DSTI School of Engineering - Applied MSc in Data Engineering for Artificial Intelligence.

Datawarehouse Project for a Brazilian E-Commerce

Mark Dakroub[1], **Annabelle Fourn**[2] **and Karam Belmoujoud**[3]

[1]*Datawarehouse*
[1]*Extract, Transform, Load*
[1]*SSIS - SMSS*

Professor Nelson Lopez

1. Introduction

Datawarehouse is an emerging technology in the field of big data. It is a collection of technologies created by Microsoft that aim to help decision makers make efficient decisions in the vast amount of complicated data today.

Prior of conducting any substantive analytical processes, it is imperative that extracted data be systematically integrated into the organizational IT infrastructure, and this integration will facilitate uniform access and ensures semantic alignment across datasets. In today's data heterogeneity, it is essential to maintain consistency and interoperability. A widely adopted solution to address these challenges is the deployment of a solution known as Datawarehousing, which centralizes the harmonizes data for analytical readiness.

The generic Datawarehouse architecture consists of 3 critical layers, the staging layer where all the raw data is ingested in its natural form, a second layer where the transformation and data processing take place and the last layer is the main Datawarehouse table, where it contains the final processed data in a well-organized and structured tables.[1][2]. To build a well-architected Datawarehouse, we must execute the ETL operations which includes 3 tasks:

Extracting the raw data (STA area): Raw data is gathered from different data sources and inserted into staging tables. No transformation takes place at this stage.

(ODS area): Data is cleaned and processed, including tasks like data type conversion and manipulation. The transformed data is stored in ODS tables.

Loading into Data Warehouse (DWH area): The fully transformed data is loaded into data warehouse tables.

The ETL tools are a category of specialized tools, with the task of dealing with Datawarehouse homogeneity, cleaning, transforming, and loading problems[3].

2. Methodology

2.1. Data Comprehension and Overview

This conducted study is based on a comprehensive dataset derived from the Brazilian e-commerce platform Olist, obtained from an official website called Kaggle. This dataset comprises 7 interrelated tables, each capturing distinct facets of the E-Commerce transaction process.

We selected the Brazilian E-commerce dataset from Kaggle — a rich and moderately complex collection of transactional data from an online marketplace in Brazil. The dataset captures multiple stages of the e-commerce lifecycle, from order placement to delivery and post-purchase customer feedback, offering a comprehensive 360-degree view of the business process. It contains over 100,000 orders placed between 2016 and 2018, and provides detailed information across several dimensions such as customers, products, payments, reviews, sellers, and logistics

Table 1. Detailed Overview of Olist Dataset Files

File Name	Description	Columns
1. Orders	**Detailed information about each customer order**, including order IDs, customer IDs, order status, order timestamps (purchase, approval, delivery), and estimated delivery dates.	8
2. Products	**Lists all products available on the platform**. Includes product IDs, product category names, product dimensions (weight, height, width, length), number of photos, and text metadata lengths. (5 columns in total)	9
3. Order_Payments	**Records payment transactions for each order**. Includes order ID, payment method (credit card, boleto, voucher, etc.), number of installments, and total payment value. (4 Columns in total)	5
4. Order_Reviews	**Contains customer feedback**, including review ID, order ID, review score, review comment/title, and timestamps for creation and reply.	7
5. Sellers	**Provides information about each seller**, including seller ID, ZIP code prefix, city, and state.	4
6. Geolocation	**Maps ZIP code prefixes to coordinates**. Contains ZIP code, latitude, longitude, city, and state. Useful for location-based analysis.	5
7. Customers	**Customer demographic profiling**. Includes customer ID, unique customer ID, ZIP code, city, state.	5
8. Order Items	**Breaks down individual products within each order.**.	7
9. Product_Translation	**Provides English translation of product category names**.	2

2.2. Pipeline Design

In this study, the data pipeline is designed into three distinct layers, each serving a critical and unique purpose in the preparation and transmission of raw data for later analytical consumption. This pipeline has been implemented using **SSIS (SQL Server Integration Services)** for the ETL (Extract, Transform, Load) operations and **SSMS (SQL Server Management Studio)** with MySQL for database processes. All data sources were obtained in **CSV** (comma-separated values) format. This pipeline ensures data governance, scalability, error tracking, and robust analytical processing by clearly separating the raw, processed, and curated data layers.

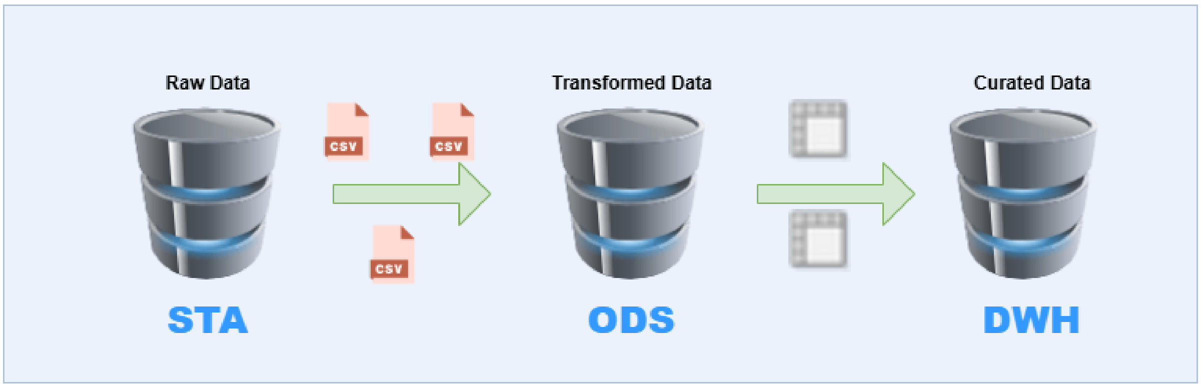
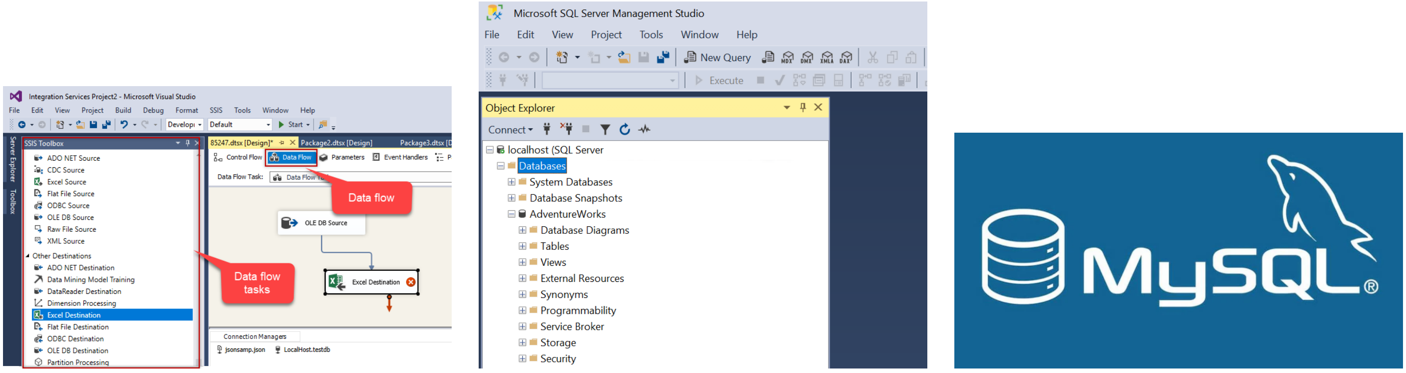


Figure 1. Pipeline processing.

- **Staging Area (STA)**: This layer is responsible for raw data ingestion from the CSV files. At this stage, data are loaded in their original format with minimal preprocessing to preserve the initial structure. Each CSV file is processed through a dedicated SSIS package, ensuring modularity and traceability.
- **Operational Data Store (ODS)**: This intermediate layer handles the data transformation process. Tasks include normalization of categorical values, handling of missing entries, and enforcement of consistent data types such as dates and integers. Records that fail to meet predefined quality criteria are redirected into a specialized table labeled *Technical Rejects*. Like the STA, each dataset is managed by its own ODS package.
- **Data Warehouse (DWH)**: The final layer consolidates the transformed data into a *star schema* architecture within the MySQL database environment. A central fact table records transaction-level metrics and connects to six dimension tables, which represent the core analytical domains: *Customers*, *Products*, *Date*, *Seller*, *Location*, and *Payments*. This schema design enhances query performance and supports multidimensional reporting and analysis.

2.3. Software and Tools

In order to conduct this process, we used several software applications. MySQL Database Engine served as the backbone for creating databases, tables, storing and managing data across pipeline layers, including the staging part, Operational Data Store part, and the Data Warehouse layer. For the design and execution processes, the SQL Server Integration Service (SSIS) was used to provide a flexible and powerful platform for data movement and transformation. Finally, SQL Server Management Studio (SSMS) was integrated in the process to manage the integration packages, monitoring processes and exciting queries. This combination of tools allowed for data ingestion, transformation and administration in curated tables.



SSIS

SSMS

MySQL Engine

Figure 2. Three images displayed side by side.

3. Staging Database and Processes

The staging Database plays a crucial role in the first phase of the pipeline by temporarily storing the raw collected data from various sources, and no transformation took place. It's primary function is to ingest data it its natural format and quality as received. This approach will ensure that ingested data regardless of their anomalies are retained for further processing. By isolating the raw data in the staging part, the organizations will gain the flexibility to perform any transformation task needed before loading them into the Data warehouse.6 files out of 9 are kept for the staging process, as seen below:

Staging Table	Purpose
STA_Orders	Basis for fact_orders
STA_Customers	Basis for dim_customers
STA_Order_items	Basis for dim_deliveries and fact_orders
STA_Order_payments	Basis for dim_payments and fact_orders
STA_Order_Reviews	Basis for dim_reviews and fact_orders
STA_Products	Basis for dim_products

Table 1. Staging tables and their corresponding purposes

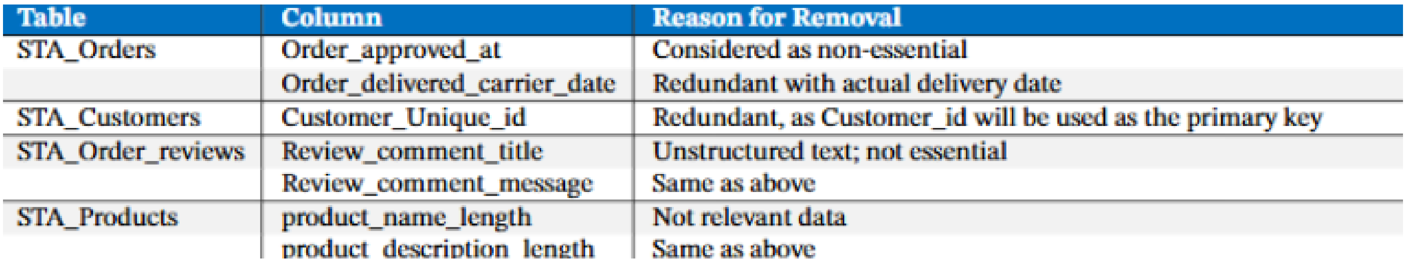
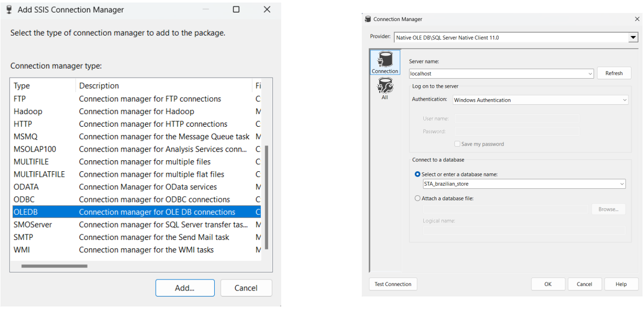


Table	Column	Reason for Removal
STA_Orders	Order_approved_at	Considered as non-essential
	Order_delivered_carrier_date	Redundant with actual delivery date
STA_Customers	Customer_Unique_id	Redundant, as Customer_id will be used as the primary key
STA_Order_reviews	Review_comment_title	Unstructured text; not essential
	Review_comment_message	Same as above
STA_Products	product_name_length	Not relevant data
	product_description_length	Same as above

Figure 3. Reasons for Removing Specific Columns from Tables

3.1. STA Sales

In this step, we will explain the process we have done in the STA Sales. First, a connection manager is created for the whole STA Packages, and the same will be done for the ODS, DWH. As in the images below, we create our new connection manager, in the Server name we choose our local host and then our Database in the below options.



Add SSIS Connection Manager

Select the type of connection manager to add to the package.

Connection manager type:

Type	Description	Fi
FTP	Connection manager for FTP connections	C
Hadoop	Connection manager for Hadoop	M
HTTP	Connection manager for HTTP connections	C
MSMQ	Connection manager for the Message Queue task	M
MSOLAP100	Connection manager for Analysis Services conn...	C
MULTIFILE	Connection manager for multiple files	C
MULTIFLATFILE	Connection manager for multiple flat files	C
ODATA	Connection Manager for OData services	M
ODBC	Connection manager for ODBC connections	C
OLEDB	Connection manager for OLE DB connections	C
SMOServer	Connection manager for SQL Server transfer tas...	M
SMTP	Connection manager for the Send Mail task	M
WMI	Connection manager for the WMI tasks	M

Choosing OLEDB

Connection Manager

Provider: Native OLE DB\SQL Server Native Client 11.0

Server name: localhost

Log on to the server

Authentication: Windows Authentication

User name:

Password:

□ Save my password

Connect to a database

● Select or enter a database name: STA_brazilian_store

○ Attach a database file:

Logical name:

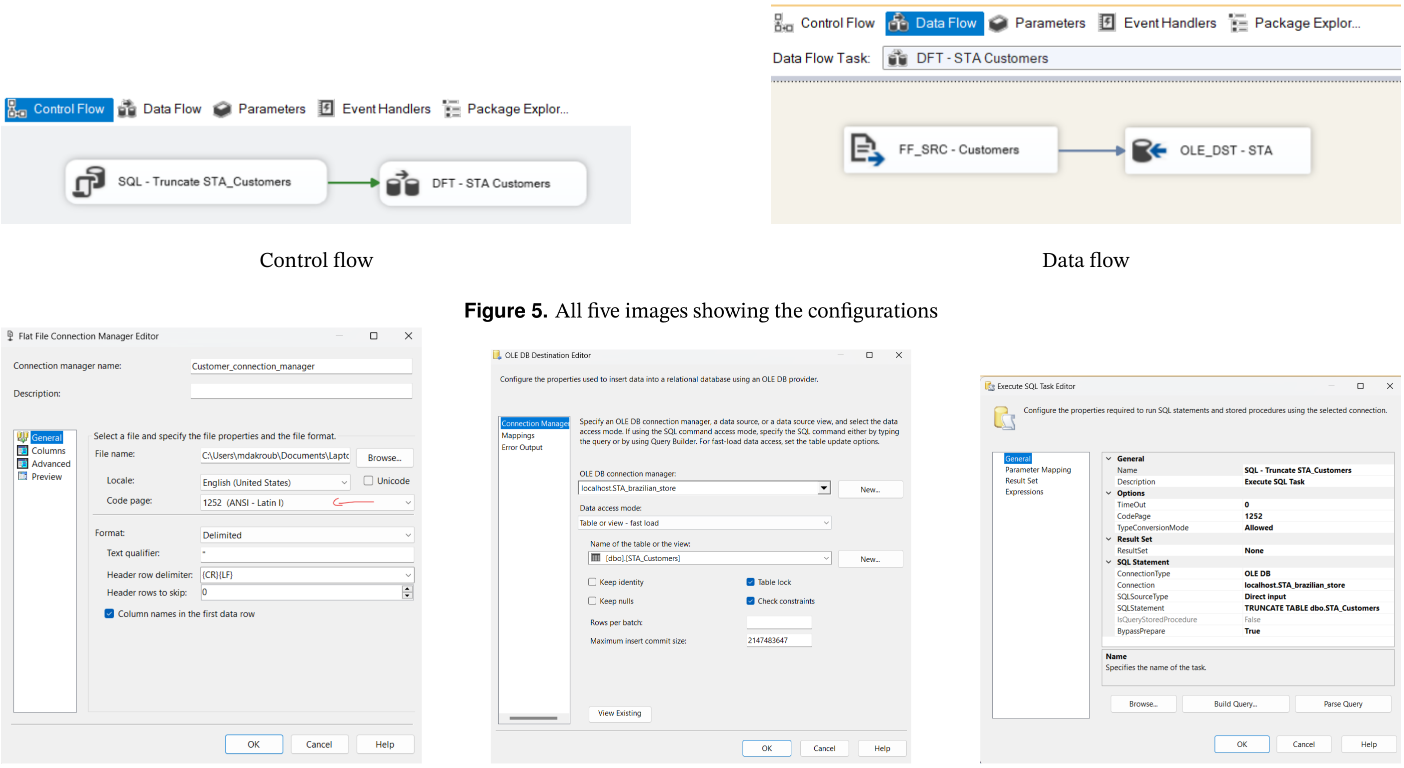
Selecting our Database

Figure 4. Set up the connection manager for all STA packages

Staging Table	Main Content	SSIS Process Description	Output
STA_Orders	**Stores customer order data per transaction**, including order ID, customer ID, order status, and key timestamps related to purchase, approval, shipping, and delivery.	• A Flat File Source was configured. • Code page selected to conform with data. • Data was loaded as-is into STA_Orders via OLE DB Destination. • Truncate is task is created in control flow	**99,441** rows loaded

Table 2. ETL Process Summary for STA_Orders

Below are the steps in visuals. These steps will not be repeated in the new packages because same steps are done for each, only new steps will be added visually.



Control flow

Data flow

Flat File Connection Manager Editor

Connection manager name: Customer_connection_manager

Description:

General
Columns
Advanced
Preview

Select a file and specify the file properties and the file format.

File name: C:\Users\mdakroub\Documents\Laptc

Locale: English (United States) □ Unicode

Code page: 1252 (ANSI - Latin I)

Format: Delimited

Text qualifier: "

Header row delimiter: {CR}{LF}

Header rows to skip: 0

☑ Column names in the first data row

OLE DB Destination Editor

Configure the properties used to insert data into a relational database using an OLE DB provider.

Connection Manager
Mappings
Error Output

Specify an OLE DB connection manager, a data source, or a data source view, and select the data access mode. If using the SQL command access mode, specify the SQL command either by typing the query or by using Query Builder. For fast-load data access, set the table update options.

OLE DB connection manager: localhost.STA_brazilian_store

Data access mode: Table or view - fast load

Name of the table or the view: [dbo].[STA_Customers]

□ Keep identity ☑ Table lock

□ Keep nulls ☑ Check constraints

Rows per batch:

Maximum insert commit size: 2147483647

Execute SQL Task Editor

Configure the properties required to run SQL statements and stored procedures using the selected connection.

General
Parameter Mapping
Result Set
Expressions

General	
Name	SQL - Truncate STA_Customers
Description	Execute SQL Task
Options	
TimeOut	0
CodePage	1252
TypeConversionMode	Allowed
Result Set	
ResultSet	None
SQL Statement	
ConnectionType	OLE DB
Connection	localhost.STA_brazilian_store
SQLSourceType	Direct input
SQLStatement	TRUNCATE TABLE dbo.STA_Customers
IsQueryStoredProcedure	False
BypassPrepare	True

Name
Specifies the name of the task.

File manager

OLE DB Destination

Truncate configuration

Figure 5. All five images showing the configurations

3.2. STA Orders

The STA Orders does not contain lots of changes, it contains the same steps as above.

Staging Table	Main Content	SSIS Process Description	Output
STA_Orders	Stores customer order data per transaction, including order ID, customer ID, order status, and key timestamps related to purchase, approval, shipping, and delivery.	• A Flat File Source was configured. • Data was loaded as-is into `STA_Orders` via OLE DB Destination.	**99,441** rows loaded

Table 3. ETL Process Summary for `STA_Orders`

3.3. STA Orders item

Staging Table	Main Content	SSIS Process Description	Output
STA_Order_items	Breaks down products per order, including order item sequencing, unit price, freight cost, and associated seller and product identifiers.	• A Flat File Source ingested the raw CSV. • Data was loaded into `STA_Order_Items` via OLE DB Destination.	**112,650** rows loaded

Table 4. ETL Process Summary for `STA_Order_items`

3.4. STA Order Payment

Staging Table	Main Content	SSIS Process Description	Output
STA_Order_payments	Captures payment transactions by order, including payment method, installment count, and total amount paid.	• A Flat File Source was configured. • Data was loaded into `STA_Order_Payments` via OLE DB Destination.	**103,886** rows loaded

Table 5. ETL Process Summary for `STA_Order_payments`

3.5. STA Order review

Staging Table	Main Content	SSIS Process Description	Output
STA_Order_reviews	Stores review customer scores, comments, and review dates.	• A Flat File Source ingested the dataset. • Data was loaded into `STA_Order_Reviews` via OLE DB Destination.	**99,441** rows loaded

Table 6. ETL Process Summary for `STA_Order_reviews`

3.6. STA Products

Staging Table	Main Content	SSIS Process Description	Output
STA_Products	Holds product catalog data, including category name, textual metadata lengths, photo quantity, and physical dimensions.	• Two Flat File Sources ingested the data. • `product_name_length`, `product description length` were removed. • Data was loaded into `STA_Products`.	**32,951** rows loaded

Table 7. ETL Process Summary for `STA_Products`

4. Operational Data Store

The Operational Data Store (ODS) serves as a transition layer between the raw data captured in the staging area and the curated data warehouse. Its primary purpose is to host cleaned, structured, and validated data, ensuring consistency and integrity across the ETL process.Within the ODS, data undergoes a series of transformations, including type casting, deduplication, standardization, and referential integrity checks. This layer enables the separation of raw ingestion from business logic, promoting a modular, auditable, and scalable ETL architecture. Only high quality, consistent data from the ODS is propagated to the data warehouse for analytical use.

4.1. ODS Customers

In this section, not much change, only a new Task of resizing the data is added.

ODS Table	Main Content	SSIS Process Description	Source
ODS_Customers	Contains unique, standardized customer records with clean location info.	• DFT and Truncate tasks are created. • Data has been resized. • Resized data inserted into OLE Destination.	Data sourced from `STA_Customers`

Table 8. ETL Process Summary for `ODS_Customers`

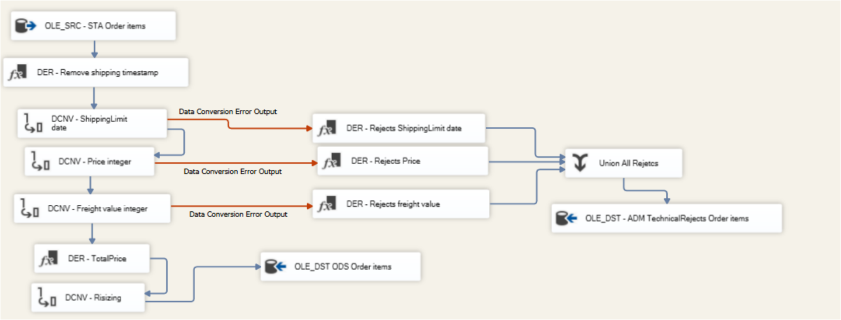
4.2. ODS Orders

A lot of transformation have been done in this file, below are the details of the transformation and a schema.

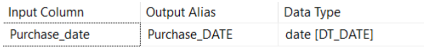
ODS Table	Main Content	SSIS Process Description	Source
ODS_Orders	Stores structured customer order data with standardized timestamps and filtered incomplete rows.	• DFT extracts data from `STA_Orders`. • Removed 2 columns: estimated delivery timestamp and purchase date. • Replaced NULLs with default value `1900-01-01`. • Converted fields to `DT_DATE` type. • Rejected conversions are sent to a Technical Reject table.	`STA_Orders`

Table 9. ETL Process Summary for `ODS_Orders`

Below are the configurations made and the schema

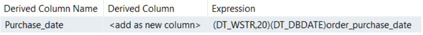

OLE_SRC - STA Order items
DER - Remove shipping timestamp
DCNV - ShippingLimit date
Data Conversion Error Output
DER - Rejects ShippingLimit date
DCNV - Price integer
Data Conversion Error Output
DER - Rejects Price
Union All Rejetcs
DCNV - Freight value integer
Data Conversion Error Output
DER - Rejects freight value
OLE_DST - ADM TechnicalRejects Order items
DER - TotalPrice
OLE_DST ODS Order items
DCNV - Risizing

ODS Order transformation schema



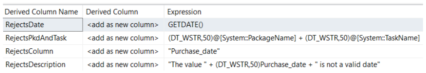
Input Column	Output Alias	Data Type
Purchase_date	Purchase_DATE	date [DT_DATE]

Data conversion to DATE format



Derived Column Name	Derived Column	Expression
Purchase_date	<add as new column>	(DT_WSTR,20)(DT_DBDATE)order_purchase_date

Removing timestamps



Derived Column Name	Derived Column	Expression
RejectsDate	<add as new column>	GETDATE()
RejectsPkdAndTask	<add as new column>	(DT_WSTR,50)@[System::PackageName] + (DT_WSTR,50)@[System::TaskName]
RejectsColumn	<add as new column>	"Purchase_date"
RejectsDescription	<add as new column>	"The value " + (DT_WSTR,50)Purchase_date + " is not a valid date"

Reject message generator

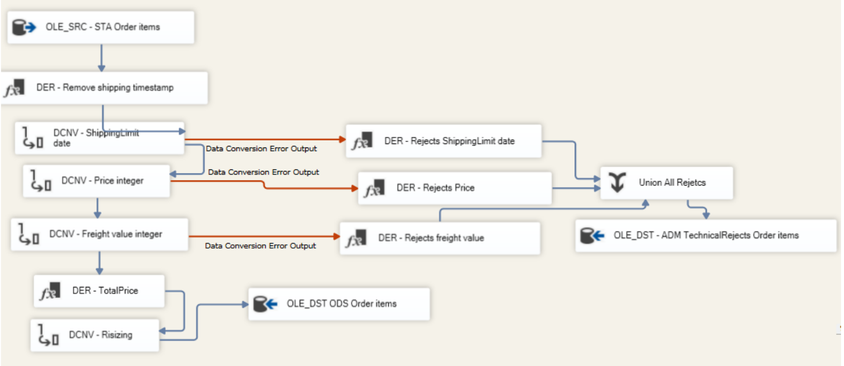
Figure 6. Overview and detailed breakdown of the process

4.3. ODS Order items

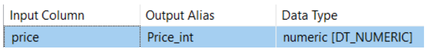
Same transformation as above, additinal task for data conversion to numeric is added.

ODS Table	Main Content	SSIS Process Description	Source
ODS_Order_Items	Breaks down product-level order details, with derived price metrics and referential integrity checks.	• Data extracted from `STA_Order_Items`. • Timestamp removed from `shipping_date`. • Data conversions to `DATE` format. • Technical reject messages generated. • Errors sent to `ADM` table. • `total_price` column created. • Data resized and sent to `OLE_DST`.	`STA_Order_Items`

Table 10. ETL Process Summary for `ODS_Order_Items`

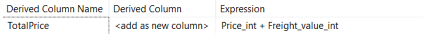

OLE_SRC - STA Order items
DER - Remove shipping timestamp
DCNV - ShippingLimit date
Data Conversion Error Output
DER - Rejects ShippingLimit date
DCNV - Price integer
Data Conversion Error Output
DER - Rejects Price
Union All Rejetcs
DCNV - Freight value integer
Data Conversion Error Output
DER - Rejects freight value
OLE_DST - ADM TechnicalRejects Order items
DER - TotalPrice
OLE_DST ODS Order items
DCNV - Risizing

ODS Order item transformation schema



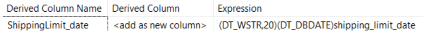
Input Column	Output Alias	Data Type
price	Price_int	numeric [DT_NUMERIC]

Data conversion to numeric format



Derived Column Name	Derived Column	Expression
TotalPrice	<add as new column>	Price_int + Freight_value_int

New Total price column



Derived Column Name	Derived Column	Expression
ShippingLimit_date	<add as new column>	(DT_WSTR,20)(DT_DBDATE)shipping_limit_date

Removing timestamp

Figure 7. Overview and detailed breakdown of the process

4.4. ODS Order payment

In the Order payment section, a paymentID column is added using C sharp Script integrated in the SSIS

ODS Table	Main Content	SSIS Process Description	Source
ODS_Order_Payments	Includes normalized payment data and ensures data integrity with orders.	• Data extracted from `STA_Order_Payments`. • Data conversion applied on `payment_value` field. • `PaymentID` added; underscores replaced with empty strings. • Converted data type to `integer`. • Conversion errors sent to `ADM` table. • Resized data sent to `OLE_ODS_Payment`.	`STA_Order_Payments`

Table 11. ETL Process Summary for `ODS_Order_Payments`

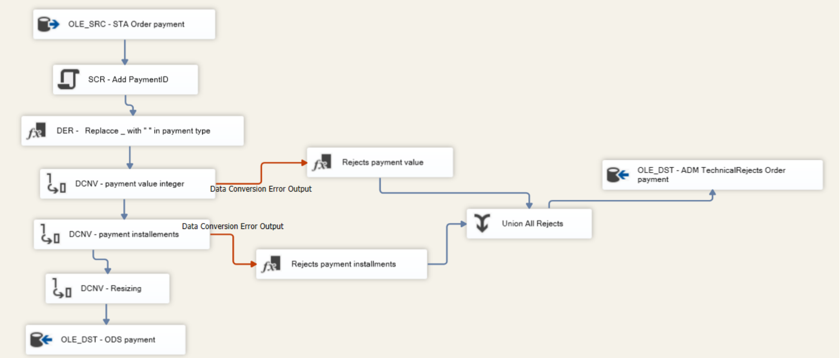

OLE_SRC - STA Order payment
SCR - Add PaymentID
DER - Replacce _ with " " in payment type
DCNV - payment value integer
Data Conversion Error Output
Rejects payment value
OLE_DST - ADM TechnicalRejects Order payment
DCNV - payment installements
Data Conversion Error Output
Union All Rejects
Rejects payment installments
DCNV - Resizing
OLE_DST - ODS payment

Figure 8. Order payment schema

4.5. ODS Order review

ODS Table	Main Content	SSIS Process Description	Source
ODS_Order_Reviews	Standardizes customer reviews and ensures consistent timestamps.	• Data extracted from `STA_Order_Reviews`. • Data conversion to `numeric`. • Conversion errors sent to `ADM` table. • Resized data sent to `ODS_Order_Review`.	`STA_Order_Reviews`

Table 12. ETL Process Summary for `ODS_Order_Reviews`

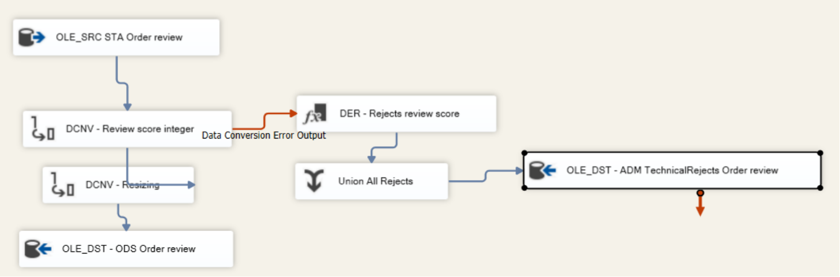

OLE_SRC STA Order review
DCNV - Review score integer
Data Conversion Error Output
DER - Rejects review score
DCNV - Resizing
Union All Rejects
OLE_DST - ADM TechnicalRejects Order review
OLE_DST - ODS Order review

Figure 9. Order review schema

4.6. ODS Order products

ODS Table	Main Content	SSIS Process Description	Source
ODS_Products	Cleans and standardizes product information and translates category names.	• Data extracted from `STA_Products`. • Data conversion to `numeric`. • Conversion errors sent to `ADM` table. • Resized data sent to `ODS_Products`.	`STA_Products`

Table 13. ETL Process Summary for `ODS_Products`

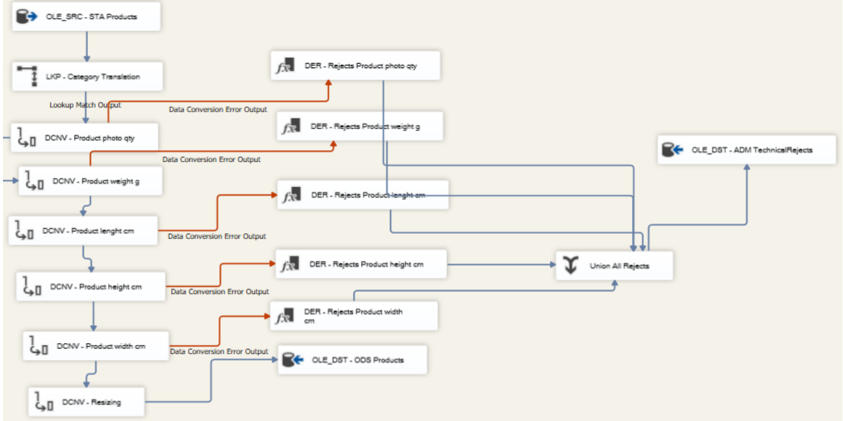

OLE_SRC - STA Products
LKP - Category Translation
Lookup Match Output
DCNV - Product photo qty
DCNV - Product weight g
DCNV - Product lenght cm
DCNV - Product height cm
DCNV - Product width cm
DCNV - Resizing
Data Conversion Error Output
DER - Rejects Product photo qty
DER - Rejects Product weight g
DER - Rejects Product lenght cm
DER - Rejects Product height cm
DER - Rejects Product width cm
OLE_DST - ODS Products
Union All Rejects
OLE_DST - ADM TechnicalRejects

Figure 10. Order product schema

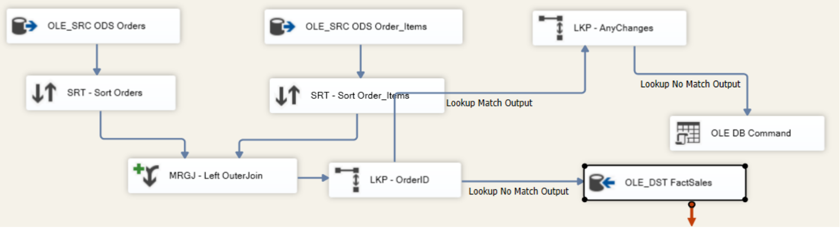
5. Description of Data Warehouse Tables Transformations

5.1. DWH FactSales

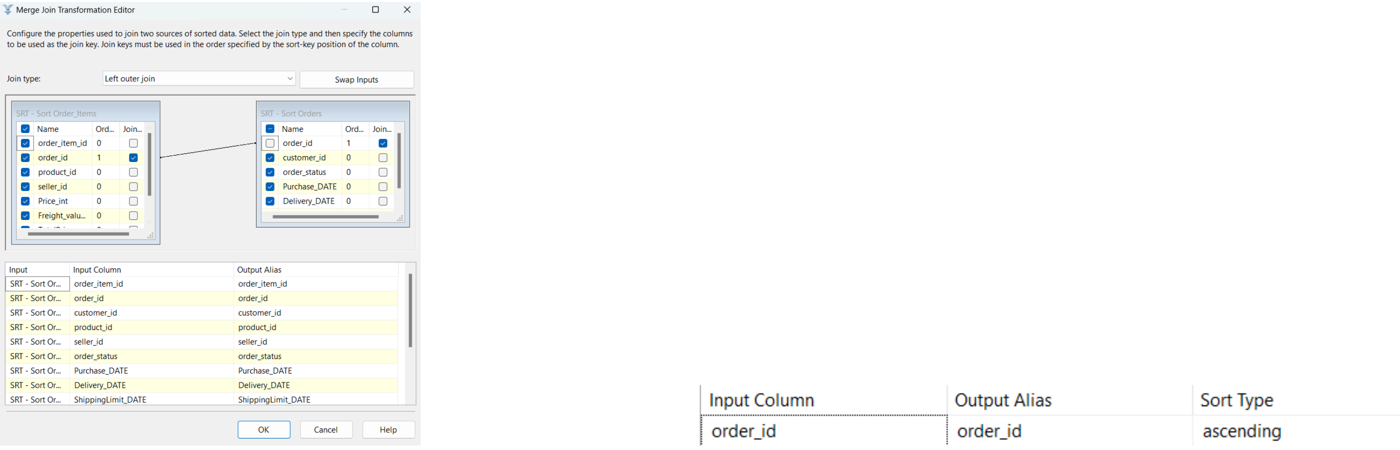
New configurations are added in this section, where the pipeline checks for any new coming data to add it to the table or if any changes, it will update the table.

Fact Table	Main Content	SSIS Process Description	Source
fact_sales	Central fact table capturing consolidated order metrics, including sales amount, freight, and review score.	• Data sourced from `ODS_Orders` and `ODS_Order_Items`. • `OrderID` and order items sorted alphabetically. • Sorted IDs merged using Left Outer Join. • `LKP - new` inserted into `OLE_fact_sales` table. • `LKP - checkChanges` inserted into `OLE_command` table.	`ODS_Orders`, `ODS_Order_Items`

Table 14. ETL Process Summary for `fact_sales`



Fact Sales Schema



Merge Join Transformation Editor

Configure the properties used to join two sources of sorted data. Select the join type and then specify the columns to be used as the join key. Join keys must be used in the order specified by the sort-key position of the column.

Join type: Left outer join

SRT - Sort Order_Items

☑	Name	Ord...	Join...
☑	order_item_id	0	☐
☑	order_id	1	☑
☑	product_id	0	☐
☑	seller_id	0	☐
☑	Price_int	0	☐
☑	Freight_valu...	0	☐

SRT - Sort Orders

☐	Name	Ord...	Join...
☐	order_id	1	☑
☑	customer_id	0	☐
☑	order_status	0	☐
☑	Purchase_DATE	0	☐
☑	Delivery_DATE	0	☐

Input	Input Column	Output Alias
SRT - Sort Or...	order_item_id	order_item_id
SRT - Sort Or...	order_id	order_id
SRT - Sort Or...	customer_id	customer_id
SRT - Sort Or...	product_id	product_id
SRT - Sort Or...	seller_id	seller_id
SRT - Sort Or...	order_status	order_status
SRT - Sort Or...	Purchase_DATE	Purchase_DATE
SRT - Sort Or...	Delivery_DATE	Delivery_DATE
SRT - Sort Or...	ShippingLimit_DATE	ShippingLimit_DATE

Input Column	Output Alias	Sort Type
order_id	order_id	ascending

Two data sources merged

Data sorted

Figure 11. Configurations of FactSales

5.2. DWH Fact Payment

Fact Table	Main Content	SSIS Process Description	Source
fact_payments	Describes payment methods, installment types, and total paid values across orders.	• Data sourced from `ODS_Orders` and `ODS_Payments`. • Orders and payments sorted alphabetically. • Sorted IDs merged using Left Outer Join. • LKP – match `OrderID` with `PaymentID`. • `LKP - new` inserted into `OLE_fact_payment` table. • `LKP - checkChanges` inserted into `OLE_command` table.	`ODS_Orders`, `ODS_Payments`

Table 15. ETL Process Summary for `fact_payments`

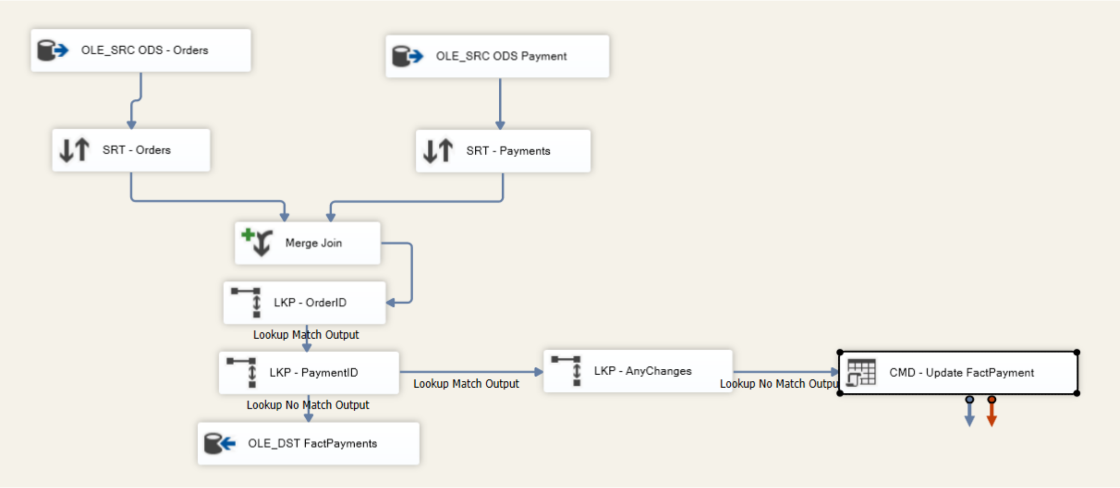


Figure 12. Fact Payment Schema

5.3. DWH Customer

The configuration is simple here as above, only checking changes to apply the SCD type 1 to update table for any changes or adding new data.

Dimension Table	Main Content	SSIS Process Description	Source
dim_customers	Contains descriptive attributes about customers, such as city, state, and ZIP code, supporting demographic analysis.	• Data sourced from `ODS_Customers`. • `LKP` – if `CustomerID` is new, insert into `DimCustomers`. • `LKP` – if any changes, update the existing record in the table.	`ODS_Customers`

Table 16. ETL Process Summary for `dim_customers`

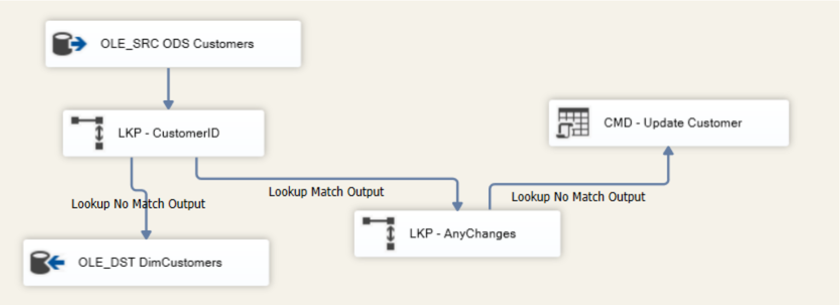


Figure 13. Dim Customer Schema

5.4. DWH Products

Dimension Table	Main Content	SSIS Process Description	Source
dim_products	Provides detailed product attributes including category, physical dimensions, and classifications.	• Data sourced from `ODS_Products`. • LKP – if `ProductID` is new, insert into `DimProducts`. • LKP – if any changes, update the existing record in the table.	`ODS_Products`

Table 17. ETL Process Summary for `dim_products`

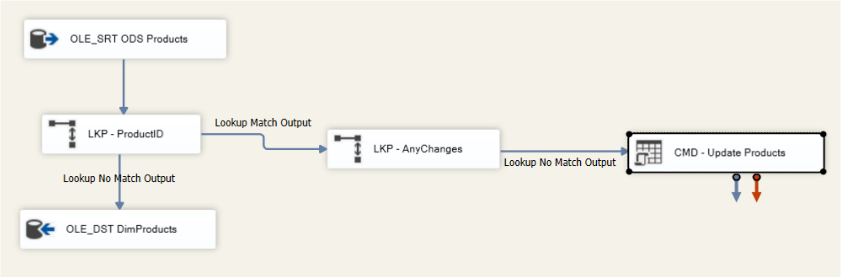


Figure 14. Dim Products Schema

5.5. DWH Review

Dimension Table	Main Content	SSIS Process Description	Source
dim_reviews	Stores customer satisfaction data based on review scores and timestamps.	• Data sourced from `ODS_Reviews`. • LKP – join `OrderID` with `ReviewID`. • LKP – if `ReviewID` is new, insert into `Order_Review` table. • LKP – if any changes, update the existing record.	`ODS_Reviews`

Table 18. ETL Process Summary for `dim_reviews`

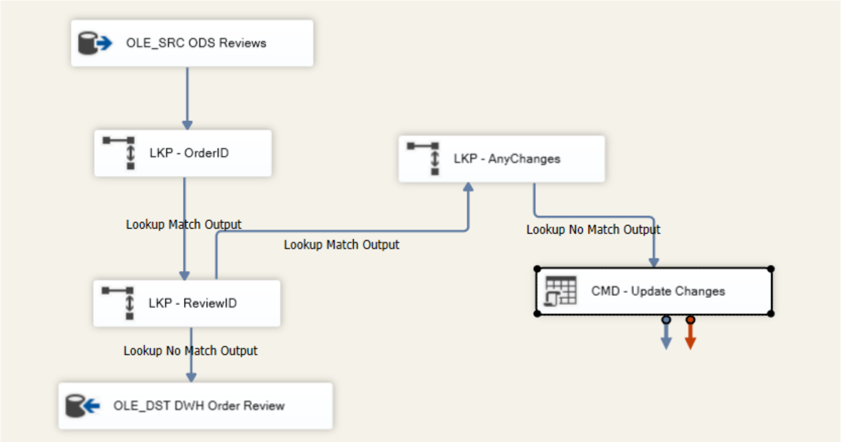


Figure 15. Dim Review Schema

5.6. DWH Order Status

Dimension Table	Main Content	SSIS Process Description	Source
dim_OrderStatus	Stores customer satisfaction data based on review scores and timestamps.	• Data sourced from `ODS_Orders`. • LKP – if `ProductID` is new, insert into `DimOrderStatus`. • LKP – if any changes, update the existing record.	`ODS_Orders`

Table 19. ETL Process Summary for `dim_OrderStatus`

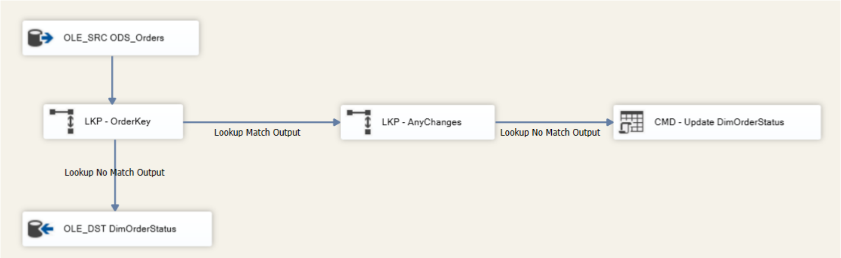


Figure 16. Dim Order status Schema

5.7. DWH Payment method

Dimension Table	Main Content	SSIS Process Description	Source
dim_PaymentMethod	Stores descriptive information about payment types and methods used across orders, supporting transaction analysis.	• Data sourced from `ODS_Payments`. • Joined `OrderID` with `PaymentID`. • LKP – if `PaymentID` is new, insert into `DimPaymentMethod`. • LKP – if any changes, update the existing record.	`ODS_Payments`

Table 20. ETL Process Summary for `dim_PaymentMethod`

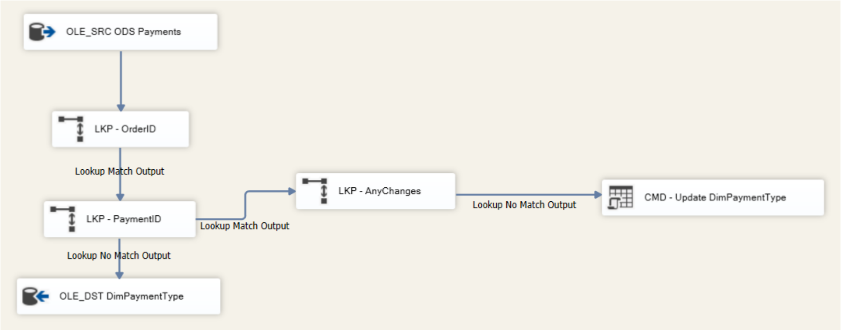


Figure 17. Dim Payment method Schema

Insights Extracted from the Data Warehouse

From our Data Warehouse, we can extract a wide range of meaningful insights by analyzing sales transactions, customer behavior, logistics performance, payment preferences, and satisfaction indicators.

Key Findings

- **Sales Trend:**
  Top selling products allow the business to understand which product categories contribute most to overall revenue.
- **Customer Behavior:**
  Average Basket Value by Region helps identify high-value regions where customers tend to spend more.
- **Preferred Payment Method:**
  Grouping by payment type helps understand customer habits and preferences, supporting optimization of the range of payment methods offered.
- **Customer Satisfaction:**
  Average review score helps the business monitor product quality and guide product development.
- **TotalPrice KPI:**
  A KPI named `TotalPrice` was created at the ODS level, combining the product price and the freight value to represent the full cost of a purchased item.
  This metric provides a more accurate view of the true financial impact of each transaction and allows the business to evaluate the profitability of each order.
- **Late Delivery Rate KPI:**
  A KPI named `Late Delivery Rate` was created at the DWH level. It measures the percentage of orders where the delivery date exceeds the shipping limit date.
  This metric helps detect operational inefficiencies and delays within the logistics process.

Related to our dataset, the output of the KPI listed above are:

KPI	Result / Insight
Top 10 Selling Product Categories	1. health_beauty 2. watches_gifts 3. bed_bath_table 4. sports_leisure 5. computers_accessories 6. furniture_decor 7. housewares 8. cool_stuff 9. auto 10. garden_tools
Preferred Payment Method	`credit_card` (**73.92%**)
Top Region by Average Basket Value	PB (Paraíba) with an average basket of **R$ 234.20**
Average Review Score	**4.09** / 5
Late Delivery Rate	**7.87%** of orders were delivered after the estimated date

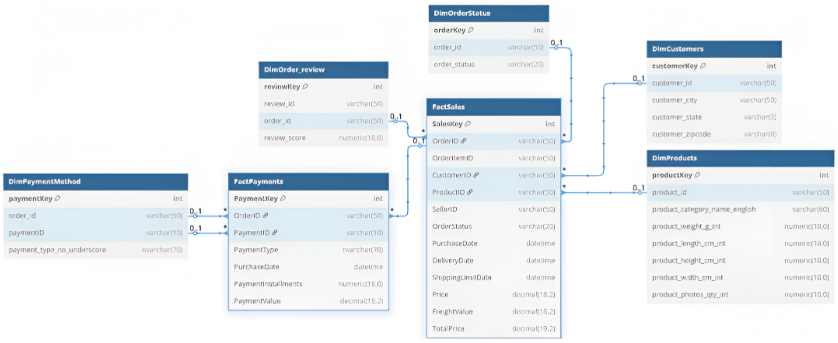
Table 21. Key Performance Indicators and Insights from the Data Warehouse

with some further analysis:

Dimension Table	Purpose	Key Attributes	Types of Analysis Supported
`dim_customers`	Stores customer information for behavior and location-based analysis	Customer ID, city, state	- Orders by region - Customer lifetime value - Repeat purchases and churn
`dim_products`	Stores product-related details for performance tracking and categorization	Product ID, category name, translated name, product type	- Top-selling products - Product trends - Cross-sell opportunities
`dim_OrderStatus`	Captures delivery performance and shipping timelines	Order item ID, estimated delivery date, actual delivery date, delay	- Average delivery time - On-time vs. late deliveries - Regional logistics efficiency
`dim_payments_method`	Records payment method and transaction details for financial analysis	Order ID, payment method, installments, payment amount	- Preferred payment methods - Total revenue by method - Customer payment behaviors
`dim_reviews`	Contains review scores and feedback for quality and sentiment tracking	Review ID, score, comment length, review date	- Average rating per product/seller - Correlation between reviews and sales - Sentiment analysis over time

Table 22. Summary of Dimension Tables and Analytical Use Cases

6. Star Schema



DimOrderStatus	
orderKey	int
order_id	varchar(50)
order_status	varchar(20)

DimOrder_review	
reviewKey	int
review_id	varchar(50)
order_id	varchar(50)
review_score	numeric(18,0)

DimCustomers	
customerKey	int
customer_id	varchar(50)
customer_city	varchar(50)
customer_state	varchar(3)
customer_zipcode	varchar(6)

FactSales	
SalesKey	int
OrderID	varchar(50)
OrderItemID	varchar(50)
CustomerID	varchar(50)
ProductID	varchar(50)
SellerID	varchar(50)
OrderStatus	varchar(20)
PurchaseDate	datetime
DeliveryDate	datetime
ShippingLimitDate	datetime
Price	decimal(18,2)
FreightValue	decimal(18,2)
TotalPrice	decimal(19,2)

DimProducts	
productKey	int
product_id	varchar(50)
product_category_name_english	varchar(60)
product_weight_g_int	numeric(18,0)
product_length_cm_int	numeric(18,0)
product_height_cm_int	numeric(18,0)
product_width_cm_int	numeric(18,0)
product_photos_qty_int	numeric(18,0)

DimPaymentMethod	
paymentKey	int
order_id	varchar(50)
paymentID	varchar(10)
payment_type_no_underscore	nvarchar(70)

FactPayments	
PaymentKey	int
OrderID	varchar(50)
PaymentID	varchar(10)
PaymentType	nvarchar(70)
PurchaseDate	datetime
PaymentInstallments	numeric(18,0)
PaymentValue	decimal(18,2)

Figure 18. Star Schema

7. Project Deployment

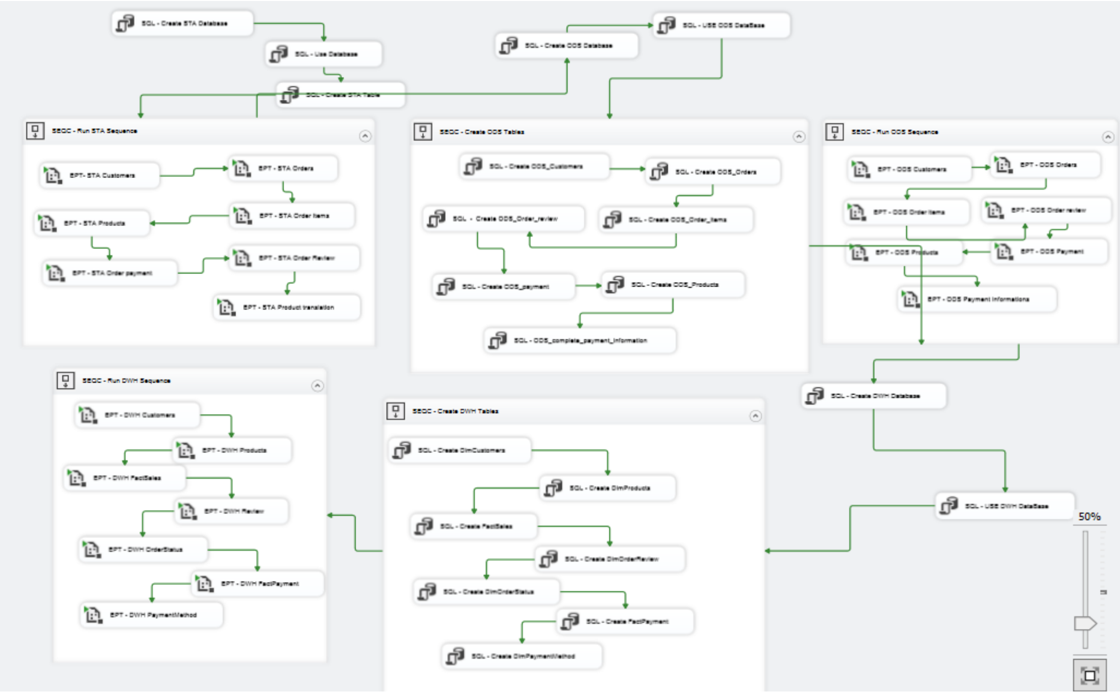


Figure 19. Project deployment

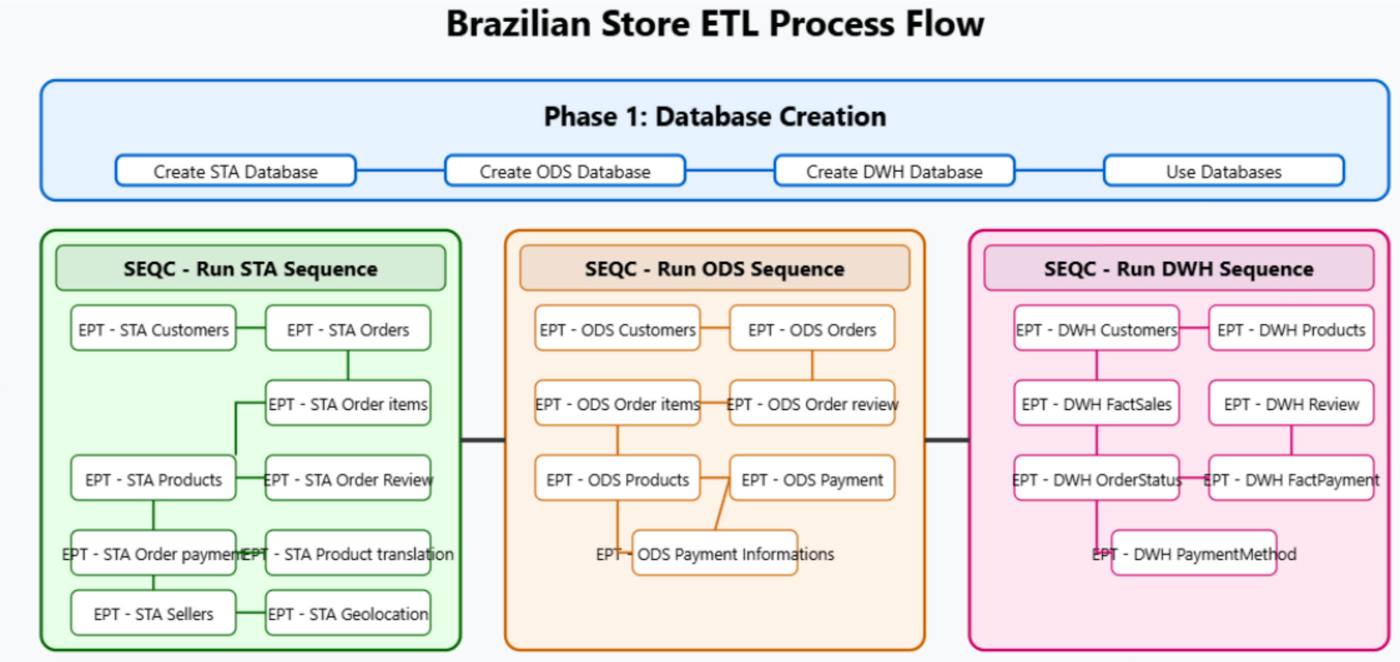
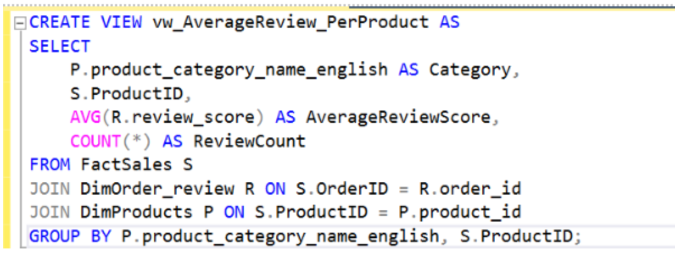


Figure 20. Project deployment

8. Use Cases

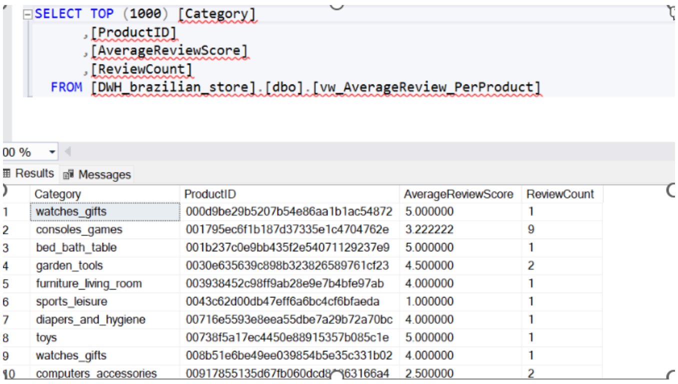
Once we have deployed the data warehouse, we can start to query it for making analysis reports. We can generate many views that can used in a BI tool to report for example:

8.1. Average customer rating per product:



```sql
CREATE VIEW vw_AverageReview_PerProduct AS
SELECT
    P.product_category_name_english AS Category,
    S.ProductID,
    AVG(R.review_score) AS AverageReviewScore,
    COUNT(*) AS ReviewCount
FROM FactSales S
JOIN DimOrder_review R ON S.OrderID = R.order_id
JOIN DimProducts P ON S.ProductID = P.product_id
GROUP BY P.product_category_name_english, S.ProductID;
```

Use Case 1



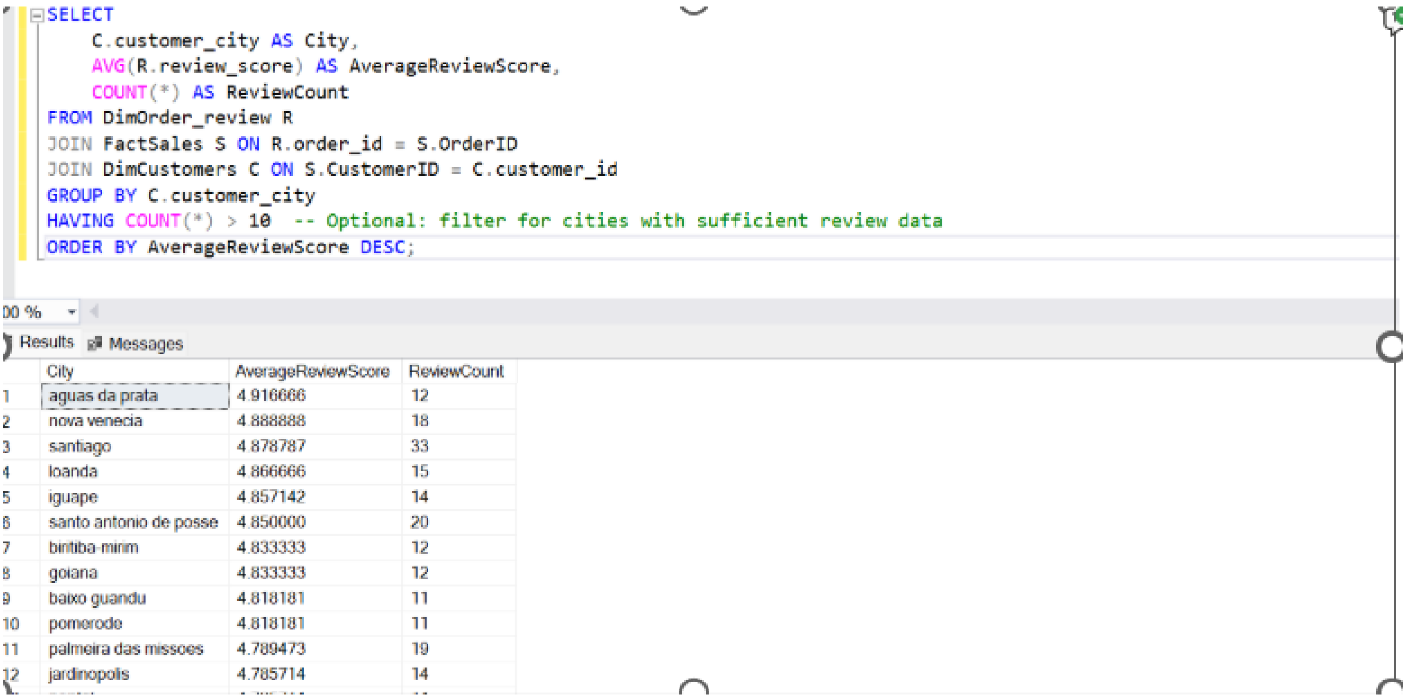
```sql
SELECT TOP (1000) [Category]
      ,[ProductID]
      ,[AverageReviewScore]
      ,[ReviewCount]
  FROM [DWH_brazilian_store].[dbo].[vw_AverageReview_PerProduct]
```

	Category	ProductID	AverageReviewScore	ReviewCount
1	watches_gifts	000d9be29b5207b54e86aa1b1ac54872	5.000000	1
2	consoles_games	001795ec6f1b187d37335e1c4704762e	3.222222	9
3	bed_bath_table	001b237c0e9bb435f2e54071129237e9	5.000000	1
4	garden_tools	0030e635639c898b323826589761cf23	4.500000	2
5	furniture_living_room	003938452c98ff9ab28e9e7b4bfe97ab	4.000000	1
6	sports_leisure	0043c62d00db47eff6a6bc4cf6bfaeda	1.000000	1
7	diapers_and_hygiene	00716e5593e8eea55dbe7a29b72a70bc	4.000000	1
8	toys	00738f5a17ec4450e88915357b085c1e	5.000000	1
9	watches_gifts	008b51e6be49ee039854b5e35c331b02	4.000000	1
10	computers_accessories	00917855135d67fb060dcd8[illegible]63166a4	2.500000	2

Use Case 2

Figure 21. Comparison of two related views

8.2. List of cities ranked by their average review score

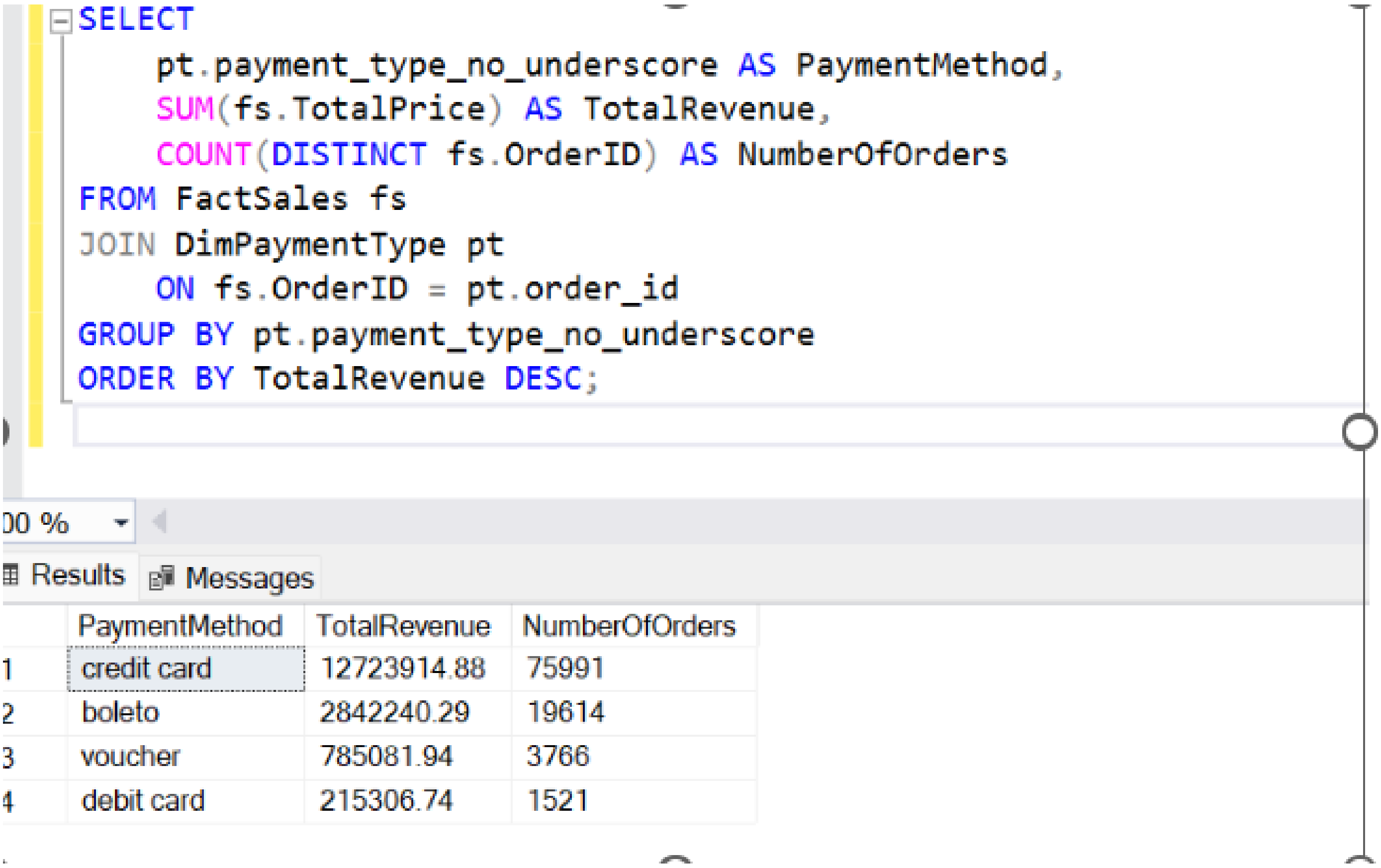


```sql
SELECT
    C.customer_city AS City,
    AVG(R.review_score) AS AverageReviewScore,
    COUNT(*) AS ReviewCount
FROM DimOrder_review R
JOIN FactSales S ON R.order_id = S.OrderID
JOIN DimCustomers C ON S.CustomerID = C.customer_id
GROUP BY C.customer_city
HAVING COUNT(*) > 10  -- Optional: filter for cities with sufficient review data
ORDER BY AverageReviewScore DESC;
```

	City	AverageReviewScore	ReviewCount
1	aguas da prata	4.916666	12
2	nova venecia	4.888888	18
3	santiago	4.878787	33
4	loanda	4.866666	15
5	iguape	4.857142	14
6	santo antonio de posse	4.850000	20
7	biritiba-mirim	4.833333	12
8	goiana	4.833333	12
9	baixo guandu	4.818181	11
10	pomerode	4.818181	11
11	palmeira das missoes	4.789473	19
12	jardinopolis	4.785714	14

Figure 22. Use Case 3

8.3. Revenue by payment method



```
SELECT
    pt.payment_type_no_underscore AS PaymentMethod,
    SUM(fs.TotalPrice) AS TotalRevenue,
    COUNT(DISTINCT fs.OrderID) AS NumberOfOrders
FROM FactSales fs
JOIN DimPaymentType pt
    ON fs.OrderID = pt.order_id
GROUP BY pt.payment_type_no_underscore
ORDER BY TotalRevenue DESC;
```

	PaymentMethod	TotalRevenue	NumberOfOrders
1	credit card	12723914.88	75991
2	boleto	2842240.29	19614
3	voucher	785081.94	3766
4	debit card	215306.74	1521

Figure 23. Use Case 4

9. Conclusion

The project shows the full automatic implementation of the ETL process of a Brazilian E-commerce platform. The tools used in this study are SSIS, SSMS and MySQL Database Engine. Data are sourced and undergone a complex transformation and table creation into a structured and analytical stay schema. This robust pipeline will generate a fully automatized data processing which enables an insightful business intelligence reporting. From our project, we were able to produce actionable insights by creating and combining several metrics, like TotalPrice which adds a significant value by enhancing financial analysis capabilities, marketing analysis and customer satisfaction and logistic performance which empowers the decision makers and comprehensive insights for strategic planning.

References

[1] B. Inmon, “The data warehouse budget”, *DM Review Magazine, January*, vol. 39, pp. 101–112, 1997.

[2] C. Shilakes and J. Tylman, “Enterprise information portals. enterprise software team”, *Enterprise Inf. Portals*, pp. 354–362, 1998.

[3] W. H. Inmon, *Building the data warehouse*. John wiley & sons, 2005.