Amazon Sales ETL & Data Warehousing Project Report

Project Group Members

Ashish **SINGH**
Nihal **SACHINDRA**
Ravichandan **KODIJUTTU**
Wanignon Justin Orpherique **DJIDONOU**

Abstract

This report outlines the design and implementation of an ETL and data warehousing project using a large-scale Amazon sales dataset. A flat structured Excel file was transformed through a robust SSIS pipeline into a star schema composed of one fact and five dimension tables. The resulting warehouse supports analytical querying, performance measurement, and business insight discovery. The ETL process includes data cleansing, validation, reject handling, transformation, dimensional modeling, and data enrichment for downstream analysis.

Introduction

The goal of this project was to develop a scalable ETL pipeline and dimensional data warehouse using a real-world retail dataset. The selected dataset emulates transaction records from Amazon sales, capturing product, shipping, and order metadata. By transforming this flat file into a normalized structure, the project enables analytical queries on trends, performance, and fulfillment efficiency. The end-to-end pipeline was developed using SQL Server Integration Services (SSIS), leveraging staging, ODS, and warehouse layers with robust reject tracking and complex business rules.

Dataset Overview

Dataset Name: Amazon Sale Report (Excel file)
Source: Custom dataset designed to simulate Amazon retail sales
Volume: Over 100,000 structured transactional records

Why This Dataset Was Chosen:
The Amazon sales dataset was selected because it mirrors real-world e-commerce activity. Although the dataset originated as a single flat file, it contains rich transactional metadata including product identifiers, shipping info, fulfillment details, and location fields. These diverse fields allowed the data to be modeled into a dimensional structure, supporting

complex analytics across multiple perspectives such as product category, sales channel, geographic location, and fulfillment strategy.

Key reasons:

- Realistic retail domain relevance
- Variety of attributes (SKU, Style, City, Fulfilment, Status, etc.)
- Enables transformation into fact and dimension models
- Provides business insight opportunities (trends, fulfillment, regions)

Data Dictionary

Column	Description
Order ID	Unique identifier for each order
Date	Order date
Status	Order status (Shipped, Cancelled, etc.)
Fulfilment	Fulfillment method (e.g., Amazon)
Sales Channel	Platform (e.g., Amazon.in)
Ship Service Level	Shipping tier (e.g., Expedited)
Style	Product style
SKU	Stock Keeping Unit
Category	Product category
Size	Size of the product
Qty	Quantity ordered
Amount	Price per unit
Ship City	Destination city
Ship State	Destination state
Ship Country	Destination country
ASIN	Amazon Standard ID Number
B2B	Indicator for business order
Courier Status	Shipment delivery status
Fulfilled By	Entity who fulfilled the order

Data Warehousing Design

The original flat file was transformed into a Star Schema consisting of one Fact and five Dimension tables.

Fact Table: FactSales

- SalesKey (PK)

- ProductKey (FK)
- TimeKey (FK)
- LocationKey (FK)
- SalesChannelKey (FK)
- OrderStatusKey (FK)
- Quantity
- Amount

Dimension Tables:

- **DimProduct**: SKU, Style, Category, Size, ProductCode (parsed), Line (parsed)
- **DimTime**: Date, Day, Month, Quarter, Year, Week
- **DimLocation**: City, State, PostalCode, Country
- **DimSalesChannel**: Sales Channel, Fulfilment, Service Level
- **DimOrderStatus**: Status, Status Category (derived)

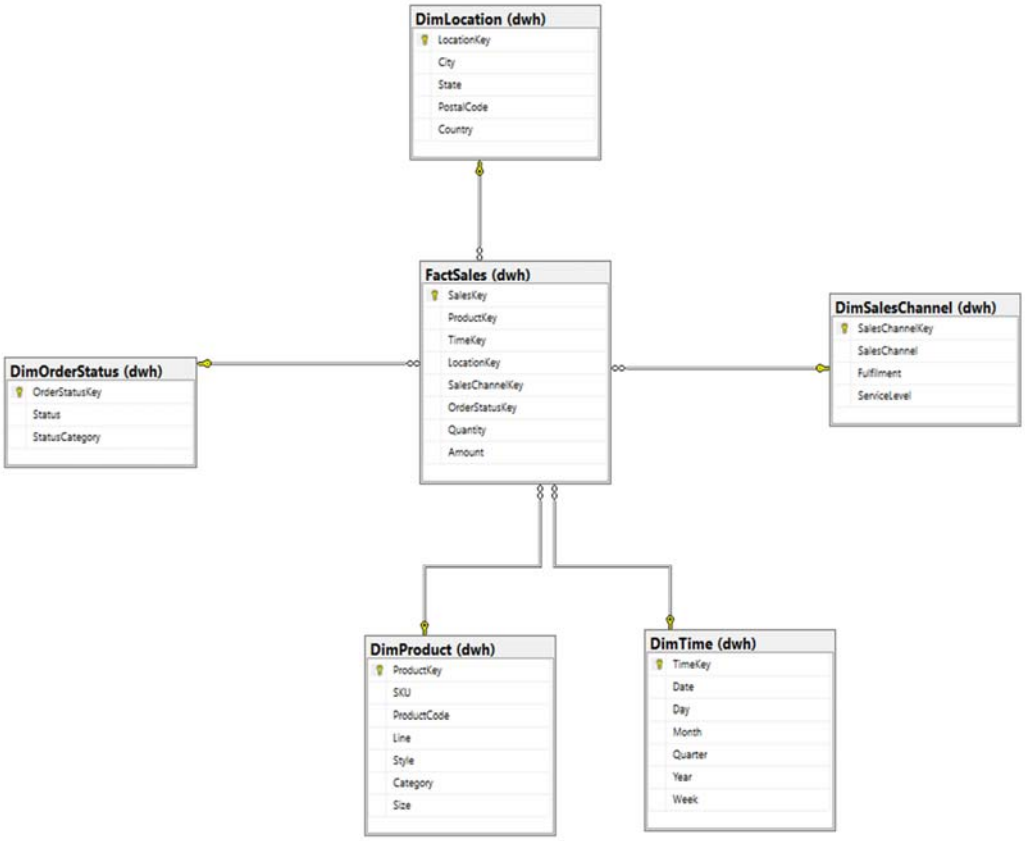


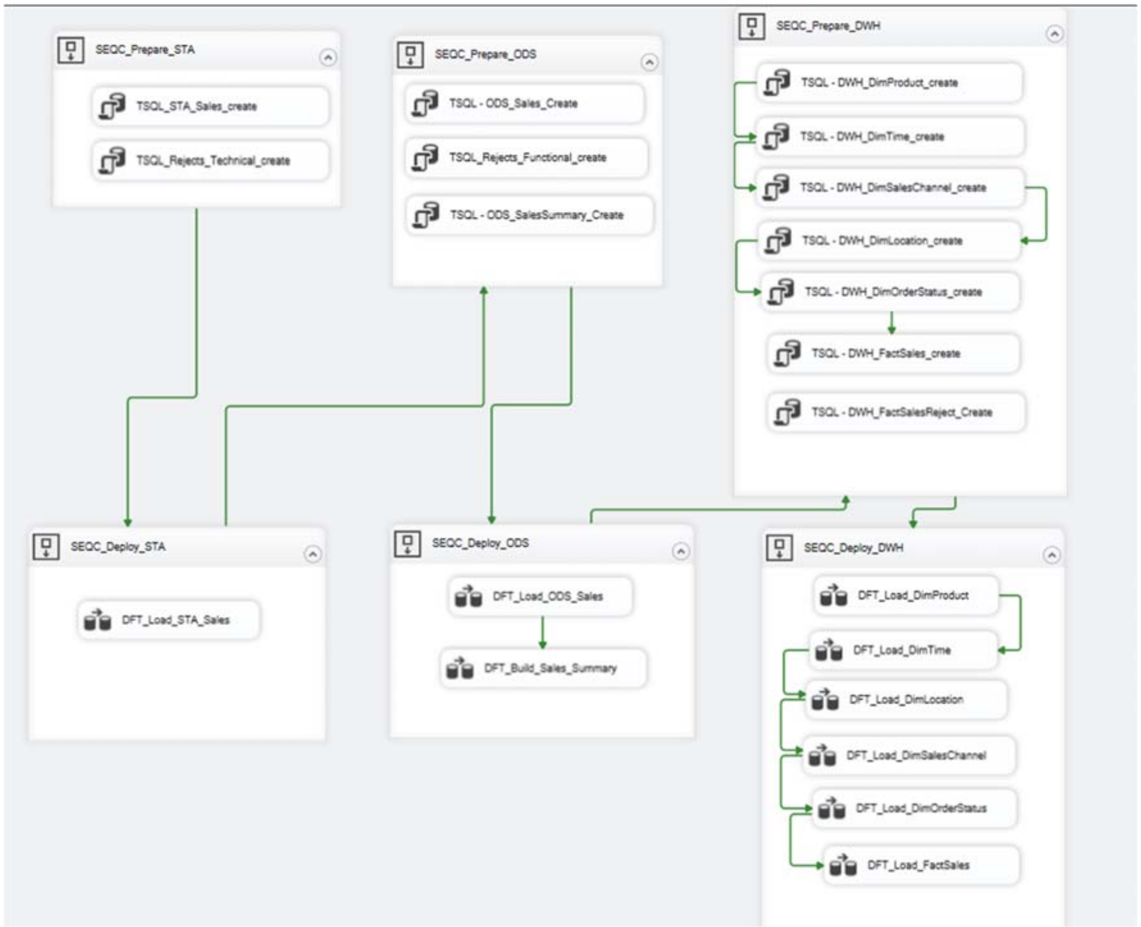
Fig. E-R Diagram modelling

ETL Process Implementation (Using SSIS)

The ETL process was implemented using **SQL Server Integration Services (SSIS)**, divided into three core layers — Staging (STA), Operational Data Store (ODS), and Data Warehouse (DWH). The process also includes reject handling for both **technical** and **functional** errors using Script Component, Derived Column, and Conditional Split.

1. High-Level ETL Control Flow

This is the master control flow containing six main Sequence Containers, each responsible for preparing or loading tables in STA, ODS, and DWH layers.

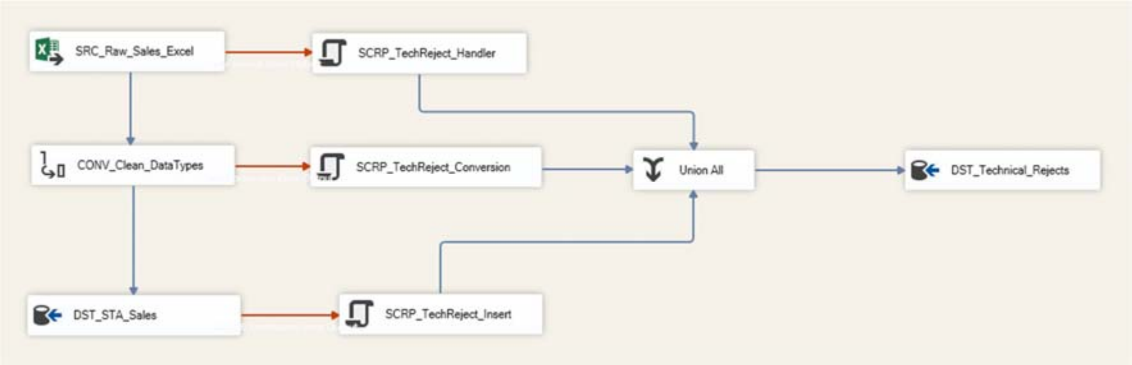


2. SEQC_Prepare_STA

- This container prepares the **staging layer tables**.
- **TSQL_STA_Sales_create**: Creates **sta.sales** to hold raw and cleaned data.
- **TSQL_Rejects_Technical_create**: Creates the **sta.technical_rejects** table to store technical issues like data type mismatches or empty fields.

3. SEQC_Deploy_STA

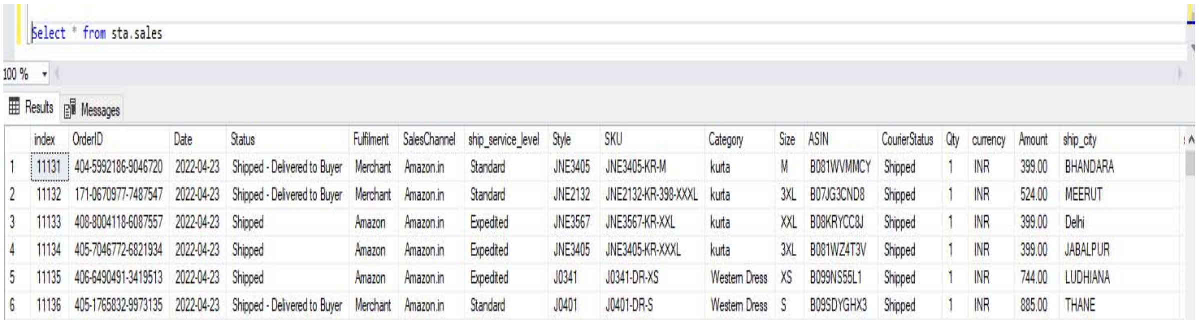
This sequence container handles the extraction, cleaning, and technical validation of raw sales data from the Excel source. It includes advanced error handling using multiple **Script Components** and a **Union All** to centralize reject logging.



Steps:

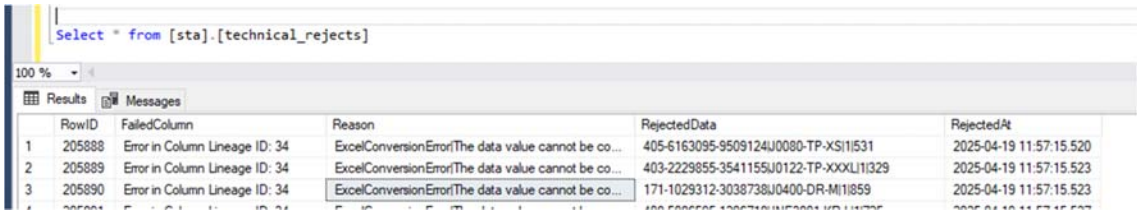
- **SRC_Raw_Sales_Excel:** Reads the Excel file containing over 100k sales records. Errors such as missing fields or bad formats are redirected to **SCRP_TechReject_Handler**.
- **CONV_Clean_DataTypes**: Converts key fields (Qty, Amount, Date) to their respective types. Errors in conversion are redirected to **SCRP_TechReject_Conversion**.
- **DST_STA_Sales**: Inserts clean and validated rows into **sta.sales**. Destination-level failures (e.g., constraint errors) are captured via **SCRP_TechReject_Insert**.
- **SCRP_TechReject_Handler / Conversion / Insert**: These Script Components extract error details, column names, and create formatted reject logs with **RejectedData**, **FailedColumn**, and **Reason**.
- **Union All**: Combines all error paths into a single stream for unified reject handling.
- **DST_Technical_Rejects**: Final destination for all technical errors. Loads into **sta.technical_rejects**.

Staging Verification Snapshots



```
Select * from sta.sales
```

	index	OrderID	Date	Status	Fulfilment	SalesChannel	ship_service_level	Style	SKU	Category	Size	ASIN	CourierStatus	Qty	currency	Amount	ship_city
1	11131	404-5992186-9046720	2022-04-23	Shipped - Delivered to Buyer	Merchant	Amazon.in	Standard	JNE3405	JNE3405-KR-M	kurta	M	B081WVMMCY	Shipped	1	INR	399.00	BHANDARA
2	11132	171-0670977-7487547	2022-04-23	Shipped - Delivered to Buyer	Merchant	Amazon.in	Standard	JNE2132	JNE2132-KR-398-XXXL	kurta	3XL	B07JG3CND8	Shipped	1	INR	524.00	MEERUT
3	11133	408-8004118-6087557	2022-04-23	Shipped	Amazon	Amazon.in	Expedited	JNE3567	JNE3567-KR-XXL	kurta	XXL	B08KRYCC8J	Shipped	1	INR	399.00	Delhi
4	11134	405-7046772-6821934	2022-04-23	Shipped	Amazon	Amazon.in	Expedited	JNE3405	JNE3405-KR-XXXL	kurta	3XL	B081WZ4T3V	Shipped	1	INR	399.00	JABALPUR
5	11135	406-6490491-3419513	2022-04-23	Shipped	Amazon	Amazon.in	Expedited	J0341	J0341-DR-XS	Western Dress	XS	B099NS55L1	Shipped	1	INR	744.00	LUDHIANA
6	11136	405-1765832-9973135	2022-04-23	Shipped - Delivered to Buyer	Merchant	Amazon.in	Standard	J0401	J0401-DR-S	Western Dress	S	B09SDYGHX3	Shipped	1	INR	885.00	THANE



```
Select * from [sta].[technical_rejects]
```

100 %

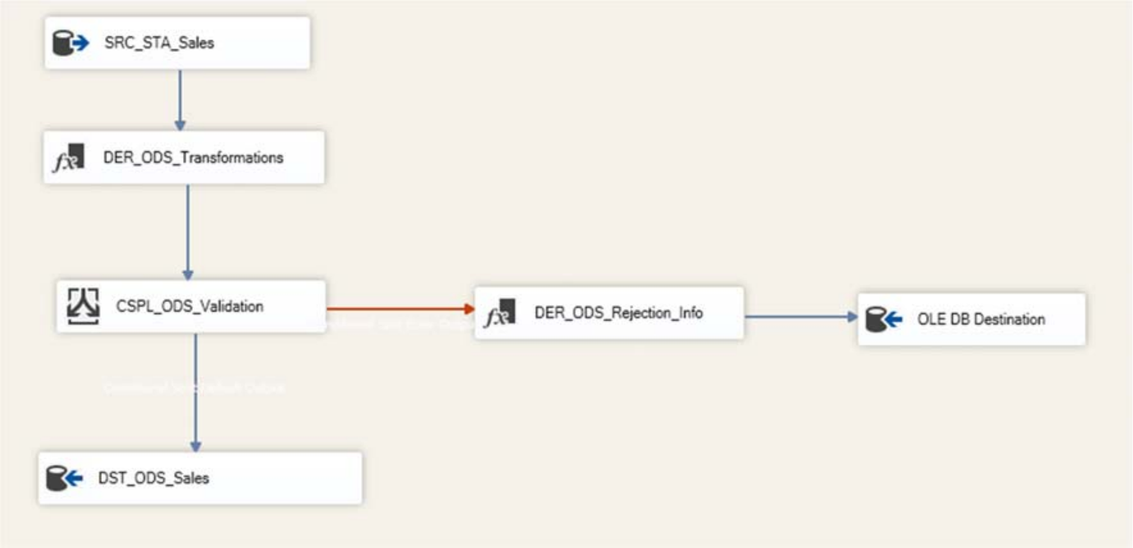
	RowID	FailedColumn	Reason	RejectedData	RejectedAt
1	205888	Error in Column Lineage ID: 34	ExcelConversionError|The data value cannot be co...	405-6163095-9509124|J0080-TP-XS|1|531	2025-04-19 11:57:15.520
2	205889	Error in Column Lineage ID: 34	ExcelConversionError|The data value cannot be co...	403-2229855-3541155|J0122-TP-XXXL|1|329	2025-04-19 11:57:15.523
3	205890	Error in Column Lineage ID: 34	ExcelConversionError|The data value cannot be co...	171-1029312-3038738|J0400-DR-M|1|859	2025-04-19 11:57:15.523

4. SEQC_Prepare_ODS

- Creates the tables required in the ODS layer.
- **TSQL_ODS_Sales_Create**: Defines the **ods.sales** table with new derived columns.
- **TSQL_Rejects_Functional_create**: Creates functional rejects table in ODS.
- **TSQL_ODS_SalesSummary_Create**: Creates the aggregation summary table for insights.

5. SEQC_Deploy_ODS

A. DFT_Load_ODS_Sales



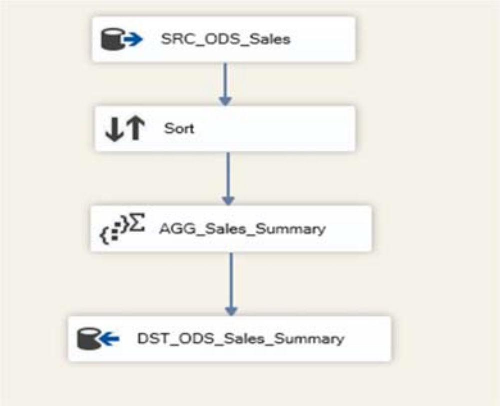
This flow performs transformation, validation, and loading of data from the staging table (**sta.sales**) to the operational store (**ods.sales**).

Steps:

- **SRC_STA_Sales**: Source component that fetches cleaned records from **sta.sales**.
- **DER_ODS_Transformations**: Adds new derived fields **Day**, **Month**, and **Year** extracted from the Date field and cleaned other fields by removing the trailing spaces.
- **CSPL_ODS_Validation**: Applies business rules to catch invalid records - null or blank OrderID, SKU, or Date

- Valid records are passed to **DST_ODS_Sales**.
- Invalid records are redirected to a rejection path.
- **DER_ODS_Rejection_Info**: Adds context to rejections with **FailedColumn**, **Reason** and **RejectedData.**
- **OLE DB Destination**: Loads functional rejects into the **ods.functional_rejects** table.

B. DFT_Build_Sales_Summary

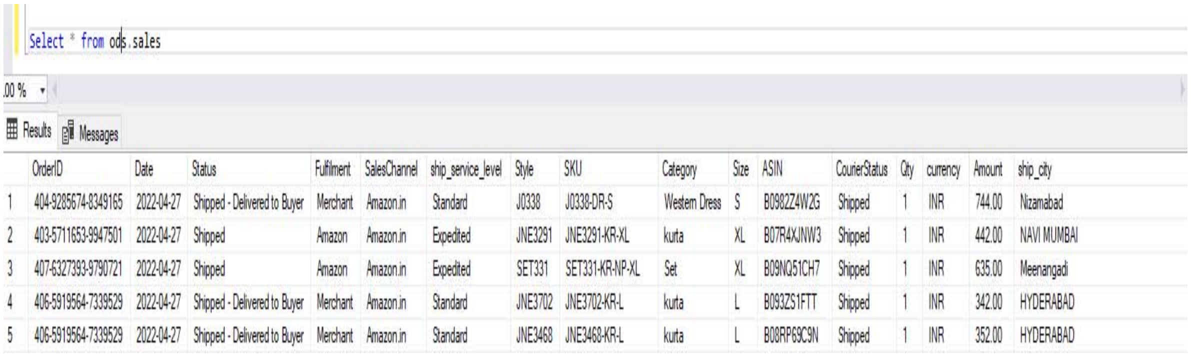


This flow builds the **ods.sales_summary** table by aggregating records across product categories and dates.

Steps:

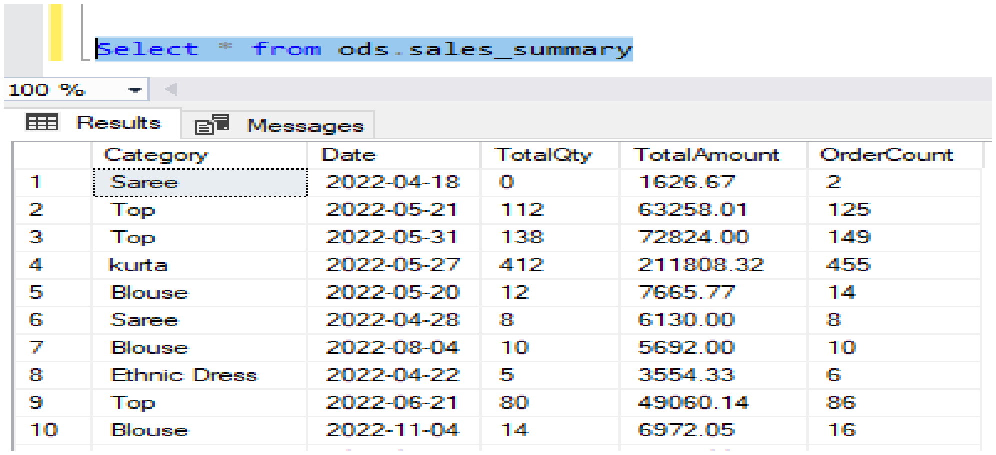
- **SRC_ODS_Sales**: Reads clean data from **ods.sales**.
- **Sort**: Ensures deterministic grouping and duplicate elimination.
- **AGG_Sales_Summary**: **Group By**: Category, Date **Aggregates**: TotalQty = SUM(Qty), TotalAmount = SUM(Amount), OrderCount = COUNT(OrderID)
- **DST_ODS_Sales_Summary**: Loads the summary metrics into **ods.sales_summary** for reporting and BI.

ODS Verification Snapshots



```
Select * from ods.sales
```

	OrderID	Date	Status	Fulfilment	SalesChannel	ship_service_level	Style	SKU	Category	Size	ASIN	CourierStatus	Qty	currency	Amount	ship_city
1	404-9285674-8349165	2022-04-27	Shipped - Delivered to Buyer	Merchant	Amazon.in	Standard	J0338	J0338-DR-S	Western Dress	S	B0982Z4W2G	Shipped	1	INR	744.00	Nizamabad
2	403-5711653-9947501	2022-04-27	Shipped	Amazon	Amazon.in	Expedited	JNE3291	JNE3291-KR-XL	kurta	XL	B07R4XJNW3	Shipped	1	INR	442.00	NAVI MUMBAI
3	407-6327393-9790721	2022-04-27	Shipped	Amazon	Amazon.in	Expedited	SET331	SET331-KR-NP-XL	Set	XL	B09NQ51CH7	Shipped	1	INR	635.00	Meenangadi
4	406-5919564-7339529	2022-04-27	Shipped - Delivered to Buyer	Merchant	Amazon.in	Standard	JNE3702	JNE3702-KR-L	kurta	L	B093ZS1FTT	Shipped	1	INR	342.00	HYDERABAD
5	406-5919564-7339529	2022-04-27	Shipped - Delivered to Buyer	Merchant	Amazon.in	Standard	JNE3468	JNE3468-KR-L	kurta	L	B08RP69C9N	Shipped	1	INR	352.00	HYDERABAD



```
Select * from ods.sales_summary
```

	Category	Date	TotalQty	TotalAmount	OrderCount
1	Saree	2022-04-18	0	1626.67	2
2	Top	2022-05-21	112	63258.01	125
3	Top	2022-05-31	138	72824.00	149
4	kurta	2022-05-27	412	211808.32	455
5	Blouse	2022-05-20	12	7665.77	14
6	Saree	2022-04-28	8	6130.00	8
7	Blouse	2022-08-04	10	5692.00	10
8	Ethnic Dress	2022-04-22	5	3554.33	6
9	Top	2022-06-21	80	49060.14	86
10	Blouse	2022-11-04	14	6972.05	16

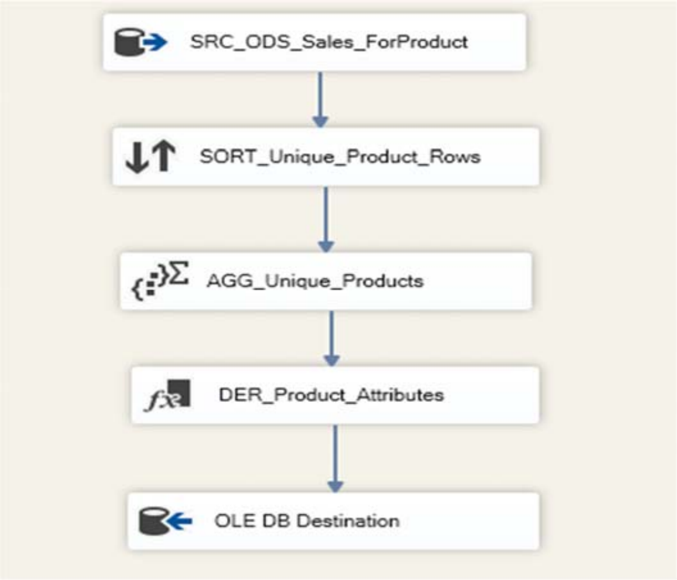
6. SEQC_Prepare_DWH

- Executes SQL scripts to create all dimension and fact tables: **DimProduct, DimTime, DimLocation, DimSalesChannel, DimOrderStatus, FactSales,** and **FactSalesReject.**
- Tables are created with identity surrogate keys and appropriate foreign key relationships.

7. SEQC_Deploy_DWH

This control flow container executes all data flow tasks responsible for populating the star schema in the data warehouse layer. These flows transform, enrich, and load data into 5 dimension tables and 1 fact table, leveraging lookups and derived logic.

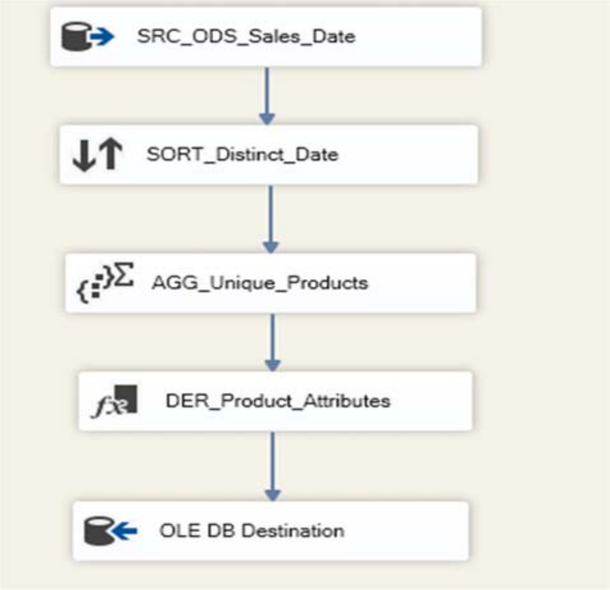
A. DFT_Load_DimProduct



- **Source:** Reads from **ods.sales** table

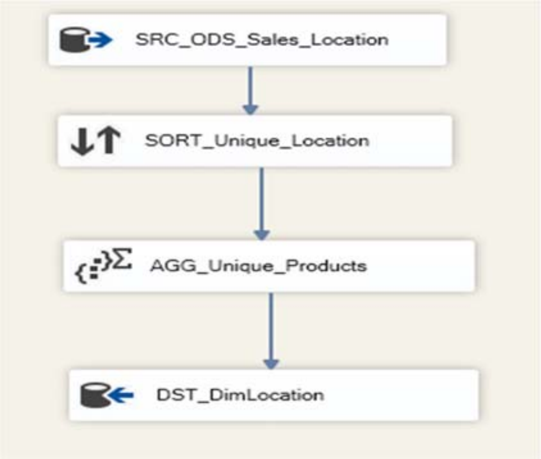
- **Sort:** Ensures unique product combinations (SKU, Style, Category, Size)
- **Aggregate:** Removes duplicates
- **Derived Column:** Parses SKU into Product Code and Line using substring logic
- **Destination:** Loads into **dwh.DimProduct**

B. DFT_Load_DimTime



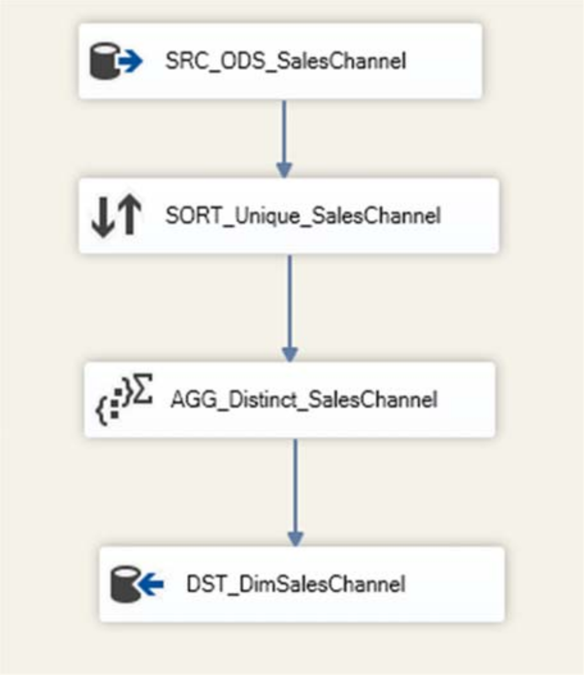
- **Source:** Pulls distinct Date from **ods.sales.**
- **Sort + Aggregate:** Ensures unique dates
- **Derived Column:** Extracts Quarter, Week
- **Destination:** Loads into **dwh.DimTime**

C. DFT_Load_DimLocation



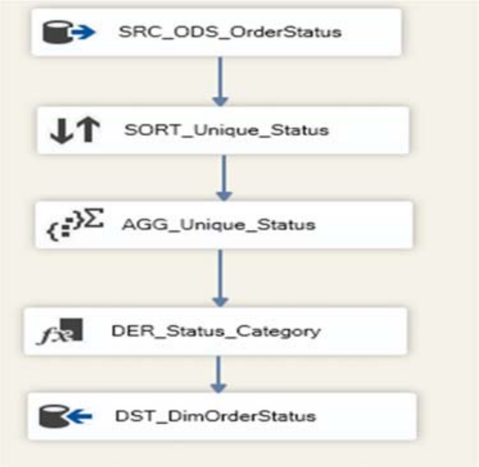
- **Source:** Reads location fields from **ods.sales** (City, State, PostalCode, Country)
- **Sort + Aggregate:** De-duplicates location entries
- **Destination:** Loads into dwh.DimLocation

D. DFT_Load_DimSalesChannel



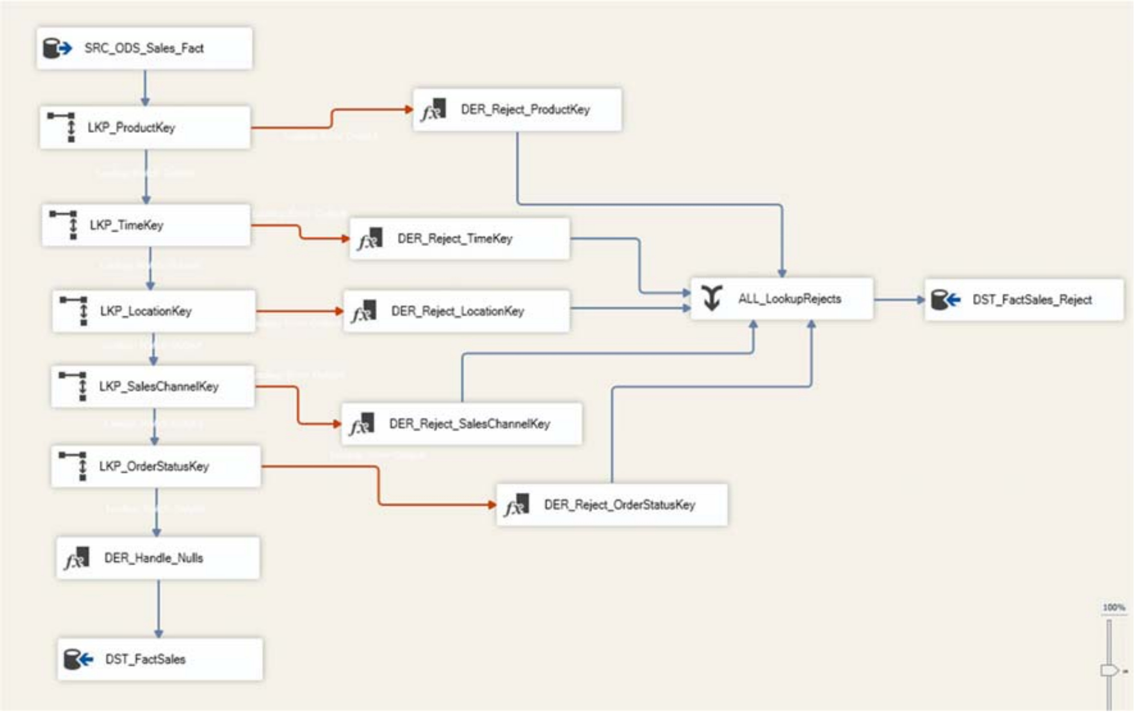
- **Source:** Reads **SalesChannel**, **Fulfilment**, **ServiceLevel** from **ods.sales**
- **Sort + Aggregate:** Ensures distinct combinations
- **Destination:** Loads into **dwh.DimSalesChannel**

E. DFT_Load_DimOrderStatus



- **Source:** Reads raw Status from **ods.sales.**
- **Sort + Aggregate:** Ensures unique statuses.
- **Derived Column:** Adds StatusCategory logic using expression like Shipped is Completed, cancelled by any reason is cancelled.
- **Destination:** Loads into dwh.DimOrderStatus

F. DFT_Load_FactSales

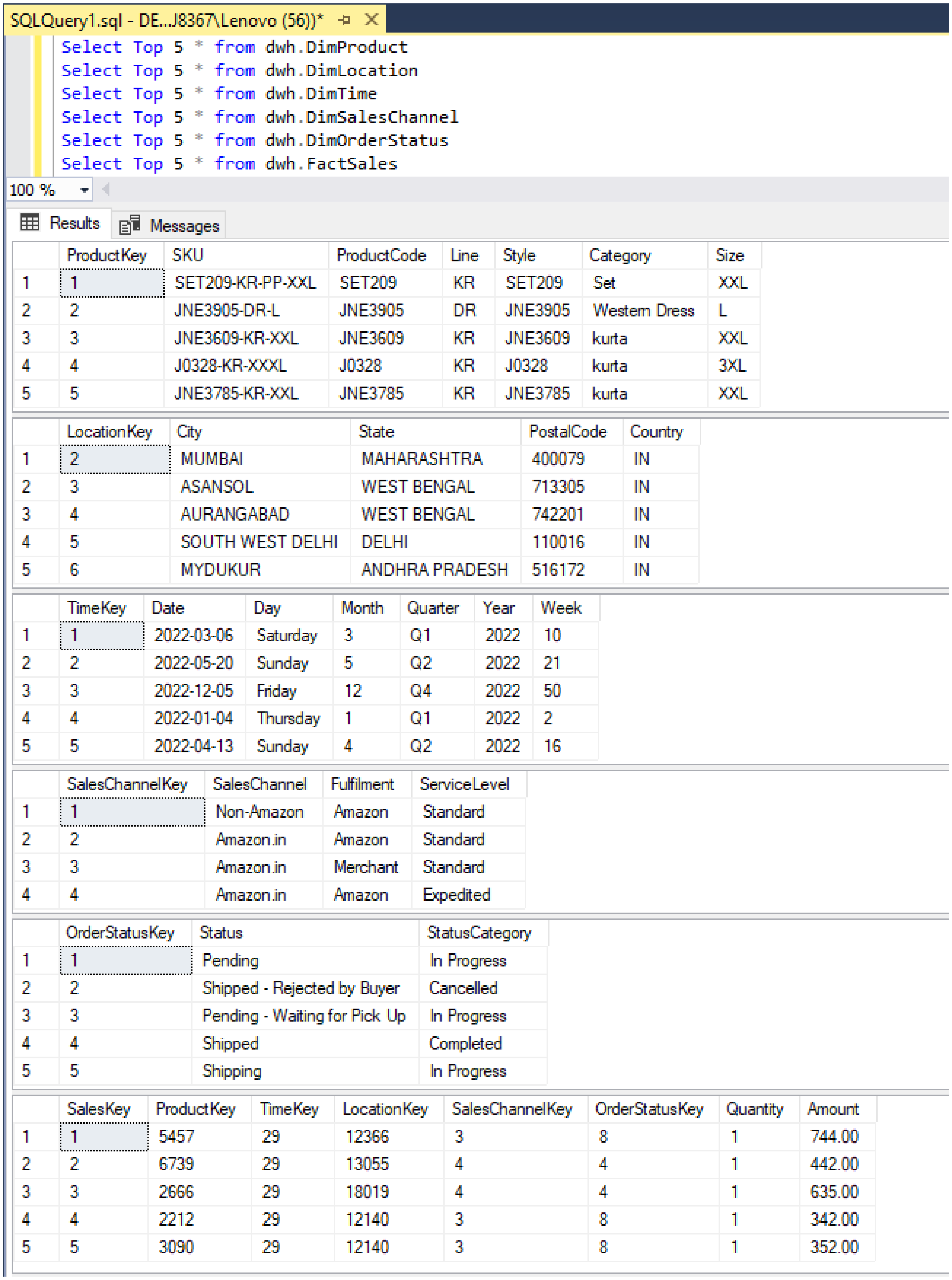


- **Source:** Reads enriched sales from **ods.sales.**
- **5 LOOKUPs:** Fetch surrogate keys from:
  DimProduct
  DimTime
  DimLocation
  DimSalesChannel
  DimOrderStatus
- **Derived Column:** Handles nulls for all the fields
- **Destination:** Loads matched records into **FactSales**

Lookup Reject Handling:

- All LOOKUPs use No Match Output
- Each branch goes to a **Derived Column** to tag: **RejectSource**, **RejectedData**, **Reason**
- **Union All** merges all unmatched rows
- Rejected rows are inserted into **dwh.FactSalesReject**

DWH Verification Snapshots



```
Select Top 5 * from dwh.DimProduct
Select Top 5 * from dwh.DimLocation
Select Top 5 * from dwh.DimTime
Select Top 5 * from dwh.DimSalesChannel
Select Top 5 * from dwh.DimOrderStatus
Select Top 5 * from dwh.FactSales
```

	ProductKey	SKU	ProductCode	Line	Style	Category	Size
1	1	SET209-KR-PP-XXL	SET209	KR	SET209	Set	XXL
2	2	JNE3905-DR-L	JNE3905	DR	JNE3905	Western Dress	L
3	3	JNE3609-KR-XXL	JNE3609	KR	JNE3609	kurta	XXL
4	4	J0328-KR-XXXL	J0328	KR	J0328	kurta	3XL
5	5	JNE3785-KR-XXL	JNE3785	KR	JNE3785	kurta	XXL

	LocationKey	City	State	PostalCode	Country
1	2	MUMBAI	MAHARASHTRA	400079	IN
2	3	ASANSOL	WEST BENGAL	713305	IN
3	4	AURANGABAD	WEST BENGAL	742201	IN
4	5	SOUTH WEST DELHI	DELHI	110016	IN
5	6	MYDUKUR	ANDHRA PRADESH	516172	IN

	TimeKey	Date	Day	Month	Quarter	Year	Week
1	1	2022-03-06	Saturday	3	Q1	2022	10
2	2	2022-05-20	Sunday	5	Q2	2022	21
3	3	2022-12-05	Friday	12	Q4	2022	50
4	4	2022-01-04	Thursday	1	Q1	2022	2
5	5	2022-04-13	Sunday	4	Q2	2022	16

	SalesChannelKey	SalesChannel	Fulfilment	ServiceLevel
1	1	Non-Amazon	Amazon	Standard
2	2	Amazon.in	Amazon	Standard
3	3	Amazon.in	Merchant	Standard
4	4	Amazon.in	Amazon	Expedited

	OrderStatusKey	Status	StatusCategory
1	1	Pending	In Progress
2	2	Shipped - Rejected by Buyer	Cancelled
3	3	Pending - Waiting for Pick Up	In Progress
4	4	Shipped	Completed
5	5	Shipping	In Progress

	SalesKey	ProductKey	TimeKey	LocationKey	SalesChannelKey	OrderStatusKey	Quantity	Amount
1	1	5457	29	12366	3	8	1	744.00
2	2	6739	29	13055	4	4	1	442.00
3	3	2666	29	18019	4	4	1	635.00
4	4	2212	29	12140	3	8	1	342.00
5	5	3090	29	12140	3	8	1	352.00

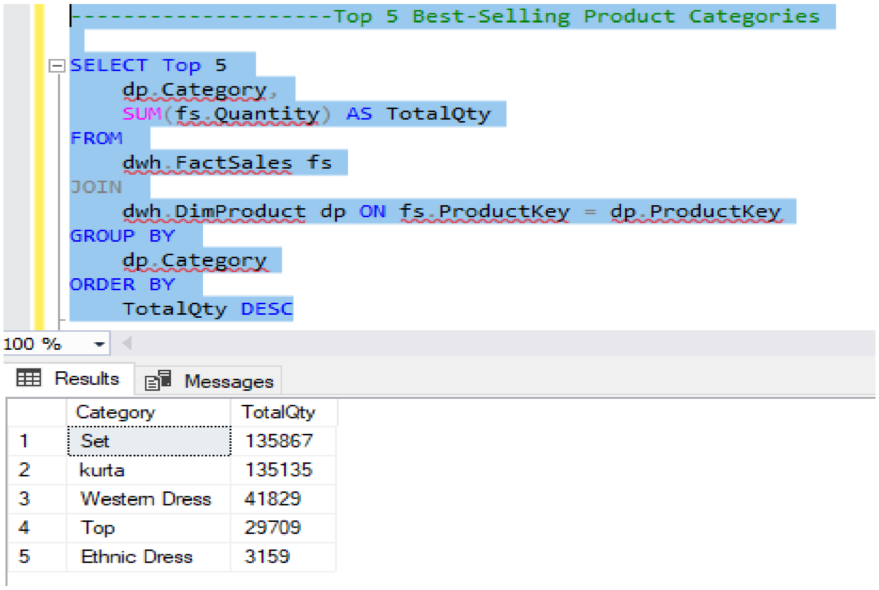
Summary

This deployment flow completes the dimensional warehouse. By enriching and validating each dimension, and performing referential integrity checks via Lookups, this stage ensures only clean, linkable rows enter the **FactSales** table while preserving all rejects.

Results and Business Insights

Insight 1: Top 5 Best-Selling Product Categories

These are the most frequently sold product categories based on total quantity.



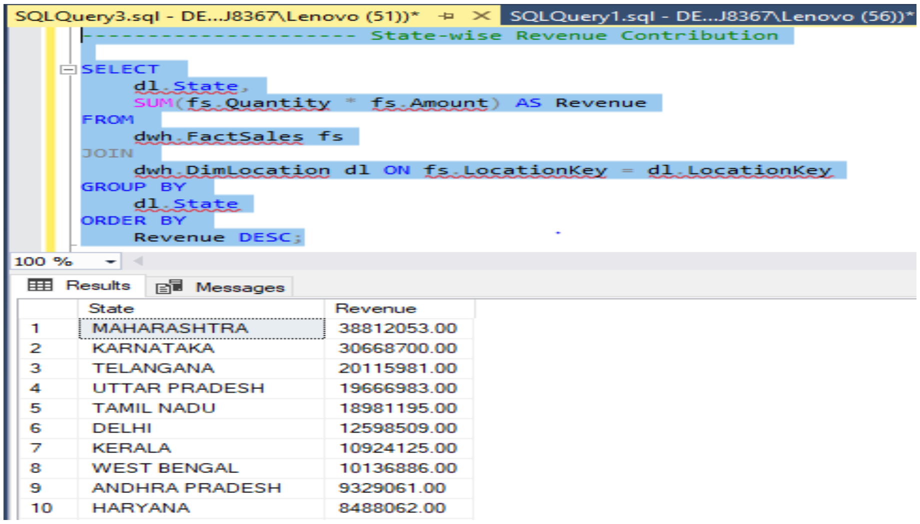
```sql
-------------------Top 5 Best-Selling Product Categories

SELECT Top 5
    dp.Category,
    SUM(fs.Quantity) AS TotalQty
FROM
    dwh.FactSales fs
JOIN
    dwh.DimProduct dp ON fs.ProductKey = dp.ProductKey
GROUP BY
    dp.Category
ORDER BY
    TotalQty DESC
```

	Category	TotalQty
1	Set	135867
2	kurta	135135
3	Western Dress	41829
4	Top	29709
5	Ethnic Dress	3159

Insight 2: State-wise Revenue Contribution

Top-performing states based on total revenue (Amount).



SQLQuery3.sql - DE...J8367\Lenovo (51))*
SQLQuery1.sql - DE...J8367\Lenovo (56))*

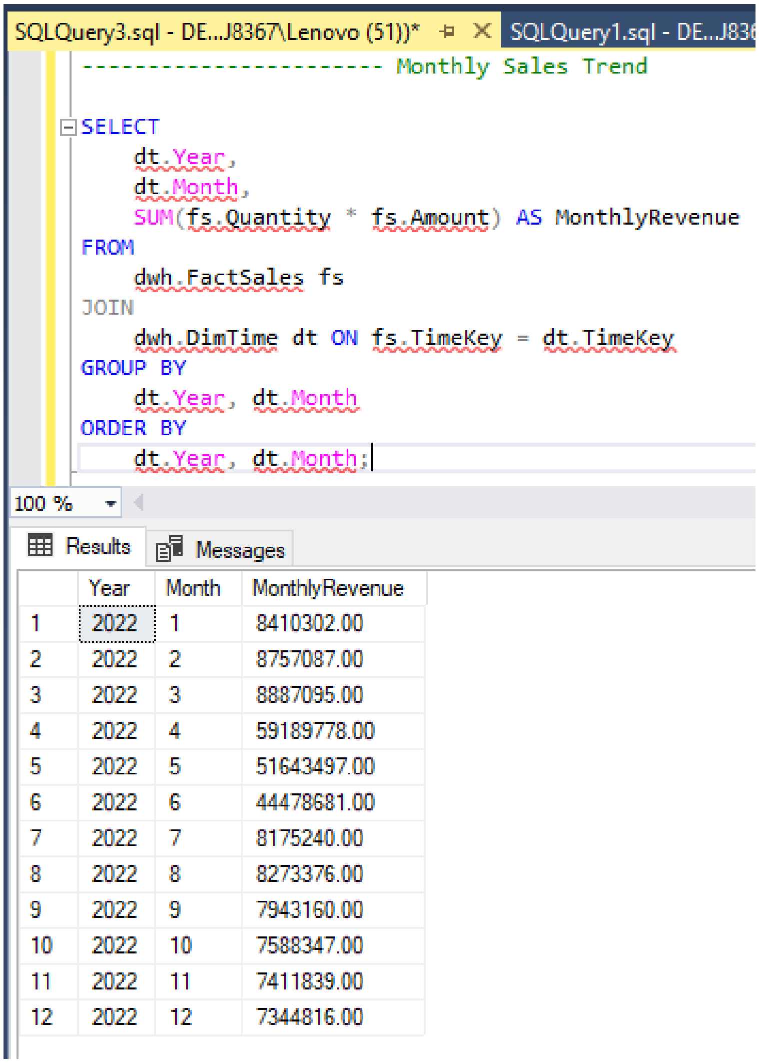
```sql
---------------------- State-wise Revenue Contribution

SELECT
    dl.State,
    SUM(fs.Quantity * fs.Amount) AS Revenue
FROM
    dwh.FactSales fs
JOIN
    dwh.DimLocation dl ON fs.LocationKey = dl.LocationKey
GROUP BY
    dl.State
ORDER BY
    Revenue DESC;
```

	State	Revenue
1	MAHARASHTRA	38812053.00
2	KARNATAKA	30668700.00
3	TELANGANA	20115981.00
4	UTTAR PRADESH	19666983.00
5	TAMIL NADU	18981195.00
6	DELHI	12598509.00
7	KERALA	10924125.00
8	WEST BENGAL	10136886.00
9	ANDHRA PRADESH	9329061.00
10	HARYANA	8488062.00

Insight 3: Monthly Sales Trend

Sales growth over time — total revenue per month.


SQLQuery3.sql - DE...J8367\Lenovo (51))*
SQLQuery1.sql - DE...J836
100 %
Results
Messages

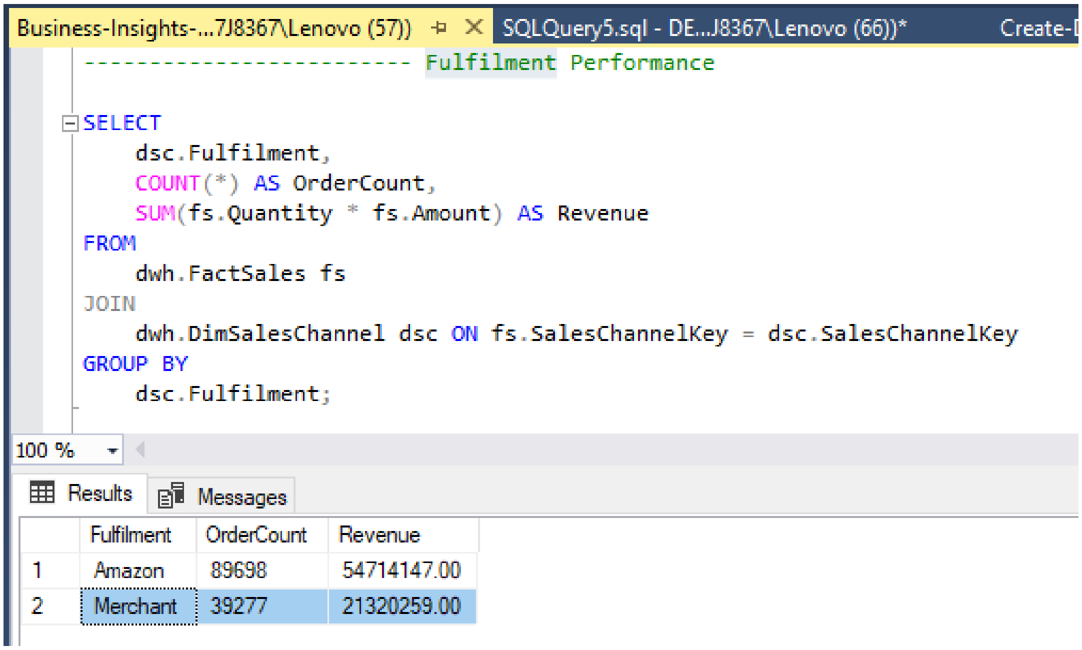
```
----------------------- Monthly Sales Trend

SELECT
    dt.Year,
    dt.Month,
    SUM(fs.Quantity * fs.Amount) AS MonthlyRevenue
FROM
    dwh.FactSales fs
JOIN
    dwh.DimTime dt ON fs.TimeKey = dt.TimeKey
GROUP BY
    dt.Year, dt.Month
ORDER BY
    dt.Year, dt.Month;
```

	Year	Month	MonthlyRevenue
1	2022	1	8410302.00
2	2022	2	8757087.00
3	2022	3	8887095.00
4	2022	4	59189778.00
5	2022	5	51643497.00
6	2022	6	44478681.00
7	2022	7	8175240.00
8	2022	8	8273376.00
9	2022	9	7943160.00
10	2022	10	7588347.00
11	2022	11	7411839.00
12	2022	12	7344816.00

Insight 4: Fulfilment Performance

Compare Amazon vs 3rd-party fulfilled orders based on sales volume.


Business-Insights-...7J8367\Lenovo (57))
SQLQuery5.sql - DE...J8367\Lenovo (66))*
Create-[
100 %
Results
Messages

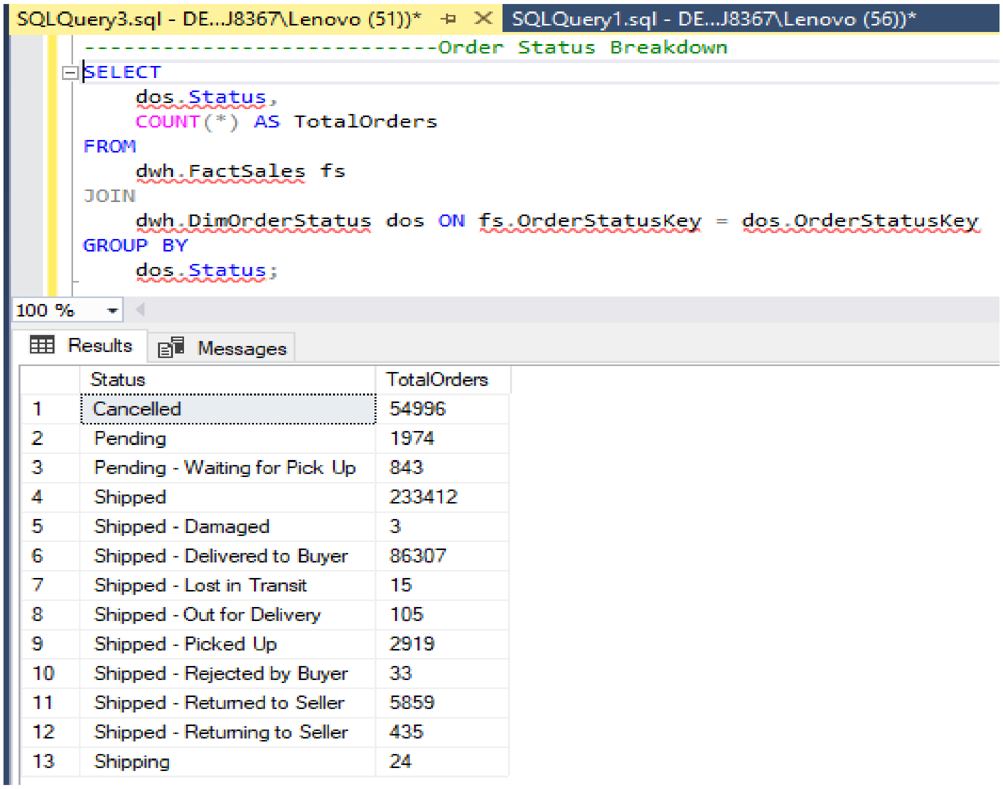
```
------------------------- Fulfilment Performance

SELECT
    dsc.Fulfilment,
    COUNT(*) AS OrderCount,
    SUM(fs.Quantity * fs.Amount) AS Revenue
FROM
    dwh.FactSales fs
JOIN
    dwh.DimSalesChannel dsc ON fs.SalesChannelKey = dsc.SalesChannelKey
GROUP BY
    dsc.Fulfilment;
```

	Fulfilment	OrderCount	Revenue
1	Amazon	89698	54714147.00
2	Merchant	39277	21320259.00

Insight 5: Order Status Breakdown

Which types of orders are getting cancelled, delivered, or returned?


SQLQuery3.sql - DE...J8367\Lenovo (51))*
SQLQuery1.sql - DE...J8367\Lenovo (56))*
100 %
Results
Messages

```
---------------------------Order Status Breakdown
SELECT
    dos.Status,
    COUNT(*) AS TotalOrders
FROM
    dwh.FactSales fs
JOIN
    dwh.DimOrderStatus dos ON fs.OrderStatusKey = dos.OrderStatusKey
GROUP BY
    dos.Status;
```

	Status	TotalOrders
1	Cancelled	54996
2	Pending	1974
3	Pending - Waiting for Pick Up	843
4	Shipped	233412
5	Shipped - Damaged	3
6	Shipped - Delivered to Buyer	86307
7	Shipped - Lost in Transit	15
8	Shipped - Out for Delivery	105
9	Shipped - Picked Up	2919
10	Shipped - Rejected by Buyer	33
11	Shipped - Returned to Seller	5859
12	Shipped - Returning to Seller	435
13	Shipping	24

Benefits and Conclusions

This project demonstrates an end-to-end ETL and Data Warehouse solution using industry standards. A single raw file was converted into a scalable dimensional model via SSIS.

Benefits:

- Modular and reusable SSIS architecture
- Robust error handling via technical and functional reject tracking
- Lookup logic ensures referential integrity in warehouse
- Star schema supports BI tools and KPI analysis

The system is extensible for use in:

- Power BI dashboards
- ML prediction pipelines (e.g., churn, returns)
- Executive summary reports