



Data Warehousing & Business Intelligent (IT)

3rd Year, 1st Semester

Assignment 1

Submitted to
Sri Lanka Institute of Information Technology

Bachelor of Science Special Honors Degree in Data Science

IT19167510 Bandara J.M.S.A
Weekday Batch

Step 1-Description Of The Data Set

O San Francisco Building permits - kaggle.com

I selected the San Francisco Building Permits as the data set. It consists of a large CSV file with a small xlsx file. Furthermore , I have partitioned the large CSV file into small sub CSV files. The sub CSV files consists of new IDs.

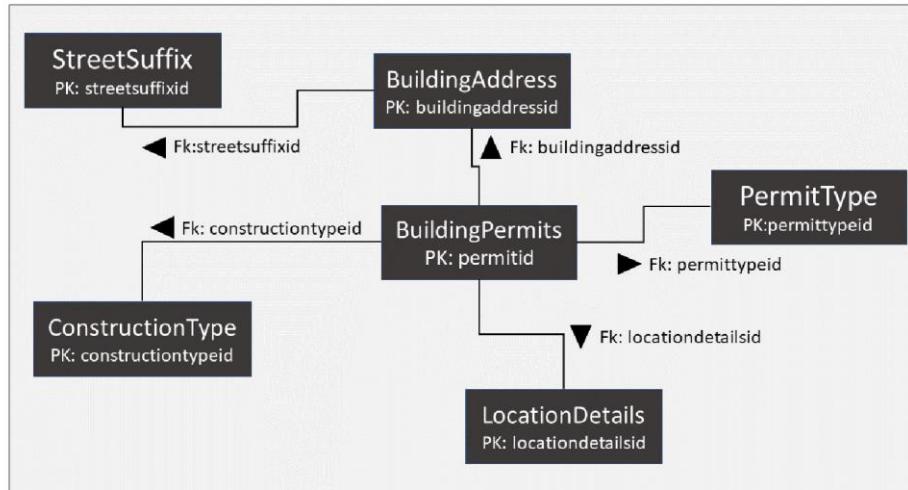
A building permit is an official approval document issued by a governmental agency that allows San Francisco's contractor to proceed with a construction or remodeling project on one's property. San Francisco has its own office related to buildings, that can do multiple functions like issuing permits, inspecting buildings to enforce safety measures, modifying rules to accommodate needs of the growing population etc.

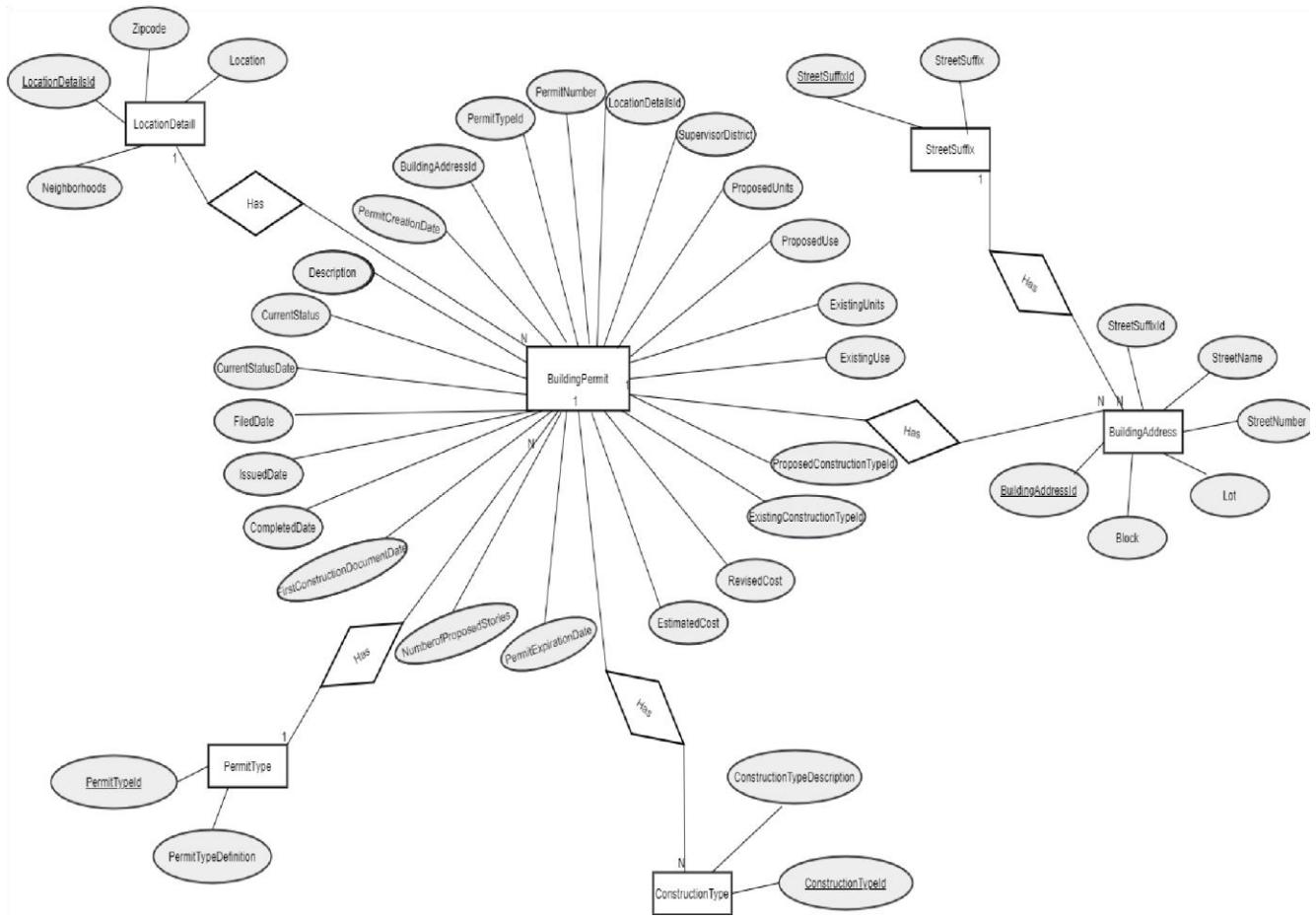
Reason for importance :- In the recent past, main discrepancy in demand and supply in real estate industry is due to delays in issuing building permits

Data set has these files named :

- Building_Permits.csv
- DataDictionaryBuildingPermit.xlsx

O ER Diagram





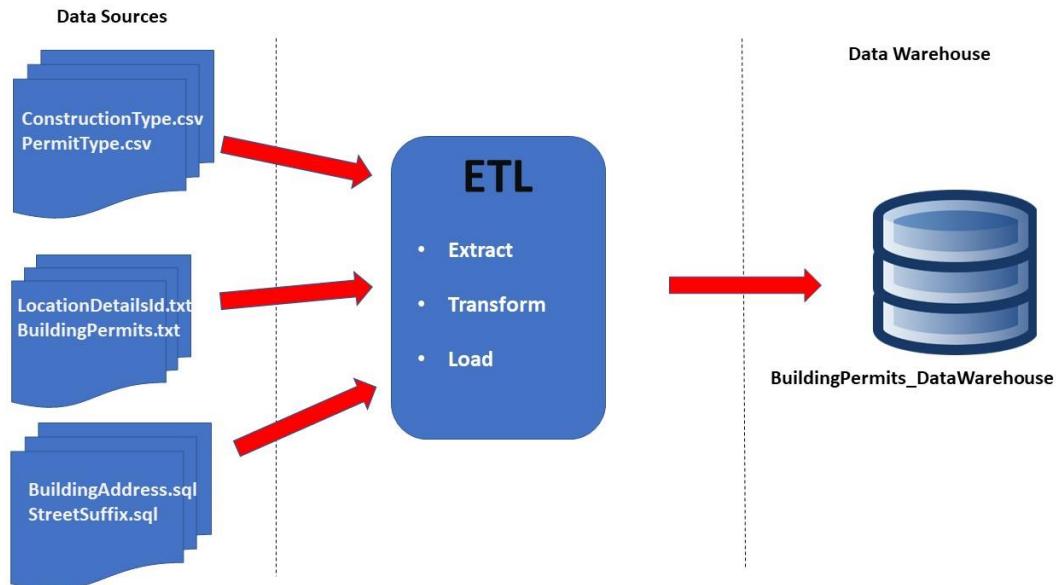
Step 2 – Preparation of data sources

This implement data warehouse used data set view, it is Building Permits data set. It partitioned in to separate Source files to implement data warehouse, such as *Constructiontype.csv* , *PermitType.csv*, *LocationDetailsId.txt*, *BuildingPermits.txt*, *BuildingAddress.sql* and *streetSuffix.sql*

Source	Source Type	Table Name	Column Name	Data Type
ConstructionType.csv	csv	ConstructionType	ConstructionTypeId	nvarchar(8)
			ConstructionTypeDescription	varchar(50)
PermitType.csv	csv	PermitType	PermitTypeId	nvarchar(8)
			PermitTypeDefinition	varchar(50)
LocationDetailsId.txt	Text File	LocationDetailsId	LocationDetailsId	nvarchar(8)
			Neighborhoods	varchar(50)
			Zipcode	numeric(18,0)
			Location	varchar(50)
BuildingPermits_Source	Microsoft SQL Database	BuildingAddress	BuildingAddressId	nvarchar(8)
			Block	nvarchar(50)
			Lot	varchar(50)
			StreetNumber	varchar(50)
			StreetName	varchar(50)
			StreetSuffixId	nvarchar(8)
		StreetSuffix	StreetSuffixId	nvarchar(8)
			StreetSuffix	varchar(50)

BuildingPermits.txt	Text File	BuildingPermits	PermitNumber	varchar(50)
			PermitTypeId	int
			BuildingAddressId	int
			PermitCreationDate	varchar(50)
			Description	varchar(50)
			CurrentStatus	int
			CurrentStatusDate	int
			FiledDate	int
			IssuedDate	int
			CompletedDate	int
			FirstConstructionDocumentDate	numeric(18,0)
			NumberofExistingStories	numeric(18,0)
			NumberofProposedStories	int
			PermitExpirationDate	money
			EstimatedCost	money
			RevisedCost	int
			ExistingConstructionTypeId	int
			ProposedConstructionTypeId	varchar(50)
			ExistingUse	numeric(18,0)
			ExistingUnits	varchar(50)
			ProposedUse	numeric(18,0)
			ProposedUnits	numeric(18,0)
			SupervisorDistrict	int
			LocationDetailsId	int

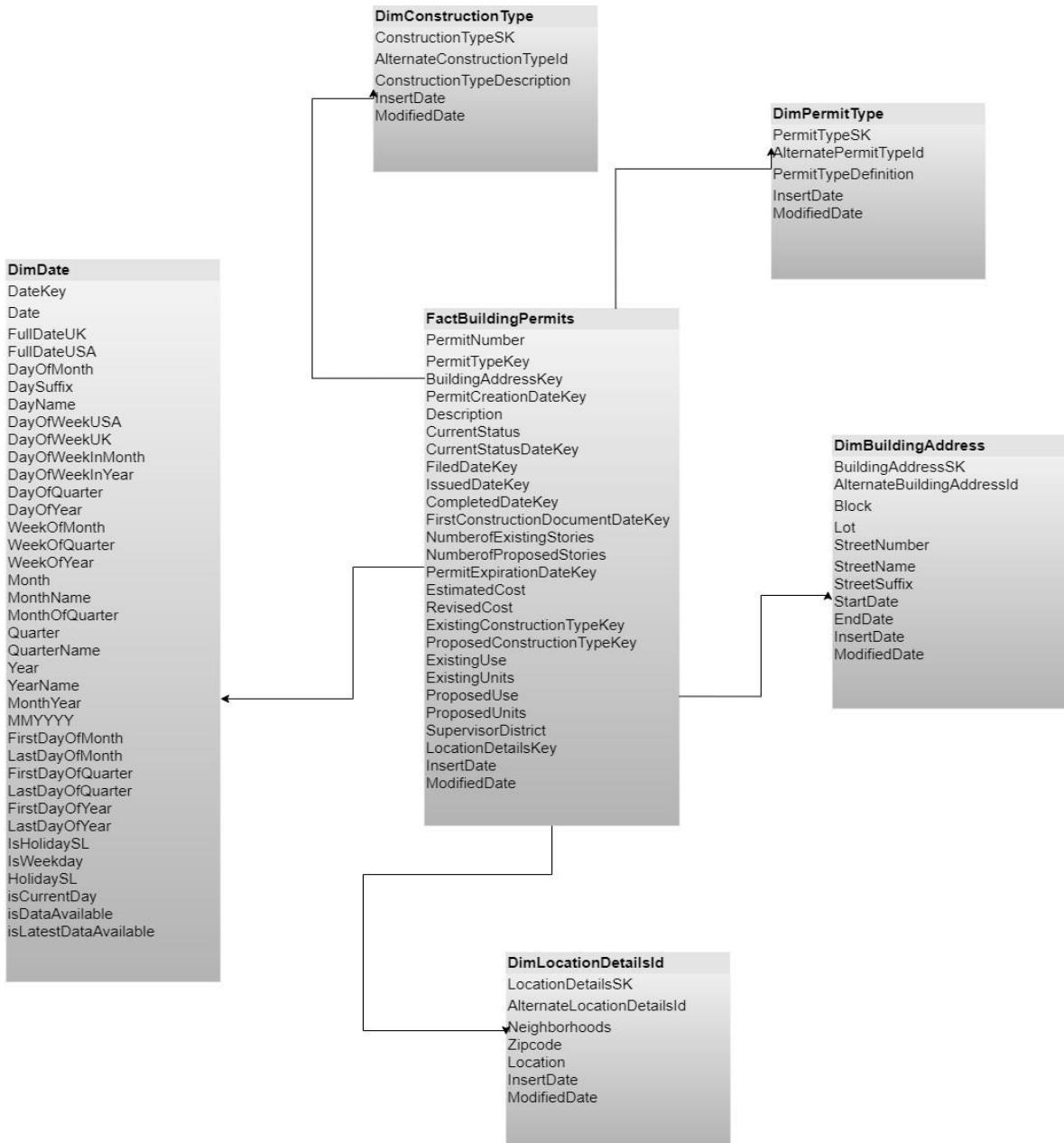
Step 3 - Solution architecture



The datawarehouse is the core of the BI system. We design a datawarehouse for the purpose of the analysis and reporting.

Step 4: Data warehouse design & development

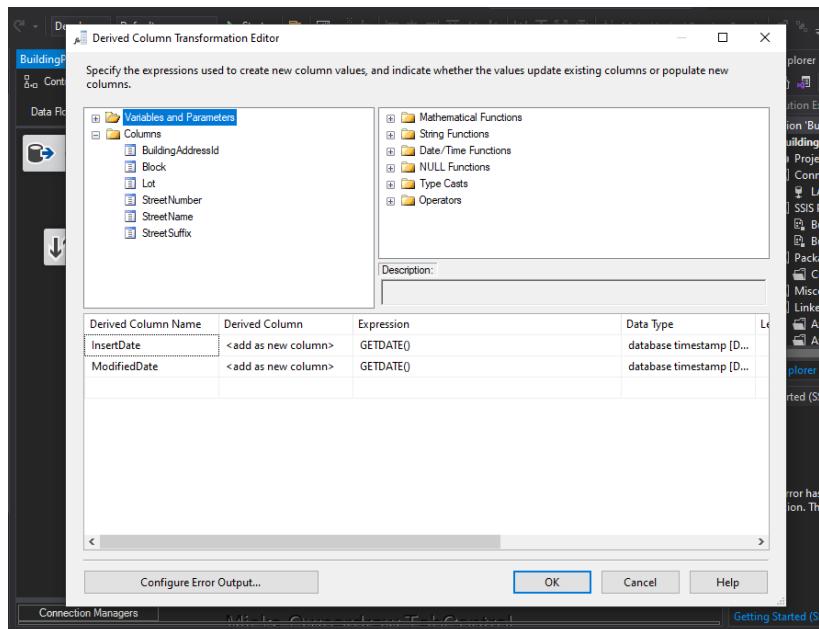
This diagram show you how the fact table and dimension tables are joined in a logical manner. In this fact table,I stored the measurable values and key values.



This Datawarehouse design consists of 5 different dimension table with one fact table. This is a star schema

† Assumption

I have taken **DimBuildingAddress** as **Slowly changing dimension**,Because I assume Street name can change time to time,So we need to keep track of their historical street names. After that I had to create two drived attributes named Insertdate and ModifiedDate when I create this slowly changing dimension.



I created **NumberOfAdditionalStories** and **CostDifference** in the **factBuildingPermits** .

- **NumberOfAdditionalStories = NumberofExistingStories - NumberofProposedStories**
- **CostDifference = RevisedCost – EstimatedCost**

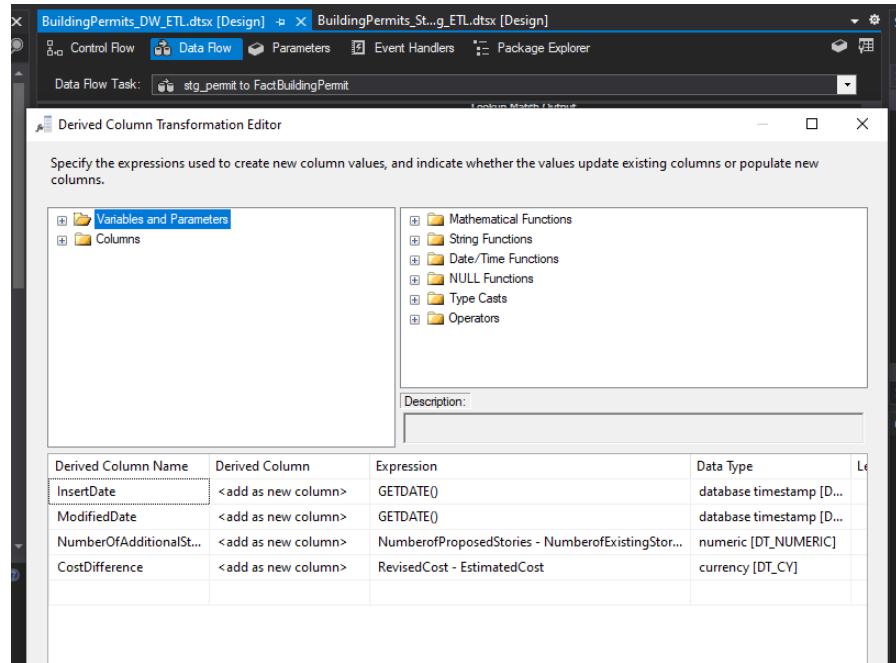
SQL Server Object Explorer

SQLQuery1.sql - LAPTOP-QVIEK211\sandu (54)*

LAPTOP-QVIEK211 (SQL Server 13.0.4001.0 - LAPTOP-QVIEK211)

Column Name	Data Type	Allow Nulls
IssuedDateKey	int	<input checked="" type="checkbox"/>
CompletedDateKey	int	<input checked="" type="checkbox"/>
FirstConstructionDocum...	int	<input checked="" type="checkbox"/>
NumberofExistingStories	numeric(18, 0)	<input checked="" type="checkbox"/>
NumberofProposedStories	numeric(18, 0)	<input checked="" type="checkbox"/>
PermitExpirationDateKey	int	<input checked="" type="checkbox"/>
EstimatedCost	money	<input checked="" type="checkbox"/>
RevisedCost	money	<input checked="" type="checkbox"/>
ExistingConstructionTyp...	int	<input checked="" type="checkbox"/>
ProposedConstructionTy...	int	<input checked="" type="checkbox"/>
ExistingUse	nvarchar(50)	<input checked="" type="checkbox"/>
ExistingUnits	numeric(18, 0)	<input checked="" type="checkbox"/>
ProposedUse	nvarchar(50)	<input checked="" type="checkbox"/>
ProposedUnits	numeric(18, 0)	<input checked="" type="checkbox"/>
SupervisorDistrict	numeric(18, 0)	<input checked="" type="checkbox"/>
LocationDetailsKey	int	<input checked="" type="checkbox"/>
InsertDate	datetime	<input checked="" type="checkbox"/>
ModifiedDate	datetime	<input checked="" type="checkbox"/>
CostDifference	money	<input checked="" type="checkbox"/>
NumberOfAdditionalSto...	numeric(18, 0)	<input checked="" type="checkbox"/>

To create these two measurable values firstly I created a stage called `stg_permit` to factBuilding Permit. Then I created its' data flow task and in a derived column component I created the formula for this values.

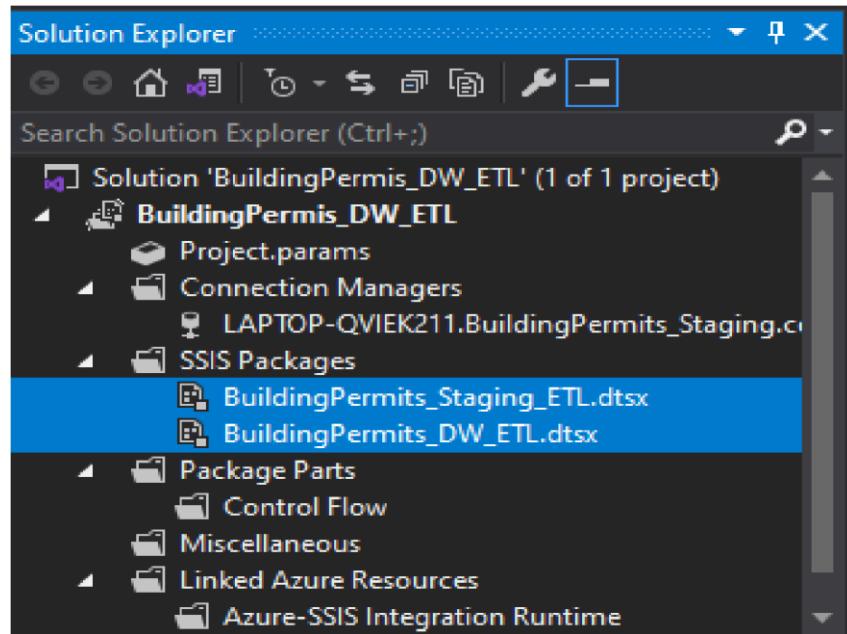


Step 5: ETL development

In the ETL process ,I created ETL process into two Packages

They are ,

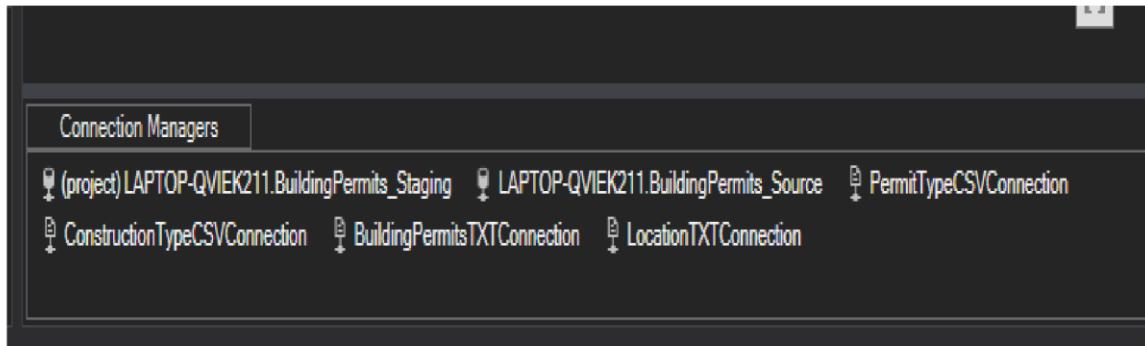
- ◆ BuildingPermits_Staging_ETL.dtsx – Source to Staging
- ◆ BuildingPermits_DW_ETL.dtsx – Staging to Datawarehouse



BuildingPermits_Staging_ETL.dtsx

I created several connections in the BuildingPermits_Staging_ETL package

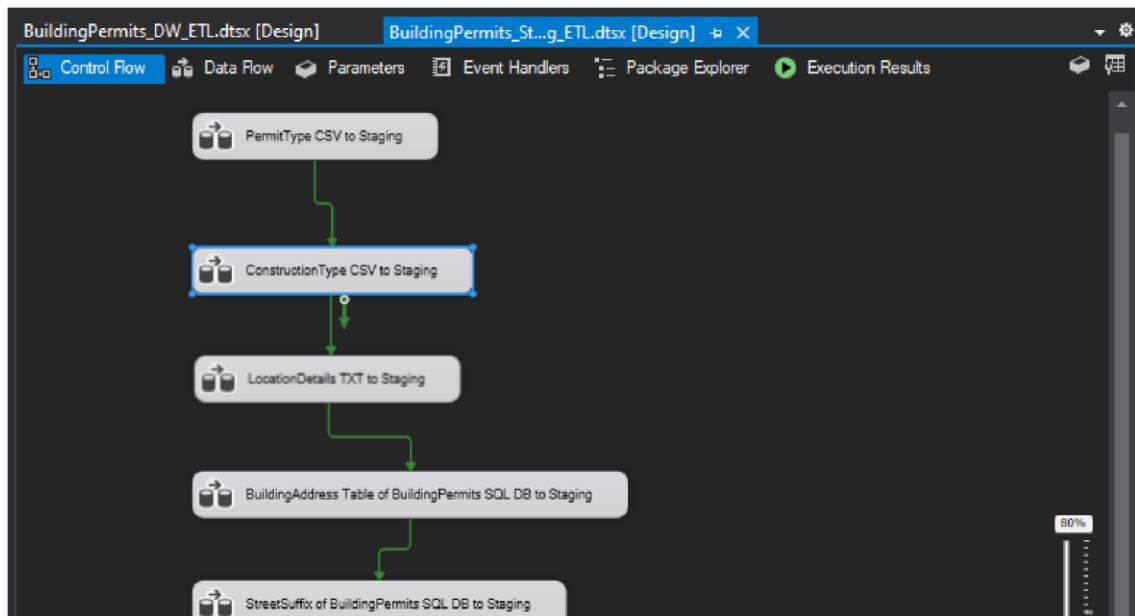
Connection	Description
BuildingPermits_Staging	Connection for OLEDB
BuildingPermits_Sources	Connection for OLEDB
PermitTypeCSVConnection	connection for platfile
ConstructionTypeCSVConnection	connection for platfile
BuildingPermitsTXTConnection	Connection for platfile
LocationTXTConnection	Connection for platfile



First stage – PermitType CSV to Staging

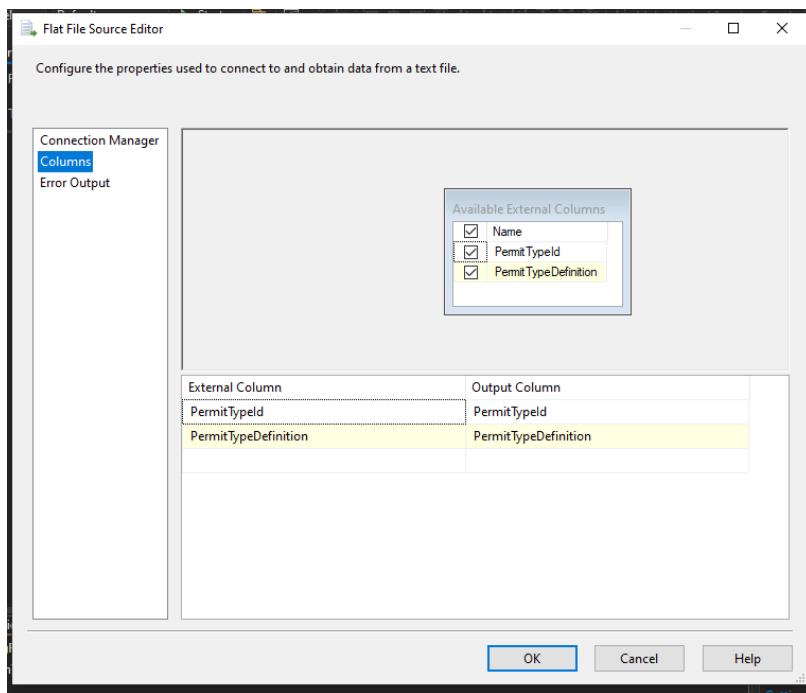
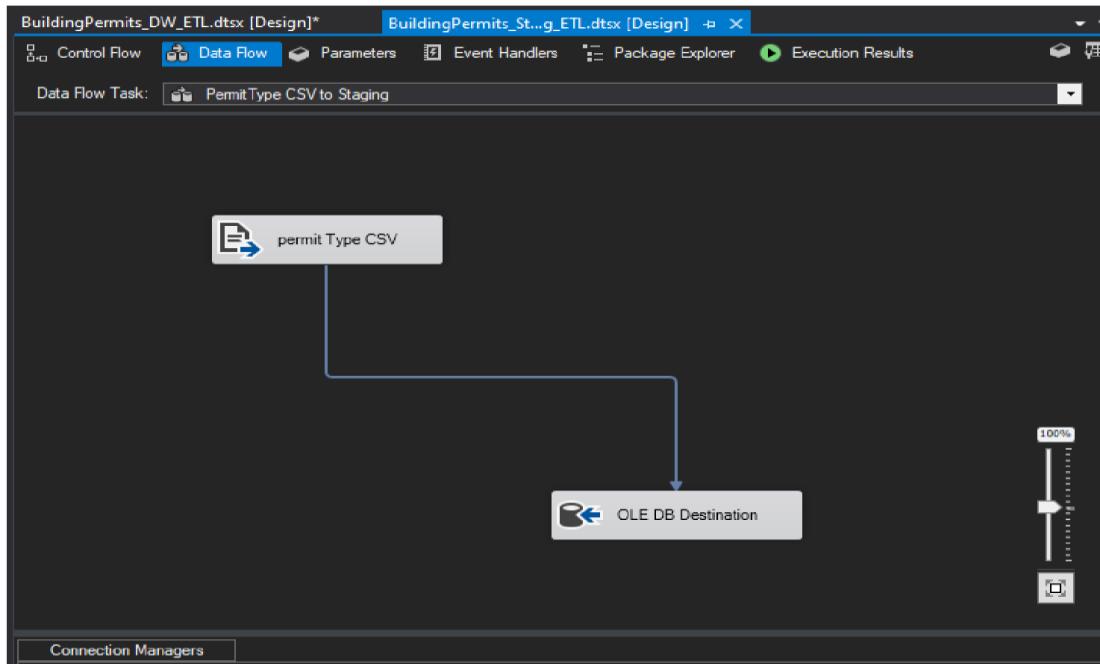
In the control Flow

Firstly I created the control flow of the BuildingPermits_Staging_ETL.



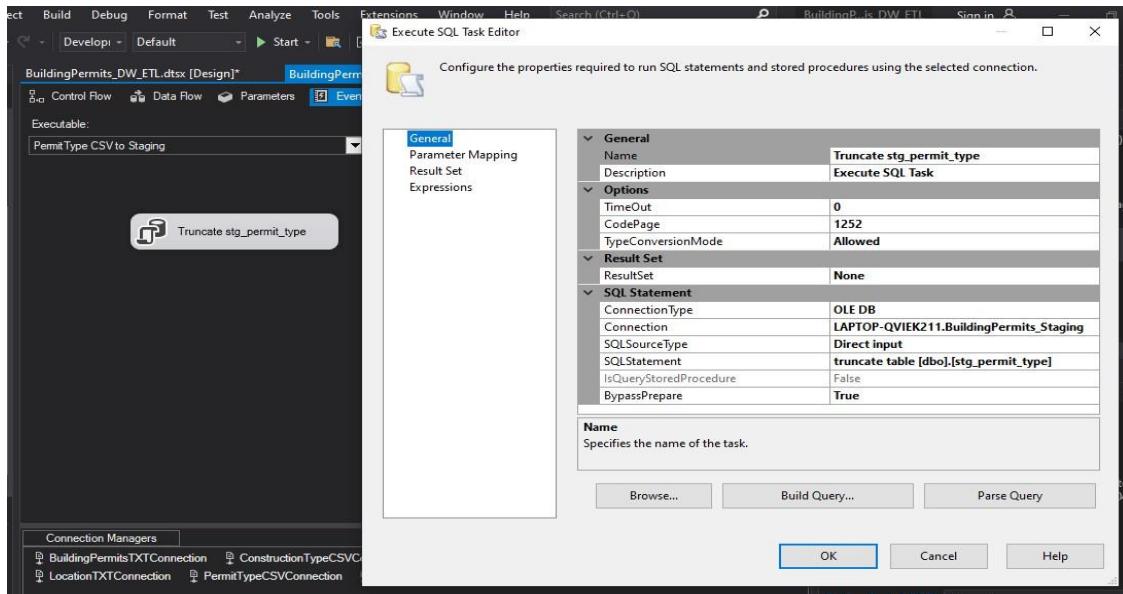
In the Data flow

In the PermitType CSV to staging part I created its' data flow task configuring properties used to connect to and obtain data from a text file.



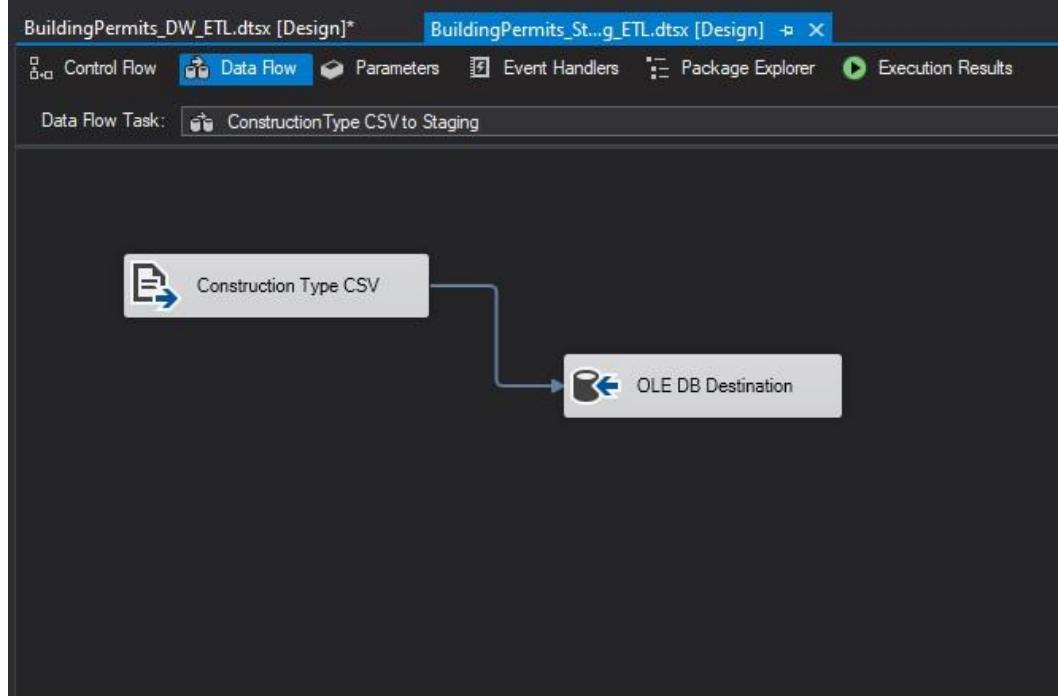
Using Event hanlder for first stage

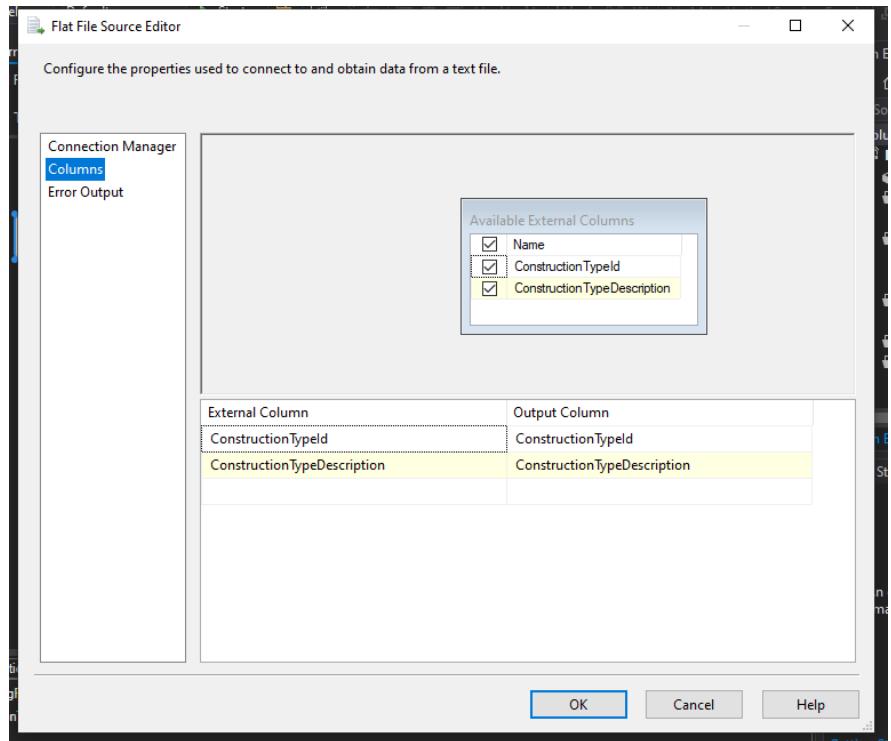
Afterthat I created an event handler For it .I truncated stg_permit_type



Second stage - ConstructionType CSV to staging

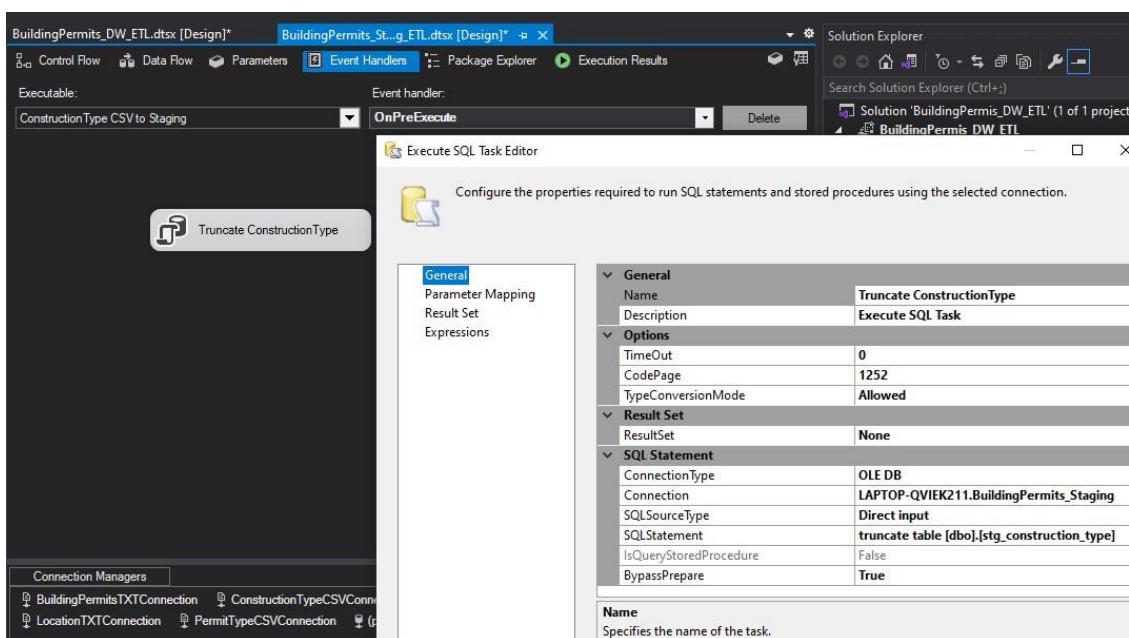
After the PermitType CSV staging I created the ConstructionType to staging, configuring properties used to connect to and obtain data from a text file.





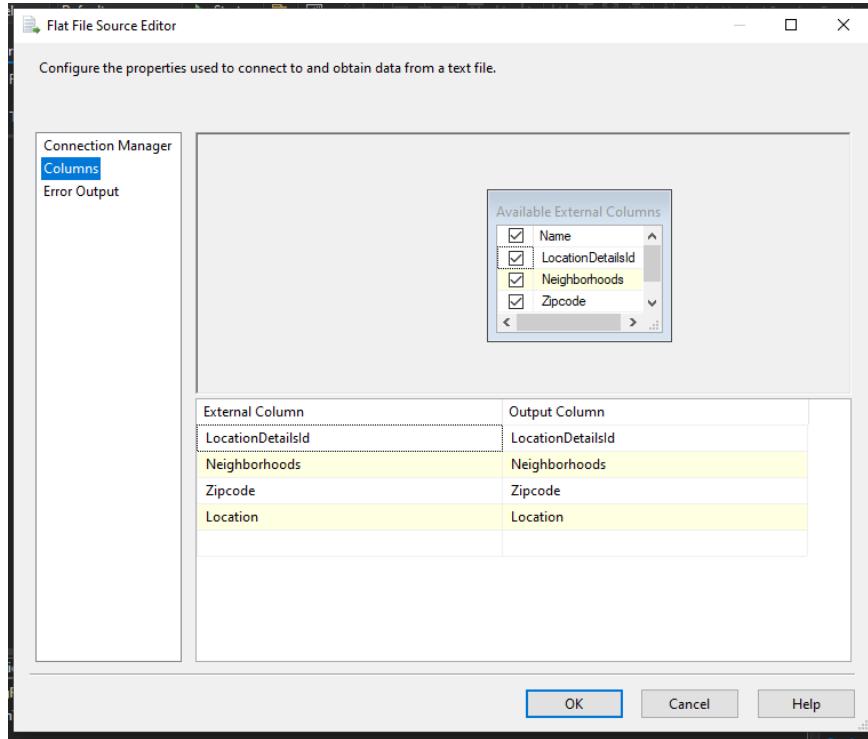
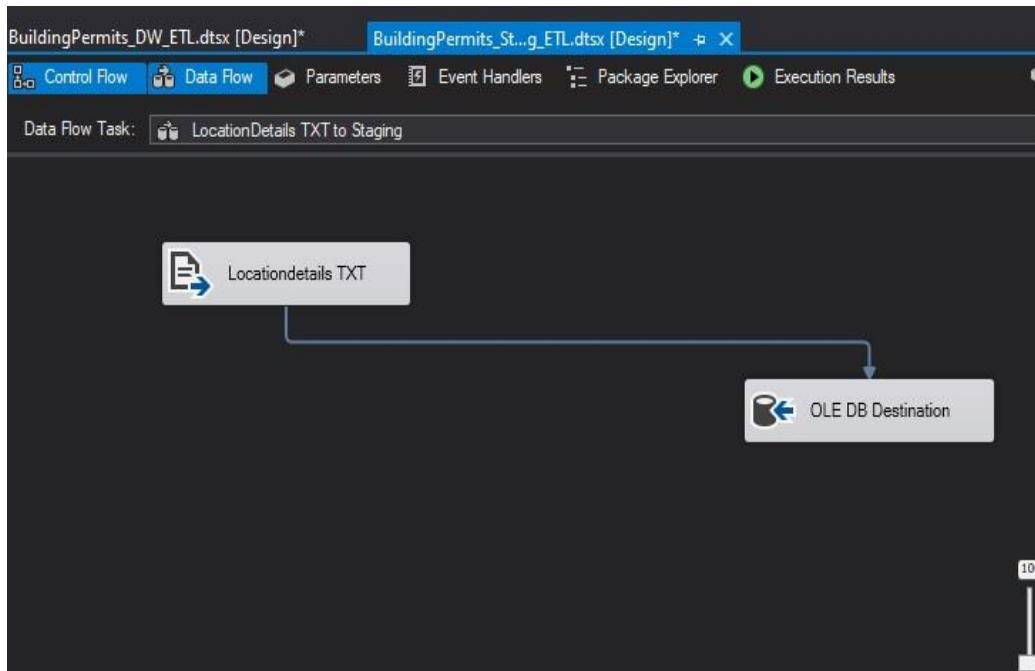
Using Event hanlder for the staging of ConstructionType

After creating Construction type staging I used an event handler For it .I truncated Construction type .



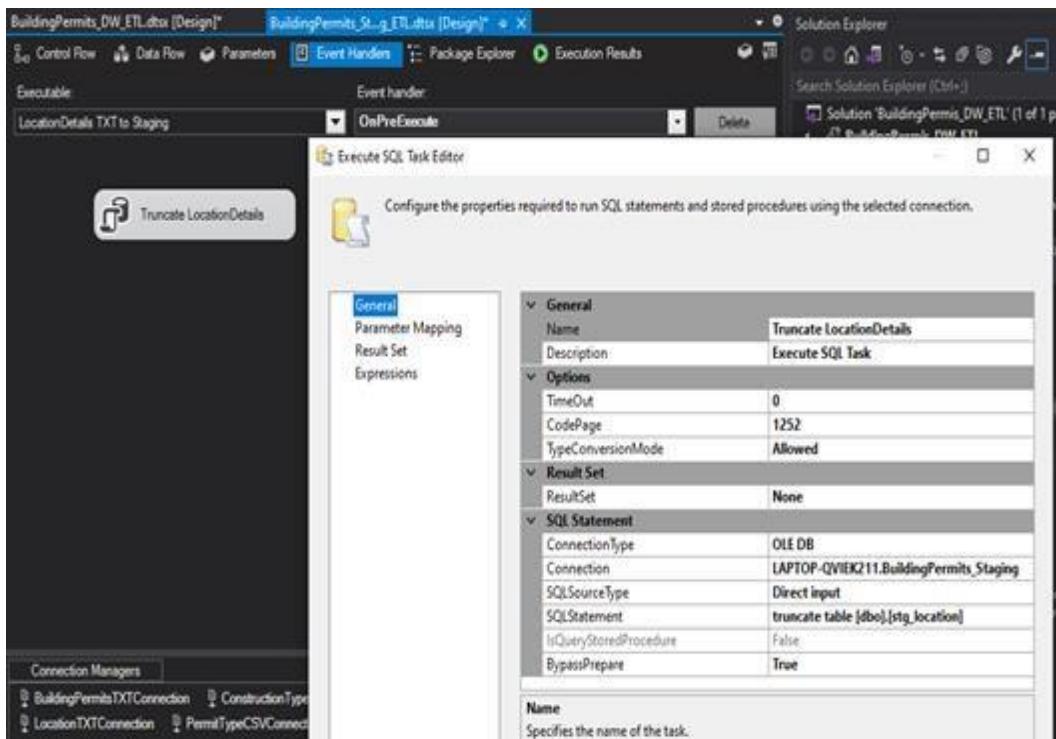
Third Stage – LocationDetails txt to staging

Then I created LocaionDetails txt to Staging, configuring properties used to connect to and obtain data from a text file.



Using Event hanlder for the staging of LocationDetails txt

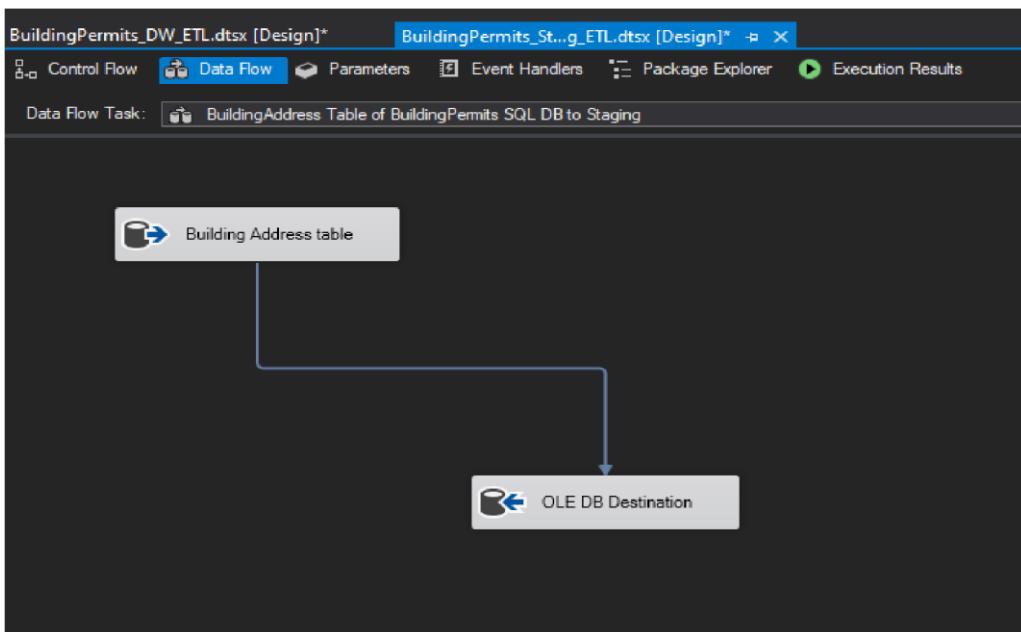
I created an event hanlder for the locationDetails staging .I truncated the location Details.

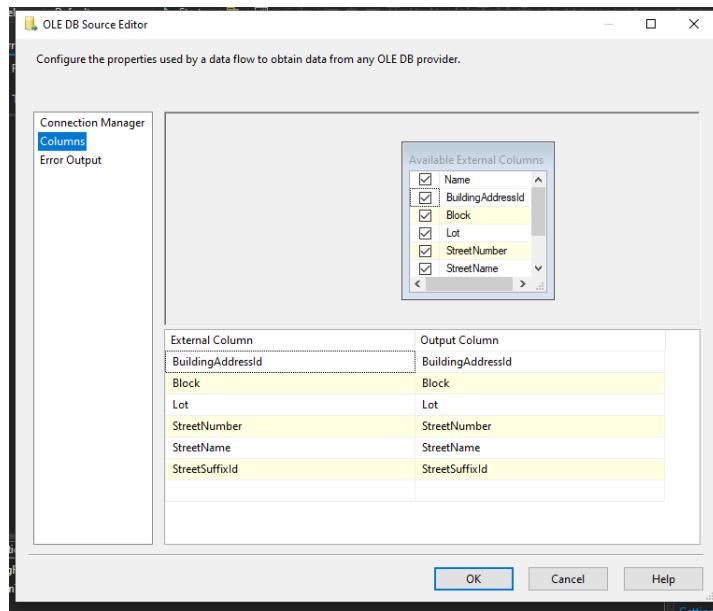


Fourth Stage - BuildingAddress Table of BuildingPermits SQL DB to Staging

In the data flow

I created the data flow task of the Building address table to staging configuring properties used to connect to and obtain data from a text file.





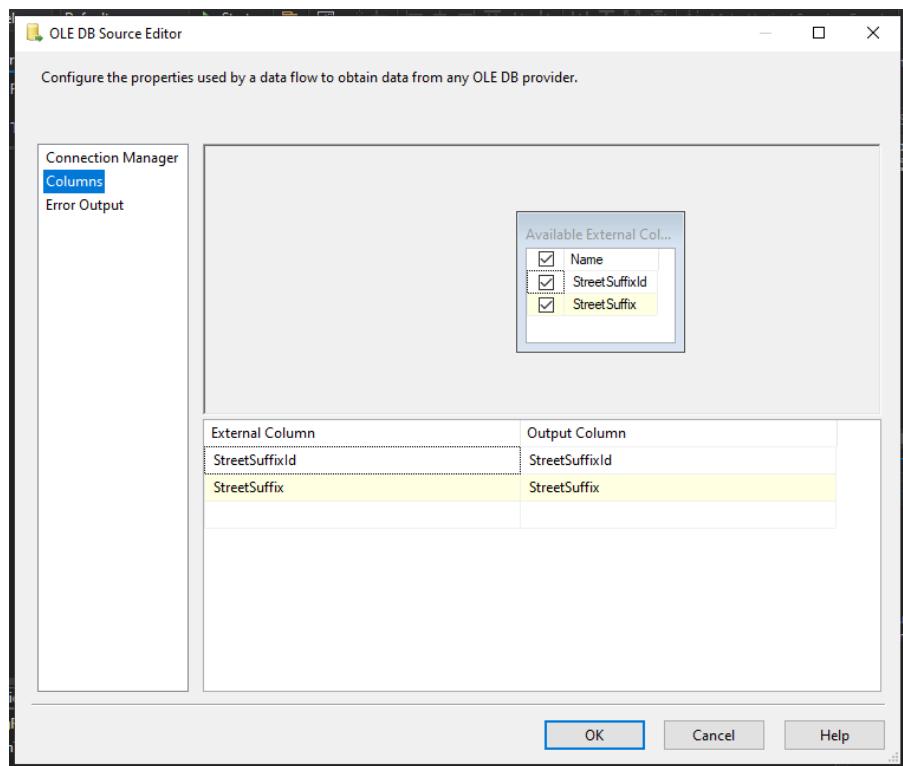
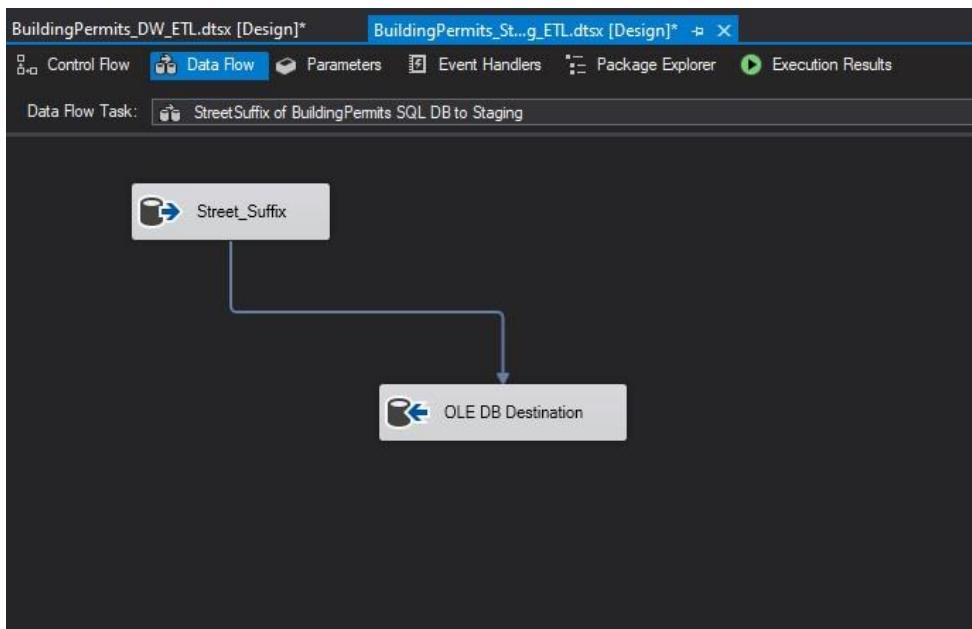
Using Event hanlder for the staging of BuildingAddress table of BuildingPermits SQL DB

After creating of Staging of location Details ,I truncate the Building Address table of BuildingPermits SQL DB

The screenshot shows the 'BuildingPermits_DW_ETL.dtsx [Design]' interface. In the top navigation bar, 'Event Handlers' is selected. Below it, 'Executable' is set to 'BuildingAddress Table of BuildingPermits SQL DB to Staging' and 'Event handler' is set to 'OnPreExecute'. A 'Delete' button is also visible. The main area is the 'Execute SQL Task Editor' window, which displays the configuration for running SQL statements and stored procedures. The 'General' tab shows the task name as 'Truncate Building Address table of BuildingPermits SQL DB to Staging' and the description as 'Execute SQL Task'. The 'Options' tab includes settings for 'TimeOut' (0), 'CodePage' (1252), and 'TypeConversionMode' (Allowed). The 'Result Set' tab has 'ResultSel' set to 'None'. The 'SQL Statement' tab contains the SQL statement 'truncate table [dbo].[stg_address]'. Other settings in the 'SQL Statement' tab include 'Connectiontype' (OLE DB), 'Connection' (LAPTOP-QVIEK211.BUILDINGPERMITS_STAGING), 'SQLSourceType' (Direct input), and 'IsQueryStoredProcedure' (False). The 'BypassPrepare' option is set to 'True'. A note at the bottom of the editor states: 'Name Specifies the name of the task.'

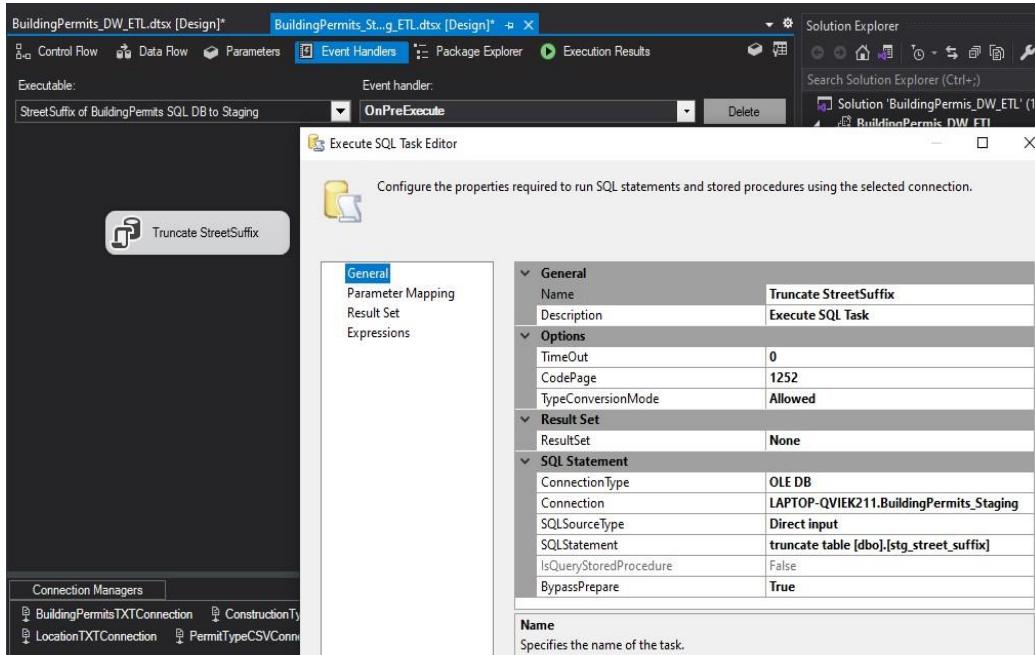
Fifth Stage - StreetSuffix of BuildingPermits SQL DB to Staging

After the Staging of location Details I created streetSuffix of buldingPermits SQL DB staging.



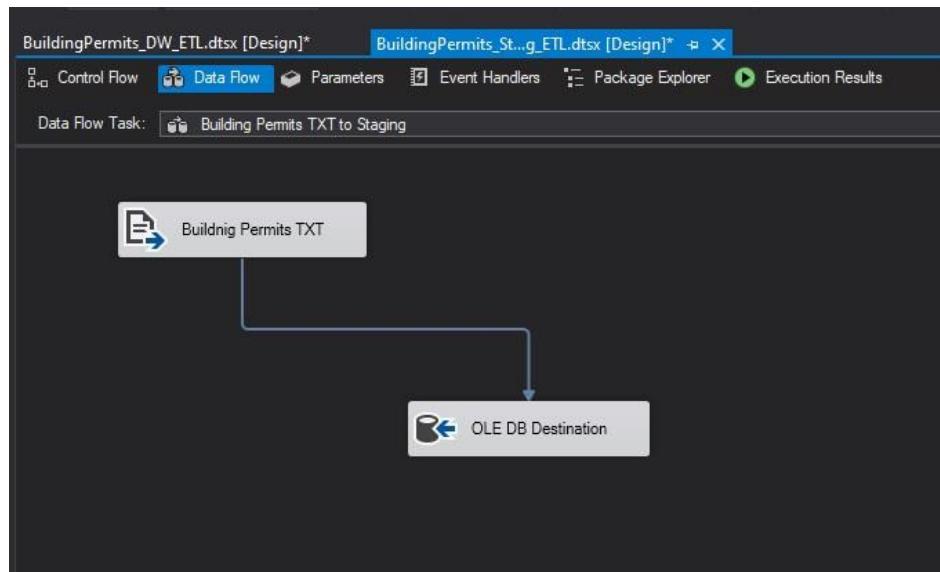
Using Event hanlder for the staging of Streetsuffix of BuildingPermits SQL DB Then

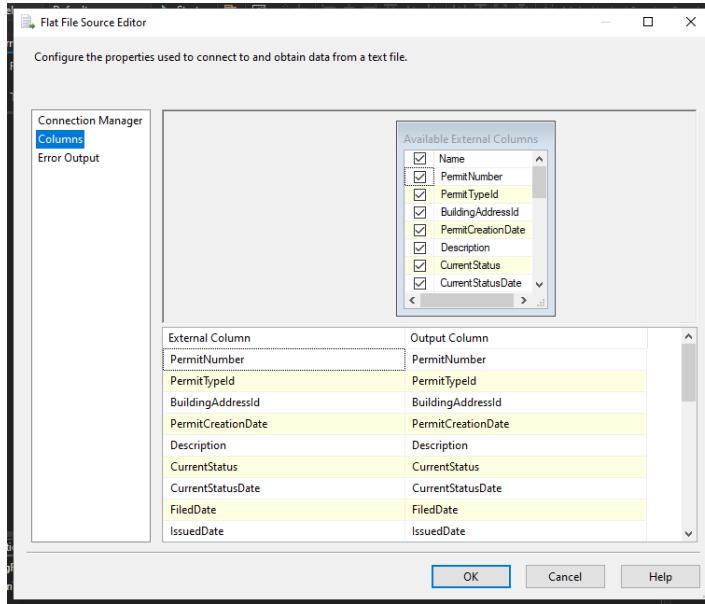
I truncated the StreetSuffix of BuildingPermits SQL DB



Sixth Stage - Building Permits TXT to Staging

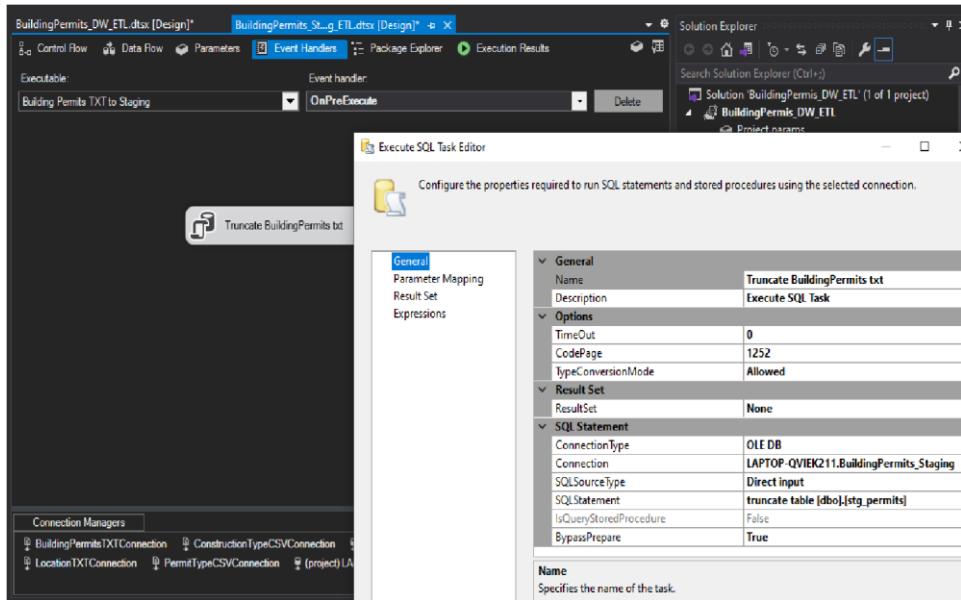
After staging of street suffix I designed the Building Permits TXT staging.





Using Event hanlder for the staging of Building Permits TXT

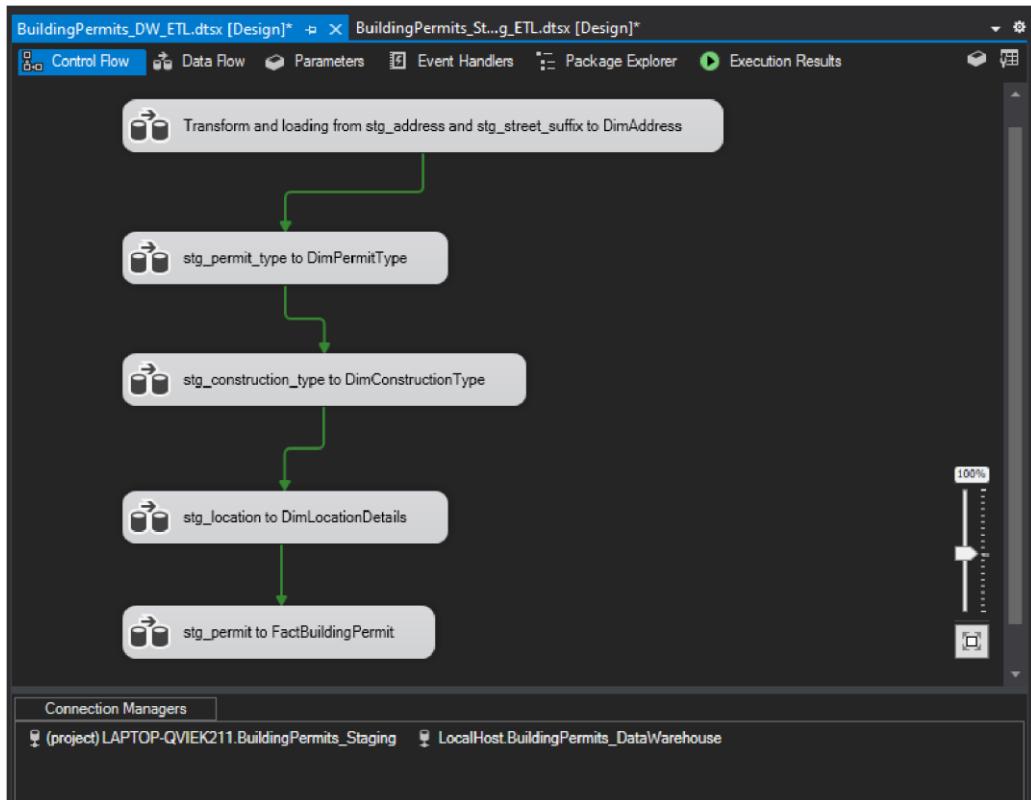
After creating the BuildingPermits staging I truncated the Building Permits



After creating staging step I create task to execute the package.

BuildingPermits_DWH_ETL.dtsx

There are two connections in the BuildingPermits_DWH_ETL package .Because I created this package for staging to warehouse.In here I designed an another controlflow for staging to datawarehouse



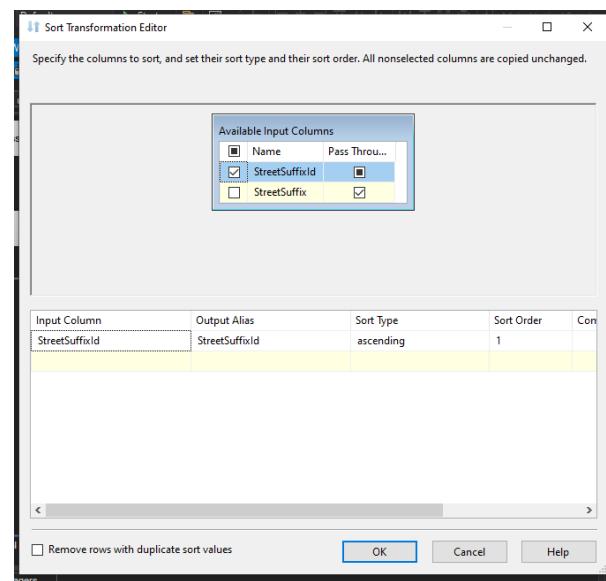
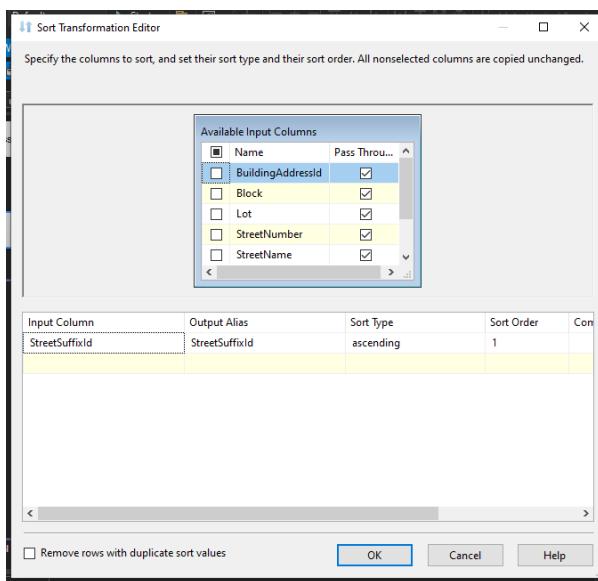
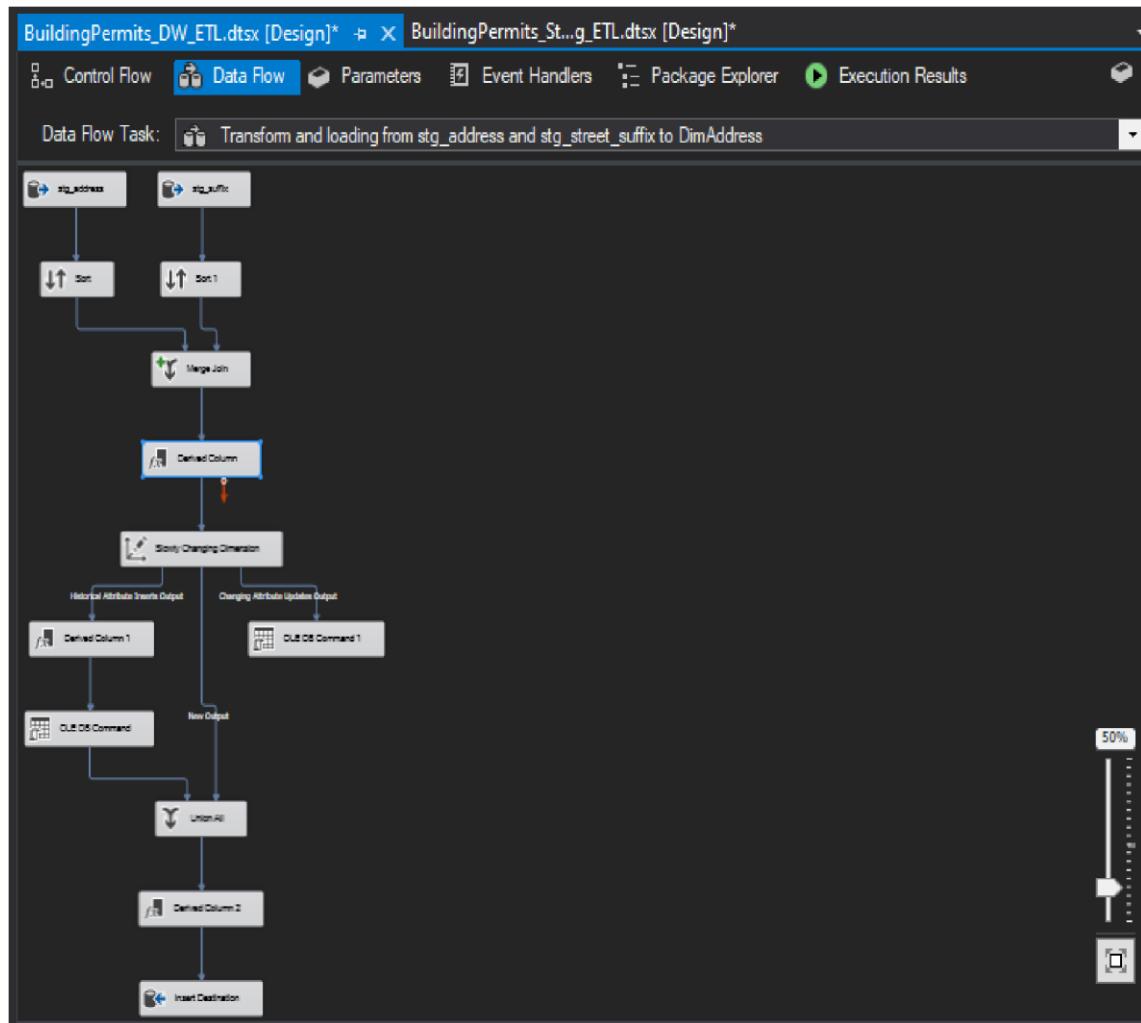
1.Transform and loading from stg_address and stg_street_suffix to DimBuildingAddress

In this step I created task to transform and loading staging address and street suffix to DimBuildingAddress.

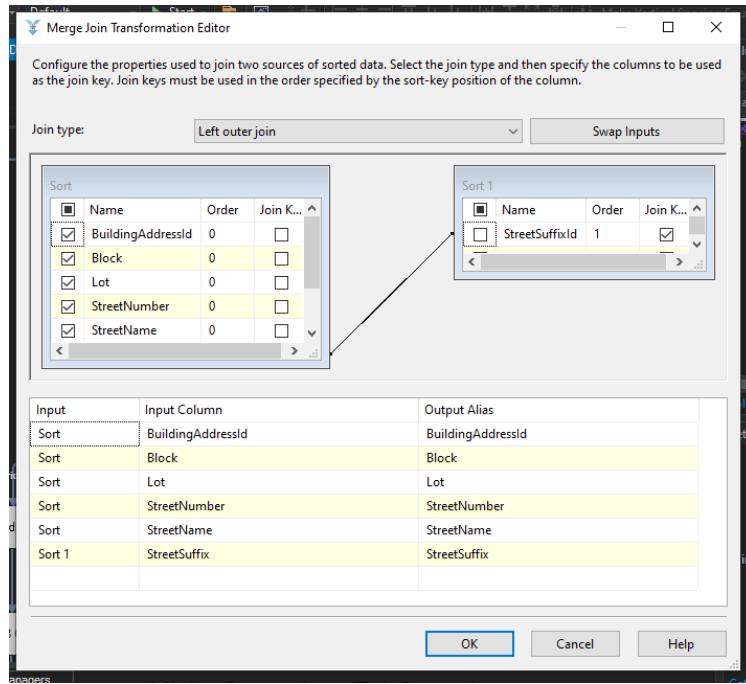
In this Transform and Loading process I have to get stg_address and stg_street_suffix tables to load the DimAddress table.So I take two **sort components** using StreetSuffixId and **merge** them(stg_address table and stg_street_suffix table) and derived columns using **derived column component**.

Then I added **slowly changing dimension component** to update the DimBuildingAddress table records.

So as shown below I have transformed and loaded data

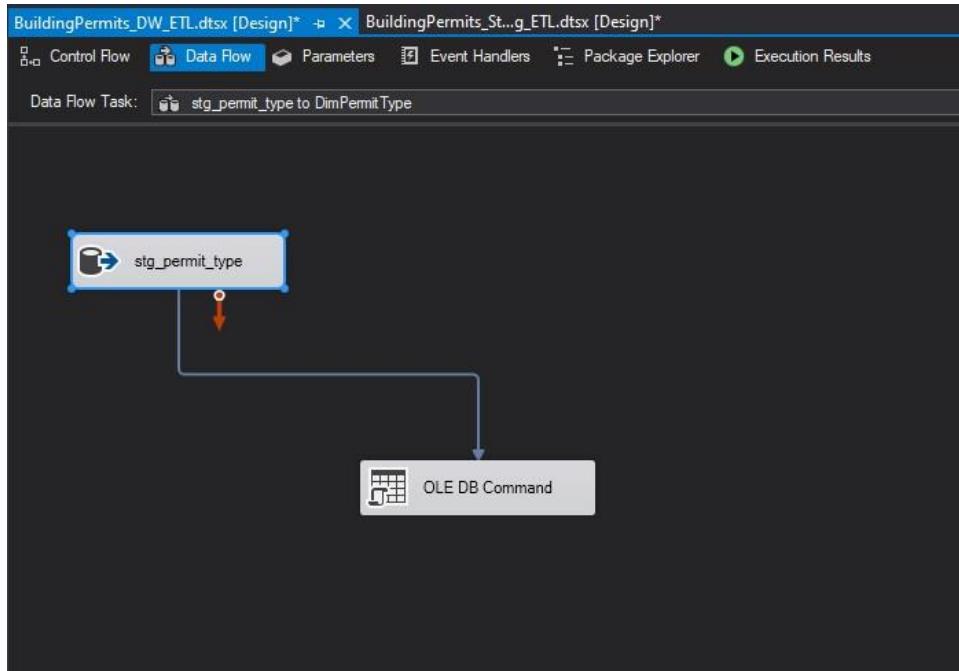


As shown below I used merge join component to merge them.



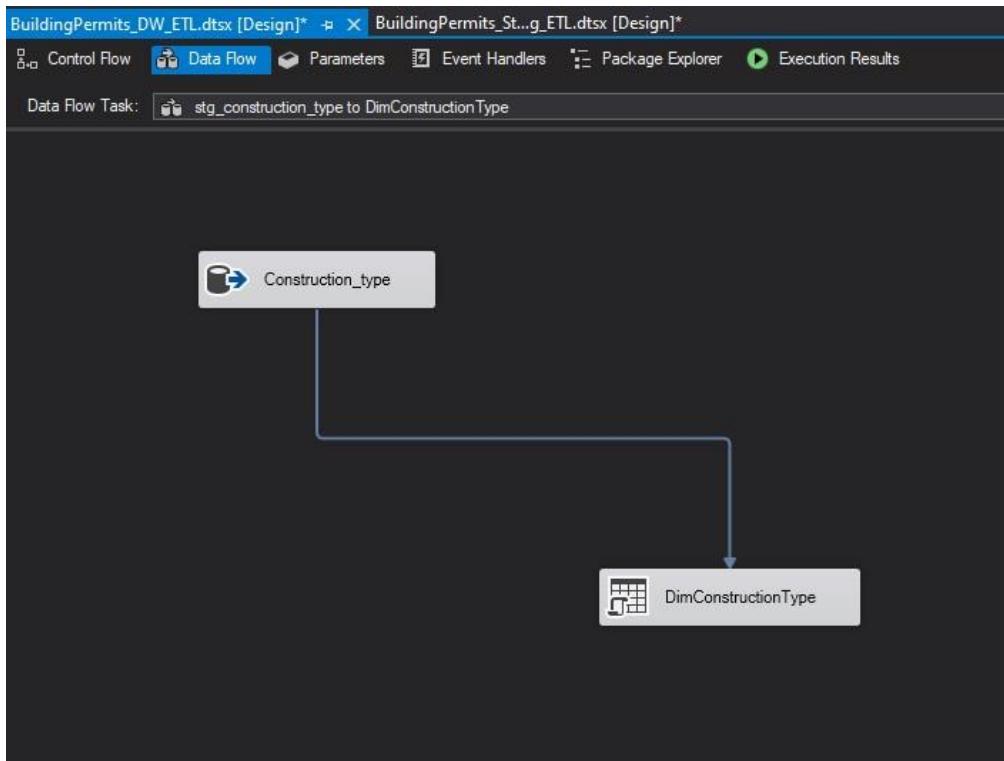
2. stg_permit_type to DimPermitType

I created the data flow task for stg_permit_type to DimPermitType



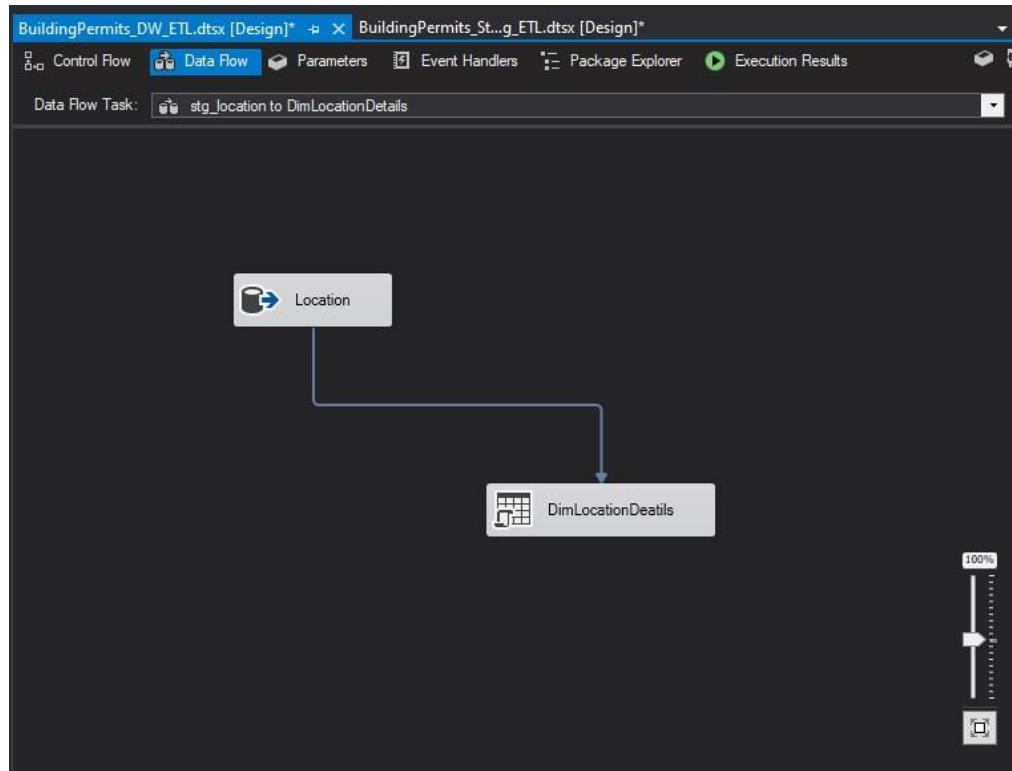
3. stg_construction_type to DimConstructionType

After creating the data flow task for stg_permit_type to DimPermitType ,I designed this task.



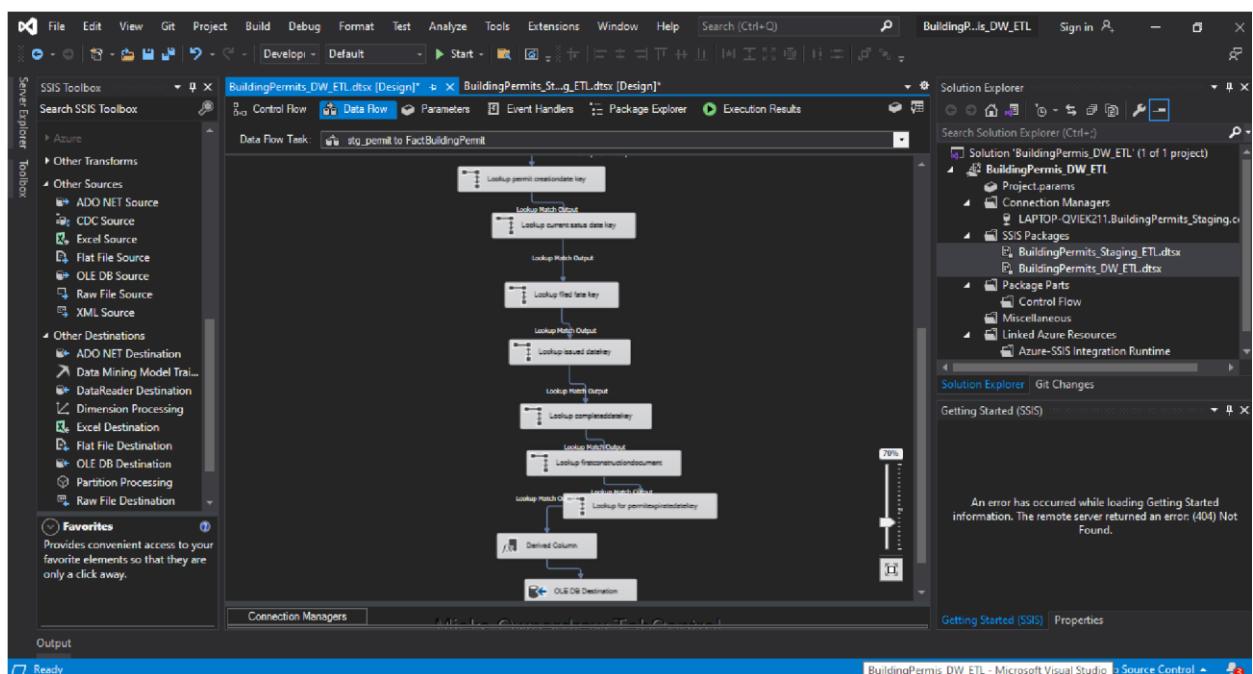
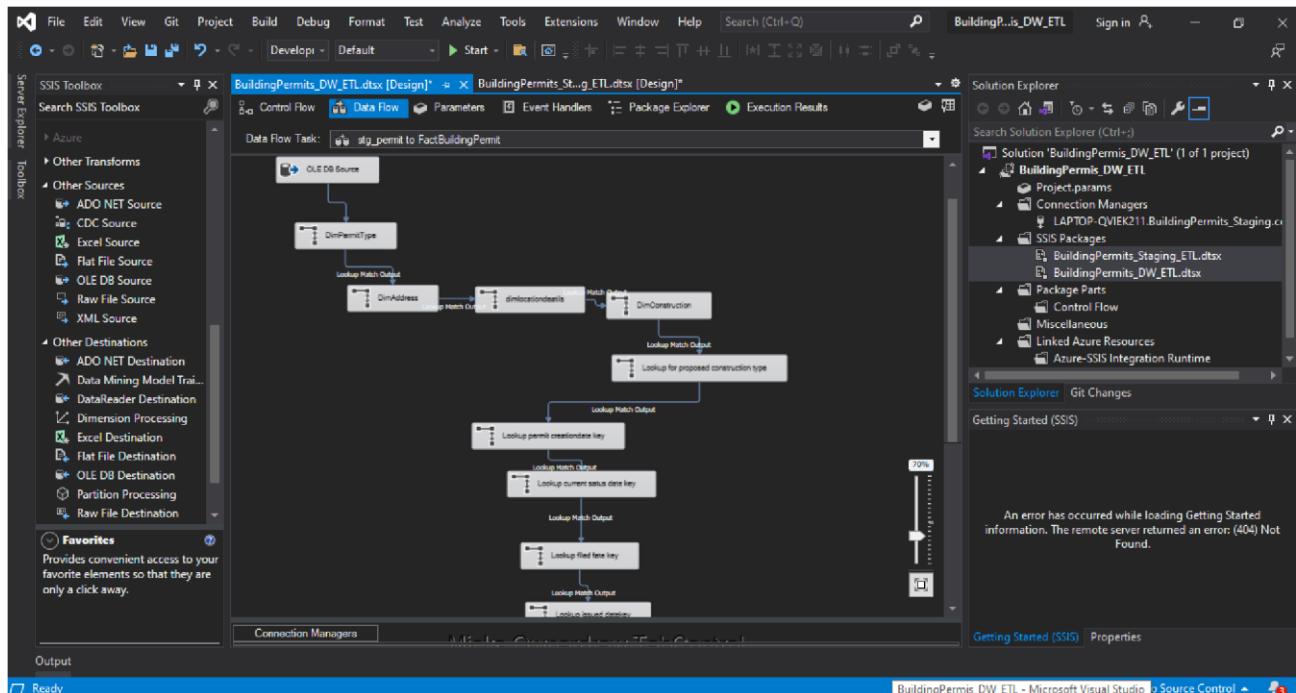
4. stg_location to DimLocationDetails

I created staging from stg_location to DimLocationDetails



5. stg_permit to FactBuildingPermit

I created staging from stg_permit to FactBuildingPermit



Stored Procedures

I have created three stored procedures in datawarehouse database as,

- UpdateDimConstructionType
- UpdateDimLocationDetailsId
- UpdateDimPermitType

UpdateDimConstructionType

```
CREATE PROCEDURE dbo.UpdateDimConstructionType
    @ConstructionTypeId nvarchar(8),
    @ConstructionTypeDescription nvarchar(50) AS BEGIN
        if not exists (select ConstructionTypeSK from dbo.dimConstructionType where
        AlternateConstructionTypeId = @ConstructionTypeId)    BEGIN
            insert into dbo.dimConstructionType (AlternateConstructionTypeId, ConstructionTypeDescription,
            InsertDate, ModifiedDate) values (@ConstructionTypeId, @ConstructionTypeDescription, GETDATE(),
            GETDATE())
        END;
        if exists (select ConstructionTypeSK from dbo.dimConstructionType where
        AlternateConstructionTypeId = @ConstructionTypeId)    BEGIN
            update dbo.dimConstructionType set ConstructionTypeDescription =
            @ConstructionTypeDescription, ModifiedDate = GETDATE() where AlternateConstructionTypeId =
            @ConstructionTypeId
        END;
    END;
```

```
EXEC dbo.UpdateDimConstructionType ?, ?
```

UpdateDimLocationDetailsId

```
CREATE PROCEDURE dbo.UpdateDimLocationDetailsId
    @LocationDetailsId nvarchar(8),
    @Neighborhoods nvarchar(50),
    @Zipcode numeric,
    @Location nvarchar(50) AS BEGIN
        if not exists (select LocationDetailsSK from dbo.dimLocationDetailsId where
        AlternateLocationDetailsId = @LocationDetailsId)    BEGIN
            insert into dbo.dimLocationDetailsId (AlternateLocationDetailsId,
            Neighborhoods,Zipcode,Location , InsertDate, ModifiedDate) values
            (@LocationDetailsId,@Neighborhoods,@Zipcode,@Location, GETDATE(), GETDATE())
        END;
        if exists (select LocationDetailsSK from dbo.dimLocationDetailsId where
        AlternateLocationDetailsId = @LocationDetailsId)    BEGIN
            update dbo.dimLocationDetailsId set Neighborhoods =
            @Neighborhoods,Zipcode=@Zipcode ,Location=@Location, ModifiedDate = GETDATE() where
```

```
AlternateLocationDetailsId = @LocationDetailsId
```

```
END;
```

```
END;
```

```
EXEC dbo.UpdateDimLocationDetailsId ?, ?, ?, ?
```

UpdateDimPermitType

```
CREATE PROCEDURE dbo.UpdateDimPermitType
```

```
@PermitTypeId nvarchar(8),
```

```
@PermitTypeDefinition nvarchar(50) AS BEGIN
```

```
    if not exists (select PermitTypeSK from dbo.dimPermitType where AlternatePermitTypeId =  
@PermitTypeId)
```

```
BEGIN
```

```
    insert into dbo.dimPermitType (AlternatePermitTypeId, PermitTypeDefinition, InsertDate,  
ModifiedDate) values (@PermitTypeId, @PermitTypeDefinition, GETDATE(), GETDATE())
```

```
END;
```

```
    if exists (select PermitTypeSK from dbo.dimPermitType where AlternatePermitTypeId =  
@PermitTypeId)
```

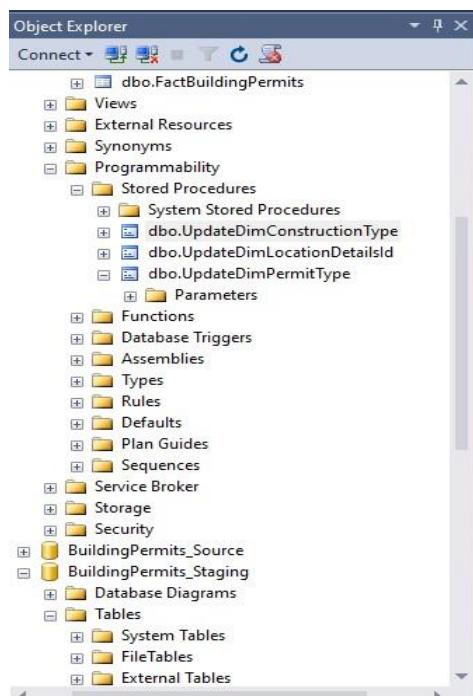
```
BEGIN
```

```
        update dbo.dimPermitType set PermitTypeDefinition = @PermitTypeDefinition,  
ModifiedDate = GETDATE() where AlternatePermitTypeId = @PermitTypeId
```

```
END;
```

```
END;
```

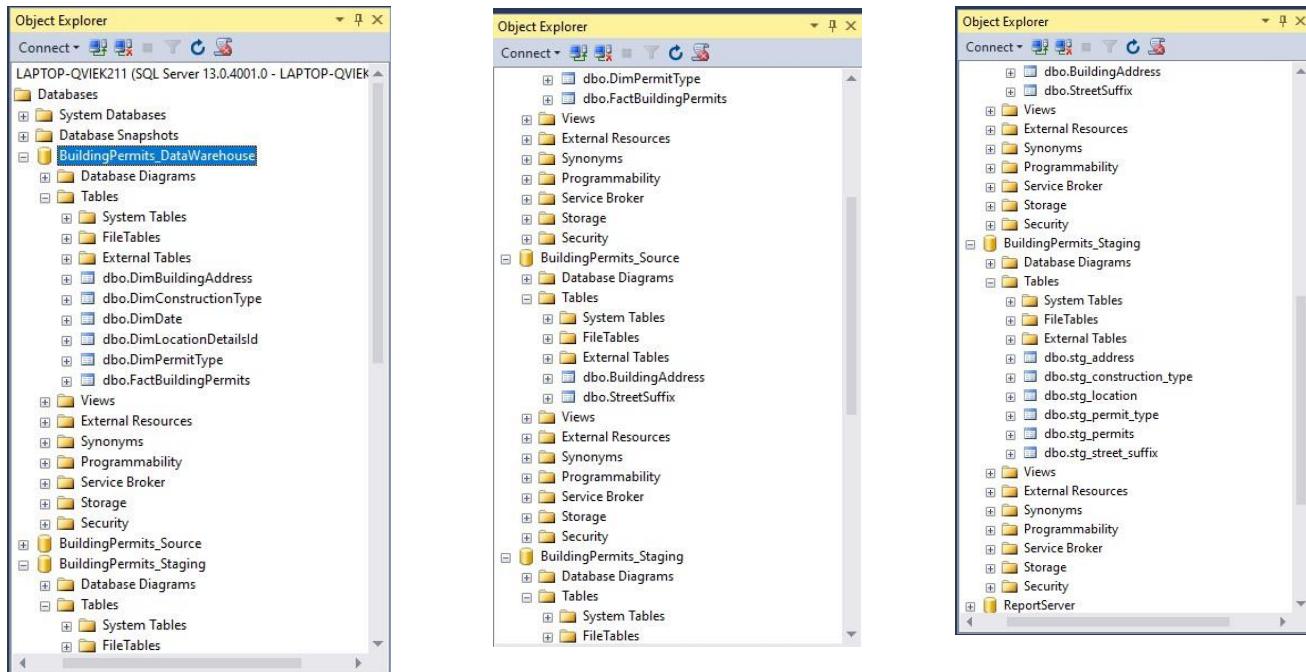
```
EXEC dbo.UpdateDimPermitType ?, ?
```



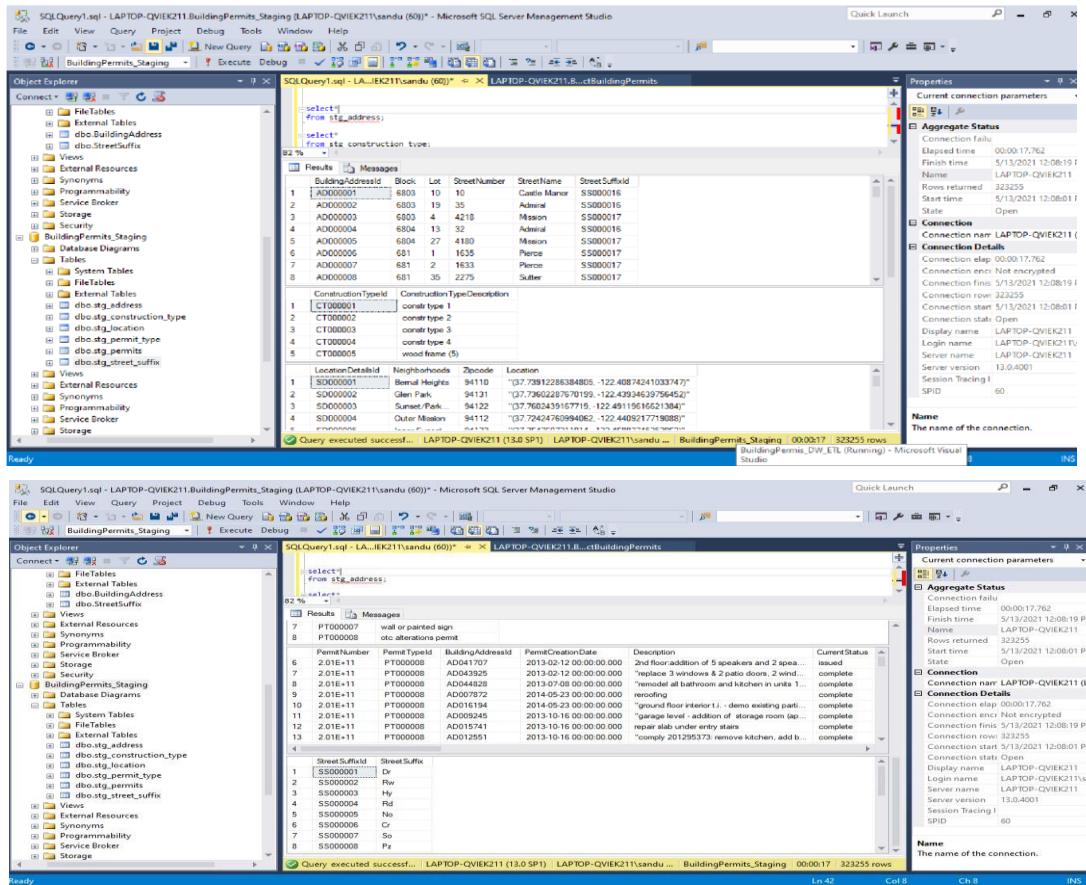
BuildingPermits DataWarehouse

BuildingPermits Source

BuildingPermits Staging



Tables of the BuildingPermits_Staging



SQLQuery1.sql - LAPTOP-QVIEK211.BuildingPermits_Staging (LAPTOP-QVIEK211\sandu (60)) - Microsoft SQL Server Management Studio

```

File Edit View Query Project Debug Tools Window Help
File Edit View Query Project Debug Tools Window Help
New Query Execute Debug
BuildingPermits_Staging
Object Explorer
Results Messages
1 CT000001 contrtype 1
2 CT000002 contrtype 2
3 CT000003 contrtype 3
4 CT000004 contrtype 4
5 CT000005 wood frame (5)

LocationDetailId Neighborhood Zipcode Location
5 SD000005 Inner Sunset 94122 "(37.7647607211014, -122.46883745252963)"
6 SD000006 Twin Peaks 94131 "(37.753741800710434, -122.4545616784311)"
7 SD000007 Twin Peaks 94114 "(37.7538860599708, -122.4496815267394)"
8 SD000008 Pacific Heights 94107 "(37.7531857910464, -122.431988809795)"
9 SD000009 Russian Hill 94118 "(37.7531857910464, -122.431988809795)"
10 SD000010 Outer Richmond 94118 "(37.7600561744591, -122.4710920889244)"
11 SD000011 Inner Richmond 94118 "(37.785494546323705, -122.465456915974)"
12 SD000012 Hayes Valley 94117 "(37.7798512071194, -122.4299156425343)"

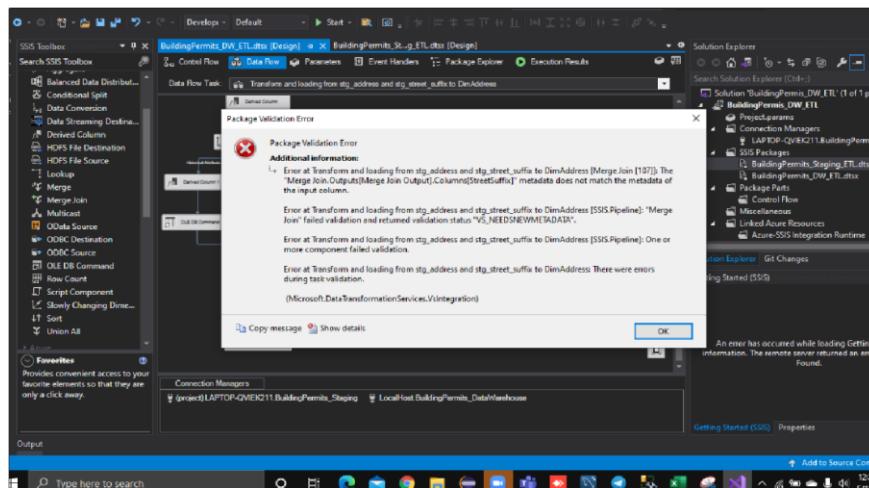
PermitTypeId Permit Type Definition
1 PT000001 new construction
2 PT000002 new construction wood frame
3 PT000003 additions alterations or repairs
4 PT000004 sign - erect
5 PT000005 grade or quarry fill or exc...
6 PT000006 demolitions

```

Query executed successfully... | LAPTOP-QVIEK211 (13.0 SP1) | LAPTOP-QVIEK211\sandu ... | BuildingPermits_Staging | 00:00:17 | 323255 rows

En 42 Col 8 CH 8 INS

These errors occurred while I was designing the datawarehouse



SQLQuery1.sql - LAPTOP-QVIEK211.BuildingPermits_DataWarehouse (LAPTOP-QVIEK211\sandu (60)) - Microsoft SQL Server Management Studio

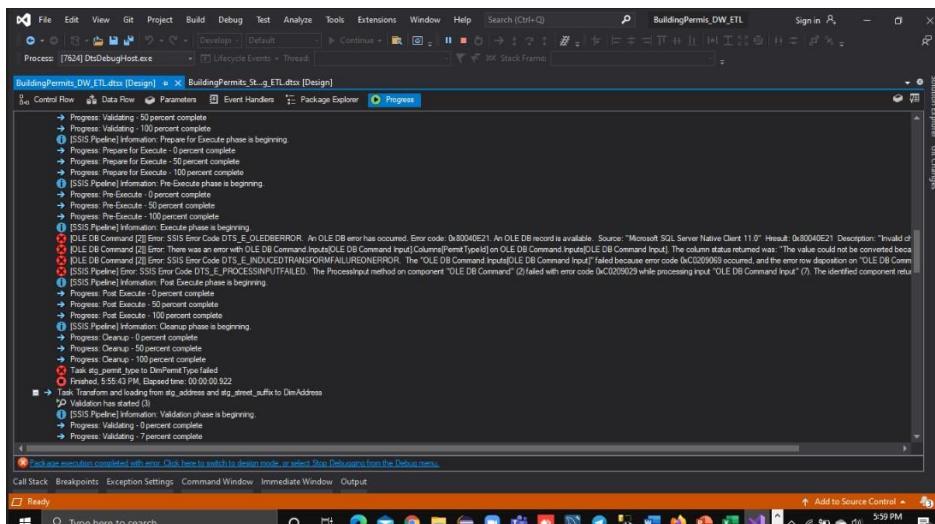
```

File Edit View Query Project Debug Tools Window Help
File Edit View Query Project Debug Tools Window Help
New Query Execute Debug
BuildingPermits_DataWarehouse
Object Explorer
Results Messages
100 BuildingPermits_DataWarehouse
select count(*) AS 'Dim loc' from DimLocationDetail15
101 BuildingPermits_Staging
select count(*) AS 'Stg permit type' from stg_permit_type
102 BuildingPermits_DataWarehouse
select count(*) AS 'Dim Per Type' from DimPermitType
103 BuildingPermits_Staging
select count(*) AS 'Stg con type' from stg_construction_type
104 BuildingPermits_DataWarehouse
select count(*) AS 'Dim con type' from DimConstructionType
105 BuildingPermits_Staging
select count(*) AS 'Stg add' from stg_address
106 BuildingPermits_DataWarehouse
select count(*) AS 'Dim add' from DimAddress
107 BuildingPermits_Staging
select count(*) AS 'Stg permits' from stg_permits
108 BuildingPermits_DataWarehouse
select count(*) AS 'Fact Permits' from FactBuildingPermits
109 BuildingPermits_Staging
truncate table [dbo].[stg_construction_type];
truncate table [dbo].[stg_location];
truncate table [dbo].[stg_permit_type];
truncate table [dbo].[stg_permits];
truncate table [dbo].[stg_street_suffix];
110 BuildingPermits_DataWarehouse;
truncate table [dbo].[factbuildingpermits];
truncate table [dbo].[dimconstructiontype];
truncate table [dbo].[dimlocationdetail15];
truncate table [dbo].[dimconstructiontype];

```

Query executed successfully... | LAPTOP-QVIEK211 (13.0 SP1) | LAPTOP-QVIEK211\sandu ... | BuildingPermits_DataWa... | 00:00:04 | 11 rows

Ready Add to Source Control



After the Execution of the full packages

We created the datawarehouse successfully

The screenshot shows the Microsoft SQL Server Management Studio interface. The title bar indicates the connection is to 'BuildingPermits_DataWarehouse' on 'LAPTOP-QVIEK211'. The left pane displays the Object Explorer with the 'BuildingPermits_Staging' database selected. The central pane shows the results of a query named 'SQLQuery1.sql' with the following output:

	stg loc
1	57605

	dim loc
1	57605

	stg permit typ
1	8

	Dim Per Type
1	8

	stg con type
1	5

	Dim con type
1	5

	stg suff
1	21

	stg add
1	76999

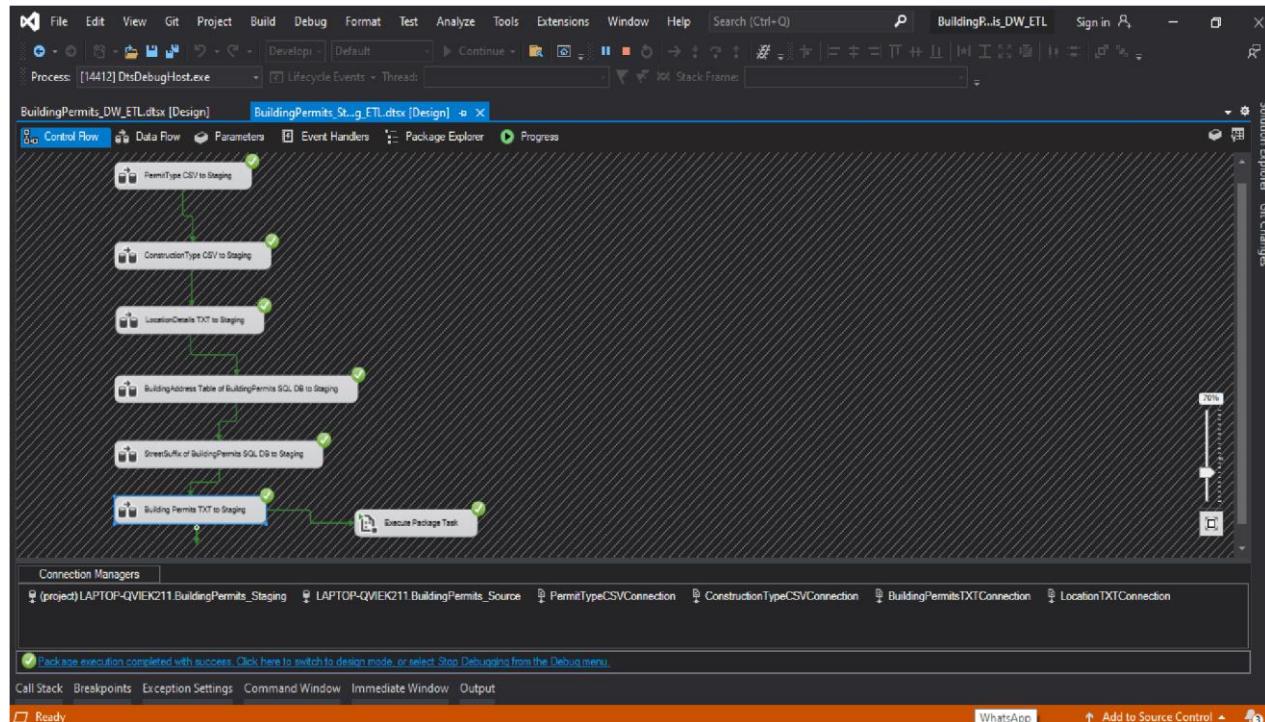
	Dim Add
1	76999

	stg permits
1	188617

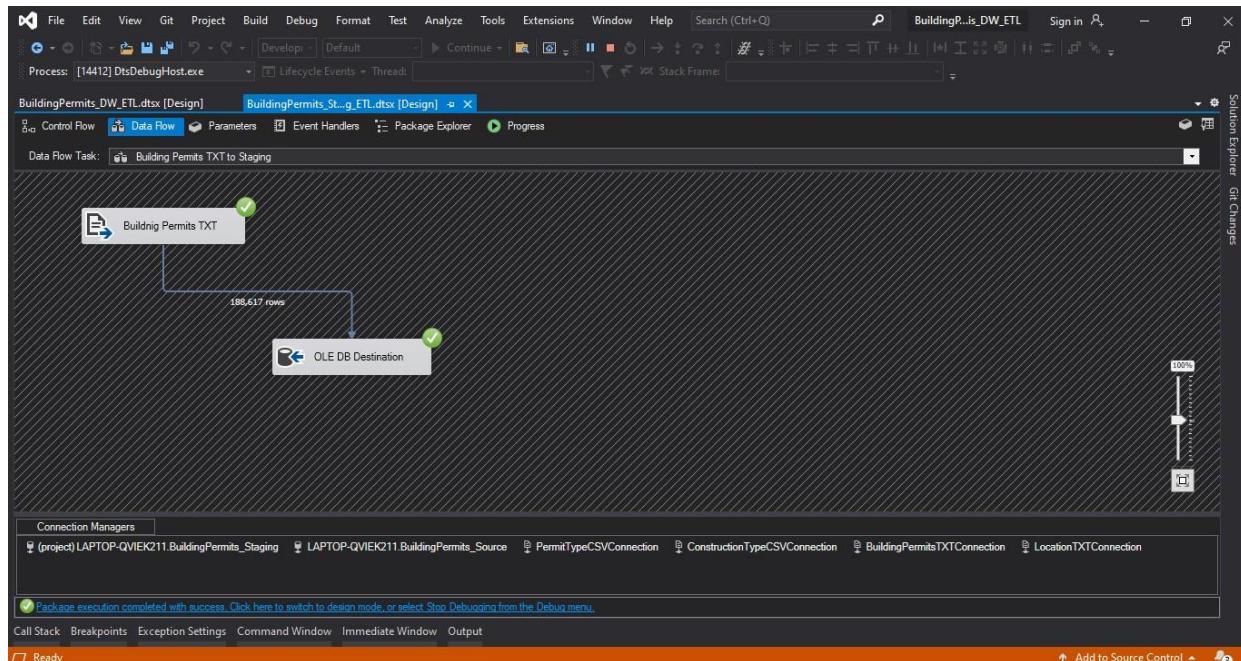
	Fac Permits
1	188617

The status bar at the bottom shows 'Query executed successfully.' and the execution time '00:00:04 | 11 rows'.

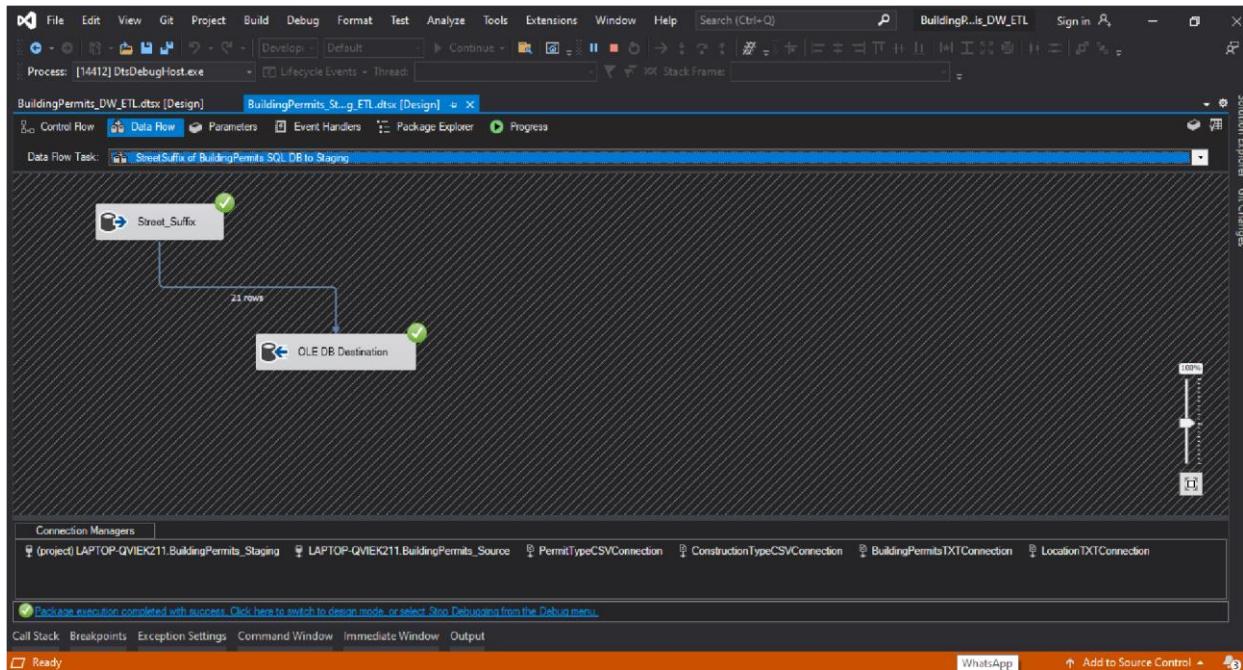
BuildingPermits Staging ETL.dtsx



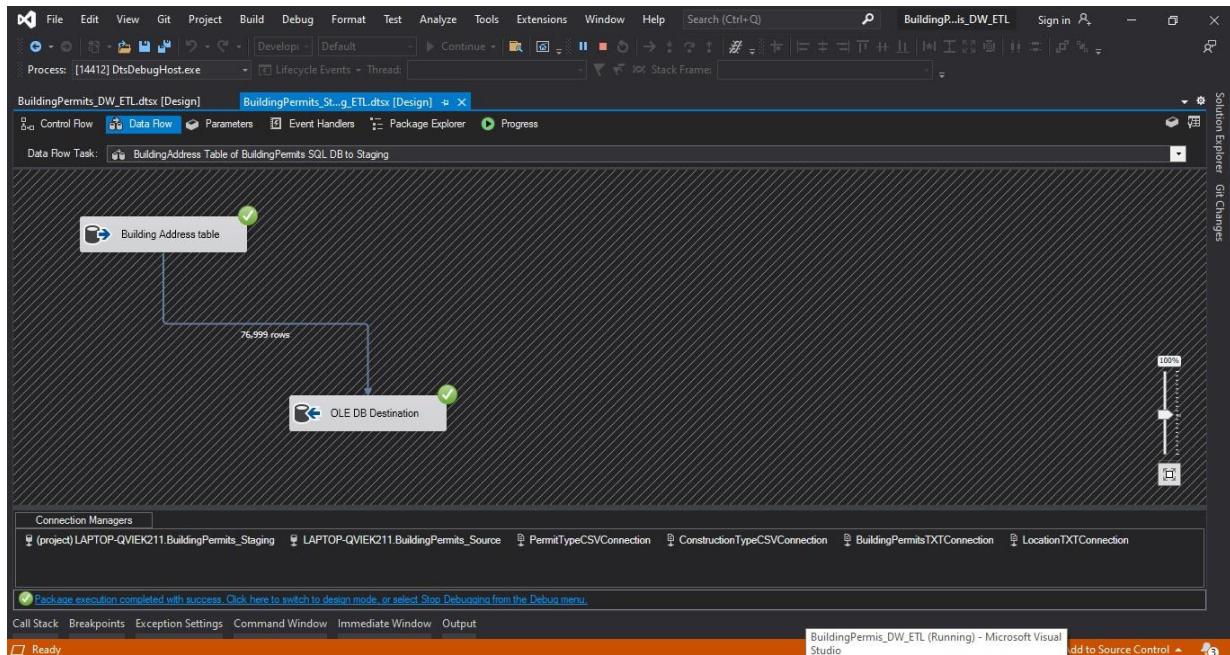
Building Permits txt to staging



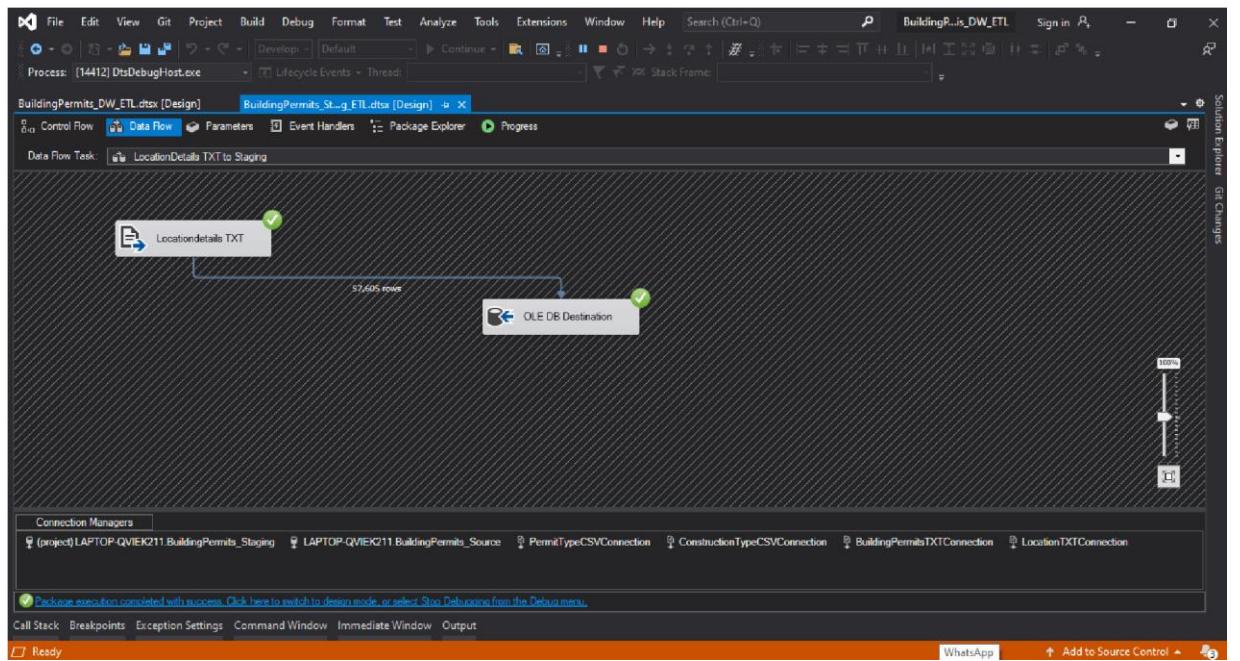
- StreetSuffix of Building Permits SQL DB to Staging



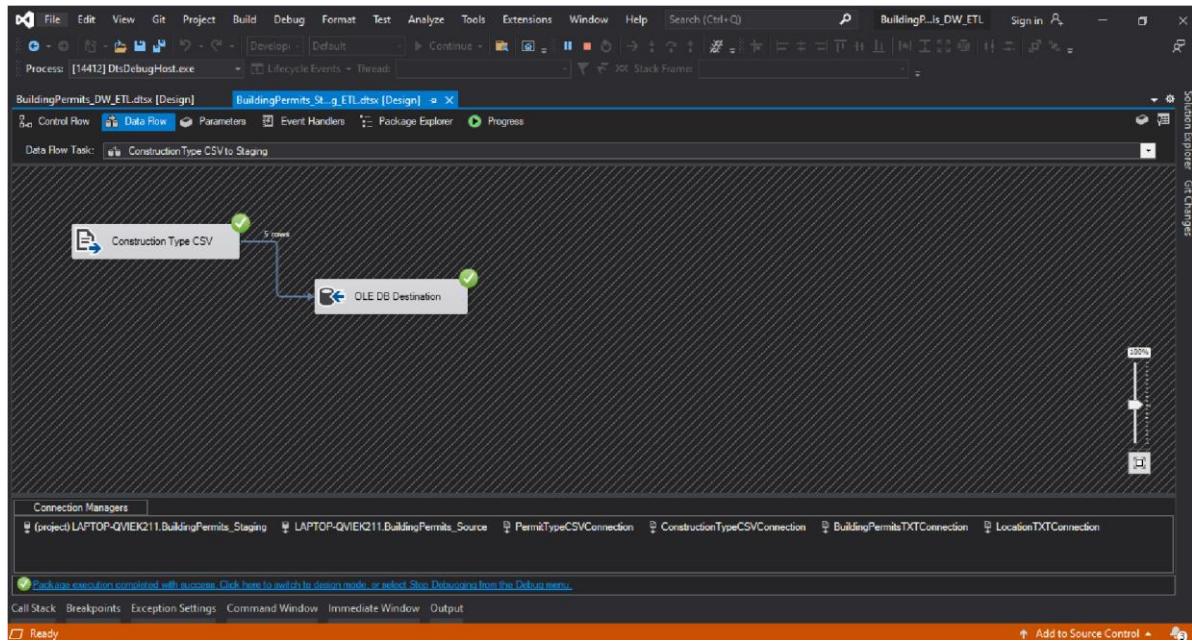
Building Address table of Building Permits SQL DB to Staging



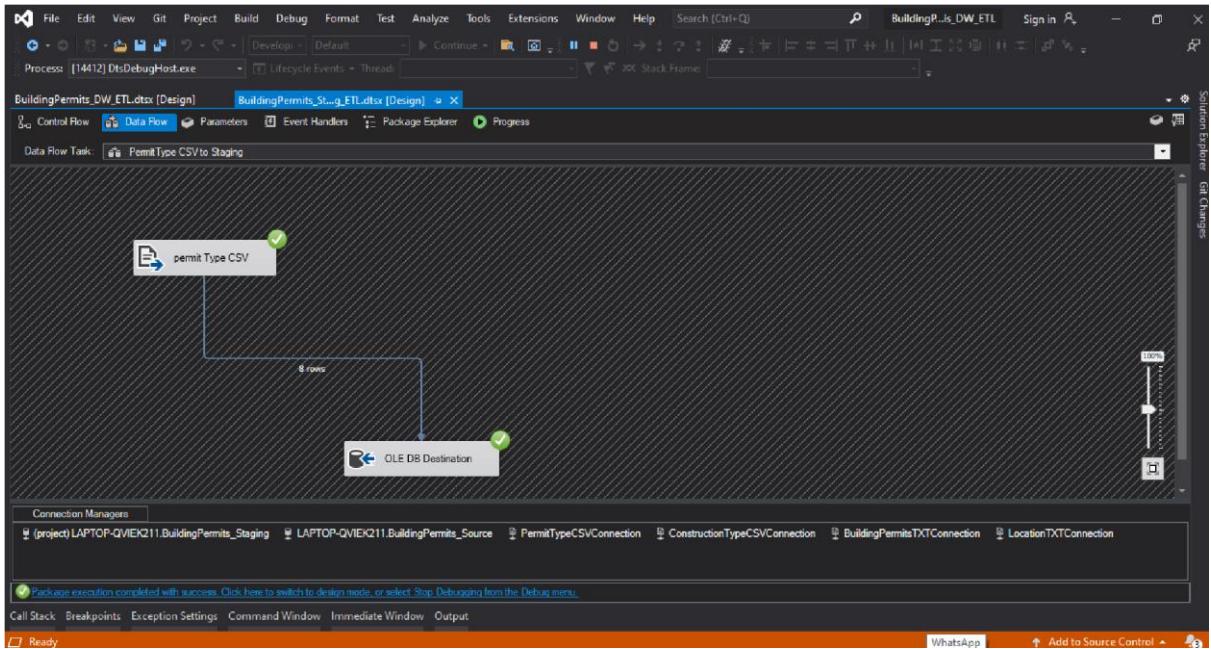
LocationFeatils txt to staging



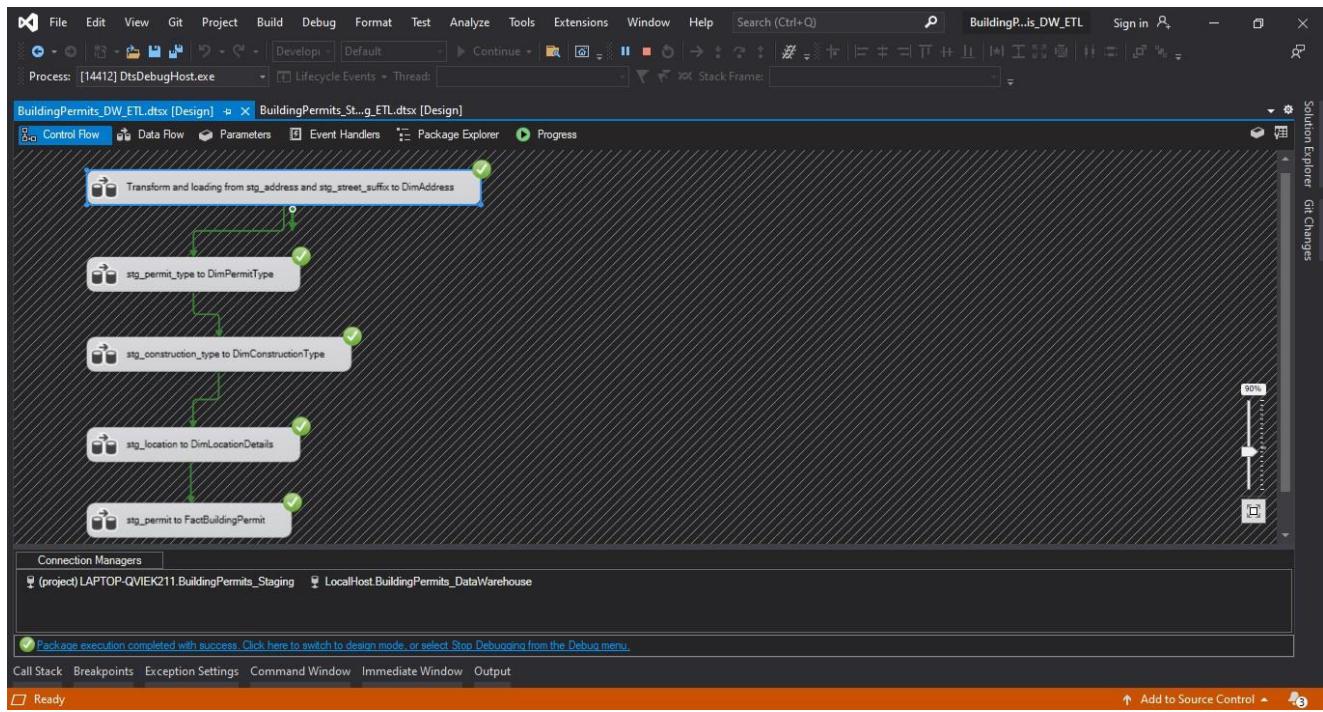
• ConstructionType csv to staging



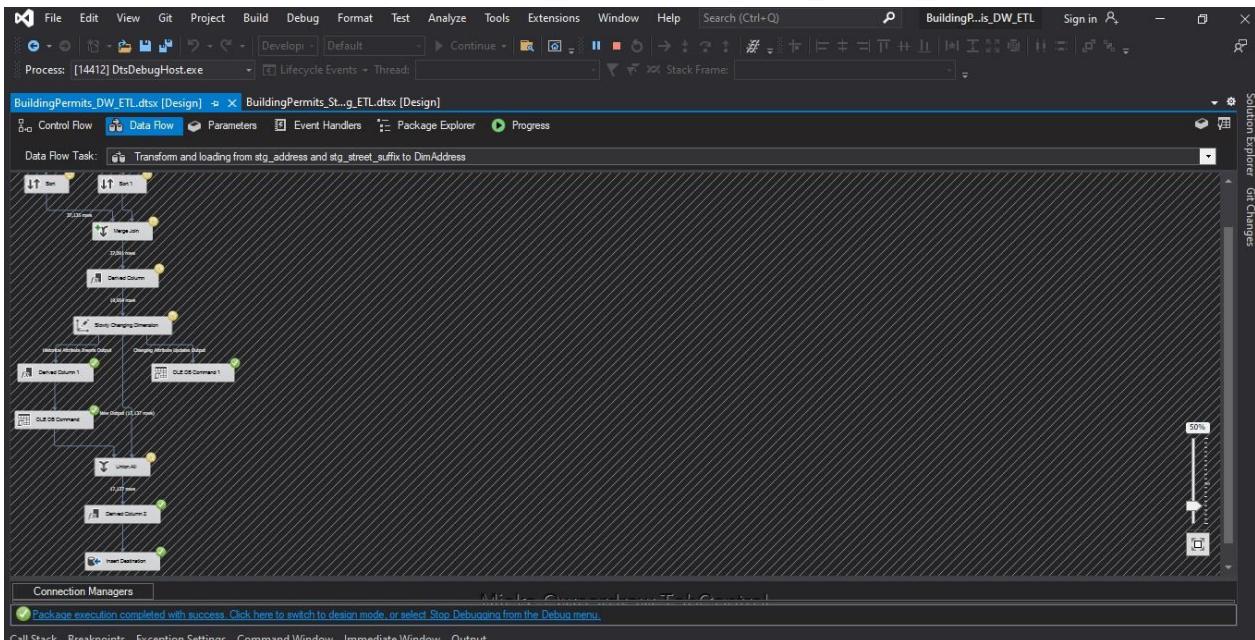
• PermitType csv to staging



• BuildingPermits_DWH_ETL.dtsx



- Transform and loading from stg_address and stg_street_suffix to Dimaddress



- Stg_permit_type to DimPermit type

