

# **Sri Lanka Institute of Information Technology**

## **IT3021 – Data Warehousing and Business Intelligence**

### **Assignment 01 Report**



**Student No: IT22060594**

**Name: Soyza W W K**

**Batch: Y3.S1 DS WE 01**

# Introduction:

This dataset is a modified OLTP dataset originally sourced from Kaggle that simulates the operations of a **Cannabis Dispensary**. It has been enhanced using AI to better reflect a real-world business scenario and structured to support analytical processing. The dataset includes a **fact table containing transactional order data** (such as order totals) and multiple **dimension tables** that describe products, product types, geographical information (states and cities), and brands. Though initially designed for online transaction processing (OLTP), the dataset has been adapted to support **OLAP operations and Power BI reporting**, making it suitable for exploring business insights such as sales performance by product and region, customer purchasing trends, and more.

## Dataset Selection

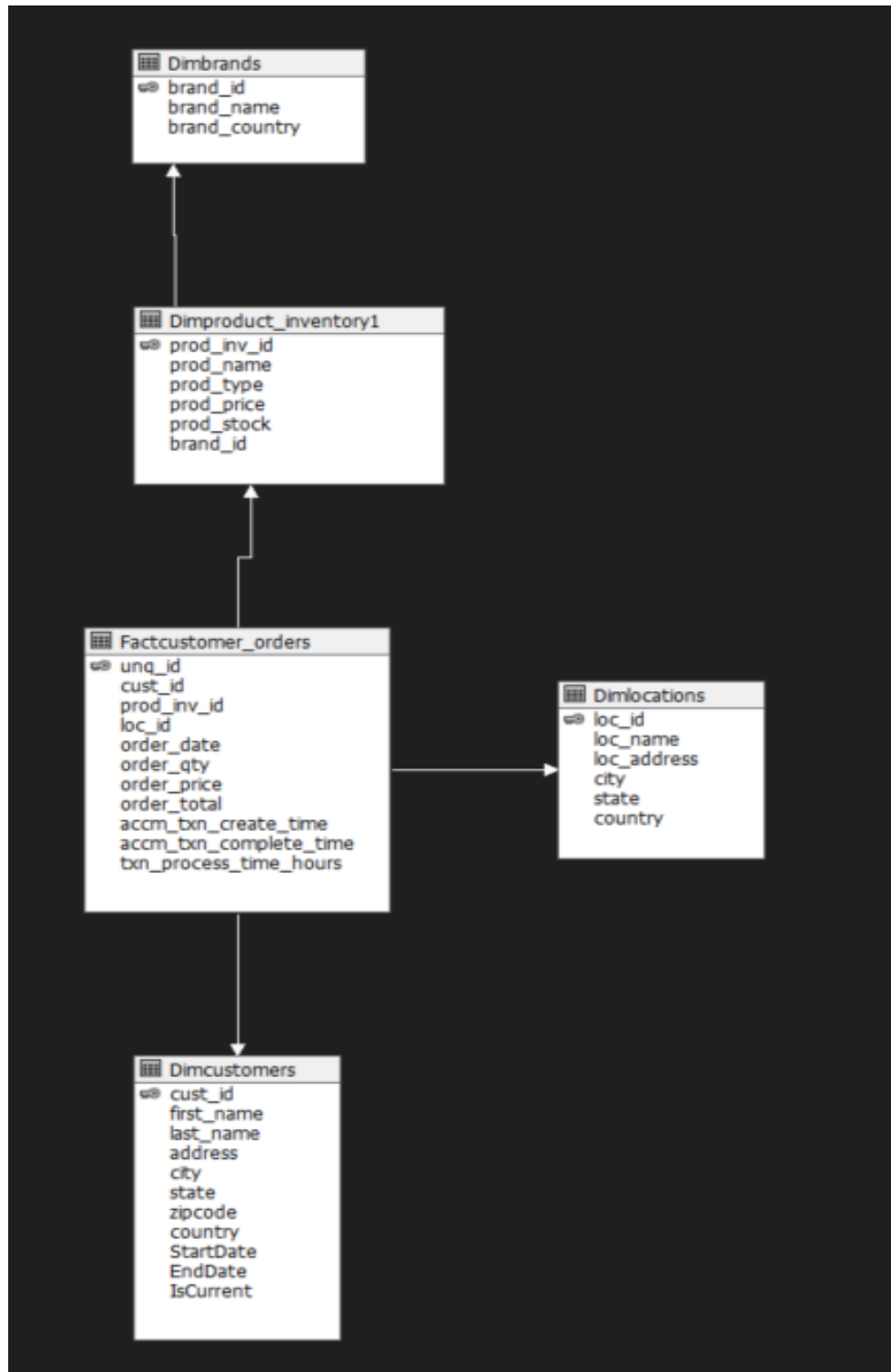
The selected dataset is the "Synthetic Cannabis Dispensary Dataset. It includes customer data, product inventory and type, customer orders, brand information. The dataset is well-suited for OLTP scenarios and suitable for DWBI use cases based on its relational schema and sufficient data variety.

The screenshot shows the Kaggle interface for the "Synthetic Cannabis Dispensary Database" dataset. The left sidebar contains navigation links: Home, Competitions, Datasets (selected), Models, Code, Discussions, Learn, and More. The main content area displays the dataset title "Synthetic Cannabis Dispensary Database" with tabs for Data Card, Code (3), Discussion (0), and Suggestions (0). Below the tabs, there's a section for "SyntheticDispoMetaData.json (4.98 kB)" with a download icon. The "About this file" section shows the file is a "json Metadata". A JSON snippet is visible: 

```
{
  "root": [
    {
      "title": "string \"brands\"
      "creator": "string \"Adam Palmouist\"
    }
  ]
}
```

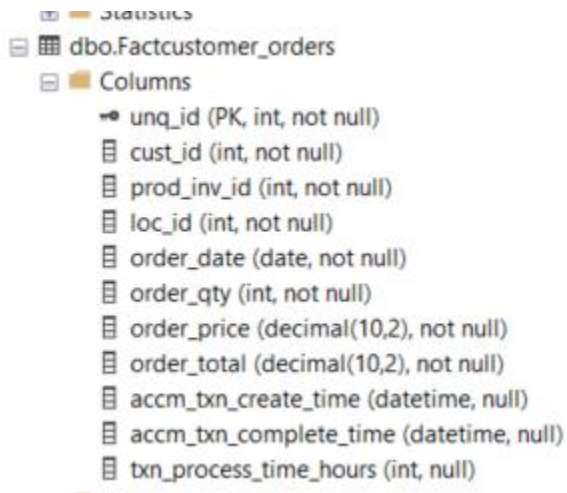
 On the right, there's a "Data Explorer" section showing a tree view of the dataset files: SyntheticDispoMetaData.json, brands.csv, customer.csv, customerOrders.csv, locations.csv, productInventory.csv, productType.csv, stateReg.csv, and strains.csv. The top right of the page has "Sign In" and "Register" buttons.

# Relationships among Tables



# Preparation of data sources

## Fact Tables- Factcustomer\_orders Table

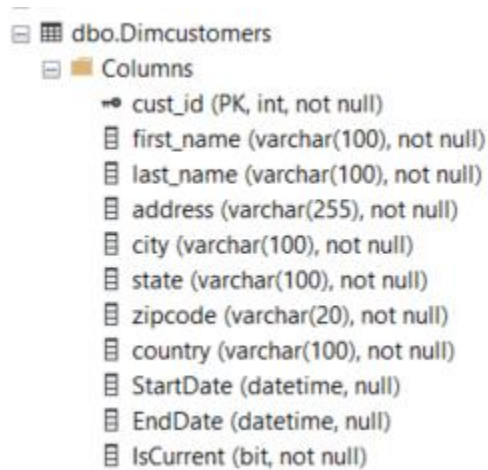


The screenshot shows the 'Columns' pane for the 'dbo.Factcustomer\_orders' table. The table has the following columns:

Column Name	Data Type	Nullable
unq_id	int	not null
cust_id	int	not null
prod_inv_id	int	not null
loc_id	int	not null
order_date	date	not null
order_qty	int	not null
order_price	decimal(10,2)	not null
order_total	decimal(10,2)	not null
accm_txn_create_time	datetime	null
accm_txn_complete_time	datetime	null
txn_process_time_hours	int	null

## Dimension Tables-

- Customer Table - Used in txt format



The screenshot shows the 'Columns' pane for the 'dbo.Dimcustomers' table. The table has the following columns:

Column Name	Data Type	Nullable
cust_id	int	not null
first_name	varchar(100)	not null
last_name	varchar(100)	not null
address	varchar(255)	not null
city	varchar(100)	not null
state	varchar(100)	not null
zipcode	varchar(20)	not null
country	varchar(100)	not null
StartDate	datetime	null
EndDate	datetime	null
IsCurrent	bit	not null

- Product Inventory Table- **Used in CSV format**

dbo.Dimproduct_inventory1
Columns
prod_inv_id (PK, int, not null)
prod_name (varchar(100), not null)
prod_type (varchar(100), not null)
prod_price (decimal(10,2), not null)
prod_stock (int, not null)
brand_id (int, not null)

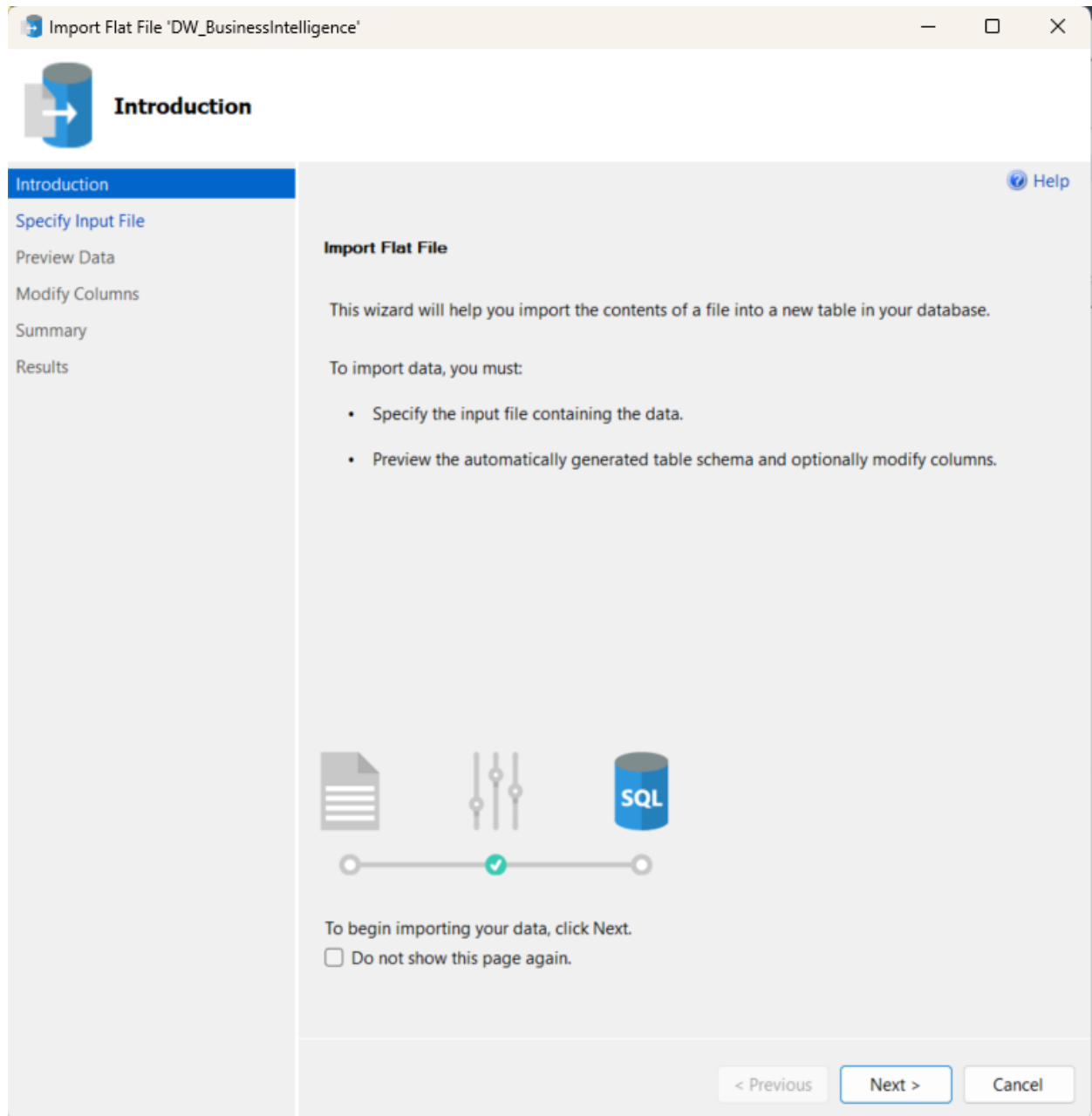
- Brands Table - **Used in CSV format**

Graph Tables
dbo.Dimbrands
Columns
brand_id (PK, int, not null)
brand_name (varchar(100), not null)
brand_country (varchar(100), not null)


- Locations Table - **Used in CSV format**

dbo.Dimlocations
Columns
loc_id (PK, int, not null)
loc_name (varchar(100), not null)
loc_address (varchar(255), not null)
city (varchar(100), not null)
state (varchar(100), not null)
country (varchar(100), not null)

- CSV files and txt files were imported using Import Flat Files



Import Flat File 'DW\_BusinessIntelligence'

 **Specify Input File**

Introduction

**Specify Input File**

Preview Data

Modify Columns

Summary

Results

Help

**Specify Input File**  
This operation will create a table from your input file.

Location of file to be imported

New table name:

Table schema:

< Previous

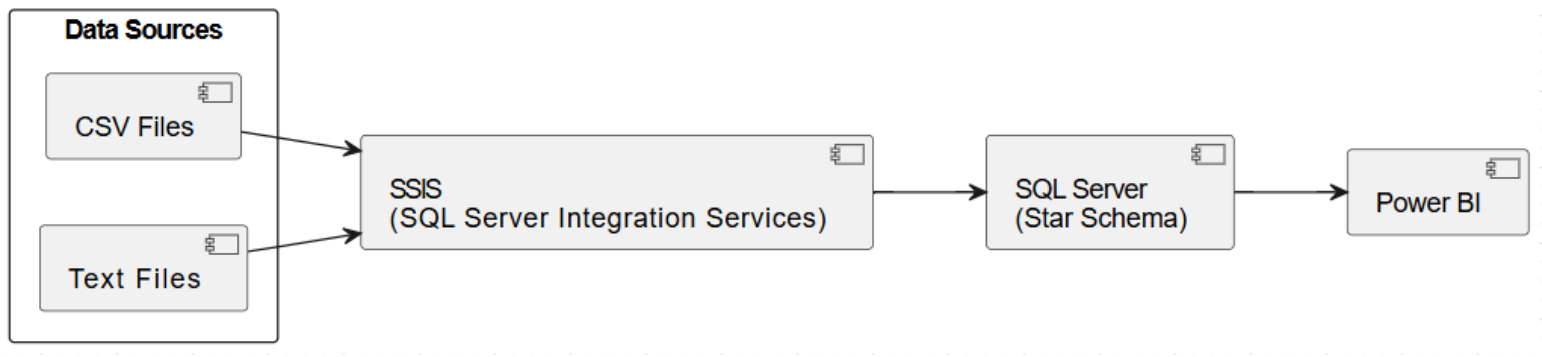
Next >

Cancel

# Solution architecture

A high-level solution architecture was designed to depict the flow of data from source to analytics:

- Data Sources: CSV and text files
- ETL Tool: SQL Server Integration Services (SSIS)
- Data Warehouse: SQL Server (Star schema)
- Reporting: Power BI



## Data Sources (.csv/.txt)



## SSIS (ETL Tool)



## SQL Server Data Warehouse

└─ Fact Table: FactCustomer\_Order

└─ Dimension Tables:

- Customer

- Brands

- Product Inventory



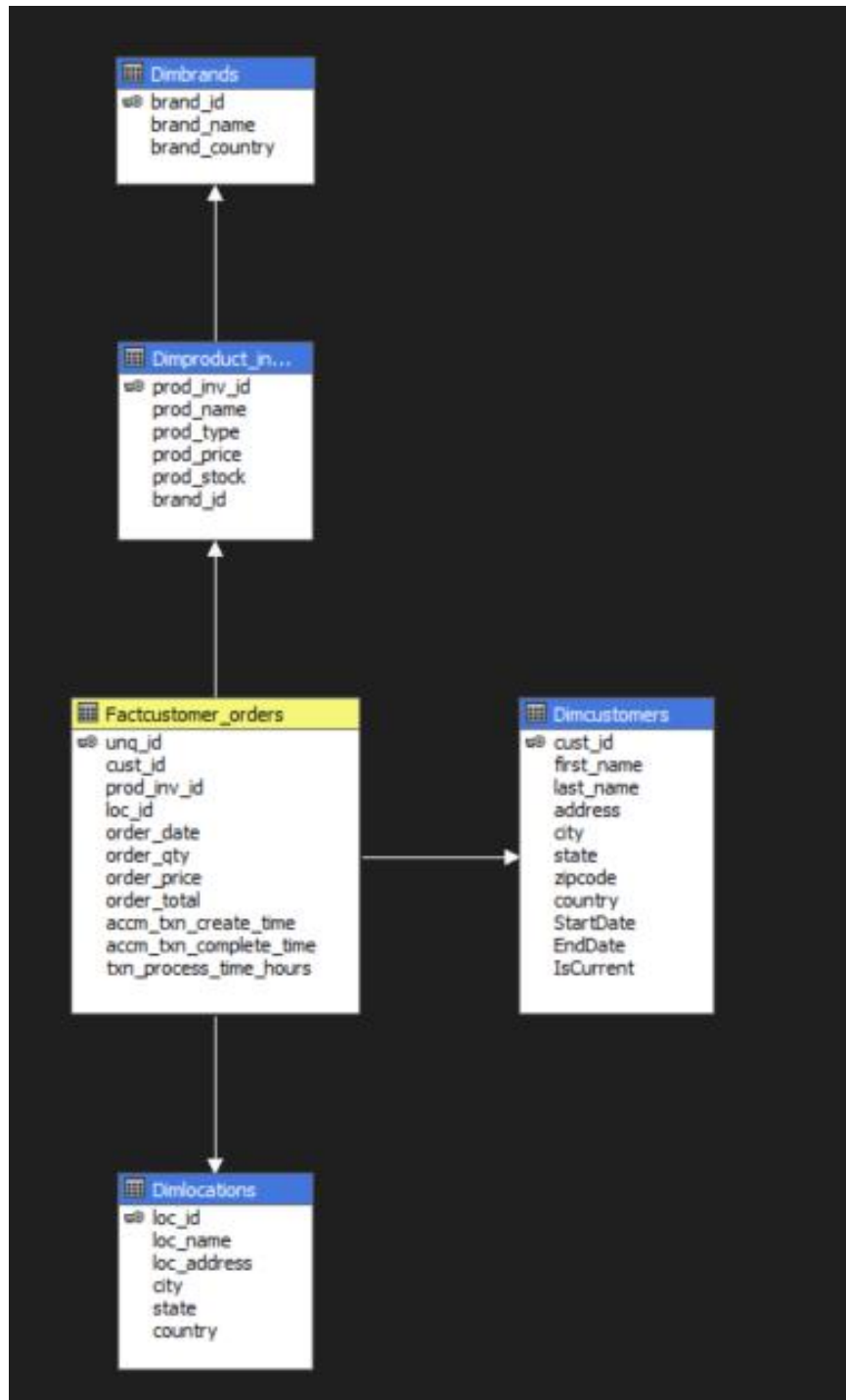
**- Locations**



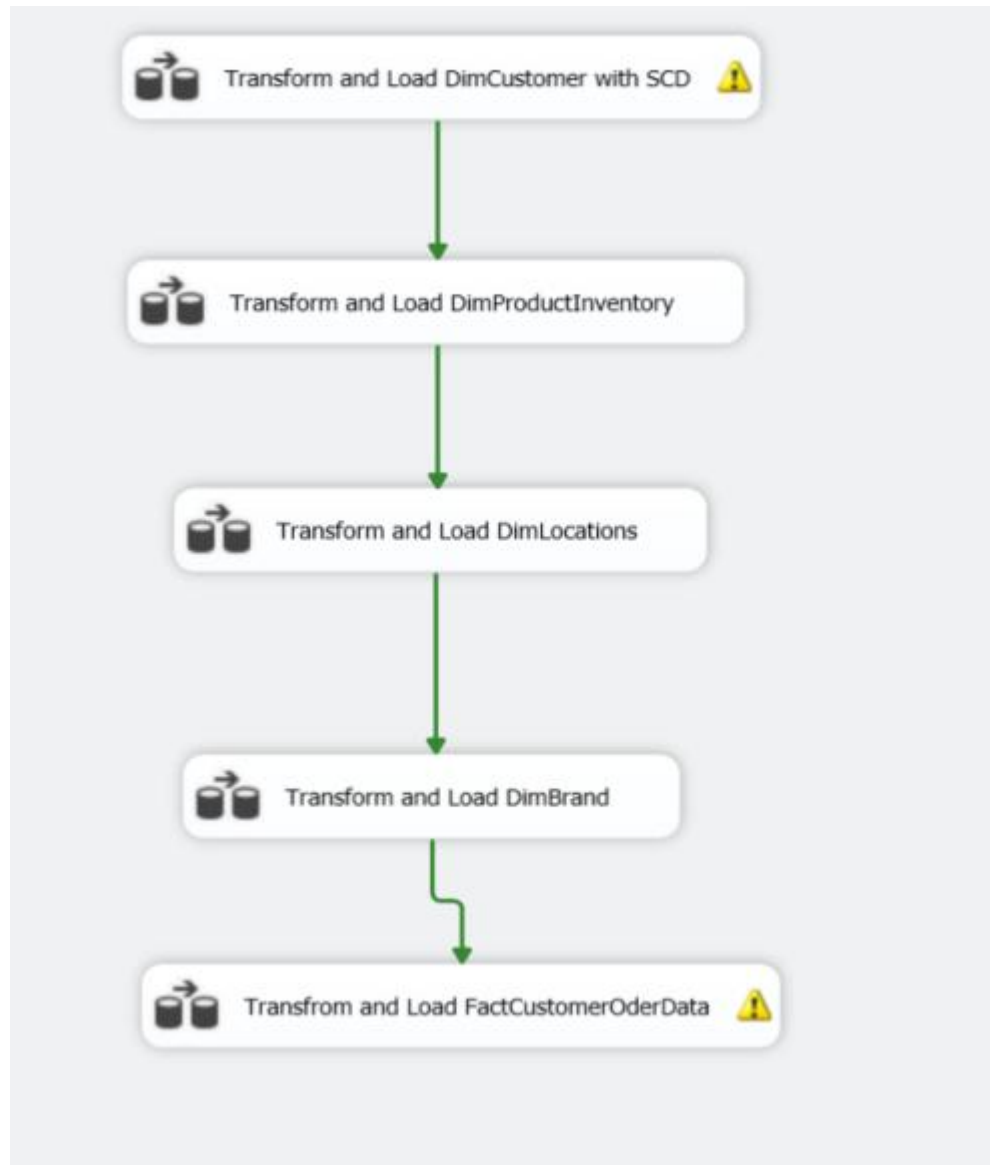
**Power BI (Visualization & Analysis)**

# Data Warehouse Design & Development

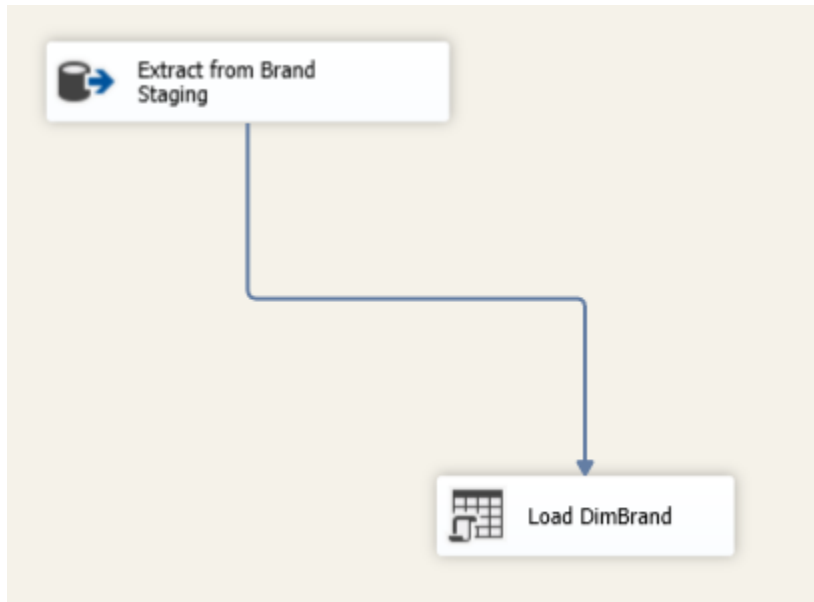
## Star Schema Design:



## ETL Process:

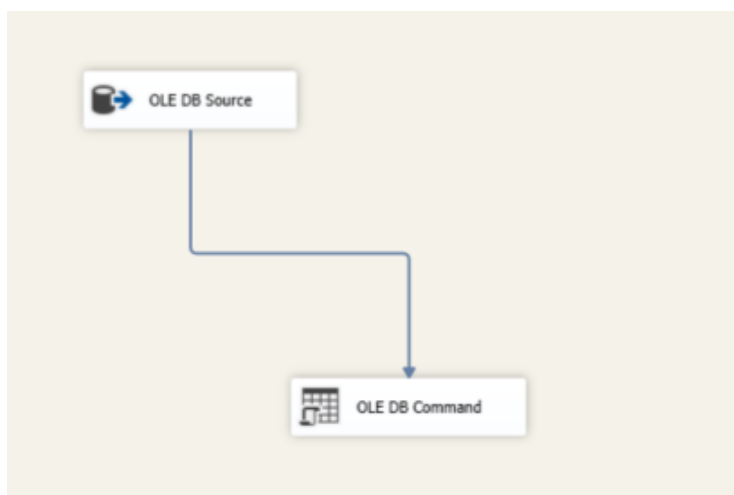


## 1. Transform and load Brand Data

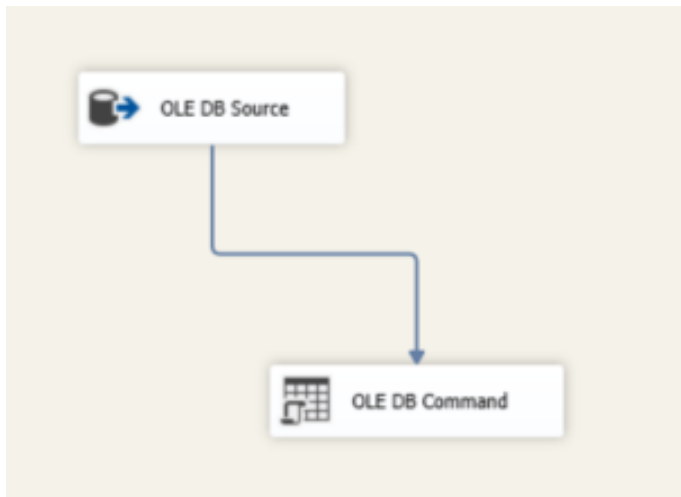


- Extracting brand-related data from data source
- Transforming the data (cleaning)
- Loading it into the target data warehouse

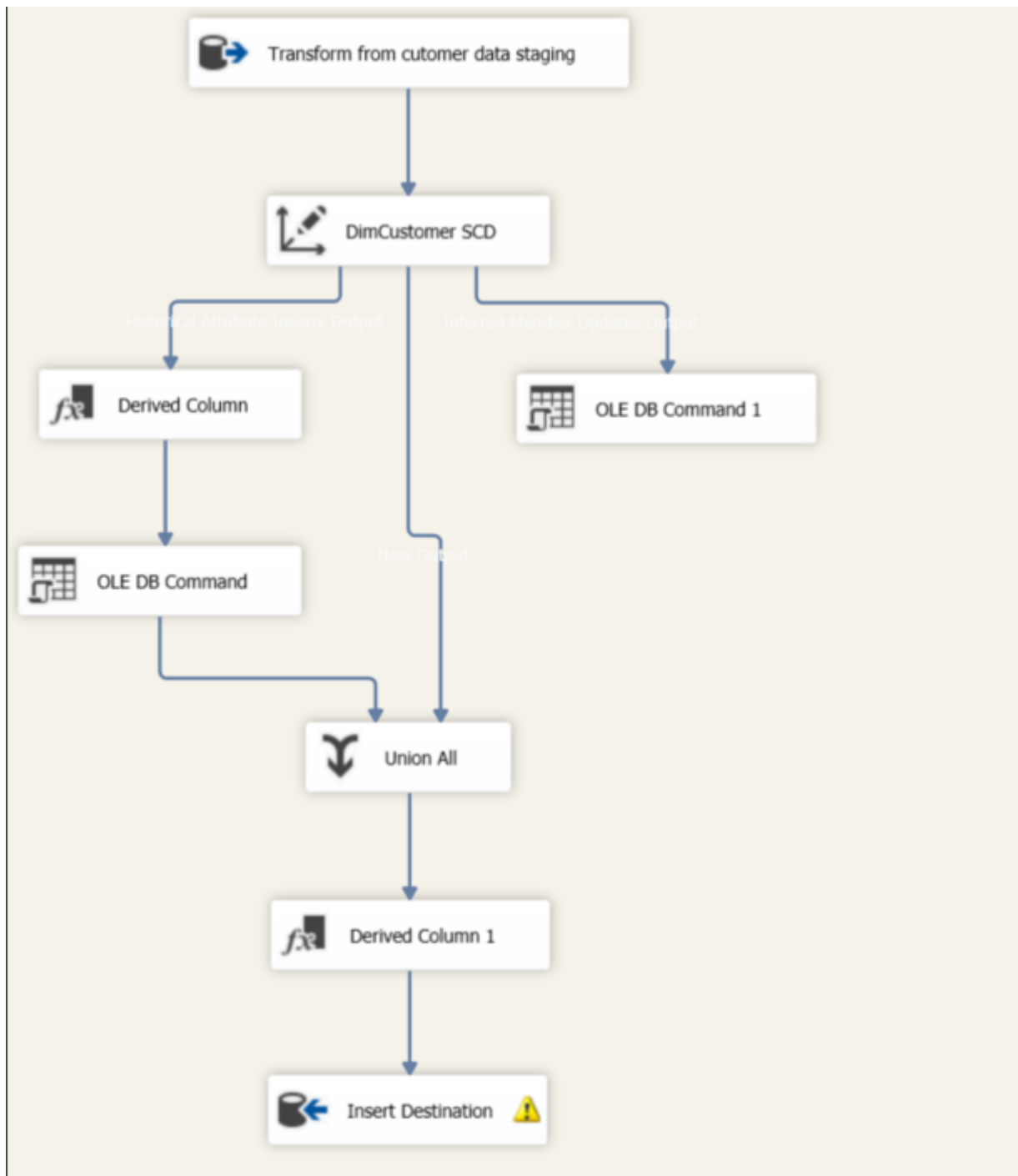
## 2. Transform and load location Data



### 3. Transform and load Product Inventory Data



#### 4. Transform and load Customer Data



## **Implementation of Slowly Changing Dimension (SCD) for Customer Data**

- In the data warehouse, the DimCustomer table was configured as a Slowly Changing Dimension (SCD) to track historical changes in customer address details while preserving fixed attributes such as first and last names. This approach ensures that whenever a customer's location (address, city, state, country, or zip code) changes, a new record is created with updated details, while the previous record is retained with an expiration timestamp. This design allows for accurate historical reporting, ensuring that past transactions are linked to the correct customer location at the time of purchase. By marking address-related fields as historical, we maintain a complete audit trail of customer movements, which is crucial for regional sales analysis, compliance, and customer behavior tracking. Meanwhile, name fields (first\_name, last\_name) were set as fixed attributes as they rarely change and overwriting them ensures consistency in customer identification across reports

Slowly Changing Dimension Wizard

### Select a Dimension Table and Keys

Select a dimension table to load and map columns in the transformation input to columns in the dimension

Connection manager:  
 DESKTOP-44D5U65.DW\_BusinessIntelligence New...

Table or view:  
 [dbo].[Dimcustomers] ▼

Input Columns	Dimension Columns	Key Type
address	address	Not a key column
city	city	Not a key column
country	country	Not a key column
cust_id	cust_id	Business key
	EndDate	
first_name	first_name	Not a key column
	IsCurrent	

Help < Back Next > Finish >>| Cancel

- Cust\_id is placed as business key

Slowly Changing Dimension Wizard

### Slowly Changing Dimension Columns

Manage the changes to column data in your slowly changing dimensions by setting the change type for

**Fixed Attribute**  
 Select this type when the value in a column should not change. Changes are treated as errors.

**Changing Attribute**  
 Select this type when changed values should overwrite existing values. This is a Type 1 change.

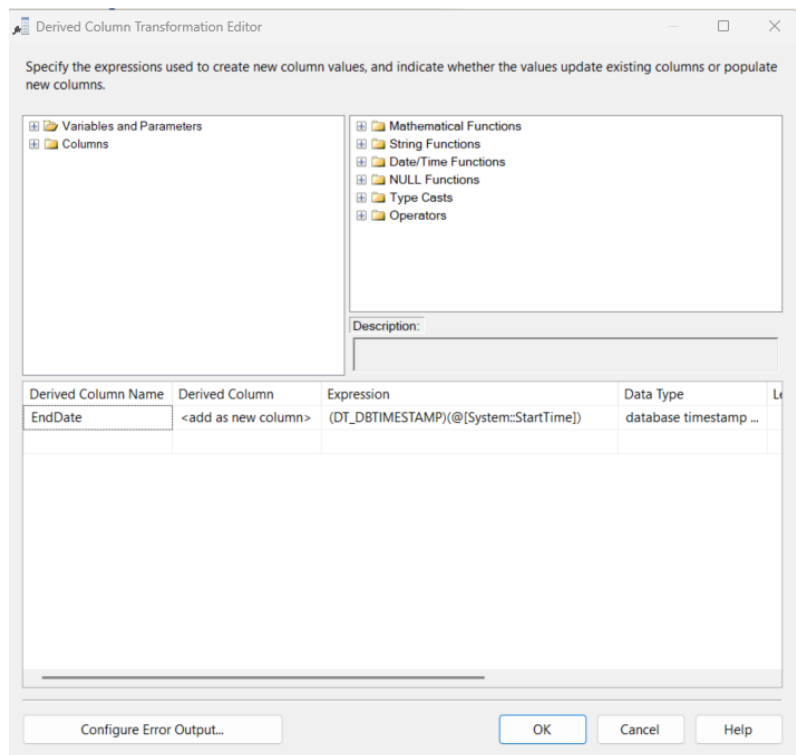
**Historical Attribute**  
 Select this type when changes in column values are saved in new records. Previous values are saved in records marked as outdated. This is a Type 2 change.

Select a change type for slowly changing dimension columns:

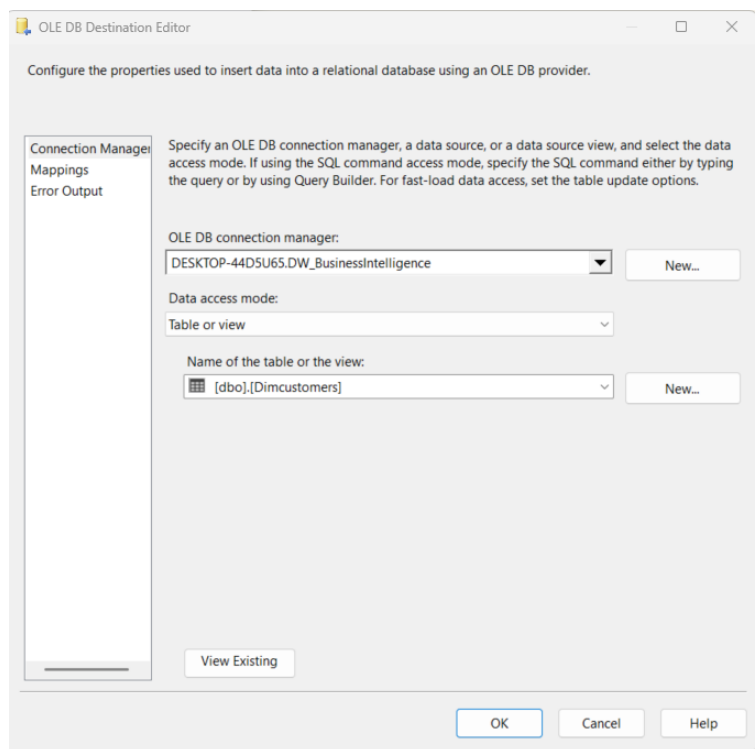
Dimension Columns	Change Type
address	Historical attribute
city	Historical attribute
country	Historical attribute
first_name	Fixed attribute
last_name	Fixed attribute
state	Historical attribute
zipcode	Historical attribute

Help < Back Next > Finish >>| Cancel

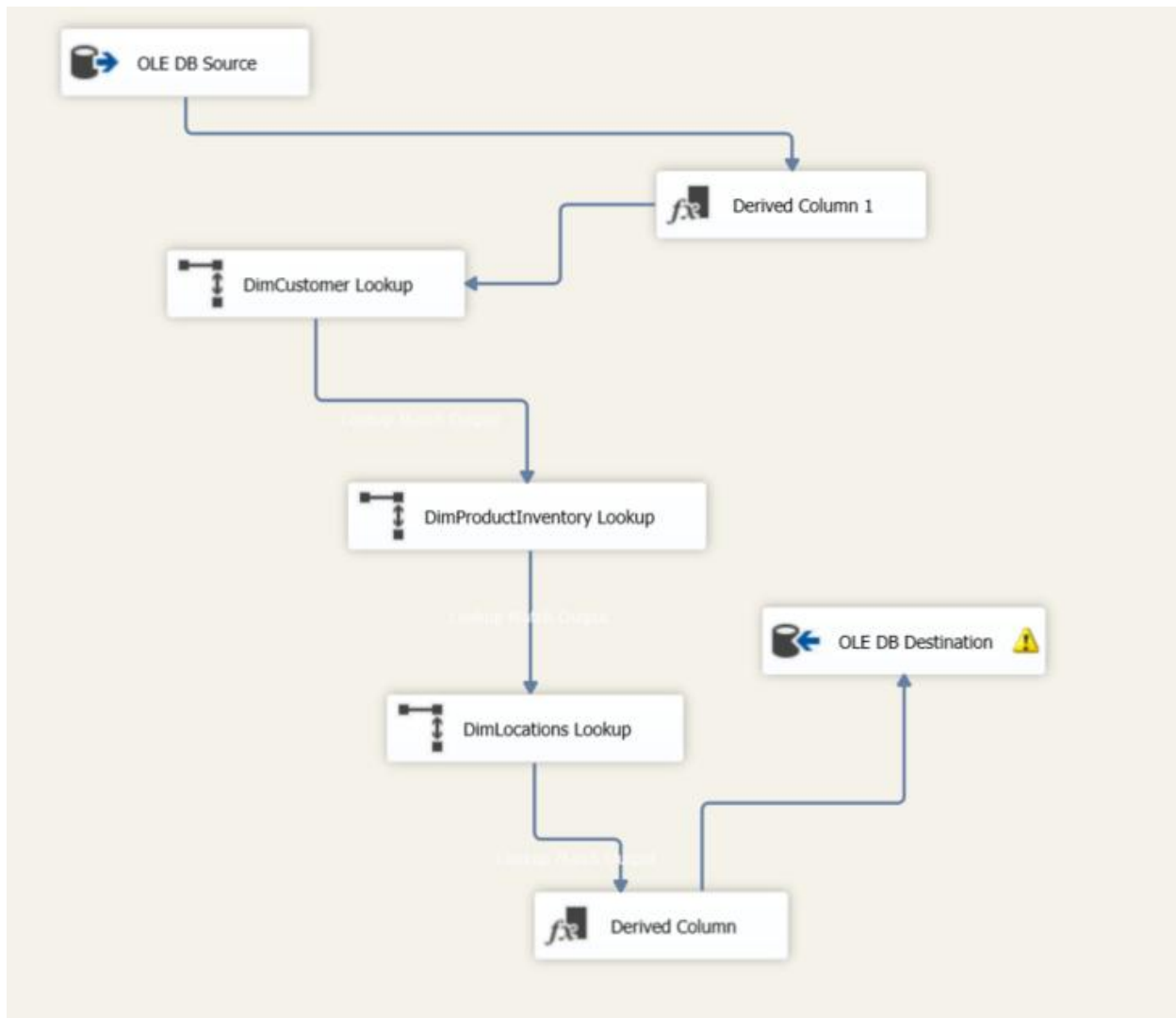




Finally loaded to the data warehouse



## 5. Transform and load FactCustomerOrder table



- The fact table is linked to the corresponding dimension tables using lookup transformations.
- Specifically, the CustomerOrder fact table is joined with the Customer, Location, and Product Inventory dimension tables to establish referential integrity and enable comprehensive analytical queries.

- Additionally, the created\_date field is derived using the GETDATE() function, based on the value of accm\_txn\_create\_time.

Derived Column Transformation Editor

Specify the expressions used to create new column values, and indicate whether the values update existing columns or populate new columns.

Variables and Parameters  
Columns

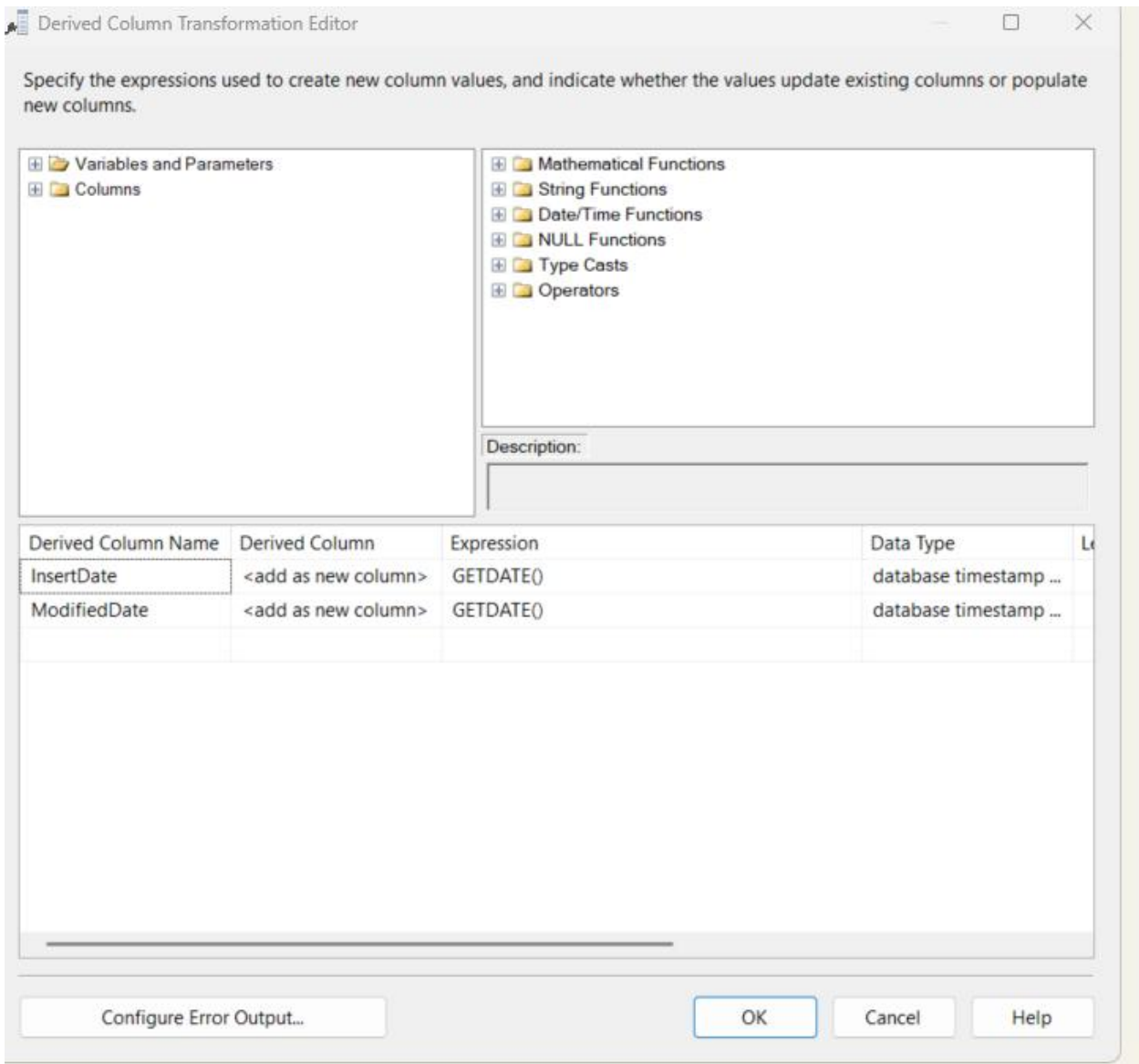
Mathematical Functions  
String Functions  
Date/Time Functions  
NULL Functions  
Type Casts  
Operators

Description:

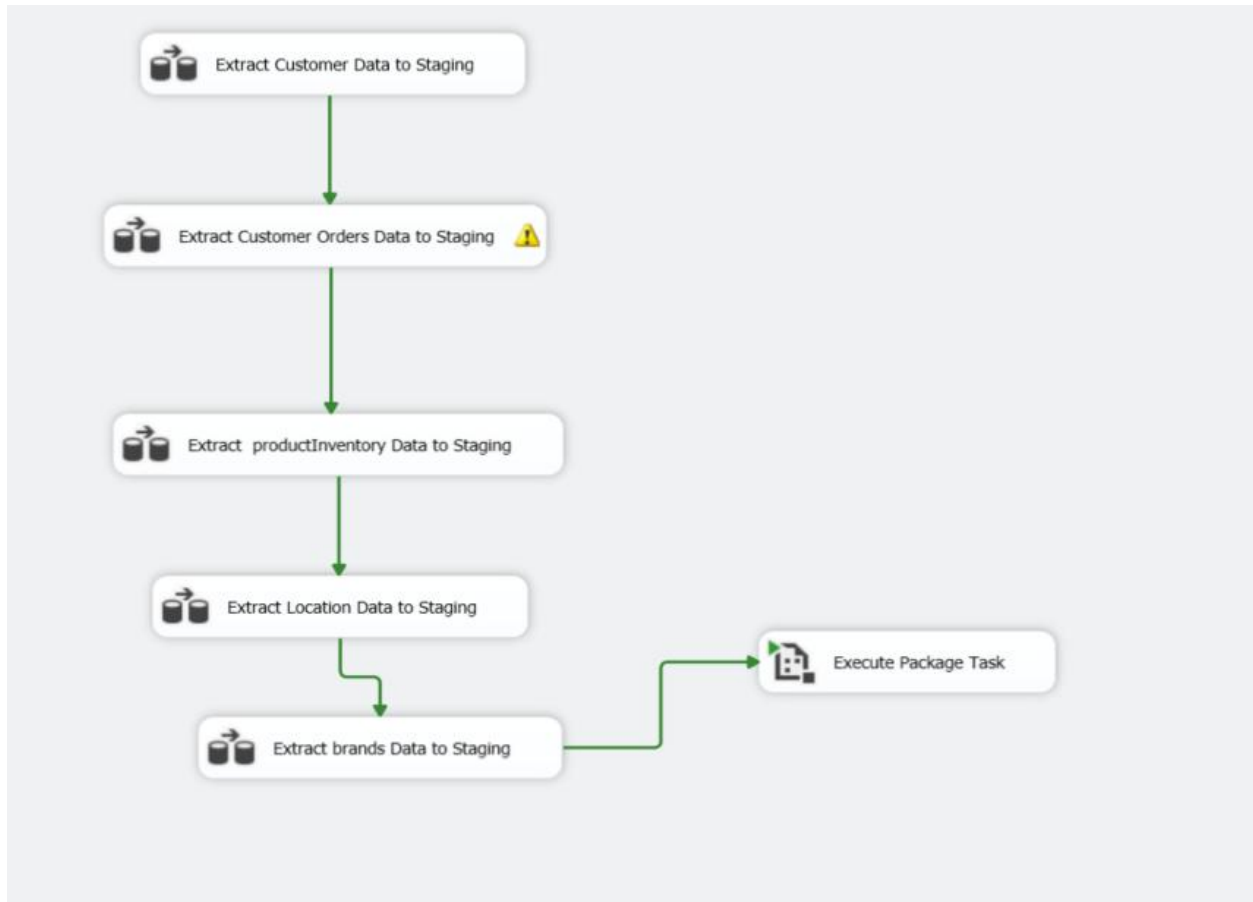
Derived Column Name	Derived Column	Expression	Data Type	Location
accm_txn_create_time	<add as new column>	(DT_DBTIMESTAMP)GETDATE()	database timestamp ...	L

Configure Error Output... OK Cancel Help

Additionally, the InsertDate and ModifiedDate field is derived using the GETDATE() function, based on the value of



## Load to staging



- We have implemented a structured data pipeline where the data is first loaded into the staging area before being transferred to the data warehouse.
- The Execute package task option demonstrates the integration between the staging and data warehouse packages, ensuring that once the staging process is completed, the data is automatically loaded into the data warehouse.