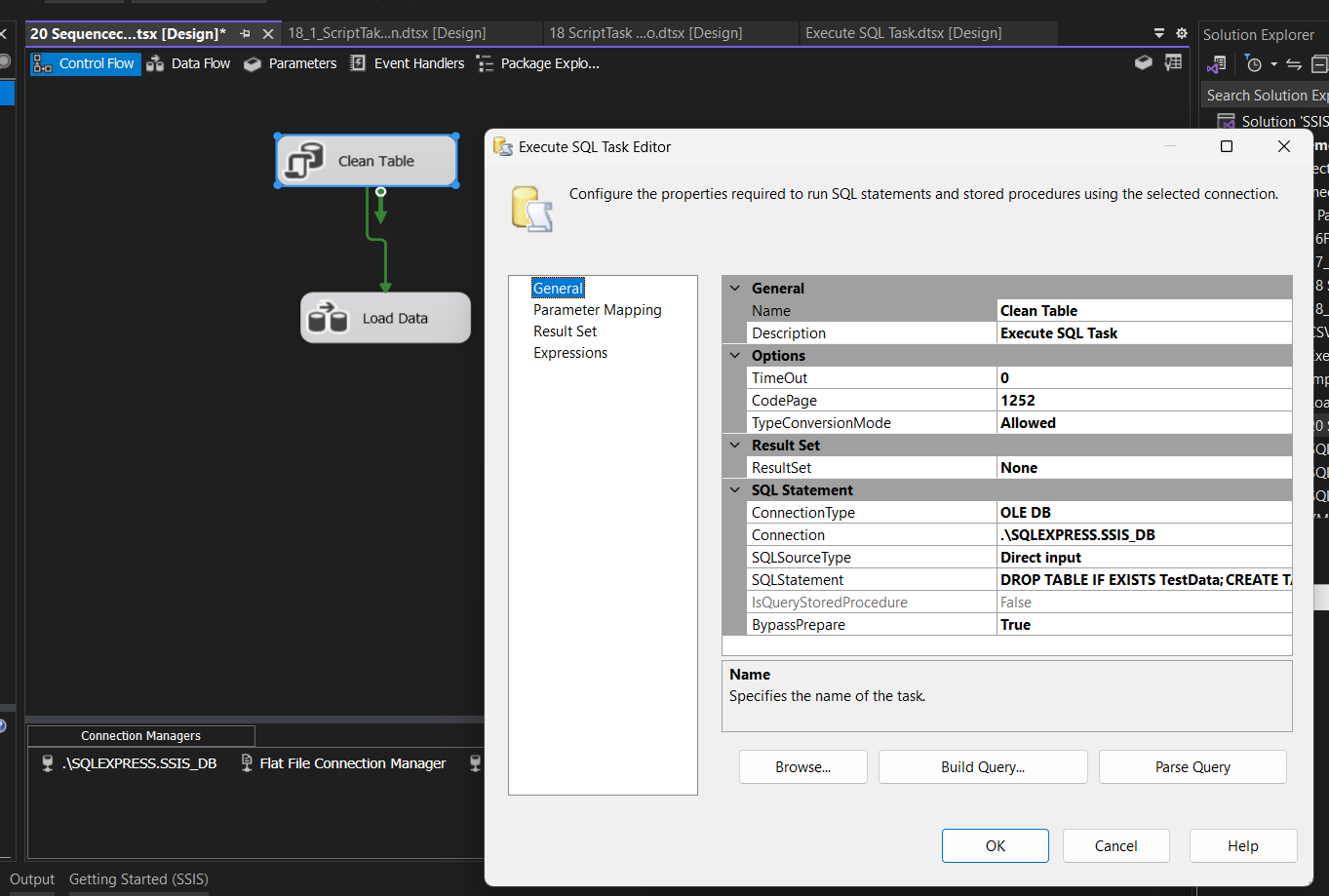
## 4. Step-by-Step Implementation in Visual Studio 2022

Step 1 – Prepare the Table in SQL Server:

DROP TABLE IF EXISTS TestData;  
CREATE TABLE TestData (  
 ID INT IDENTITY(1,1) PRIMARY KEY,  
 Column1 VARCHAR(100),  
 Column2 VARCHAR(100),  
 Column3 VARCHAR(100)  
);

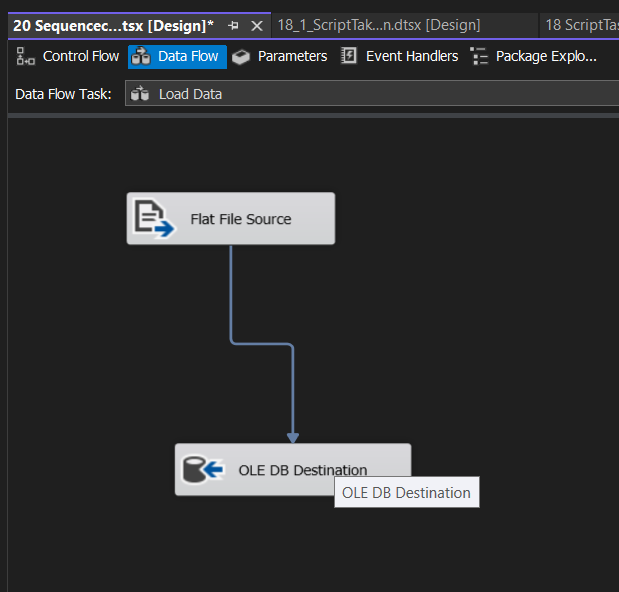
Step 2 – Create a New SSIS Package:

1. Open Visual Studio 2022 → Create a new Integration Services Project.  
2. In Control Flow, drag an Execute SQL Task → Name it 'Clean and Create Table'.  
3. Configure the connection:  
 - Connection Type: OLE DB  
 - Server Name: YourServerName  
 - Database: SSIS\_DB  
4. SQL Statement: Same as above to drop and recreate the table.

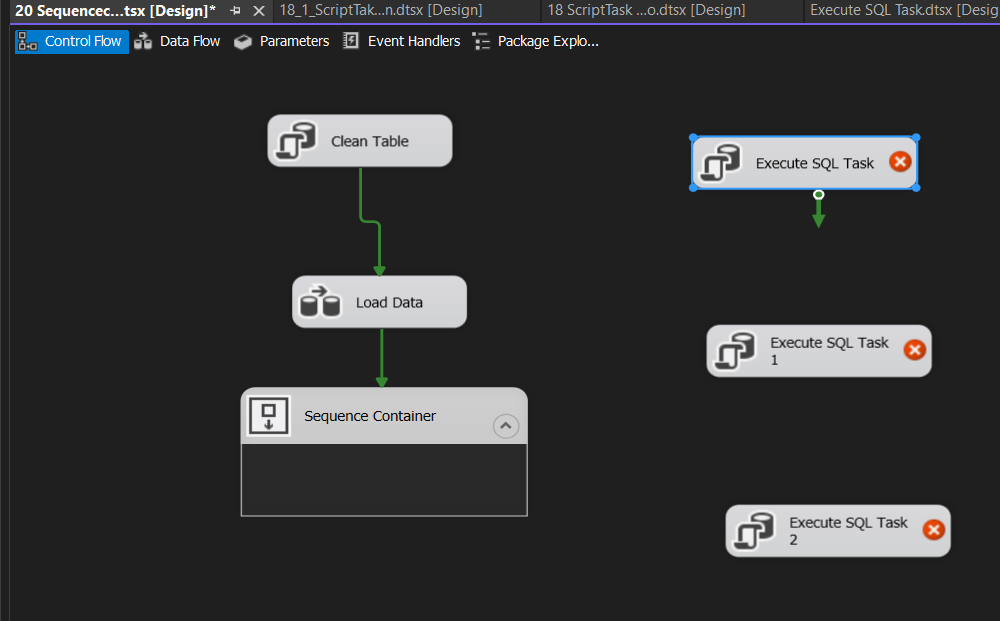
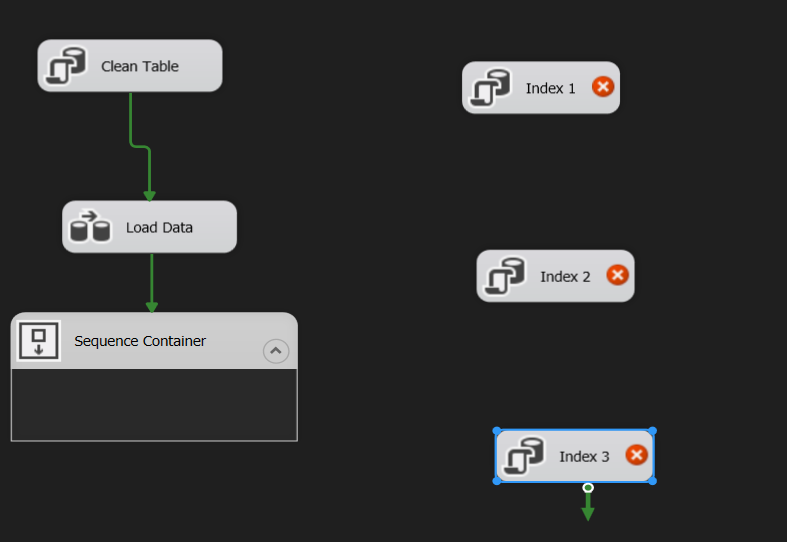


Step 3 – Load Data from CSV into SQL Server Table:

1. Drag a Data Flow Task → Name it 'Load Data'.  
2. Inside Data Flow:  
 - Flat File Source: Browse and select your CSV file, set column delimiters and preview data.  
 - OLE DB Destination: Map the CSV columns to TestData columns.



Step 4 – Create a Sequence Container:

1. Drag a Sequence Container into the Control Flow.  
   2. Connect the Load Data task to the Sequence Container.
2. 
3. 

Step 5 – Add Index Creation Tasks:

1. Inside the Sequence Container, drag three Execute SQL Tasks:  
 - Create Indexes – Task 1  
 - Create Indexes – Task 2  
 - Create Indexes – Task 3

Index 1🡺Right Click 🡺Edit

2. Example SQL for Task 1:  
 CREATE INDEX IDX\_TestData\_Col1 ON TestData(Column1);  
 CREATE INDEX IDX\_TestData\_Col2 ON TestData(Column2);  
3. Task 2:  
 CREATE INDEX IDX\_TestData\_Col3 ON TestData(Column3);  
 CREATE INDEX IDX\_TestData\_Col1\_Col2 ON TestData(Column1, Column2);  
4. Task 3:  
 CREATE INDEX IDX\_TestData\_Col2\_Col3 ON TestData(Column2, Column3);  
 CREATE INDEX IDX\_TestData\_AllCols ON TestData(Column1, Column2, Column3);

Step 6 – Enable Parallel Execution:

Do not connect these index creation tasks to each other inside the Sequence Container. This allows SSIS to run them simultaneously.

Step 7 – Execute and Verify:

1. Run the package.  
2. Execution order: Table cleaned & recreated, Data loaded from CSV, 3 index creation tasks run in parallel.  
3. Check indexes in SQL Server using:  
 SELECT \* FROM sys.indexes WHERE object\_id = OBJECT\_ID('TestData');

## 5. Benefits of Running in Parallel

- Time Savings: Large tables with millions of records benefit greatly.

- Resource Utilization: Uses multiple threads, CPU cores, and SQL Server’s parallel execution capabilities.

- Scalability: Can handle multiple unrelated data processing tasks at once.

## 6. Additional Use Case – Loading Multiple Files

Example:  
- 3 file formats: .csv, .xls, .json  
- Each file has its own Data Flow Task.  
- Put all tasks in a Sequence Container without linking them → they run simultaneously.

## 7. Summary Table

|  |  |  |
| --- | --- | --- |
| Feature | Sequential Execution | Parallel Execution |
| Execution Time | Longer | Shorter |
| Resource Utilization | Lower | Higher |
| Complexity | Low | Moderate |
| When to Use | Tasks depend on each other | Tasks are independent |