Sri Lanka Institute of Information Technology

Data warehousing and Business Intelligence (IT3021)

Assignment 01

Report

Dasanayake D. R. L

IT22350800

Table of Contents

1. Introduction

This project focuses on building a **Data Warehouse (DW)** for an e-commerce dataset using **SQL Server** and **SSIS** for the **ETL process**. The dataset includes transactional data from Brazilian e-commerce, such as customer orders, product details, and payments. A **star schema** is used to design the warehouse, allowing efficient analysis and reporting. The ETL process extracts data from CSV files and SQL databases, transforms it, and loads it into the warehouse to generate insights for business intelligence and decision-making.

2. Data set selection

Dataset Chosen: Brazilian E-Commerce Public Dataset by Olist

Source: Kaggle (https://www.kaggle.com/datasets/olistbr/brazilian-ecommerce)

Description: The Brazilian Olist dataset contains data collected from a real e-commerce business over approximately two years. It includes multiple CSV and SQL-compatible tables representing various operational aspects such as customer information, orders, products, sellers, and payments. The dataset contains sufficient records and attributes, supporting a rich DW & BI environment with various hierarchies and measures.

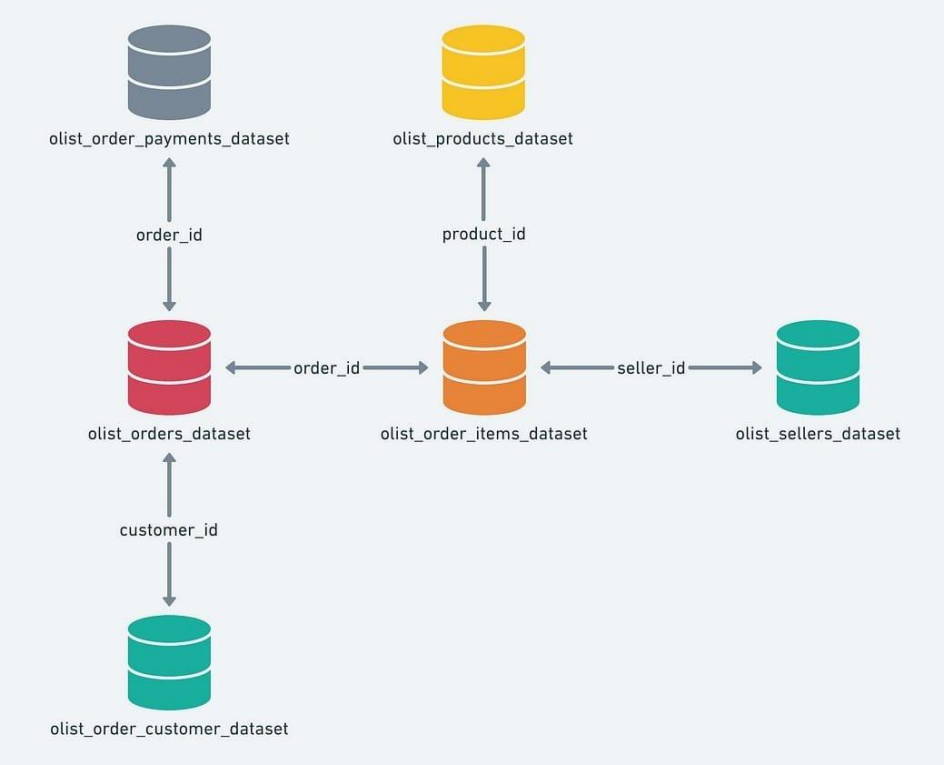
Key Characteristics:

- Sufficient attributes and records
- Covers more than one year of data
- Real-world sales transactions
- Can be used to simulate multiple source types

Data sources:

- olist_orders_dataset.csv
- olist_order_items_dataset.csv
- olist_customers_dataset.csv
- olist_products_dataset.csv
- olist_order_payments_dataset.csv
- olist_sellers_dataset.csv

Entity relationship diagram of the dataset is given below:



3. Preparation of data sources

To create a more realistic ETL process, the original dataset was divided into multiple data sources. Although all the data came from CSV files, some of these were loaded into SQL Server tables to simulate a typical enterprise environment where data exists in different formats.

The following data sources were used:

- **CSV Files:**
    - olist_customers_dataset.csv – Contains customer details like customer ID, ZIP code, and location.
    - olist_orders_dataset.csv – Includes order information such as order ID, status, and order date.
- **SQL Server Tables:**
    - products – Lists product names, categories, and size details.
    - sellers – Includes seller IDs and their location.

- order_items – Shows which products were included in each order.
- order_payments – Stores payment details like amount and method.

Using both CSV files and database tables makes the project more complete and shows how to manage data from multiple sources in a data warehouse system.

4. Solution architecture

4.1 Data Sources

- CSV Files: Raw data files such as olist_customers_dataset.csv and olist_orders_dataset.csv are used as flat file inputs.
- SQL Tables: Other CSVs are first imported into SQL Server tables (e.g., products, sellers, order_items, order_payments) to simulate database sources.

4.2 SSIS Dataflow (ETL Process)

- Extraction: SSIS packages are created to extract data from both flat files (CSV) and SQL tables.
- Transformation: In SSIS, transformation tasks are performed:
  - Data cleaning (e.g., null handling, type conversion)
  - Business logic application (e.g., status mapping, date formatting)
  - Lookup operations to match keys across datasets
- Loading to Staging Area: Cleaned and transformed data is loaded into staging tables in SQL Server for intermediate storage.

4.3 ETL Processing & Merging

- In SQL Server:
  - Data from staging tables is joined, filtered, and enriched as needed.
  - Surrogate keys are generated for dimension tables.
  - Slowly Changing Dimensions (SCD) logic is applied where necessary.

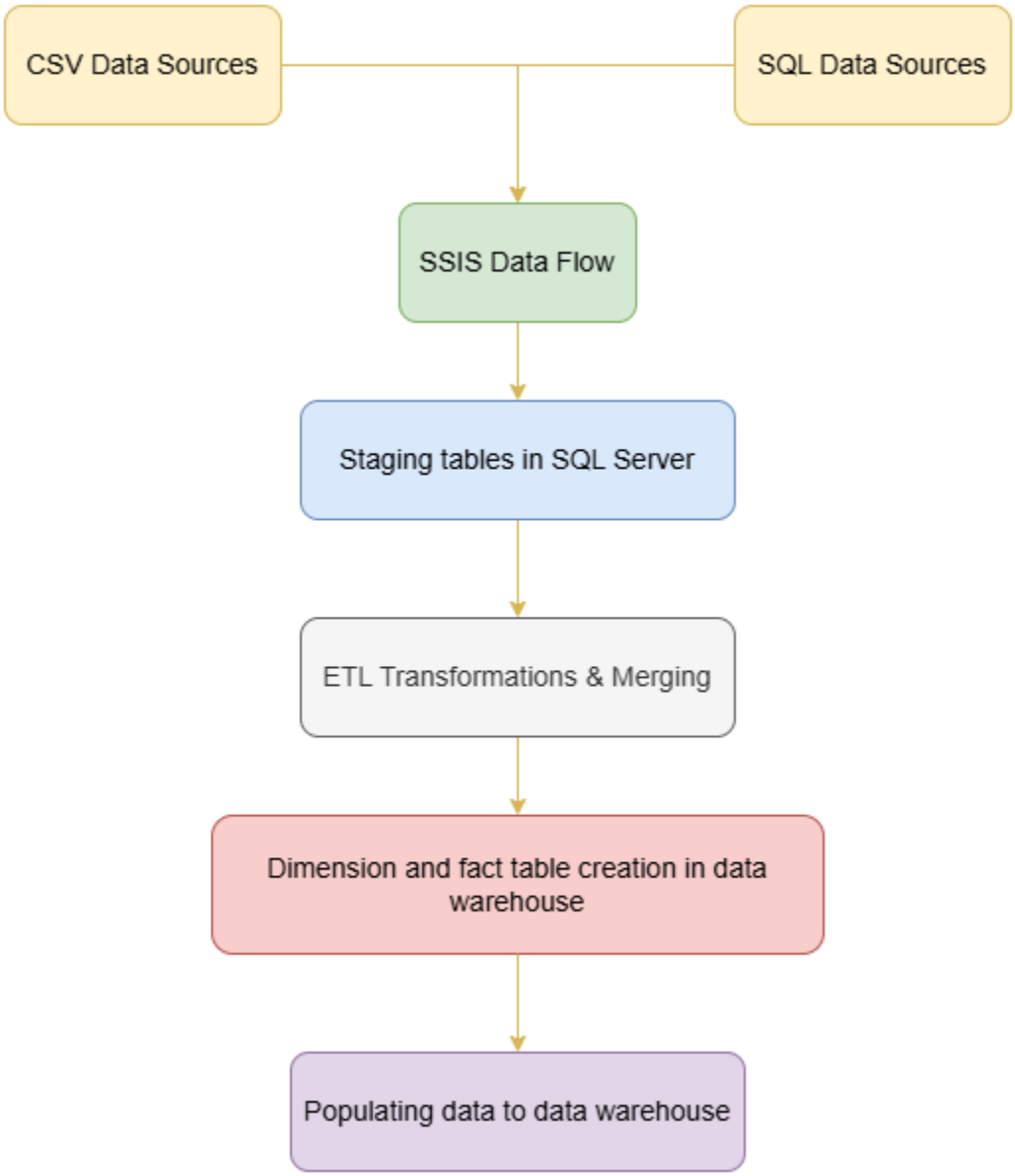
4.4 Data Warehouse Schema

- Star Schema is used with:
  - Dimension Tables: e.g., dim_customer, dim_product, dim_seller, dim_date
  - Fact Table: e.g., fact_order – contains measures such as order value, quantity, and time taken to process.

4.5. Populating the Data Warehouse

- Final step in SSIS loads the cleaned and structured data from staging tables into the Data Warehouse tables.

- This includes both initial inserts and update logic (e.g., for handling SCDs or updating processing time in the fact table).



5. Data warehouse design & development

5.1 Fact Table – fact_Order

- The central fact table fact_Order captures comprehensive order-level data by integrating records from the stg_orders, stg_order_items, and stg_order_payments staging tables. To prepare the data for loading, Sort and Merge operations were applied in the data flow pipeline, ensuring efficient joins and clean deduplication before insertion.

5.2 Key metrics and attributes stored include:

- **Payment details**: PaymentValue, FreightValue, PaymentInstallments, and PaymentType
- **Order fulfillment status**: OrderStatus
- **Product price**: Captured per order at the time of transaction

5.3 Accumulating fact table:

- **accm_txn_create_time**: Timestamp representing when the transaction began (order placed)
- **accm_txn_complete_time**: When the transaction completed (order delivered)
- **txn_process_time_hours**: Automatically calculated field representing total processing time in hours between order creation and completion

These fields help in measuring order lifecycle durations, enabling performance tracking and delay analysis for business reporting.

Foreign keys connect this table to all four dimensions - customer, product, seller, and date supporting a star schema optimized for analytical workloads.

5.4 Dimension Tables

- **dim_Customer**
  Tracks customer-related attributes such as ID, city, state, and zip code. This dimension uses Slowly Changing Dimension (SCD) Type 2 logic to maintain historical versions of customer data. For instance, if a customer moves to a different state, the new record is inserted with an updated EffectiveStartDate, while the old record is retained with an EffectiveEndDate.

- **dim_Product**
  Stores product metadata including category, physical dimensions, and calculated fields like ProductNameLength and ProductDescriptionLength.

- **dim_Seller**
  Includes seller details such as location and zip code, which can be used for geographical segmentation or delivery performance evaluations.

- **dim_Date**
  A static date dimension prepopulated from 2016 to 2018. Each record includes calendar attributes such as Day, Month, Quarter, and DayName, along with a Boolean IsWeekend flag. This enables flexible temporal aggregations, such as weekday vs weekend sales.

5.5 Key Implementation Notes

- Sort and Merge operations were essential in the ETL process to cleanly align records from different staging tables.
- SCD Type 2 was implemented for customers, using EffectiveStartDate, EffectiveEndDate, and IsCurrent to track historical changes.
- The date dimension helps in joining multiple order lifecycle events (OrderPurchaseDate, OrderApprovalDate, etc.) with a consistent and query-efficient format (DateKey in YYYYMMDD).
- This data warehouse design follows dimensional modeling best practices, ensuring scalability, auditability, and high-performance analytics for decision-makers.

5.6 Assumptions for design

- Granularity of Data:
  - The fact table will store data at the order level - each row represents a single transaction or order.
  - It will include metrics like the total order value, quantity ordered, and the time it took to process the order.

- Slowly Changing Dimensions (SCD):
  - The Customer dimension will be Slowly Changing Type 2 (SCD2), meaning if a customer updates their information (e.g., moves to a new location), we'll keep track of the previous data as well.

- Relationships:
  - The Fact Table connects to each Dimension Table via foreign keys. This creates a Star Schema where the fact table is at the centre, and the dimensions surround it.

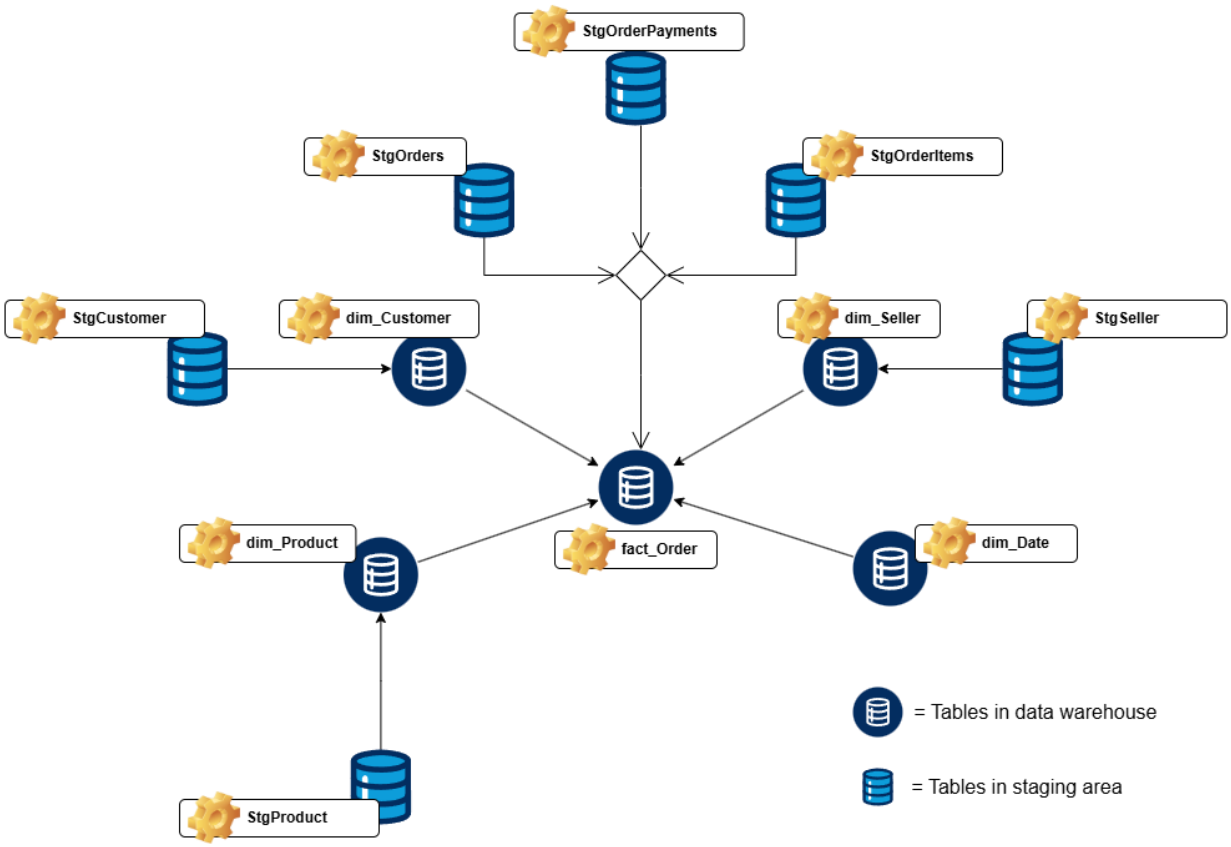
- Date Dimension:
  - The Date Dimension will store details like year, quarter, month, and day of the week to allow reporting by time periods.

- Historical Data:
  - For the initial data load, we'll assume all historical data is available in the source systems.

    - As time goes on, we'll add new records for any changes (like updated customer info).
- Data Quality:
    - We will clean the data during the ETL process, ensuring there are no missing values or inconsistent formats.
    - We'll handle incremental loads so only new transactions are added to the warehouse.

Detailed structure of the tables in data warehouse is presented below.

Table Name	Description	Key Columns / Notable Attributes
dim_Customer	Customer dimension table with Slowly Changing Dimension (SCD) Type 2	CustomerKey (PK), CustomerID, CustomerUniqueID, CustomerZipCodePrefix, CustomerCity, CustomerState, EffectiveStartDate, EffectiveEndDate, IsCurrent
dim_Product	Product dimension table storing product-specific attributes	ProductKey (PK), ProductID, ProductCategoryName, ProductNameLength, ProductDescriptionLength, ProductWeightGrams, ProductPhotosQty, ProductLengthCM, ProductHeightCM, ProductWidthCM
dim_Seller	Seller dimension table storing location and identity info	SellerKey (PK), SellerID, SellerZipCodePrefix, SellerCity, SellerState
dim_Date	Date dimension table for time-based analysis	DateKey (PK), FullDate, Day, Month, Year, Quarter, DayName, MonthName, IsWeekend
fact_Order	Fact table capturing transaction-level order and payment data	OrderKey (PK), OrderID, CustomerKey (FK), SellerKey (FK), ProductKey (FK), OrderStatus, OrderPurchaseDateKey (FK), OrderApprovedDateKey (FK), OrderDeliveredCustomerDateKey (FK), OrderEstimatedDeliveryDateKey (FK), PaymentType, PaymentInstallments, PaymentValue, FreightValue, ProductPrice, accm_txn_create_time, accm_txn_complete_time, txn_process_time_hours



6. ETL development

Tools used:

- SQL Server Integration Services (Visual studio)
- SQL Server Management Studio

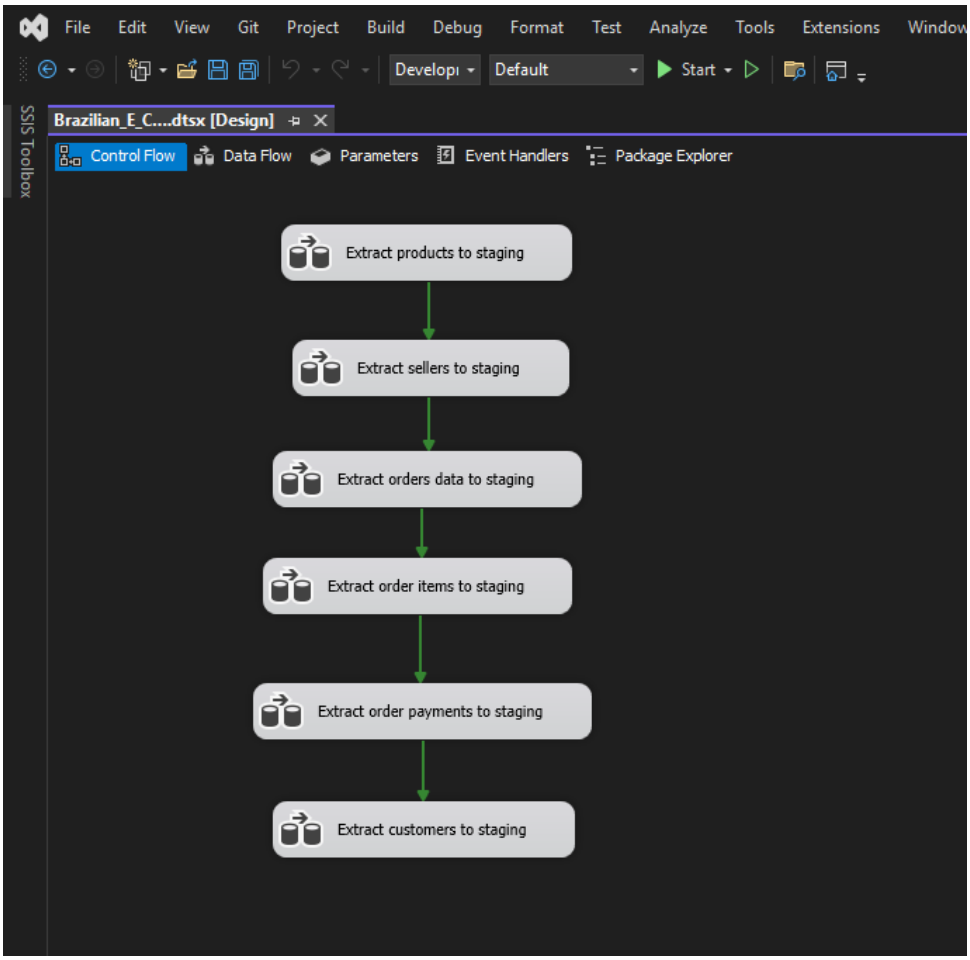
ETL Process Overview:

- **Flat File Source**: Loaded customer and order data from the .csv file.
- **OLE DB Source**: Loaded data from SQL Server tables - order_items, order_payments, products, and sellers.
- **Data Sorting**: Used sort transformation to order data on natural key fields required for merge operations.
- **Merge Join Transformations**: Used to combine related records.
- **Derived Column**: Added calculated fields dynamically using transformations.
- **Slowly Changing Dimension (SCD) Task**: Implemented for the Customer dimension (Type 2) to track historical changes.
- **Lookup Tasks**: Mapped natural keys to surrogate keys in dimension tables.
- **Conditional Split**: Filtered out null or invalid values before loading to the warehouse.

- **Data Conversion**: Ensured compatibility between source and destination column types, especially when moving data between flat files and SQL tables.
- **Logging and Error Handling**: Added custom SSIS logging enabled to track package execution, task status, and row counts.

6.1 Data Extraction to Staging

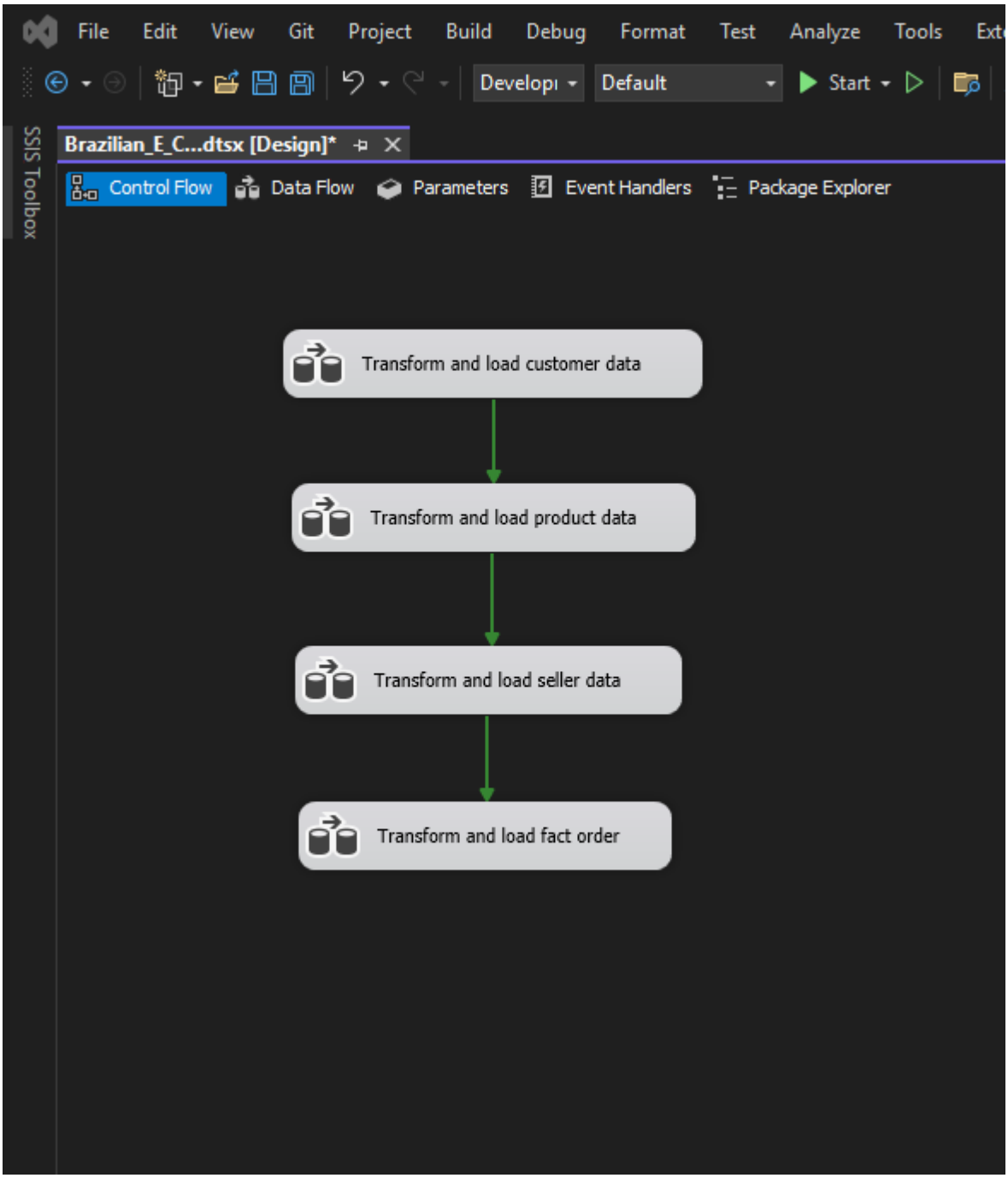
Initially all the data was extracted from all the data sources into separate staging tables. SQL server Integration service was used for this process. Following is the image of the control flow designed in the SSIS to extract data into staging tables.



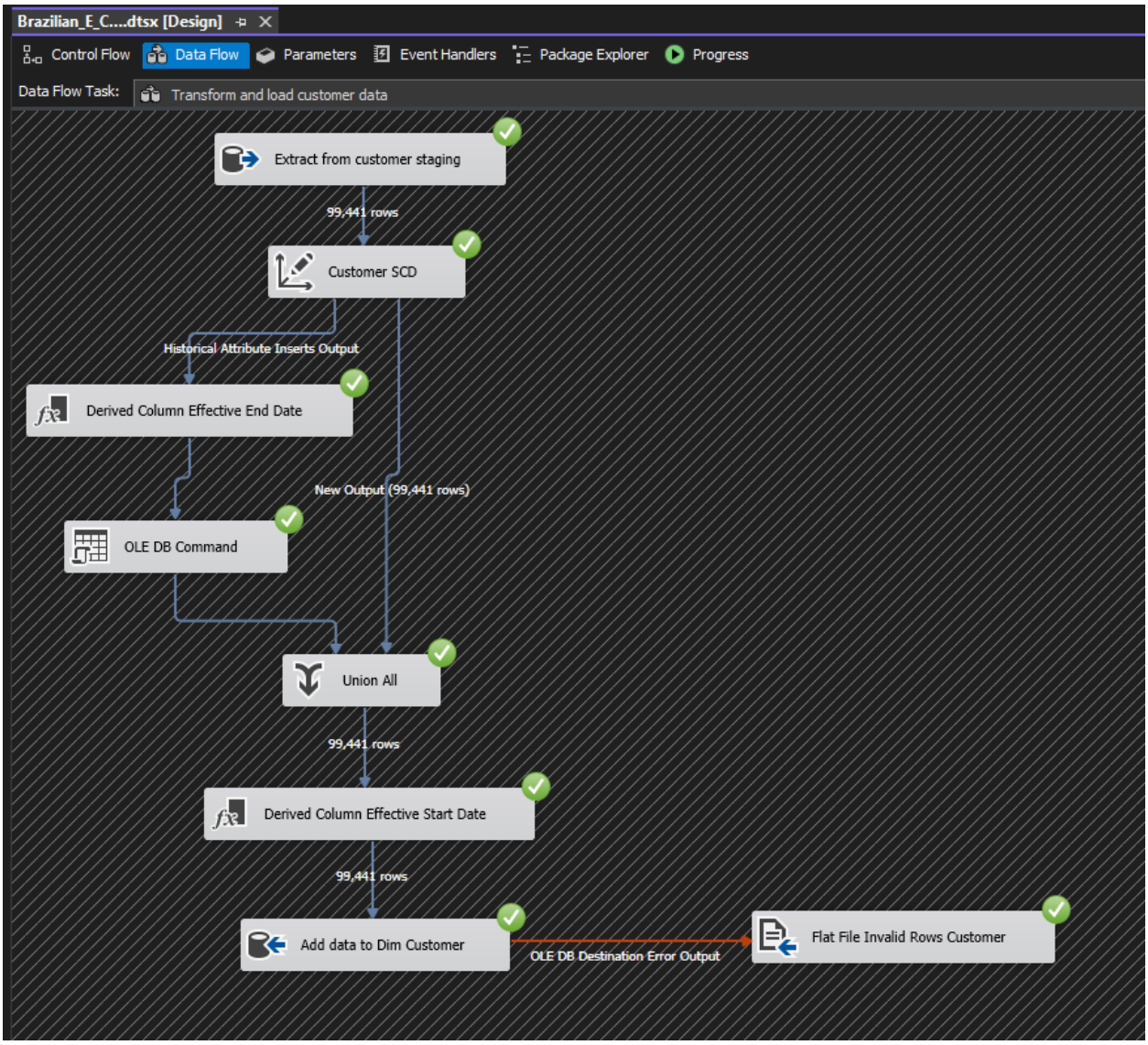
6.2 Transform and load data into data warehouse

Once data was extracted from the source systems, it was transformed and loaded into the data warehouse using SSIS (SQL Server Integration Services). The transformation phase involved cleaning, enriching, and reshaping the data to match the dimensional model design.

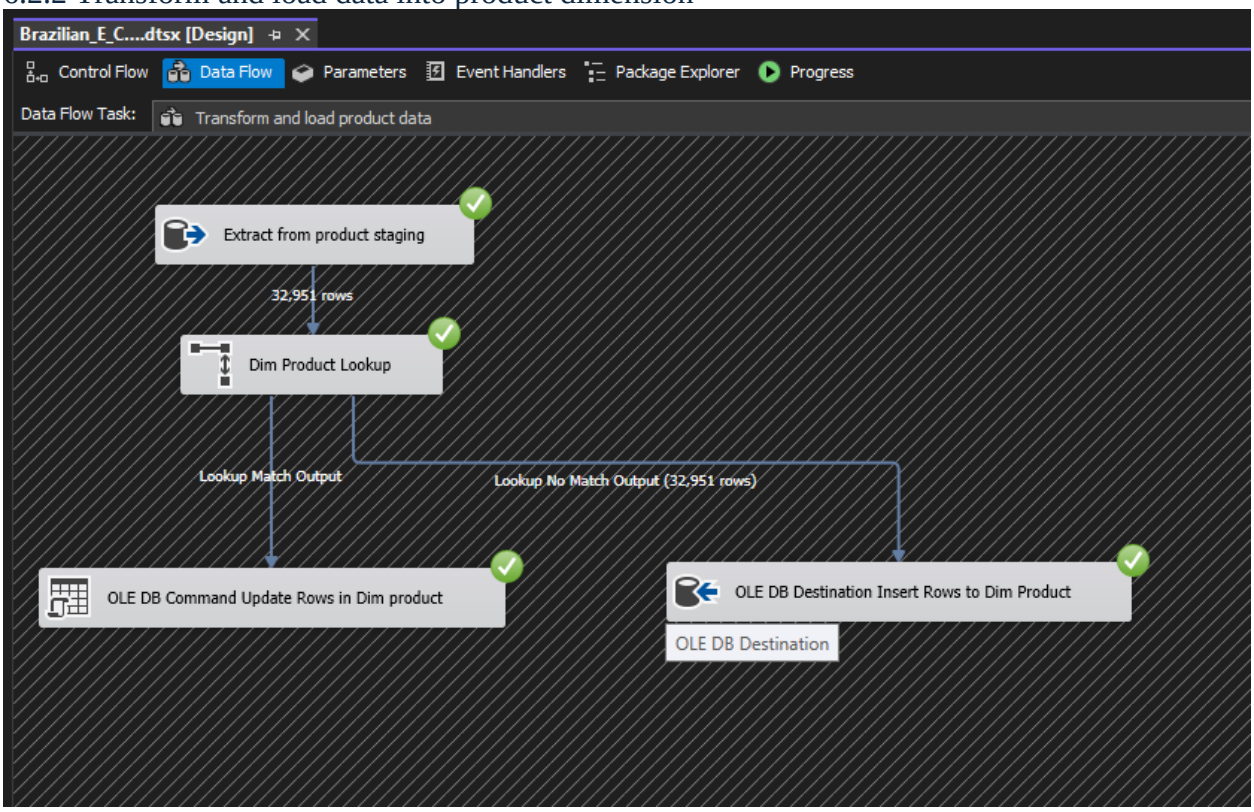
Following is the image of the control flow designed in the SSIS to transform and load data into the data warehouse.



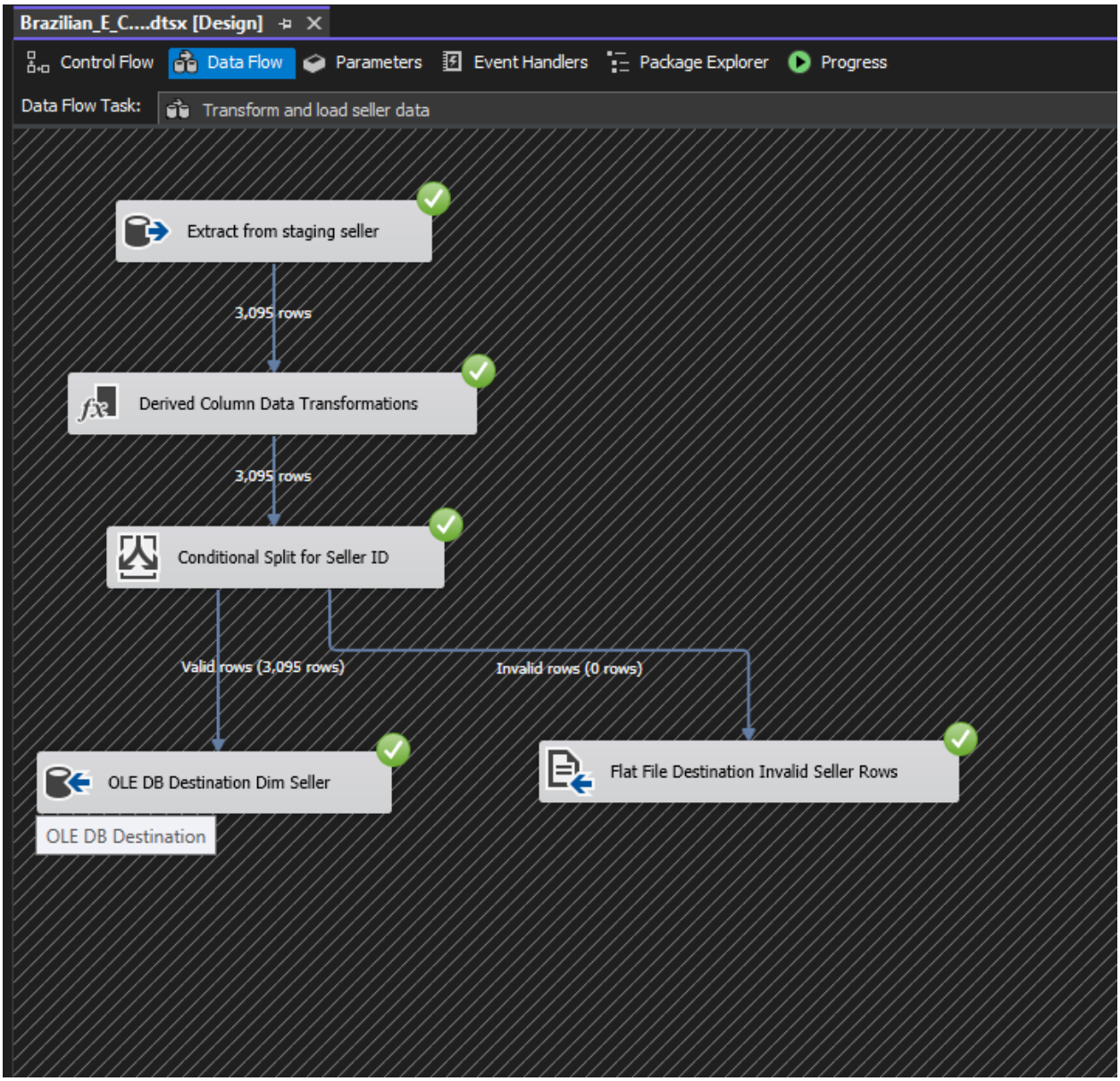
6.2.1 Transform and load data into customer dimension



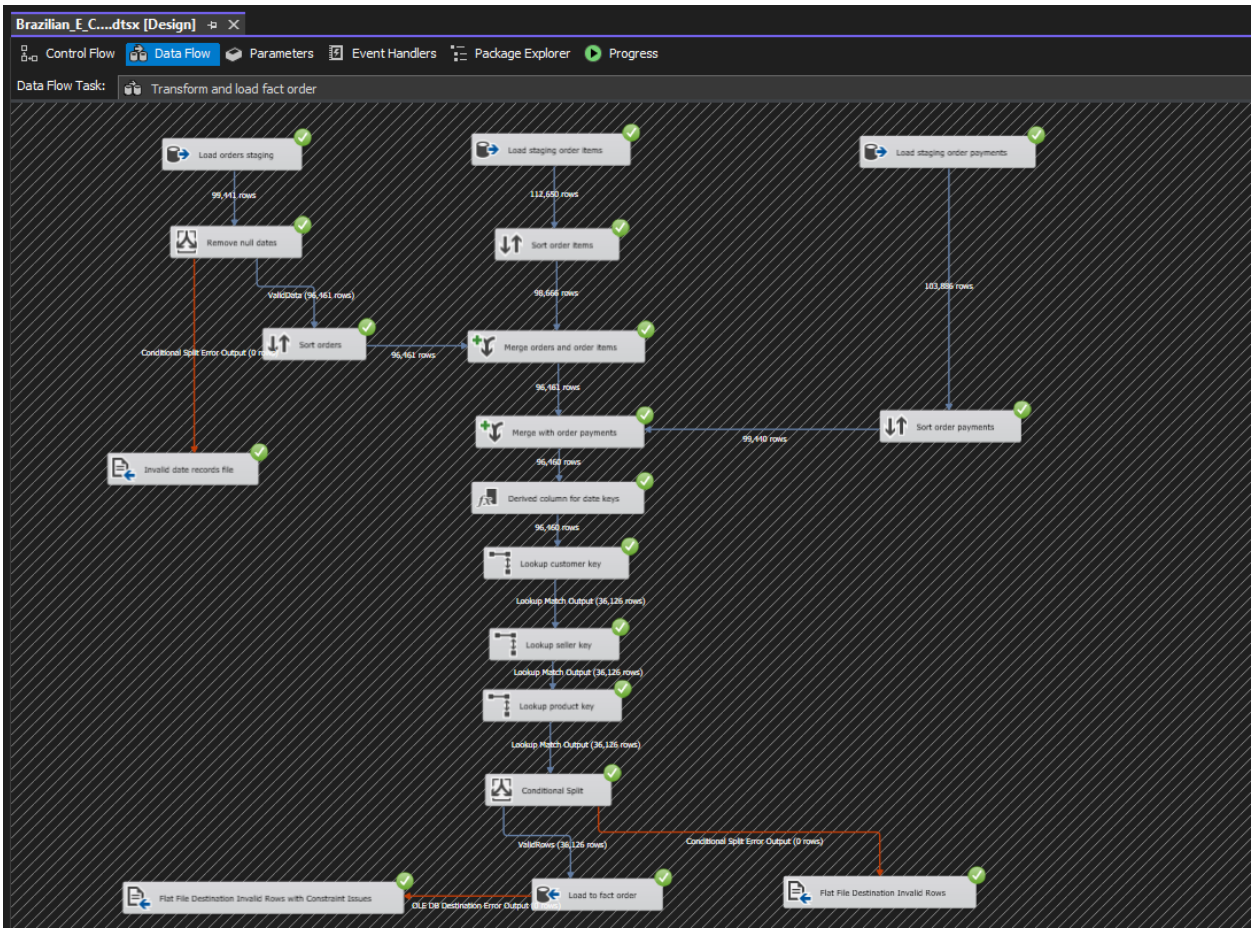
6.2.2 Transform and load data into product dimension



6.2.3 Transform and load data into seller dimension



6.2.4 Transform and load data into orders fact table

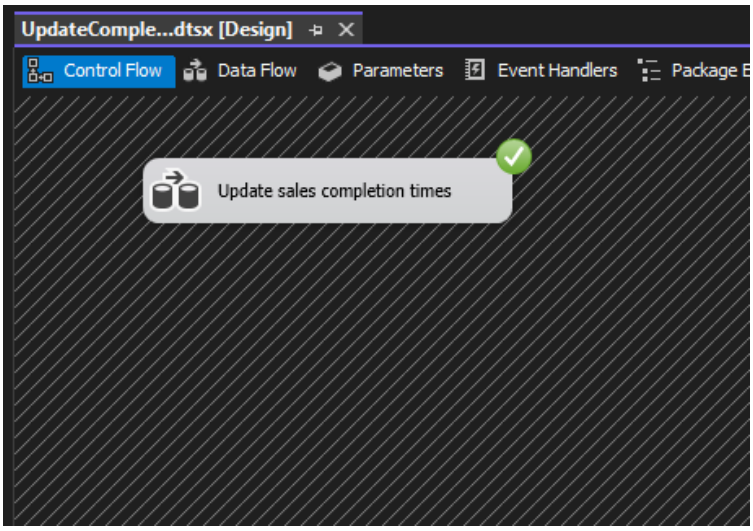


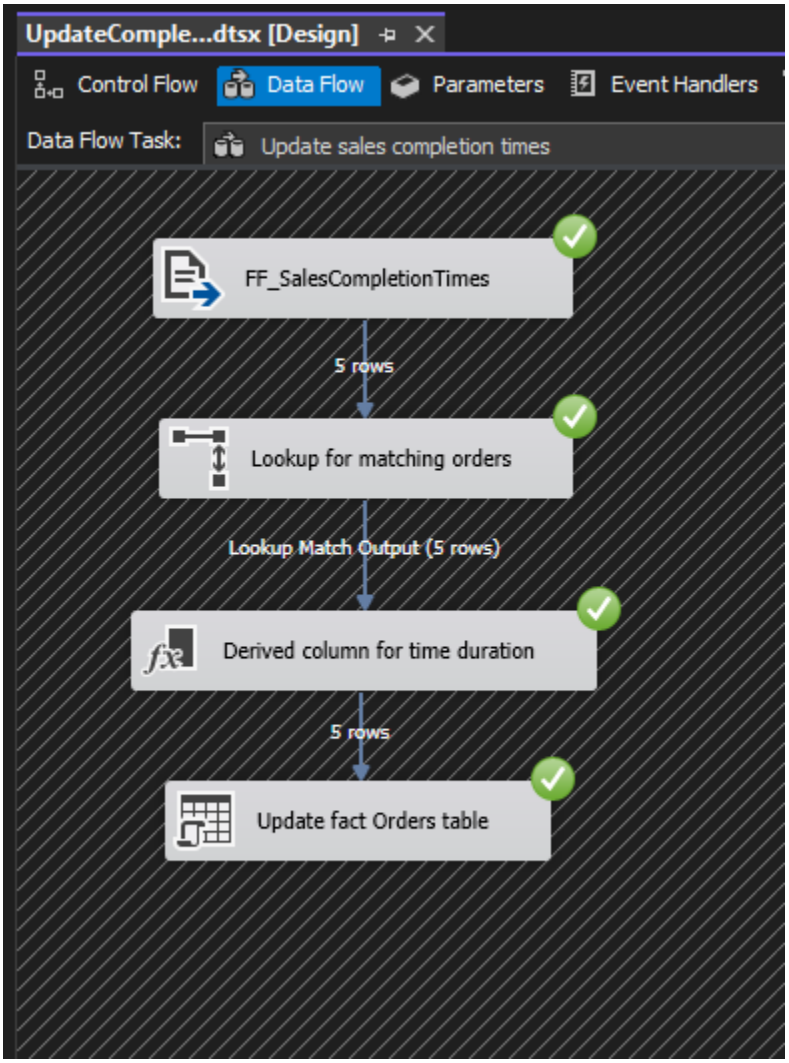
6.3 Accumulating fact table

An **accumulating fact table** is a type of fact table used to track the lifecycle of a business process, where multiple date-related events occur over time. In this project, the fact_Order table was extended to include two additional timestamp columns:

- **accm_txn_create_time**: Captures the date and time when the transaction record is initially created.
- **accm_txn_complete_time**: Records the time when the transaction is completed (e.g., delivery of the order).
- **txn_process_time_hours**: A derived metric that calculates the processing duration in hours between creation and completion.

A separate ETL package was developed using SSIS to update the accm_txn_complete_time and compute txn_process_time_hours. This enables analysis on process efficiency and helps identify delays in the order lifecycle.


UpdateComple...dtsx [Design]
Control Flow
Data Flow
Parameters
Event Handlers
Package E
Update sales completion times


UpdateComple...dtsx [Design]
Control Flow
Data Flow
Parameters
Event Handlers
Data Flow Task:
Update sales completion times
FF_SalesCompletionTimes
5 rows
Lookup for matching orders
Lookup Match Output (5 rows)
Derived column for time duration
5 rows
Update fact Orders table

7. Conclusion

In conclusion, this assignment helped me understand how to build a Data Warehousing and Business Intelligence (DW & BI) system using SQL Server. By choosing a suitable dataset, preparing data from different sources, designing a data warehouse, and developing the ETL process with SSIS, I was able to apply important DW & BI concepts. I also learned how to manage and update data in the fact table. Overall, this project gave me hands-on experience with data processing and reporting, helping me better understand how to turn raw data into useful business insights.

8. References

[1] Microsoft, "SQL Server Integration Services (SSIS)," *Microsoft Learn*, https://learn.microsoft.com/en-us/sql/integration-services/sql-server-integration-services

[2] Microsoft, "SSIS Control Flow," *Microsoft Learn*, https://learn.microsoft.com/en-us/sql/integration-services/control-flow/control-flow