Aniket Sarawgi – SSIS CSE STUDY

Exercise Set: SSIS Fundamentals & Implementation

Exercise 1: Environment Setup & Basic Package

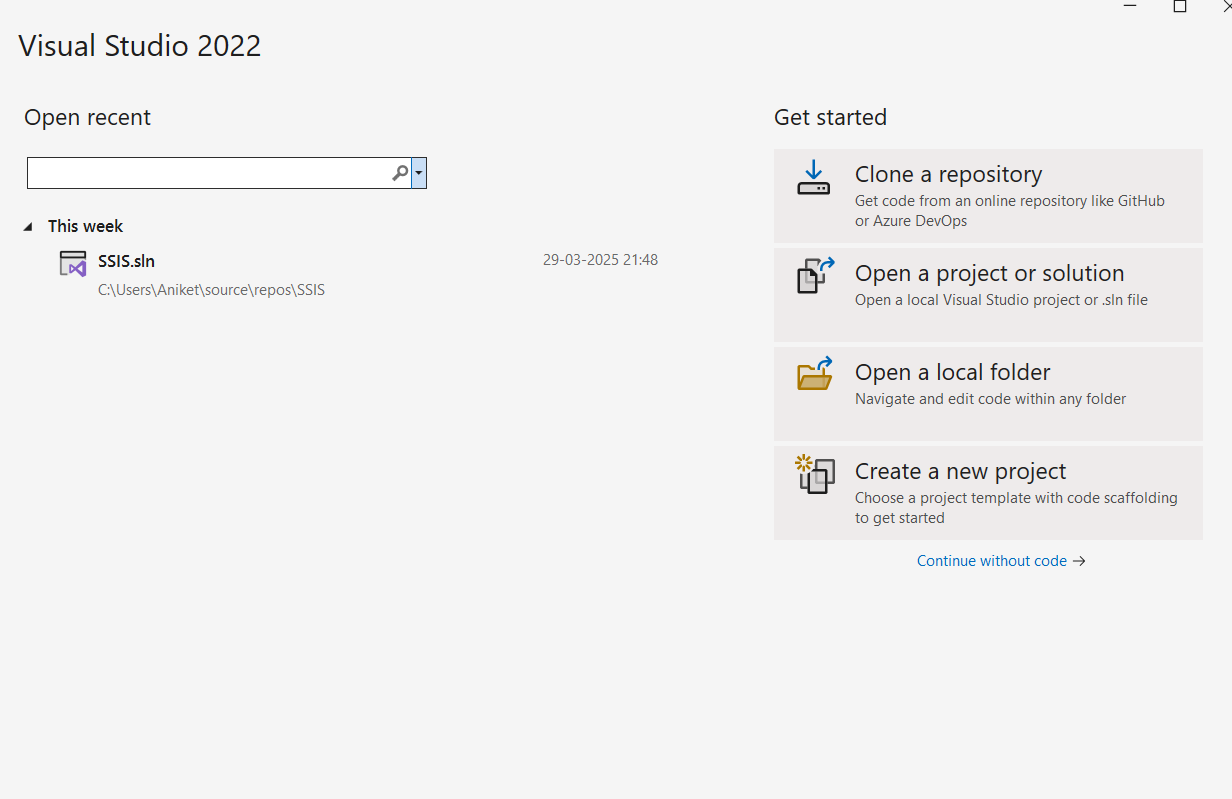
• Objective: Setup SSIS environment and create a simple package.

• Task:

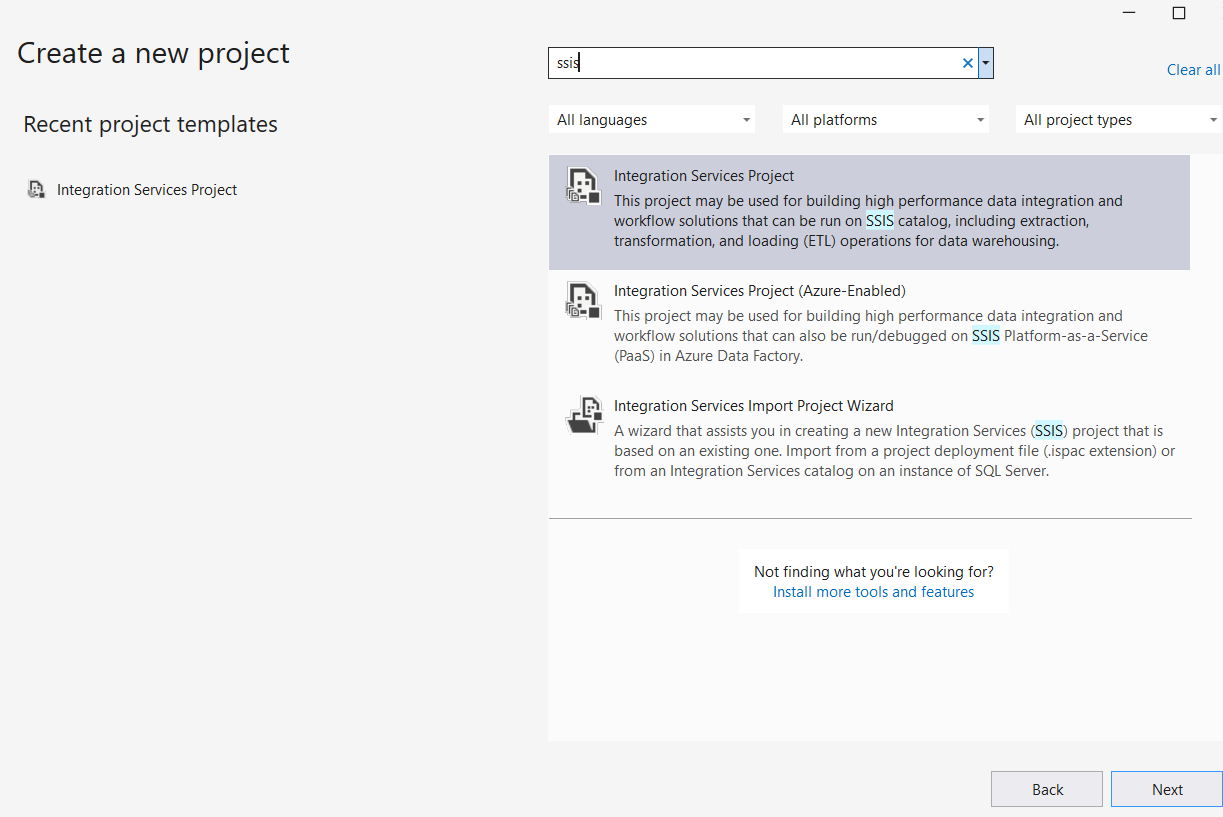
1. Setup:

Install SQL Server Integration Services (SSIS) and SQL Server Management Studio (SSMS).

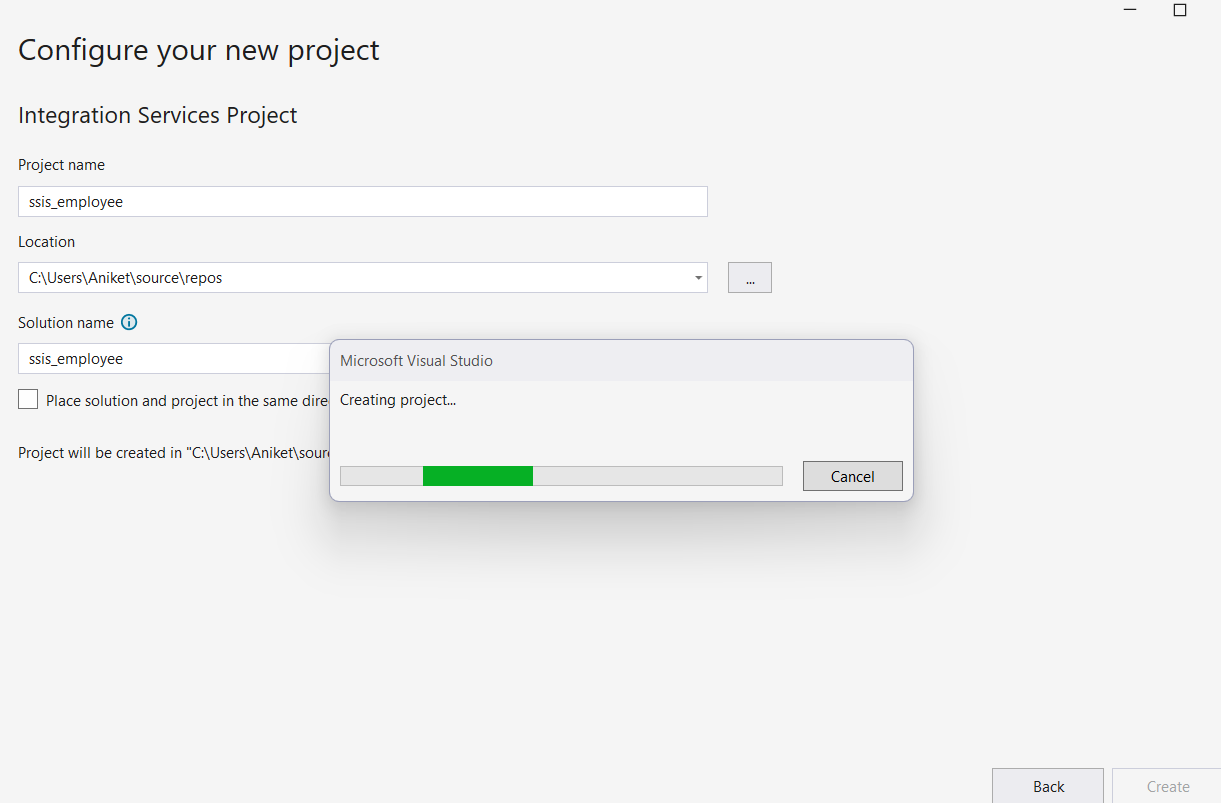
* Download and install sql server and ssms , make sure that while installing server with database engine you always install integration services for ssms.
* Then for the frontend for ssis you need to install visual studio.



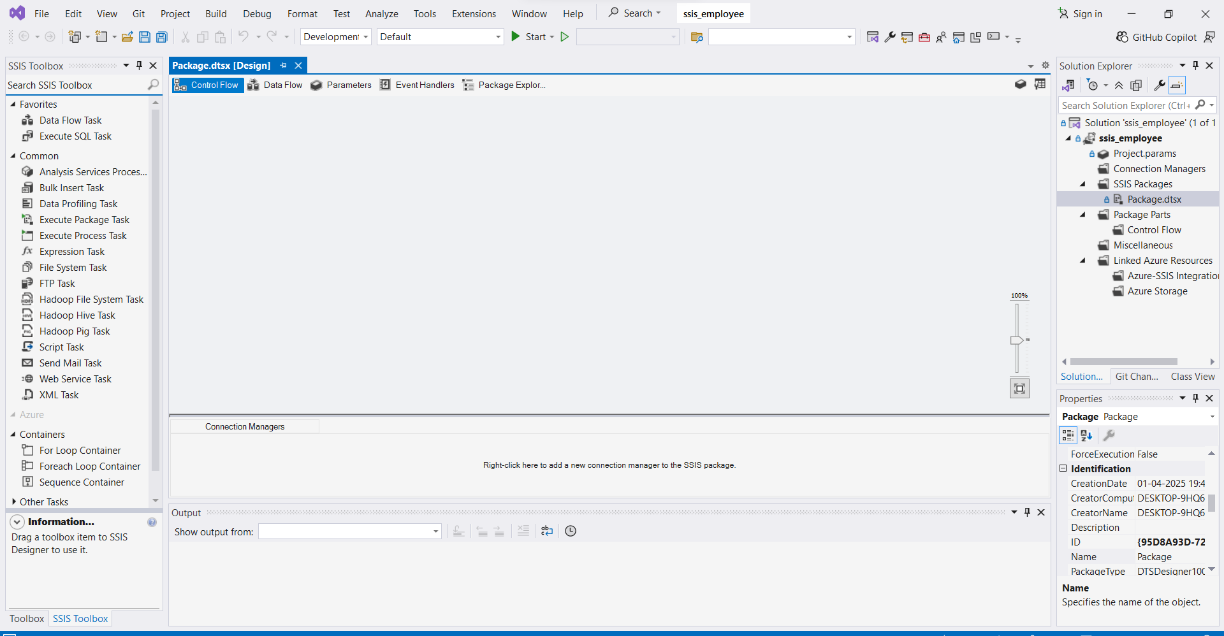
* After opening visual studio download ssis extension for executing dataflow.



* After downloading and installing ssis , create a new project with ssis extension shown above.



* Familiarize yourself with the SSIS Toolbox and Control Flow.



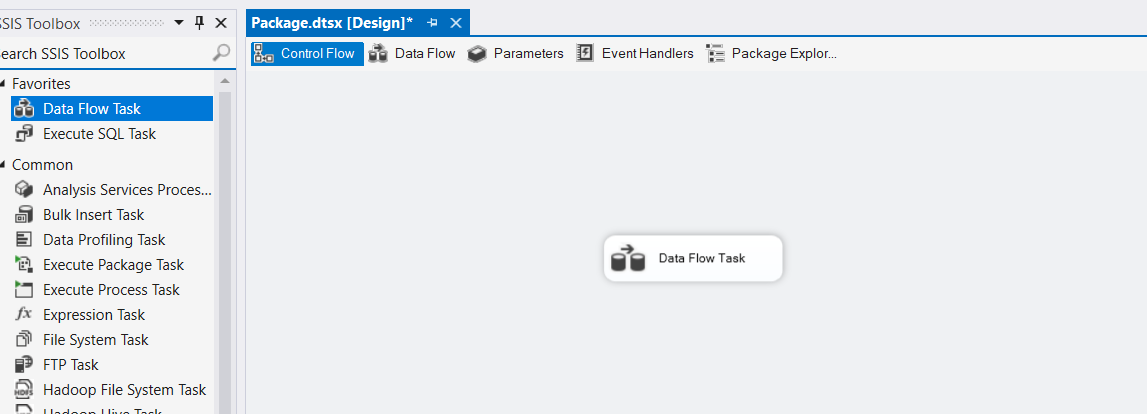
2. Simple Package:

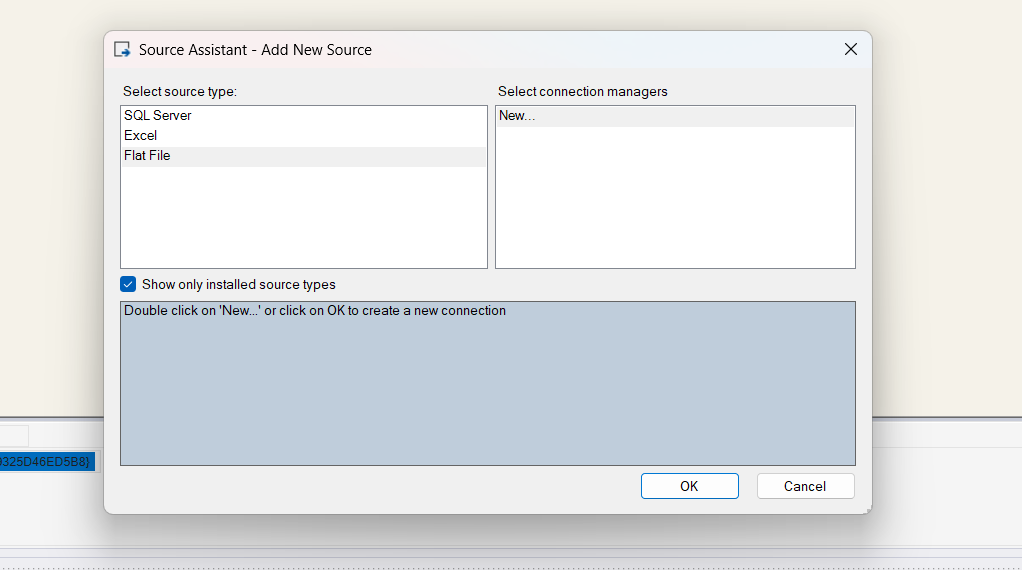
• Task: Create a package that transfers data from a flat file (employees.csv, provided) to a SQL Server table (dbo.Employees).

• Components to Use:

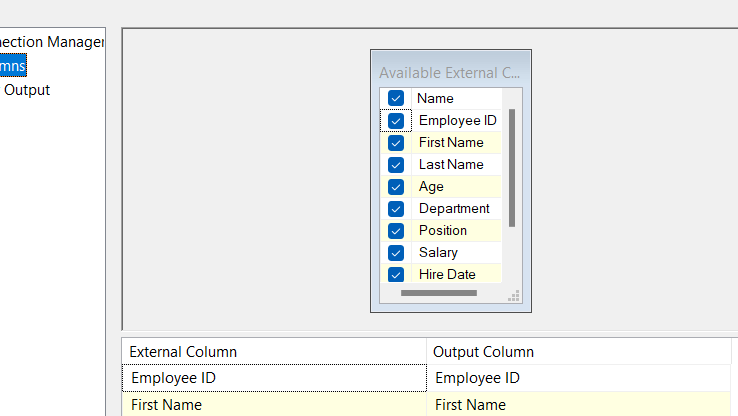
• Flat File Source

* First drag and drop a data flow task
* Double click on the data flow to configure it, on the lefts ide there will be ssis toolbox from which we are going to choose different components.
* After drag and drop source assistant to configure source path.





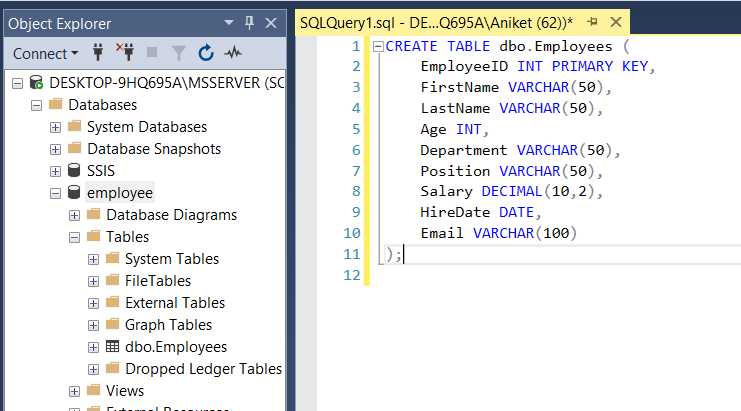
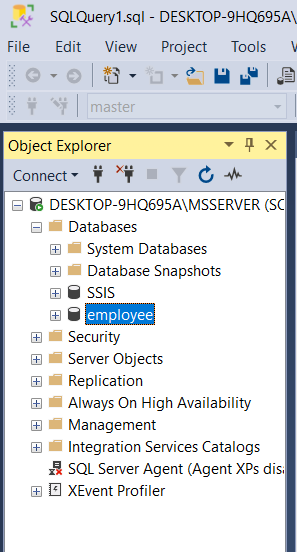
* In this we will choose flat file as a source
* The click on new
* And choose the source for flat file



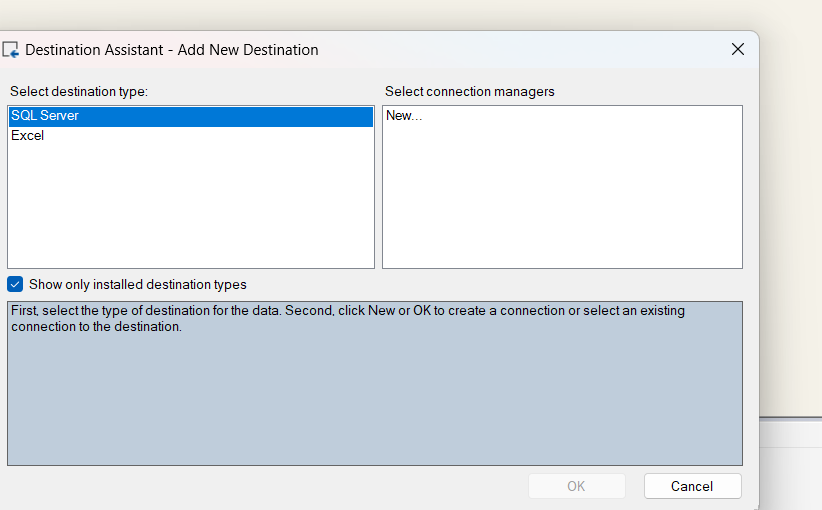
* Then go to columns and select which columns which we will chose for your project and configure it
* After this we have to configure a destination connection

• OLE DB Destination

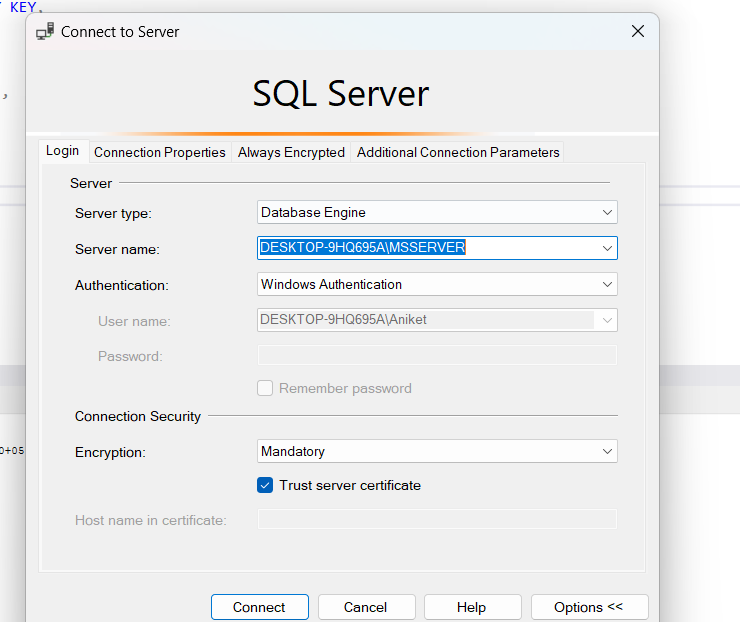
To configure connection, it is necessary to make sure there is database and table present into your SSMS for which you are choosing as a destination.



* Now drag and drop destination assistant from components in dataflow task.



* Choose SQL server and click on new
* Then choose server, database and table for your project





There is still cross coming in destination because we haven't connected task so connect the both task with blue arrow

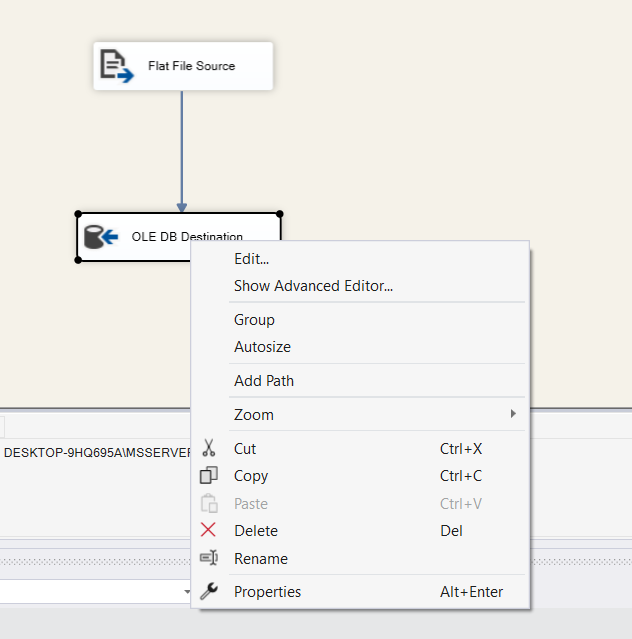
• Data Conversion (if necessary)

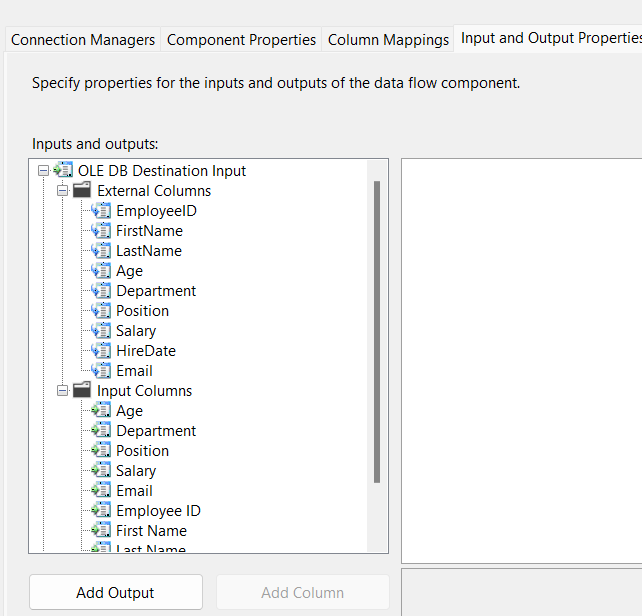
• Constraints:

• Ensure data types match between source and destination.

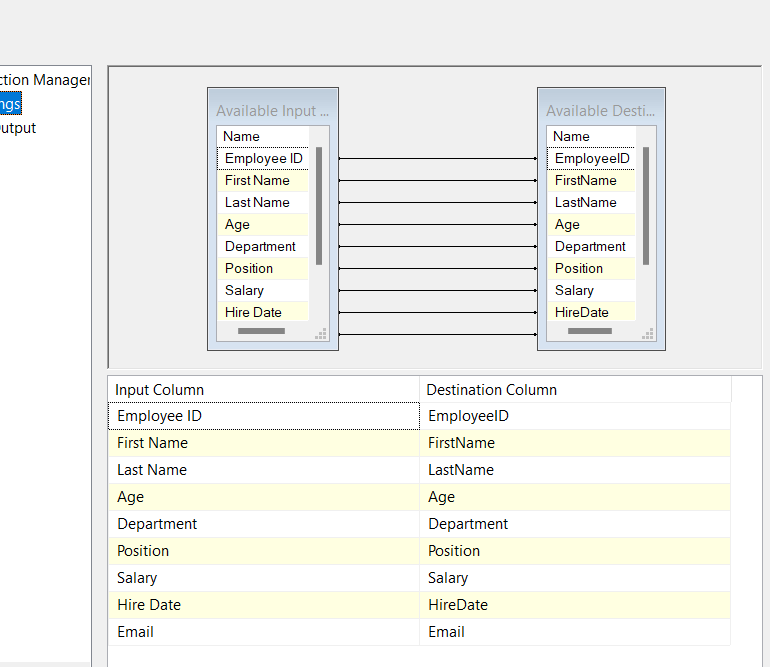
• Handle potential errors during data transfer.

* For checking the datatypes select OLEDB connection and selected advanced properties





Got to input and output properties and check data type for input and external columns, as for this it is same for both, but if not, you can also change by data conversion, derived columns etc.



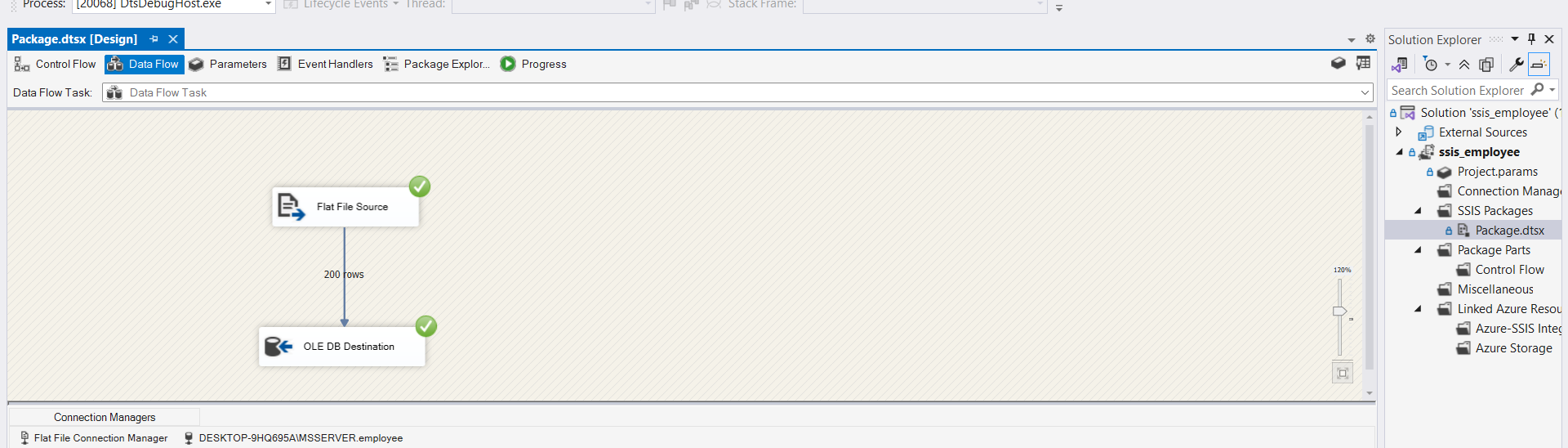
Go to connection manager and got to columns to check for mapping, whether each column is mapped to correct columns, as for this project as you can see input column and out column name have a space between first and last name, so you can do manually map columns.

• Submission:

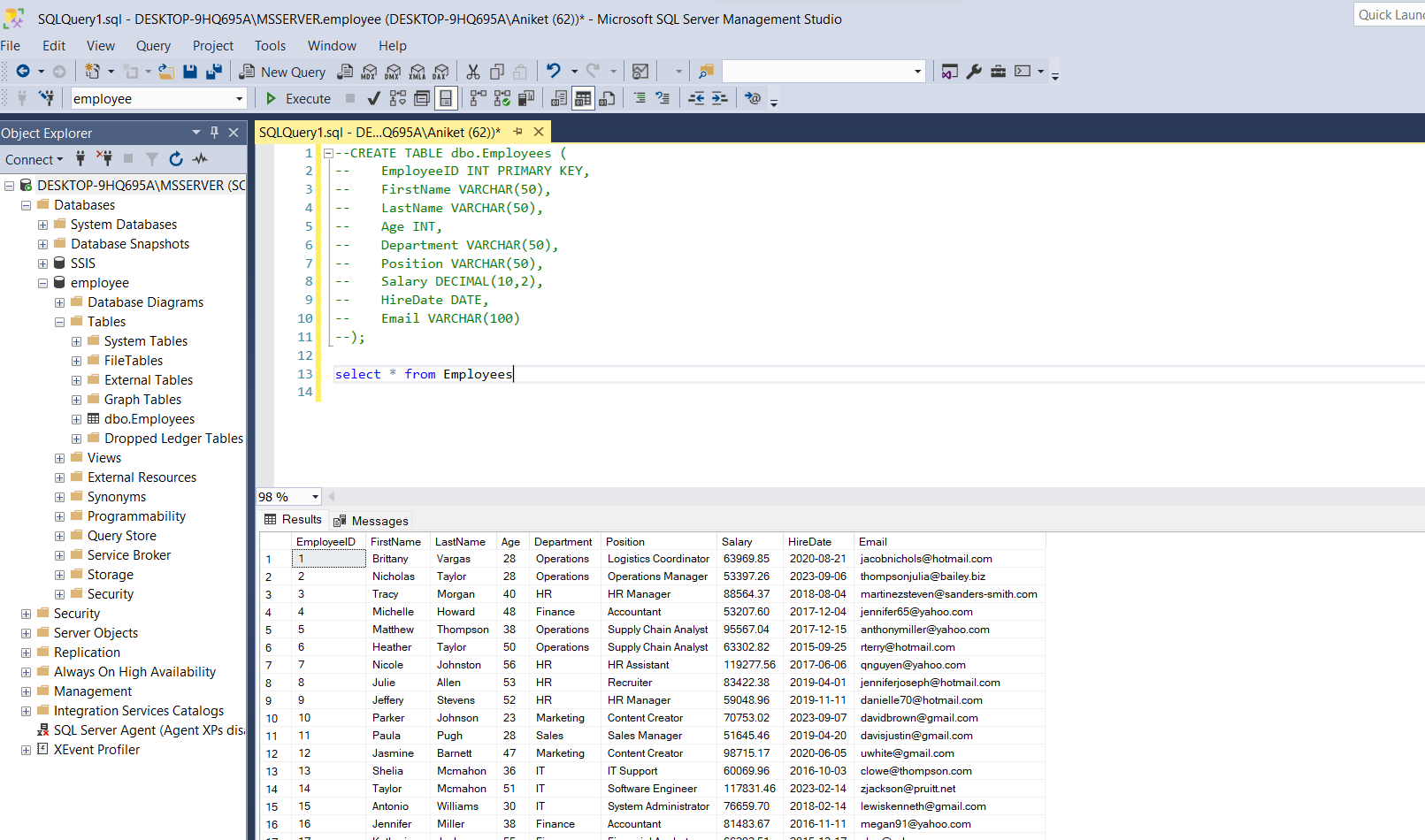
• SSIS package file (BasicPackage.dtsx)

• Screenshots of: Successful package execution dbo.Employees table data after execution

Everything seems good so run the task

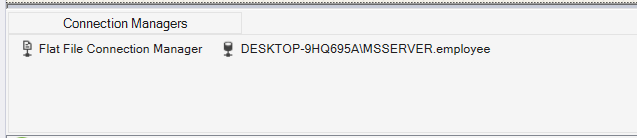


For checking whether destination got updated or not check SSMS



As we can see it is updated. So, task was completed successfully.

The connection manager used for this first task: -



Exercise 2: Data Transformation and Loading

• Objective: Apply transformations and load data into multiple destinations.

• Task:

• Scenario: Enhance the previous package to:

1. Transform:

• Use the Derived Column transformation to create a Full\_Name column.

• Apply the Aggregate transformation to calculate the average Salary by Department.

2. Load:

• Load Full\_Name and original columns into a new table (dbo.Employee\_Extended).

• Output the aggregated data to an Excel file (AverageSalaries.xlsx).

• Components to Use (in addition to previous ones):

• Derived Column Transformation

• Aggregate Transformation

• Excel Destination

• Submission:

• Updated SSIS package file (TransformationPackage.dtsx)

• Screenshots of:

• Successful package execution

• dbo.Employee\_Extended table data

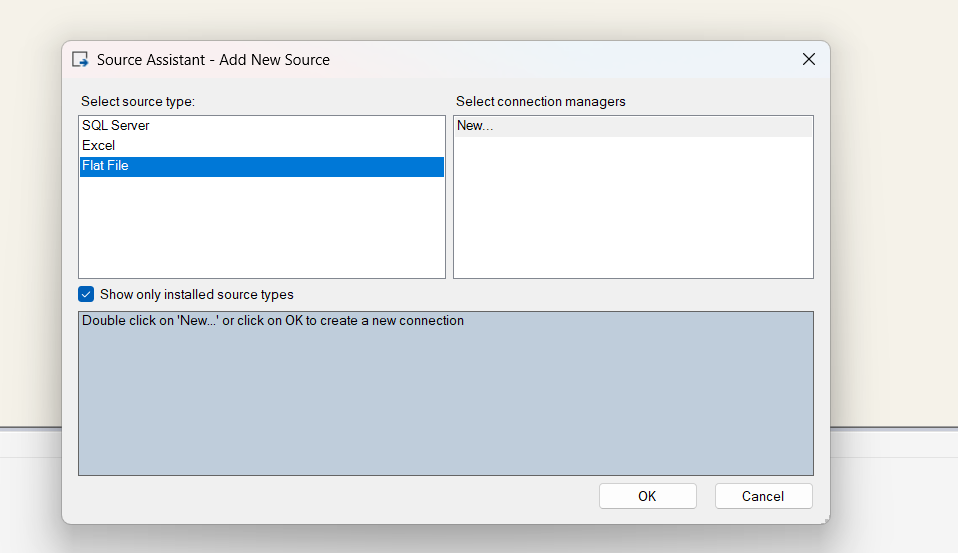
• AverageSalaries.xlsx content

EXPLANATION: -

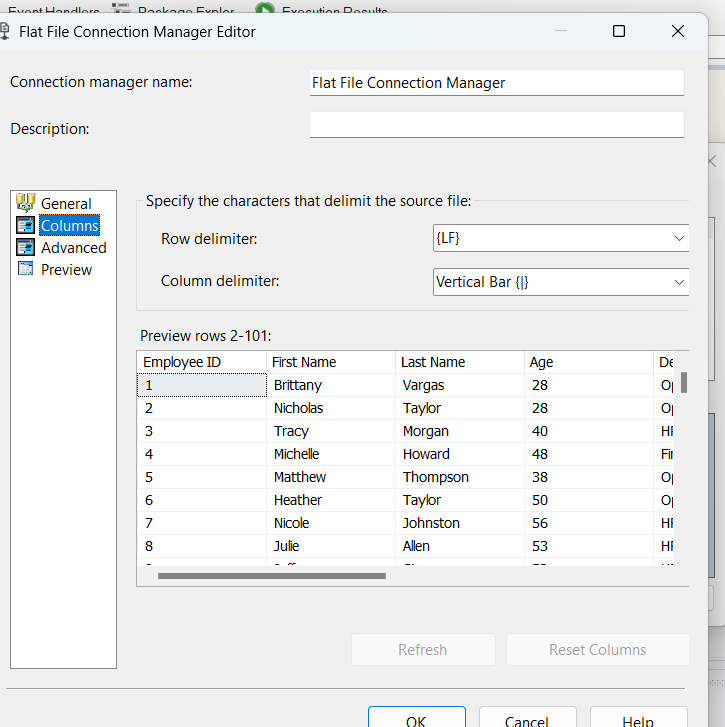
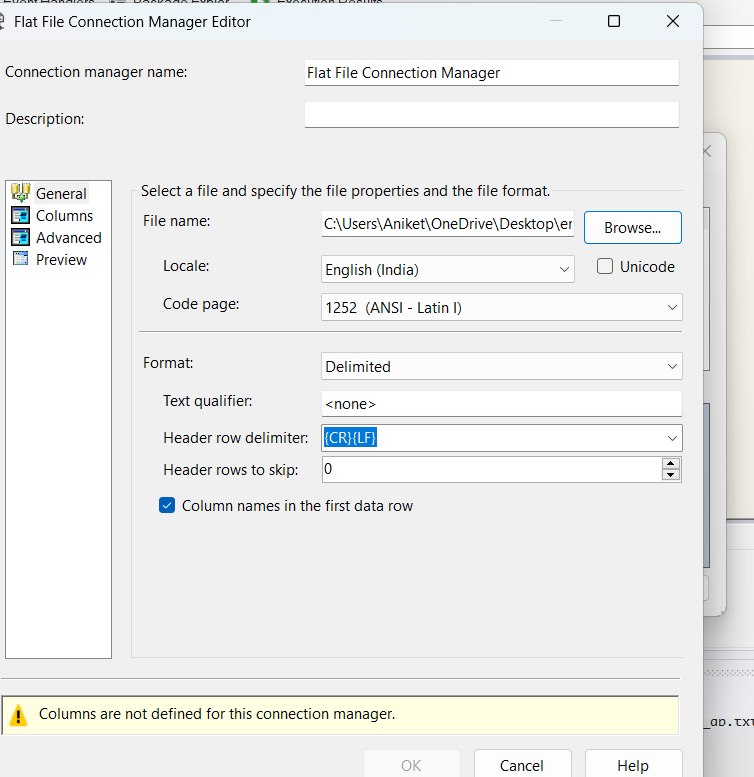
* First task is to create file name transformationPackage.dtsx
* Then drag and drop data flow task



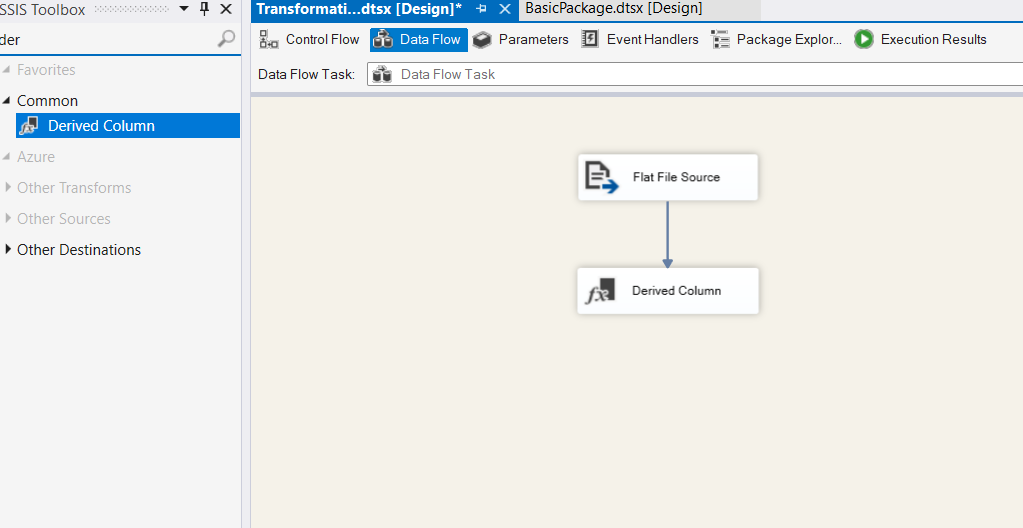
* The choose flat file as a source assistant and add a new flat file connection

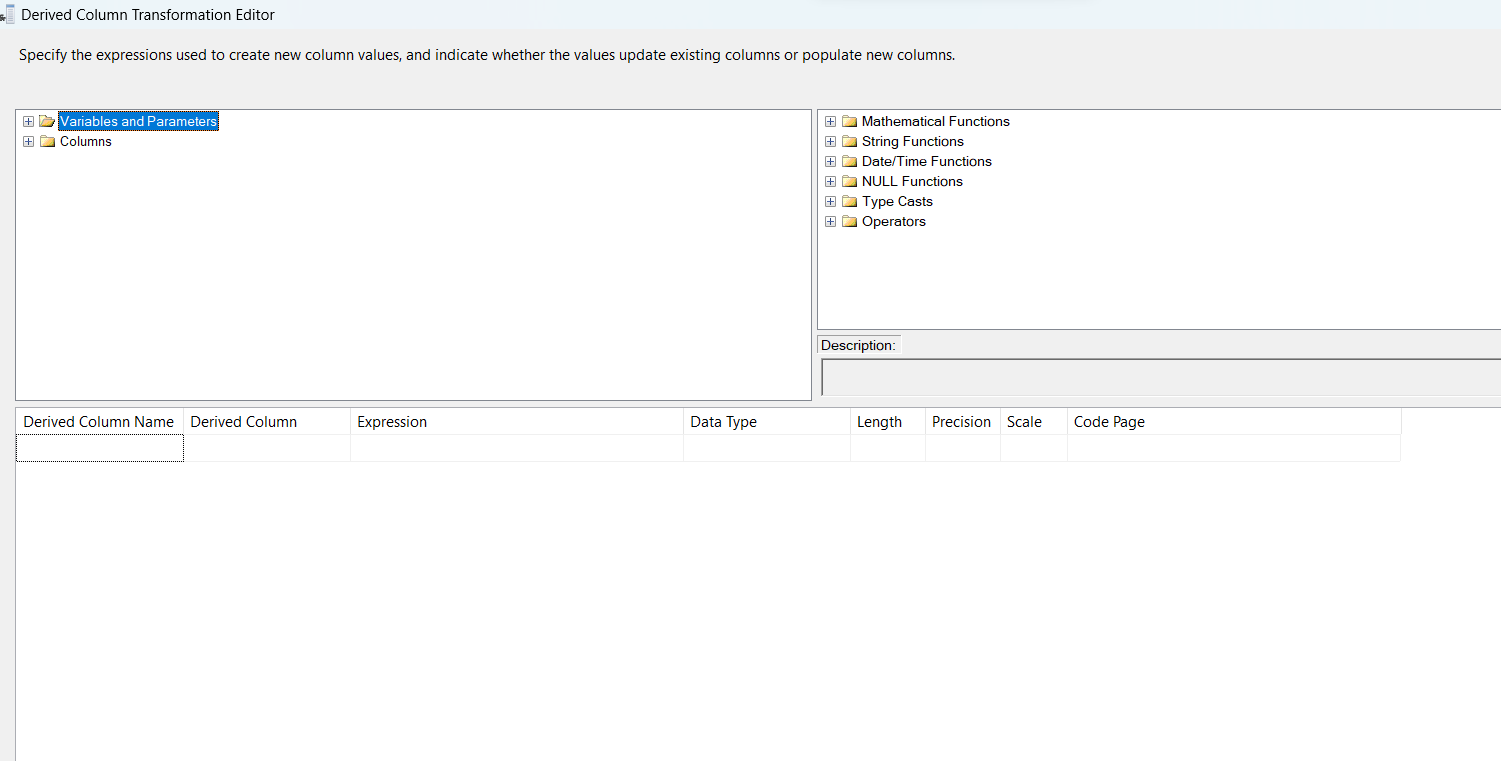


* Choose the location of the flat file and check for columns

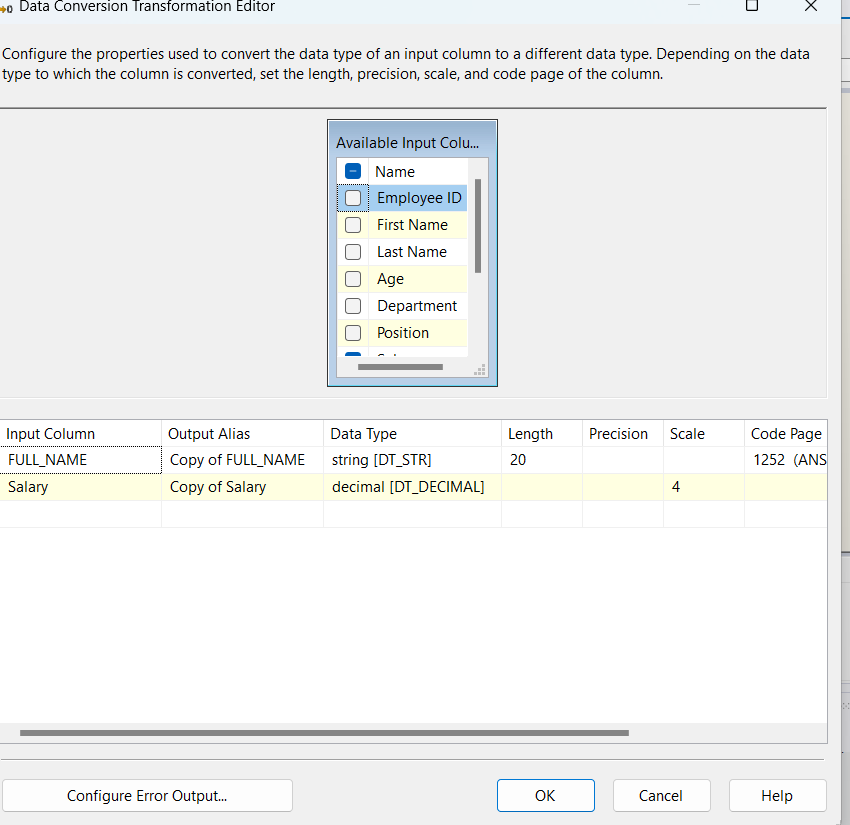


* The drag and drop derived columns to create a full name column and perform the operation

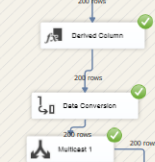




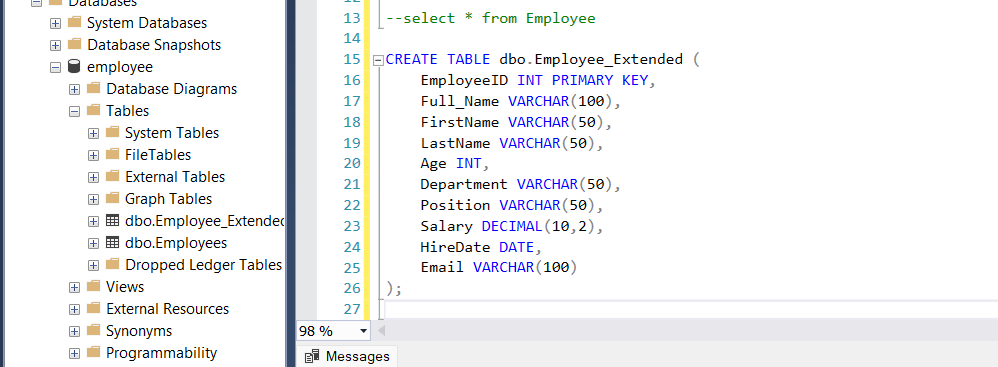
* In this derived column name type – full name and expression as [FIRSTNAME] +[LASTNAME]
* Then we have to do some data conversion



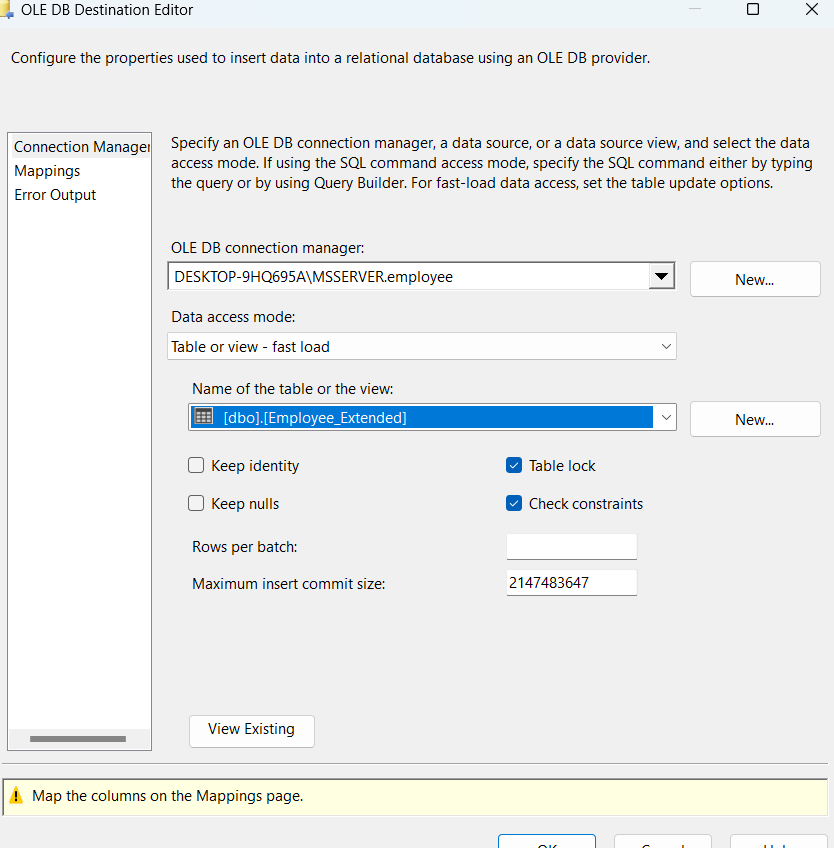
* Convert full name to string and salary to decimal to perform mathematical computation on them.
* Now we have to do task
* First is to create a OLEDB connection to put full name with the rest of the columns
* And next is to do group salary by department and put it in excel file for this we can choose to connect multicast with data conversion as we can add multiple components in that, and we don't have to do multiple task same.



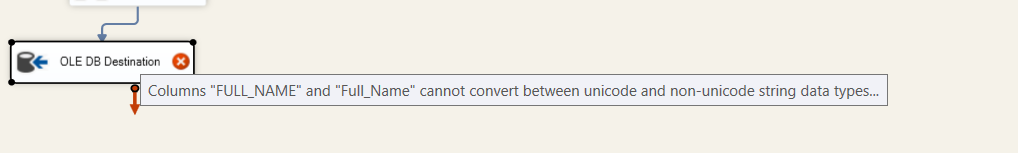
* In this now first task is to create a OLEDB connection to put full name with the rest f the details
* For that first create table name employee\_extended in SSMS to put details there.

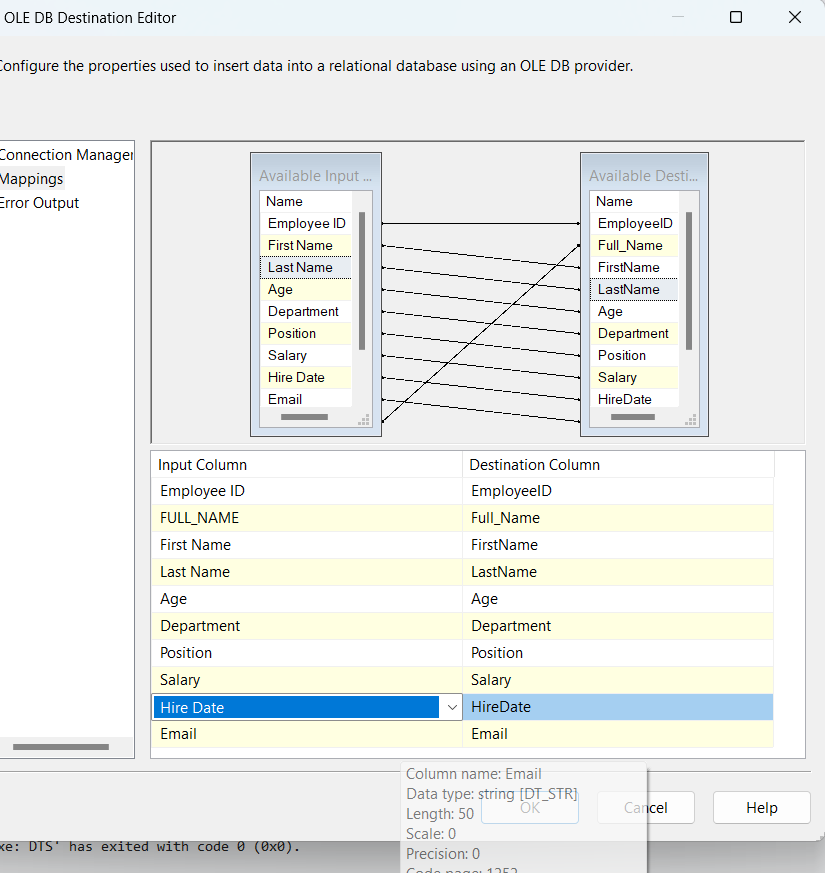


* Create a OLEDB connection, choose server, database, table

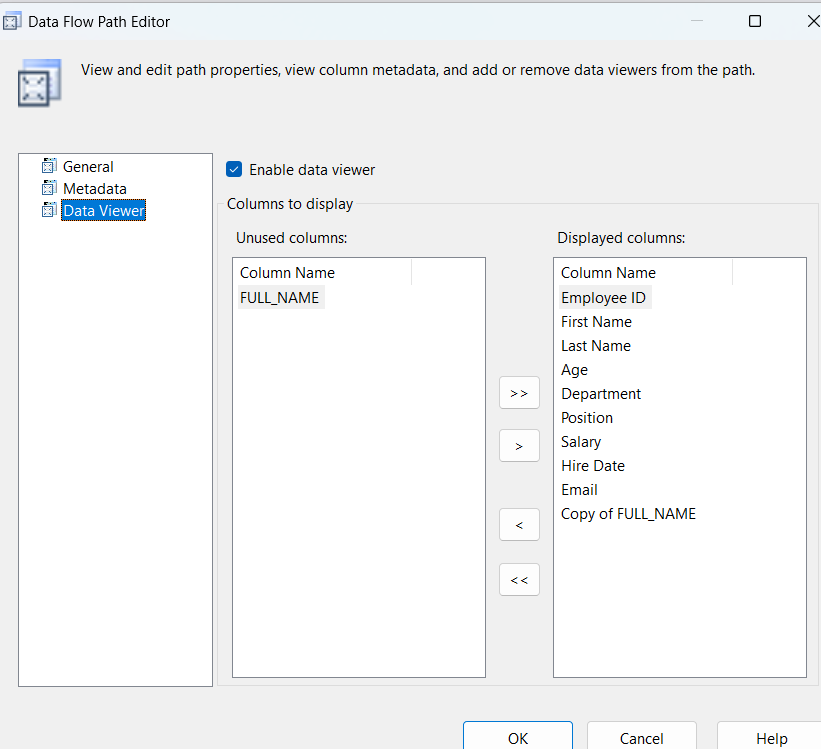


* Go to mappings make sure that you map copy of full name to full name in SSMS schema, because that is name of new constraint after converting into correct datatype for destination, otherwise it will throw error.
* In case of error

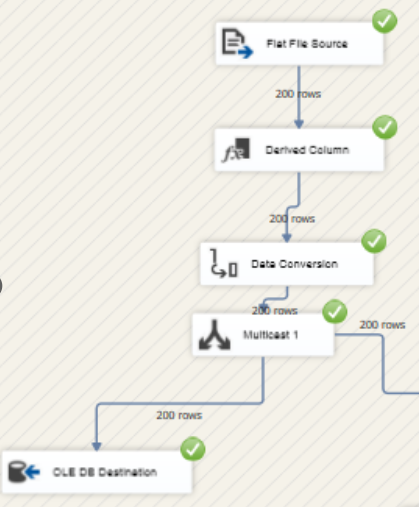
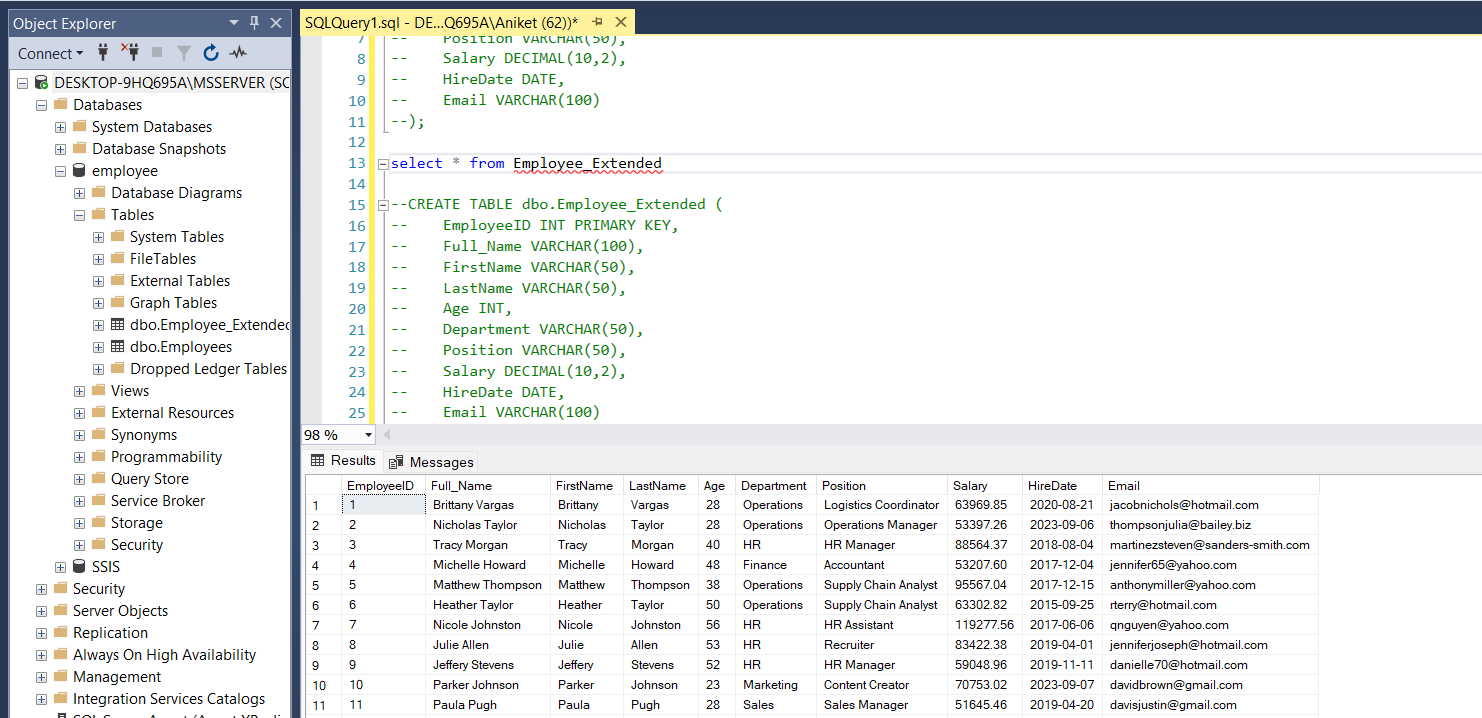




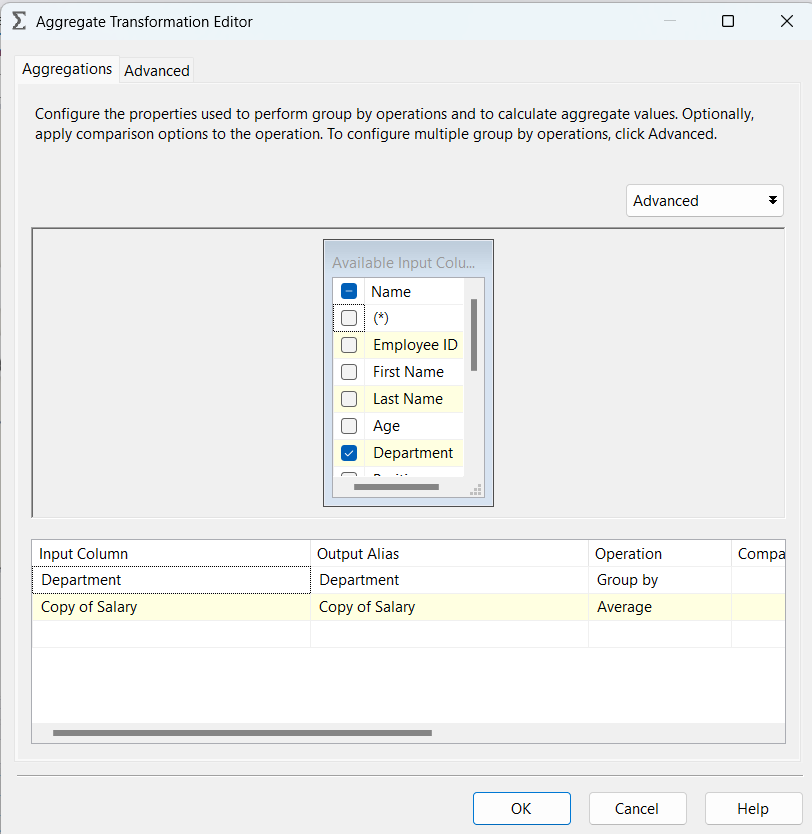
* Now connect the multicast and OLEDB connection and open data flow path editor by clicking on arrow.
* In this we have to choose what columns will be going to put in SSMS.
* We will put full name to unused columns as to not make double copies of the same thing and to not face conversion error.



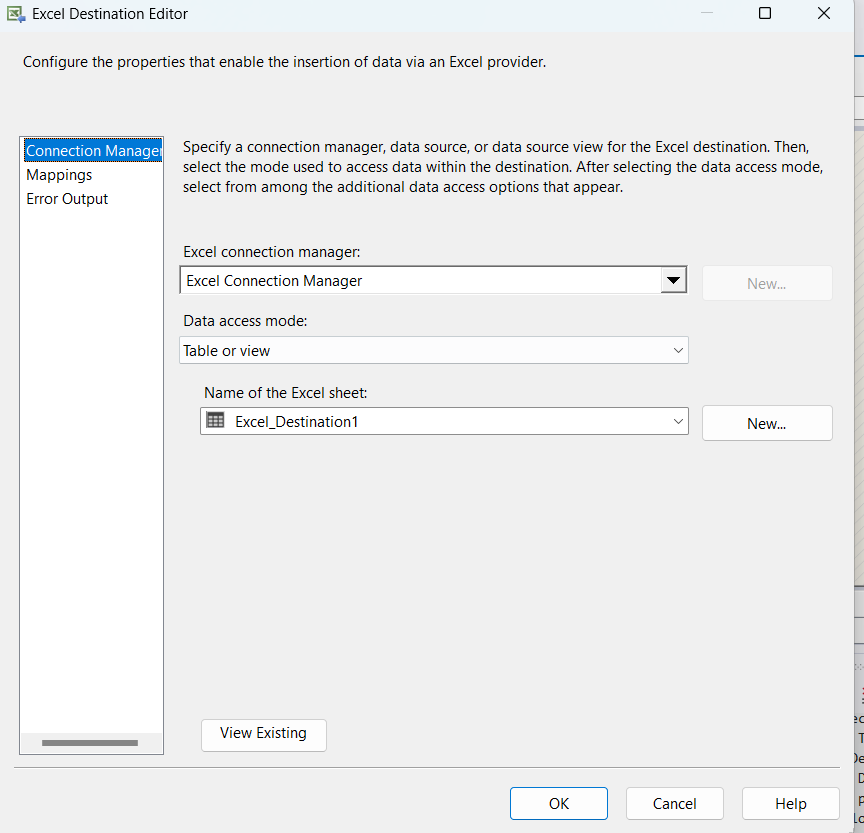
* We can run the first half f task to see if it is working or not



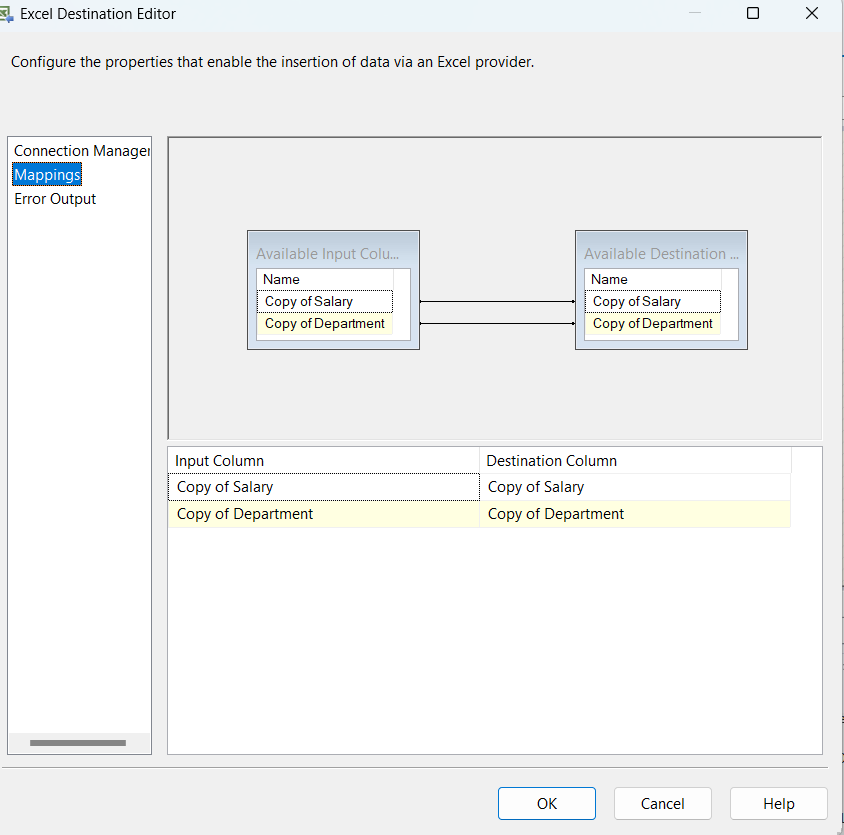
* As we can see the first task is completed successfully so now, we have to group salary
* For that we can drag and drop aggregate component in dataflow and connect with multicast



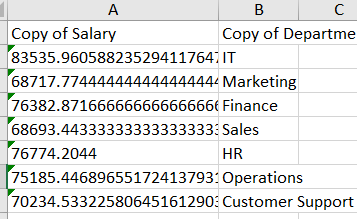
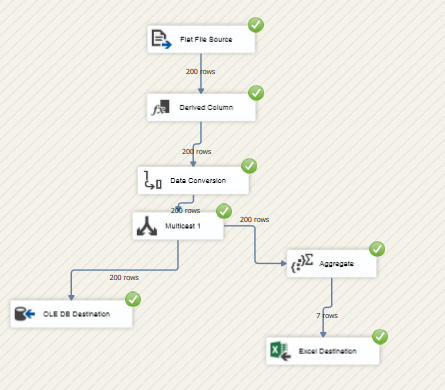
* In this choose department and select group by as operation and second salary as average
* Now as aggregation is done, we have to make excel file destination



* Choose the connection and select name of excel, if not you can also create aby clicking on new.
* Then open editor of excel to map and make sure you map correct columns with each other.



* Now we can run the whole package



* As we can see all the task are successfully completed and also excel file is also creating.

Exercise 3: Advanced SSIS - Parameters, Variables, and Error Handling

• Objective: Implement parameterization, variable usage, and robust error handling.

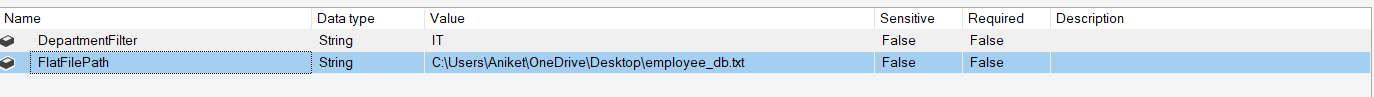
• Task:

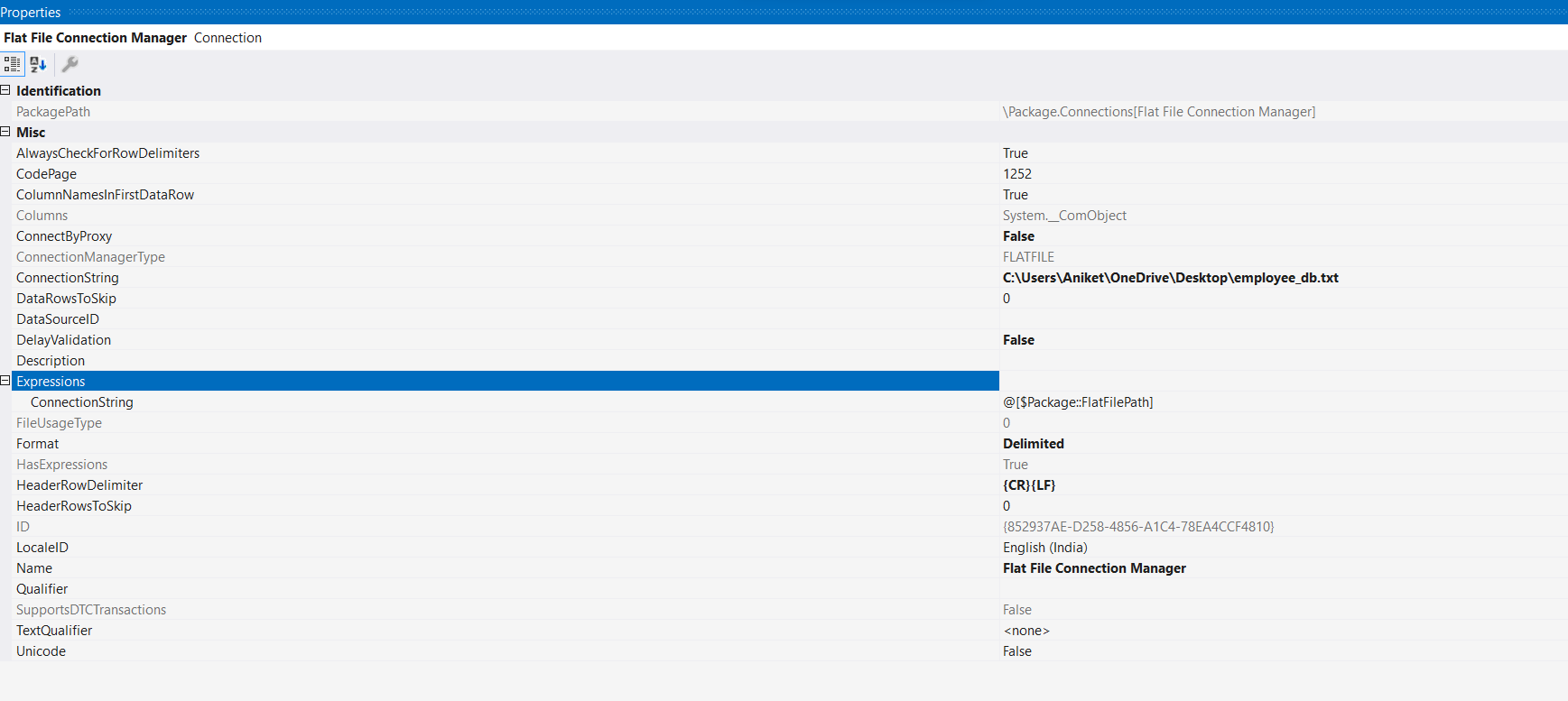
• Enhancements:

1. Parameterize:

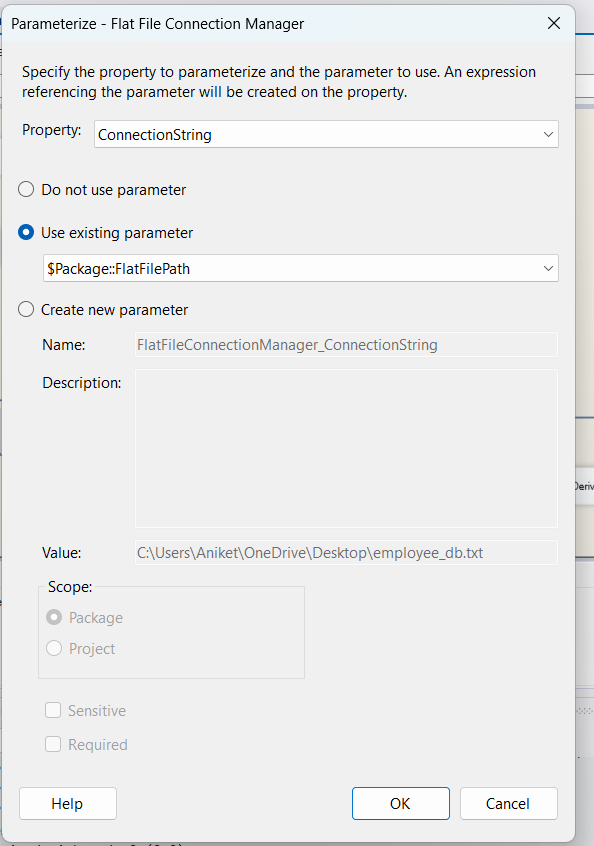
* Convert the flat file source path to a package parameter.

1. Click **Add Parameter** and set:
2. **Name**: FlatFilePath
3. **Data Type**: String
4. **Value**: C:\SSIS\Input\employees.csv *(Set the default path)*
5. Open **Flat File Connection Manager** → Select **Expressions**.
6. Set **ConnectionString** to use the parameter:
7. Click **Expressions** → Choose **ConnectionString** → Set to @[$Package::FlatFilePath]



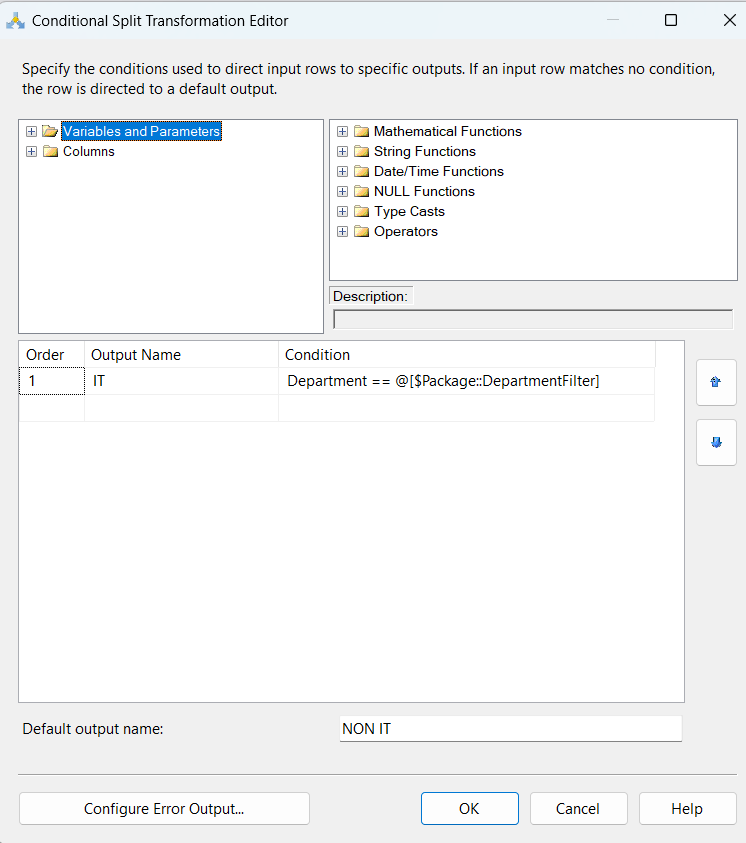
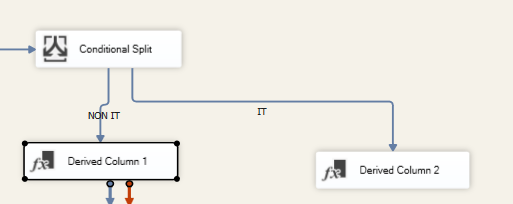


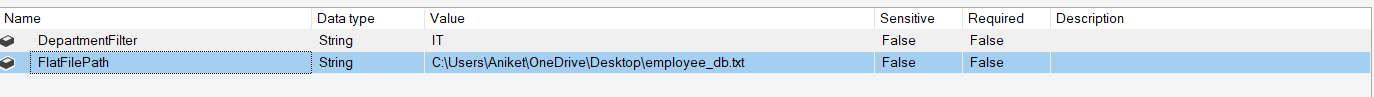
Do this in parameterize



* Pass the Department filter value as an execution parameter.

1. Click **Add Parameter** and set:
2. **Name**: DepartmentFilter
3. **Data Type**: String
4. **Value**: "IT"
5. Open **Conditional Split Transformation** in Data Flow.



2. Variables:

* Use a variable to dynamically set the Excel output file name based on

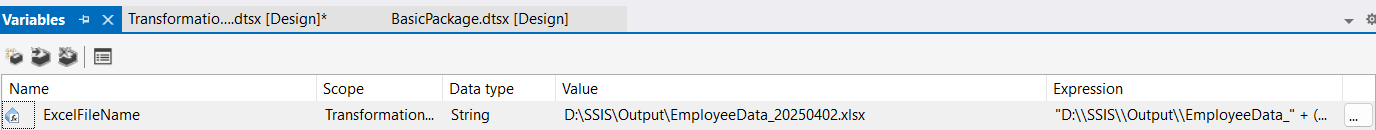
the execution date.

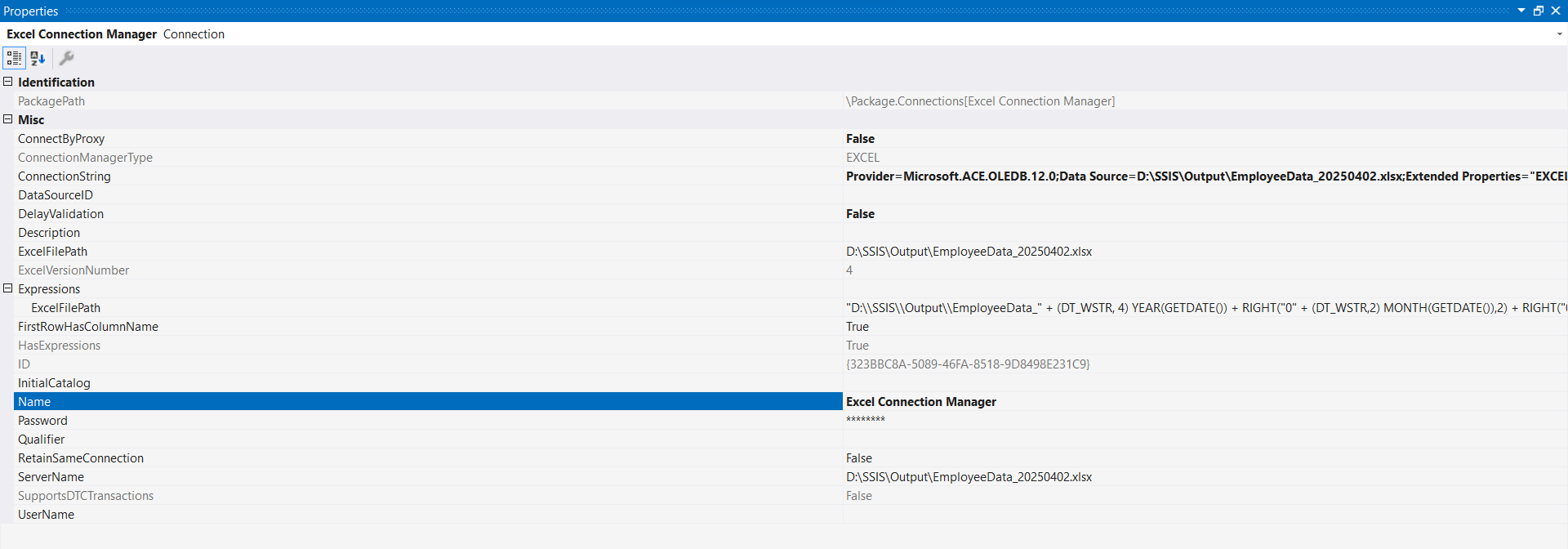
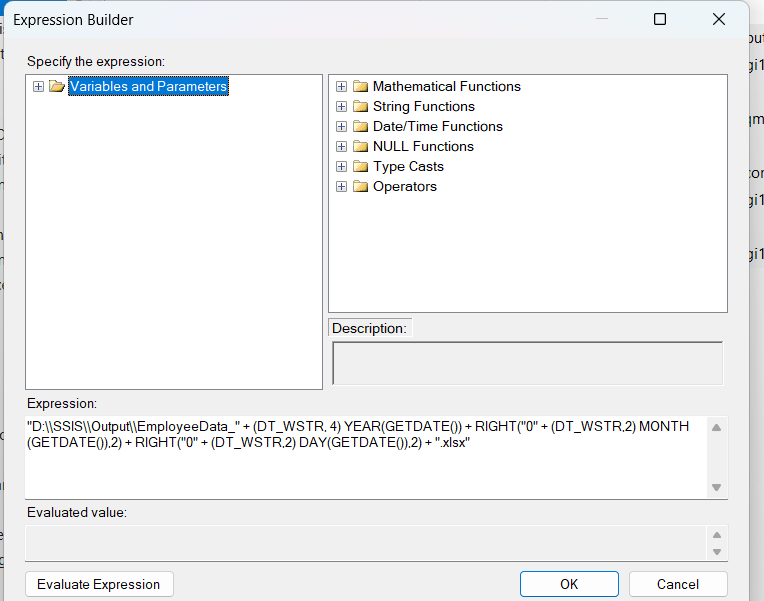
1. Click **Add Variable** and set:
2. **Name**: ExcelFileName
3. **Data Type**: String
4. **Value**: Leave blank
5. Open **Expressions** for **Excel Connection Manager**.
6. Set **ExcelFilePath** expression:

"C:\\SSIS\\Output\\EmployeeData\_" + (DT\_WSTR, 10) YEAR(GETDATE()) +

RIGHT("0" + (DT\_WSTR,2) MONTH(GETDATE()),2) +

RIGHT("0" + (DT\_WSTR,2) DAY(GETDATE()),2) + ".xlsx"

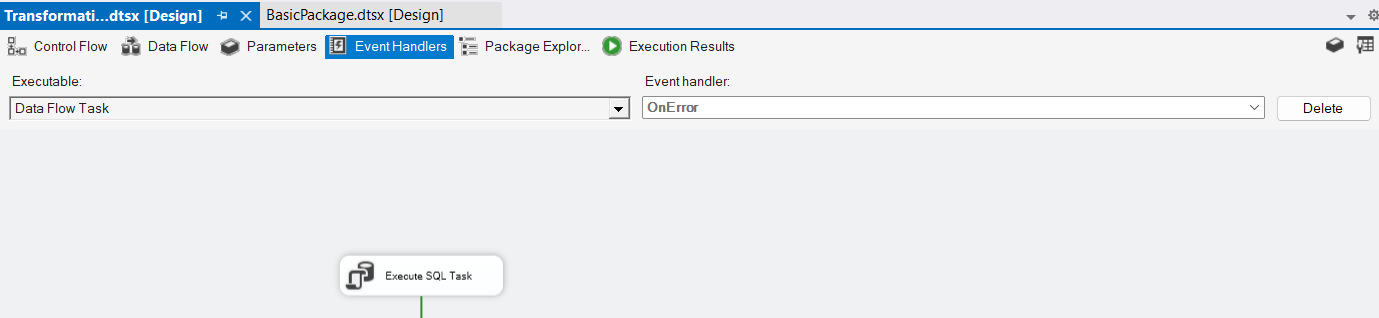




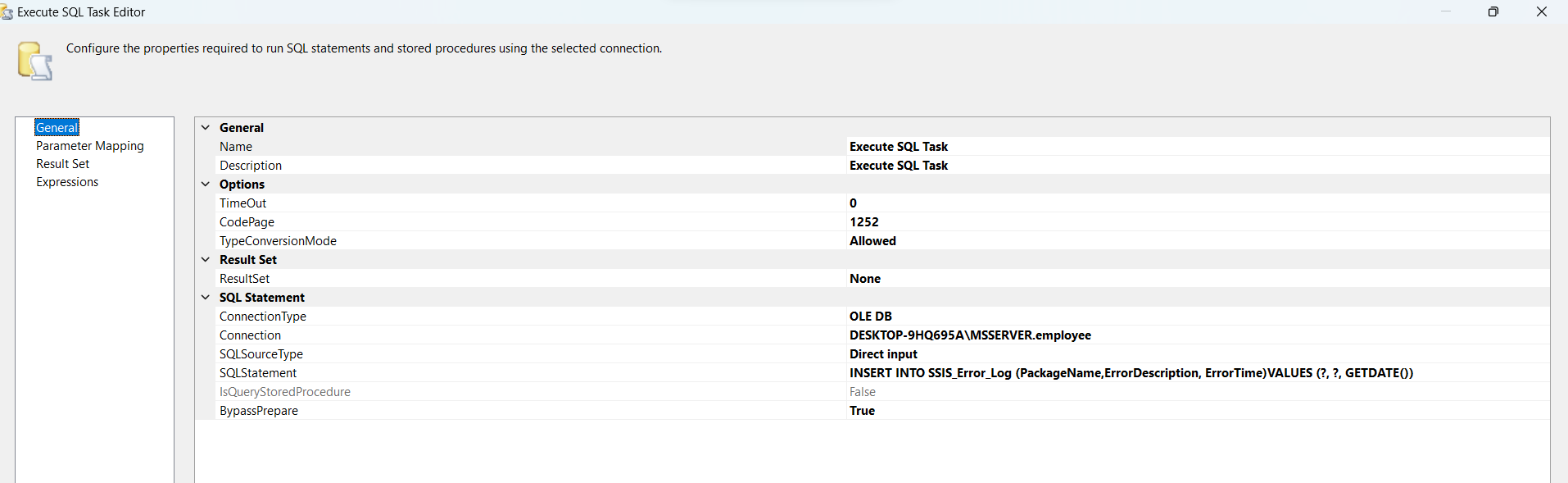
3. Error Handling:

* Implement try-catch blocks in the Control Flow to handle and log errors.

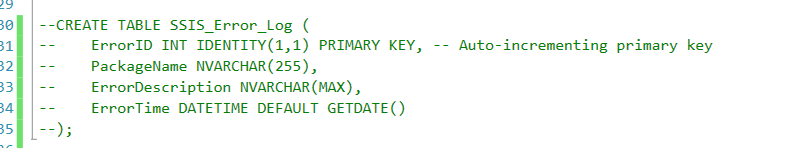
For first error handling go to event handler and drag drop execute sql task and set executable to data flow task and event handler to on error



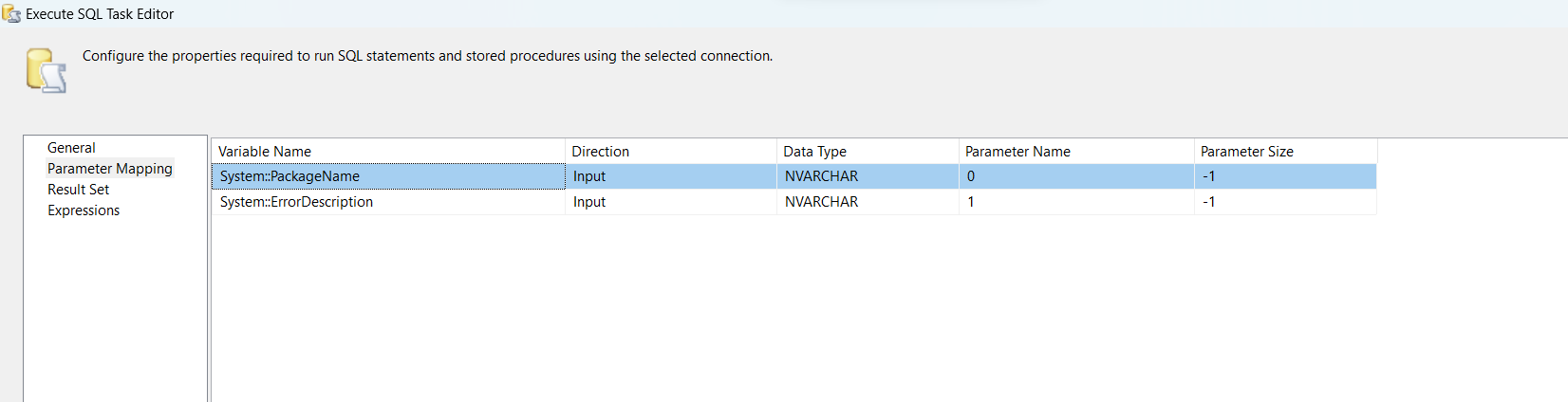
Then double click on execute task



In this configure connection and write sql statement for which details yu want to log into your database ex- package name, error description, error time. It is also important to note that before inserting make sure that there is a table name ssis\_error\_log in your database.



Then go to parameter mapping

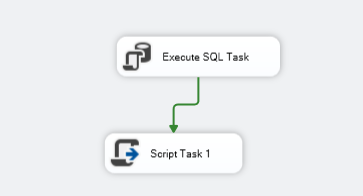


As you can see that first fetch package name and error description from system and set datatype correctly and make sure while entering parameter name that it is the indexing in whch manner the variable is onna insert in the database. As example package name is first so zero and error description is second so first acc to indexing and error time will automatically be fetched by datetime function.

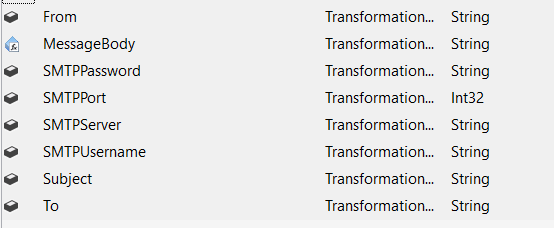
* Ensure failed packages send a notification email (use Send Mail Task).

Now for the email verification you caan do by send mail task , but for that as credentials cannot be authenticated use script task for sending mail.

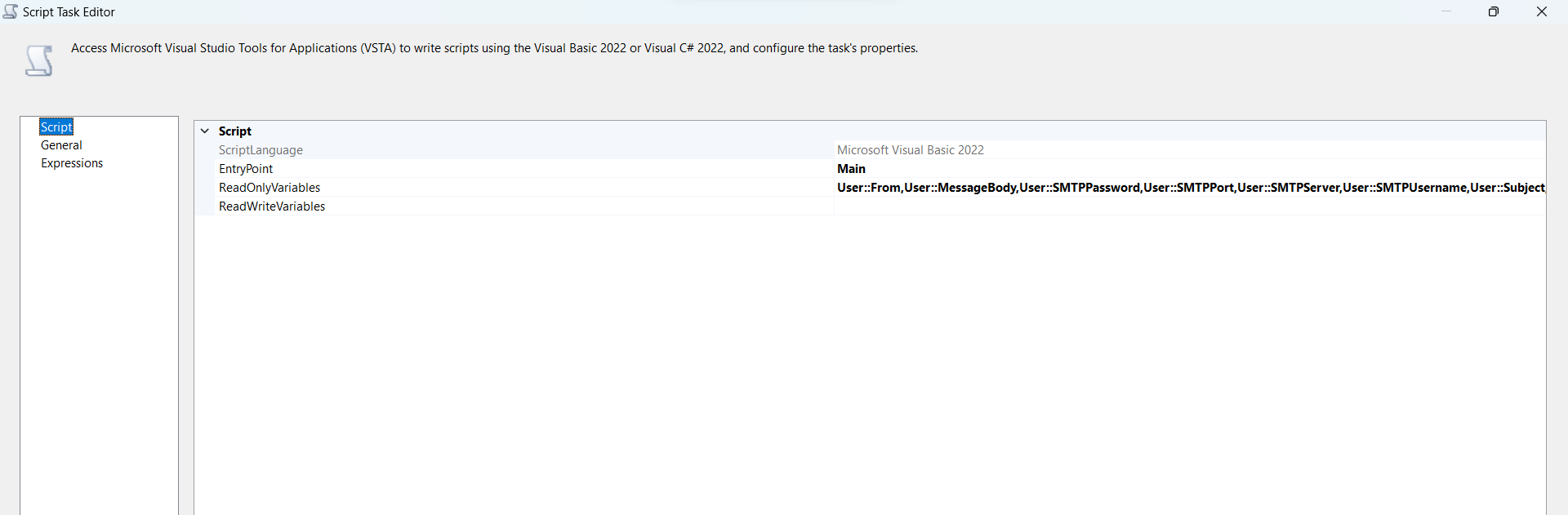
So first drag and drop script task.



For this first declare few variabales as to to this



And also make sure to this first go to your gmail, do two step verification, add app address and give name save password, as to make smtp gamil server , you can also check online whether server is working or not, then save the variable with value.



Then select script language as visual studio and select the read only variables that you just declared for sending email.

The open edit script

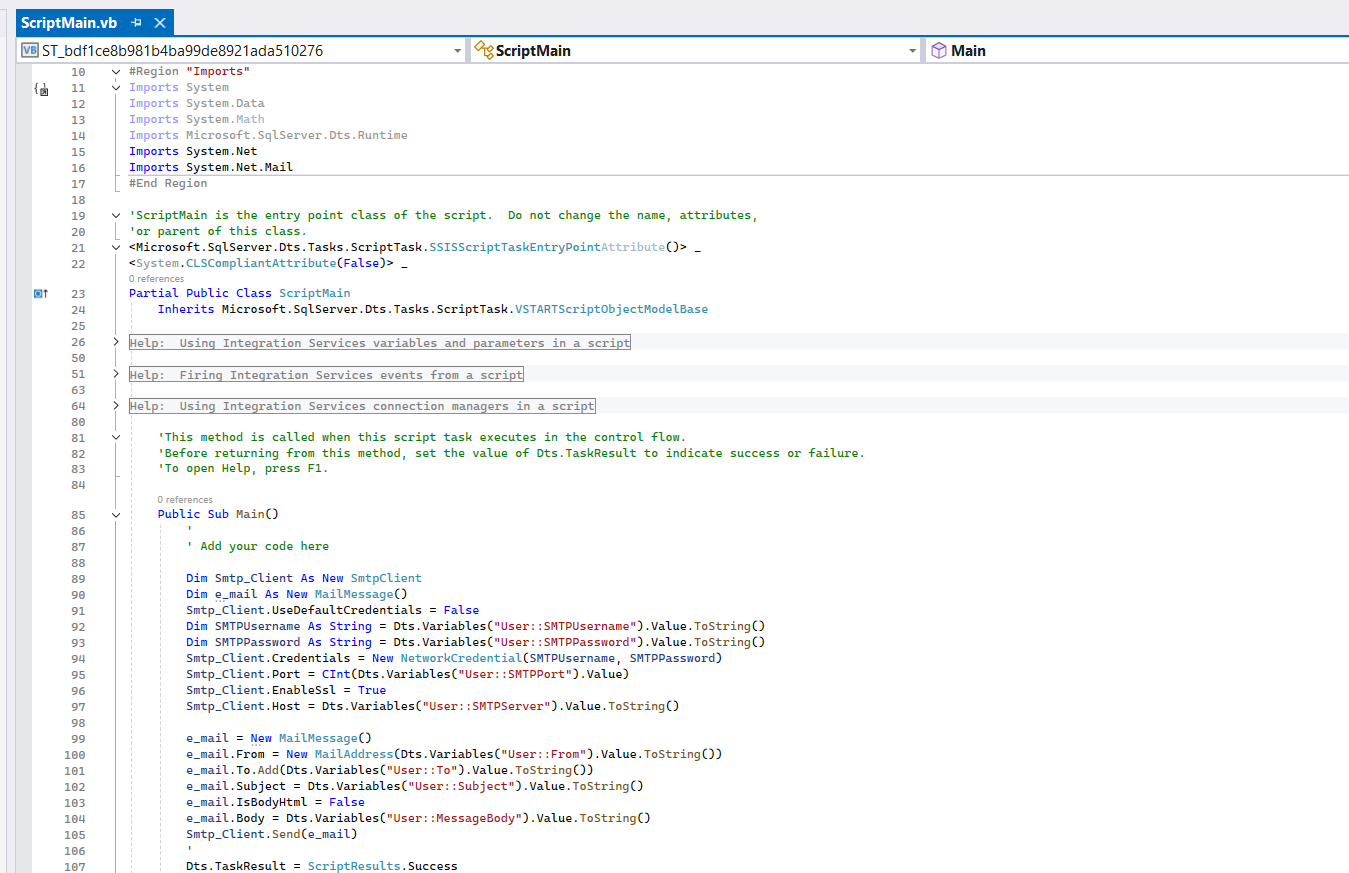
And add first import

1. Imports System.Net
2. Imports System.Net.Mail

Then in addind code section add

|  |
| --- |
| Dim Smtp\_Client As New SmtpClient  Dim e\_mail As New MailMessage()  Smtp\_Client.UseDefaultCredentials = False  Dim SMTPUsername As String = Dts.Variables("User::SMTPUsername").Value.ToString()  Dim SMTPPassword As String = Dts.Variables("User::SMTPPassword").Value.ToString() Smtp\_Client.Credentials = New NetworkCredential(SMTPUsername, SMTPPassword) Smtp\_Client.Port = CInt(Dts.Variables("User::SMTPPort").Value)  Smtp\_Client.EnableSsl = True Smtp\_Client.Host = Dts.Variables("User::SMTPServer").Value.ToString()  e\_mail = New MailMessage() e\_mail.From = New MailAddress(Dts.Variables("User::From").Value.ToString()) e\_mail.To.Add(Dts.Variables("User::To").Value.ToString())  e\_mail.Subject = Dts.Variables("User::Subject").Value.ToString()  e\_mail.IsBodyHtml = False  e\_mail.Body = Dts.Variables("User::MessageBody").Value.ToString()  Smtp\_Client.Send(e\_mail) |

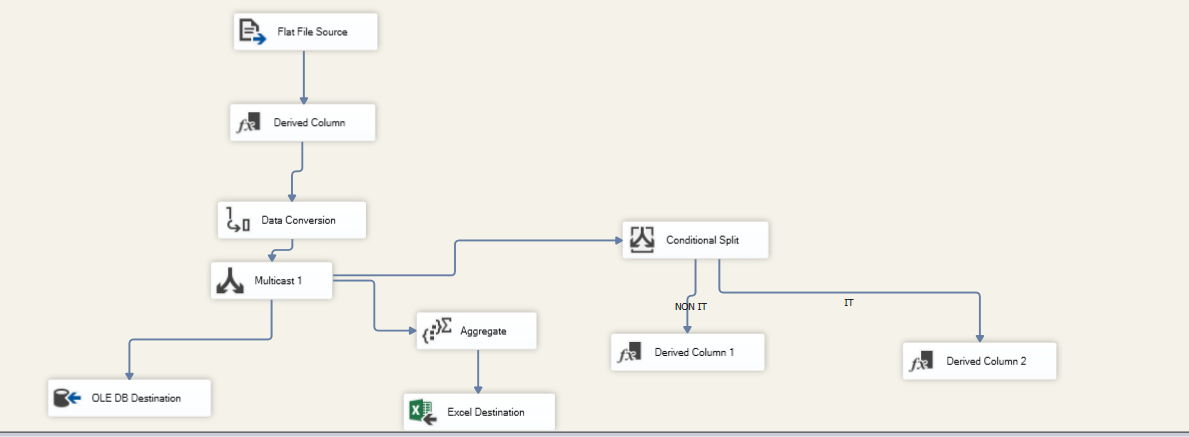
Make sure variable name is corretly typed



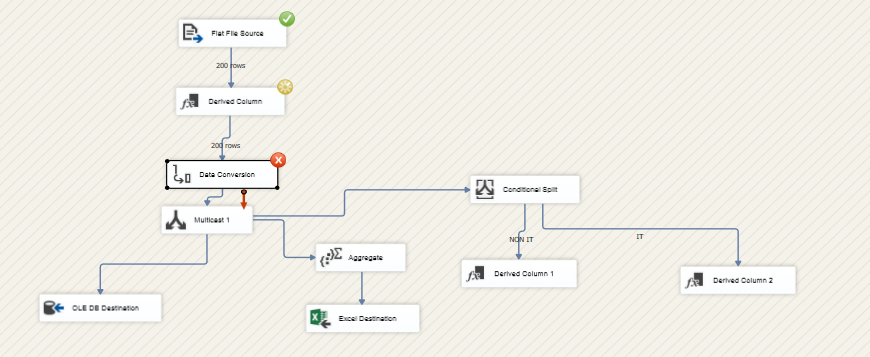
Submission:

* Updated SSIS package file (AdvancedPackage.dtsx)
* Screenshots of:
  + Parameter configuration
  + Variable usage in Excel file naming
  + Error handling and email notification setup

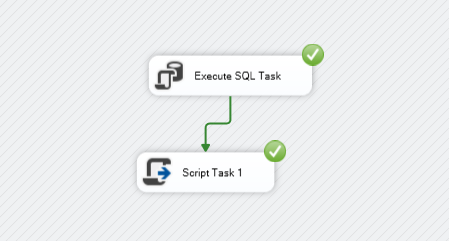
Now check whether whether error and email is working or not so

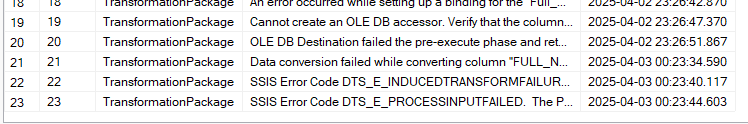


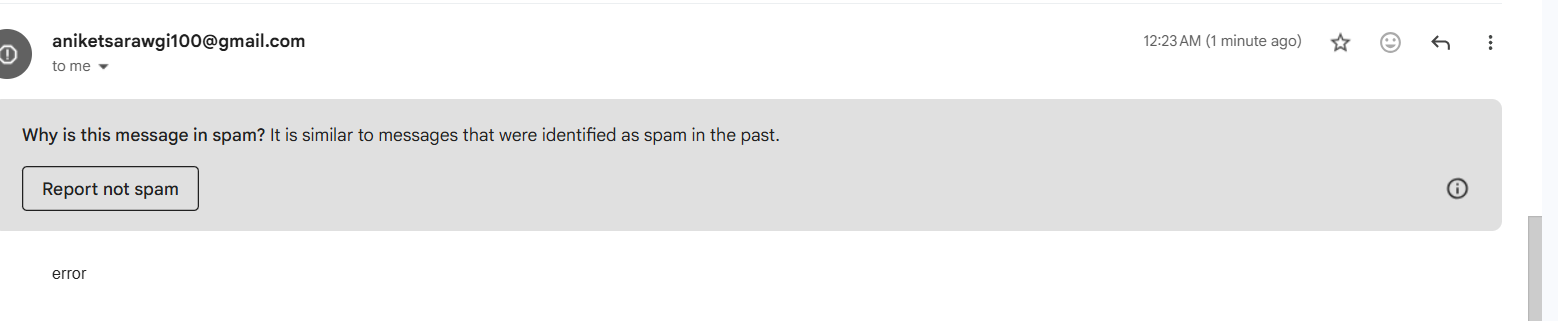
In data conversion convert some wrong data type to send error and email and run.



Error was made and both task error and mail completed succesfully







Exercise 4: Deployment and Scheduling

• Objective: Deploy and schedule an SSIS package.

• Task:

• Deployment:

1. Deploy one of your packages to the SSIS Catalog.

2. Configure environments for different execution contexts (e.g., Dev, Prod).

• Scheduling:

1. Use SQL Server Agent to schedule the deployed package for daily execution.

2. Set up a job to send a success/failure email after package execution.

• Submission:

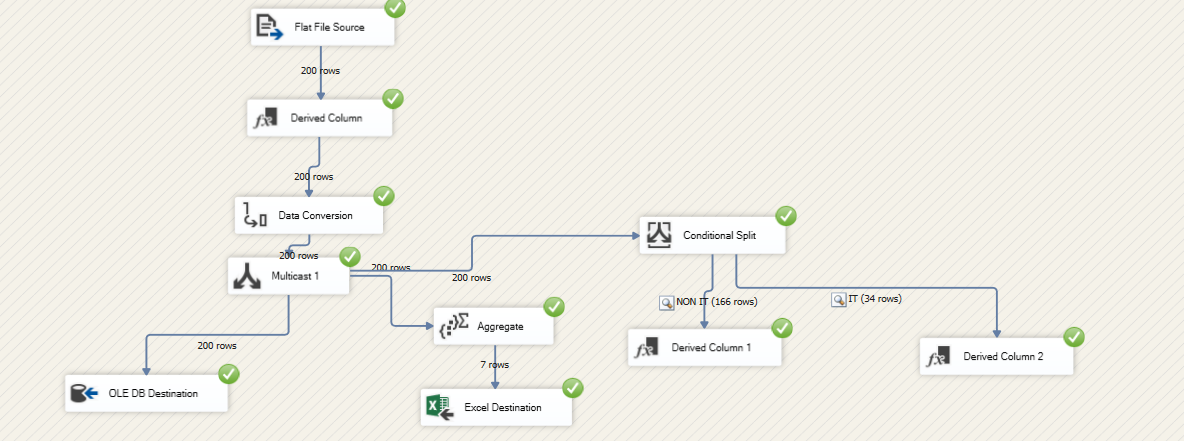
• Screenshots of:

• Successful package deployment

• Environment configurations

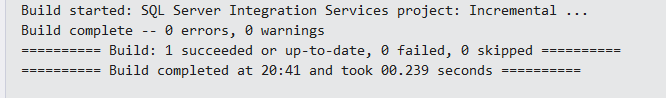
• SQL Server Agent job setup

• Email notifications (success and failure, if triggered)

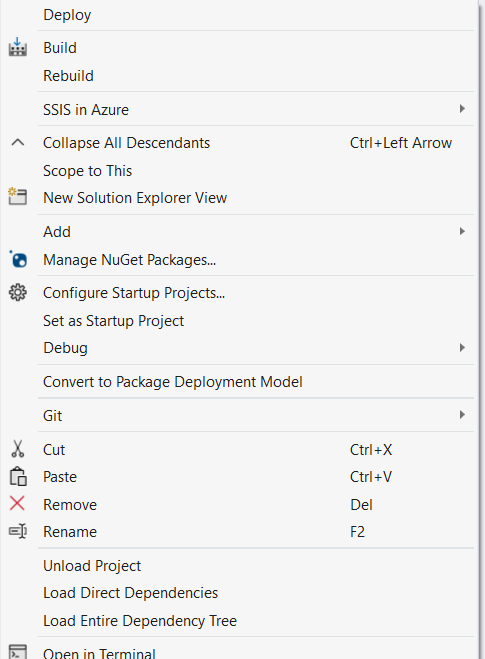


Part 1- Deployment

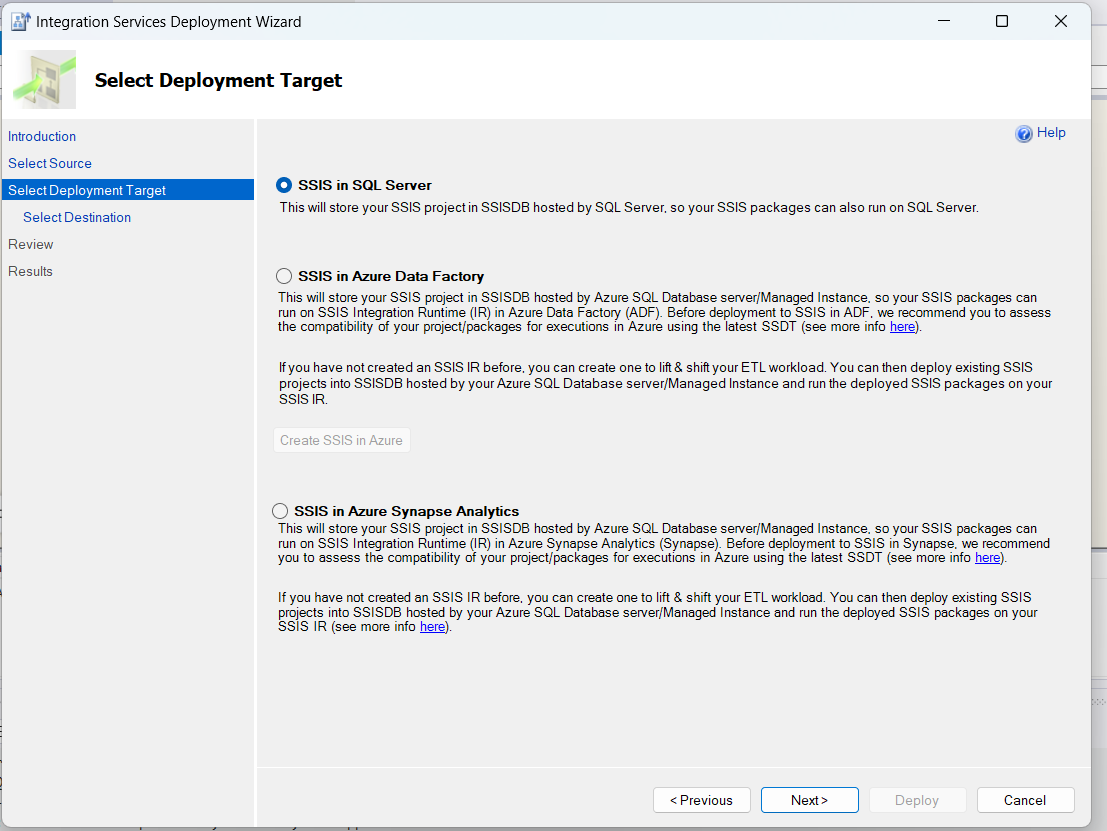
* To deploy , first we have to build the project to ensure there are no errors. We have to als make sure that we have created an integraton service catalog in ssms , to ensure after deployemnt all the packages can be deployed there. For scheduling jobs also we have to make sure that sql server agent is turned on to use this service.



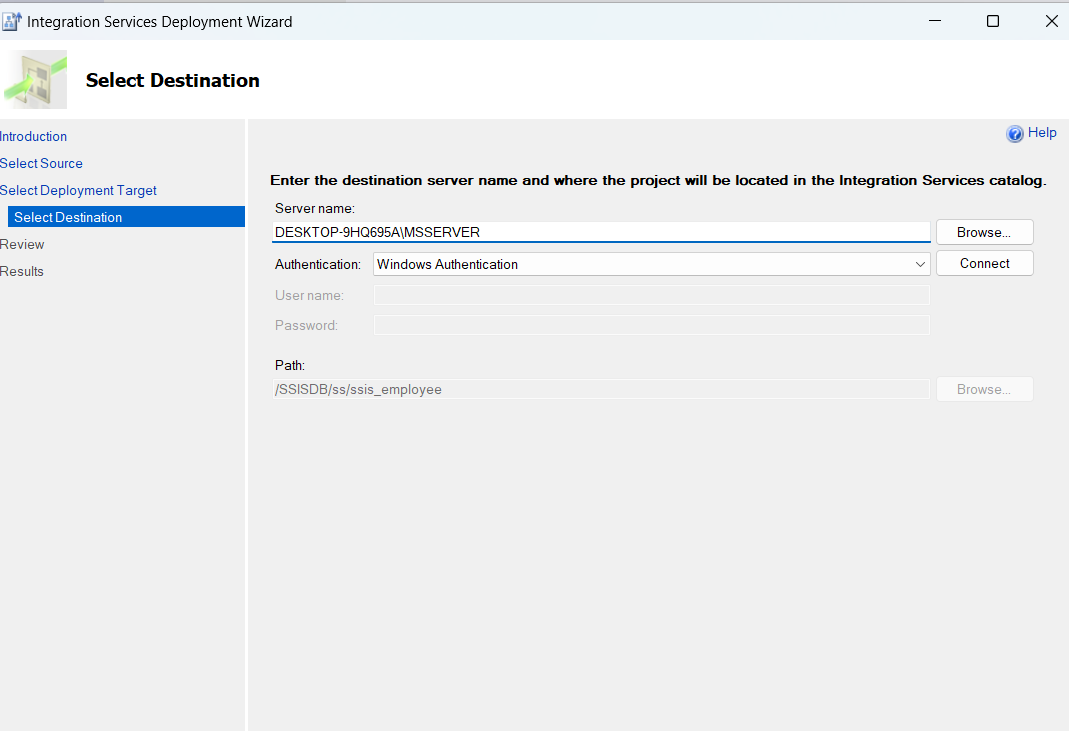
* After successful building, then deploy



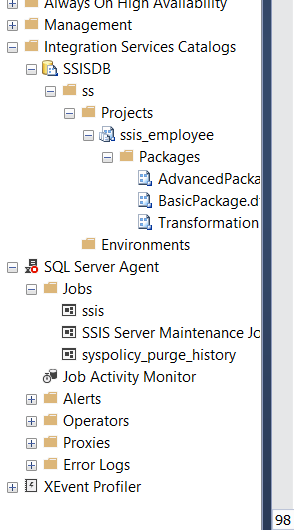
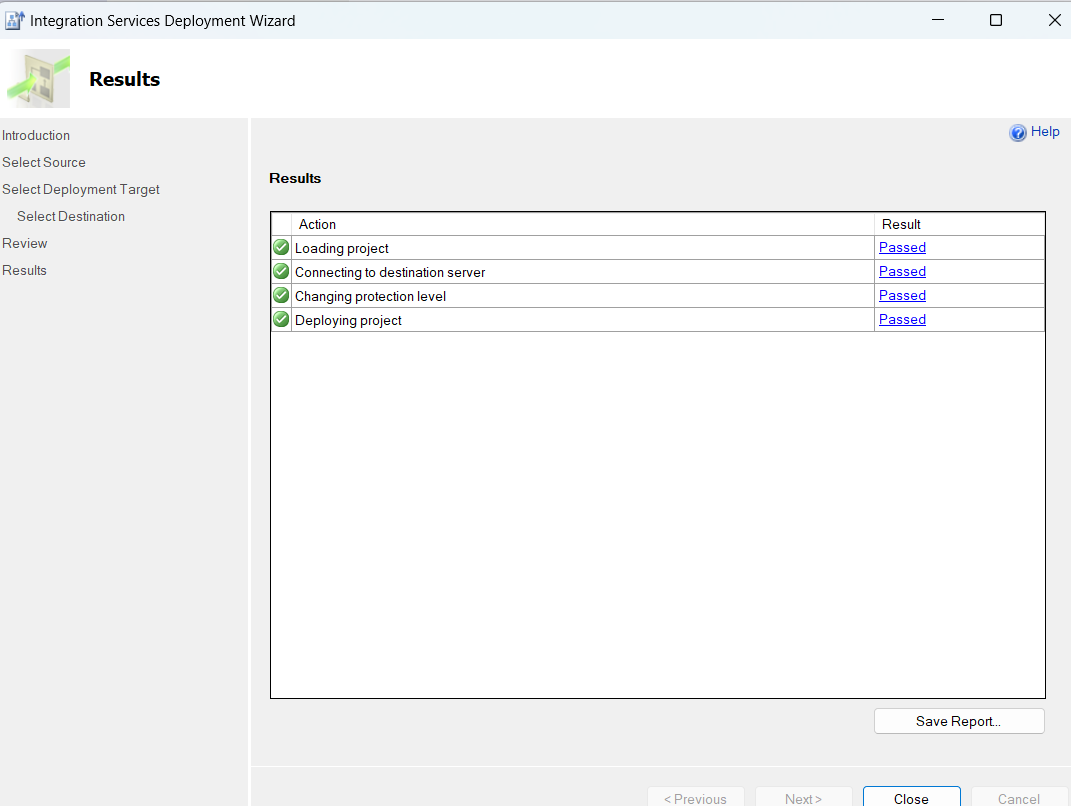
* In this first choose source



* Then select path for folder in integration service runtime for deployement



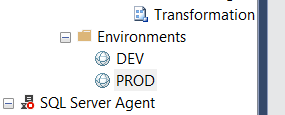
* After successfully deploymen check ssms for files



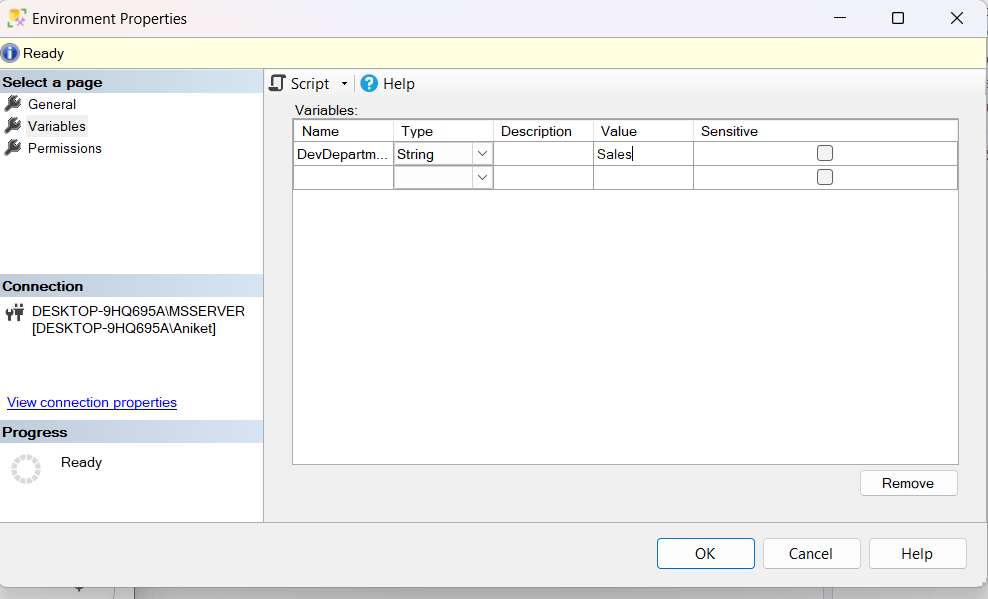
Deployment is done successfully.

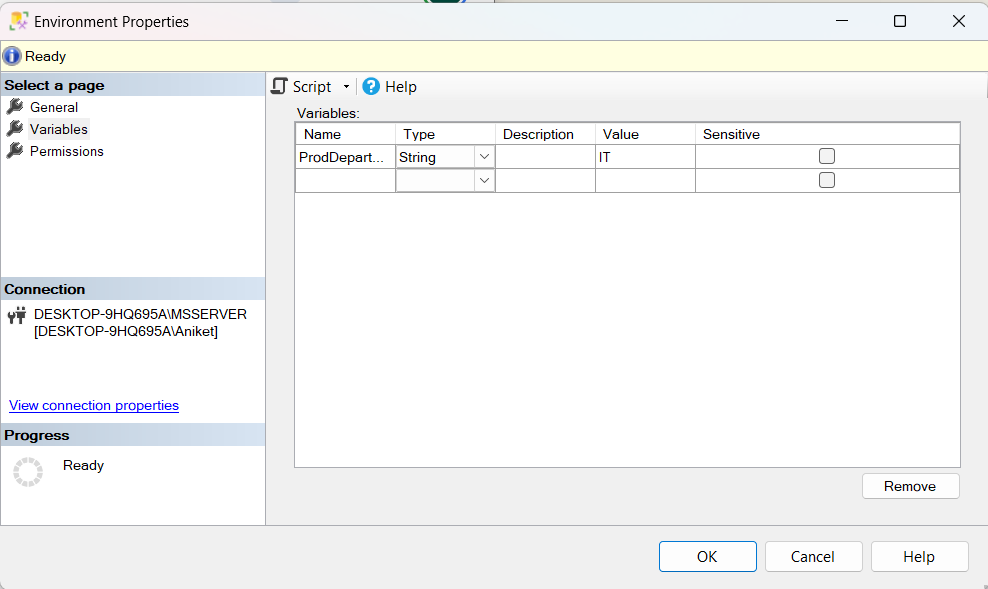
1. Configure Environments for Different Execution Contexts (e.g., Dev, Prod):

First create toe environments dev and prod

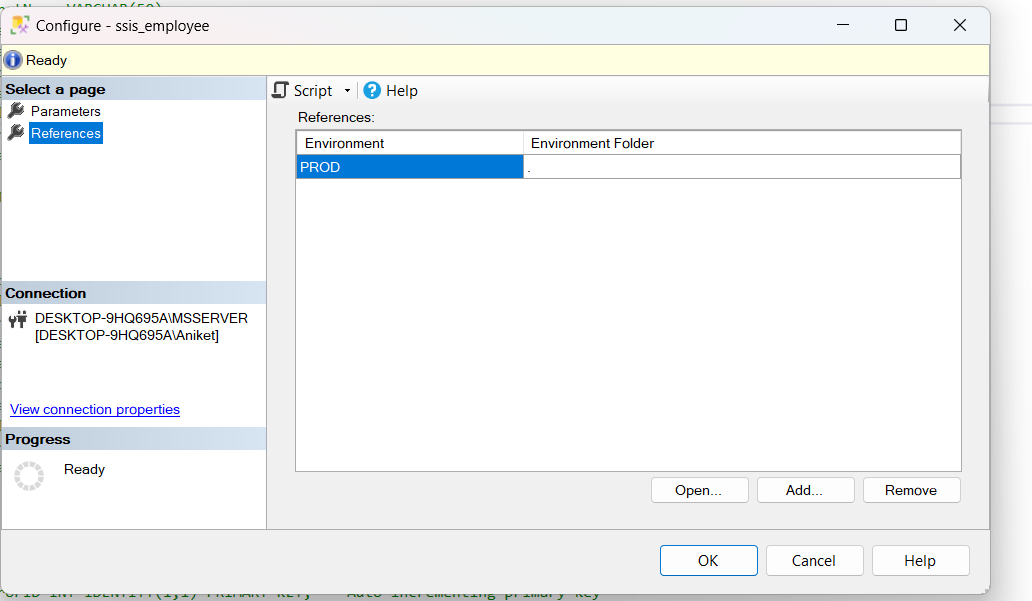


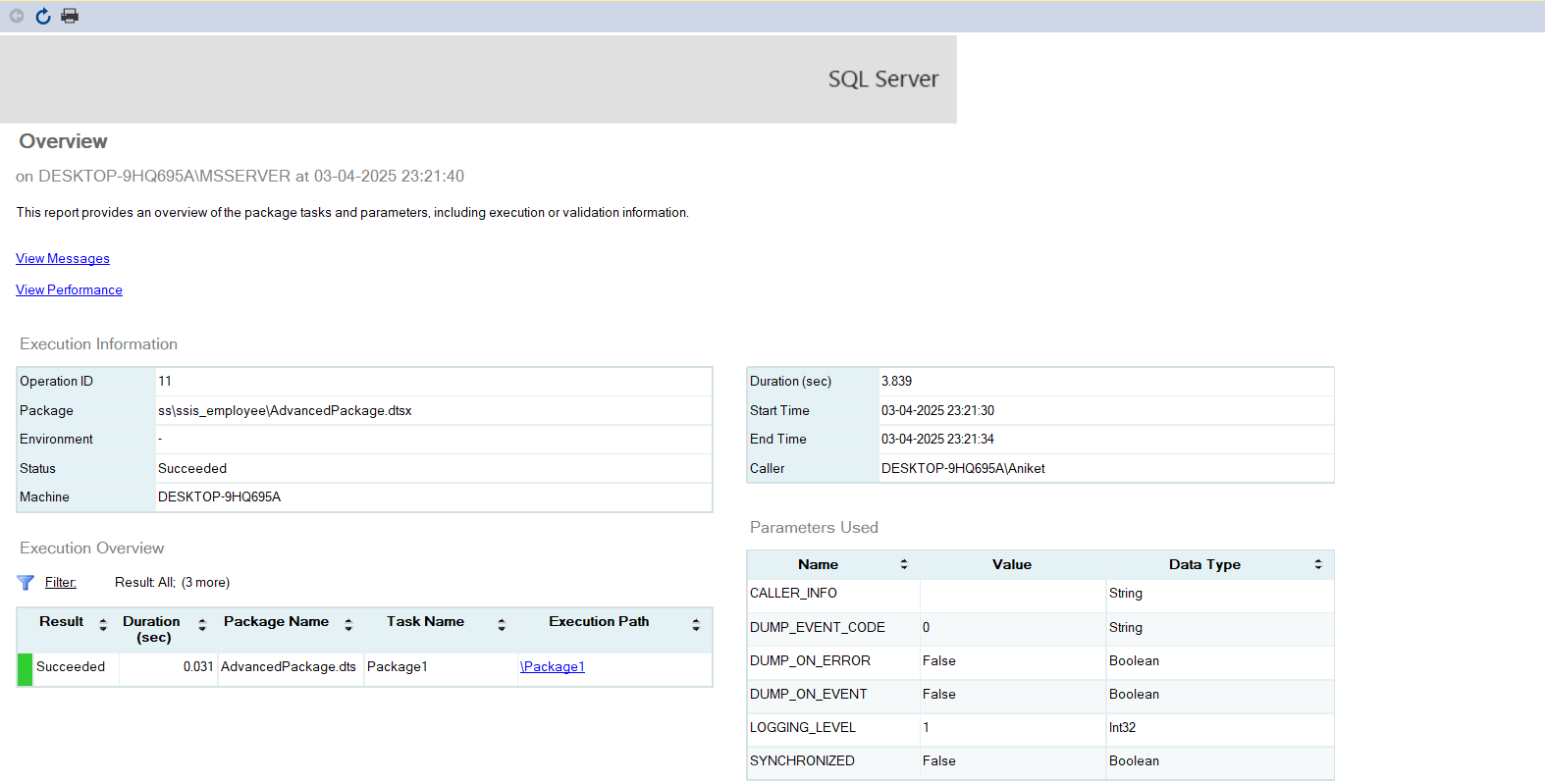
Now add variables for each , as for ths project I added departnement filter in both different





Now configure one fo the files for any environment

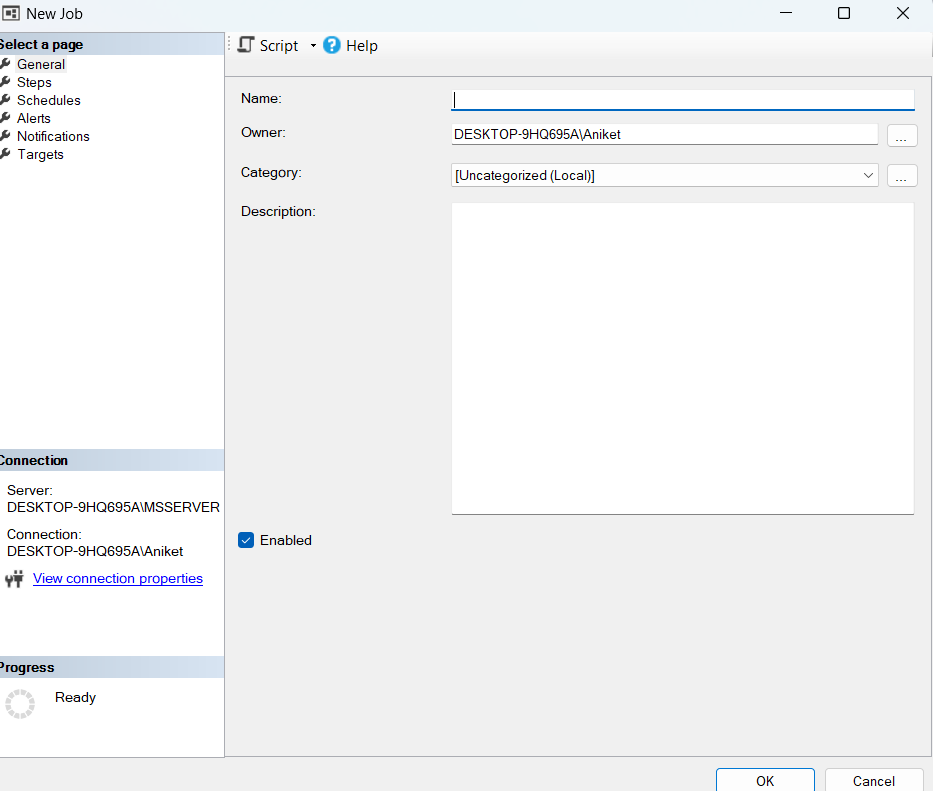


Now click ok and execute the package and see whether it is running in current scenario

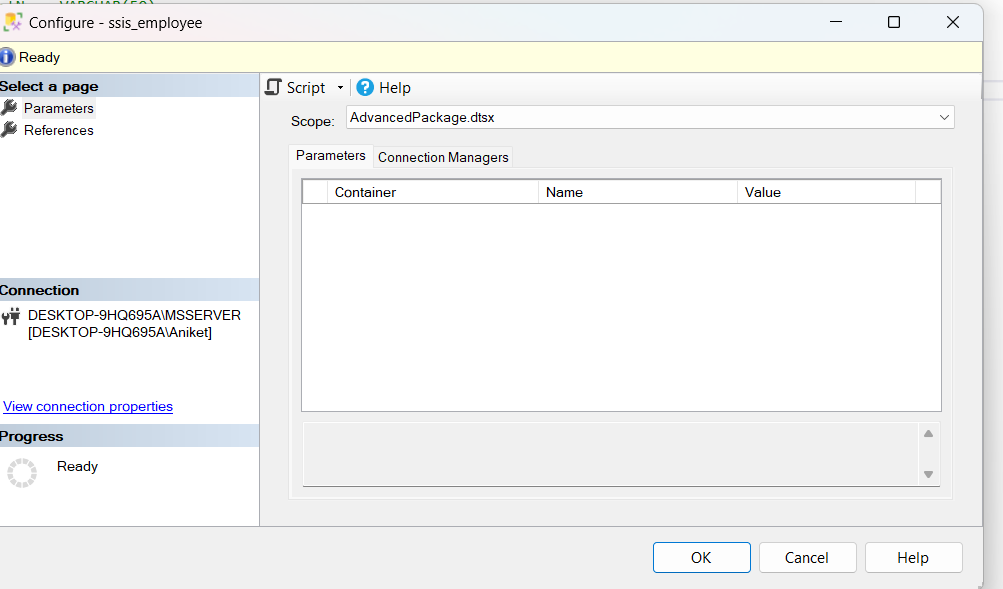
As we can see package is successfully executed.

Part 2: Scheduling

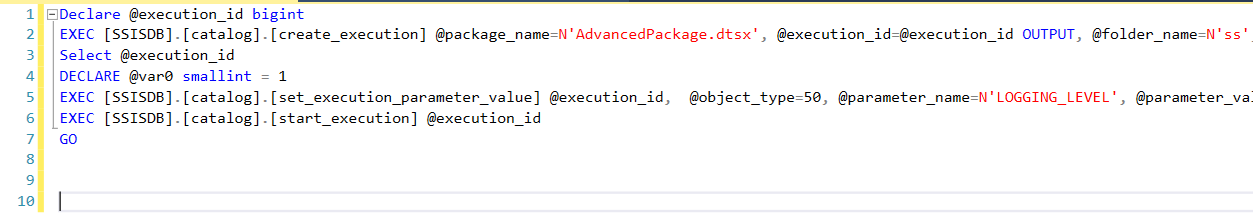
* Go to jobs and click to create new
* First- place name for job



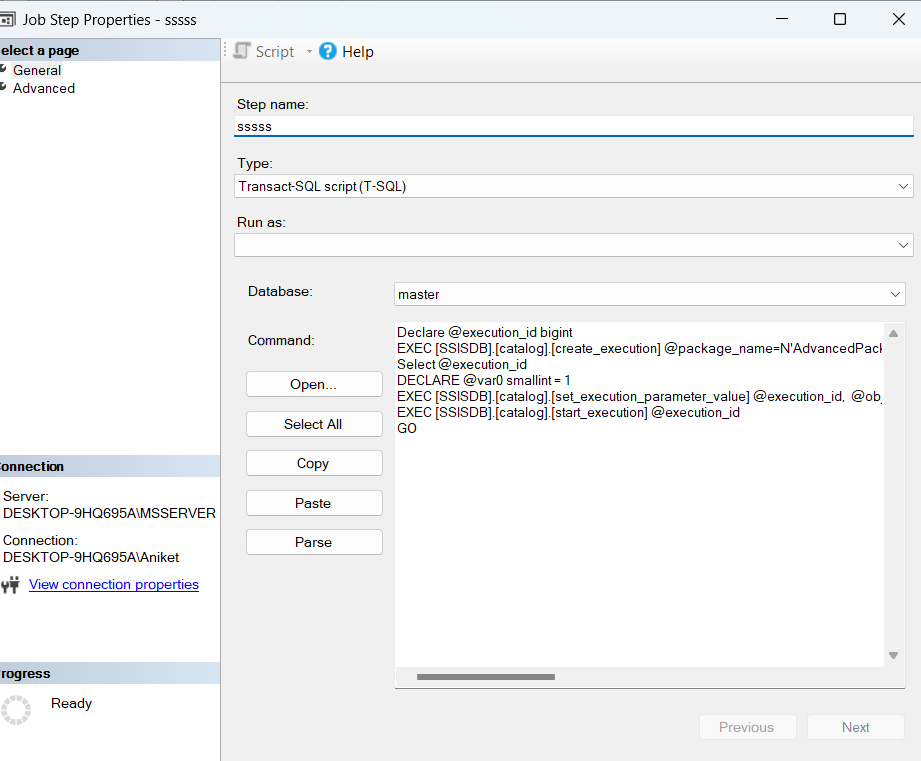
* Then go to steps and give step name and in command you can get this code for which sssis package you want to create job for by clicking configure



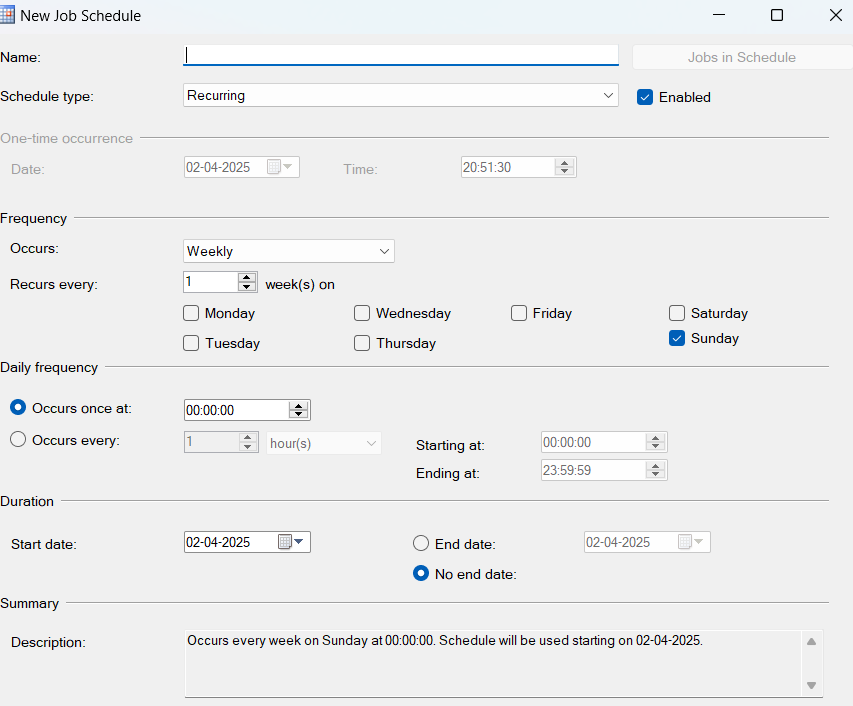
* Then click on script



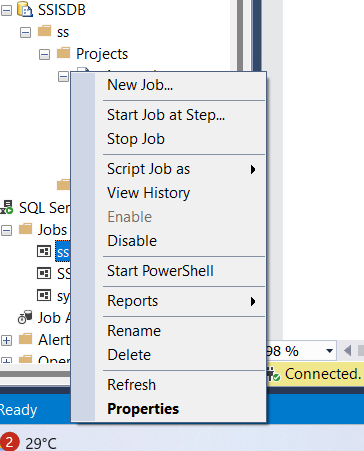
* Then put the command and go to schedule.

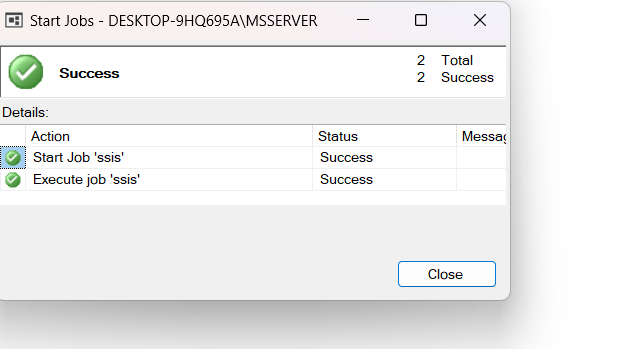


* In this schedule the job acc to your project.



* After scheduling to check whther job created job is running succesfully go to job name created and click in start job





* Job is executed succesfully.
* Email notification s for error in ssms and email.

